

Computer Science & Information Technology

175

Data Science and Machine Learning

David C. Wyld,
Dhinaharan Nagamalai (Eds)

Computer Science & Information Technology

- 9th International Conference on Artificial Intelligence & Applications (ARIA 2022)
- 8th International Conference on Signal Processing and Pattern Recognition (SIPR 2022)
- 8th International Conference on Software Engineering and Applications (SOFEA 2022)
- 9th International Conference on Computer Science and Engineering (CSEN 2022)
- 3rd International Conference on Data Science and Machine Learning (DSML 2022)
- 11th International Conference on Natural Language Processing (NLP 2022)
- 3rd International Conference on Education and Integrating Technology (EDTECH 2022)
- 8th International Conference of Networks, Communications, Wireless and Mobile Computing (NCWC 2022)

Published By



AIRCC Publishing Corporation

Volume Editors

David C. Wyld,
Southeastern Louisiana University, USA
E-mail: David.Wyld@selu.edu

Dhinaharan Nagamalai (Eds),
Wireilla Net Solutions, Australia
E-mail: dhinthia@yahoo.com

ISSN: 2231 - 5403
ISBN: 978-1-925953-75-6
DOI: 10.5121/csit.2022.121501 - 10.5121/csit.2022.121524

This work is subject to copyright. All rights are reserved, whether whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the International Copyright Law and permission for use must always be obtained from Academy & Industry Research Collaboration Center. Violations are liable to prosecution under the International Copyright Law.

Typesetting: Camera-ready by author, data conversion by NnN Net Solutions Private Ltd., Chennai, India

Preface

9th International Conference on Artificial Intelligence & Applications (ARIA 2022), 8th International Conference on Signal Processing and Pattern Recognition (SIPR 2022), 8th International Conference on Software Engineering and Applications (SOFEA 2022), 9th International Conference on Computer Science and Engineering (CSEN 2022), 3rd International Conference on Data Science and Machine Learning (DSML 2022), 11th International Conference on Natural Language Processing (NLP 2022), 3rd International Conference on Education and Integrating Technology (EDTECH 2022), 8th International Conference of Networks, Communications, Wireless and Mobile Computing (NCWC 2022) was collocated with 3rd International Conference on Data Science and Machine Learning (DSML 2022). The conferences attracted many local and international delegates, presenting a balanced mixture of intellect from the East and from the West.

The goal of this conference series is to bring together researchers and practitioners from academia and industry to focus on understanding computer science and information technology and to establish new collaborations in these areas. Authors are invited to contribute to the conference by submitting articles that illustrate research results, projects, survey work and industrial experiences describing significant advances in all areas of computer science and information technology.

The ARIA 2022, SIPR 2022, SOFEA 2022, CSEN 2022, DSML 2022, NLP 2022, EDTECH 2022 and NCWC 2022. Committees rigorously invited submissions for many months from researchers, scientists, engineers, students and practitioners related to the relevant themes and tracks of the workshop. This effort guaranteed submissions from an unparalleled number of internationally recognized top-level researchers. All the submissions underwent a strenuous peer review process which comprised expert reviewers. These reviewers were selected from a talented pool of Technical Committee members and external reviewers on the basis of their expertise. The papers were then reviewed based on their contributions, technical content, originality and clarity. The entire process, which includes the submission, review and acceptance processes, was done electronically.

In closing, ARIA 2022, SIPR 2022, SOFEA 2022, CSEN 2022, DSML 2022, NLP 2022, EDTECH 2022 and NCWC 2022 brought together researchers, scientists, engineers, students and practitioners to exchange and share their experiences, new ideas and research results in all aspects of the main workshop themes and tracks, and to discuss the practical challenges encountered and the solutions adopted. The book is organized as a collection of papers from the ARIA 2022, SIPR 2022, SOFEA 2022, CSEN 2022, DSML 2022, NLP 2022, EDTECH 2022 and NCWC 2022.

We would like to thank the General and Program Chairs, organization staff, the members of the Technical Program Committees and external reviewers for their excellent and tireless work. We sincerely wish that all attendees benefited scientifically from the conference and wish them every success in their research. It is the humble wish of the conference organizers that the professional dialogue among the researchers, scientists, engineers, students and educators continues beyond the event and that the friendships and collaborations forged will linger and prosper for many years to come.

David C. Wyld,
Dhinaharan Nagamalai (Eds)

General Chair

David C. Wyld,
Dhinaharan Nagamalai (Eds)

Organization

Southeastern Louisiana University, USA
Wireilla Net Solutions, Australia

Program Committee Members

Abdel-Badeeh M. Salem,
Abdelhadi Assir,
Abdellatif I. Moustafa,
Abderrahim Siam,
Abderrahmane EZ-Zahout,
Abdullah,
Abhishek Shukla,
Addisson Salazar,
Adrian Olaru,
Adriana Carla Damasceno,
Ahmad A. Saifan,
Ahmed Kadhim Hussein,
Ajit Singh,
Akhil Gupta,
Ali Abdrhman Mohammed Ukasha,
Ali El-Zaart,
Aliasghar Tarkhan,
Alireza Valipour Baboli,
Allel Hadjali,
Altay Guvenir,
Amal Azeroual,
Amando P. Singun Jr,
Amar Ramdane Cherif,
Amari Houada,
Amir H Gandomi,
Amit Agarwal,
Amizah Malip,
Anas Alsobeh,
Angelina Tzacheva,
Anita Dixit,
Anouar Abtoy,
Antinisca Di Marco,
António Abreu,
Aridj Mohamed,
Aridj Mohamed,
Arti Jain,
Asif Khan,
Assem abdel hamied moussa,
Assem Moussa,
Assia Djenouhat,
Atanu Nag,
Atul Garg,
B Nandini,
B.K.Tripathy,
Ain Shams University, Egypt
Hassan 1st University, Morocco
Umm AL-Qura University, Saudi Arabia
University of Khenchela, Algeria
Mohammed V University, Morocco
Chandigarh University, India
R D Engineering College, India
Universitat Politècnica de València, Spain
University Politehnica of Bucharest, Romania
Universidade Federal da Paraíba (UFPB), Brazil
Yarmouk University, Jordan
University of Babylon, Iraq
Patna University, India
Lovely Professional University, India
Sebha University, Libya
Beirut Arab University, Lebanon
University of Washington, USA
University Technical and Vocational, Iran
LIAS/ENSMA, France
Bilkent University, Turkey
Mohammed V University, Morocco
University of Technology and Applied Sciences, Oman
University Paris Saclay, France
Networking & Telecom Engineering, Tunisia
University of Technology, Australia
Wells Fargo, India
University of Malaya, Malaysia
Yarmouk University, Jordan
University of North Carolina, USA
SDM College of Engineering and Technology, India
Abdelmalek Essaadi University, Morocco
University of L'Aquila, Italy
ISEL – Polytechnic Institute of Lisbon, Portugal
Hassiba Benbouali University, Algeria
Mohamed Hassiba Benbouali University Chlef, Algeria
Jaypee Institute of Information Technology (JIIT), India
Integral University, India
Chief Eng Egyptair, Egypt
GGA, Egypt
University of Algiers 3, Dely Brahim, Algeria
IFTM University, India
Chitkara University, India
Telangana University, India
Vellore Institute of Technology, India

Bashir Ido,	Arsi University, Ethiopia
Benyamin Ahmadnia,	Occidental College, USA
Beshair Alsiddiq,	Riyad Bank, Saudi Arabia
Bilal Alatas,	Firat University, Turkey
Bouchra Marzak,	Hassan II University, Morocco
Brahim Lejdel,	University of El-Oued, Algeria
Charalampos Karagiannidis,	University of Thessaly, Greece
Cheng Siong Chin,	Newcastle University, Singapore
Christian Mancas,	Ovidius University, Romania
Chuan-Ming Liu,	National Taipei University of Technology, Taiwan
Dallel Sarnou,	Abdelhamid Ibn Badis University , Algeria
Daniel Hunyadi,	"Lucian Blaga" University of Sibiu, Romania
Daniela Cristina,	University Politehnica of Bucharest, Romania
Dario Ferreira,	University of Beira Interior, Portugal
Dariusz Jacek Jakobczak,	Technical University of Koszalin, Poland
Debjani Chakraborty,	Indian Institute of Technology Kharagpur, India
Deepak Mane,	Tata Consulting Services, Australia
Dereje Regassa,	Seoul National University, South Korea
Dhirendra Pal Singh,	University of Lucknow, India
Dhruv Sheth,	Embedded ML Research at EdgeImpulse. Inc, India
Dimitris Kanellopoulos,	University of Patras, Greece
Djiguimkoudre Nathalie,	Universite Joseph KI-ZERBO, Burkina Faso
Douglas Chai,	Edith Cowan University, Australia
El Habib Nfaoui,	Sidi Mohamed Ben Abdellah University, Morocco
El Kabtane Hamada,	Cadi Ayyad University, Morocco
Elaheh Yadegaridehkordi,	The National University of Malaysia, Malaysia
Elzbieta Macioszek,	Silesian University of Technology, Poland
Eng Islam Atef,	Alexandria University, Egypt
F. M. Javed Mehedi Shamrat,	Daffodil International University, Bangladesh
Fangyuan Li,	Zhengzhou University, China
Faouzia Benabbou,	University Hassan II of Casablanca, Morocco
Faycal Bensaali,	Qatar University, Qatar
Felix J. Garcia Clemente,	University of Murcia, Spain
Fernando Zacarias Flores,	Universidad Autonoma de Puebla, Mexico
Fitri Utaminigrum,	Brawijaya University, Indonesia
Francesco Zirilli,	Sapienza Universita Roma , Italy
Furkan Rabee,	University of Kufa, Iraq
Fzlollah Abbasi,	Islamic Azad University, Iran
Gabriela Grosseck,	West University of Timisoara, Romania
Ghasem Mirjalily,	Yazd University, Iran
Giambattista Bufalino,	University of Catania, Italy
Grigorios N. Beligiannis,	University of Patras, Greece
Grzegorz Sierpinski,	Silesian University of Technology, Poland
Guilong Liu,	Beijing Language and Culture University, China
Gulden Kokturk,	Dokuz Eylul University, Turkey
Hala Abukhalaf,	Palestine Polytechnic University, Palestine
Hamed Taherdoost,	University Canada West, Canada
Hamid Ali Abed AL-Asadi,	Iraq University, Iraq
Hamid Khemissa,	USTHB University Algiers, Algeria
Hamidreza Rokhsati,	Sapienza University of Rome, Italy
Hamzeh Khalili,	CTTC, Spain
Hatem Yazbek,	Broadcom, Israel

Hedayat Omidvar,	Research & Technology Dept, Iran
Hlaing Htake Khaung Tin,	University of Information Technology, Myanmar
Hongrui Liu,	San Jose State University, USA
Hosna Ghandeharioun,	Khorasan Institute of Higher Education, Iran
Hui Li,	Wuxi University, China
Hwang-Cheng Wang,	National Ilan University, Taiwan
Iancu Mariana,	Bioterra University of Bucharest, Romania
Ilham Huseyinov,	Istanbul Aydin University, Turkey
Isa Maleki,	Science and Research Branch, Iran
Islam Tharwat Abdel Halim,	Nile University, Egypt
Israa Shaker Tawfic,	Ministry of Migration and Displaced, Iraq
Iyad Alazzam,	Yarmouk University, Jordan
Jagadeesh HS,	APS College of Engineering (VTU), India
Jakhongir Shaturaev,	Tashkent State University, Uzbekistan
Janaki Raman Palaniappan,	Brunswick Corporation, USA
Jawad K. Ali,	University of Technology, Iraq
Jesuk Ko,	Universidad Mayor de San Andres, Bolivia
Jia Ying Ou,	York University, Canada
Joao Antonio Aparecido Cardoso,	The Federal Institute of São Paulo, Brazil
Jonah Lissner,	technion - israel institute of technology, Israel
Jong-Ha Lee,	Keimyung University, South Korea
Jubraj Khamari,	Sambalpur University Odisha, India
Jun Hu,	Harbin University of Science and Technology, China
Juntao Fei,	Hohai University, P. R. China
Kamel Benachenhou,	Blida University, Algeria
Kanstantsin MIATLIUK,	Bialystok University of Technology, Poland
Karim El Moutaouakil,	FPT/USMBA, Morocco
Katrina Sundus,	University of Jordan, Jordan
Keneilwe Zuva,	University of Botswana
Kevin Matthe Caramancion,	University at Albany, New York
Khalid M.O Nahar,	Yarmouk University, Jordan
Kire Jakimoski,	FON University, Republic of Macedonia
Kiril Alexiev,	Bulgarian Academy of Sciences, Bulgaria
Kirtikumar Patel,	Hargrove Engineers and Constructors, USA
Klenilmar Lopes Dias,	Federal Institute of Amapa, Brazil
Koffi Kanga,	Ecole supérieure Africaine des TIC, Côte d'Ivoire
Koh You Beng,	University of Malaya, Malaysia
Kurada Ramachandra Rao,	Shri Vishnu Engineering College for Women, India
Liliana Mata,	Vasile Alecsandri University of Bacau, Romania
Lixin Wang,	Columbus State University, USA
Loc Nguyen,	Loc Nguyen's Academic Network, Vietnam
Luisa Maria Arvide Cambra,	University of Almeria, Spain
M A Jabbar,	Vardhaman College of Engineering, India
M V Ramana Murthy,	Osmania university, India
MA. Jabbar,	Vardhaman College of Engg, India
Mabroukah Amarif,	Sebha University, Libya
Mahdi Sabri,	Islamic Azad University, Iran
Manish Kumar Mishra,	University of the People, USA
Manoj Kumar,	University of Petroleum and Energy Studies, India
Marco Battaglieri,	INFN, Italy
Mario Versaci,	Associate Professor - Electrical Engineering, Italy
Marta Fernandez-Diego,	Universitat Politecnica de Valencia, Spain

Masoomah Mirrashid,	Semnan University, Iran
Maumita Bhattacharya,	Charles Sturt University, Australia
Md. Monjurul Islam,	Prime University, Dhaka, Bangladesh
Meera Ramadas,	Machine Intelligence Research Lab, USA
Mervat Bamiah,	Alnahj for IT Consultancy, Saudi Arabia
Michail Kalogiannakis,	University of Crete, Greece
Micheline Al Harrack,	Marymount University, USA
Mihai Carabas,	University POLITEHNICA of Bucharest, Romania
Mihai Horia Zaharia,	Gheorghii Asachi Technical University of Iasi, Iasi
Mirsaeid Hosseini Shirvani,	Islamic Azad University, Iran
Mohamed El Ghazouani,	Chouaib Doukkali University, Morocco
Mohamed Fakir, S	ultan Moulay Slimane University, Morocco
Mohamed Hassiba,	Benbouali University Chlef, Algeria
Mohamed Ismail Roushdy,	Ain Shams University, Egypt
Mohamed Khalefa,	SUNY College at Old Westbury, United States
Mohammad A. Alodat,	Sur University College, Oman
Mohammad Jafarabad,	Qom University, Iran
Mohammed Al-Sarem,	Taibah University, Saudi Arabia
Mohd Norazmi bin Nordin,	Universiti Kebangsaan Malaysia, Malaysia
Morteza Alinia Ahandani,	University of Tabriz, Iran
Mostafa S. Shadloo,	INSA Rouen Normandie, France
Mourad Chabane Oussalah,	University of Nantes, France
Mridula Prakash,	L&T Technology Services (LTTS), India
Mu-Chun Su,	National Central University, Taiwan
Müge Karadağ,	İnönü University, Türkiye
Muhammad Aslam Javed,	The University of Central Punjab, Pakistan
Munshi Md Shafwat Yazdan,	Idaho State University, USA
Murat Tolga Ozkan,	Gazi University, Turkey
Mu-Song Chen,	Da-Yeh University, Taiwan
Mustafa S. Abd,	Baghdad university, Iraq
N.Ch.Sriman Narayana Iyengar,	Professor Information Technology, India
Nadia Abd-alsabour,	Cairo University, Egypt
Nadine Akkari,	Lebanese University, Lebanon
Nahlah Shatnawi,	Yarmouk University, Jordan
Nameer N. El-Emam,	Philadelphia University, Jordan
Naziahd Abd Kadir,	Universiti Selangor, Malaysia
Ngoc Hong Tran,	Vietnamese-German University, Vietnam
Nicolas Durand,	Aix-Marseille University, France
Nikola Ivkovic,	University of Zagreb, Croatia
Nikolai Prokopyev,	Kazan Federal University, Russia
Nishant Doshi,	Pandit Deendayal Energy University, India
Nor Syazwani Binti Mat Salleh,	Sultan Idris Education University, Malaysia
Oleksii K. Tyshchenko,	University of Ostrava, Czechia
Omar Khadir,	Hassan II University of Casablanca, Morocco
Osman Toker,	Yildiz Technical University, Turkey
Parameshchhari B D,	Professor & Head, India
Paulo Quaresma,	University of Évora, Portugal
Pavel Loskot,	ZJU-UIUC Institute, China
Ping Zhang,	Anhui Polytechnic University, China
Pranita Mahajan,	SIESGST, India
Prasang Gupta,	Emerging Technologies, India
Priyanka Srivastava,	Banaras Hindu University, India

Prudhvi Parne,	Bank of Hope and University of Louisiana, USA
Przemyslaw Falkowski-Gilski,	Gdansk University of Technology, Poland
Quang Hung Do,	University of Transport Technology, Vietnam
Rachid Zagrouba,	Imam Abdulrahman Bin Faisal University, Saudi Arabia
Rahul Kosarwal,	OAARs CORP, United Kingdom
Ramadan Elaiess,	University of Benghazi, Libya
Ramgopal Kashyap,	Amity University Chhattisgarh, India
Rana Mukherji,	ICFAI University, India
Rishabh Garg,	Birla Institute of Technology and Science, India
Robert Ssali Balagadde,	Kampala international University, Uganda
Rodrigo Pérez Fernández,	Universidad Politécnica de Madrid, Spain
Roshan Karwa,	Ram Meghe Institute of Technology & Research, India
Roya Khoii,	Islamic Azad University, Iran
Ruchi Doshi,	Universidad Azteca, Chalco, Mexico
S. M. Emdad Hossain,	University of Nizwa, Oman
Saad Al Janabi,	Al- Hikma College University, Iraq
Sabila Al Jannat,	BRAC University, Bangladesh
Said Agoujil,	Moulay Ismail University, Morocco
Saif aldeen Saad Obayes AlKadhim,	Al-Furat Al-Awsat Technical University, Iraq
Samir Kumar Bandyopadhyay,	University of Calcutta, India
Samrat Kumar Dey,	Bangladesh Open University, Bangladesh
Saroja Kanchi,	Kettering University, USA
Sarra Nighaoui,	National Engineering School of Tunis, Tunisia
Sasikumar P,	Vellore Institute of Technology, India
Sebastian Fritsch,	IT and CS enthusiast, Germany
Sébastien Combéfis,	ECAM Brussels Engineering School, Belgium
Seppo Sirkemaa,	University of Turku, Finland
Shah Khalid Khan,	RMIT University, Australia
Shahid Ali,	AGI Education Ltd, New Zealand
Shahnaz N.Shahbazova,	Azerbaijan Technical University, Azerbaijan
Shahram Babaie,	Islamic Azad University, Iran
Shahzad Ashraf,	Hohai University, China
Shamneesh Sharma,	upGrad Education Private Limited, India
Sharipbay Altynbek,	Eurasian National University, Kazakhstan
Shashikant Patil,	ViMEET ,India
Shashikant Patil,	Vishwaniketan iMEET Khalapur Raigad ,India
Shervan Fekri-Ershad,	Islamic Azad University, Iran
Shi Dong,	Zhoukou Normal University, China
Shilpa Gite,	Symbiosis International Deemed University, India
Shing-Tai Pan,	National University of Kaohsiung, Taiwan
Shin-Jer Yang,	Soochow University, Taiwan
Siarry Patrick,	Universite Paris-Est Creteil, France
Siddhartha Bhattacharyya,	Rajnagar Mahavidyalaya, India
Sidi Mohammed Meriah,	University of Tlemcen, Algeria
Sikandar Ali,	China University of Petroleum, China
Smain Femmam,	UHA University, France
Sofiane Bououden,	University Abbes Laghrour Khenchela, Algeria
Sonali Patil,	Pimpri Chinchwad College of Engineering, India
Sridhar Iyer,	SG Balekundri Institute of Technology, India
Stamatis Papadakis,	School of Education, University of Crete, Greece
Stefano Michieletto,	University of Padova, Italy
Subarna Shakya,	Tribhuvan University, Nepal

Subhendu Kumar Pani,	Krupajal Engineering College, India
Suhad Faisal Behadili,	University of Baghdad, Iraq
sukhdeep kaur,	punjab technical university, India
Sun-yuan Hsieh,	National Cheng Kung University, Taiwan
T V Rajini Kanth,	SNIST, India
Taha Mohammed Hasan,	University of Diyala, Iraq
Taleb zouggar souad,	Oran 2 University, Algeria
Tamer Mekky Ahmed Habib,	Research Associate Professor, Egypt
Tanzila Saba,	Prince Sultan University, Saudi Arabia
Taruna,	JK Lakshmiapat University, India
Tasher Ali Sheikh,	Madanapalle Institute of Technology and Science, India
Thai-Son Nguyen,	Tra Vinh University, Vietnam
Thenmalar S,	SRM Institute of Science and Technology, India
Titas De,	Data Scientist - Glance Inmobi, India
Tran Cong Manh,	Le Quy Don Technical University, Hanoi, Vietnam
Umesh Kumar Singh,	Vikram University, India
Uranchimeg Tudevdagva,	Chemnitz University of Technology, Germany
Usman Naseem,	University of Sydney, Australia
V.Ilango,	CMR Institute of Technology, India
Valerianus Hashiyana,	University of Namibia, Namibia
Vanlin Sathya,	University of Chicago, USA
Venkata Siva Kumar Pasupuleti,	VNR VJIET, India
Victor Mitrana,	Polytechnic University of Madrid, Spain
Wadii Boulila,	University of Manouba, Tunisia
Wei Lu,	Airforce Early Warning Academy, China
Weili Wang,	Case Western Reserve University, USA
William R. Simpson,	Institute for Defense Analyses, USA
WU Yung Gi,	Chang Jung Christian University, Taiwan
Xianzhi Wang,	University of Technology Sydney, Australia
Xiaodong Liu,	FHEA Edinburgh Napier University, UK
Xiao-Zhi Gao,	University of Eastern Finland, Finland
Yassine El Khanboubi,	Hassan II University of Casablanca, Morocco
Yazid Basthomi,	Universitas Negeri Malang, Indonesia
Yew Kee Wong,	BASIS International School Guangzhou, China
Yongbiao Gao,	Southeast University, China
Yousfi Abdellah,	University Mohamed V Rabat, Morocco
Yuan-Kai Wang,	Fu Jen Catholic University, Taiwan
Yu-Chen Hu,	Providence University, Taiwan
Zamira Daw,	Raytheon Technologies Research Center, USA
Zhihao Wu,	Shanghai Jiao Tong University, China
Zhihui Wu,	Harbin University of Science and Technology, China
Ziyu Jia,	Beijing Jiaotong University, China
Zopran Bopjkovic,	University of Belgrade, Serbia

Technically Sponsored by

Computer Science & Information Technology Community (CSITC)



Artificial Intelligence Community (AIC)



Soft Computing Community (SCC)



Digital Signal & Image Processing Community (DSIPC)



9th International Conference on Artificial Intelligence & Applications (ARIA 2022)

Comparing Spectroscopy Measurements in the Prediction of in Vitro Dissolution Profile using Artificial Neural Networks.....01-11
Mohamed Azouz Mrad, Kristóf Csorba, Dorián László Galata, Zsombor Kristóf Nagy and Brigitta Nagy

8th International Conference on Signal Processing and Pattern Recognition (SIPR 2022)

Fast Rank Optimization Scheme by the Estimation of Vehicular Speed and Phase Difference in MU-MIMO.....13-24
Shin-Hwan Kim, Kyung-Yup Kim, Sang-Wook Kim and Jae-Hyung Koo

8th International Conference on Software Engineering and Applications (SOFEA 2022)

An Empirical Study of the Performance of Code Similarity in Automatic Program Repair Tool.....25-36
Xingyu Zheng, Zhiqiu Huang, Yongchao Wang and Yaoshen Yu

FindMyPet: An Intelligent System for Indoor Pet Tracking and Analysis using Artificial Intelligence and Big Data.....37-48
Qinqin Guo and Yu Sun

9th International Conference on Computer Science and Engineering (CSEN 2022)

Review on Deep Learning Techniques for Underwater Object Detection.....49-63
Radhwan Adnan Dakhil and Ali Retha Hasoon Khayeat

Brand Name (To do): An Interactive and Collaborative Drawing Platform to Engage the Autism Spectrum in Art and Language Learning using Artificial Intelligence.....65-74
Xuanxi Kuang and Yu Sun

3rd International Conference on Data Science and Machine Learning (DSML 2022)

Cyberbullying Detection using Ensemble Method.....75-94
Saranyanath K P, Wei Shi and Jean-Pierre Corriveau

A Data-Driven Analytical System to Optimize Swimming Training and Competition Performance using Machine Learning and Big Data Analysis.....95-104
Tony Zheng and Yu Sun

Mining Online Drug Reviews Database for the Treatment of Rheumatoid Arthritis by using Deep Learning.....105-113
Pinar Yildirim

Generative Approach to the Automation of Artificial Intelligence Applications.....115-129
Calvin Huang and Yu Sun

Performance Evaluation for the use of ELMo Word Embedding in Cyberbullying Detection.....131-144
Tina Yazdizadeh and Wei Shi

An Intelligent Food Inventory Monitoring System using Machine Learning and Computer Vision.....145-155
Tianyu Li and Yu Sun

An Intelligent Community-Driven Mobile Application to Automate the Classification of plants using Artificial Intelligence and Computer Vision.....157-166
Yifei Tong and Yu Sun

A Simple Neural Network for Detection of Various Image Steganography Methods.....281-290
Mikołaj Plachta and Artur Janicki

Early Detection of Parkinson’s Disease using Machine Learning and Convolutional Neural Networks from Drawing Movements.....291-301
Sarah Fan and Yu Sun

11th International Conference on Natural Language Processing (NLP 2022)

Classification of Depression using Temporal Text Analysis in Social Network Messages.....167-177
Gabriel Melo, KaykeBonafé and Guilherme Wachs-Lopes

Learning Chess with Language Models and Transformers.....179-190
Michael DeLeo and Erhan Guven

A Transformer based Multi-Task Learning Approach Leveraging Translated And Transliterated Data to Hate Speech Detection in Hindi.....191-207
Prashant Kapil and Asif Ekbal

WassBERT: High-Performance BERT-based Persian Sentiment Analyzer and Comparison to Other State-of-the-art Approaches.....209-220
Masoumeh Mohammadi and Shadi Tavakoli

GRASS: A Syntactic Text Simplification System based on Semantic Representations.....221-236
Rita Hijazi, Bernard Espinasse and N ria Gala

Bantu Spell Checker and Corrector using Modified Edit Distance Algorithm (MEDA)303-316
Boago Okgetheng, Gabofetswe Malema, Ariq Ahmer, Boemo Lenyibi and Ontiretse Ishmael

3rd International Conference on Education and Integrating Technology (EDTECH 2022)

Comparison of Various Forms of Serious Games: Exploring the Potential use of Serious Game Walkthrough in Education Outside the Classroom.....237-248
Xiaohan Feng and Makoto Murakami

3DHero: An Interactive Puzzle Game Platform for 3D Spatial and Reasoning Training using Game Engine and Machine Learning.....249-262
David Tang and Yu Sun

8th International Conference of Networks, Communications, Wireless and Mobile Computing (NCWC 2022)

Frame Size Optimization Using a Machine Learning Approach in WLAN Downlink MU-MIMO Channel.....263-280
Lemlem Kassa, Jianhua Deng, Mark Davis and Jingye Cai

COMPARING SPECTROSCOPY MEASUREMENTS IN THE PREDICTION OF IN VITRO DISSOLUTION PROFILE USING ARTIFICIAL NEURAL NETWORKS

Mohamed Azouz Mrad, Kristóf Csorba
Dorián László Galata, Zsombor Kristóf Nagy and Brigitta Nagy

Department of Automation and Applied Informatics,
Budapest University of Technology and Economics, Budapest, Hungary

ABSTRACT

Dissolution testing is part of the target product quality that is essential in approving new products in the pharmaceutical industry. The prediction of the dissolution profile based on spectroscopic data is an alternative to the current destructive and time-consuming method. Raman and near-infrared (NIR) spectroscopies are two fast and complementary methods that provide information on the tablets' physical and chemical properties and can help predict their dissolution profiles. This work aims to compare the information collected by these spectroscopy methods to support the decision of which measurements should be used so that the accuracy requirement of the industry is met. Artificial neural network models were created, in which the spectroscopy data and the measured compression curves were used as an input individually and in different combinations in order to estimate the dissolution profiles. Results showed that using only the NIR transmission method along with the compression force data or the Raman and NIR reflection methods, the dissolution profile was estimated within the acceptance limits of the f_2 similarity factor. Adding further spectroscopy measurements increased the prediction accuracy.

KEYWORDS

Artificial Neural Networks, Dissolution prediction, Comparing spectroscopy measurement, Raman spectroscopy, NIR spectroscopy & Principal Component Analysis.

1. INTRODUCTION

In the pharmaceutical industry, a target product quality profile is a term used for the quality characteristics that a drug product should go through to satisfy the promised benefit from the usage and are essential in the approval of new products or the post-approval changes. A target product quality profile would include different essential characteristics. One of these is the in vitro (taking place outside of the body) dissolution profile [1]. A dissolution profile represents the concentration rate at which capsules and tablets emit drugs into the bloodstream over time. It is essential for tablets that yield a controlled release into the bloodstream over several hours. That offers many advantages over immediate release drugs, like reducing the side effects due to the reduced peak dosage and better therapeutic results due to the balanced drug release [2]. In vitro dissolution testing has been a subject of scientific research for several years and has become a vital tool for accessing product quality performance [3]. However, this method is destructive since it requires immersing the tablets in a solution simulating the human body. It is time-consuming as the measurements require taking samples over several hours. As a result, the tablets

measured represent only a limited amount of the tablets produced, also called a batch. Therefore, there is a need to find different methods that do not have the limitations of the *in vitro* dissolution testing. The prediction of the dissolution profile based on spectroscopic data is an alternative on which many articles have been published and showed promising results. RAMAN and Near Infrared (NIR) spectroscopy are two evolving techniques in the pharmaceutical industry. The interaction of NIR and RAMAN with tablets in reflection (what is reflected from the tablet) and transmission (what is transmitted through the tablet) offer the opportunity to obtain information on the physical and chemical properties of the tablets that can help predict their dissolution profiles in few minutes without destroying them. RAMAN spectroscopy is very sensitive for analyzing Active Pharmaceutical Ingredient (APIs), the part of the drug that produces the intended effect. NIR spectroscopy, on the other hand, is better used for the tableting excipients, the substances added to aid in manufacturing the tablets. Hence, RAMAN and NIR are considered to be complementary methods, straight-forward, cost-effective alternatives, and non-destructive tools in the quality control process [4, 5]. The utilization of NIR and RAMAN spectroscopy in the pharmaceutical industry has been increasing quickly. They have been applied to determine content uniformity [6], detect counterfeit drugs [7], and monitor the polymorphic transformation of tablets [8].

RAMAN and NIR spectroscopies produce a large amount of data as they consist of measurements of hundreds of wavelengths. This data can be filtered out or maintained depending on how much valuable information it provides. The information can be extracted using multivariate data analysis techniques such as Principal Component Analysis (PCA). Several researchers have used spectroscopy data and multivariate data analysis techniques to predict the dissolution profiles. Zan-Nikos et al. worked on a model that permits hundreds of NIR wavelengths to be used to determine the dissolution rate [9]. Donoso et al. [10] used the NIR reflectance spectroscopy to measure the percentage of drug dissolution from a series of tablets compacted at different compressional forces using linear regression, nonlinear regression, and Partial Least Square (PLS) models. Freitas et al. [11] created a PLS calibration model to predict drug dissolution profiles at different time intervals and for media with different pH using NIR reflectance spectra. Hernandez et al. [12] used PCA to study the sources of variation in NIR spectra and a PLS-2 model to predict the dissolution of tablets subjected to different strain levels.

Artificial Neural Networks (ANNs) are suitable for complex and highly nonlinear problems. They have been used in the pharmaceutical industry in many aspects, such as the prediction of chemical kinetics [13], monitoring a pharmaceutical freeze-drying process [14], and solubility prediction of drugs [15]. ANN models have also been used to predict the dissolution profile based on spectroscopic data. Ebube et al. [16] trained an ANN model with the theoretical composition of the tablets to predict their dissolution profile. Galata et al. [17] developed a PLS model to predict the contained drotaverine (DR) and the hydroxypropyl methylcellulose (HPMC) content of the tablets, which are respectively the drug itself and a jellying material that slows down the dissolution, based on both RAMAN and NIR Spectra. They used the predicted values and the measured compression force as input to an ANN model to predict the dissolution profiles. Mrad et al. [18] used RAMAN and NIR spectroscopy along with the compression force to estimate the dissolution profiles of the tablets defined in 53-time points using ANN models. Using NIR spectra, RAMAN spectra, and the concentration force to predict the dissolution profile is a fast method that requires minimal human labor and makes it easier to evaluate a more significant amount of the batch. The decision of which measurements (Raman Reflection, Raman Transmission, NIR Reflection, NIR transmission, and compression force) to use in predicting dissolution profiles can be supported if the methods are compared. We aimed to support the decision of which measurements to use by comparing how well these measurements are in predicting dissolution profiles. In this paper, our goal was to extract helpful information from the NIR, RAMAN spectroscopy, and the compression curve of the tablets using a multivariate data

analysis technique. ANN models were then created using the extracted information as input individually and in different combinations to predict the dissolution profiles represented in the 53-time points.

2. DATA AND METHODS

In Section 2, the data used will be described, and the methods used for the data pre-processing will be presented. The artificial neural network models will be presented, and finally, the error measurement methods adopted to evaluate the results.

2.1. Data Description

The chemical engineers in this paper prepared the NIR and RAMAN spectroscopy measurements (Figure. 1), along with the pressure curves extracted during the compression of the tablets. The data consists of the NIR reflection and transmission, RAMAN reflection and transmission spectra, the compression force-time curve, and the dissolution profile of 148 tablets (Figure. 2).

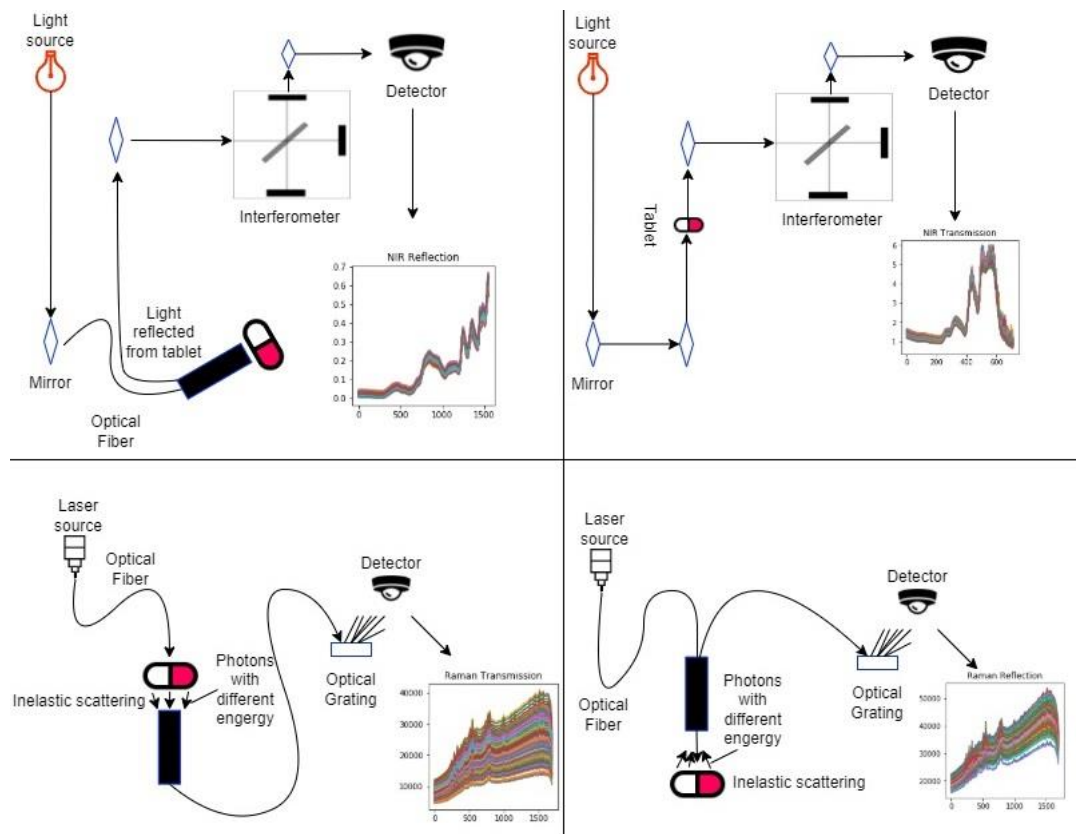


Figure 1. Spectroscopy methods: NIR reflection, NIR transmission, Raman transmission and Raman Reflection (Clockwise starting from Top left corner)

The tablets were produced with a total of 37 different settings. Three parameters were varied: drotaverine content, HPMC content, and the compression force. From each setting, four tablets were selected for analysis (37*4). The NIR and RAMAN measurements on the tablets were carried out by Bruker Optics MPA FT-NIR spectrometer, and Kaiser RAMAN RXN2 Hybrid Analyzer equipped Pharmaceutical Area Testing (PhAT) probe. The spectral range for NIR reflection spectra was 4,000–10,000 cm^{-1} with a resolution of 8 cm^{-1} , representing 1556

wavelength points. NIR transmission spectra were collected in the 4000-15,000 cm^{-1} wavenumber range with 32 cm^{-1} spectral resolution representing 714 wavelength points. RAMAN spectra were recorded in the range of 200-1890 cm^{-1} with 4 cm^{-1} spectral resolution for transmission and reflection measurements representing 1691 points. Two spectra were recorded for each tablet in both NIR and RAMAN. The pressure during the compression of the tablet was recorded in 6037 time points. The dissolution profiles of the tablets were recorded using Hanson SR8-Plus in vitro dissolution tester. The length of the dissolution run was 24 hours. During this period, samples were taken at 53-time points (at 2, 5, 10, 15, 30, 45, and 60 min, after that once every 30 min until 1440 min).

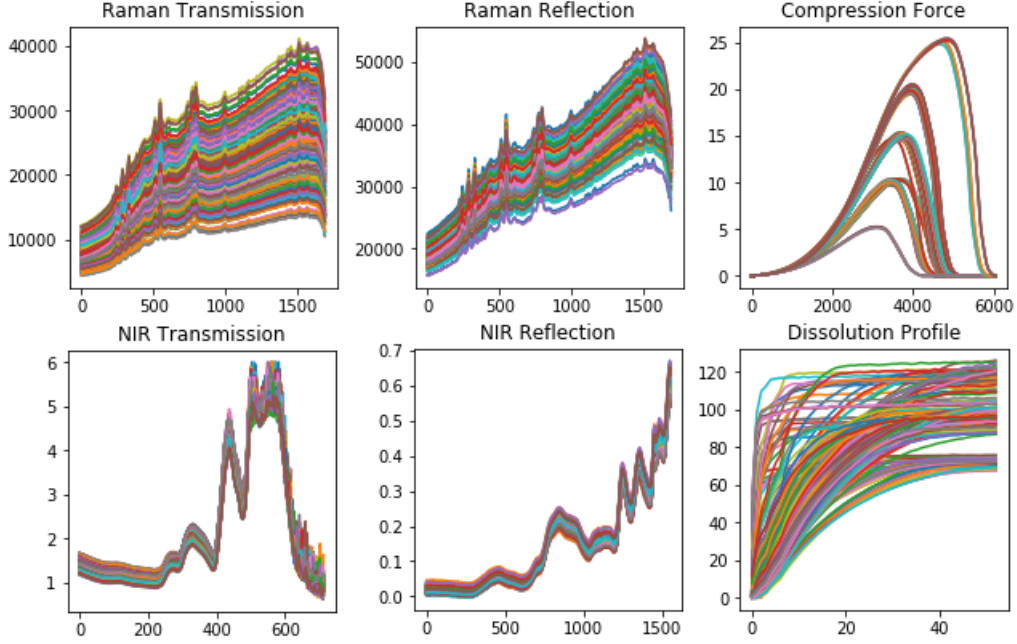


Figure 2. Dataset composed of NIR, RAMAN transmission and reflection the compression curve and the dissolution profiles

2.2. Data Analysis

The collected data were visualized and analyzed using MATLAB and Excel in order to detect and fix missed and wrong values: Setting first point of the dissolution curves to zero, detecting missed values, and fixing negative values found due to error of calibration, etc. Specifically, the data is represented in matrices N_i^n for NIR transmission data and M_j^n for NIR reflection data, where $i=1556$, $j=714$. R_k^n and Q_k^n respectively for Raman reflection and transmission data where $k=1691$. C_l^n for the compression force data where $l=6037$ and P_s^n for the dissolution profiles where $s=54$. With n representing the number of samples which is equal to 296. All the different NIR, RAMAN and the compression force matrices have been standardized using scikit-learn preprocessing method: StandardScaler. StandardScaler fits the data by computing the mean and standard deviation and then centers the data following the equation $\text{Std}(NS) = (NS - u)/s$, where NS is the non-standardized data, u is the mean of the data to be standardized, and s is the standard deviation. All the different standardized NIR, RAMAN and the compression force matrices have been row-wise concatenated to form a new matrix D_m^n where $n=296$ and $m=i+j+2k+l=11686$ as follow: $D_m^n = (N_i^n | M_j^n | R_k^n | Q_k^n | C_l^n)$.

After standardization, PCA was applied to the different standardized matrices as well as the merged data D_m^n in order to reduce the dimension of the data while extracting and maintaining the most useful variations. Basically, taking D_m^n as an example we construct a symmetric $m \times m$ dimensional covariance matrix Σ (where $m=11686$) that stores the pairwise covariances between the different features calculated as follow:

$$\sigma_{j,k} = \frac{1}{n} \sum_{i=1}^n (x_j^{(i)} - \mu_j)(x_k^{(i)} - \mu_k) \quad (1)$$

With μ_j and μ_k are the sample means of features j and k . The eigenvectors of Σ represent the principal components, while the corresponding eigenvalues define their magnitude. The eigenvalues were sorted by decreasing magnitude in order to find the eigenpairs that contains most of the variances. Variance explained ratios represents the variances explained by every principal component (eigenvectors), it is the fraction of an eigenvalue λ_j and the sum of all the eigenvalues. The following plot (Figure. 3) shows the variance explained ratios and the cumulative sum of explained variances. It indicates that the first principal component alone accounts for 50% of the variance. The second component account for approximately 20% of the variance.

The plot indicates that the seven first principal components combined explain almost 96% of the variance in D . These components are used to create a projection matrix W which we can use to map D to a lower dimensional PCA subspace D' consisting of less features:

$$D = [d_1, d_2, d_3, \dots, d_m] \in R^m \rightarrow D' = DW, W \in R^{m \times v} \quad (2)$$

$$D' = [d_1, d_2, d_3, \dots, d_m], d \in R^m \quad (3)$$

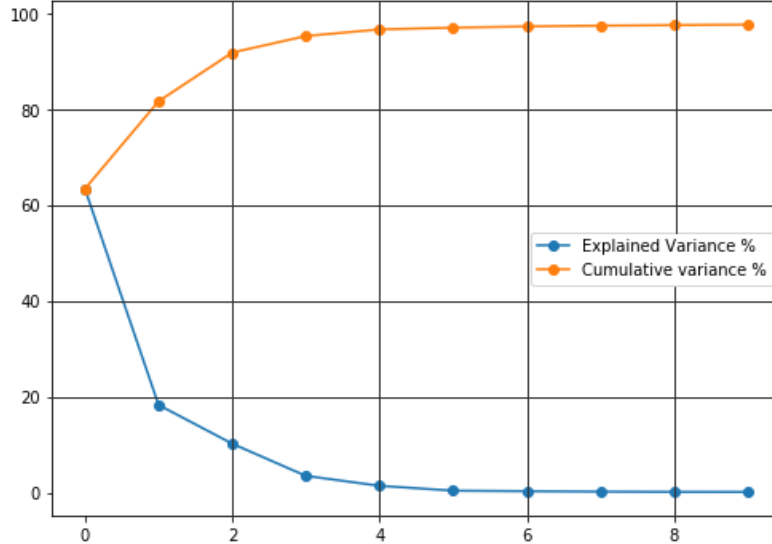


Figure 3. Explained PCA and Cumulative variances.

2.3. Artificial Neural Networks

ANN models were used to predict the dissolution profiles of the tablets. The models were created using the Python library Sklearn. Different ANN models were created, with different inputs and output targets each time. The models used the rectified linear unit activation function referred to as ReLU on the hidden layers and the weights on the models were optimized using LBFGS

optimizer which is known to perform better and converge faster on dataset with small number of samples (296 in our case). Adam optimizer was tried as well but did not perform as good as LBGFS. The mean-squared error (MSE) was the loss function used by the optimizer in the different models. The training target for the models were the remaining part of the dissolution profiles, e.g., the dissolution curves are described in 53 points, if 10 points are used in the input then the remaining part of 43 points is the training target. The number of layers on the models and the number of neurons were optimized based on their performances. Regularization term has been varied in order to reduce overfitting. In each training, 16% of the training samples (49 samples) were selected randomly for testing. The accuracy of the model's predictions was calculated by evaluating the similarity of the predicted and measured parts of the dissolution profiles using the f_2 similarity values.

2.4. Error Measurement

Two mathematical methods are described in the literature to compare dissolution profiles [19]. A difference factor f_1 which is the sum of the absolute values of the vertical distances between the test and reference mean values at each dissolution time point, expressed as a percentage of the sum of the mean fractions released from the reference at each time point. This difference factor f_1 is zero when the mean profiles are identical and increases as the difference between the mean profiles increases:

$$f_1 = \frac{\sum_{t=1}^n |R_t - T_t|}{\sum_{t=1}^n |R_t|} * 100 \quad (4)$$

Where R_t and T_t are the reference and test dissolution values at time t . The other mathematical method is the similarity function known as the f_2 measure, it performs a logarithmic transformation of the squared vertical distances between the measured and the predicted values at each time point. The value of f_2 is 100 when the test and reference mean profiles are identical and decreases as the similarity decreases.

$$f_2 = 50 \log_{10} \left[\left(1 + \frac{1}{n} \sum_{t=1}^n (R_t - T_t)^2 \right)^{-0.5} \right] * 100 \quad (5)$$

Values of f_1 between zero and 15 and of f_2 between 50 and 100 ensure the equivalence of the two dissolution profiles. The two methods are accepted by the FDA (U.S. Food and Drug Administration) for dissolution profiles comparison, however the f_2 equation is preferred, thus in this paper maximizing the f_2 will be prioritized.

3. RESULTS AND DISCUSSIONS

In this section the results after the PCA dimensionality reduction will be discussed. The results and the performance of the Artificial Neural Network models created will be presented.

3.1. Dimensionality reduction using PCA

Principal component analysis transformation was applied in a first step to the standardized NIR and Raman spectra recorded in reflection and transmission mode ($N_i^n, M_j^n, R_k^n, Q_k^n$ matrices) and the standardized compression force curve C_1^n , and in a second step on all the data merged in matrix D_m^n in order to investigate the effect of the transformation on the merged and the separated data.

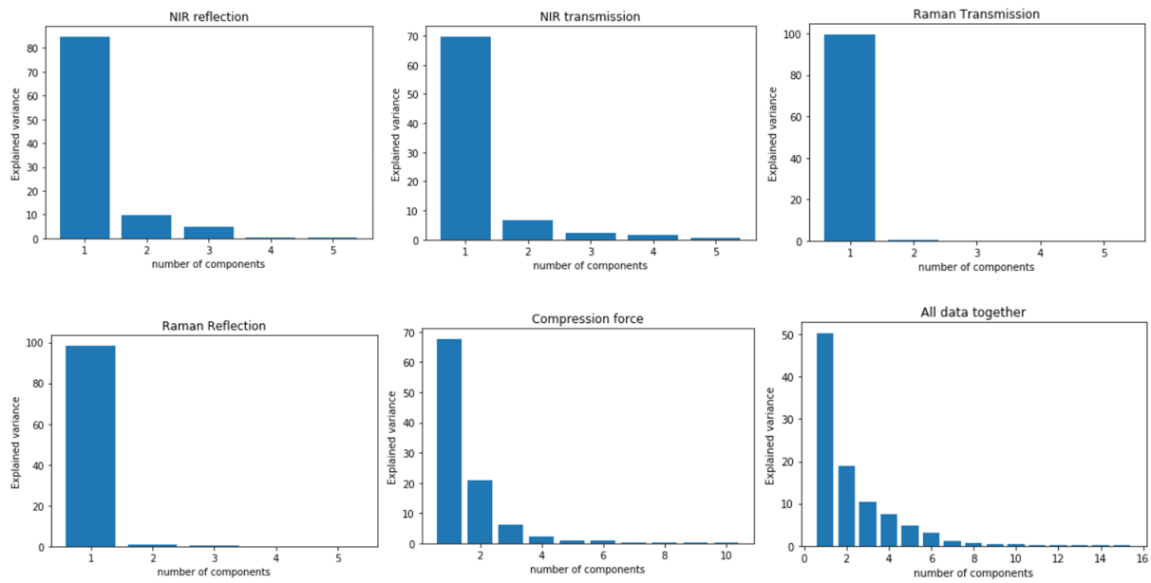


Figure 4. Explained variance of spectral data, compression force, and all data merged.

The resulting PCA decompositions, showed that in the case of NIR reflection, three principal components explaining 84.79%, 9.67% and 4.83% of the total variance in the data, respectively, leading to a cumulative explained variance of more than 99%. Four principal components explained more than 80% of the total variances of the NIR transmission data and 95% of the compression force data. However, for Raman transmission, the first principal component alone explains 99.69% of the variance in the data. The first two principal components explain 98.51% and 1.01% of the variance in the Raman Reflection data, respectively. For matrix D_m^n , 7 principal components explain more than 95 % of the variance and 33 explain more than 99% of the merged standardized data. These data resulting from the PCA decompositions were used as inputs for the Artificial neural network models individually and in different combinations in order to compare them based on how helpful they are in the prediction of dissolution profiles. For all measurements, the number of components explaining 99% of the total variance were kept.

3.2. Predicting the Dissolution Profile using Artificial neural network

The results showed that by using only one measurement as input, the artificial neural network models were not able to predict the dissolution profiles within the acceptance range (50-100) of the f_2 factor, as the maximum average f_2 was 47.56 using the compression force as input for the ANN model. Thus, further measurements were added in order to improve the results. By Combining two measurements, two ANN models were able to predict the dissolution profiles within the acceptance range of f_2 . The first model used NIR transmission along with the compression force measurements as an input, this model was able to reach an f_2 average of 60.69. The second ANN model used the Raman Reflection with NIR Reflection methods, and had an f_2 average of 50.22. Further measurements were added to verify the effect on the prediction accuracy. The results showed that ANN models that used the combination of NIR Transmission and the compression force along with either Raman Reflection or NIR Reflection, were able to predict the dissolution profile with an $f_2 > 60$. The results show that NIR transmission and the compression force are very important in the prediction of dissolution profiles, adding further measurements to these two can slightly improve the results.

Table 1. Results of the predictions using one measurement.

	F_2 Results of the Prediction
RAMAN TRANSMISSION	40.82
RAMAN REFLECTION	41.64
NIR TRANSMISSION	43.89
NIR REFLECTION	43.89
COMPRESSION FORCE	47.56

Table 2. Results of the predictions using combination of two measurements.

	F_2 Results of the Prediction
NIR TR+RAMAN RE	45.55
NIR TR+ Compression Force	60.69
NIR TR+NIR RE	42.52
NIR TR+RAMAN TR	44.10
RAMAN RE+ Comp Force	47.73
RAMAN RE + NIR RE	50.22
RAMAN RE+RAMAN TR	43.05
Comp Force+ NIR RE	49.09
Comp Force+ RAMAN TR	47.68
NIR RE+ RAMAN TR	46.62

Table 3. Results of the predictions using combination of three measurements

	F_2 Results of the Prediction
NIR TR+RAMAN RE+ Comp	61.03
NIR TR+ RAMAN RE + NIR RE	49.77
NIR TR+RAMAN RE + RAMAN TR	47.76
NIR TR+ Comp+ NIR RE	61.24
NIR TR+ Comp+ RAMAN TR	59.12
NIR TR + NIR RE+ RAMAN TR	45.56
RAMAN RE+ Comp +NIR RE	55.98
RAMAN RE+ Comp + RAMAN TR	51.86
RAMAN RE+NIR RE+ RAMAN TR	47.94
Comp+ NIR RE + RAMAN TR	48.52

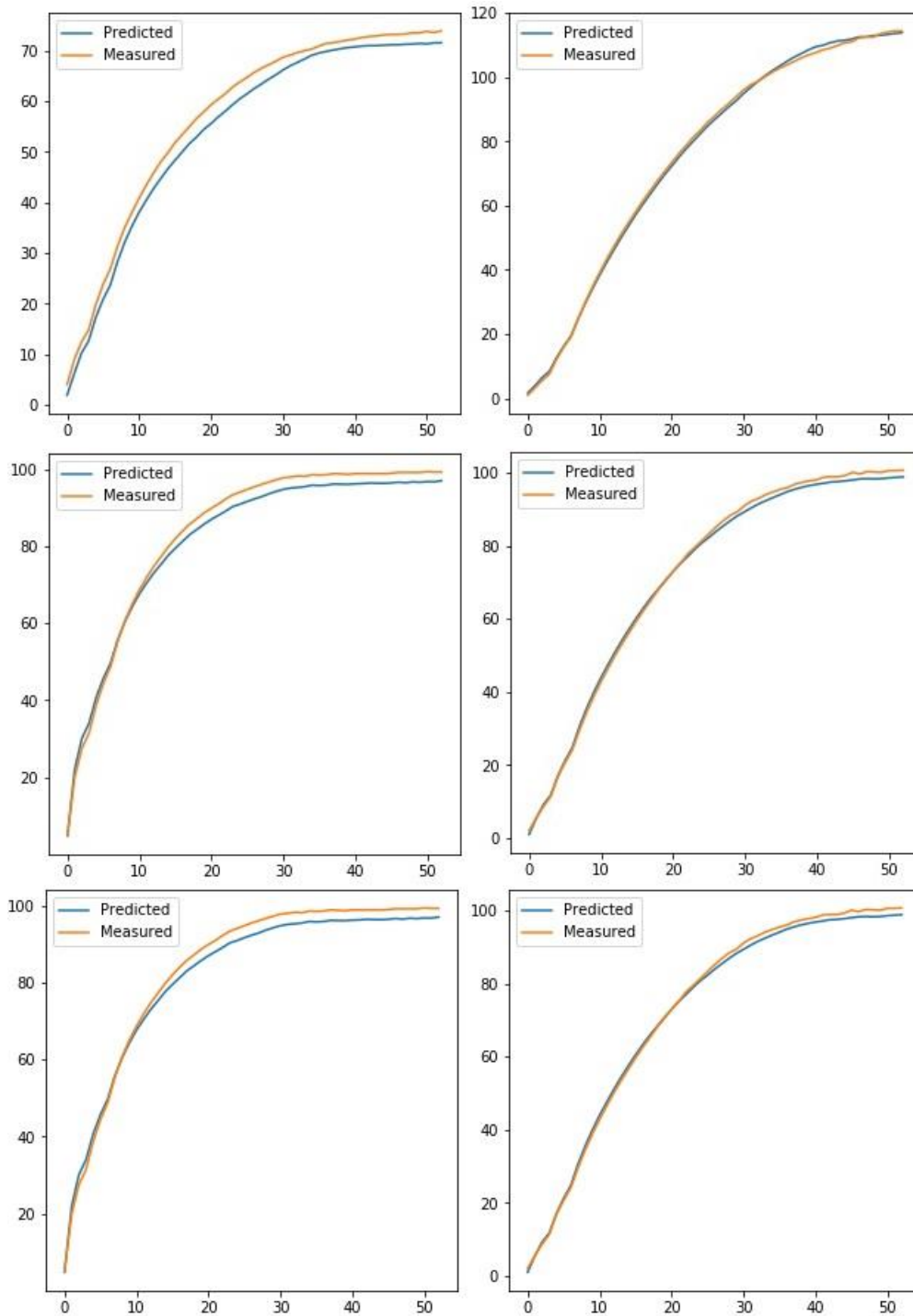


Figure 5. Sample predicted dissolution curves using NIR TR+ Comp+ NIR RE combination

4. CONCLUSIONS

The current work aimed to compare the measurements in the prediction of dissolution profiles using artificial neural network models. The spectroscopy data along with the compression force were standardized, and their dimensionality were reduced using PCA. ANN models were created using these data as input both as individual measurements, then a combination of two

measurements then finally three measurements. The results showed that using only the NIR transmission method along with the compression force data or the Raman and NIR reflection methods, the dissolution profile was estimated within the acceptance limits of the f_2 similarity factor. The results showed that NIR transmission and the compression force are very important in the prediction of dissolution profiles, adding further measurements to these two slightly improved the results.

ACKNOWLEDGEMENTS

Project no. FIEK_16-1-2016-0007 has been implemented with the support provided from the National Research, Development and Innovation Fund of Hungary, financed under the Centre for Higher Education and Industrial Cooperation Research infrastructure development (FIEK_16) funding scheme.

REFERENCES

- [1] X. Y. Lawrence, "Pharmaceutical quality by design: product and process development, understanding, and control," *Pharmaceutical research*, vol. 25, no. 4, pp. 781791, 2008.
- [2] G. A. Susto and S. McLoone, "Slow release drug dissolution profile prediction in pharmaceutical manufacturing: A multivariate and machine learning approach," in *2015 IEEE International Conference on Automation Science and Engineering (CASE)*, pp. 1218-1223, IEEE, 2015
- [3] R. Patadia, C. Vora, K. Mittal, and R. Mashru, "Dissolution criticality in developing solid oral formulations: from inception to perception," *Critical Reviews in Therapeutic Drug Carrier Systems*, vol. 30, no. 6, 2013.
- [4] A. Hédoux, "Recent developments in the raman and infrared investigations of amorphous pharmaceuticals and protein formulations: a review," *Advanced drug delivery reviews*, vol. 100, pp. 133–146, 2016.
- [5] J. U. Porep, D. R. Kammerer, and R. Carle, "On-line application of near infrared (nir) spectroscopy in food production," *Trends in Food Science & Technology*, vol. 46, no. 2, pp. 211–230, 2015.
- [6] Arruabarrena, J., J. Coello, and S. Maspocho. "Raman spectroscopy as a complementary tool to assess the content uniformity of dosage units in break-scored warfarin tablets." *International journal of pharmaceutics* 465.1-2, pp. 299-305, 2014.
- [7] Dégardin, Klara, Aurélie Guillemain, Nicole Viegas Guerreiro, and Yves Roggo. "Near infrared spectroscopy for counterfeit detection using a large database of pharmaceutical tablets." *Journal of pharmaceutical and biomedical analysis* 128, pp. 89-97, 2016.
- [8] Terra, Luciana A., and Ronei J. Poppi. "Monitoring the polymorphic transformation on the surface of carbamazepine tablets generated by heating using near-infrared chemical imaging and chemometric methodologies." *Chemometrics and Intelligent Laboratory Systems* 130, pp. 91-97, 2014.
- [9] P. N. Zannikos, W.-I. Li, J. K. Drennen, and R. A. Lodder, "Spectrophotometric prediction of the dissolution rate of carbamazepine tablets," *Pharmaceutical research*, vol. 8, no. 8, pp. 974–978, 1991.
- [10] M. Donoso and E. S. Ghaly, "Prediction of drug dissolution from tablets using near-infrared diffuse reflectance spectroscopy as a nondestructive method," *Pharmaceutical development and technology*, vol. 9, no. 3, pp. 247–263, 2005.
- [11] M. P. Freitas, A. Sabadin, L. M. Silva, F. M. Giannotti, D. A. do Couto, E. Tonhi, R. S. Medeiros, G. L. Coco, V. F. Russo, and J. A. Martins, "Prediction of drug dissolution profiles from tablets using nir diffuse reflectance spectroscopy: a rapid and nondestructive method," *Journal of pharmaceutical and biomedical analysis*, vol. 39, no. 1-2, pp. 17–21, 2005.
- [12] E. Hernandez, P. Pawar, G. Keyvan, Y. Wang, N. Velez, G. Callegari, A. Cuitino, B. Michniak-Kohn, F. J. Muzzio, and R. J. Románach, "Prediction of dissolution profiles by non-destructive near infrared spectroscopy in tablets subjected to different levels of strain," *Journal of pharmaceutical and biomedical analysis*, vol. 117, pp. 568–576, 2016.
- [13] M. Szaleniec, M. Witko, R. Tadeusiewicz, and J. Goclon, "Application of artificial neural networks and dft-based parameters for prediction of reaction kinetics of ethylbenzene dehydrogenase," *Journal of computer-aided molecular design*, vol. 20, no. 3, pp. 145–157, 2006.

- [14] E. N. Dr̂agoi, S. Curteanu, and D. Fissore, "On the use of artificial neural networks to monitor a pharmaceutical freeze-drying process," *Drying Technology*, vol. 31, no. 1, pp. 72–81, 2013.
- [15] A. G. JOUYBAN, S. Soltani, and Z. K. ASADPOUR, "Solubility prediction of drugs in supercritical carbon dioxide using artificial neural network," 2007.
- [16] N. K. Ebube, T. McCall, Y. Chen, and M. C. Meyer, "Relating formulation variables to in vitro dissolution using an artificial neural network," *Pharmaceutical development and technology*, vol. 2, no. 3, pp. 225–232, 1997.
- [17] D. L. Galata, A. Farkas, Z. K̄onyves, L. A. M̄esz̄aros, E. Szab̄o, I. Csontos, A. P̄alos, G. Marosi, Z. K. Nagy, and B. Nagy, "Fast, spectroscopy-based prediction of in vitro dissolution profile of extended release tablets using artificial neural networks," *Pharmaceutics*, vol. 11, no. 8, p. 400, 2019.
- [18] Mrad, Mohamed Azouz, Kristóf Csorba, Dorián László Galata, Zsombor Kristóf Nagy, and Brigitta Nagy. "Spectroscopy-Based Prediction of In Vitro Dissolution Profile Using Artificial Neural Networks." In *International Conference on Artificial Intelligence and Soft Computing*, pp. 145-155. Springer, Cham, 2021.
- [19] J. Moore and H. Flanner, "Mathematical comparison of dissolution profiles," *Pharmaceutical technology*, vol. 20, no. 6, pp. 64–74, 1996.

FAST RANK OPTIMIZATION SCHEME BY THE ESTIMATION OF VEHICULAR SPEED AND PHASE DIFFERENCE IN MU-MIMO

Shin-Hwan Kim¹, Kyung-Yup Kim¹,
Sang-Wook Kim² and Jae-Hyung Koo³

¹Access Network Technology Team, Korea Telecom, Seoul, Korea

²Access Network Technology Department, Korea Telecom, Seoul, Korea

³Network Research Technology Unit, Korea Telecom, Seoul, Korea

ABSTRACT

Resent MU-MIMO (Multi User-Multi Input Multi Output) scheme is one of the important and advanced technologies. In particular, it is a suitable technique to increase the capacity from the point of view of solving cell load, which is one of the big issues in the contents of 5G commercial field optimization. While this MU-MIMO technology has an important advantage of cell capacity expansion, there is a disadvantage like an interference problem due to each multi-user beams. It is important to use the advanced beamforming technology for MU-MIMO to overcome these disadvantages. Therefore, by applying the interference cancelling technology among inter UE (User Equipment) beams to improve each UE's performance, it will contribute to improving the cell throughput. This paper introduces the various techniques of eliminating interference in MU-MIMO system. Also, it is important that UE reports rank indicator reflected the interference of multi-user beams. This paper analyses the problem of the conventional method of the rank decision in MU-MIMO system, estimates the vehicular speed quickly with the proposed rank optimization technique, and shows the DL (Downlink) UE's performance is improved by applying a proposed rank value suitable for vehicular speed. This technique will be effectively applied to increase the overall cell capacity by improving the DL UE's throughput in the MU-MIMO system.

KEYWORDS

MU-MIMO, 5G, multi-user, interference, UE, DL, rank indicator, cell capacity.

1. INTRODUCTION

As NR (New Radio) system was commercialized, many of the basic techniques required for wireless access became commercialized and stable. Now, beyond the basic techniques of 5G, 5G system is required to transmit the reliable traffic having high-quality to more users through more advanced technologies. Among them, MU-MIMO technology, which enables the cell capacity of 5G system to be dramatically increased, has attracted attention. The prerequisite for the commercialization of this MU-MIMO technology is to increase the single UE throughput by solving the interference problem between multi-user beams, and in order to do so, a rank optimization technique suitable for the MU-MIMO system is required. This paper introduces new MU-MIMO rank optimization technique suitable for MU-MIMO system and compares and analyses it with conventional technique.

2. BACKGROUND

This section mentions the definition of MU-MIMO, and the interference that occurs during MU-MIMO operation, and then introduces the main techniques at the transceiver necessary to eliminate the interference respectively.

2.1. MU-MIMO Definition

MU-MIMO is a set of MIMO technologies for multipath wireless communication, in which multiple users or terminals, each radioing over one or more antennas, communicate with one another. In contrast, SU-MIMO (Single-User Multi-Input Multi-Output) involves a single multi-antenna-equipped user or terminal communicating with precisely one other similarly equipped node. Analogous to how OFDMA (Orthogonal Frequency Division Multiplexing Access) adds multiple-access capability to OFDM (Orthogonal Frequency Division Multiplexing) in the cellular-communications, MU-MIMO adds multiple-user capability to MIMO in the wireless communication. Figure.1 is the summary picture of MU-MIMO.

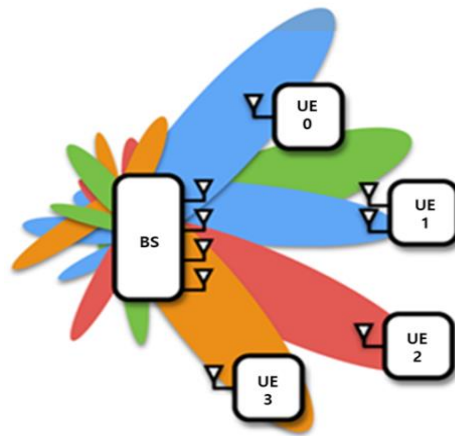


Figure 1. Definition of MU-MIMO

2.2. MU Interference

The inter-user interference characteristics are an essential factor for system evaluation. For example, when base station transmit UE1 beam and UE2 beam as below Figure.2, interference such as Figure.2 occurs when the interference between each beam is not taken into account at all. When the base station of upper Figure.2 transmits the blue UE1 main lobe beam, the power of the red UE2 side lobe beam on the lower Figure.2 is transmitted very strongly, so the SINR (Signal to Interference plus Noise Ratio) of the blue UE 1 main lobe beam becomes very small to -2dB. Similarly, when the base station of lower Figure.2 transmits the red UE2 main lobe beam, the power of the blue UE1 side lobe beam on the upper is strongly transmitted, so the SINR of the red UE2 main lobe beam becomes smaller to 4.8dB.

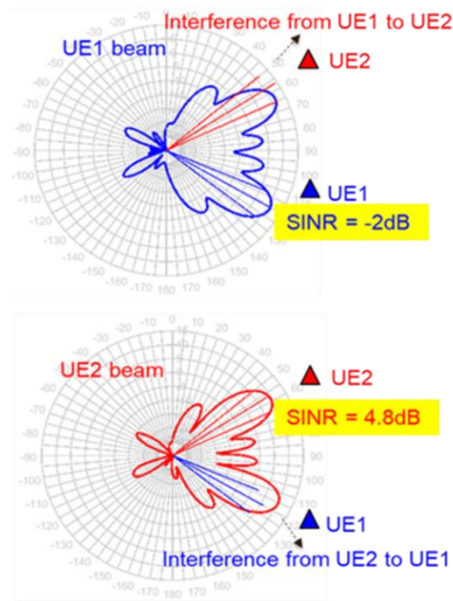


Figure 2. Beams before interference nulling

2.3. MU Interference Cancellation of Transmitter

To eliminate the interference between UE beams described in Figure.2, the nulling techniques of various methods are introduced in the section below. Figure.3 is a new MU-MIMO beam shape after nulling in these various ways. For example, when the upper base station transmits the blue UE1 main lobe beam on the upper Figure.3, the blue UE1 main lobe beam with the newly calculated weight with nulling algorithm may be slightly weakened, but the power of the red UE2 side lobe beam on the lower Figure.3 is considerably nulled, so the SINR of the blue UE1 main lobe beam is very large to 24.2dB. Similarly, when the lower base station transmits the red UE2 main lobe beam on the lower Figure.3, the red UE2 main lobe beam with the newly calculated weight with nulling algorithm can be slightly weakened, but the power of the blue UE1 side lobe beam on the upper Figure.3 is nulled, so the SINR of the red UE2 main lobe beam is restored to 24.7dB.

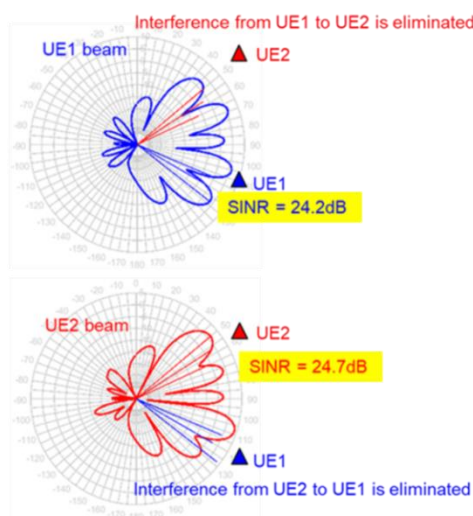


Figure 3. Beams after interference nulling

2.4. MU Interference Cancellation of Transmitter: Zero-Forcing Transmitter

Zero-forcing beamforming is a method of spatial signal processing by which a multiple antenna transmitter can null the multi-user interference in a multi-user MIMO wireless communication system. When the channel state information is perfectly known at the transmitter, then the zero-forcing beamformer is given by the pseudo-inverse of the channel matrix. Figure.4 briefly represents the channel model of MU-MIMO. Figure.5 represents a block diagram including a channel of MU-MIMO and zero-forcing beamforming. Here, the X character labeled I is an interference signal. Its mathematical model may be represented as shown in (1), and if the (1) is solved, each is represented (4), (5). The $h_1 w_2 s_2$ of (4) is the interference signal of UE2 beam. In addition, the $h_2 w_1 s_1$ of (5) is the interference signal of UE1 beam. In order to get rid of this interference signal, (6) that is, the zero-forcing beamforming function is assigned. As a result, the new weight is multiplied, such as (8), (9), and the interference signal is eliminated.

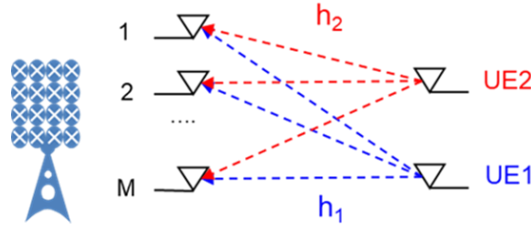


Figure 4. Channel modeling of MU-MIMO

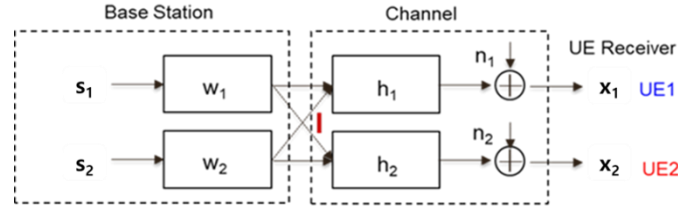


Figure 5. The Block diagram of MU-MIMO

$$x = HWS + n \quad (1)$$

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} \begin{bmatrix} w_1 & w_2 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} + \begin{bmatrix} n_1 \\ n_2 \end{bmatrix} \quad (2)$$

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} h_1 w_1 & h_1 w_2 \\ h_2 w_1 & h_2 w_2 \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \end{bmatrix} + \begin{bmatrix} n_1 \\ n_2 \end{bmatrix} \quad (3)$$

$$x_1 = h_1 w_1 s_1 + h_1 w_2 s_2 + n_1 \quad (4)$$

$$x_2 = h_2 w_1 s_1 + h_2 w_2 s_2 + n_2 \quad (5)$$

$$W = H^H (HH^H)^{-1} \quad (6)$$

$$x = HWS + n = (HH^H (HH^H)^{-1})s + n \quad (7)$$

$$x_1 = h_1 w_{1,new} s_1 + n_1 \quad (8)$$

$$x_2 = h_2 w_{2,new} s_2 + n_2 \quad (9)$$

2.5. MU Interference Cancellation of Transmitter: SVD Transmitter

SVD (Singular Value Decomposition) is a method of obtaining pseudo inverse by decomposition with singular value, when the inverse matrix of the channel cannot be solved as an abbreviation of the singular value decomposition. This method does precoding in the transmitter by

decomposing into singular value (Σ) and unitary matrix (U, V) as shown in the following (10), and post-coding in the receiver to obtain identity matrix. Overall, this overcomes the channel by pseudo-inverse as shown in (14).

$$H = U\Sigma V^H \quad (10)$$

$$\bar{x} = U^H(Hs + n) \quad (11)$$

$$\bar{x} = U^H(U\Sigma V^H s + n) \quad (12)$$

$$\bar{x} = U^H U \Sigma V V^H \bar{s} + U^H n \quad (13)$$

$$\bar{x} = \Sigma \bar{s} + \bar{n} \quad (14)$$

3. CONVENTIONAL RANK DECISION

The Conventional rank decision method is a popular method used by SU-MIMO. That is, it is a method of determining rank when the correlation coefficient among the path of UE is transmitted by CSI-RS (Channel State Information-Reference Signal) is a certain value or less. That is, by identifying the degree of correlation among the paths, it is a way to increase rank only when the signals among paths are guaranteed a certain level of independence. To use this method of independence among these paths in MU-MIMO, a special CSI-RS must be transmitted that can well reflect the characteristics of the signal among multi-users. A detailed description of this method is as follows.

3.1. The Use of NZP-CSI-RS-CM and NZP-CSI-RS-IM

RI (Rank Indicator) reported by the UE receives the CSI-RS transmitted by the base station to determine the independence among the UE paths. Therefore, in order to well represent the characteristics of the MU-MIMO beam, it is necessary to transmit a CSI-RS that represents the interference among UEs well. That's the signal of NZP-CSI-RS-CM (Non-Zero Power-Channel State Information-Reference Signal-Channel Measurement), NZP-CSI-RS-IM (Non-Zero Power-Channel State Information-Reference Signal-Interference Measurement). Figure.6 is an example of RE(Resource Element) mapping for transmitting NZP-CSI-RS-CM, NZP-CSI-RS-IM. When transmitting the CSI-RS to the base station as shown below, the UE will perform interference measurement by estimating the level of interference at the empty white color RE position and will determine the rank accordingly. The white empty RE position is defined as NZP-CSI-RS-IM in the base station to empty the signal, and the red RE position is defined as NZP-CSI-RS-CM in the base station to inform and transmit the CSI-RS-CM signal to the UE. In case of single UE beam, empty RE position doesn't exist intra-cell interference, and only exist inter-cell interference, so rank value can be high. However, in case of multi UE beams, empty RE position does exist the intra-cell interference relatively highly and inter-cell interference is also present, so the rank value is likely to be low. The characteristics of this conventional method are as follows.

- It uses UE beam, which is QCLed (Quasi-Co-Located) MU-MIMO CSI-RS beam.
- It depends on UE which reports RI.

It chooses multi-user as RRC (Radio Resource Control) configuration message transmitted by base station.

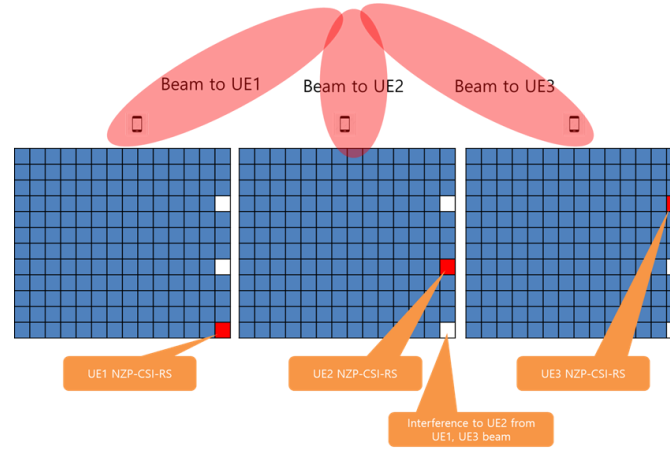


Figure 6. NXP-CSI-RS-CM, NXP-CSI-RS-IM in case 3 UEs

3.2. Disadvantage of Conventional Scheme

The method of changing the CSI-RS for MU pairing to RRC message is very slow, and is not suitable for mobile environments, because the MU-MIMO system, which has a lot of interference, is to be needed to change rank in real time faster than in a typical SU-MIMO system. Its performance is shown in the performance section 5. Also, the disadvantages of conventional scheme were described as follows.

- RRC message technique for MU pairing is slow to variation of channel.
- Because the UE depends on the RI value it reports, it reflects the characteristics of the UE type rather than the variation of channel.
- This is how the UE relies on the RI value it reports, which is unfavorable to the optimization of base station driving.
- Even if multi-user pairing is matched to each other instantaneously, the UE-specific CSI-RS beam must also guarantee the mobility as often as UE is moved.

4. OPTIMIZED RANK DECISION

To compensate for the disadvantages of this conventional method, the optimized method is introduced as follows. This new approach is divided into two main parts. The first scheme is the rank decision method according to the range of vehicular speed. The second proposed scheme is a method of estimating and calculating the vehicular speed in order to decide rank value. The combination of these two methods is not only based on a UE dependent scheme, but also based on a base station dependent scheme in the case rank decision. It is possible to cope with more precise and the variety of channel quickly. Table 1 is compares the pros and cons of conventional, proposed scheme.

Table 1. Pros and cons

Scheme	Pros	Cons
Conventional	High rank	Slow change of rank
Proposed	Fast change of rank	Complex implementation

4.1. Rank Value decided by Vehicular Speed Value

The first way to overcome these disadvantages in the high interference MU-MIMO environment is to estimate the vehicular speed at the base station to reduce rank when the speed is above a certain value. At this time, if there are no other fading elements, we can select the largest value from the range for each speed like table 2.

Table 2. Optimized rank estimation value each vehicular speed

Speed	Optimized Rank Indicator
0 ~ 2km/h	1 ~ 4
2 ~ 200km/h	1 ~ 3
200km/h ~	1 ~ 2

The background of the values 2km/h and 200km/h is the result of field test. Figure.7 and Figure.8 specify field test results at low and high speed, respectively. Figure.7 shows the throughput of rank 3 is better than that of rank 4 in case over 2km/h. Also, Figure.8 shows the throughput of rank 2 is better than that of rank 3 in case over 200km/h. These are fixed rank test results because of before being applied optimized rank scheme. Each commercial test condition of Figure.7 and Figure.8 was specified in table 3. The section below introduces the second technique, which is how to estimate the vehicular speed required to apply the optimized rank scheme above at the base station.

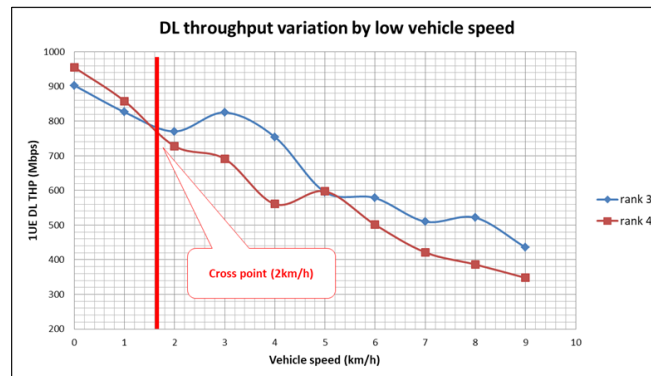


Figure 7. DL throughput variation in case of low vehicular speed

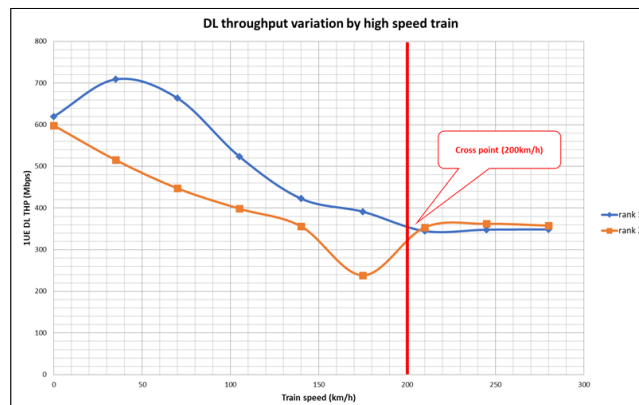


Figure 8. DL throughput variation in case of high speed train

Table 3. Test condition at commercial environment

Figure	Test condition
Figure 7, 8	LOS(Line of Sight)
	UMa(Urban Macro)
	MU-MIMO
	Rank Fixed
	Zero-forcing transmitter
	MMSE-IRC(Minimum Mean Square Estimation-Interference Rejection Combining) receiver

4.2. Speed Estimation by SRS Phase Difference

The second proposed scheme is what calculates vehicular speed by obtaining the difference between beamforming phase values and solving the distance by the pathloss estimated by the uplink SRS (Sounding Reference Signal). The procedure is as follows:

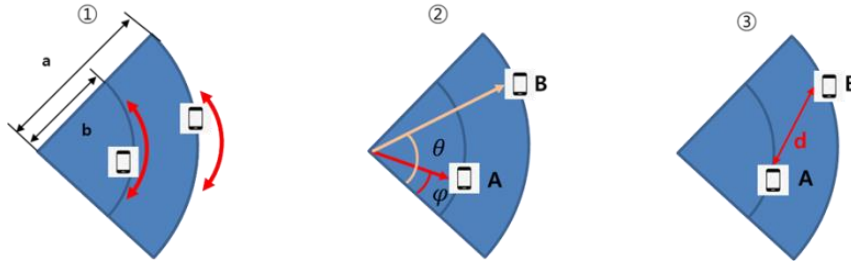


Figure 9. Speed calculation by channel estimation during SRS duration

- ① Derive the distance between the base station and the previous and subsequent points with SRS Pathloss, respectively.
- ② Derive the received beamforming angle(θ) by the SRS channel estimation.
- ③ Calculate the vehicular speed by obtaining the distance by position change per SRS long duration.

$$\text{Speed (km/h)} = \frac{d_{A-B} \text{ distance}}{\text{TAS long duration}}$$

5. PERFORMANCE COMPARISON

Figure.10 shows overall performance graph applied the new rank optimized scheme. The performance of stationary UE is similar regardless test conditions, and the performance of low speed UE has been significantly improved to 16.7% compared to the conventional method. Also, Figure.10 shows fixed low rank scheme for reference. This method was added to compare the DL performance in the stationary and mobile environment with fixed low rank scheme. In conclusion, optimized scheme performs slightly better than fixed low rank scheme and fixed low rank scheme is better than conventional scheme. However, its method has a limitation because it cannot follow the variation of channel.

Figure.11 is a commercial UE log when the conventional rank decision method is applied. Figure.12 is a commercial UE log when the new optimized rank decision scheme is applied. In Figure.11 and Figure.12, the red background is a part of the MU mode operation when the two UEs are separated from each other and the blue background is a part that SU mode operation when the distance of 2 UEs is very close to each other such as the map of Figure.13. The

condition of commercial environment is like Table 4. This mode operation transition is designed to automatically switch to the MU mode operation if the correlation coefficient is increased and the SU mode operation if the value is decreased according to the correlation coefficient value of the two UEs. If you look at the RB(Resource Block) of Figure.11 and Figure.12, the RB falls down because the frequency regions must be shared with each other if SU mode operation as an easy separation method of SU and MU operation.

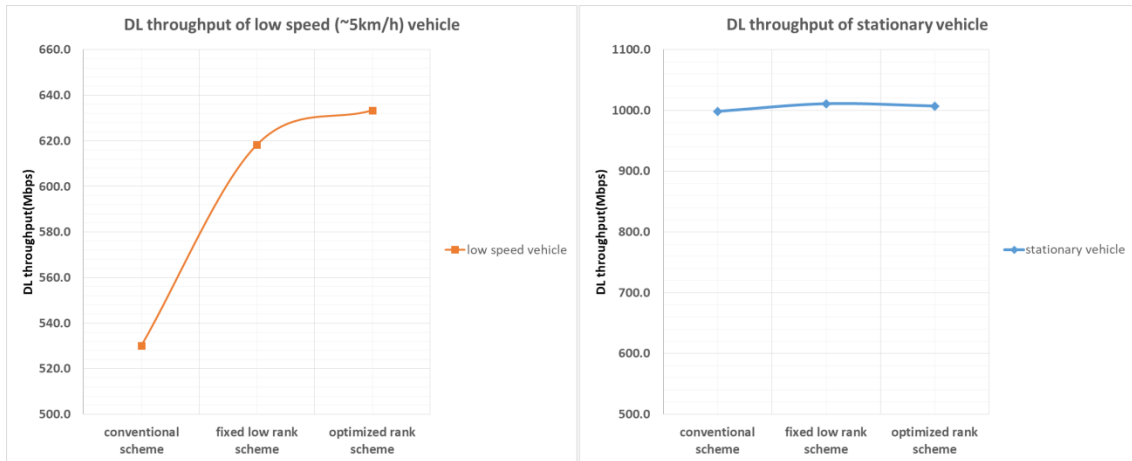


Figure 10. Overall performance comparison



Figure 11. UE's log of conventional rank decision method



Figure 12. UE's log of optimized rank decision method



Figure 13. The map of commercial test's environment in KT laboratory, Seoul

Table 4. Overall test condition of commercial environment

Parameter	Value
Morphology	UMa
MIMO	MU-MIMO, SU-MIMO
The number of UEs	2
Field environment	LOS
Maximum number of layer/UE	4
Maximum number of layer/cell	8
UE Mobility	0 ~ 20km/h
Modulation	QPSK ~ 256QAM
BS(Base Station) antenna height	20m
BS-UE distance	50 ~ 150m
Interference cancelling of transmitter	Zero-forcing
Interference cancelling of receiver	MMSE-IRC

6. CONCLUSIONS

MU-MIMO technique is one of the important and advanced technologies. Also, it is a suitable technique to increase the capacity from the point of view of solving cell capacity. However, there is a disadvantage that interference due to each multi-user beams is increased. It is important to use the advanced MU-MIMO beamforming technology to overcome these inter-beam interferences. Also, rank optimization technique is very important to increase the performance in MU-MIMO environment. However, the conventional MU-MIMO rank optimization scheme has several problems. The problem of the conventional method is slow to channel change, there are many steps, and it is only a way of relying on the RI of the UE.

This paper proposes the new first scheme to be sensitive to variation of channel. It is that the base station estimates the vehicular speed by uplink SRS. And second new scheme is that decides the optimal rank value experimentally determined suitable for the calculated vehicular speed. By these two ways, we raise the user performance as much as possible by optimal rank value.

As a result, the mobile UE was significantly improved by 16.7% compare with to the conventional method. The reason is that the conventional scheme was a method of relying only on the value that the UE reports using CSI-RS, but the new scheme was the method of quickly calculating the vehicular speed directly from the base station to respond sensitively to variation of channel and applying an optimized rank value according to the vehicular speed.

ACKNOWLEDGEMENTS

I would like to express my deep gratitude to Kyung-Yup Kim, Sang-Wook Kim, and Jae-Hyung Koo my research supervisors, for their patient guidance, enthusiastic encouragement and useful critiques of this research work.

I would also like to thank Hye-Soo Chang, for her advice and assistance in keeping my progress on schedule. My grateful thanks are also extended to Dong-Un Cha for his help in doing the data analysis, to them for their support in the site measurement.

REFERENCES

- [1] Adeel Razi, Daniel J. Ryan, Jinhong Yuan, Iain B. Collings, "Performance of Vector Perturbation Multiuser MIMO Systems over Correlated Channels", *Wireless Communications and Networking Conference (WCNC) 2010 IEEE*, pp. 1-5, 2010.
- [2] Wenbo Xu, Tao Shen, Yun Tian, Yifan Wang, Jiaru Lin, "Compressive Channel Estimation Exploiting Block Sparsity in Multi-User Massive MIMO Systems", *Wireless Communications and Networking Conference (WCNC) 2017 IEEE*, pp. 1-5, 2017.
- [3] Zhiyi Zhou, Xu Chen, Dongning Guo, Michael L. Honig, "Sparse Channel Estimation for Massive MIMO with 1-Bit Feedback Per Dimension", *Wireless Communications and Networking Conference (WCNC) 2017 IEEE*, pp. 1-6, 2017.
- [4] Yang Nan, Li Zhang, Xin Sun, "Weighted compressive sensing based uplink channel estimation for time division duplex massive multi-input multi-output systems", *Communications IET*, vol. 11, no. 3, pp. 355-361, 2017.
- [5] Ghassan Dahman, Jose Flordelis, Fredrik Tufvesson, "Experimental evaluation of the effect of BS antenna inter-element spacing on MU-MIMO separation", *Communications (ICC) 2015 IEEE International Conference on*, pp. 1685-1690, 2015.
- [6] Christian Schneider, Reiner S. Thomä, "Empirical study of higher order MIMO capacity at 2.53 GHz in urban macro cell", *Antennas and Propagation (EuCAP) 2013 7th European Conference on*, pp. 477-481, 2013.

- [7] Narendra Anand, Ryan E. Guerra, Edward W. Knightly, Proceedings of the 20th annual international conference on Mobile computing and networking, pp. 29, 2014.

AUTHOR

Senior Manager, Access Network Technology Department, Korea Telecom, Seoul, Korea



© 2022 By AIRCC Publishing Corporation. This article is published under the Creative Commons Attribution (CC BY) license.

AN EMPIRICAL STUDY OF THE PERFORMANCE OF CODE SIMILARITY IN AUTOMATIC PROGRAM REPAIR TOOL

Xingyu Zheng, Zhiqiu Huang, Yongchao Wang and Yaoshen Yu

College of Computer Science and Technology,
Nanjing University of Aeronautics and Astronautics, Nanjing, China

ABSTRACT

Recently, code similarity has been used in several automated program repair (APR) tools. It suggests that similarity has a great contribution to APR. However, code similarity is not fully utilized in some APR tools. For example, SimFix only uses structure similarity (Deckard) and name (variable and method) similarity to rank candidate code blocks that are used to extract patches and do not use similarity in patch filtering. In this paper, we combine the tool with longest common sequence (LCS) and term frequency-inverse document frequency (TFIDF) to rank candidate code blocks and filter incorrect patches. Then we design and set up a series of experiments based on the approach and collect the rank of the correct patch and time cost for each selected buggy program. In the candidate ranking phase, LCS and TFIDF improve the rank of the block, which contains the correct patch for several bugs. In the patch validation phase, LCS filters out 68% of incorrect patches on average. It shows that code similarity can greatly improve the performance of APR tools.

KEYWORDS

APR, empirical study, LCS, TFIDF.

1. INTRODUCTION

In recent years, automated program repair has become a hot research direction. APR consists of automatically finding a solution to software bugs without human intervention [1] due to the fact that debugging software failures is still a painful, time consuming, and expensive process [2]. Generally, the process of APR can be divided into three stages: fault localization, patch generation and patch validation [3]. When given a buggy program, the buggy code block is first determined in fault localization stage, and then the APR tool tries to create patches in the patch generation stage, the program is finally executed to see whether the patch is correct in patch validation stage.

Automated program repair can be simply divided into four categories: manual template (PAR [4], SketchFix [5]), semantic constraint (Angelix [6], SOSRepair [7]), statistical analysis (GenPat [8], SequenceR [9]), heuristic search (SimFix [10], CapGen [11]). Redundancy-based program repair is fundamental to APR, and it is based on the hypothesis that code may evolve from existing code that comes from somewhere else, for instance, from the program under repair [12]. When given the fault localization result, APR tools based on redundancy search code blocks that are similar to the buggy code block in the current program, the patches generated from these code blocks are then validated to find the first correct patch that passes all test cases. Recently, there have been several redundancy-based APR techniques (such as SimFix [10], CapGen [11], ssFix [13],

David C. Wyld et al. (Eds): ARIA, SIPR, SOFEA, CSEN, DSML, NLP, EDTECH, NCWC - 2022

pp. 25-36, 2022. CS & IT - CSCP 2022

DOI: 10.5121/csit.2022.121503

CRSearcher [14]). These techniques leverage different similar metrics like LCS, TFIDF, name similarity, Deckard and so on to compute the similarity between the buggy code block and the other blocks. The experimental results indicate that they all perform well in repair effectiveness. However, the techniques do not take full advantage of code similarity. On the one hand, for candidate code block ranking in patch generation stage, similar metrics used in different techniques can be combined to see if the priority operation is improved in this way. On the other hand, in patch validation stage, each of the techniques needs to validate a large number of patches generated in the last stage. We can utilize similar metrics to dynamically reduce the search space of patches before validation.

Our study of how code similarity works on the performance of APR tool takes SimFix, a state-of-the-art approach to design the experiment and analyze the results. First, we only select the buggy programs that have been successfully repaired by the tool for the following experiments. We exclude the buggy programs that the tool incorrectly repaired (the generated patch passes all test cases but is not correct) and failed to repair (fail to generate a patch that can pass all test cases or time out) for convenience. Second, we utilize two similarity metrics that capture syntactic messages —LCS and TFIDF in both patch generation stage and patch validation stage of the tool, then we design and perform a series of experiments. The main goal of the experiments is to study how good the new similarity metrics are at improving the rank of the first correct patch in the search space. For evaluating the experimental performance, we collect the rank of the first correct patch and the time cost of each buggy program before and after applying new metrics and compare the experimental data. Our experimental results are clear-cut. Firstly, LCS and TFIDF effectively rank candidate similar code blocks, improving the mean reciprocal rank (MRR) by 26%. Secondly, utilizing LCS in patch filtering successfully excludes 68% incorrect patches on average.

To sum up, the main contributions of our work are as follows:

- An empirical study of how code similarity performs in a concrete APR tool.
- When we add new code similarity metrics to improve the rank of the first correct patch for a buggy program in APR tool, the results perform well. LCS and TFIDF rank the true candidate code block higher in patch generation stage and patch validation stage, LCS filter out a large number of incorrect patches, let the correct patch be validated earlier.
- We collect the rank distributions and time cost of the buggy programs after applying new similar metrics. The results show what contributes to the huge distinction of the result of ranking and time cost and why new metrics do not perform better in some cases.

The remainder of this paper is structured as follows: The related work is given in Section II. Section III describes the research methodology. The numerical results and analysis are presented in Section IV. Finally, conclusions are drawn in Section V.

2. RELATED WORK

In this section, we discuss the redundancy-based APR tools and several empirical studies.

2.1. Redundancy-based APR

In this subsection, we will present an overview of redundancy-based automated program repair. As mentioned before, the repair process can be divided into three stages: fault localization, patch generation and patch validation. In fault localization stage, given a buggy program, APR tool identifies an ordered list of suspicious buggy blocks using approaches like Ochiai [15]. In patch

generation stage, the tool first searches for donor code blocks that are similar to the buggy code for each buggy location, then compares the buggy code and donor code to extract patches for each donor code block and get the search space of patches. In patch validation stage, the patches are tried one by one from the search space to find the first correct patch that can pass all test cases.

In particular, SimFix [10] leverages Deckard [16] and name similarity to calculate the similarity between the buggy code block and candidate code blocks in patch generation stage and exclude patches that use less frequent modifications based on offline mining of the frequency of different modifications. CapGen [11] prioritizes candidate code blocks that are contextually similar to buggy code block and rank the generated patches by integrating the fault space (suspicious value of buggy code), the modification space (replacement, insertion or deletion) and the ingredient space (context similarity) together. ssFix [13] extracts tokens from buggy code block and leverages TFIDF to search for candidate code blocks and rank them. CRSearcher [14] treats similar code search as a clone detection problem and utilizes a variant of Running Karp Rabin Greedy-String-Tiling Algorithm (RKR-GST), a token based approach to search for candidate code blocks.

2.2. Empirical Studies on APR

In this subsection, we will present recent empirical studies on automatic program repair. Mounita Asad et al. [17] analyze the impact of syntactic and semantic similarity on patch prioritization. They rank patches by integrating genealogical, variable similarity (semantic similarity) and LCS (syntactic similarity), and the results are better than using one similarity metric alone. Xiong et al. [18] perform an empirical study that identify patch correctness in test-based APR. For each test case, they leverage LCS to calculate the similarity of complete-path spectrum (the sequence of executed statement IDs) between the two executions before and after applying a patch and judge whether a patch is correct. The approach successfully filters out 56.3% of incorrect patches. Liu et al. [19] systematically investigates the repair effectiveness and efficiency of 16 APR tools in recent years. They find that fault localization plays an extremely important role in a repair process, and accurate localization can greatly reduce the number of generated patches. Chen et al. [12] define and set up a large-scale experiment based on four code similarity metrics that capture different similarities—LCS, TFIDF, Decksrd and Doc2vec. The paper considers two cases—context-less and context-aware (i.e., one-line buggy code and multi-line buggy code) and performs the experiments. According to their rank statistic, all of the 4 similar metrics reduce over 90% of the search space in both cases; at the same time, LCS and TFIDF can rank the correct patch higher than Deckard and Doc2vec in context-less cases, and the performance of the 4 metrics is close in context-aware case. Furthermore, the paper studies the feasibility of combining different metrics and points out that LCS and TFIDF can be used together in context-aware cases. Consider the great performance of LCS and TFIDF and the fact that most bugs selected in our experiment have multiple lines. We decide to utilize them in SimFix to investigate how code similarity can improve the performance of APR tool.

3. RESEARCH METHODOLOGY

In this section, we will describe our research methodology.

The overall workflow of our work is illustrated in Fig. 1. For each buggy program, the fault localization stage returns a list of buggy code block list based on their suspicious score, then the APR tool searches similar code blocks in the program for the buggy blocks, the similarity between the candidate code block and the buggy block is requested to be more than a threshold

(dynamically changed based on the number of lines of the buggy block). The searched similar code blocks will be ranked based on the similarity metrics of the APR tool. Here we apply LCS and TFIDF, and this step returns the list of ranked candidate code blocks. Patches generated from the candidates compose the search space. Different from the original process that directly validated the patches one by one, when a patch is to be validated, we first calculate whether it is highly similar to the patches that have been validated to be incorrect. If so, we exclude this patch and verify the next one; otherwise, we validate this patch and add it to the set of incorrect patches if it can not repair the bug. Finally, a correct patch is returned.

3.1. Overview of the Research Methodology

Our research methodology is as follows. In the patch generation stage, when given the list of candidate code blocks, The original tool will rank them based on structure similarity and name similarity. We first directly add LCS and TFIDF, respectively, to the original metric and collect the experiment data. Next, we combine LCS and TFIDF, execute the tool with and without name similarity and collect the data. We investigate the statistical data of the rank of the first correct patch and the time cost for each bug based on different settings and compare which condition the tool performs best.

In the patch validation phase, the tool ranks the generated patches based on three rules and excludes delete operation, and the tool includes 16 most frequent modifications (like inserting an If statement in the buggy point) to reduce the search space of patches. However, there still exists a great number of incorrect patches. Moumita Asad et al. [17] confirms that similar metrics based on syntactic message like LCS can greatly prioritize patches. We leverage LCS to filter out those incorrect patches. Since unvalidated incorrect patches are similar to validated incorrect patches and are not similar to the correct unvalidated patch, we calculate the similarity scores between the next patch to be validated and the patches which have been validated and filter out those patches that are very similar to the incorrect patch (for example, the similarity score > 0.9). We investigate the statistical data of the rank of the first correct patch and time cost and then analyze the performance of the filtering metric.

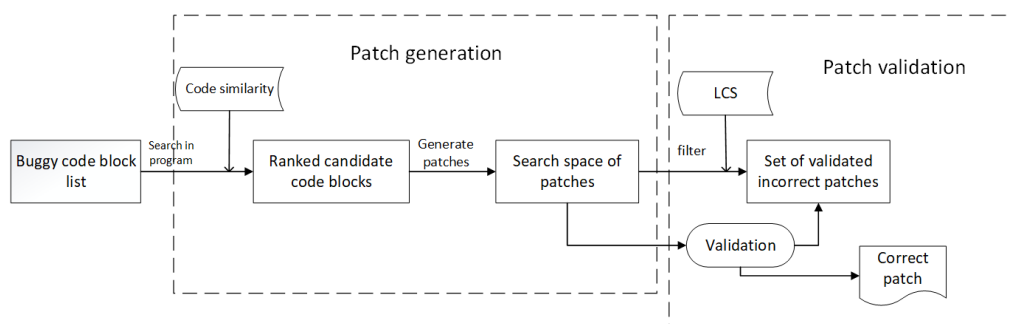


Figure 1. Illustration of our work in APR tool.

3.2. Similarity Metrics

Our core idea is to analyze how code similarity contributes to APR. The metrics used by the original tool and the metrics we use are:

- 1) Structure similarity between AST (Deckard)
- 2) Variable similarity and method similarity (Name similarity)
- 3) Longest common sequence (LCS)

4) Term frequency–inverse document frequency (TFIDF)

The four metrics are explained in detail in the following:

- 1) Deckard: Deckard is a metric that calculates similarity at AST level. It first generates feature vectors from the buggy code and candidate code blocks. Each dimension of the feature vector captures different information of the code block, such as the number of operations. Deckard then calculates cosine similarity of the feature vectors of buggy block and candidate block. SimFix uses Deckard for measuring AST similarity and selects the top 1000 at most candidates to further measure name similarity.
- 2) Name similarity: Name similarity consists of variable similarity and method similarity. Variable similarity calculates how similar the variables in two code blocks are. The tool first obtains two sets of variables from two code blocks and then calculates the similarity of the two sets using Dice's coefficient. Like variable similarity, method similarity calculates how similar the names of methods in two code blocks are, and method similarity is calculated in the same way as the variable similarity.
- 3) LCS: LCS treats the source code as a sequence of characters to calculate the longest common sequence of the code blocks. So LCS is the most syntactic metric. In the current scene, similar blocks or patches are highly consistent in the sequence of words and are, of course, highly consistent in the sequence of characters. So in our methodology, LCS is used to calculate the similarity between candidate code blocks and the buggy code block and the similarity between the next verified patch and the validated patches.
- 4) TFIDF: TFIDF, i.e., term frequency-inverse document frequency, is used to evaluate the importance of a word to a document set or one of the documents in a corpus. This means the importance of a word increases in proportion to the number of times it appears in the document but decreases in inverse proportion to the frequency of its appearance in the corpus. The similarity calculation of TFIDF is based on a token level, so TFIDF is less syntactic than LCS. TFIDF can more effectively utilize unique tokens. If we have a candidate code block that contains the same variable that only occurred once in the buggy block, it is likely that the code block is a good candidate. In our methodology, TFIDF is used to calculate the similarity between candidate code blocks and the buggy code block. We do not use TFIDF to filter incorrect patches because the set of validated patches is dynamically changing. Every time the next patch is to be validated, IDF needs to be calculated again for all the patches in the set.

4. EVALUATION

To evaluate how code similarity can improve the performance of APR tool, we design a series of experiments. Our experiment was conducted on a 64-bit Linux system with Intel(R) Core(TM) i5-6300HQ CPU and 12GB RAM, which is close to the environment of the original experiment of the tool that we study.

4.1. Research Questions

RQ1: How do LCS and TFIDF perform in candidate ranking? When given the buggy code block after fault localization, the original tool searches the current subject and gets a set of candidate code blocks based on a similarity threshold. Then the tool uses Deckard and name similarity to rank the candidates. The rank of candidate code blocks directly affects the rank of generated patches in the following validation step. The higher the true candidate ranking, the earlier the correct patch will be validated. Therefore the main purpose of RQ1 is to see how LCS and TFIDF can rank the true candidate code block which contains the correct patch over the other.

RQ2: How does LCS perform in incorrect patch filtering? After the candidate code blocks are ranked, the original tool matches the candidate code block with the buggy code block at AST level and extracts patches for validation. Due to a large amount of ranked candidate code blocks, the corresponding patch search space is enormous. For those bugs whose true candidate ranks low, a great number of incorrect patches will be validated, which has a bad effect on the tool’s performance. We use LCS to filter out patches that are similar to validated incorrect ones and see to what extent the approach will improve the rank of correct patch.

RQ3: What affect the performance of LCS and TFIDF? We repeat the original experiment of the tool in our machine and collect the result of repair time and rank of the first correct patch. Then we integrate LCS and TFIDF with the tool and get the experimental data. The result of the original experiment shows that in different bugs from different subjects, the rank of the first correct patch and the time cost is greatly distinct. the rank of the first correct patch varies from 1 to 1500, and the time cost ranges from 1 to 54 minutes. The purpose of RQ3 is to analyze the factors that influence the performance of code similarity metrics.

4.2. Data set

To evaluate the effectiveness of LCS and TFIDF, we select the 26 bugs from 4 subjects in Defects4j [20] benchmark that has been fixed by the tool and perform our experiment. We exclude 4 bugs from JFreechart (Chart) because they can not be successfully fixed on our machine. Table 1 shows statistics about the projects.

Table 1. Subjects For Evaluation

Subjects	Bugs
Closure compiler (Closure)	4
Apache commons-math (Math)	13
Apache commons-lang (Lang)	8
Joda-Time (Time)	1
Total	26

4.3. Experimental Results

We now present the result of our experiments on the performance of code similarity in APR tool.

4.3.1. Research Question 1: How do LCS and TFIDF Perform in Candidate Ranking?

We investigate the research question on the 26 bugs, and the rank of the first correct patch and time cost (in minutes) for the bugs are shown in TABLE II. In the table, column "Origin" denotes the ranking result of the original tool, column "L+T" denotes the ranking result of the combination of LCS, TFIDF and the tool, column "L+T-NS" denotes the ranking result of LCS, TFIDF and Deckard. It is nice that integrating LCS and TFIDF with the original tool in the candidate ranking phase does not change the time cost a lot whether the rank of the first correct patch is changed. So we do not need to analyze the time cost performance (in minutes) in this research question.

From TABLE II, we can see that both LCS and TFIDF successfully improve the rank of the first correct patch in many cases, and for those bugs whose correct patch has been ranked first in the tool, LCS and TFIDF do not make the result worse. We calculate the mean reciprocal rank for each setting and see that the performance of LCS and TFIDF is very close. They both improve the MRR by 10%. When combining the two metrics and integrating with the tool, the

performance is almost unchanged from the results of using the two tools alone. This may be because LCS and TFIDF both calculate similarity scores based on the syntactic message of code blocks. As shown in the L+T-NS column, we replace the name similarity metric with the combination of LCS and TFIDF and execute the program. The results show that in this setting, MRR has an improvement of 26%, which is higher than integrating all the metrics. This means LCS and TFIDF rank candidate code blocks better than name similarity. It is because name similarity only collects the variable name and method name in a code block, but TFIDF collects all tokens, i.e., all words in the block. LCS similarly calculates the scores according to all the characters in the block.

Table 2. Rank and Time Statistic in Different Settings

Bugs	Origin	LCS	TFIDF	L+T	L+T-NS	Time
Math5	1	1	1	1	1	2
Math50	8	10	9	9	4	4
Math53	2	2	2	2	2	1
Math63	16	16	14	16	9	5
Math70	1	1	1	1	1	1
Math71	14	9	9	9	9	5
Math75	8	4	8	4	4	1
Lang27	7	6	4	4	1	2
Lang41	2	1	1	1	1	10
Lang58	1	1	1	1	1	1
Closure73	1	1	1	1	1	7
Math33	55	62	62	62	61	4
Math35	5	5	5	5	5	6
Math57	112	119	113	110	111	5
Math59	9	9	9	9	9	10
Math79	562	559	559	559	559	18
Math98	212	211	212	211	212	10
Lang16	1491	1497	1494	1489	1436	51
Lang33	42	36	39	35	32	1
Lang39	1296	1316	1314	1310	1302	54
Lang43	6	6	6	6	6	7
Lang60	130	117	116	118	115	10
Time7	431	428	438	430	440	23
Closure14	344	344	344	344	347	53
Closure57	21	25	21	24	22	17
Closure115	3	3	3	3	3	6
MRR	0.248	0.274	0.273	0.277	0.313	

4.3.2. Research Question 2: How does LCS Perform in Incor Rect Patch Filtering?

We investigate the research question on the 26 bugs, and the rank of the first correct patch and time cost (in minutes) are shown in TABLE III. In the table, column "Rank(o)" denotes the ranking result of the original tool, column "Rank(f)" denotes the ranking result with incorrect patch filtering, column "Time(o)" denotes the time cost of the tool, column "Rank(f)" denotes the ranking result with incorrect patch filtering. As analyzed in Section III and RQ1, The ranking performance of LCS and TFIDF are highly similar, and calculating TFIDF in a dynamically changing set of patches is time-consuming. So we only focus on LCS. In this research question, we can see that LCS makes significant progress in filtering incorrect patches and reducing time cost.

From TABLE III, we can see that the rank of the first correct patch for some bugs is extremely low, especially for lang16 and lang39. The rank of the correct patch is over 1000. This means the vast majority of patches in the search space are incorrect. We utilize LCS to calculate the similarity scores between the next patch to be validated and the set of validated incorrect patches and exclude the related patch if one of the similarity scores exceeds a threshold. Here we set the threshold as 0.9 in common.

Table 3. Rank and Time Statistic After Filtering.

Bugs	Rank(o)	Rank(f)	Time(o)	Time(f)
Math5	1	1	2	2
Math50	8	6	4	2
Math53	2	2	1	1
Math63	16	13	5	2
Math70	1	1	1	1
Math71	14	12	5	5
Math75	8	5	1	1
Lang27	7	6	2	2
Lang41	2	1	10	4
Lang58	1	1	1	1
Closure73	1	1	7	7
Math33	55	26	4	4
Math35	5	5	6	6
Math57	112	39	5	3
Math59	9	9	10	10
Math79	562	110	18	5
Math98	212	58	10	4
Lang16	1491	354	51	15
Lang33	42	15	1	2
Lang39	1296	173	54	9
Lang43	6	6	7	7
Lang60	130	59	10	3
Time7	431	244	23	15
Closure14	344	42	53	12
Closure57	21	21	17	17
Closure115	3	2	5	5
AVERAGE	189	47	12	5.4

We can see that the ranks of the first correct patch of the bugs in TABLE III are extremely improved, and the time cost is greatly reduced. 75% improves the average rank of the first correct patch and the average time cost is improved by 55%, and the average reduction of search space is 68% for the bugs whose ranking results have changed. This means in the patch validation stage, filtering unvalidated incorrect patches greatly improves the rank of the first correct patch. For math33 and lang33, the time cost does not reduce and even increases a little. It is because the number of exclusions is in the dozens, which is inadequate to offset the time consumption of calculating similarity scores. For other bugs, the time cost reduces greatly because the time consumed in executing the tests is significantly cut down, achieving an average reduction of 56%.

4.3.3. Research Question 3: What Affect the Performance of LCS and TFIDF?

Due to the distinct experimental result of the 26 bugs from 4 subjects, we further analyze what contributes to the difference in this research question. To do this, we print out the buggy code

block list (the rank of the true buggy block of each bug is shown in TABLE IV), the ordered list of candidate code blocks and the list of generated patches in relative repair stages and investigate the collected message together with the log message of each bug printed by the original tool.

Table 4. Rank of the Buggy Code.

Bugs	Rank	Bugs	Rank
Math33	6	Math5	1
Math35	3	Math50	1
Math57	12	Math53	1
Math59	2	Math63	1
Math79	18	Math70	1
Math98	3	Math71	1
Lang16	20	Math75	1
Lang33	3	Lang27	1
Lang39	19	Lang41	1
Lang43	4	Lang58	1
Lang60	7	Closure73	1
Time7	11		
Closure14	2		
Closure57	2		
Closure115	5		

As shown in TABLE IV, the 26 bugs can be divided into two groups according to the rank of the true buggy block. On the one hand, for those bugs that the true buggy block is ranked first, the rank of the first correct patch improved when combining LCS and TFIDF with the original tool, but filtering incorrect patches did not contribute to a higher ranking. This is because these bugs' search space is too small and does not contain many similar incorrect patches. On the other hand, for those bugs where the true buggy block is not ranked first, the search space was extremely reduced by leveraging LCS to filter out incorrect patches, but leveraging LCS and TFIDF in ranking candidate blocks did not work on these bugs. This is because ranking candidate blocks for the current true buggy block can not exclude the patches generated before for the false buggy blocks. All in all, the result of fault localization and the size of search space greatly influence the performance of LCS and TFIDF.

4.3.4. Case Analysis

We now present case analyses of great rankings. We investigate the buggy code block list, the ordered list of candidate code blocks, the list of generated patches and the log message of the bugs to explain the excellent performance in these cases.


```

buggy code block:
if(expPos > -1){
    mant = str.substring(0, expPos);
}else {
    mant = str;
}

false candidate:
if(expPos > -1){
    if(expPos < decPos){
        throw new NumberFormatException(str + " is not a valid number.");
    }
    dec = str.substring(decPos+1, expPos);
} else{
    dec = str.substring(decPos+1);
}
mant = str.substring(0, decPos);

true candidate:
if(expPos > -1 && expPos < str.length()-1){
    exp = str.substring(expPos+1, str.length());
} else{
    exp = null;
}

```

Listing 1. Lang27 from Apache commons-lang

```

buggy code block;
int sum = 0;
for(int i = 0; i < pointSet.size(); i++){
    final T p = pointSet.get(i);
    final Cluster<T> nearest = getNearestCluster(resultSet, p);
    final double d = p.distanceFrom(nearest.getCenter());
    sum += d * d;
    dx2[i] = sum;
}

incorrect patch;
int sum = 0;
- for(int i = 0; i < pointSet.size(); i++){
+ for(int i = sum; i < pointSet.size(); i++){
    final T p = pointSet.get(i);
    final Cluster<T> nearest = getNearestCluster(resultSet, p);
    final double d = p.distanceFrom(nearest.getCenter());
    sum += d * d;
    dx2[i] = sum;
}

an incorrect patch similar to the incorrect patch;
int sum = 0;
- for(int i = 0; i < pointSet.size(); i++){
+ for(int i = 0; i <= pointSet.size(); i++){
    final T p = pointSet.get(i);
    final Cluster<T> nearest = getNearestCluster(resultSet, p);
    final double d = p.distanceFrom(nearest.getCenter());
    sum += d * d;
    dx2[i] = sum;
}

```

Listing 2. Math57 from Apache commons-math

1) Case analyse 1: For the bug Lang27, the correct patch is a replacement of an if statement. Combining LCS and TFIDF and excluding name similarity (setting 5, i.e., column 6 in table 2) improve the rank of the correct patch from 7 to 1 for lang27. If we do not exclude name similarity (setting 4), the rank is 4. According to the buggy block list, the true buggy code is ranked first.

This means all patches in search space are generated for this block. The original approach ranks the candidate block that contains the correct patch at 6th, but the best setting ranks it at 2nd. The tool does not extract any patch from the most similar block because it is equal to the buggy code. So the best setting successfully ranks the correct patch generated from the true candidate block first. As shown in Listing 1, the false candidate is ranked first in setting 4, and the true candidate is ranked first in setting 5. This is because the false candidate block shares more variable and method names with the buggy code block than the true candidate. This leads to a low rank of the correct block. However, the correct patch is most similar to the buggy code when treating code block as a sequence of characters or tokens, which LCS and TFIDF do. So in this case, including all similarity metrics contributes to a worse result.

2) Case analyse 2: For the bug math57, the correct patch is to replace "int" with "double". This modification is too small. Utilize LCS successfully improve the rank of the correct patch from 112 to 39, and the time cost is reduced from 5 to 3 minutes. We present the buggy code block and two incorrect patches in Listing 2. The two incorrect patches are extremely similar in character level. The approach generates a large amount number of patches like this. Filtering out these patches can greatly reduce the search space and accelerate the repairing process. And as mentioned before, we initially set the filtering threshold to 0.9. However, the APR tool failed to repair this bug because the correct patch was identified as incorrect and filtered out before validating. This is because the patches in search space are highly similar to each other due to the small size of the repair. We finally fixed the buggy when the threshold was adjusted to 0.97. The good news is that although the threshold is really high, LCS still filters out 65% incorrect patches.

5. CONCLUSION

In this paper, we design and set up a series of experiments to investigate how code similarity can improve the performance of APR tool. We study a state-of-the-art approach called SimFix to see where we can utilize code similarity and a recent empirical study to select suitable similar metrics. To rank the first correct patch higher in the search space and reduce time cost, we apply two similarity metrics—LCS and TFIDF in candidate code block ranking and incorrect patch filtering. We analyze the experimental result and get the following conclusion: First, in candidate code block ranking phase, combining LCS and TFIDF and excluding name similarity used in SimFix perform best and improve the MRR by 26%. Second, in patch filtering stage, applying LCS to calculate the similarity between the next verified patch and the set of validated patches can greatly prevent incorrect patches from being verified. The method reduces search space of patches by an average of 68% and time cost by 56%.

REFERENCES

- [1] M. Monperrus, "Automatic software repair: a bibliography," *ACM Computing Surveys (CSUR)*, vol. 51, no. 1, pp. 1–24, 2018.
- [2] L. Gazzola, D. Micucci, and L. Mariani, "Automatic software repair: A survey," *IEEE Transactions on Software Engineering*, vol. 45, no. 1, pp. 34–67, 2017.
- [3] K. Liu, L. Li, A. Koyuncu, D. Kim, Z. Liu, J. Klein, and T. F. Bissyande, "A critical review on the evaluation of automated program ' repair systems," *Journal of Systems and Software*, vol. 171, p. 110817, 2021.
- [4] D. Kim, J. Nam, J. Song, and S. Kim, "Automatic patch generation learned from human-written patches," in *2013 35th International Conference on Software Engineering (ICSE)*. IEEE, 2013, pp. 802–811.
- [5] J. Hua, M. Zhang, K. Wang, and S. Khurshid, "Towards practical program repair with on-demand candidate generation," in *Proceedings of the 40th international conference on software engineering*, 2018, pp. 12–23.

- [6] S. Mechtaev, J. Yi, and A. Roychoudhury, “Angelix: Scalable multiline program patch synthesis via symbolic analysis,” in Proceedings of the 38th international conference on software engineering, 2016, pp. 691–701.
- [7] A. Afzal, M. Motwani, K. T. Stolee, Y. Brun, and C. Le Goues, “Sosrepair: Expressive semantic search for real-world program repair,” IEEE Transactions on Software Engineering, vol. 47, no. 10, pp. 2162–2181, 2019.
- [8] J. Jiang, L. Ren, Y. Xiong, and L. Zhang, “Inferring program transformations from singular examples via big code,” in 2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE). IEEE, 2019, pp. 255–266.
- [9] Z. Chen, S. Kommrusch, M. Tufano, L.-N. Pouchet, D. Poshyvanyk, and M. Monperrus, “Sequencer: Sequence-to-sequence learning for end-to-end program repair,” IEEE Transactions on Software Engineering, vol. 47, no. 9, pp. 1943–1959, 2019.
- [10] J. Jiang, Y. Xiong, H. Zhang, Q. Gao, and X. Chen, “Shaping program repair space with existing patches and similar code,” in Proceedings of the 27th ACM SIGSOFT international symposium on software testing and analysis, 2018, pp. 298–309.
- [11] M. Wen, J. Chen, R. Wu, D. Hao, and S.-C. Cheung, “Contextaware patch generation for better automated program repair,” in 2018 IEEE/ACM 40th International Conference on Software Engineering (ICSE). IEEE, 2018, pp. 1–11.
- [12] Z. Chen and M. Monperrus, “The remarkable role of similarity in redundancy-based program repair,” arXiv preprint arXiv:1811.05703, 2018.
- [13] Q. Xin and S. P. Reiss, “Leveraging syntax-related code for automated program repair,” in 2017 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE). IEEE, 2017, pp. 660–670.
- [14] Y. Wang, Y. Chen, B. Shen, and H. Zhong, “Crsearcher: Searching code database for repairing bugs,” in Proceedings of the 9th Asia-Pacific Symposium on Internetware, 2017, pp. 1–6.
- [15] X. Xie and B. Xu, Essential Spectrum-based Fault Localization. Springer, 2021.
- [16] L. Jiang, G. Mishherghi, Z. Su, and S. Glondu, “Deckard: Scalable and accurate tree-based detection of code clones,” in 29th International Conference on Software Engineering (ICSE’07). IEEE, 2007, pp. 96–105.
- [17] M. Asad, K. K. Ganguly, and K. Sakib, “Impact analysis of syntactic and semantic similarities on patch prioritization in automated program repair,” in 2019 IEEE International Conference on Software Maintenance and Evolution (ICSME). IEEE, 2019, pp. 328–332.
- [18] Y. Xiong, X. Liu, M. Zeng, L. Zhang, and G. Huang, “Identifying patch correctness in test-based program repair,” in Proceedings of the 40th international conference on software engineering, 2018, pp. 789–799.
- [19] K. Liu, S. Wang, A. Koyuncu, K. Kim, T. F. Bissyande, D. Kim, P. Wu, J. Klein, X. Mao, and Y. L. Traon, “On the efficiency of test suite based program repair: A systematic assessment of 16 automated repair systems for java programs,” in Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering, 2020, pp. 615–627.
- [20] R. Just, D. Jalali, and M. D. Ernst, “Defects4j: A database of existing faults to enable controlled testing studies for java programs,” in Proceedings of the 2014 International Symposium on Software Testing and Analysis, 2014, pp. 437–440.

FINDMYPET: AN INTELLIGENT SYSTEM FOR INDOOR PET TRACKING AND ANALYSIS USING ARTIFICIAL INTELLIGENCE AND BIG DATA

Qinqin Guo¹ and Yu Sun²

¹Portola High School, 1001 Cadence, Irvine, CA 92618

²California State Polytechnic University, Pomona, CA, 91768, Irvine, CA 92620

ABSTRACT

Pet tracking has been an important service in the pet supply industry, as it is constantly needed by countless pet owners [1]. As of 2021, about 90 million families in the U.S. alone have a pet, that is about 70% of all American households. However, for most owners of smaller pets such as cats, hamsters, and more, not being able to find the pet within the house has been a problem bothering them. This paper proposes a tool to use Raspberry Pi for gathering signal strength data of the blue tooth devices and using Artificial Intelligence to interpret the gathered data in order to get the precise location of the indoor moving object [2]. The system is applied to arrive with the location of pets within the house to an accurate level where the room that the pet is located in is correctly predicted. A qualitative evaluation of the approach has been conducted. The results show that the intelligent system is effective at correctly locating indoor pets that are constantly moving.

KEYWORDS

Raspberry Pi, Firebase, machine learning, Artificial Intelligence (AI).

1. INTRODUCTION

It is undoubtedly an annoying and worrying experience for pet owners to mistakenly think that their pet is lost. My household has 4 cats, and a lot of times, some of them hide away under the bed or behind the couch, and I cannot find them for the whole day, so I just keep worrying if they can't get enough food or water [3]. A similar problem occurred when I was actually much younger. At the time, my pet turtle climbed out of the aquarium and was missing for months. By the time I found it under the cabinet, it was already dead. Since then, I always wanted to figure out a way to know where my pets are located all the time. Before I have started this project, I have surveyed pet owners through an online form with random sampling and random assignment, and it turned out that it is a universal concern for pet owners with small animals such as cats, turtles, and reptiles to be unable to identify where the pet is at while know it is at home. My design can help with this common problem among pet holders. With the Find My Pet system, users can quickly identify the specific location of the intended animal as long as the animal is wearing a collar with or attached to a signal emitting device such as a Bluetooth beacon [6]. Therefore, many problems can be solved. For example, cat owners do not need to worry that they cannot find the newly arrived cat who is shy to meet people and always hides away. When pets can be found in a short time, other problems such as dehydration, starvation, and even death can also be prevented.

There are existing domestic animal tracking techniques and systems, such as GPS collars, which allow the users to know the approximate location of the animal [4][5]. However, such designs are only suitable for outdoor tracking where the designated animal may move in great distances, and it rarely works for small-range accurate tracking. For example, the Air Tag collar, commonly used for cats, is a very popular product, but it was not originally designed for pet tracking; Airtags can usually be accurate up to which street and even which household the intended object is, but it is far from enough for the subject in discussion [7]. In addition, Airtag devices often make substantial sounds when being manipulated on the phone to find the lost animal, and this results in further inaccurate results because noises can often scare animals and prompt them to move around even more. Other techniques, such as inserting microchips, have always been used as an identification tool for animals; by inserting an electronic chip with pet and owner's information, the lost animals with microchips can be easily returned to home when found and scanned with a corresponding device. Nevertheless, it is not useful in an active search for the animal because the method was designed mainly to keep the information about the animal and its owner well-organized. There are also animal tracking devices that can be used for precise small range tracking, which are usually Bluetooth devices; however, because the algorithm used cannot be too sophisticated, such devices only show the user the approximate distance between the Bluetooth collar and device that is used to track the Bluetooth collar instead showing the direct location, it can not give enough useful information to locate the pet.

Our goal is to generate an intelligent system that can accurately estimate the location of indoor moving animals. There are some good features of the system. First, the signal emitter that the animal will wear is a Bluetooth device. For the hardware designing and experimenting process, we used Bluetooth iBeacons, and they are proven to be effective for small distance tracking. The iBeacons have relatively low cost and easy accessibility compared to many other similar Bluetooth products, so it is a good tool to be used for this project. Second, Our hardware part includes three Raspberry Pi. Raspberry Pi is a microcomputer that can perform many complex functions including coding, search engineering, and many more. These Raspberry Pi devices are very compatible tools for the project because they are microcomputers that can receive, process, and send signals speedily. Third, we trained artificial intelligence(AI) to interpret the received data and make reasonable and accurate results out of the given data and stored information regarding the data range of specific rooms. This AI is well-adapted for the purpose mentioned above. Last, we use the received number of signals indicator(RNSI) to track locations instead of the commonly used received signal strength indicator(RSSI) because research has found that RNSI shows more significant differences in value when the distance of the moving object from the Raspberry Pi changes more compared to RSSI [8][9]. Also, RNSI is more stable in an environment with high signal interference, where many other irrelevant Bluetooth devices may be present. Therefore, we believe that our model is strong enough to be implemented for the purpose of indoor moving pets tracking.

In the two application scenarios for our tracking system, we demonstrated how the above combination of techniques gives off a useful and accurate prediction model. First, use three BLE beacons and one Raspberry Pi to test the eligibility of the beacons-if they are sending out signals in a consistent way and could that signal be received by our receivers, or the Raspberry Pi. Second, we analyze if the system could produce a precise prediction of the pet's location by using a real pet wearing a beacon for testing. After we obtained RNSI value ranges for each of the rooms in the testing house (single floor), we put the beacon on the experimenting pet(a cat)

and placed the cat in a designated room. The results show that our prediction model is accurate. In the 10 trials that we performed, ten out of ten, or 100% of the time the prediction was accurate.

The rest of the paper is organized as follows: Section 2 gives the details on the challenges that we met during the experiment and designing the sample; Section 3 focuses on the details of our solutions corresponding to the challenges that we mentioned in Section 2; Section 4 presents the relevant details about the experiment we did, following by presenting the related work in Section 5. Finally, Section 6 gives the concluding remarks, as well as points out the future work of this project.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Using Raspberry Pi to properly carry out the process

During our hardware development stage, we encountered challenges while trying to explore how to use Raspberry Pi as a signal receiver and data storage and transferer. We had problems such as the Raspberry Pi was not updated enough for us to carry out certain processes. For example, while looking for tools to detect beacon signals in the library, we were restrained from using the library that incorporated such functions because the version of Raspberry Pi at the time was not capable of running such tools. To use beacon signal detecting tools, we had to manually update the system by running scripts in the Terminal, which took a long time but turned out to be working properly. Another problem was that the Raspberry Pi could not receive the signals sent from the beacons. When we run the script for finding the RNSI value by detecting the signals, the results sometimes turn out to show an error has occurred. This happened many times during the development of the project, especially in the experimenting stage, and we found out that this was caused by when the script started running, the signal from the beacons have not yet been sent, and the will cause the script to interpret the situation as an error and no longer receive signal. To solve this challenge, we added a 60 seconds sleep time at the beginning of the script, so as long as there are signals detected in the first 60 seconds when the scripts start running, the script will continue working and detect the signals.

2.2. Receive accurate RNSI using Raspberry Pi

The signal emitting tools in the project are Bluetooth low energy (BLE) beacons, a class of BLE devices. The BLE beacons broadcast their identifier to neighboring electronic devices through radio waves. However, beacons often do not have a very stable signal emitting pattern; rather, the RSSI may vary even when the beacon is not moved. Using RNSI instead of RSSI has helped, but not much at first because the value still does not make much sense sometimes. Fortunately, we found out that some frequencies of radio waves are more stable compared to others, so we manually adjusted the signal emitting frequency of the beacons through their mobile application. For example, when the frequency of Daisybeacons_{small} (the smallest beacon used in the experiment) is adjusted to be twice as high as before, its results turned out to be more accurate and stable: when put done at a certain place and run the programming to receive its signals for 10 times, the RNSI value tends to be the same or 1 or 2 differences from the median value.

2.3. Making an accurate prediction model using the collected data

It is very sophisticated to use machine learning to train an AI to interpret the collected RNSI and end up with the right prediction because the process involves positioning analysis, trigonometry calculations, and calibration. In addition, because there is the interference of signals, it is even harder to interpret the data. To optimize the results from the collected data, we collected a range of data in each of the rooms used for experimenting, so each room (specific area) corresponds with a range of data that consists of three RNSI values, one from each Raspberry Pi. Because the Raspberry Pis are placed in a static position, their distances to each of the rooms will remain constant, so theoretically the RNSI values from a BLE beacon received by the Raspberry Pis will be the same or with small errors as long as the beacon is placed in the same room. This way, we could predict which room is the intended pet (wearing a BLE beacon) based on the range of RNSI values for each of the rooms.

3. SOLUTION

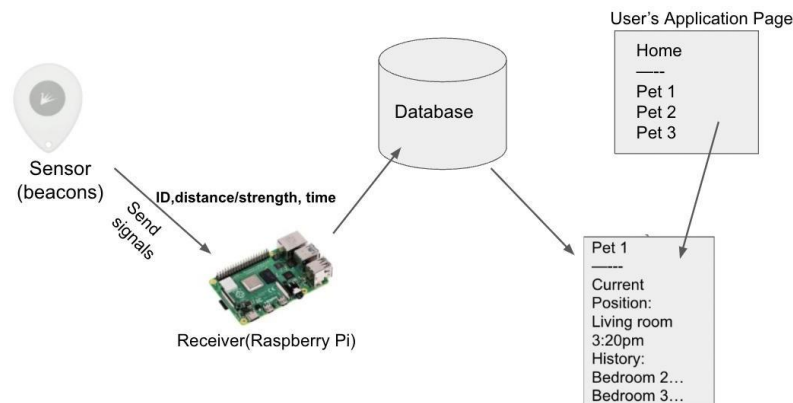
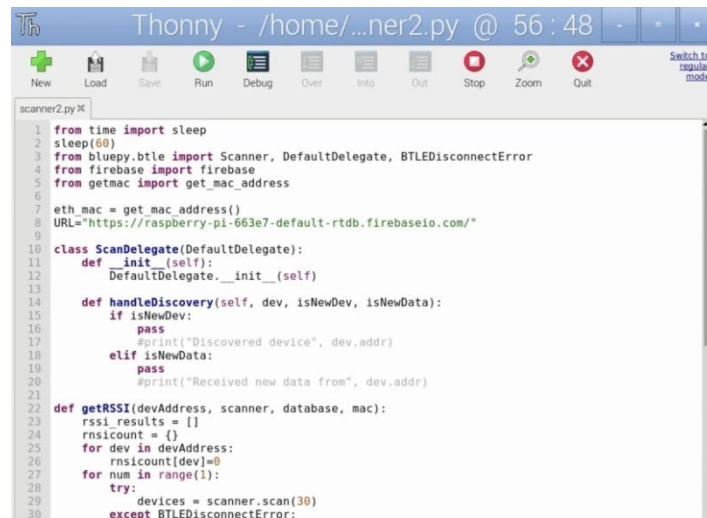


Figure 1. Overview of the solution

FindMyPet is a pet tracking system that is mainly composed of these steps: sending a signal, receiving the signal, processing the signal into useful data, storing data, and giving results based on the perceived data. The sensors are responsible for sending out the signal, they are low-energy Bluetooth (BLE) beacons. These beacons are usually small in size and wearable for animals in a variety of sizes. After the beacons send out radio signals, Raspberry Pis will receive the signals accordingly [14]. The programming language that we use in Raspberry Pi is python, and we programmed the three Raspberry Pis to receive signals every interval of one minute. After receiving signals, Raspberry Pis are programmed to send the RNSI value to Google Firebase where our data is stored and replaced with new data every minute. Then, our Replit program will retrieve the data from Firebase and calibrate it to make a prediction of the location of the pet given that certain values of RNSI from each Raspberry Pi correspond to a specific room of the household. Lastly, the same results that are shown on our Replit website will be shown on the user's application that we developed for this tracking system. When the user clicks "Where is My Pet" on the application side, the whole system of programming will run and give the most recent update to the pet's location. The next section will discuss the details of each of the components mentioned in the project overview.

1. Code on Raspberry Pi-Thonny



```

1 from time import sleep
2 sleep(60)
3 from bluepy.btle import Scanner, DefaultDelegate, BTLEDisconnectError
4 from firebase import firebase
5 from getmac import get_mac_address
6
7 eth_mac = get_mac_address()
8 URL="https://raspberrypi-663e7-default-rtdb.firebaseio.com/"
9
10 class ScanDelegate(DefaultDelegate):
11     def __init__(self):
12         DefaultDelegate.__init__(self)
13
14     def handleDiscovery(self, dev, isNewDev, isNewData):
15         if isNewDev:
16             pass
17             #print("Discovered device", dev.addr)
18         elif isNewData:
19             pass
20             #print("Received new data from", dev.addr)
21
22 def getRSSI(devAddress, scanner, database, mac):
23     rssi_results = []
24     rnsicount = {}
25     for dev in devAddress:
26         rnsicount[dev]=0
27     for num in range(1):
28         try:
29             devices = scanner.scan(30)
30         except BTLEDisconnectError:

```

Figure 2. Screenshot of Thonny

On Raspberry Pis, we use python in Thonny to set up the program for receiving signals from BLE beacons and sending RNSI data to Firebase [10]. We added 60 seconds of sleep time before each round of getting data so we would not miss the signals of the beacons when they are just turned on. We imported packages from different python libraries to help us. For example, we import “Firebase” from the “Firebase” library and we import multiple tools from “bluepy.btle”.

2. Data on Firebase

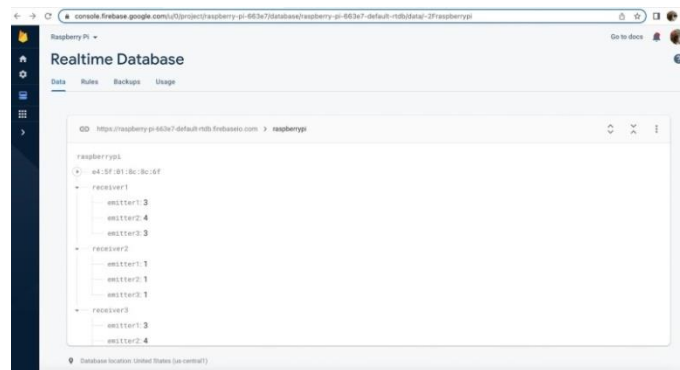
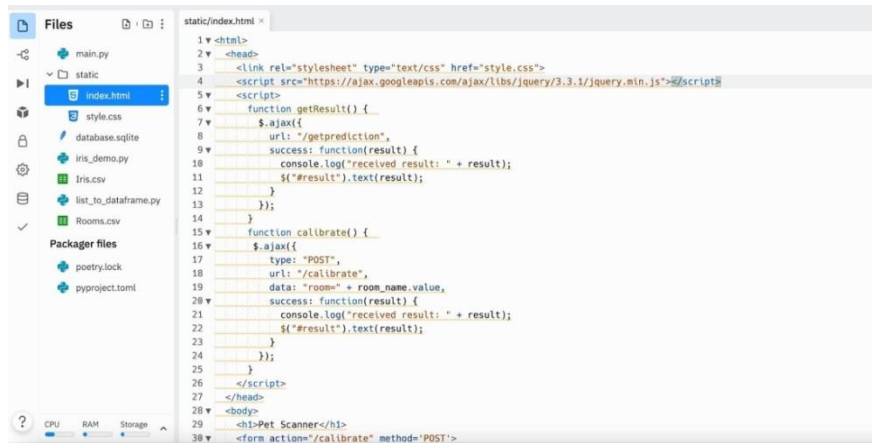


Figure 3. Screenshot of Database

Through Firebase, we receive the RNSI data from the Raspberry Pi. We store and manage the data on this database. The database is named “raspberrypi”, and sections under it include receiver 1, receiver 2, and receiver 3, and each has three components of emitter 1, emitter 2, and emitter 3 under them. Raspberry Pi is the receiver and BLE beacons are the emitters. The URL of this database is incorporated into the program in Replit, where the RNSI data will be transferred when the program starts calibrating and making predictions.

3. Code on Replit



```

1 <html>
2 <head>
3 <link rel="stylesheet" type="text/css" href="style.css">
4 <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
5 <script>
6 function getResult() {
7   $.ajax({
8     url: "/getprediction",
9     success: function(result) {
10      console.log("received result: " + result);
11      $("#result").text(result);
12    }
13  });
14 }
15 function calibrate() {
16   $.ajax({
17     type: "POST",
18     url: "/calibrate",
19     data: "room=" + room_name.value,
20     success: function(result) {
21       console.log("received result: " + result);
22       $("#result").text(result);
23     }
24   });
25 }
26 </script>
27 </head>
28 <body>
29 <h1>Pet Scanner</h1>
30 <form action="/calibrate" method="POST">

```

Figure 4. Code on Replit 1

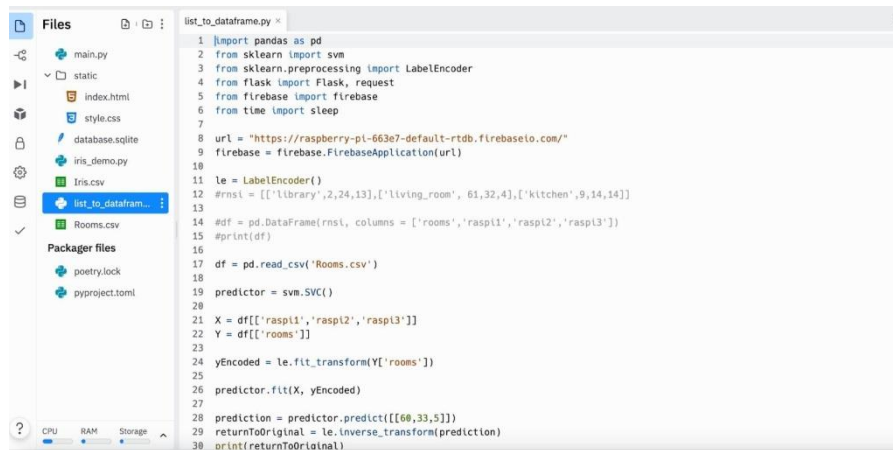


```

1 import pandas as pd
2 from sklearn import svm
3 from sklearn.preprocessing import LabelEncoder
4
5 le = LabelEncoder()
6 df = pd.read_csv("Iris.csv")
7 #print(df)
8
9 X = df[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']]
10 Y = df[['Species']]
11
12 print(X)
13
14 print(Y)
15
16 yEncoded = le.fit_transform(Y['Species'])
17
18 irisPredictionModel = svm.SVC()
19
20 irisPredictionModel.fit(X, yEncoded)
21
22 prediction = irisPredictionModel.predict([[4.9, 3, 1.4, .2]])
23 returnToOriginal = le.inverse_transform(prediction)
24 print(returnToOriginal)

```

Figure 5. Code on Replit 2



```

1 import pandas as pd
2 from sklearn import svm
3 from sklearn.preprocessing import LabelEncoder
4 from flask import Flask, request
5 from firebase import firebase
6 from time import sleep
7
8 url = "https://raspberrypi-663e7-default-rtdb.firebaseio.com/"
9 firebase = firebase.FirebaseApplication(url)
10
11 le = LabelEncoder()
12 #rnsi = [['library', 2, 24, 13], ['living_room', 61, 32, 4], ['kitchen', 9, 14, 14]]
13
14 #df = pd.DataFrame(rnsi, columns = ['rooms', 'raspl1', 'raspl2', 'raspl3'])
15 #print(df)
16
17 df = pd.read_csv("Rooms.csv")
18
19 predictor = svm.SVC()
20
21 X = df[['raspl1', 'raspl2', 'raspl3']]
22 Y = df[['rooms']]
23
24 yEncoded = le.fit_transform(Y['rooms'])
25
26 predictor.fit(X, yEncoded)
27
28 prediction = predictor.predict([[66, 33, 5]])
29 returnToOriginal = le.inverse_transform(prediction)
30 print(returnToOriginal)

```

Figure 6. Code on Replit 3

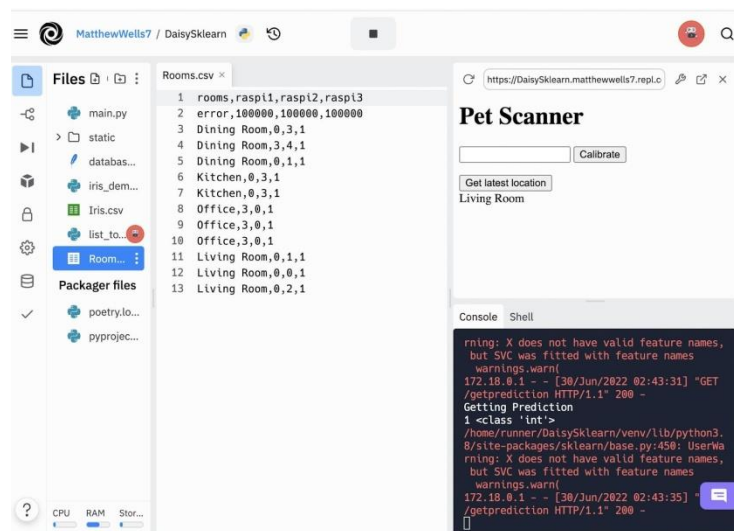


Figure 7. Code on Replit 4

As Figure 4 shows, in “index.html”, we created the web page for scanning. There are basic functions of calibrating, getting predictions, and receiving results. The name of the web is “Pet Scanner”. First, we scan a range of RNSI values for the rooms, which is an input to the calibration. The “getResult” button allows us to get the latest location of the signal emitter, which is the pet wearing the BLE beacon. In Figure 5, here is the code of the package we installed. We use “svm” from the “sklean” library and “LableEncoder” from “sklearn.preprocessing”. We use the Iris.csv package for testing the program. This package allows a function to take in different variables and make a prediction based on the variable ranges. Figure 6 shows “list_to_dataframe.py”. We use the data frame to receive and organize the RNSI data into tables with a range of values corresponding to each of the rooms in the testing area. Figure 7 shows the HTML webpage of the control page with the buttons.

4. Code On Kivy



Figure 8. Code on Kivy 1



```

petfinder.kv - C:/Users/wells/petfinder.kv (3.10.5)
File Edit Format Run Options Window Help
<ControlPage>:
Label:
    text: Welcome to PetFinder
    text_size: self.width-30, self.height-30
TextInput:
    id: room_name
Button:
    id: calibrate
    on_press: root.calibration()
Button:
    id: locate
    on_press: root.locate_pet()
Label:
    id: output

```

Figure 9. Code on Kivy 2

We use Kivy to create a desktop user interface and mobile application for the PetScanner users. The application consists of basic functions and buttons. On the control page, we have a calibration button and a pet button. The output will be shown on the same page.

4. EXPERIMENT

4.1. Experiment 1

Design Experiment: In the first experiment, we want to prove the eligibility of the BLE beacons that are used in the project. To do so, we used three beacons and one Raspberry Pi (#1, or receiver 1). The goal of the experiment is to show that all three beacons are working properly (sending signals). There are ten trials, which is enough sampling. In each trial, we placed the three beacons near each other in one room/one area of the testing area, and moved them from place to place each trial.

Trial	Emitter 1	Emitter 2	Emitter 3
1	2	2	2
2	6	8	7
3	9	8	8
4	10	13	11
5	4	4	4
6	5	8	5
7	10	17	10
8	4	7	5
9	6	6	6
10	5	7	6

Figure 10. Experiment 1 trial and results

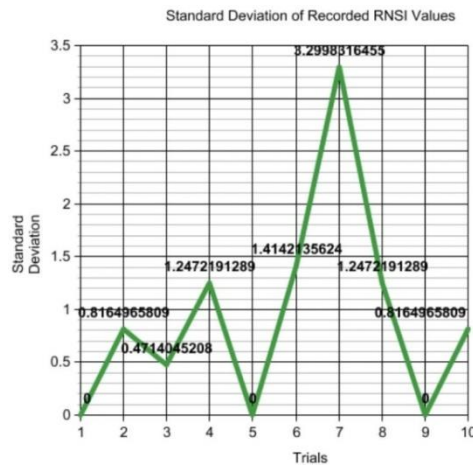


Figure 11. Result of experiment 1

Summary: The results of the experiment shows that the three beacons are functioning properly. The three beacons are each assigned a column, and they are named in this case, Emitter 1, Emitter 2, and Emitter 3. All three of the beacons are sending out signals that are receivable by the Raspberry Pi. In each trial, the number below each of the emitter columns is the corresponding RNSI value of the beacon in that trial. For trial 1, 5, and 9, all three emitters resulted in the same RNSI value, and in other cases, the RNSI value differences for emitters in the same trial range from 1-7, and this shows that the beacons are consistent in sending signals. There is also a graph showing the standard deviation of the standard deviation(SD) values for each trial, with the SD values ranging between 0 to 3.2998316455. The SD values are relatively small, and this means that the RNSI values are clustered around the mean value, or they are less spread out.

4.2. Experiment 2

Design Experiment: In the second experiment, we used the same beacon/same pet to test at the same location for 10 times to test the accuracy of the prediction model. In this experiment, a real pet, in this case, a cat, wore a collar with a BLE beacon, and was released to move freely around the testing area (the house that the cat usually lives in). After 10 minutes, we started to run the program to get predictions regarding the cat's location. We consistently run the program for 10 times in a row in a short time, to see if the results are consistent (same), and in the end, if they are accurate.

Dining Room,0,3,1	Dining Room,1,3,1
Dining Room,3,4,1	Dining Room,1,3,1
Dining Room,0,1,1	Dining Room,2,2,1

(Example of calibration for a room)

Figure 12. Experiment 2 examples

Trial	Result
1	Dining Room
2	Dining Room
3	Dining Room
4	Dining Room
5	Dining Room
6	Dining Room
7	Dining Room
8	Dining Room
9	Dining Room
10	Dining Room

Figure 13. Result of experiment 2

The order is arranged in: room name, rasp 1, rasp 2, rasp 3, where each Raspberry Pi's RNSI value is shown. The above figure shows the RNSI values that are calibrated to correspond with the room "Dining Room". It is stored and updated in Firebase and shown in Replit. As the figure shows, there is a set of values that correspond with this room (note: RNSI values are only shown in integers). We performed over 20 calibrations for each of the rooms we used in the experiment, and the example shows that of the Dining Room. After getting the RNSI value for each room, we placed the pet with the beacon collar in the Dining Room and ran the program to see if the prediction would be accurate. The results turned out to be 100% accurate, with ten out of ten predictions to be "Dining Room", which is where the pet with the beacon is actually located.

The experiment results for both experiments show that our system is functioning according to our expectations. In the first experiment, we have shown that the Beacons are consistently working and sending signals, and when three beacons are placed together, they will have similar RNSI values. This shows that the challenge of inconsistent RNSI value is partially solved, that at least we know the three beacons are responding with signals and the signals are properly caught by the Raspberry Pi. In the second experiment, the results show that the calibration is functioning well, and the artificial intelligence used for prediction is also very accurate. This indicates that our algorithm is useful at indoor location prediction and it was a good choice to use RNSI instead of RSSI because RNSI gives accurate results according to our experiment. The challenges have been mostly solved because Raspberry Pi is carrying out the process correctly, the RNSI values are producing meaningful results, and our prediction model using the RNSI data is giving accurate predictions.

5. RELATED WORK

In the research paper "tracking a moving user in indoor environments using Bluetooth low energy beacons", the researchers discussed their approach of using RNSI-based location tracking system instead of the commonly used RSSI-based tracking approach and proved that RNSI has a more accurate reflection of the object's location in an indoor location compared to RSSI [11]. The researchers want to use the results to further the study of tracking human movements, especially in the setting of healthcare locations where high levels of signal interference are present and the environment is very dynamic. The big difference between the research of the aforementioned

paper and our research is that our research focuses on the use of RNSI-based location tracking systems for animal tracking in common households, where the area is usually significantly smaller than in healthcare places (where the researchers of the other paper did their experiments) and has less signal interference factors.

In the research paper "Protection of the Child/Elderly/Disabled/Pet by Smart and Intelligent GSM and GPS based Automatic Tracking and Alert System", researchers have developed a tracking system where they use the existing GSM network and GPS satellites [12]. This approach focuses on such a method because it is possible to implement their system on a large scale at a relatively low cost compared to many other tracking approaches. Both this research and our research is aimed at developing tracking systems for moving individuals, whether it's for animals or humans, or even vehicles. The big difference between our work is that the aforementioned research emphasized large range tracking while mine focused on small range tracking. For example, their paper discusses the purpose of their research, which includes preventing the kidnapping of children, loss of soldiers, cognitive difficulties for elders and mentally disabled people, and more.

In "Detepet Mobile Application for Pet Tracking", researchers from Bina Nusantara University discussed their new application developed for pet tracking and extended pet care services [13]. Their application allows GPS tracking of pets with GPS collars, forums to post lost pet information, an online pet supply store, and information for pet-related events. Their tracking system is very mature and has the extended function of keeping track of the footprints of animals; by calculating the footprints using the size of the animal, the owner can know more about the health status of the pets by knowing how much the pets have exercised. This system is intended for large range animal tracking to prevent the animal loss, while it could work at a small range, though it was not designed for small range indoor tracking. My system is designed and calibrated for accurate small-scale indoor environment pet tracking.

6. CONCLUSIONS

In our work, we designed a system named "FindMyPet" to make accurate indoor location predictions for moving objects (primarily pets) using RNSI values interpreted by artificial intelligence. The main components of the system include BLE beacons, Raspberry Pis, Firebase, Replit, and Kivy. The beacons are responsible for sending out signals and Raspberry Pis are responsible for receiving the signals and making them into RNSI values. The RNSI value is calculated through artificial intelligence in Raspberry Pi and is sent to Firebase to be stored and managed [15]. Then, the RNSI data will be used by the python program on Replit for calibration and calculating the location. In the end, users of the system can access the system through a mobile application or desktop user interface. We performed two experiments using the established system: 1. using one Raspberry Pi and three beacons to test the eligibility of the beacons. 2. Having a moving pet wearing the beacon to test the accuracy of the current prediction model. Both experiment results indicate that the system is effective and have solved the major challenges. The beacons are working properly and the prediction system is mostly accurate at indoor tracking.

Currently, there are still a few limitations regarding this system. First, in households with multiple electronic devices, the RNSI values may be interfered with to produce inaccurate results. Common objects such as microwaves, which can release electromagnetic radiation, can cause significant flaws in receiving signals from beacons. Second, the current system is moderately complicated to set up. If it is sold as a product in the future on the pet service market, it might cause some difficulties for the users because in order to use the system correctly, the user needs to first record the RNSI value range for each of the rooms properly and store in the

administration section of the APP, and technical difficulties in setup may occur. Lastly, the current system only works with single-floor households because it uses three Raspberry Pi and trigonometry analysis. For multi-floor households, the application of this system is largely limited.

There are possible future works that could solve the current difficulties. To reduce as much signal interference as possible, we could test a wider range of signal frequencies. Signal frequencies that are significantly different will make it easier for the receivers to filter out the untargeted signals. If the system is too hard to set up, we could offer straightforward instructions and customer service to help. Very importantly, by using four Raspberry Pi instead of three, we could achieve pet tracking at multi-floor households, and that is something we plan to start in the near future.

REFERENCES

- [1] Lin, Cindy Xide, et al. "Pet: a statistical model for popular events tracking in social communities." Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining. 2010.
- [2] McCarthy, John. "What is artificial intelligence." URL: <http://www-formal.stanford.edu/jmc/whatisai.html> (2004).
- [3] Blouin, David D. "Understanding relations between people and their pets." *Sociology Compass* 6.11(2012): 856-869.
- [4] Johnson, Chris J., Douglas C. Heard, and Katherine L. Parker. "Expectations and realities of GPS animallocation collars: results of three years in the field." *Wildlife Biology* 8.2 (2002): 153-159.
- [5] Cagnacci, Francesca, and Ferdinando Urbano. "Managing wildlife: a spatial information system for GPScollars data." *Environmental Modelling & Software* 23.7 (2008): 957-959.
- [6] Chawathe, Sudarshan S. "Beacon placement for indoor localization using bluetooth." 2008 11th International IEEE Conference on Intelligent Transportation Systems. IEEE, 2008.
- [7] Haskell-Dowland, Paul. "Remember, Apple AirTags and 'Find My' app only work because of a vast, largely covert tracking network." *The Conversation* (2021).
- [8] Parker, Ryan, and Shahrokh Valaee. "Vehicular node localization using received-signal-strength indicator." *IEEE Transactions on Vehicular Technology* 56.6 (2007): 3371-3380.
- [9] Liu, Chong, Kui Wu, and Tian He. "Sensor localization with ring overlapping based on comparison of received signal strength indicator." 2004 IEEE International Conference on Mobile Ad-hoc and Sensor Systems (IEEE Cat. No. 04EX975). IEEE, 2004.
- [10] Annamaa, Aivar. "Introducing Thonny, a Python IDE for learning programming." Proceedings of the 15th Koli Calling Conference on Computing Education Research. 2015.
- [11] Surian, Didi, et al. "Tracking a moving user in indoor environments using Bluetooth low energy beacons." *Journal of Biomedical Informatics* 98 (2019): 103288.
- [12] Punetha, Deepak, and Vartika Mehta. "Protection of the child/elderly/disabled/pet by smart and intelligent GSM and GPS based automatic tracking and alert system." 2014 International conference on advances in computing, communications and informatics (ICACCI). IEEE, 2014.
- [13] Aqraldo, Brian Wijaya, et al. "Detepet mobile application for pet tracking." 2021 International Conference on Emerging Smart Computing and Informatics (ESCI). IEEE, 2021.
- [14] Parsons, J. D., and A. M. D. Turkmani. "Characterisation of mobile radio signals: model description." *IEE Proceedings I (Communications, Speech and Vision)* 138.6 (1991): 549-556.
- [15] Moroney, Laurence. "The firebase realtime database." *The Definitive Guide to Firebase*. Apress, Berkeley, CA, 2017. 51-71.

REVIEW ON DEEP LEARNING TECHNIQUES FOR UNDERWATER OBJECT DETECTION

Radhwan Adnan Dakhil and Ali Retha Hasoon Khayeat

Department of Computer Science, University of Kerbala, Karbala, Iraq

ABSTRACT

Repair and maintenance of underwater structures as well as marine science rely heavily on the results of underwater object detection, which is a crucial part of the image processing workflow. Although many computer vision-based approaches have been presented, no one has yet developed a system that reliably and accurately detects and categorizes objects and animals found in the deep sea. This is largely due to obstacles that scatter and absorb light in an underwater setting. With the introduction of deep learning, scientists have been able to address a wide range of issues, including safeguarding the marine ecosystem, saving lives in an emergency, preventing underwater disasters, and detecting, sporing, and identifying underwater targets. However, the benefits and drawbacks of these deep learning systems remain unknown. Therefore, the purpose of this article is to provide an overview of the dataset that has been utilized in underwater object detection and to present a discussion of the advantages and disadvantages of the algorithms employed for this purpose.

KEYWORDS

Underwater Object Detection, Deep Learning, Convolutional Neural Network (CNN), Underwater Imaging.

1. INTRODUCTION

Algorithms for accurately detecting and recognizing objects in images and real-world data are used for tasks such as tracking the location, motion, and orientation of objects. For an object to be detected and recognized, the algorithm must determine whether or not an object or objects are present. Object detection is "the process of accurately identifying an object, localizing that object inside an image, and performing semantic or instance segmentation [1]. The problem statement for object detection is to figure out where objects are in an image (called "object localization") and what class each point belongs to (object classification). Object classification, selection of an informative region, and feature extraction are the three main components that comprise the pipeline of traditional object detection models.

- 1) Selecting Informative Regions – Since objects can appear anywhere in the frame and in various sizes, it makes sense to deploy a sliding window with many scales to search the full image.
- 2) Extracting Features – Identifying a variety of objects requires the extraction of visual features that can provide a semantic and robust representation. Representative features include SIFT [2], HOG [3], and Haar-like [4]. As a result of their ability to produce representations associated with complex brain cells [2], these features are important.
- 3) Classification – In addition to differentiating a target object from all other categories, a classifier is required to make representations more hierarchical, semantic, and

informative for visual identification. Among the many choices available, the Supported Vector Machine (SVM) [5], the AdaBoost [6], and the Deformable Part-based Model (DPM) [7] are commonly suggested.

Object recognition of computer vision is a method to determine the identity of an object was seen in still images or moving videos. It involves distinguishing between two targets that are extremely similar as well as between one, two, or even more types of targets that are depicted in an image. Object recognition's ultimate objectives are to first recognize objects within an image in the same way that humans do and then to train a computer to acquire some level of image comprehension. The same object can be recognized when viewed from a variety of perspectives, including front, rear and side views. Additionally, the object can be identified whether it is a different size or when there is some obstruction between the viewer and the object [8]. In recent years, numerous object recognition tasks, such as handwriting [9, 10], license plate recognition [11], speech recognition [12], lane line recognition [13], face recognition [14], ship and military object recognition [15, 16], fish and underwater creature recognition [17, 18], etc., have been the subject of extensive research. Even though the oceans occupy approximately two-thirds of the globe, relatively few technologies related to marine research have been investigated to a sufficient degree [19, 20]. Feature extraction and classification are the two essential phases that comprise marine object recognition from a practical standpoint. Nevertheless, feature extraction is the more important of the two steps. The processes of pre-processing, feature extraction, feature selection, modeling, matching, and positioning are all included in object recognition [21].

Recently, deep learning, also known as deep machine learning or deep structured learning-based techniques, has seen significant success in digital image processing for object recognition and categorization. Consequently, they are rapidly becoming a focal point of interest among computer vision scientists. There has been a significant rise in the use of digital imaging for tracking marine environments like seagrass beds. As a result, automatic detection and classification based on deep neural networks have become more important tools.

Deep learning's ability to process large amounts of data has the potential to provide solutions to a number of issues pertaining to the marine industry, including marine disaster prevention and mitigation, ecological environmental protection, emergency rescue, and underwater target detection, tracking, and recognition, to mention few of. There are a number of factors that could explain deep learning's comeback, including those listed below:

- The introduction of large-scale annotated training data, such as those provided by ImageNet [22], to display fully its very vast learning capability;
- Accelerated development of high-performance parallel computing systems, such as GPU clusters; and
- Substantial progress made in the development of various network architectures and instructional methods.

The primary contributions of this paper are as follows:

- 1) A detailed discussion of the most widely used methods and deep network architectures for the analysis of underwater targets
- 2) Large collections of underwater images and video recordings being compiled and studied extensively
- 3) A full review and comparison of experiments with different deep learning methods for the detection and recognition of marine objects
- 4) Deep learning techniques being used to discuss in depth future trends and possible challenges in recognizing marine objects.

The remainder of this paper is organized as follows: Section 2 presents a Review of Traditional Object Detection Methods that have been used. In Section 3, typical deep learning methods together with comprehensive comparisons are systematically presented. Popular datasets are revisited in Section 4. Previous research methods are discussed in Section 5 and the conclusions are drawn and presented in Section 6.

2. REVIEW OF TRADITIONAL OBJECT DETECTION METHODS

The Viola-Jones object detection framework was proposed in 2001 [23, 24]. This framework for face detection is based on the AdaBoost algorithm [25] and uses Haar-like wavelet characteristics and integral graph technology. The combination of Haar and AdaBoost had not hitherto been used in a detection approach. Moreover, it is the first detection framework to operate in real-time. The Viola-Jones detector has been widely used as a foundation for face identification algorithms [26, 27] prior to the development of deep learning technology.

The histogram is computed using the gradient instead of the color value in Histogram of Oriented Gradient (HOG) [3]. The feature is built by computing the local gradient direction histogram of the image. Image recognition applications have made extensive use of HOG features in conjunction with SVM classifiers, particularly for the purpose of pedestrian identification [3]. The invariant histograms of oriented gradients (Ri HOG) [37] use cells of an annular spatial binning type and the radial gradient transform (RGT) to produce gradient binning invariance for feature descriptors, and this is only one example of many related studies. The detection concepts of enhanced HOG, support vector machine classifier, and sliding window are all incorporated into the DPM [29] algorithm, which uses a multicomponent approach to solve the target's multiview problem. In order to address the issue of target deformation, it uses a component model technique with a graphical representation of the target. DPM is a detection method that relies on individual components and has high robustness against target deformation. DPM is the backbone of several deep learning-based algorithms for tasks such as classification, segmentation, posture estimation, etc. [30, 31].

Machine learning-based object detection techniques still have advantages in certain use cases. Data from images were chunked and encoded as vectors in [32]. Sub-features are taken from the color and texture of the images and are then added together to form a feature vector. The use of the Random Forest technique resulted in a classification accuracy of 99.62 percent. By using a 1 master + 4 workers clustering design in Apache Spark, the execution time of each method was accelerated on average by a factor of 3.40.

3. DEEP LEARNING-BASED OBJECT DETECTION

We will now investigate various popular state-of-the-art CNN architectures. The convolution layer, the sub-sampling layer, the dense layers, and the soft-max layer form the backbone of the majority of deep convolutional neural networks. The architectures typically consist of stacks of multiple convolutional layers and max-pooling layers followed by fully linked and SoftMax layers at the end. LeNet [33], AlexNet [34], VGG Net [35], NiN [36], and all convolutional (also Conv) [37] are all instances of such models. Other potentially more effective advanced architectures have also been proposed. These include GoogLeNet with Inception units [38, 39], Residual Networks [40], DenseNet [41] and FractalNet [42]. Most of the fundamental building blocks (convolution and pooling) are shared by these many designs. However, newer deep learning architectures have been found to have some topological 'quirks' of their own. In terms of state-of-the-art performance on various benchmarks for object identification tasks, the DCNN

designs, namely AlexNet [34], VGG [35], GoogLeNet [38, 39], Dense CNN [41], and FractalNet [42], are widely considered to be the most popular architectures. Some of these architectures (such as GoogLeNet and ResNet) are tailored specifically for processing massive amounts of data, while others (such as the VGG network) are more general in nature. DenseNet [41] is one of the architectures that have a high density of connections. Alternatively, for ResNet, one might try the more flexible Fractal Network.

4. DATASETS

Due to the fact that underwater image processing is a relatively new field of study, only a small number of datasets are available for use in underwater computer vision [43]. The following are some of the most important reasons for the small number:

- 1) Due to a late start in the field, sufficient attention has not been devoted to the relevant underwater image datasets.
- 2) Although academic researchers have recently begun to recognize the value of an underwater image collection, creating such a dataset is laborious and time-consuming due to the unique challenges presented by the ocean environment.
- 3) The underwater world is incredibly diverse, making manual collection and classification of ground truths for a wide range of underwater images difficult.

Table 1. Review of some existing databases that can be made available to the general public for underwater object detection.

Database Name	Introduction
Underwater Image Enhancement Benchmark (UIEB) [44]	There are 950 genuine underwater images in the UIEB, of which 890 have associated references and 60 do not. The academic goal is to improve underwater images for academic purposes.
Marine Underwater Environment Database (MUED) [43]	430 various classes of interesting objects are represented in MUED's 8,600 underwater images, which vary in stance, position, illumination, turbidity of the water, and more. The academic goal is saliency detection and object recognition in underwater images
Real-time Underwater Image Enhancement (RUIE) Dataset [45]	Over 4,000 underwater real images are included in RUIE's Underwater Image Quality Sub-aggregate, Underwater Color Cast Sub-aggregate, and Underwater higher-level task-driven Sub-aggregate. The academic goal has focused on improving underwater images and finding objects in them.
The TrashCan dataset [46]	This dataset includes observations of trash, remotely operated vehicles (ROVs), and a diverse range of marine life, all cataloged in a database of annotated images (7,212 images as of this publishing). Instance segmentation annotations are used to label which pixels in the image correspond to which objects in this dataset. collected from a variety of sources.
UOT32 (Underwater Object Tracking) Dataset [47]	The benchmark dataset for underwater tracking has 32 videos with a total of 24,241 annotated frames and an average duration of 29.15 seconds and frame count of 757.53. sequences for objects of interest.

SUIM Dataset [48]	This is the first comprehensive dataset for underwater image semantic segmentation (SUIM). Fish (vertebrates), reefs (invertebrates), aquatic plants, wrecks/ruins, human divers, robots, and the seafloor are only a few of the eight object categories covered by more than 1,500 images with pixel annotations. Participants in oceanographic expeditions and human-robot cooperation studies capture and meticulously annotate the images.
SeabedObjects-KLSG [49]	A real side-scan sonar image dataset called SeabedObjects-KLSG can be used to identify wrecks, drowning victims, airplanes, mines, and the seafloor. This was done in an effort expeditiously to promote underwater object classification in side-scan sonar images, especially civilian object classification.
Fish4K [50]	The resource is referred to as a resource since it comprises sample images of 23 different species. These images are mainly free of noise; however, most are out of focus.
Kyutech-10K [51]	This is the first dataset of deep-sea marine organisms provided by the Japan Agency for Marine-Earth Science and Technology (JAMES).

Figure 1 shows a subset of the 890 identical pairs of original underwater images and reference images that comprise the Underwater Image Enhancement Benchmark (UIEB), and these underwater images are collected from Google, YouTube, related papers and paper researcher self-captured videos [44].

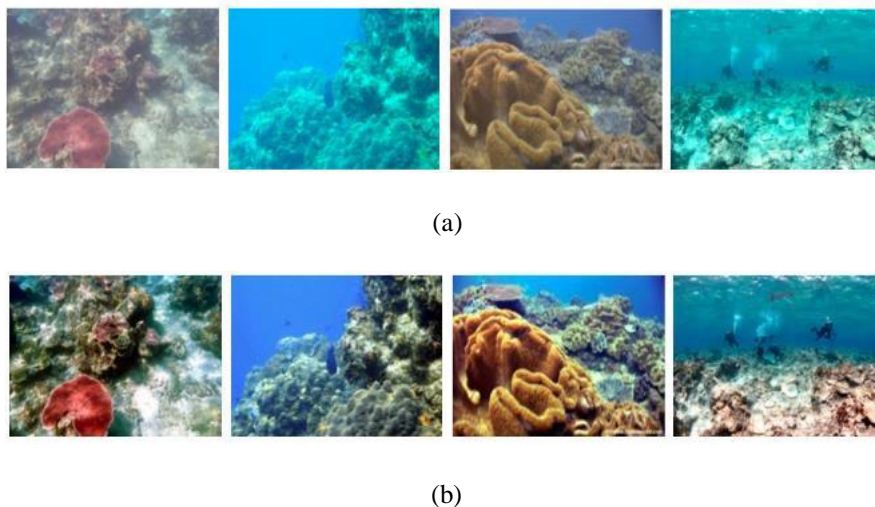


Figure 1. Examples from UIEB with subclasses: (a) original underwater images, (b) corresponding reference images.

Some examples of underwater images from MUED [43] with high turbidity, uneven illumination, monotonous hues, and intricate underwater-background are shown in Figure 2. These issues have a significant impact on the reliability and availability of underwater images in real-world applications.

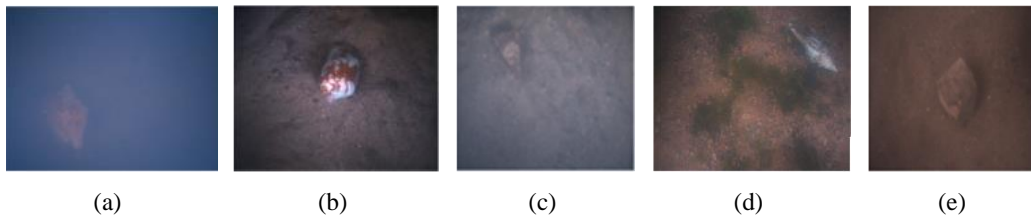


Figure 2. Some examples of detrimental elements present in the marine environment that can affect the use of underwater vision. (a) Water with high turbidity, (b) Uneven illumination, (c) Low contrast, (d) Complicated underwater-background, and (e) Monotonous color

Images captured using an underwater optical imaging and capturing device as part of the Real-time Underwater Image Enhancement (RUIE) Dataset are shown in Figure 3. The Underwater Image Quality Subclass, Underwater Color Cast Subclass, and Underwater higher-level task-driven Subclass are the three subclasses of underwater images that comprise RUIE. In order to gather image examples for the RUIE benchmark, they put up a multi-view underwater image capture system with twenty-two water-proof video cameras.

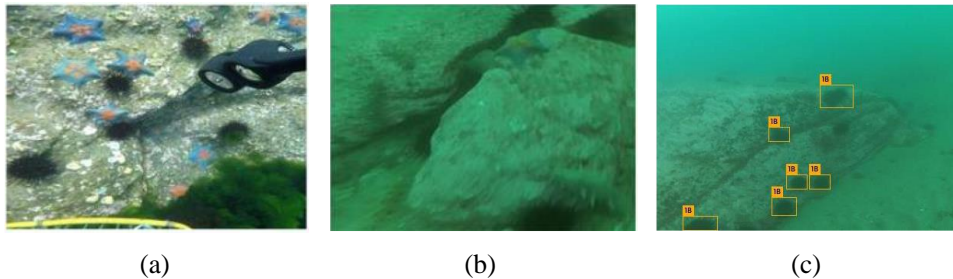


Figure 3. Some images from the RUIE dataset with a triple of subclasses of underwater images: (a) Underwater Image Quality Sub-aggregate, (b) Underwater Color Cast Sub-aggregate, (c) Underwater higher-level task-driven Sub-aggregate.

Figure 4 illustrates a sampling of the results of object detection and instance segmentation models trained on both versions of the datasets [46]. The outcomes encompass an extensive range of object sizes and situations.

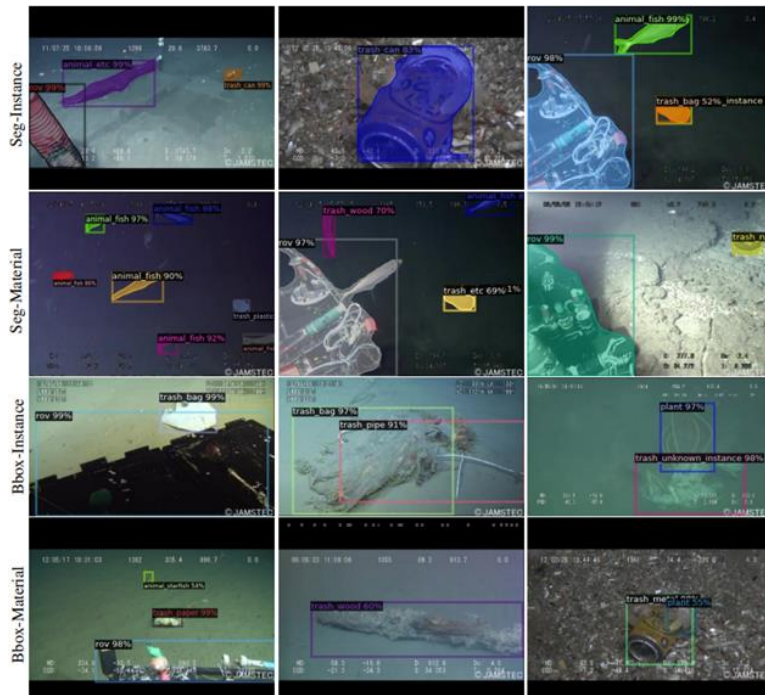


Figure 4. Sampled results for object detection and image segmentation for both versions of the TrashCan dataset.

The first large-scale, diverse, underwater benchmarking dataset (UOT100) was created with over 74,000 annotated frames spread across 104 video sequences. Both synthetic and natural underwater imagery have similarly distributed aberrations in the dataset as a whole, many different YouTube channels and other internet video platforms contributed to the dataset, as did preposted and manually annotated ground truth bounding box. Figure 5 shows a visual summary of the distortions as categories that represent the color of the water, such as blue, green, and yellow.

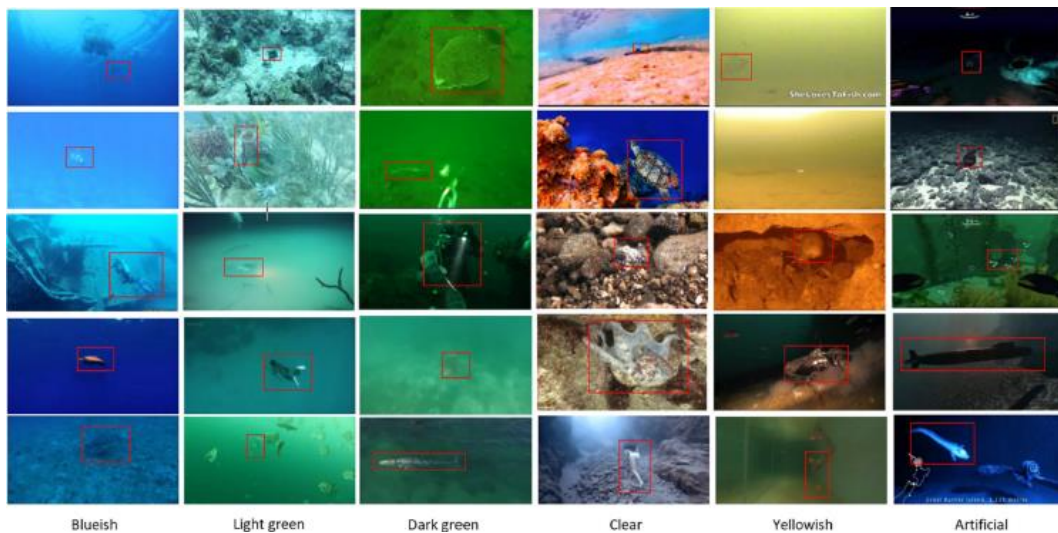


Figure 5. Sample tracking data from our UOT100 dataset showing various types of distortions. The red bounding boxes denote the object of interest and the text below each column indicates the category of the visual data

In total there are 1,525 RGB images in the SUIM dataset that may be used for either training or validation, and an additional 110 test images can be used as a benchmark for assessing the performance of semantic segmentation models. There is a wide range of spatial resolutions present in the photos, including 256×256 , 640×480 , 1280×720 and 1906×1080 . Seven human volunteers labeled every pixel of the SUIM dataset. An example or two can be seen in Figure 4.6.

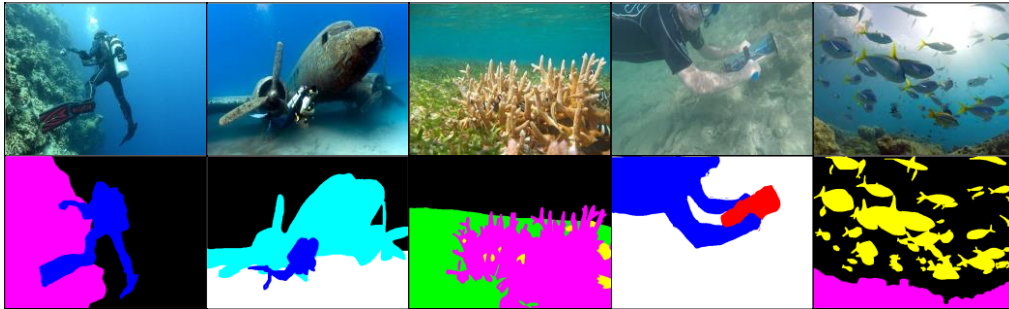


Figure 6. A few sample images and corresponding pixel-annotations are shown on the top and bottom rows, respectively

There are currently 385 wreck images, 36 drowning victim images, 62 aircraft images, 129 mine images, and 578 seafloor images in the dataset known as SeabedObjects-KLSG. All of the images were taken directly from the raw data of the large sides can sonar images. Figure 7 shows some data from the SeabedObjects-KLSG dataset.

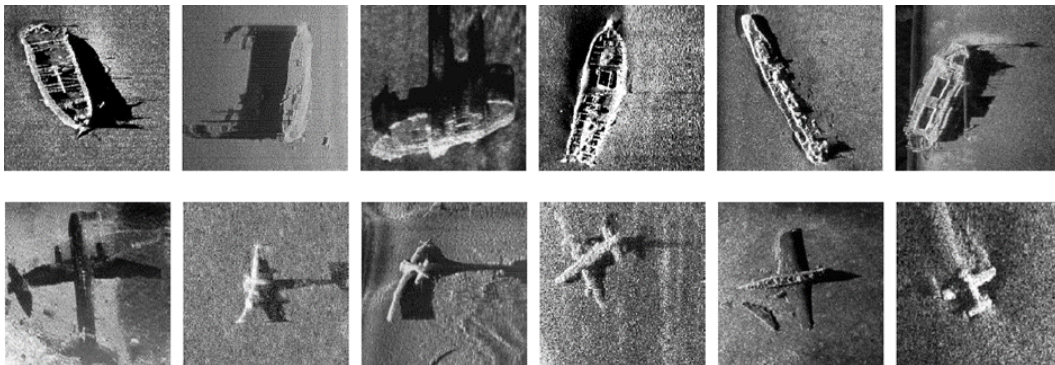


Figure 7. Samples from the SeabedObjects KLSG dataset.

Research on marine ecosystems is aided by the Fish4Knowledge dataset, which was released by the Taiwan Ocean Research Institute and numerous other partner institutes. Figure 8 depicts a handful of images from the dataset consisting of 27,370 tagged underwater images of 23 distinct fish species acquired over the course of two years by 10 underwater cameras in Taiwanese inland lakes.



Figure 8. Examples of underwater images on a Taiwan reef with different background variability.

Kyutech10K has 10,728 images and 1,489 videos over seven different categories (shrimp, squid, crab, shark, sea urchin, manganese and sand). Every still image and video clip will always be displayed at a maximum resolution of 480×640 pixels. In Figure 9, we provide a sample of images for each group.

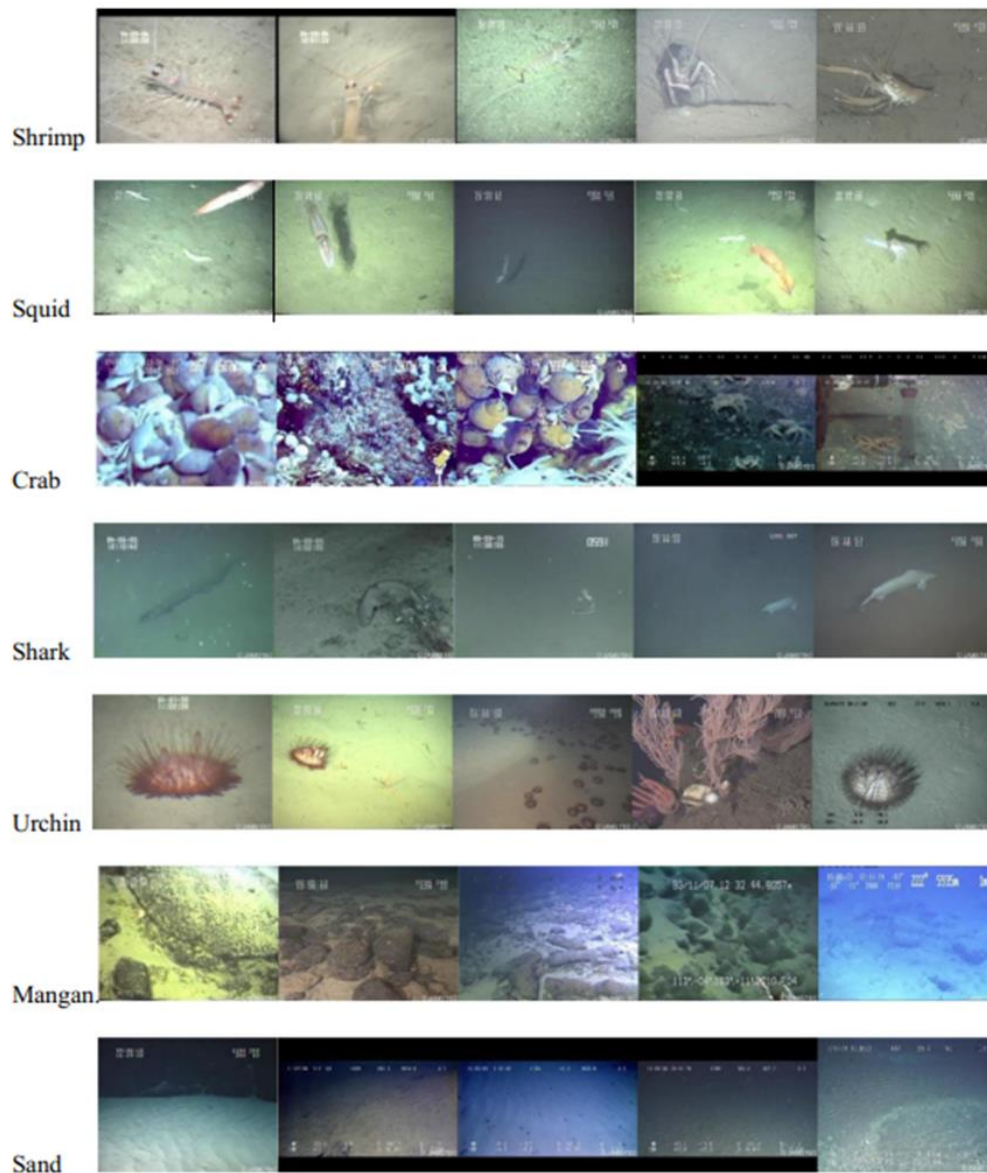


Figure 9. The Kyutech10K dataset.

5. PREVIOUS RESEARCH METHODS

It has been shown that a deep Convolutional Neural Network, such as the one proposed by Nicole Seese et al. [52], performs admirably in a dynamic setting, hence these researchers proposed an Adaptive Foreground Extraction Method using a deep Convolution Neural Network for classification. Because of its emphasis on lighting uncertainty, background motion and non-static imaging platforms, it performs well in practical settings. A Gaussian Mixture Model is employed in dynamic settings, while a Kalman filter is reserved for less complex circumstances. Therefore, the method's efficiency and speed are likely to deteriorate.

The paper by Xiu Li Min Shang et al. [53] uses a fast R CNN approach designed specifically for the detection of fish. The approach returns values with higher mean average precision and is faster than R CNN (map). In total, the study contributed to the creation of a brand-new, massive dataset consisting of 242,722 images over 12 distinct classes. Time-consuming selective search is

used to collect the input of 2,000 regions of interest (ROI) for the network. Despite its rapidity, this operation is not real-time.

With regard to hybrid features, the deep learning method utilizing VGGNet presented by A. Mahmood et al. [54] proposes an extraction strategy based on the Spatial Pyramid Pooling (SPP) approach, in which a pre-trained VGGNet is used to improve categorization by combining deep features from the VGGNet with texture and color-based characteristics. The CNN is then trained using the MLC dataset.

More accurate detection of zooplankton with the Convolutional Neural Network-based ZooplanktonNet model is presented by Jialun Dai et al. [55]. In order to reduce overfitting, it leverages augmentation of existing data to make the classification process more accurate. CNN is a more efficient image classification system since it does not rely on training data or previous samples. Despite there being an insufficient number of zooplankton images to train deep neural networks, this study appeared to work well with less knowledge.

In order to achieve fine-grained classification using a CNN, Hansang Lee et al. [56] combine transfer learning with a pre-trained CNN. A combination of data augmentation methods, including transfer learning, was employed to correct the issue of class imbalance. It is applicable and efficient to produce a satisfying outcome, and it is particularly useful for large-scale class imbalance datasets.

Sebastien Villon et al. [57] proposed a combination of Convolutional and Deep Learning techniques, a Neural Network, and HOG+SVM to detect submerged objects. This combination is able to identify coral reef fish from video stills taken underwater. The study titled "A Comparative Study of Robust Underwater Object Detection with Autonomous Underwater Vehicle" ICCA 2020, Dhaka, Bangladesh found that deep learning yields better detection accuracy than conventional approaches. With the use of image contours, HOG can uncover intricate situations that are otherwise obscured, such as those hiding in coral reefs.

Convolutional neural networks (CNNs) with a global average pooling (GAP) layer before each fully connected layer to generate a class activation map were proposed by Gebhardt et al. in [58]. To locate MLOs in sidescan sonar images, the researchers in [58] used a DNN. The authors examined how several factors, including DNN depth, memory, calculation, and training data distribution, affected detection performance. Furthermore, they used visualization methods to make the model's behavior more understandable to end users. Complex DNN models produce higher accuracy (98%) than simple DNN models (93%) and perform better (78%) than SVM models. The most complex DNN models improved performance by 1 percent but required 17 times as many trainable parameters to do so. The described method uses less computing power than DNNs designed for multi-class classification workloads. For this reason, it can be used by unmanned marine vehicles.

In order to perform semantic segmentation, the SegNet [59] uses a fully convolutional encoder-decoder architecture. All thirteen convolutional layers used by the VGG16 image classifier are replicated topologically in its encoder network. The SegNet's decoder network architecture allows for far less memory to be used, which is its primary advantage over alternative segmentation systems. Since the SegNet is a traditional CNN-based image segmentation architecture, we deemed it to be a good candidate for evaluation.

Table 2. Review of existing databases for underwater object detection that can be made accessible to the public.

Method	Advantages	Disadvantages
Adaptive Foreground Extraction Method [52]	CNN for classification works well in dynamic environments, focuses on uncertain illumination factors and non-static environments, and works well in dynamic environments.	Use of the Gaussian Mixture Model and the Kalman Filter, both of which diminish speed and efficiency, should be relegated to more complicated and dynamic circumstances.
R-CNN [60]	Utilizes a filtered search in order to generate regions. Approximately two thousand regions are retrieved from each image.	Because each region is handed over to the CNN model on an individual basis, a significant amount of processing time is consumed. In addition, it uses three distinct networks to make predictions.
Fast R-CNN [53]	Faster than R-CNN, the dataset for recreation of fish. The CNN model only has to be trained once with each image before extracting feature maps. Predictions are generated via a selective search on these feature maps. It utilizes all three models used by R-CNN.	The use of 2,000 regions of interest as input necessitates a significant amount of startup time and is therefore inapplicable to real-life scenarios
Faster RCNN [60]	Selective search has been replaced in this model by the use of a technique called Region Proposal Network (RPN). In comparison to the other versions listed above, RPN increases the speed of the model significantly.	-To successfully extract all items from a single image, the method requires multiple iterations. -Due to the sequential nature of these algorithms, the success of subsequent stages of the network is contingent on the results of previous systems.
VGGNet [54]	Features are hybrid and deep features are used for pre-training.	Utilization of the MLC Dataset, which is inappropriate for use in image classification.
ZooplanktonNet [55]	A high accuracy rate, the use of data augmentation to reduce the amount of data overfitting, and reduced preprocessing are all features of this model.	The absence of images of plankton, which is necessary for a deep neural network, which requires massive datasets.
CNN+Transfer Learning [56]	Pre-trained CNN, overcoming the class imbalance problem, use of numerous data augmentation approaches.	Optimal for massive data-intensive tasks, not at all for more modest endeavors.

HOG+SVM [57]	Used for locating submerged items that may otherwise go undetected.	More time-consuming and less effective than deep learning approaches in terms of both detection and efficiency.
Different structures of convolutional neural networks (CNNs) [58]	- High accuracy (93%) - Can be used with self-driving underwater vehicles	When compared to DNNs designed for multi-class classification applications, the computational requirements of the proposed method are lower.
SegNet [59]	Decoder network's capacity severely reduces RAM consumption.	The precision of feature extraction is linearly proportional to the complexity of the model.

6. CONCLUSIONS

Because of its promise, deep learning has already altered many facets of public life. Generic object detection has been quite successful thanks to the availability of large amounts of data and powerful computers. The field of marine engineering has focused much attention in recent years to methods of detecting objects submerged in the ocean using deep learning. This can be used for a variety of marine pursuits. Based on the current state of the art in underwater object identification research, this study provides a thorough categorization and analysis of relevant publications. Well-known reference datasets have been covered. A comparison is made between various deep learning methods and more traditional methods. The ideal approach for underwater item detection seems to be the Convolutional Neural Networks (CNN), which are generally regarded for computer vision models and classification in complicated situations. The goal of this article is to provide readers with a thorough understanding of the current state of underwater object detection in the hope that it will help them in their own research endeavors.

REFERENCES

- [1] Wu, H., Q. Liu, and X. Liu, A review on deep learning approaches to image classification and object segmentation. TSP, 2018. 1(1): p. 1-5.
- [2] Lowe, D.G.J.I.j.o.c.v., Distinctive image features from scale-invariant keypoints. 2004. 60(2): p. 91-110.
- [3] Dalal, N. and B. Triggs. Histograms of oriented gradients for human detection. in 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05). 2005. Ieee.
- [4] Lienhart, R. and J. Maydt. An extended set of haar-like features for rapid object detection. in Proceedings. international conference on image processing. 2002. IEEE.
- [5] Cortes, C. and V.J.M.I. Vapnik, Support vector machine. 1995. 20(3): p. 273-297.
- [6] Freund, Y., R.E.J.J.o.c. Schapire, and s. sciences, A decision-theoretic generalization of on-line learning and an application to boosting. 1997. 55(1): p. 119-139.
- [7] Felzenszwalb, P.F., et al., Object detection with discriminatively trained part-based models. 2010. 32(9): p. 1627-1645.
- [8] Yang, H., et al., Research on underwater object recognition based on YOLOv3. Microsystem Technologies, 2021. 27(4): p. 1837-1844.
- [9] LeCun, Y., et al., Backpropagation applied to handwritten zip code recognition. Neural computation, 1989. 1(4): p. 541-551.
- [10] LeCun, Y., et al., Handwritten digit recognition with a back-propagation network. Advances in neural information processing systems, 1989. 2.

- [11] Anagnostopoulos, C.-N.E., et al., License plate recognition from still images and video sequences: A survey. *IEEE Transactions on intelligent transportation systems*, 2008. 9(3): p. 377-391.
- [12] El Ayadi, M., M.S. Kamel, and F. Karray, Survey on speech emotion recognition: Features, classification schemes, and databases. *Pattern recognition*, 2011. 44(3): p. 572-587.
- [13] Borkar, A., M. Hayes, and M.T. Smith, A novel lane detection system with efficient ground truth generation. *IEEE Transactions on Intelligent Transportation Systems*, 2011. 13(1): p. 365-374.
- [14] Liu, W., et al. SpheroFace: Deep hypersphere embedding for face recognition. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.
- [15] Yang, X., P. Molchanov, and J. Kautz. Making convolutional networks recurrent for visual sequence learning. in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018.
- [16] Zabidi, M.M., et al. Embedded vision systems for ship recognition. in *TENCON 2009-2009 IEEE region 10 conference*. 2009. IEEE.
- [17] Jin, L. and H. Liang. Deep learning for underwater image recognition in small sample size situations. in *OCEANS 2017-Aberdeen*. 2017. IEEE.
- [18] Meng, L., T. Hirayama, and S.J.I.A. Oyanagi, Underwater-drone with panoramic camera for automatic fish recognition based on deep learning. 2018. 6: p. 17880-17886.
- [19] Yuh, J., G. Marani, and D.R.J.I.s.r. Blidberg, Applications of marine robotic vehicles. 2011. 4(4): p. 221-231.
- [20] Liu, Z., et al., Unmanned surface vehicles: An overview of developments and challenges. 2016. 41: p. 71-93.
- [21] Yang, H., et al., Research on underwater object recognition based on YOLOv3. 2021. 27(4): p. 1837-1844.
- [22] Deng, J., et al. Imagenet: A large-scale hierarchical image database. in *2009 IEEE conference on computer vision and pattern recognition*. 2009. Ieee.
- [23] Viola, P. and M. Jones. Rapid object detection using a boosted cascade of simple features. in *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition*. CVPR 2001. 2001. Ieee.
- [24] Viola, P. and M.J.J.I.j.o.c.v. Jones, Robust real-time face detection. 2004. 57(2): p. 137-154.
- [25] Rätsch, G., T. Onoda, and K.-R.J.M.I. Müller, Soft margins for AdaBoost. 2001. 42(3): p. 287-320.
- [26] Yang, B., et al. Aggregate channel features for multi-view face detection. in *IEEE international joint conference on biometrics*. 2014. IEEE.
- [27] Cerf, M., et al., Predicting human gaze using low-level saliency combined with face detection. 2007. 20.
- [28] Luo, Z., et al. Rotation-invariant histograms of oriented gradients for local patch robust representation. in *2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*. 2015. IEEE.
- [29] Felzenszwalb, P., D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. in *2008 IEEE conference on computer vision and pattern recognition*. 2008. Ieee.
- [30] Liu, W., et al. Ssd: Single shot multibox detector. in *European conference on computer vision*. 2016. Springer.
- [31] Newell, A., K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. in *European conference on computer vision*. 2016. Springer.
- [32] DOLAPCI, B., C.J.J.o.I.S.T. ÖZCAN, and Applications, Automatic ship detection and classification using machine learning from remote sensing images on Apache Spark. 2021. 4(2): p. 94-102.
- [33] LeCun, Y., et al., Gradient-based learning applied to document recognition. 1998. 86(11): p. 2278-2324.
- [34] Krizhevsky, A., I. Sutskever, and G.E.J.A.i.n.i.p.s. Hinton, Imagenet classification with deep convolutional neural networks. 2012. 25.
- [35] Simonyan, K. and A.J.a.p.a. Zisserman, Very deep convolutional networks for large-scale image recognition. 2014.
- [36] Lin, M., Q. Chen, and S.J.a.p.a. Yan, Network in network. 2013.
- [37] Springenberg, J.T., et al., Striving for simplicity: The all convolutional net. 2014.
- [38] Szegedy, C., et al. Going deeper with convolutions. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015.
- [39] Szegedy, C., et al. Inception-v4, inception-resnet and the impact of residual connections on learning. in *Thirty-first AAAI conference on artificial intelligence*. 2017.

- [40] He, K., et al. Deep residual learning for image recognition. in Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
- [41] Huang, G., et al. Densely connected convolutional networks. in Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.
- [42] Larsson, G., M. Maire, and G.J.a.p.a. Shakhnarovich, Fractalnet: Ultra-deep neural networks without residuals. 2016.
- [43] Jian, M., et al., The extended marine underwater environment database and baseline evaluations. 2019. 80: p. 425-437.
- [44] Li, C., et al., An underwater image enhancement benchmark dataset and beyond. 2019. 29: p. 4376-4389.
- [45] Liu, R., et al., Real-world underwater enhancement: Challenges, benchmarks, and solutions under natural light. 2020. 30(12): p. 4861-4875.
- [46] Hong, J., M. Fulton, and J.J.a.p.a. Sattar, Trashcan: A semantically-segmented dataset towards visual detection of marine debris. 2020.
- [47] Kezebou, L., et al. Underwater object tracking benchmark and dataset. in 2019 IEEE International Symposium on Technologies for Homeland Security (HST). 2019. IEEE.
- [48] Islam, M.J., et al. Semantic segmentation of underwater imagery: Dataset and benchmark. in 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 2020. IEEE.
- [49] Huo, G., Z. Wu, and J.J.I.a. Li, Underwater object classification in sidescan sonar images using deep transfer learning and semisynthetic training data. 2020. 8: p. 47407-47418.
- [50] Lines, J., et al., An automatic image-based system for estimating the mass of free-swimming fish. 2001. 31(2): p. 151-168.
- [51] Lu, H., et al., FDCNet: filtering deep convolutional network for marine organism classification. 2018. 77(17): p. 21847-21860.
- [52] Seese, N., et al. Adaptive foreground extraction for deep fish classification. in 2016 ICPR 2nd Workshop on Computer Vision for Analysis of Underwater Imagery (CVAUI). 2016. IEEE.
- [53] Li, X., et al. Fast accurate fish detection and recognition of underwater images with fast r-cnn. in OCEANS 2015-MTS/IEEE Washington. 2015. IEEE.
- [54] Mahmood, A., et al. Coral classification with hybrid feature representations. in 2016 IEEE International Conference on Image Processing (ICIP). 2016. IEEE.
- [55] Dai, J., et al. ZooplanktoNet: Deep convolutional network for zooplankton classification. in OCEANS 2016-Shanghai. 2016. IEEE.
- [56] Lee, H., M. Park, and J. Kim. Plankton classification on imbalanced large scale database via convolutional neural networks with transfer learning. in 2016 IEEE international conference on image processing (ICIP). 2016. IEEE.
- [57] Villon, S., et al. Coral reef fish detection and recognition in underwater videos by supervised machine learning: Comparison between Deep Learning and HOG+ SVM methods. in International Conference on Advanced Concepts for Intelligent Vision Systems. 2016. Springer.
- [58] Gebhardt, D., et al. Hunting for naval mines with deep neural networks. in OCEANS 2017-Anchorage. 2017. IEEE.
- [59] Badrinarayanan, V., et al., Segnet: A deep convolutional encoder-decoder architecture for image segmentation. 2017. 39(12): p. 2481-2495.
- [60] Fayaz, S., et al., Underwater object detection: architectures and algorithms—a comprehensive review. 2022: p. 1-46.

BRAND NAME (TO DO): AN INTERACTIVE AND COLLABORATIVE DRAWING PLATFORM TO ENGAGE THE AUTISM SPECTRUM IN ART AND LANGUAGE LEARNING USING ARTIFICIAL INTELLIGENCE

Xuanxi Kuang¹ and Yu Sun²

¹University High school, 4771 Campus Drive, Irvine, CA 92612

²California State Polytechnic University, Pomona, CA, 91768, Irvine, CA 92620

ABSTRACT

Special communities specific to Autism Spectrum disorder face difficulties both socially and communicably [1]. Autism spectrum disorder will affect their expression and response to society, and they'll have a hard time learning and following complex directions [2]. This paper proposes software to promote one's collaborative skills and drawing skills with interaction with the AI system. At the same time, it also tries to raise awareness of the special group in our society. As an open platform, each individual will have opportunities to work with other users to cooperate, and they'll have a chance to learn drawing step by step from drawing that is contributed by more than 15 million players around the world. They can decorate the object with a color adjective to enhance their sense of beauty. In order to test the usability of the software, we did two experiments to test the accuracy of the graph and color combination. The result shows this software achieves a high accuracy on color input and obtains a correct graph from the input.

KEYWORDS

Interactive, Artificial intelligent, Self learning process.

1. INTRODUCTION

Autism spectrum disorder (ASD) is a developmental disability that affects their response to the environment not only behaviorally, but also socially and communicatively [3]. Autism children may have difficulty with meaning and understanding what others are trying to express—they might misunderstand others. Communication might be a question for them, and they may have problems with developing languages and arranging words for others [4]. This project is designed for a special community—specific to autistic teens and kids who need a platform to practice their collaborative skills and who are trying to learn to draw on their own. I believe that capitalizing on each child's interest will be effective in promoting positive behavioral growth. Collaboration using art and composition between each individual not only will improve their aesthetic ability to surroundings, but also improve their communication skill [5]. The project includes a stick figure which they can design by using a different color. Even if they don't like the random figure, they still can refresh until they find the one they like. The project also included step-by-step learning, coloring, and collaboration for them to improve their language skill with others. It will be a great platform for special kids who need someone to talk to, practice their social skills, or purely just

David C. Wyld et al. (Eds): ARIA, SIPR, SOFEA, CSEN, DSML, NLP, EDTECH, NCWC - 2022
pp. 65-74, 2022. CS & IT - CSCP 2022 DOI: 10.5121/csit.2022.121506

want to learn drawing. A platform is one of the resources and tools to express their feeling and their opinion in this world. At the same time, this project raises the attention on supporting autistic children's education and provides more opportunities to learn for the special groups [6].

Some of the AI techniques and systems that have been proposed to interact with Autism kids. For example, The app Emotion Charades allows the AI to monitor a person's facial reaction with the corresponding emoji, and check the sign for anxiety when needed [7]. The AI interactive techniques are also proposed to provide opportunities for the Autism group to draw. However, most of the drawing apps miss the part of teaching. Their implementations are also limited. Most of the teaching process is taking the form of either human-to-human interaction through watching a video and taking a class from the internet, or console games that provide opportunities for them to draw based on the picture given. Without the internet, the learning resources will be limited. For example, The doodle Buddy App provides opportunities for children to draw, write, and upload photos. It is an interactive app that opens up space for creativity, but it misses the learning drawing technique process which is essential for children to obtain knowledge. The second practical problem is that some users might find it hard to understand the design of the website and app. Autism might have difficulties following along with the games and drawing process due to the disabilities of response to the information on the website or app. Either the website has too many instructions, the language is too hard, or the guideline is too vague can all affect their cooperation and interaction with the AI systems. Distracting advertisements on the App and website may also become a hidden problem for the ability to concentrate. For many autistic children, experiencing overwhelm and overloading the information might lead to a lack of focus. But more Autism usually tends to increase the ability to focus especially when they're exploring their interest.

In this paper, we follow the same line of research by using AI to generate different images. By using the Quickdraw database to create a variety of pictures based on the object [8]. To create a Graphic user interface, we import Tkinter as one of the frameworks built within Python [9]. Also, use spacy as one of the Language processing systems. Our goal is to build a simple interactive platform to provide opportunities for children to learn how to draw and collaborate with others. Our method is inspired by one of my volunteer experiences. By tutoring an autistic girl whose mindset changes so fast that leads up to misunderstanding and problems with communication. So this program provides a resource that demonstrates how to draw step by step by themselves, so they can draw the images in their mind. There are some good features of the drawing programs. First, the quickdraw database provides a variety of image resources from everywhere around the earth to meet the demand. Second, By providing the option of redraw and slow mode, children are easier to follow the step-by-step drawing. Third, By separating nouns and adjectives, the user can add different color arrangements to the drawing. Therefore, we believe that a good platform with simple instruction and showing a teaching step is a tool for children who are trying to obtain art knowledge.

In two application scenarios, we demonstrate how the quickdraw database can be used as an effective database that contains various resources, and the Language processing system can be an effective tool to identify color using the difference between nouns and adjectives. First, we show the usefulness of the quickdraw database corresponding to our Algorithm that an AI system can reorganize user input and find a random graph from more than a million resources [10]. As the result of the experiment demonstrates, the project displays a highly accurate graph from the input. Quickdraw database not only provides massive graph resources from the world but also provides the straw-by-straw step that is easy for the user to follow up on the process of drawing. The algorithm efficiently analyzes the input into the graph and then sends it back to the back-end server. Second, we use Spacy as our Language processing system to provide an additional color option for the user. Its main function is to recognize color within the user input. It is effective on

the basic color which can be separated with nouns and adjectives. With the coloring option, the user can interact with different color options and try out different color arrangements in the graph. Our result shows within the basic 10 colors arrangement, the accuracy of the color can be up to 90%. So the system can provide high usability on the interaction with the color and more possibility.

The rest of the paper is organized as follows: Section 2 gives the details on the challenges that we met during the experiment and designing the sample; Section 3 focuses on the details of our solutions corresponding to the challenges that we mentioned in Section 2; Section 4 presents the relevant details about the experiment we did, following by presenting the related work in Section 5. Finally, Section 6 gives the concluding remarks, as well as pointing out the future work of this project.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Satisfaction from user

One of the challenges is to ensure customer satisfaction. During the process of building the platform, we need to take into account the quality of the program from an Autistic child's perspective. In order to build friendly user software, it is necessary to have simple instructions and an easy to follow interface. Complex interfaces and guidelines that are too vague might engender discomfort and loss of cooperation and interaction. The overload of information might also lead to emotional stress. Bad quality safeguards within a software may cause dissatisfaction from the user, and if the application didn't work as users expect, it may lead to a credibility decrease. So ensuring customer satisfaction is one of the important challenges when building the program.

2.2. Usability of the program

Due to the consideration that this is an interactive program that opens up space for the creation of a special group, the usability of the program is an essential part to consider. The usefulness of a product is important to society, and the product meets the expectation of the designer. In Which program should be designed easy for users to perform an assigned task without confusion. The product needs to ensure its functions, as well as possibilities to raise the value of the product, can be hard to operate. if a product does not have good usability, the level of dissatisfaction with the product will increase as well as complaints from the user. Higher usability will engender learning quicker and retain knowledge longer, which reduces the training costs and time users spend on learning. In order to generate a user-friendly product, usability must be taken into consideration during the construction process.

2.3. Limitation of the software

Software with no limitation can create a barrier for the user to access the resource. When a software product doesn't fit with the user's computer system, the computer will be unable to perform the assigned task that the code's trying to do. This is also the biggest challenge in which different users have a variety of computer systems. We need to consider the user's situation, and we can't meet all users' computer capabilities, and the access to the resource will be limited as well. That's also one of my problems, my original computer didn't have the capabilities to support me to finish my project, and that forced me to change a computer for my code work. If

the software has a large limitation, that will cause the usability to decrease and the feedback of dissatisfaction from the user will increase.

3. SOLUTION

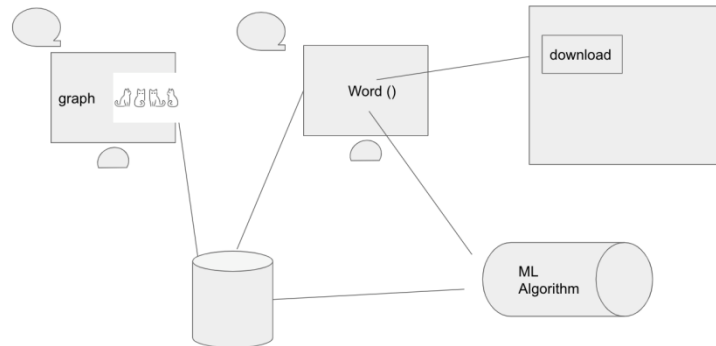


Figure 1. Overview of the solution

- 1) Customer Front End GUI (Python Thinker)
 - a) User could input the words
 - b) User could receive the graph
 - c) User could interact with other users (The step)
- 2) Backend Server (Python Flask)
 - a) Server to provide service to receive the word data stable
 - b) Server to host AI algorithm to generate the graph
 - c) Server to send the graph back to users
- 3) Algorithm (Python ML)
 - a) Algorithm to generate pictures from words
- 4) Demo Web (Html CSS Javascript)
 - a) Provide the demo and path where users could download the app

This project mainly serves for the Autism group to interact with AI drawing systems which include step by step learning, coloring, and collaborating for them to improve their language skill with others. The project includes four main components. Consumer Front End GUI, Backend server, Algorithm, and Demo Web. Customer Front End GUI is written in Python Tkinter which provides a user interface where the user could input the words, receive the graph, and interact with other users. GUI is one type of computer graphics technology, which generally consists of graphical controls such as windows, drop-down menus, or dialog boxes. The user interacts with the machine by clicking the menu bar, button, or pop-up dialog box. Customer Front End GUI connected to a Backend server that is mainly provided by Python Flask. This main function is to host the AL algorithm to generate the graph, receive the word data stable, and send the graph back to users. It is mainly used for server operation and maintenance management, database management, and interfaces (private interfaces) used to connect the front and back end. The third part of the components is the Algorithm composed of Python ML, and its function is to generate pictures from words. In this part of the component, Computer learns and stimulates the unordered data to its own knowledge and reorganizes the data that send from the user to a piece of useful information that ultimately finds the corresponding graph. The last part of the main component is the Demo Website written by Html CSS Javascript, which provides the demo and path where users could download the app.

Customer Front End GUI (140)

Customer Front End GUI connected to the backend servers. It is the platform that users would usually see when they run the program. It allows the user to input the words through the windows system provided, receive the graph from the Algorithm, and interact with others. The program's graphical user interface provides a basic window for displaying graphs, several buttons for different uses, and dialog boxes to type in their thoughts. Users can interact with the AI by clicking the button or sending the input words. Tkinter is one of the GUI toolkits that python usually provides. After installing python, you can directly use Tkinter without installing it separately. As a Python GUI tool, Tkinter has good cross-platform support and supports Windows, Mac, etc. It inherits the basic features of Python's concise syntax and high coding efficiency for beginners to learn.

Backend End (110)

The backend server is connected by the Customer front end GUI and supports delivering the data to the Algorithm. It provides a data stable platform that could receive the words from the user, consists of a host AL algorithm to generate the graph, and is able to send the graph back to the users (Customer Front) system. The feature is primarily provided by Python Flask, which is a framework implemented by Python that allows users to quickly implement web services using Python Language. Its main function focuses on the management of maintenance, database, and interfaces. It controls the content of input words and interacts with the database to process the corresponding input.

Algorithm (97)

The algorithm is connected by the Backend server that is supposed to translate the input word into pictures and then send them back to the server. The algorithm is one branch of an AI system that learns from the user input and reorganizes the unordered data into a piece of useful information then sends it back to the backend. It allows users to provide a computer algorithm large amounts of data drawn from the quickdraw database, and then analyze the data from the database in order to make a data-driven decision-- specifically drawing the graph from the database and then sending it back to the Backend.

Demo Web (Html CSS Javascript)

The last main component of the project is the Demo Website written in HTML. Its main function is to provide a platform where users can access the resource. On the Website, Users will be able to download the app, check our slogan, and the necessary procedure required for downloading. This platform will open wide and be easy to follow for everyone. It contains the link of the project, download option, server's description, possible video, and images of the project that allow users to access.

```

37 def draw(data):
38     global xcoordinate
39     if data['stroke'] == 'slow':
40         drawing = qd.get_drawing(data['noun'], data['doodleNumber'])
41         adj = data['adj'].split(' ')
42         for stroke in drawing.strokes:
43             strokecolor = adj[random.randint(0, len(adj) - 1)]
44             for i in range(len(stroke) - 1):
45                 canvas.create_line(
46                     (stroke[i][0] + xcoordinate, stroke[i][1], stroke[i + 1][0] + xcoordinate, stroke[i + 1][1]),
47                     fill=strokecolor, width=4)
48                 time.sleep(0.2)
49
50             root.update()
51             xcoordinate += 300
52     elif data['stroke'] == 'animate':
53         drawing = qd.get_drawing(data['noun'], data['doodleNumber'])
54         adj = data['adj'].split()
55
56         for stroke in drawing.strokes:
57             strokecolor = adj[random.randint(0, len(adj) - 1)]
58             for i in range(len(stroke) - 1):
59                 canvas.create_line(
60                     (stroke[i][0] + xcoordinate, stroke[i][1], stroke[i + 1][0] + xcoordinate, stroke[i + 1][1]),
61                     fill=strokecolor, width=4)

```

Figure 2. Screenshot of code 1

This section of the code was for specifying to the computer how to draw an image As the user is given a certain input data. It is a section of front-end code that allows us to insert input content through the code. The data that the user types in, for example, color information, object information, slow mode, or regular animations presenting can come from either the user or the back-end servers which the AI will analyze the information and user's data and then transfer to the front end. The If statement is provided for adjusting the speed of the straw, either slow or regular speed, providing a more convenient way for users if they can't follow up the normal speed. And the slow speed is 0.2 seconds between each stroke.

4. EXPERIMENT

4.1. Experiment 1

Experiment 1 is related to the words and interaction with graphs. As we introduce that the project should be able to generate graphs from the user input words. We want to make sure that our algorithm is functional which translates the inputs into a graph and then sends them back to the server. As mentioned before, usability is an essential part to consider when doing experiments. One of the goals is to test the graph and layout of the program user-friendly and easy to operate. With the idea in mind, we built up an experiment that was designed to test the accuracy of the graph with different easy input words.

For the experiment, we randomly choose 20 different objects from our daily life as predicted user input to test the accuracy of the graph. The result of the experiment is user input can correctly correspond to the graph that shows up. But because the Quickdraw database is an open, free database where more than 15 million people can contribute, the quality of the graph can be varied. Some of the graphs' shapes are vague compared to what the object really is. For example, some of the panda graphs look like a bear, and the stick figure of the helicopter can't be recognized. Compared to the complex animal and architecture, simple shaped objects can be well present, for example, moon and star. Overall more than 85% of the graph's shape can be recognized. And it shows an above-average for usability that the system works properly.

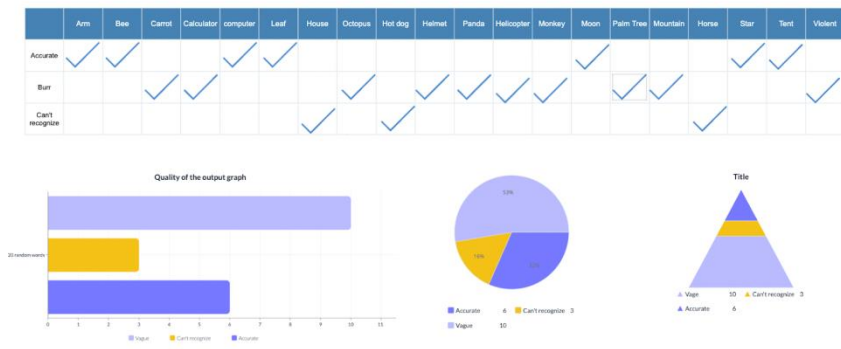


Figure 3. Data of experiment 1

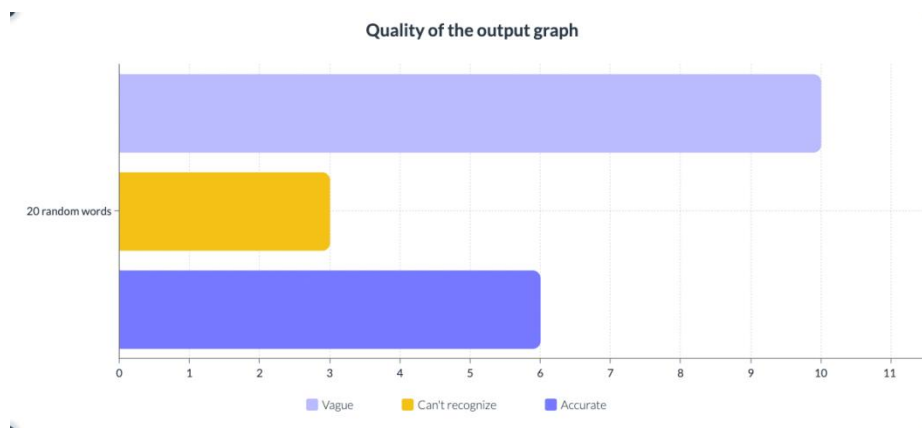


Figure 4. Quality of the output graph

4.2. Experiment 2

The second experiment is related to the interaction with color. Color provides more possibility with the engagement from the user that will raise user satisfaction with the app. We want to make sure that our Language processing system identifies the typing adjective as color and then fills our graph corresponding to it. Users can play around with different color combinations and enjoy it. We build an experiment based on our idea to test the accuracy of one color of the graph and the accuracy of the multi-color graph. To have enough samples, we test 10 different objects for some of the available color options.

For the experiment, we focus on one object: Panda to test the different colors available. Since our language process system separates different words as nouns and adjectives, some of the colors can't be recognized as an adjective. We test the basic daily 10 colors: red, blue, yellow, brown, purple, white, green, orange, and gray. The result shows that 90% of the graph can correspond to the input color. We also test on multi-colors with combinations, for example, red yellow, and yellow-brown. Find out if the single color is accurate with the graph, the color combination will also be accurate. Data demonstrate that more than 70% of the two color inputs are accurate, and 20% of the combinations only display one of the colors among the choices. Good to know that only 10% of the combined amount to 10 trails that none of the colors appear. Overall, users can test basic color combinations on the object, and there will be a high possibility to display the correct basic color.

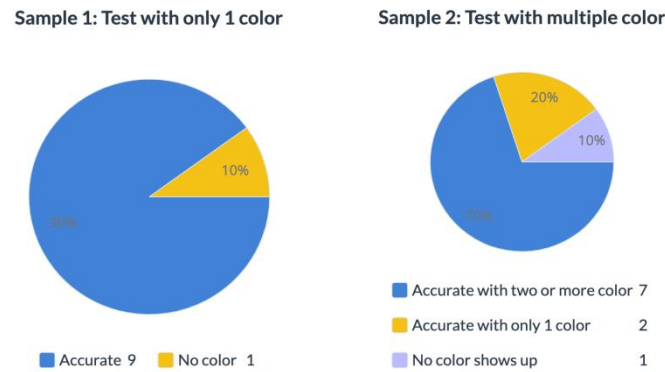


Figure 5. Sample of experiment 2

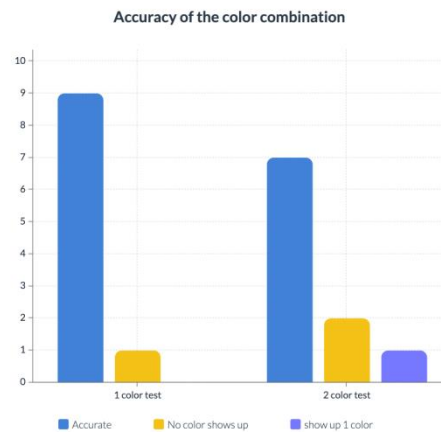


Figure 6. Accuracy of the color combination

The result of the two experiments provides the usability of the software that it obtains high accuracy on the color combination and words. Since the database obtains more than a million graphs, the quality of the graph might vary. The appearance of the software is designed to be easy for the user to complete the interaction without confusion. High accuracy of color meets the expectation of the customer satisfaction that they can decorate the graph. The algorithm works properly to analyze user input and then send back the graph to the user. The high accuracy of the image demonstrates software can provide a stable platform for users who are trying to interact with the AI system. From the result of both experiments, Algorithm is functional to translate the input into a graph, and most of the common colors in the Language processing system can be identified as an adjective to decorate the graph.

5. RELATED WORK

Golinski Pawel's project CoPainter: Interactive AI drawing experience applies the google quickdraw database and aims to provide more educational activity [11]. The activity is based on the Sketch-RNN model interactive provided by Google, so some of the intelligent-- either robots or abstract entities --can cooperate with users that ask the user to draw a painting of some object. Its main contribution is porting the Sketch RNN web app experience to Qt. Similar to my program, it uses the google quickdraw database for educational use. It is very similar to the

Quickdraw application. The main difference between CoPainter and my program is that CoPainter ask the user to draw in order to interact with AI, but my program is used to type in the input object they want to gain, so the AI can analyze the code in order to find an image that corresponds with the input.

Adam Moren and Thomas Indrais's project is similar to the Telestration game that makes the game based on doodle classification using a convolutional Neural network [12]. Its project was also inspired by the google quickdraw databases that allow users to interact with one of the AI players. The main methods the project used are Jupyter (Python) through Google Colab, and the client structure runs through JavaScript. Similar to the Telestration project, my project also uses quickdraw databases and is able to interact with AI. It is a good program for Telestration Games.

The project: From Quickdraw To story Generation System for Kids' Robot is aim to be a model for robots that accompany children to group up [13]. This project is inspired by the Google Quickdraw game that children would be able to interact with input information, and short stories based on image input. It used Multitask Transformer Network to generate the sentence based on the information from the quickdraw database, and it also used OpenAI Generative Pre-Training Model to generate stories based on the content of the sentence. The idea behind this project is really good and fun that the child would be able to draw a narrative story based on simple lines. Similar to this project, my project also uses AI interactive with the Quickdraw database, but this project transfers the input from the image to the sentence and then narrows it down into a story. It uses two different networks to generate sentences.

6. CONCLUSIONS

People who have ASD will face difficulties in understanding and communicating with others. This program is aimed to create a platform for the special community to practice collaborative skills and learn drawing from their own. The interaction between each individual will improve their communication skill and their aesthetic ability. The project is focused on a straw-by-straw process drawing and coloring program for those who want to learn simple stick figures on their own. This platform also is one of the available resources to raise awareness about supporting autistic groups. This program is composed of four main parts: Consumer Front End GUI(Python Thinker), Back end Server(Python Flask), Algorithm(Python ML), and Demo website(Html Css Javascript) [14]. The front end provides a place for the user to input the words and receive the graph. It is connected to the backend server which receives the words from the user, hosts an AI algorithm to generate the graph, and sends the graph back to the users. The algorithm uses Python ML which analyzes the input words into a picture from the database. We did two basic experiments to ensure the accuracy of the graph and color system works properly. The first experiment gives us 85% readable graphs from the quickdraw database [15]. Prove that the system can draw the corresponding word from the database. The second experiment aims to test the proper color combination on 10 major colors, and the result shows that when the user only puts 1 color, the accuracy of the color can be as high as 90%, and when the user input 2 color, the accuracy for the combination of color can reach to 70%. The system shows steady usability for the user and easy to-run system so as to improve customer experience while using it.

There are limitations on the availability of the different computer systems. The current application requires a Mac OS 11 or later. which will limit the user's practicability that not every user will be able to use it. This method requires a higher computer software standard which might decrease the practicality. The optimization might be applied to the beauty part of the program since the current program system only provides the necessary function mode, without any coloring and decoration. The accuracy of the quality of the image might also need to be

optimized due to the data of drawing can contain more than a thousand choices. The coloring system of the image can also be enhanced and optimized to provide more accuracy and open up more choices for users.

Some of the possibilities of solving the quality of the image might be writing a section of code that provides a reporting system for the user so each individual has the ability to report the quality of the drawing image. Based on the popularity of the image, the system will automatically generate and recommend the “good” image to the user so as to avoid some poor drawings. The coloring system enhancement might be more viable. The limitation of computer software systems might change a different method that is available for fitting different systems and computers.

REFERENCES

- [1] Newschaffer, Craig J., et al. "The epidemiology of autism spectrum disorders." *Annual review of public health* 28 (2007): 235.
- [2] Matson, Johnny L., and Alison M. Kozlowski. "The increasing prevalence of autism spectrum disorders." *Research in Autism Spectrum Disorders* 5.1 (2011): 418-425.
- [3] Sutherland, Georgina, Murray A. Couch, and Teresa Iacono. "Health issues for adults with developmental disability." *Research in developmental disabilities* 23.6 (2002): 422-445.
- [4] Rescorla, Leslie. "The Language Development Survey: A screening tool for delayed language in toddlers." *Journal of Speech and Hearing disorders* 54.4 (1989): 587-599.
- [5] Burleson, Brant R., and Wayne H. Denton. "The relationship between communication skill and marital satisfaction: Some moderating effects." *Journal of Marriage and the Family* (1997): 884-902.
- [6] Cahyo Adi Kistoro, Hanif, et al. "Teachers' Experiences in Character Education for Autistic Children." *International Journal of Evaluation and Research in Education* 10.1 (2021): 65-77.
- [7] Piana, Stefano, et al. "Emotional charades." *Proceedings of the 16th International Conference on Multimodal Interaction*. 2014.
- [8] Cheema, Salman, Sumit Gulwani, and Joseph LaViola. "QuickDraw: improving drawing experience for geometric diagrams." *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2012.
- [9] Beniz, Douglas, and Alexey Espindola. "Using Tkinter of python to create graphical user interface (GUI) for scripts in LNLS." *WEPOPRPO25* 9 (2016): 25-28.
- [10] Crawford, Kate, and Vladan Joler. "Anatomy of an AI System." Retrieved September 18 (2018): 2018.
- [11] Golinski, Pawel. *CoPainter: Interactive AI drawing experience*. No. STUDENT. 2019.
- [12] Gray, James H., Emily Reardon, and Jennifer A. Kotler. "Designing for parasocial relationships and learning: Linear video, interactive media, and artificial intelligence." *Proceedings of the 2017 Conference on interaction design and children*. 2017.
- [13] Wang, Lecheng, et al. "From Quick-draw To Story: A Story Generation System for Kids' Robot." *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2019.
- [14] Schifreen, Robert. "How to create Web sites and applications with HTML, CSS, Javascript, PHP and MySQL." (2009).
- [15] Sato, Shuji, Kazuo Misue, and Jiro Tanaka. "Readable representations for large-scale bipartite graphs." *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*. Springer, Berlin, Heidelberg, 2008.

CYBERBULLYING DETECTION USING ENSEMBLE METHOD

Saranyanath K P¹ and Wei Shi² and Jean-Pierre Corriveau¹

¹School of Computer Science, Carleton University, Ottawa, Canada

²School of Information Technology, Carleton University, Ottawa, Canada

ABSTRACT

Cyberbullying is a form of bullying that occurs across social media platforms using electronic messages. This paper proposes three approaches and five models to identify cyberbullying on a generated social media dataset derived from multiple online platforms. Our initial approach consists in enhancing Support Vector Machines. Our second approach is based on DistilBERT, a lighter and faster Transformer model than BERT. Staking the first three models we obtain two more ensemble models. Contrasting the ensemble models with the three others, we observe that the ensemble models outperform the base model concerning all evaluation metrics except precision. While the highest accuracy, of 89.6% was obtained using an ensemble model, we achieved the lowest accuracy, at 85.53% on the SVM model. The DistilBERT model exhibited the highest precision, at 91.17%. The model developed using the different granularity of features outperformed the simple TF-IDF.

KEYWORDS

Machine Learning, Natural Language Processing, Support Vector Machine, DistilBERT, Cyberbullying.

1. INTRODUCTION

The emergence of Internet and various multimedia applications has enabled the communication over social-media platforms. The number of users accessing such applications is increasing rapidly. This has resulted in bullying general or specific users and user groups, either knowingly or unknowingly. The abuses resulting from cyberbullying can cause psychological harm to the target users and groups [1].

Cyberbullying is defined as ‘an aggressive, intentional act carried out by a group or individual, using electronic forms of contact, repeatedly and over time against a victim who cannot easily defend him or herself [2]. Sending vulgar messages, posting private information without an individual’s consent, frequently sending offensive messages, online gossip spreading, cyberstalking etc can be considered as actions that could be termed as Cyberbullying. Studies show that about half of American teenagers have experienced cyberbullying and victims often have psychiatric and psychosomatic disorders. 8% teens have reported some form of cyberbullying among the total reported 19% bullying cases.

Cyberbullying can take place using any type of data. Text-based cyberbullying can be defined as the act of cyberbullying using texts for sending bullying messages or posts. To identify text-based cyberbullying, text classification plays a prominent role. A classification example of email involves categorising them into spam or non-spam, bullying or non-bullying. The data

classification can be achieved using classification algorithms like Naive Bayes, SVM, Neural networks and NLP.

Due to an increase in the volume of data being shared over the social media platforms, it is tedious to implement a manual approach to cyberbullying detection. Hence, machine learning models for text-based cyberbullying detection can be used as an initial mechanism to reduce the manual efforts in reviewing the content [3]. The count, density and value of offensive words can be used as features to detect cyberbullying messages. Instead of actual textual features, few works have promoted the usage of complementary information that would supplement textual cyberbullying detection. The history of user's activities, location, user personalities and emotions were considered.

Several models have been developed and modified to date using many of the state-of-the-art technologies in identifying and preventing cyberbullying detection. These models were developed using machine learning and deep learning algorithms, which have the capability of learning human data. Supervised machine learning algorithms were used to classify online harassment on MySpace and Slashdot datasets, to compare the performance of various classifiers on binary and multi-classification problems using Naive Bayes (NB), SVM on Youtube comments [4]. These algorithms can be used to identify cyberbullying by combining it with the labelled data. The existing works utilizes the capabilities of only one of the machine learning, deep learning, or word embeddings techniques. We focus on combining the approaches to leverage the capabilities and to improve the performance. A list of contributions is summarized below:

1. We perform Cyberbullying detection using SVM, DistilBERT and Stacked ensemble model on our newly generated social media dataset.
2. We conduct an empirical evaluation on different levels of granularity of feature extraction methods in TF-IDF such as Word, Character and N-gram sequencing on SVM model.
3. We perform and present the results of a comparative evaluation of the five developed models in terms of various evaluation metrics. The sets of models evaluated are:
 - (a) Traditional SVM model implemented using TF-IDF for Words.
 - (b) An improved SVM model proposed by Sharma et al.[5] combined with the tokens of Word, Character and N-gram in TF-IDF for feature extraction.
 - (c) DistilBERT model with classification layer on top.
 - (d) Stacked ensemble model by combining the base models explained in (a), (b) and the DistilBERT model in (c).
4. We present a detailed analysis on impact of these models on cyberbullying detection. In summary:
 - (a) The traditional SVM model with TF-IDF for words yields the worst accuracy and SVM model with different tokens of TF-IDF (i.e., Words, characters, and N-gram) yield accuracies similar to that of the DistilBERT model.
 - (b) The DistilBERT model yields the best precision.
 - (c) The ensemble models outperform all individual base models. Furthermore, when using combined tokens of TF-IDF with SVM and DistilBERT embeddings, we achieve an accuracy of 89.6%.

The rest of the paper is organized as follows: we briefly introduce the background in Section 2 and review the related work in Section 3. In Section 4, we present the data pre-processing steps

and explain the details on the models developed. We report on the analysis of our obtained results in Section 5 and make the conclusions in Section 6.

2. BACKGROUND

In this Section, we provide an insight to the technical information on the methodologies which are relevant for the text classification approaches. The major topics discussed in Section 2.1 is Feature extraction. Section 2.2 describes the TF-IDF used in traditional machine learning algorithms. The DistilBERT is described in Section 2.3.

2.1. Feature Extraction

Feature extraction is a method by which raw data, of any formats such as text, image, video is transformed into an acceptable internal representation or a feature vector from which any learning sub-system such as a classifier, can identify input patterns [6]. Feature extraction is considered a critical step in cyberbullying for text classification [7]. The basis of an enormous amount of text processing is the text feature extraction, in which the text information is extracted to represent a text message [6]. An important factor in classifying texts, according to the machine learning models is to digitize them [8]. The machine learning classifiers are trained using the numerical format of the input data. By applying various feature extraction techniques, every text information needs to be converted into a numerical representation. The dimension of a feature space is reduced by means of feature extraction[9]. Redundant and uncorrelated information will be deleted through feature extraction. The reduction of features will assist in improving the accuracy of the algorithms and hence speeds up the processing time. Text feature extraction directly influences the accuracy of text classification. The text feature extraction is based on the vector space model, and the text is observed as a dot in N-dimensional space. The common methods of text feature extraction are Filtration, Fusion, Mapping.

2.1.1. Filtering Method

Filtering method faster and is suitable for extensive text feature extraction. Filtration of text feature extraction comprises of word frequency, information gain, and mutual information method [9].

1. **Word frequency:** Word frequency is defined as the number of times a word appears in a text. To reduce the dimensionality of feature space using feature selection, words whose frequencies which are less than a certain threshold are deleted. The deletion criteria are based on a hypothesis that words with small frequencies will have a less impact on filtration. In terms of information retrieval, the words with less frequency of occurrences may have more information. Thus, it may be unseemly to remove the words only based on the word frequency.
2. **Mutual information:** MI (mutual information) is a commonly used method for mutuality in the analysis of computational linguistics models. MI helps to retrieve the differentiation of features. MI represents the relationships between information and the statistical measurement of correlation of two random variables. MI helps to create a table of association of words from a large corpus. If a feature belongs to a class, it is said to have largest amount of mutual information. A drawback of MI is that the score is regulated by the marginal probabilities of words.
3. **Information gain:** IG (information gain) is employed in machine learning to measure whether a known feature appears in a text of a certain applicable topic and the prediction rate of the information on the topic. The features that occur frequently in positive or

negative samples can be obtained by computing IG. The IG is computed on each feature based on the training data and deletes those features which has small information gain, and the remaining features are ranked in descending order based on the IG.

2.1.2. Mapping Method

1. Latent Semantic Index: Mapping has been used in text classification and has shown to achieve good results [9]. The commonly used mapping methods is LSI (latent semantic index). LSI is an algebraic model introduced in 1988 by S.T. Dumais. LSI reduces the dimensionality of text vectors by extracting and employing the latent semantic structure between words and texts. The mapping is achieved through SVD (singular value decomposition) of item or document matrix. LSI can be used in text classification, information extraction, information filtering.
2. Least squares mapping method is based on centre vector and least squares. The clustered centre vectors reflect the structures of raw data, whereas SVC did not consider these structures.

2.2. TF-IDF

TF-IDF is a combination of TF and IDF (Term Frequency and Inverse document frequency). The TF-IDF score indicates the relative importance of a specific term in any dataset[7]. The TF-IDF algorithm is based on word statistics for text feature extraction. TF-IDF is used to vectorize the input [1]. The model considers only the expression of words, that are similar in all texts. The TF-IDF is a commonly used feature extraction technique in text detection. A TF-IDF vector can be generated using different tokens such as words, characters, and n-grams.

- Word TF-IDF: Matrix representation of TF-IDF scores of words
- N-gram TF-IDF: Matrix representation of TF-IDF scores of n-grams, where n-grams are the combination of “n” words
- Char TF-IDF: Matrix representation of TF-IDF scores of character-level ngrams

2.3. Distil BERT

DistilBERT can be defined as a distilled version of BERT in which a compression technique termed as “Knowledge Distillation” is performed on a larger model- BERT to train a smaller model and to reproduce the behaviour of actual BERT model [10]. The actual larger model is termed as “the teacher” and the compact model is termed as “the student” in distillation mechanism. The architecture of DistilBERT is same as that of the transformer architecture, BERT, but to reduce the model size, a smaller number of layers is used. The token type embeddings and pooler are removed from DistilBERT (which BERT uses for the next sentence classification task). The Batch size was also changed from original BERT that led to an increase in performance. DistilBERT has relied on the same training data as that of BERT model. Three training losses was taken into consideration for DistilBERT namely Distillation loss, Masked Language Modelling loss (from the MLM training task) and Cosine embedding loss (to align the directions of the student and teacher hidden states vectors).

Triple losses ensure the DistilBERT model learns properly and has efficient transfer of knowledge. The distilled model has about half the total number of parameters of BERT base and retains 97% of BERT’s performances on language understanding capabilities. The DistilBERT model is 60% faster, and the model size was reduced by 40% when compared to the BERT model, has been constantly faster. The Parameter count of different pretrained language models is

depicted in Figure 2.4. The similarity in performances on various downstream tasks performed by DistilBERT and BERT was also validated. DistilBERT requires only a small computational training budget, while maintaining the flexibility of larger models. The DistilBERT models are small enough to run on platforms such as on mobile devices.

3. LITERATURE REVIEW

This Section provides a review of the existing literature on various text classification methodologies on different domains. Section 3.1 describes an overview of different datasets on which text classification has been performed. The machine learning algorithms implemented for text classification are discussed in Section 3.2. Section 3.3 highlights the works that have used Feature extraction techniques for classification.

3.1. Various Social Datasets

Engaging with the online platforms, people use social networks as a prominent way for expressing their opinion about an issue or presenting their experiences about an experienced product or service from a company. The data posted on these networks make users potentially vulnerable or abusive, which results in cyberbullying. Instagram, Twitter, Youtube are the commonly used social media platforms. The datasets are usually collected by crawling the target social media using its Application Programming Interface (API). The commonly used datasets for cyberbullying detection are described below.

Raisi et al. [11] described Twitter as one of the public-facing social media platforms with high frequency of cyberbullying. To the best of our knowledge, Twitter is the most available source in the field of Natural Language Processing (NLP) for researchers since a large portion of reviewed papers have benefited from Twitter contents. One of the reasons that this social media is popular among researchers to check their proposed algorithm is that registered users can broadcast short posts (280 character per post) which are mostly textual posts providing a direct to the point source of data. Moreover, people can tweet on Twitter in different languages, so datasets for other languages than English may also be achieved through Twitter. Twitter daily use is increasing rapidly. Muneer et al. [12] mentioned that this platform raised many issues due to misunderstanding regarding the concept of freedom of speech meaning the users share their unfiltered opinion even if they have offensive contents. Thus, this platform is considered as a vital data source in the field of cyberbullying detection.

Instagram dataset is a mix modal dataset that contains text, video, and photo at the same time. It seems that Instagram dataset is not suitable for employing NLP techniques, but it is worth mentioning that NLP is not limited to text analysis. However, there are several info-graphic posts which can be analysed using text analysis and image processing. Although there are rules on Instagram for reporting the abusive and harsh posts, the posts' comments are good place for cyberbullying.

The Ask.fm is a question and answering social network where users can ask their burning question, anonymously or publicly. This social network became the largest QA network in the world in 2017 [13]. A subsample of Ask.fm dataset was used for evaluating the weak supervision model. They filtered the dataset by removing anonymous users' question-answers and the posts that contained only "thanks" word. Samghabadi et al. [13] collected a dataset which contained the full history of question-answer pairs for 3K users.

YouTube is an online video sharing social media. Although this social media is a suitable platform for sharing tutorials and informative videos, it is an open environment that each user can share different kinds of video with harsh contents such as racism videos, porn videos, and so on. This makes YouTube as a good source for researchers to evaluate their detection models on. Bruwaene et al. [14] used two datasets for evaluating their model. They chose about 11,000 posts from VISR dataset which is a dataset from SafeToNet application, an application for parents to control their children's account in different social media. This dataset contains randomly chosen posts from six social media including YouTube. It has 7188 posts from total 603,379 posts. A hashtag collection and then crawled YouTube using the list of hashtags to download posts which are related to selected hashtags.

Wikipedia is a well-known, free content, and multilingual encyclopaedia. Volunteers can edit the texts using wiki-based system. The editors can share their opinion and discuss about improvements of articles in an environment named Wikipedia Talk pages. These pages are associated with each article in the form of "Talk: Article's name". Editors post their messages as new thread and other can share their view about the issue. These threads may be a potential environment for cyberbullying between the editors. Existing works used the Wikipedia talk pages dataset which were collected by Wulczyn et al. [15]. The dataset was gathered by processing the public dump of full history of English Wikipedia. The corpus contains 63M comments from talk pages for the articles dating 2004-2015. The labelled dataset has about 14000 comments which is labelled as personal attacks. Gada et al. [16] used the Toxic Comment Classification Challenge dataset which is a Wikipedia comments dataset labelled by human for toxic behaviour. The dataset has around 1.6M rows.

3.2. Cyberbullying Detection Using Machine Learning Algorithms

In this section, the machine learning algorithms which mostly used in cyberbullying detection are reviewed. The recent literature accounts the use of different machine learning and deep learning algorithms for detecting the hate speech, harsh contents including the pornography and abusive languages.

The most popular machine learning in text classification is linear SVM as the most text analysis problems are linearly separable. Moreover, the significant characteristic of SVM is that it can be learnable with any number of features. Thus, as the texts have lots of features, this algorithm is appropriate choice for their classification problems. Hani et al. [17] compared two supervised machine learning algorithms which are SVM and CNN on two different types of features namely Term Frequency- Inverse Document Frequency (TFIDF) and Sentiment Analysis features. Like other approaches, they aimed to have a machine learning model for detecting the harassments in a text data, so their model followed the three main steps: pre-processing, feature extraction, and classification in which they used Support Vector Machine (SVM) as the machine learning algorithm. Besides the TFIDF features, they used N-Gram as the feature extraction method and for the sentiment features, they used Text Blob Library which is a pre-trained model on movie reviews. The results showed that SVM gets highest accuracy in 4-Gram while NN gets highest accuracy in 3-Gram. However, in average of n-Gram, NN works better than SVM. Kumar Sharma et al. [5] experimented different methods to identify bully content in a text and find the best classifier in this way. Among the four classifiers that they used SVM was the second one in terms of AUC score. Soni et al. [18] instead of doing research on only text data, they implemented an audio-visual-textual cyberbullying detection platform. They used 5 different machine learning algorithms including SVM for detecting cyberbullying in audio, visual, and textual features. The results showed that the proposed approach which applied the machine learning algorithms on multi modal features (Audio+Visual+Textual) compared to applying proposed approach on all comments achieved about 2.75% decrease in F1 score. The lowest F1

score in all features belongs to SVM, which means this algorithm is not suitable for multi modal cyber bullying detection. As a hot topic in this field is detection of bullies in different languages. Leon-Paredes et al. Authors of [19] developed an online prevention tool for detecting cyberbullying in Spanish language. They used three different classifiers based on the characteristic of algorithms namely Naïve Bayes, SVM, and Logistic Regression on three different size of dataset which are small corpus, medium corpus, and large corpus. They measured accuracy, average precision, and F1 score as the evaluation metrics for a total 90 executions. The results showed that the average precision of the detection was between 80% to 91%, however, SVM got the best accuracy of 93% on the medium corpus at the training rate of 10%. In addition, Nurrahmi et al. Authors of [20] proposed a cyberbullying actors detection system based on the reliability analysis of the users for notifying them about their offensive content in Indonesian language. They classified the tweets based on normal behaviour and abnormal behaviour and then used the number of bully and non-bully tweets for each user to calculate the probability of user's behaviour so that they can use this probability in finding the reliability of the user. They categorized the users in four groups based on the probability of their behaviour: if the probability is less than 50% then user is normal, and if the probability is equal and more than 50% then the user lies under bullying actors. Their web-based tool used SVM as one of the two machine learning algorithms and tried it using two techniques, linear and RBF, to recognize whether the dataset is fitted to linear function or non-linear function. The results showed that SVM got higher F1 score than KNN algorithm, and between linear kernel or RBF kernel in SVM, the RBF with C=4 achieved the highest F1 score.

3.3. Feature Extraction on Cyberbullying Detection

The usage of social media platforms such as Facebook, WhatsApp, Twitter, and Instagram had increased over the past years [21]. A huge amount of data is transferred through these platforms among users which also includes obfuscated content and hateful words. The data contributing to cyberbullying could be of different formats such as text, images, and videos. Every dataset is comprised of features, which could be considered as variables. The data analysis, prediction, and classification are dependent on these features. The accuracy of any machine learning algorithm relies upon the features that have been used for training the models. The datasets are expanding with various features in the cyberworld, and this increases the challenge of selecting features for prediction. The quality of a dataset can be improved by optimizing features and hence feature extraction plays a vital role, as it helps in defining complex datasets with a reduced number of features. The feature extraction methods play an important role in improving the accuracy of the different Machine learning algorithms used to identify cyberbullying. The performance of cyberbullying detection using classifiers could be improved by using text-based features instead of non-text-based features such as image and network graph [2].

The different data types contribute different features that are used for cyberbullying prediction. A major classification of features includes content, user, sentiment, and network-based features [5]. Feature extraction methods implemented on any dataset depend on the data type. The content-based features could be further classified as profanity, negativity, and subtlety [22]. Negativity and Profanity seems to appear among most of the cyberbullying instances [23]. Special features could be further used to predict the label that includes Sexuality, Intelligence, and Race.

The most identified cyberbullying involves the usage of text data types irrespective of the social media platforms. Text data types consist of the negative connotation, profane words, context related to minority races, physical characteristics, religion [3]. The textual features help in improving the analysis of cyberbullying content includes the density of inappropriate words, number of special characters such as question mark and exclamation, the density of upper-case letters, number of smileys and part of speech tags. A combination of features was identified to

detect cyberbullying in Youtube comments [11]. Online user-based features, cyberbullying-specific features, content-based features were used to identify cyberbullying in social network videos that include Youtube user comments.

TF-IDF is a commonly used feature extraction technique in text detection. Dinakar et al. In [23], the authors used TF-IDF on multiple machine learning algorithms to compare the accuracy of cyberbullying detection on Form spring and Youtube datasets. The feature extraction method was used in predicting the accuracy of the model generated using SVM on MySpace, Slashdot, Kongregate by Raisi et al. [11]. The accuracy prediction of cyberbullying detection on Turkish language was performed by obtaining TF-IDF properties [8]. As an initial approach to get the baseline model, Gada et al. [16] used TF-IDF on simple classification techniques. Among different feature engineering techniques carried out in the early detection of cyberbullying, the lexical features were weighted using TF-IDF [13]. The TF-IDF vectors generated using different levels of input tokens such as Word TF-IDF, N-gram TF-IDF and Char TF-IDF was used by Chen et al. [7] to compare their HANCD model with baseline models such as KNN, Random Forest, Naive Bayesian, XGBoost and Logistic Regression. Chen et al. [7] identified that TF-IDF vectors was more effective when compared to the pre-trained word embedding technique, Glove. Sharma et al. [5] created a Machine learning model by extracting all the feature vector sets and stacked them to a single feature set. Word and characters were taken as token for TF-IDF feature extraction. Features were extracted using TF-IDF along with sentiment analysis to design the cyberbullying detection model designed by Hani et al. [17]. Analysis of tweets to identify bully and non-bully tweets were performed using TF-IDF vectorization. TF-IDF is a simple and proven method in text classification [7].

DistilBERT pre-trained language model is built by leveraging the knowledge distillation on BERT models. The DistilBERT models are lighter and has a faster inference time. This recently released pre-trained language model is getting popular and researchers are working to exploit its capabilities on various downstream tasks.

Herath et al. [1] developed and evaluated a cyberbullying classification model using DistilBERT and state-of-the-art NLP technology. The dataset collected from Twitter for the SemEval 2019-Task 5 (HatEval) challenge was utilized for the study. The addressed problem in this challenge was to identify cyberbullying against Women and Immigrants. To identify cyberbullying, three classification models, each built on DistilBERT along with a classification layer was developed. The three models were built by changing the ratio of positive and negative classes as explained below:

1. Model A: Training data was imbalanced, and majority class was positive.
2. Model B: Training data was imbalanced, and majority class was negative.
3. Model C: Training data was balanced.

All the three models mentioned above were ensembled using a Simple Voting Classifier to predict the results. This ensemble model achieved a result of 0.41% F1-Score.

Ratnayaka et al. [24] implemented DistilBERT in identifying cyberbullying detection through role modelling. Ask.fm dataset was utilized to categorize the participant roles into victim and harasser, which is a multi-class classification problem. The evaluation of cyberbullying classification was done based on the model developed by Herath et al.[1] as explained above, where in three models where ensembled in which each model was fed with a training dataset in which the majority class was positive, negative, and balanced. The Twitter dataset was used to evaluate this model, in which the tweets were categorized into “Offensive” and “Not Offensive”. This ensemble model achieved an accuracy of 0.906 on F1 score.

4. OUR PROPOSED CYBERBULLYING DETECTION APPROACHES

In the following section, we first present the data processing steps performed. Then in subsection 4.2 we present the modified SVM models. The two SVM models were developed using different tokens of TF-IDF vectors. The proposed DistilBERT-based model is presented in subsection 4.3. The Ensemble models of stacking the base models are explained in detail in subsection 4.4.

4.1. Data Pre-processing

Data pre-processing plays a major role in developing any machine learning model, as the model performance relies on the data input. It is an important step in cleaning the data before feeding them to any model, to avoid any error during training. The NLTK library is commonly used to perform the pre-processing tasks such as tokenization, lemmatization removing stop words and unwanted characters, stemming the raw data. The type of data pre-processing required depends on the task for which the models are developed. The blank rows were removed, and the text case was converted to lower case. This was followed by the tokenization, word-stemming, and lemmatization process. The stop words were not removed in our cyberbullying detection task, because the important indicators for cyberbullying detection could be the second and third nouns.

4.2. Applying Two Different Length of Feature Extraction Tokens on SVM

Two different feature extraction tokens were used to implement the enhanced SVM model. The models differed in terms of TF-IDF vectors fed into the classifier. The SVM Model 1 utilized the TF-IDF for words, and the SVM Model 2 was built using the TF-IDF vectors of Words, Characters, and N-gram.

4.3. Applying Word Embeddings: DistilBERT

The raw data was pre-processed. The processed data saved to a data frame. The data is split into Training Set, which constitutes 80% of the entire dataset and Testing set, that contains 20% of the remaining data. The DistilBERT Tokenizer and DistilBERT model is loaded. The Training dataset is fed to the DistilBERT Tokenizer. The Tokenizer converts the raw data into a format that DistilBERT can process. The DistilBERT Tokenizer performs below actions to prepare the input to the model:

1. Tokenizer transforms the sentence's words into an array of DistilBERT tokens.
2. Adds a special starting token ([CLS] token) to the above generated sequence.
3. Adds the necessary padding to have a unique size for all sentences (we used the maximum length value as 32).

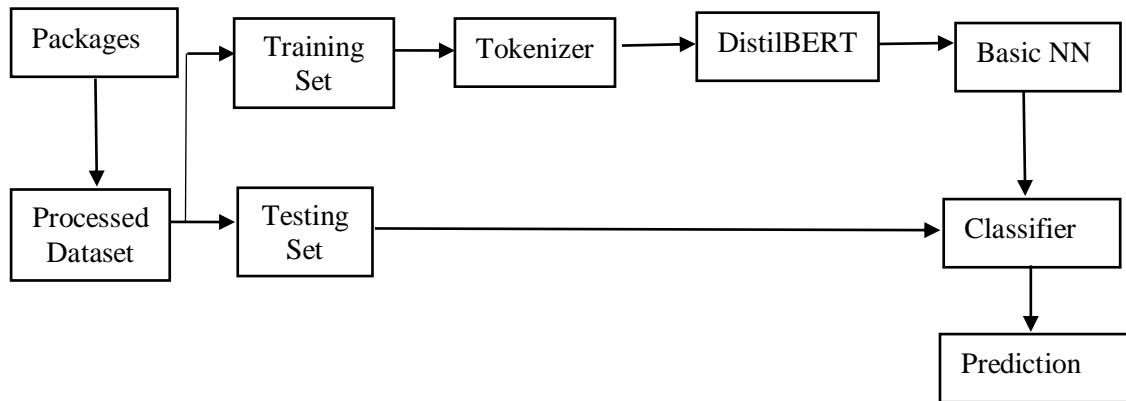


Figure 1. The DistilBERT Model

The output from the DistilBERT Tokenizer contains input IDs, Attention masks and Special Tokens. This is fed to the DistilBERT fine-tuned model. The Trained DistilBERT model was used to generate the sentence embeddings. The output of this model is a vector of length 768 (default length).

To utilize this output from the pre-trained DistilBERT embedding model for cyberbullying detection, a basic neural network architecture with Dense and Dropout layers is implemented. This layer gets the input from the DistilBERT transformer and produces a vector, that is used for predictions in classification tasks. The model was trained for 3 epochs. Adam was used as the optimizer for the model. Since the samples belong to exactly one class, the Sparse Categorical Cross entropy is used to estimate the loss calculation. The block diagram of the DistilBERT Model developed is illustrated in Figure 1.

4.4. Stacked Ensemble Models

We have developed two models that are based on two different approaches: the enhanced SVM is based on the textual features of the data, and the DistilBERT word embeddings is based on the ability of language understanding capabilities of NLP transformers. These heterogeneous models are combined using the Stacking Ensemble method for the classification task. In the ensemble model, a meta-learning classification algorithm is used to combine the predictions from the two base models, SVM model and DistilBERT model. Since stacking model has the ability to exploit the potential of various well-performing models, it was chosen to make predictions on the cyberbullying detection task, expected to exhibit better performance than the individual base models.

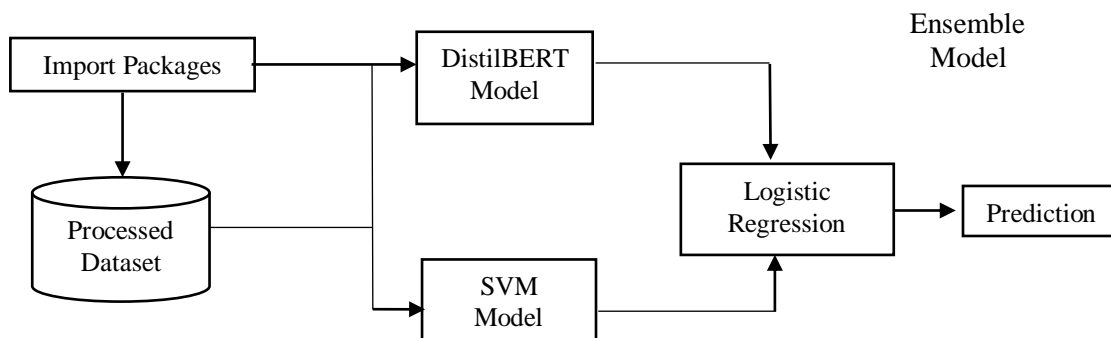


Figure 2. The General Ensemble Model Architecture

Figure 2 represents the general stacked model architecture. Cyber-bullying detection is a binary classification problem, and the input features are independent. Hence, the Logistic Regression model is used as a meta-model for classification of cyber-bullying content.

5. EXPERIMENTAL RESULTS ANALYSIS

5.1. Evaluation Metrics

The potential of any model can be evaluated using few metrics which helps in determining the ability of a model to differentiate texts as cyberbullying or not. To analyse the performance of models, it is important to examine the assessment metrics. The evaluation of models was performed based on various parameters such as Accuracy, Precision, Recall and F1-measure from the confusion matrix. Confusion matrix can be used to measure the performance of any machine learning classification problem.

The Accuracy of a model can be defined as the ratio of the number of correct predictions against the total number of predictions made. The Accuracy can be estimated using below formula.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

The Precision of a model is determined as the proportion of predicted positive cases to the total predicted positives. It helps us to calculate the ratio of relevant data among true positive (TP) and false positive (FP) data belonging to a specific class.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Recall can be defined as the proportion of Real Positive cases that are correctly Predicted Positive.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

F1-Score is the weighted average of Precision and Recall. F1 Score is calculated using below formula. F1-score helps to combine precision and recall into a single measure.

$$\text{F1Score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

5.2. Impact of Feature Extraction on SVM models

Muneer et al. [12] performed a comparative analysis of various machine learning models for cyberbullying detection on twitter dataset. The dataset used was relatively smaller in size when compared to other works and was similar to the smaller dataset used in our work. The work employed the TF-IDF vectorization for feature extraction as applied in this thesis. The SVM model developed by utilizing TF-IDF features exhibited a lower accuracy of 67.13% and a precision of 0.67. These metrics were much lower when compared to our results, where in an accuracy of 85.53% and precision value of 0.86 was achieved. Salminen et al. [25] conducted an analysis of different classifiers using a combined dataset which was extracted from different social media platforms such as Youtube, Twitter, and Wikipedia. The generated dataset had a

class imbalance of 1:4, in which most of the data samples had non-cyberbullying content. Due to the similarity in the generation of dataset and the class imbalance exhibited in the study, a comparison of results was performed using the results obtained from this paper. The study was done using different stand-alone feature extraction methods such as TF-IDF and BOW, word embedding techniques such as Word2Vec and BERT, simple features such as punctuation and use of upper-case characters etc in combination with individual ML algorithms such as LR, NB, SVM, XGBoost etc. F1-score was used as an evaluation measure in this study. SVM model exhibited an F1-score of 64.8% with TF-IDF vectorization, which was clearly less compared to the results obtained in our study, where we obtained an F1-score of 71.48%.

In addition to the above comparisons, due to the similarity in the dataset features and source of extracted data, the performance of SVM model was compared with the machine learning model developed by Sharma et al. [5]. The dataset was extracted from sources such as UCI, Twitter and Kaggle. The extracted dataset was pre-processed and labelled resulting in a final set with columns Date, Comment and Label. This is similar to the dataset used for this work, though we have limited features and our dataset had contents extracted from Twitter.

Two SVM models were implemented using different tokens of extracted TF-IDF vectors to understand the impact of feature selection on traditional SVM models. The initial model was based on the simple TF-IDF word tokens which was fed to the SVM model. The second model was built by using the various tokens of words, characters, and N-grams of TF-IDF vectors into the SVM model.

5.2.1. Evaluating Different Feature Extraction Token Sizes on SVM Models

Different N-gram word tokens were tested on the SVM-TF-IDF model. The N-gram range chosen was between 1 and 7. We identified that, with an increase in the word n-grams, the accuracy was decreasing. The model performed better when the N-gram was set as (1,1) and resulted in an accuracy of 85.53%. The corresponding accuracies of different N-grams are listed in Table 1.

Table 1. Comparison of different word tokens.

Word N-gram	Accuracy (in %)
1	85.53
2	85.28
3	85.22
4	85.39
5	85.33
6	85.37
7	85.13

An analysis was done by changing both the word and character tokens using N-gram. A unigram character token was also used by default in addition to the other two tokens. The word and character tokens were tested for different N-gram values within the range of 1 to 7 to determine the impact of the increase in token size on accuracy. The accuracy was dropping when both the word and character token sizes were increased simultaneously. The comparison of accuracies on different word and character tokens is illustrated in Table 2.

Table 2. Comparison of different word and character tokens.

Word N-gram	Character N-gram	Character Unigram	Accuracy (in %)
2	2	1	87.03
3	3	1	88.02
4	4	1	87.83
5	5	1	87.56
6	6	1	87.26
7	7	1	86.86

Hence, to understand the impact of “Character” tokens, the unigram of word token was considered, and the experiment was performed by changing only the “Character” token sizes. The unigram of character token was used in combination with the word unigram and character N-gram. The results obtained from changing the token sizes are illustrated in Table 3. The best accuracy was achieved when the N-gram was set as (1,5) for the character token. Hence the feature vector for the base model was created using a combination of unigram character, unigram word token, and an N-gram of (1,5) for character tokens.

Table 3. Comparison of different character tokens, word and character unigram.

Word Unigram	Character N-gram	Character Unigram	Accuracy (in %)
1	2	1	88.02
1	3	1	88.04
1	4	1	88.05
1	5	1	88.1
1	6	1	88.03
1	7	1	88.06

The accuracy of the SVM Model 1 achieved was 85.53%. This accuracy was achieved by implementing the TF-IDF vectorization on words. The model exhibited a better Recall and F1 score was achieved for Class 0 data. The accuracy of the SVM Model 2 achieved was 88.1%. This accuracy was achieved by implementing the TF-IDF vectorization on words, character and N-gram tokens. The Evaluation metrics comparison of both SVM models is illustrated in Table 4. Figure 3 represents the Confusion matrix of SVM models 1 and 2 respectively.

Table 4. Comparison of two SVM models.

Model	Parameters			
	Accuracy	Precision	Recall	F1-Score
SVM: TF-IDF of Words	85.53	82.05	63.33	71.48
SVM: TF-IDF of Words, Characters and N-gram	88.1	84.38	71.71	77.53

The increase in accuracy of the SVM model 2 can be attributed towards the combination of different granularity of features. The results also showed a drastic increase in the Recall and F1 scores. Thus, combining different tokens prove to perform better on traditional SVM algorithms, than relying on a single feature set.

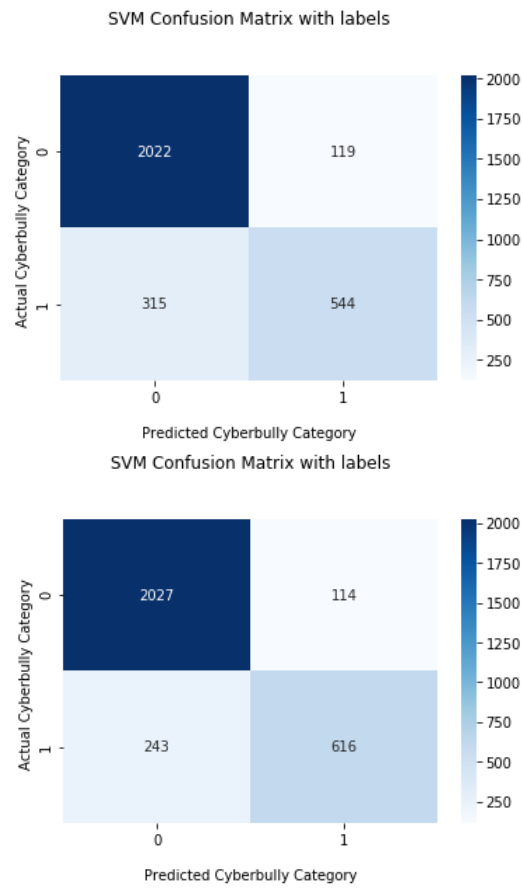


Figure 3. Confusion matrix of SVM models 1 and 2 respectively

5.2.2. Evaluating Different Length of Feature Extraction Tokens on SVM against Existing Work

The analysis of the existing work is done by comparing the developed models based on similarities such as the feature extraction method and ML algorithm.

A summary of the comparison of the related work based on TF-IDF vectors is illustrated in Table 5.

Table 5. Comparison of Related Studies.

Authors	Feature	N-gram	Classifier
Muneer et al. [12]	TF-IDF: Words	Unigram	SVM
Salminen et al. [25]	TF-IDF: Words	Unigram	SVM
Sharma et al. [5]	TF-IDF: Words, Characters & N-gram	(1,5) Character	SVM
In this paper	TF-IDF: Words	Unigram	SVM
In this paper	TF-IDF: Words, Characters & N-gram	(1,5) Character	SVM

For comparing the results with the model developed by Sharma et al. [5], the dataset used for this paper was deployed on their model which we developed based on their work. The model was developed by generating TF-IDF vectors of three types. The TF-IDF vectors of both words and characters as tokens along with an n-gram sequencing from 1 to level 5 was generated. The extracted feature vectors were stacked into a single set. This stacked set of features were divided into training and test data sets. The SVM model trained using the stacked feature set resulted in an accuracy of 88.1%. The results of the SVM Model with TF-IDF word tokens was compared with the model developed by Sharma et al. [5]. The baseline model outperformed in this scenario compared to the SVM Model with TF-IDF word tokens, which was based on only word vectors that was developed in this paper. This increase in performance could be due to the combination of different extracted feature vectors. Due to the high performance of the traditional SVM using different tokens of TFIDF such as Words, Characters and N-gram sequencing, we have chosen this as the base model instead of the simple SVM with TF-IDF for words.

5.3. Comparative Evaluation of DistilBERT Model on Cyberbullying Detection

DistilBERT pre-trained language model developed by Sanh et al. [10] is an emerging word-embedding technique, that uses lesser number of parameters when compared to the existing BERT embeddings. Researchers are implementing DistilBERT in many downstream tasks and few works has focused on using DistilBERT for Cyberbullying detection. The classification model used by Herath et al. [1] to identify cyberbullying against Women and Immigrants uses DistilBERT. Three models which were built on a Training dataset by changing the ratios of the majority classes acts as base models. The final ensemble model was built using a Simple Voting Classifier.

Since the dataset used in this research is comparatively balanced, a simple DistilBERT model was developed, with a classification layer on top. Reducing the number of DistilBERT models reduces the training time. This model provided an accuracy of 87.53% and the highest precision of 91.17%. This model was slightly better when compared to traditional SVM with TF-IDF in terms of accuracy. The Processing time of training DistilBERT model was 30 minutes for the 3 epochs. The DistilBERT exhibited the maximum training time when compared to all other models developed. The better recall and f1 scores were exhibited for class 0 data in DistilBERT models as well. In addition to that, this model outperformed in terms of Precision for bullying content. The Confusion matrix generated for the DistilBERT model is shown in Figure 4.

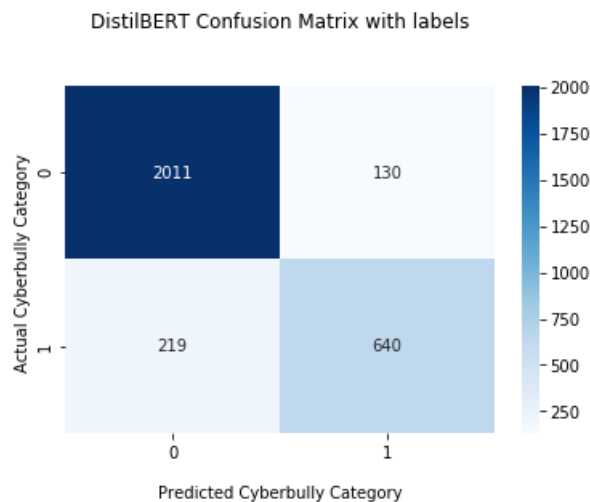


Figure 4. Plot of DistilBERT Confusion Matrix

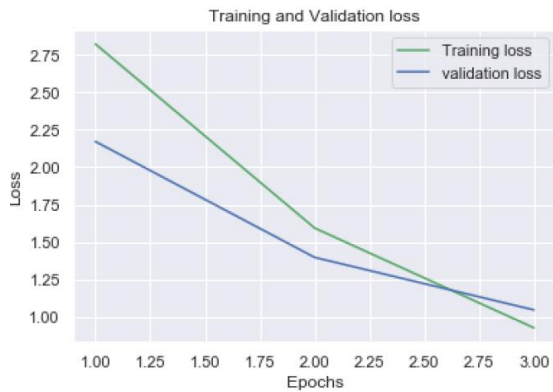


Figure 5. Training and Validation Loss

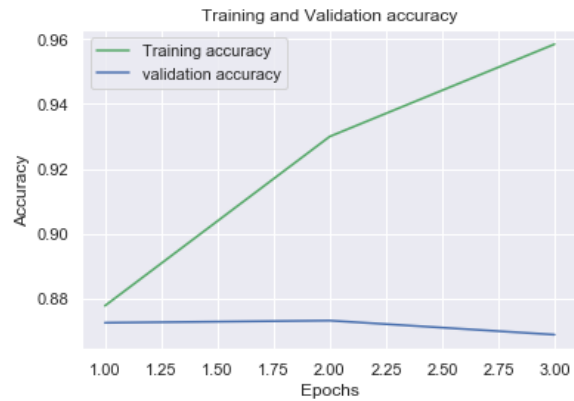


Figure 6. Training and Validation Accuracy

The training and validation loss of three epochs in DistilBERT model is plotted in Figure 5. Figure 6 represents the training and validation accuracy for the three epochs. The training and validation losses drastically reduced with the increase in the number of training epochs. The training accuracy improved significantly over the epochs, however, there was a slight dip in the validation accuracy at the end of the third epoch.

5.4. Proposed Ensemble Models

We developed and analyzed the performance of two ensemble models. The initial ensemble model is using the traditional SVM with the TF-IDF for words and the DistilBERT model. The second ensemble model is developed using the combined feature extraction levels of TF-IDF using different tokens such as word, character, and n-gram on SVM and the DistilBERT model. More details of these two ensemble models are explained below:

5.4.1. Ensemble Model 1: Ensemble Using SVM (TF-IDF for Words) and DistilBERT

This ensemble model is built using SVM and TF-IDF for words along with the DistilBERT model. It yields an accuracy of 88.3%. The Recall and F1 score of this ensemble model were much better while compared to the base SVM and DistilBERT models. The Confusion Matrix of the Stacked ensemble model 1 is shown in below Figure 7. The ensemble model 1 outperformed the base models in terms of evaluation metrics except for precision.

5.4.2. Ensemble Model 2: Ensemble Using SVM (TF-IDF For Words, Characters, and n-gram) And Distilbert

The accuracy of the second ensemble model developed using SVM along with the different tokens of TF-IDF vectorization such as words, characters, n-gram and the DistilBERT model is over 90%. The Confusion Matrix of the Stacked ensemble model 2 is shown in Figure 8. The increase in accuracy is due to the efficient base models. The Base model 1 has taken into consideration the different granularity of TF-IDF vectors and the word embeddings in the DistilBERT model accounted for the better performance.

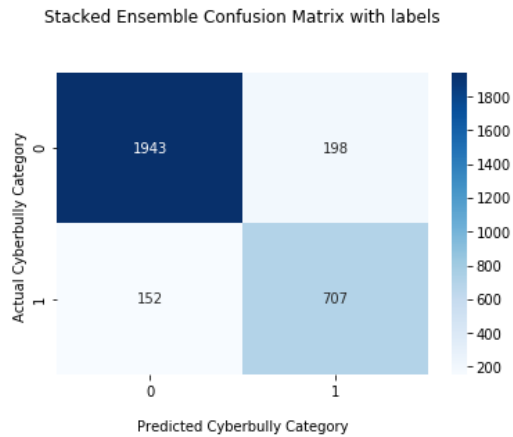


Figure 7. Ensemble model 1

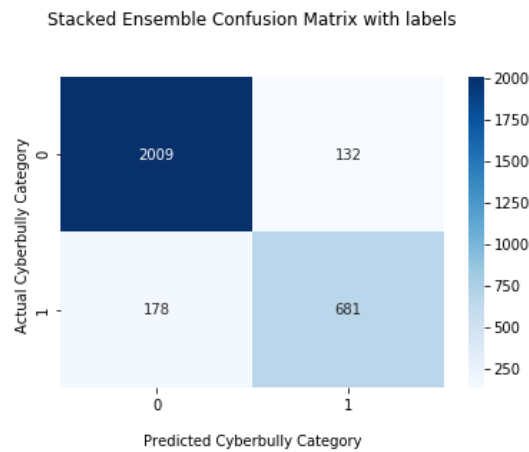


Figure 8. Ensemble model 2

5.5. Performance Comparison of All Models

A summary of the performance of all the five models in terms of various evaluation metrics is represented in Table 6.

Table 6. Model Parameters.

Model	Parameters			
	Accuracy	Precision	Recall	F1-Score
SVM: TF-IDF of Words	85.53	82.05	63.33	71.48
SVM: TF-IDF of Words, Characters and N-gram	88.1	84.38	71.71	77.53
DistilBERT	87.53	91.17	62.51	74.17
Ensemble: Model 1	88.3	78.12	82.30	80.15
Ensemble: Model 2	89.66	83.76	79.27	81.45

DistilBERT model exhibited the best precision when compared to all the other models, however ensemble models outperformed in terms of all other parameters. The highest accuracy of 89.66% is exhibited by the Ensemble model 2 among all other models. This model also yields the best F1-score of 81.45%. Among both SVM models developed, SVM model 2 outperforms the other

in all aspects due to the combined features fed into the model. Thus, with the increase in levels of tokens fed as features, traditional models perform better when compared to models built using a single set of features. The accuracy of this SVM model is very similar to the proposed DistilBERT model, which had inbuilt word embeddings. Thus, the addition of different tokens as features acts as a substitute for the word embeddings, while implemented on smaller datasets.

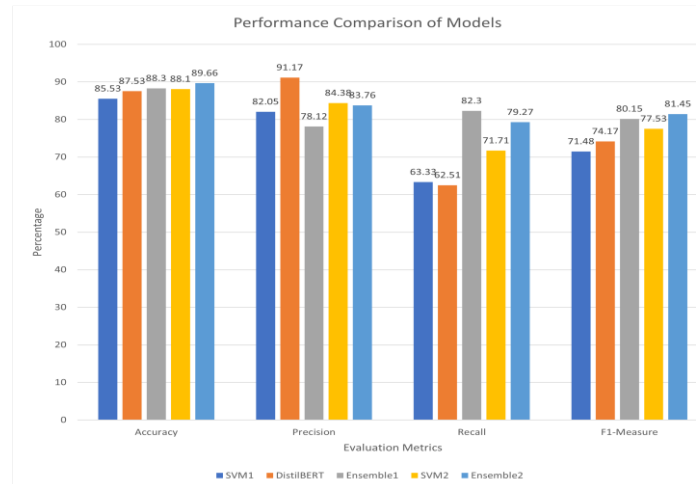


Figure 9. Performance Comparison of all models

The ensemble models can be used to develop systems that can predict the cyberbullying with better accuracy and exhibits better performance than individual base models. The Performance comparison of all the models is illustrated in Figure 9. All the models exhibited better accuracies and had slight improvements while implemented using word embeddings, the increase in features tokens and ensembling the models. We observed fluctuations in the Precision and Recall of various models. The lowest precision was demonstrated by the Ensemble model 1. However, the ensemble model 1 exhibited the highest Recall when compared to other models. Both ensemble models achieved a better F1-score than all the individual base models.

6. CONCLUSION

The impact of cyberbullying is dramatically increasing due to ease of access to Internet. This results in psychological and physical harm to victims. There are several systems available to tackle cyberbullying. This work identifies three different models for cyberbullying detection using a newly generated dataset that was extracted from the Enron email dataset, Twitter parsed data and Youtube parsed data from the Mendeley Cyberbullying dataset. These models were based on traditional machine learning algorithms and recent state-of-the-art word embeddings that consists of a single neural layer on top. We have also introduced an ensemble model using a stacking method for combining two base models which were based on completely different approaches to leverage the performance.

The evaluation of the proposed ensemble models shows good performance in cyberbullying detection. The traditional machine learning models require feature extraction techniques for better performance; however, the DistilBERT word embeddings have inbuilt tokens and do not require any explicit tokenization. The traditional SVM models were based on TF-IDF feature extraction of words and a combined TF-IDF vectors of words, characters, and N-gram. The experiment results indicated that the SVM model with the combined vectors outperformed the simple SVM-TF-IDF model. The DistilBERT exhibited the best precision of 91.17%. The

Stacked ensemble models outperformed the base models in terms of Accuracy, Recall and F1-Score. The Ensemble model using the combined vectors along with SVM and the DistilBERT model had the best accuracy of 89.6%.

REFERENCES

- [1] T. Atapattu, M. Herath, G. Zhang, and K. Falkner, "Automated Detection of Cyberbullying Against Women and Immigrants and Cross-domain Adaptability," *arXiv:2012.02565 [cs]*, Dec. 2020, Accessed: Feb. 10, 2022. [Online]. Available: <http://arxiv.org/abs/2012.02565>
- [2] A. K. Goodboy and M. M. Martin, "The personality profile of a cyberbully: Examining the Dark Triad," *Computers in Human Behavior*, vol. 49, pp. 1–4, Aug. 2015, doi: 10.1016/j.chb.2015.02.052.
- [3] K. R. Purba, D. Asirvatham, and R. K. Murugesan, "A Study on the Methods to Identify and Classify Cyberbullying in Social Media," in *2018 Fourth International Conference on Advances in Computing, Communication & Automation (ICACCA)*, Subang Jaya, Malaysia, Oct. 2018, pp. 1–6. doi: 10.1109/ICACCAF.2018.8776758.
- [4] W. N. Hamiza Wan Ali, M. Mohd, and F. Fauzi, "Cyberbullying Detection: An Overview," in *2018 Cyber Resilience Conference (CRC)*, Putrajaya, Malaysia, Nov. 2018, pp. 1–3. doi: 10.1109/CR.2018.8626869.
- [5] H. Kumar Sharma, K. Kshitiz, and Shailendra, "NLP and Machine Learning Techniques for Detecting Insulting Comments on Social Networking Platforms," in *2018 International Conference on Advances in Computing and Communication Engineering (ICACCE)*, Paris, Jun. 2018, pp. 265–272. doi: 10.1109/ICACCE.2018.8441728.
- [6] K. R. Talpur, S. S. Yuhaniz, N. N. B. Amir, B. Ali, and N. B. Kamaruddin, "CYBERBULLYING DETECTION: CURRENT TRENDS AND FUTURE DIRECTIONS," . *Vol.*, no. 16, p. 12, 2005.
- [7] L. Cheng, R. Guo, Y. Silva, D. Hall, and H. Liu, "Hierarchical Attention Networks for Cyberbullying Detection on the Instagram Social Network," p. 10, 2019.
- [8] E. C. Ates, E. Bostanci, and M. S. Güzel, "Comparative Performance of Machine Learning Algorithms in Cyberbullying Detection: Using Turkish Language Preprocessing Techniques," p. 19.
- [9] H. Liang, X. Sun, Y. Sun, and Y. Gao, "Text feature extraction based on deep learning: a review," *J Wireless Com Network*, vol. 2017, no. 1, p. 211, Dec. 2017, doi: 10.1186/s13638-017-0993-1.
- [10] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," *arXiv:1910.01108 [cs]*, Feb. 2020, Accessed: Dec. 04, 2021. [Online]. Available: <http://arxiv.org/abs/1910.01108>
- [11] E. Raisi and B. Huang, "Cyberbullying Identification Using Participant-Vocabulary Consistency," *arXiv:1606.08084 [cs, stat]*, Jun. 2016, Accessed: Nov. 14, 2021. [Online]. Available: <http://arxiv.org/abs/1606.08084>
- [12] A. Muneer, "A Comparative Analysis of Machine Learning Techniques for Cyberbullying Detection on Twitter," *Future Internet*, vol. 12, no. 11, p. 187, Oct. 2020, doi: 10.3390/fi12110187.
- [13] N. S. Samghabadi, A. P. L. Monroy, and T. Solorio, "Detecting Early Signs of Cyberbullying in Social Media," p. 6.
- [14] D. Van Bruwaene, Q. Huang, and D. Inkpen, "A multi-platform dataset for detecting cyberbullying in social media," *Lang Resources & Evaluation*, vol. 54, no. 4, pp. 851–874, Dec. 2020, doi: 10.1007/s10579-020-09488-3.
- [15] E. Wulczyn, N. Thain, and L. Dixon, "Ex Machina: Personal Attacks Seen at Scale," *arXiv:1610.08914 [cs]*, Feb. 2017, Accessed: Sep. 02, 2021. [Online]. Available: <http://arxiv.org/abs/1610.08914>
- [16] M. Gada, K. Damania, and S. Sankhe, "Cyberbullying Detection using LSTM-CNN architecture and its applications," in *2021 International Conference on Computer Communication and Informatics (ICCCI)*, Coimbatore, India, Jan. 2021, pp. 1–6. doi: 10.1109/ICCCI50826.2021.9402412.
- [17] J. Hani, M. Nashaat, M. Ahmed, Z. Emad, E. Amer, and A. Mohammed, "Social Media Cyberbullying Detection using Machine Learning," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 5, 2019, doi: 10.14569/IJACSA.2019.0100587.
- [18] D. Soni and V. K. Singh, "See No Evil, Hear No Evil: Audio-Visual-Textual Cyberbullying Detection," *Proc. ACM Hum.-Comput. Interact.*, vol. 2, no. CSCW, pp. 1–26, Nov. 2018, doi: 10.1145/3274433.

- [19] G. A. Leon-Paredes *et al.*, “Presumptive Detection of Cyberbullying on Twitter through Natural Language Processing and Machine Learning in the Spanish Language,” in *2019 IEEE CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON)*, Valparaiso, Chile, Nov. 2019, pp. 1–7. doi: 10.1109/CHILECON47746.2019.8987684.
- [20] H. Nurrahmi and D. Nurjanah, “Indonesian Twitter Cyberbullying Detection using Text Classification and User Credibility,” in *2018 International Conference on Information and Communications Technology (ICOIACT)*, Yogyakarta, Mar. 2018, pp. 543–548. doi: 10.1109/ICOIACT.2018.8350758.
- [21] V. Krithika and V. Priya, “A Detailed Survey On Cyberbullying in Social Networks,” in *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*, Vellore, India, Feb. 2020, pp. 1–10. doi: 10.1109/ic-ETITE47903.2020.031.
- [22] W. Medhat, A. Hassan, and H. Korashy, “Sentiment analysis algorithms and applications: A survey,” *Ain Shams Engineering Journal*, vol. 5, no. 4, pp. 1093–1113, Dec. 2014, doi: 10.1016/j.asej.2014.04.011.
- [23] K. Dinakar, B. Jones, C. Havasi, H. Lieberman, and R. Picard, “Common Sense Reasoning for Detection, Prevention, and Mitigation of Cyberbullying,” *ACM Trans. Interact. Intell. Syst.*, vol. 2, no. 3, pp. 1–30, Sep. 2012, doi: 10.1145/2362394.2362400.
- [24] G. Rathnayake, T. Atapattu, M. Herath, G. Zhang, and K. Falkner, “Enhancing the Identification of Cyberbullying through Participant Roles,” in *Proceedings of the Fourth Workshop on Online Abuse and Harms*, Online, 2020, pp. 89–94. doi: 10.18653/v1/2020.alw-1.11.
- [25] J. Salminen, M. Hopf, S. A. Chowdhury, S. Jung, H. Almerakhi, and B. J. Jansen, “Developing an online hate classifier for multiple social media platforms,” *Hum. Cent. Comput. Inf. Sci.*, vol. 10, no. 1, p. 1, Dec. 2020, doi: 10.1186/s13673-019-0205-6.

AUTHORS

Ms. Saranyanath is currently pursuing Masters in Computer Science at Carleton University. She holds a Bachelor of Electronics and Communication Engineering degree from Anna University India. She has 7 years of experience in Software Industry as Project Manager, Software Consultant and Business analyst. She specializes in Machine learning, Pattern recognition and Data analysis.



Dr Wei Shi is a Professor in the School of Information Technology, cross-appointed to the Department of Systems and Computer Engineering in the Faculty of Engineering & Design at Carleton University. She specializes in algorithm design and analysis in distributed environments such as Data Centres, Clouds, Mobile Agents and Actuator systems and Wireless Sensor Networks. She has also been conducting research in data privacy and Big Data analytics. She holds a Bachelor of Computer Engineering from Harbin Institute of Technology in China and received her Master's and Ph.D. in Computer Science from Carleton University in Ottawa, Canada. Dr Shi is also a Professional Engineer licensed in Ontario, Canada.



Dr. Corriveau received his Master's in Computer Science from University of Ottawa in 1984. During that time, he also worked at Nortel developing an industrial code generator. In 1986, after starting his Ph.D. at University of Toronto, he returned to Nortel becoming a founding member of the TELOS project. This tool spawn off as the ObjecTime start-up in early 1991 and eventually evolved into ROSE Real-Time, and now IBM Rational Software Architect Realtime Edition. Dr. Corriveau completed his Ph.D. in Natural Language Processing in 1991 and soon after joined the School of Computer Science at Carleton.



A DATA-DRIVEN ANALYTICAL SYSTEM TO OPTIMIZE SWIMMING TRAINING AND COMPETITION PERFORMANCE USING MACHINE LEARNING AND BIG DATA ANALYSIS

Tony Zheng¹ and Yu Sun²

¹Troy High School, 2200 Dorothy Ln, Fullerton, CA 92831

²California State Polytechnic University, Pomona,
CA, 91768, Irvine, CA 92620

ABSTRACT

Many swimmers are constantly incorporating new and different training regimes that would let them improve quickly [2]. However, it is difficult for a swimmer to see their progress instantly. This paper develops a tool for swimmers, specifically swimmers, to predict their future results. We applied machine learning and conducted a qualitative evaluation of the approach [3]. The results show that it is possible to determine their future performance with decent accuracy. This application considers the swimmer's performance history, age, weight, and height to predict the most accurate results.

KEYWORDS

Machine Learning, Mobile APP, database.

1. INTRODUCTION

Millions of young people dedicate themselves to the sport of competitive swimming [1]. They endure hours of training to push their athletic abilities forward. But the graph of effort vs progression is not linear. Sometimes swimmers experience a period of stagnant growth, causing them to lose faith in themselves and their efforts [4]. This application of machine learning will allow swimmers to see the light at the end of the tunnel. Since every swimmer will experience this problem of plateaued progress, this application will be utilized multiple times by millions of swimmers across the nation [5]. The application of machine learning is unlikely to leave, as each time the swimmers update their data, the algorithm will produce a different result [6]. By allowing the users to see a graph of progression, it will help swimmers get a clear sense of where they are in terms of progression. It also serves as a tracker for the swimmer's athletic performance. Using this application, swimmers can easily access data about their past performance. This will let the swimmer themselves compare and see the progression that they have achieved. It is very likely that the application of machine learning in performance data becomes an essential part of the swimmer's tool for checking the growth of their athletic abilities and part of the coach's strategy for examining the swimmer's potential.

It is common knowledge to all time-based sports athletes that the graph of progression, performance time vs age, resembles the $y=1/x$ graph [7]. In the beginning, there are huge improvements for athletes, with it not being uncommon to improve 5, 8, or even more than 10 seconds within weeks. But as they progress toward the limits of the human body, their progress slows dramatically or even comes to a halt. This is the ideal case, and as the world is not ideal. The first problem is that everyone has a different rate of progression that could be affected by numerous factors, causing impacts on accuracy. For example, some athletes experience a plateau in progression, which is stagnation in their athletic improvements. There are even those who experience a dip in performance even when they are training. The point is that everyone's progression is unique to their situation. The second problem is the inability to do so at scale. In order to determine the potential of the athlete, a person would have to know the athlete's performance and training. After obtaining this information, the person would have to deeply analyze the data for a long time. As a coach who has many athletes under their supervision, it is difficult to map out the rate of progression that their athletes are going through.

The solution proposed in this paper is the usage of machine learning algorithms [8]. Our goal is to accurately predict the performance times of swimming athletes. This method was inspired when I noticed a trend while viewing a graph of my swimming performance vs time. This graph shows a clear general curve that my past performance follows. So if it is possible to figure out the function that could generate that curve, then I will be able to accurately predict what kind of performance I will be able to have in a given year. With machine learning, it is possible to map out a progression graph for the athlete accurately and at scale.

In order to ensure that results were being generated, we ran tests to see the accuracy of the machine learning algorithm named AdaBoost [9]. In order to test the algorithm, we first create a model based on data that is available to us. Out of the 100% of our data, only around 80% of the data are actually used to generate these models. The rest of the 20% are used as tests to determine the model's accuracy. When giving the machine learning model some data that it has never seen before effectively tests if the model is accurate. If the model-generated values are similar to the test data, the model would be considered accurate. The algorithm AdaBoost has tested a 99% accuracy after several trials. This means that the model is 99% accurate to match with the predicted results compared to actual data. With an accuracy this high, it is considered a valid model to use.

The paper will be organized into a total of 6 parts. Part 1 is the introduction so far. The next part, Part 2, will be discussing the challenges that were met on the way to the solution. Part 3 will be the solution to the problem described in the introduction, as well as the solution to challenges from Part 2. Part 4 will be detailing the experiments that were conducted. Part 5 is the related work that's similar to this paper. Finally part 6 is the conclusion that will summarize and give future works potential.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Picking a specific Machine Learning Model for our predictions is difficult

Picking a machine learning model is a challenge, due to the different types of data that it could potentially have to process. There are many machine learning models that could solve a problem, but depending on the situation, one might yield higher accuracy than the other. For our problems, we have to work with an athlete's performance, which is measured in time. We need a model

that considers an individual's performance history to predict future times. Any regression model would work, but it would be better to have higher accuracy. This would be done by testing out the accuracy of each potential model. Using cross-validation to compare various regressive and classification models, we were able to pick out the best model for our specific use case.

2.2. Gathering related sports Swimming Data for our project database can prove challenging as there are not many resources

Gathering sufficient data is challenging because it is absolutely necessary for machine learning models. The more well-organized a model's data-set is, the more easily it can be trained and the more accurate its results will be. Therefore it is ideal to have both plenty of data and well-organized data. To predict the performance of a swimmer, we must have the swimmer's past performances. It is also crucial to have plenty of data so that it can be as accurate as possible. Their results also change as they improve over time. Overall, a database API would give all the data that is needed, if one is available. Many sites or databases allow one to simply import the data through their premade library. Then it is easy to format and make usable. Since there was no API available for any of the online databases, we obtained data through web scraping. Scraping through each individual's listed page of times, this method was able to gather the complete history of performance by any swimmer.

2.3. Predictions for each user cannot rely upon the data of other users

When running machine learning related to individuals, it is important to note that each person has a different condition or abilities. One person's data is not reflective of the future of another person. There are a huge range of athletes, from beginner to Olympic level. It would not make sense to impose a professional's requirements on a novice. A general solution would be to separate the data from person to person. This is usually done by creating profiles for each user. Similarly, we also separated each person's data by having users sign in to their own accounts. This way the user would have their own profile, unaffected by other people's data.

3. SOLUTION

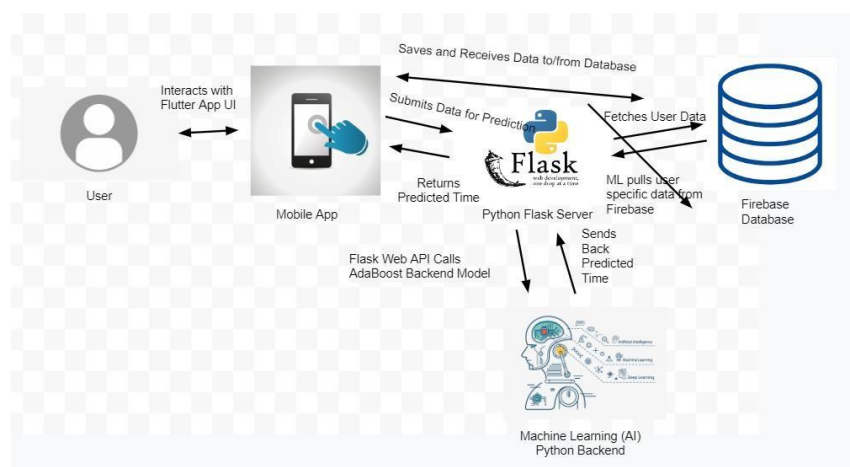


Figure 1. Overview of the solution



Figure 2. Mobile App

The interface for the user is a mobile app that allows the user to access and change information. This application, named Swim Wizard, is connected to a database as well as a server that responds to the user's prompts to run the machine learning algorithm. With the connection to the database, users can manually add, take away, and view the data that are under the user's access. The server, which utilizes flask, is connected to both the database as well as the mobile application to know when the user wishes to run the machine learning algorithm and can directly access the database for data to feed the machine learning algorithm. When the server needs to run the algorithm, it calls upon the AdaBoost backend model to run the code. When it is done, the results will be sent back to the server, then the mobile app, and ultimately the user. There are many components within this project. Each of them is closely connected to work as intended. All of the connections are two-way, as information needs to be both accessed and changed at all components. For ease of access as well as clarity, the user only has access to the mobile end, which will ultimately open access to every component within the project.

The mobile app was constructed using a developing platform called Android Studio [10]. Firstly built was the user interface. The creation of pages was followed by the population of buttons, text fields, drop down menus, graphs, user text fields, and much more. After the essential components were laid down, each component was given its own role and functions, so they can either act or be acted upon. For example, the functionality of what happens after the user presses a button was necessary for the user interface to work as intended.

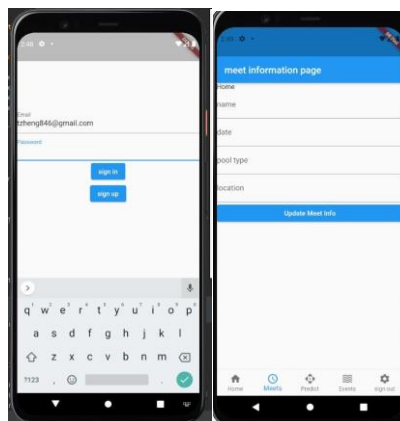


Figure 3. Sign in and information page

The database was built using Google's Firebase. It is an online database that can be easily connected to apps. Its most useful features, its free price, and the ease of integration were the reasons that this specific database was chosen. After creating a database, it is necessary to organize the data by categorizing them based on hierarchy. The hierarchy would be ordered as the user's id, meet, event, and performance time. This ensures ease of access when the user is trying to find a specific performance time.

```

84
85 #Return a Meet object (Dictionary)
86 def createMeet(name, events, _date, poolType, location):
87     meetDict = {
88         name : {
89             'events' : events,
90             "date": _date,
91             "pool type": poolType,
92             "location" : location
93         },
94     }
95     return meetDict
96
97 createUser("tempPerson@gmail.com", 26, 5.11, 160, "temperson", "team Team")
98 getUserInfo("tempPerson@gmail.com")
99
100 event_info = createEvents([[ "100 FR", "afternoon", 49], [ "50 FR", "morning", 23]])
101 meet_info = createMeet("Cheezit Invitational", event_info, "Mon 4th", "LCM",
102                       "Columbia")
103 updateUser("tempPerson@gmail.com", meet_info)
104

```

Figure 4. Screenshot of code 1

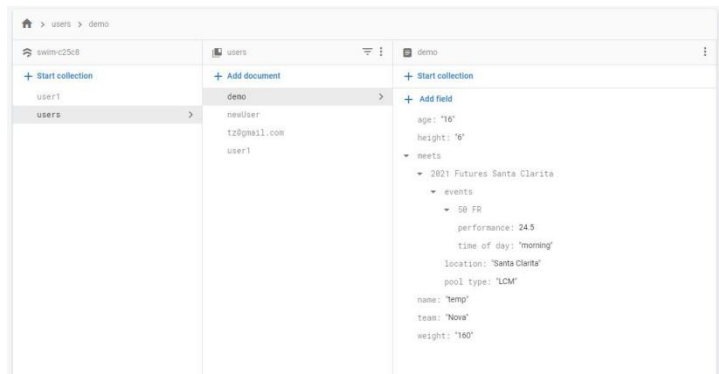


Figure 5. Screenshot of code 2

The server was made using Python Flask. The server's task is to listen for requests to run certain codes. When it receives a request from the mobile app, it runs a snippet of code. For example, when the mobile app requests to run the predict function, the server, upon receiving the request, obtains necessary data from the database and runs the Adaboost machine learning algorithm. Once the algorithm returns a result, the server sends the result back to the mobile app. AdaBoost was chosen as the prime choice of machine learning algorithm due to its superior accuracy during tests.

```

15
16 app = Flask(__name__)
17
18 @app.route("/")
19 def home():
20     return "Welcome to our Swim App"
21
22 @app.route("/test")
23 def test():
24     return "TEST SUCCESSFUL!"
25
26 def sec_to_mins(seconds):
27     a=str(int((seconds%3600)//60))
28     b=str(int((seconds%3600)%60))
29     c=["{} mins and {} seconds".format(a, b)]
30     return c
31
32
33
34 # https://Tony-Swim-Time-Web-
35 Scraping.gunnellevan.repl.co/runPrediction/12_3_2020/home/Free/200/15/0/0/Yd
36 @app.route("/runPrediction/<Date>/<location>/<stroke>/<distance>/<age>/<gender>/<cte
37 am>/<pool_type>/<rankId>")
38 def doPredictions(Date, location, stroke, distance, age, gender, team,
39 pool_type,rankId):
40     f = open("data.txt", "r")
41     data = json.load(f)
42     f.close()
43
44

```

Figure 6. Screenshot of code 3

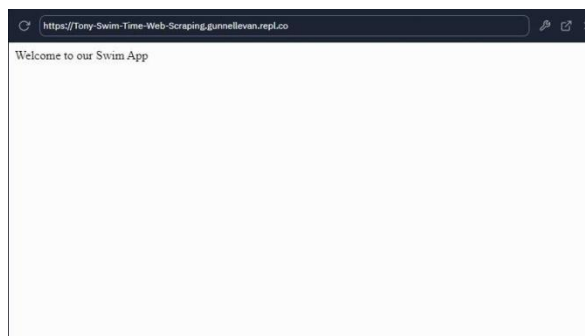


Figure 7. Screenshot of website

Then came the final part of integrating all the parts. Using the instructions from Google's Firestore, it was easily integrated with both the mobile app and the server. It establishes a two-way stratosphere of data for each part. Finally, the mobile app was easily linked up with Python Flask when the URL was provided to the app.

4. EXPERIMENT

Our goal is to determine if it is possible to utilize machine learning to predict the future performance of a swimmer. Some questions need to be addressed. What is the best machine learning algorithm? What CV is the best for machine learning? What parameter has the most effect on the results of the prediction? The experiment that we set up specifically addresses these questions. First, we gathered many different machine learning algorithms. Then we tested each algorithm using built-in scikit-learn functions. We used the "cross_val_score()" function to obtain the accuracy of each function. The function requires many parameters such as the model, input, output, and CV score. By setting all algorithm's parameters as constants, we were able to determine the accuracy of each algorithm. The second question is solved by setting the algorithm and all parameters except for the CV score as constants. By adjusting the CV score, we were able to determine the most optimal CV score. The third question is solved by using the selected machine learning algorithm and using a function from scikit-learn library to test out the effectiveness of each parameter.

We were able to collect the data that we intended to gather. The data that is shown in figure 4.1 and 4.2 shows the accuracy performance of each model. Using the two figures, it is clearly shown that AdaBoost has the highest accuracy with an outstanding 95.06% accuracy. Thus we determined that AdaBoost is the machine learning algorithm that is best suited for predicting future swimmer performance. The data for experiment two is recorded in figure 4.3 and graphed in figure 4.4. Using these data, we concluded that using a CV score of 7 produces the best results. We also speculated that the higher the CV score, the better the accuracy. But since we have not tested any higher CV count higher than 7, we can not conclude that the statement is correct. The data for experiment three is recorded in figure 4.5 and graphed in 4.6 for visual clarity. Using the two figures, we can see that there are only two major factors to the production of results. Age takes up a majority, with date filling up almost the rest. The effects of location are almost negligible.

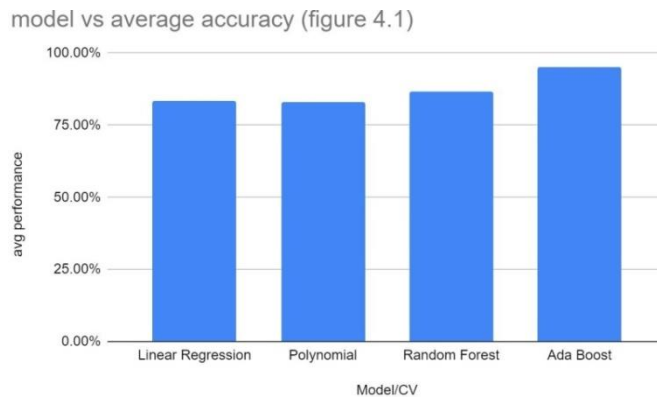


Figure 4.1 Model vs average accuracy

Figure 4.2	
Model	avg performance
Linear Regression	83.52%
Polynomial	83.09%
Random Forest	86.74%
Ada Boost	95.06%

Figure 4.2 Model vs average performance two

figure 4.3			
Model/CV	3	5	7
Linear Regression	86.45	77.65	86.45
Polynomial	83.05	83	83.22
Random Forest	86.84	86.45	86.92
Ada Boost	94.83	94.52	95.83

Figure 4.3 The data of experiment

model CV vs accuracy (figure 4.4)

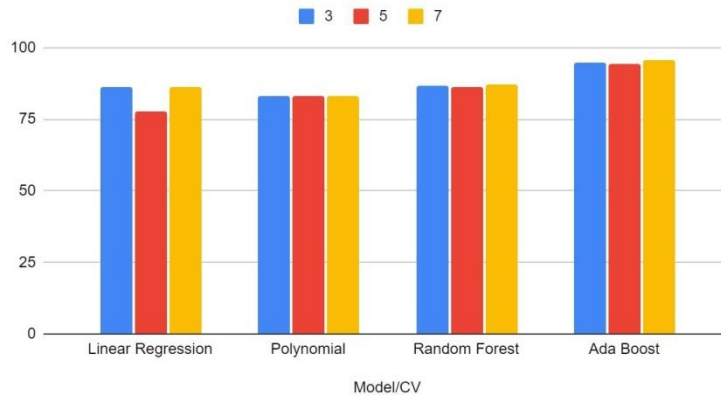


Figure 4.4 The graph of experiment two

Figure 4.5	
Date	0.20556553
location	0.00486057
stroke_type	0
distance	0
age	0.7895739
gender	0
team	0

Figure 4.5 The data of experiment three

data weight distribution (figure 4.6)

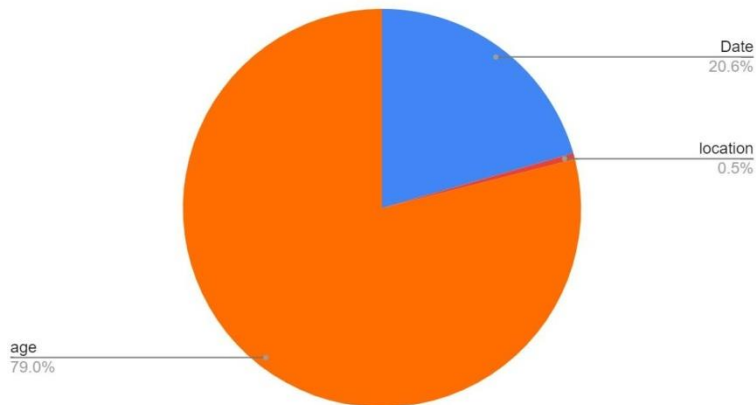


Figure 4.6 The graph of experiment three

The three experiments went as expected. We were able to gather the data that we expected. For experiment one, we initially thought polynomial regression would be the top pick. But the results of the experiment proved that AdaBoost is superior for our intended purposes. AdaBoost outperformed the second best choice, Random Forest, by 9% in accuracy. This is a significant performance difference. For the second experiment, we did not know what to expect. But the results showed that a CV of 7 outperformed CV score of 3 and 5. This was true for all algorithms

that we tested. Finally, experiment three is mostly what we have expected. We also predicted that the most prominent factor would be time related. Both date and age are very similar data that relates to time.

5. RELATED WORK

Using a prediction model of machine learning, Zhu aims to accurately predict the athlete's performance [11]. It improves the prediction model by incorporating specific changes of the athlete's performance, finding hidden rules using chaotic theory, and using vector machines and particle swarms. Zhu uses advanced techniques in order to obtain extremely accurate results. The application of this paper is strictly analytical. Compared to this paper, SwimWizard is tailored to the swimmers, allowing them to view their performance history as well as get a decent prediction of their future results.

This paper regression analysis in order to research the critical period of swimmer's athletic training [12]. In addition, it also reviews many methods of predicting swimming performance using correlation of swimmer age. "Machine learning of swimming data via wisdom of crowd and regression analysis" is a very in depth analysis of using quantitative data in order to find the answers to many important answers to swimming. It is significantly more advanced than this paper. The main difference in our work is that we explore not only age, but other factors that could affect performance. These factors that we explored include location, and team. Since some locations may provide better facilities, causing a difference in performance. In addition, each team offers a different training regiment and different coaches.

This paper focuses on the classification of breaststroke styles for each swimmer [13]. Using machine learning, the author hopes to find a way to identify the difference in technique for each swimmer. Although both this author and this paper focuses on swimmer performance, there is a huge difference. "A Machine Learning Approach to Breaststroke " uses categorization machine learning algorithm. They have highly complex algorithms that feed on visuals on the technique of breaststroke. It analyzes the technique qualitatively and produces results via clustering. We focus purely on quantitative analysis, using a set of values to produce another value.

6. CONCLUSIONS

Using data from the history of a swimmer's performance to predict the swimmer's future performance. This could be done by feeding data into a regression type machine learning algorithm. With Scikit learn library functions, we have found that AdaBoost is the best machine learning algorithm [14]. Using the AdaBoost machine learning algorithm, we have achieved 95% accuracy in prediction. It produces a reasonable result. Although 95% is very high in terms of general accuracy, unfortunately it is not enough as a satisfactory result for a swimmer, as even 1% could cause a huge variance. We tested many different parameters such as age, location, team, gender and date. We found out which variable can cause differences in results, as well as how much of an impact each variable makes.

One of the biggest limitations is the amount of data we had access to during the experiment. Using only one swimmer's data, we were very limited in our options. As such, the accuracy as well as the conclusions will be skewed. The practicability of producing a prediction using machine learning that is better than the prediction of a professional coach is very low. Although we found out that we can achieve 95% accuracy, it is not enough. A professional coach is able to create a prediction that comes close to 99% accuracy. This could optimize this by trying this experiment on a larger data set.

We want to get access to more data. In order to do this, we could apply to gain access to the official USA Swimming data set, which contains information about millions of swimmers [15]. With a larger data set, the machine learning model would be able to train much more than previously. This would likely result in a much higher accuracy in its predictions.

REFERENCES

- [1] Mujika, Inigo, et al. "Effects of training on performance in competitive swimming." *Canadian journal of applied physiology* 20.4 (1995): 395-406.
- [2] Geiger, Mario, Leonardo Petrini, and Matthieu Wyart. "Landscape and training regimes in deep learning." *Physics Reports* 924 (2021): 1-18.
- [3] Jordan, Michael I., and Tom M. Mitchell. "Machine learning: Trends, perspectives, and prospects." *Science* 349.6245 (2015): 255-260.
- [4] Petrakis, Panagiotis E., Dionysis G. Valsamis, and Kyriaki I. Kafka. "From optimal to stagnant growth: The role of institutions and culture." *Journal of Innovation & Knowledge* 2.3 (2017): 97-105.
- [5] Conner, Deondra. "The effects of career plateaued workers on in-group members' perceptions of PO fit." *Employee Relations* (2014).
- [6] Mahesh, Batta. "Machine learning algorithms-a review." *International Journal of Science and Research (IJSR)*. [Internet] 9 (2020): 381-386.
- [7] Pierson, William R., and Henry J. Montoye. "Movement time, reaction time and age." *Journal of Gerontology* 13.4 (1958): 418-421.
- [8] Singh, Amanpreet, Narina Thakur, and Aakanksha Sharma. "A review of supervised machine learning algorithms." 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom). Ieee, 2016.
- [9] Vezhnevets, Alexander, and Vladimir Vezhnevets. "Modest AdaBoost-teaching AdaBoost to generalize better." *Graphicon*. Vol. 12. No. 5. 2005.
- [10] Esmael, Hana R. "Apply android studio (SDK) tools." *International Journal of Advanced Research in Computer Science and Software Engineering* 5.5 (2015).
- [11] Zhu, Pan, and Feng Sun. "Sports athletes' performance prediction model based on machine learning algorithm." *International Conference on Applications and Techniques in Cyber Security and Intelligence*. Springer, Cham, 2019.
- [12] Xie, Jiang, et al. "Machine learning of swimming data via wisdom of crowd and regression analysis." *Mathematical Biosciences & Engineering* 14.2 (2017): 511.
- [13] Zanchi, Marco. "A Machine Learning Approach to Breaststroke."
- [14] Pedregosa, Fabian, et al. "Scikit-learn: Machine learning in Python." *the Journal of machine Learning research* 12 (2011): 2825-2830.
- [15] McCubbrey, Donald J., Paul Bloom, and Brad Younge. "USA Swimming: the data integration project." *Communications of the Association for Information Systems* 16.1 (2005): 13.

MINING ONLINE DRUG REVIEWS DATABASE FOR THE TREATMENT OF RHEUMATOID ARTHRITIS BY USING DEEP LEARNING

Pinar Yildirim

Department of Computer Engineering,
Faculty of Engineering and Natural Sciences,
Istanbul Okan University, Istanbul, Turkey

ABSTRACT

In this paper, a research study for online patient reviews is introduced. Rheumatoid arthritis is a long-term and disabling autoimmune disease. Today, a huge amount of people have rheumatoid arthritis in the world. Considering the importance of the medication of rheumatoid arthritis, we aimed to investigate patient reviews in WebMD database and get some useful information for this disease. Our results revealed that etanercept treatment has the highest number of reviews. Data analysis was applied to discover knowledge on this drug. Deep learning approach was used to predict the effectiveness of etanercept and classification results were compared with other traditional classifiers. According to the comparison of classifiers, deep neural network has better accuracy metrics than others. Therefore, the results highlight that deep learning can be encouraging for medical data analyses. We hope that our study can make contributions to intelligent data analysis in medical domain.

KEYWORDS

Classification, Deep Learning, Etanercept, Online Drug Reviews.

1. INTRODUCTION

Digital technologies provide many opportunities for healthcare treatment and research [1]. Thanks to these technologies, patients can communicate with other patients and type reviews about their medications in some social media sites. These reviews are important for both healthcare experts and drug companies who goal to follow the results of medications and increase the efficiency of them. In this study, WebMD medical website was used and an analysis was conducted using the patients' reviews on the medication of rheumatoid arthritis (RA). RA is a long-term, growing, and disabling autoimmune disease [2]. Considering the importance of the treatment of RA, we explored WebMD reviews and aimed to get some useful information for this disease [3]. Recently, the introduction of deep learning techniques is a promising trend in intelligent data analysis. Still, there are few studies based on these techniques for online patient reviews. We used these approaches in our study and targeted to make contributions to intelligent data analysis in medical domain.

2. RELATED WORKS

Several studies related to patient reviews exist in the literature. Bordes et al. investigated the patient with RA acceptance of social networking sites for the self management of disease. They performed a qualitative study by using interviews in patients. They found that the patients usually use Internet for health information but have limited aspect of social networks for their disease

David C. Wyld et al. (Eds): ARIA, SIPR, SOFEA, CSEN, DSML, NLP, EDTECH, NCWC - 2022

pp. 105-113, 2022. CS & IT - CSCP 2022

DOI: 10.5121/csit.2022.121509

management. This reveals that an Internet based tool is needed to help the management of RA [4].

Kanzaki et al., created an Internet based system to gather data. Women patients with RA participated in this study and used this management system and communicated with the researchers. Their study showed that the use of web based system can positively affect on symptom management of RA [5].

Ellis et al., investigated arthritis patients' health literacy through their social network. They developed qualitative study based on semi-structured interviews. According to their results, patients have limited literacy capabilities and little information about their medications. Further, this study shows the patients with higher education level are more apt to health information searching behaviour [6].

3. METHODS

3.1. Data Sources

Data was gathered from the patient reviews for the treatment of RA on the WebMD website. This network provides web based system for patients to share their reviews of medication. In WebMD system, patients can rate effectiveness, ease of use, drug satisfaction from 1 to 5 stars and choose why they use the drug. Patients can also enter their age, gender, medication duration data and free text comments (Figure 1).

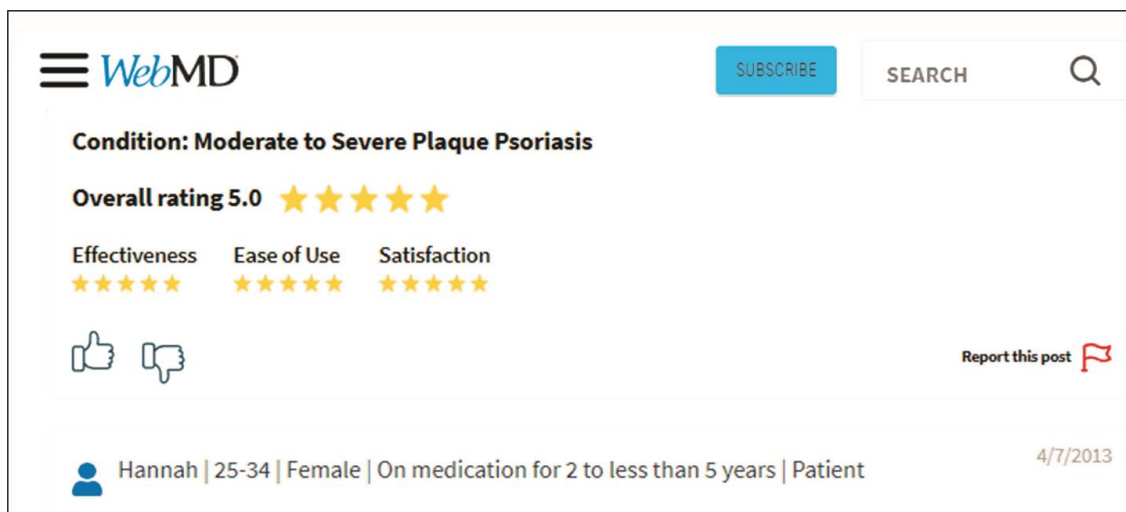


Figure 1. An example of patient review on WebMD website.

3.2. Deep Learning

Deep learning is a neural network approach which has many concealed layers in the network. The network operates huge amount of data through multiple layers and it can easily learn complex features at each layer (Figure 2). Thanks to this feature, deep neural network can handle and analyze many different types of data and decrease the drawbacks of traditional machine learning algorithms [7]. The main idea of deep learning is to search a function that generates the expected output for given inputs. Deep Learning requires intensive computational complexity and graphics processing units (GPUs) are important for the performance of network.

There are several deep learning models developed for different tasks. A simple Deep Neural Network (DNN) model is the auto encoder (AE) that consists of encoder and decoder functions for input and output layers. Convolutional neural network (CNN) has an interleaved set of feed forward layers containing convolutional filters, reduction, rectification or pooling layers. For each layer the CNN generates high-level abstract feature. CNN have been broadly applied image recognition and natural language processing. Another deep learning model is recurrent neural network (RNN). RNN exhibits dynamic structure and neurons in the network are related to time steps. Therefore, RNNs can easily handle sequential data. Deep belief network (DBN) is another type of deep neural network which consists of multiple layer of graphical model having both directed and undirected edges. A Deep Boltzmann Machine is a type of a Deep Neural Network formed from multiple layers of neurons with nonlinear activation functions. The architecture of a Deep Boltzmann Machine allows it to come to know very complicated relationships between attributes and provides advanced performance in learning of high-level representation of attributes [8]. Some types of deep learning techniques have been used in biomedical area such as biomedical imaging, medical diagnosis, electronic health records and biomedical signals [9]. Some concepts are explained below;

Activation function

An activation function decides the output of each node in an artificial neural network basically and it can be called a transfer function that is used to map the output of one layer to another. Activation functions are important components of artificial neural networks (ANN), they affect on the performance of network. There are several activation functions such as sigmoid, hyperbolic tangent, softmax, rectified linear unit (ReLU) and softplus used in neural networks [9].

Learning rate

The learning rate is a hyper parameter that checks how much to vary the network in reply to the expected error each time the weights are changed. Small learning rate can cause a long training, whereas large rates may result in learning a sub-optimal set of weights too fast. The learning rate may be a significant hyper parameter for the architecture of neural network and it affect on the performance of the network [10].

Epoch

Epochs can be defined as how many iterations of the data the network will be used to train a model [11].

Loss function

Loss functions are used to measure how accurate the prediction is performed. If the prediction is obtained far away from true value i.e. prediction deviates more from real value, then the loss function generates high numeric value. In order to get good prediction, it must have low loss function values.

Regularization

In neural network studies, sometimes training data cannot be enough and the network model can face overfitting and under fitting problems. To handle these problems, some regularization techniques have been generally used for data analysis [12].

Batch size

Batch size *refers* the number of training examples in one iteration in the neural network.

Optimization

A learning task can be defined as an optimization problem, to find the minima of the objective function by choosing hyper parameters [9]. Stochastic gradient descent approach repeatedly makes small regulations to neural network configuration to reduce the error of the network. In deep learning networks, this method and its variants are widely used to achieve optimization [13].

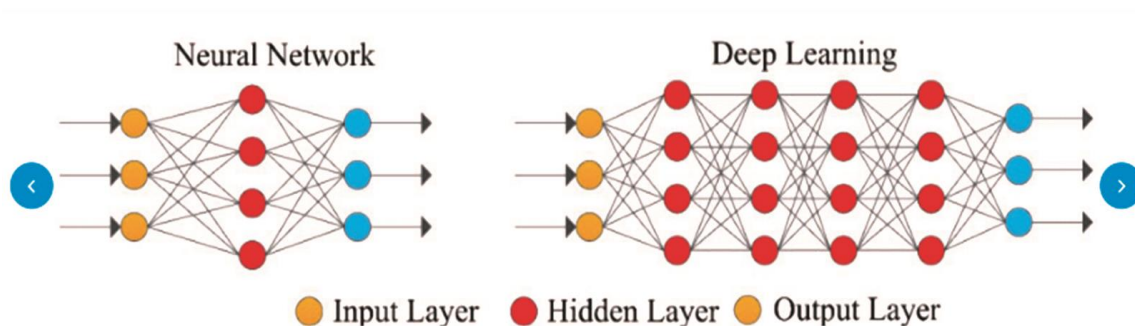


Figure 2. Architecture of Deep Learning [14].

3.3. Traditional Classification Algorithms

IBK

K-nearest neighbour algorithm is known as IBK in Weka software. When a new sample is given, a k-nearest neighbour classifier investigates the patterns for the training instances that are nearest to the unknown sample. The new sample is classified by its k nearest neighbours [15].

RandomTree

Random Tree is a kind of classifier; it is a type of ensemble learning algorithm that produces many individual learners. It uses a bagging idea to generate a random set of data for building a decision tree. In standard tree each node is split using the best split among all features [15,16, 17].

Random Forest

A Random Forest is a kind of ensemble of classification trees, where each tree makes contributions with a vote for the assignment of the most frequent class to the input data [18].

Naive Bayes

Naive Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problem. It is a probabilistic classifier, which means it predicts on the basis of the probability of an object [19].

Kstar

Kstar is an instance-based classifier, that is the class of a test instance is based upon the class of those training instances similar to it, as determined by some similarity function. It differs from other instance-based learners in that it uses an entropy-based distance function [20].

Logistic Regression

Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables [21].

4. EXPERIMENTAL RESULTS

4.1. Data Analysis

We processed RA related patient reviews from WebMD website [8]. WebMD contains both structured data and free text comments. We collected patient reviews and processed these data. After data preprocessing, we converted WebMD data into structured form and created a MySQL database. According to analysis, etanercept has the highest number of reviews (248), we selected this drug for analysis [22-23]. Table 1 shows etanercept related attributes used for classification. These attributes are user's gender, user's age group, the time on the drug, user rating of ease of use, user rating of several satisfaction and user rating of effectiveness.

4.2. Classification Results

We selected six attributes of etanercept dataset (Table 1) to predict user rating of drug effectiveness. We implemented DNN for classification. The network was designed as dense layer (35 nodes) and output layer 6 nodes). We used some DNN parameters and these parameters were kept fixed in all experiments performed in this study:

(a) ActivationSoftmax was selected as a activation function,

$$g(a) = \frac{e^{a_i}}{\sum_j e^{a_j}}$$

The softmax output, which a_n be considered as a probability distribution over the categories, is commonly used in the final layer [9].

(b) Stochastic Gradient Descent was used as an optimization method which an iterative method for optimizing an objective function with suitable smoothness properties [24].

(c) LossMCXENT, a type Multi-Class Cross Entropy loss function, was selected as a loss function

The performance of the DNN model was affected by hyper parameters which results in optimal classification results. We tried to get optimal values for the DNN model by changing the numbers of epoch and mini batch size. The same classification experiment was run in increasing steps by varying the number of epoch between 10 and 100 and the value of mini batch size between 1 and 10 (Table 4-5). WekaDeeplearning4j software was used. The software is a deep learning package

in DeepLearning4j. This software ensures a graphical user interface (GUI) for deep learning applications [25].

Table 1. Attributes used for classification

Attributes	Data Type
User's gender	Categorical Male,Female
User's age group	Categorical 3-6, 7-12, 13-18,19-24, 25-34, 35-44, 45-54,55-64,65-74,75 or over
User rating of ease of use	Categorical Rating: 1,2,3,4,5
User rating of several satisfaction	Categorical Rating: 1,2,3,4,5
The time on the drug	Categorical Less than 1 month 1 to 6 months 6 months to less than 1 year 1 to less than 2 years 2 to less than 5 years 5 to less than 10 years 10 years or more
User rating of effectiveness	Categorical Rating: 1,2,3,4,5

Performance of classification algorithms are evaluated by some accuracy measures.

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F - measure = \frac{2TP}{2TP + FN + FP}$$

TP: The number of True Positives

TN: The number of Negatives instances

FP: The number of False Positives

FN: The number of Negatives instances

The evaluation analysis by root mean squared error is also widely used where n is the number of data, $y_{p,m}$ shows the predicted, $t_{m,m}$ is the measured value of one data point m and $\bar{t}_{m,m}$ is the mean value of all measure data values. Root Mean Squared Error (RMSE) can be shown as follows [26]:

$$RMSE = \sqrt{\frac{\sum_{m=1}^n (y_{p,m} - t_{m,m})^2}{n}}$$

Table 2. Performance evaluation for different epoch values

Epoch	Precision	Recall	F-measure	RMSE
10	0.700	0.677	0.680	0.2884
20	0.722	0.702	0.707	0.276
30	0.704	0.702	0.700	0.2735
40	0.701	0.710	0.703	0.2738
50	0.697	0.706	0.699	0.2746
60	0.684	0.694	0.687	0.2755
70	0.679	0.690	0.683	0.2763
80	0.672	0.681	0.675	0.2771
90	0.667	0.677	0.671	0.2777
100	0.667	0.677	0.671	0.2783

Table 3. Performance evaluation for different mini-batch sizes.

Mini-batch size	Precision	Recall	F-Measure	RMSE
1	0.700	0.677	0.680	0.2884
2	0.680	0.633	0.645	0.3059
3	0.650	0.585	0.606	0.3193
4	0.647	0.556	0.586	0.3298
5	0.636	0.524	0.563	0.3376
6	0.630	0.504	0.548	0.3433
7	0.610	0.472	0.521	0.3492
8	0.608	0.464	0.514	0.3534
9	0.573	0.395	0.452	0.3573
10	0.568	0.387	0.445	0.3596

Table 4. Performance evaluation for different classification algorithms

Classification Algorithm	Precision	Recall	F-measure	RMSE	Execution Time
Deep Neural Network	0.700	0.677	0.680	0.2884	5.04
Random Forest	0.668	0.673	0.670	0.2791	0
IBK	0.585	0.593	0.587	0.3128	0
Random Tree	0.615	0.605	0.607	0.3257	0
Kstar	0.582	0.625	0.592	0.2836	0

Table 2-3 show the performance evaluation for DNN with varying epoch values and mini-batch sizes. According to Table 2, the highest precision values were got for the dataset with epoch 20. For instance, the recall of DNN with epoch 20 is 0.702 in the Table 2. These results reveal that smaller epoch values may produce higher accuracy values. Similarly, small mini batch sizes generates good accuracy values for classification tasks. For example, mini batch size 1 has highest F-measure with 0.680 and smallest RMSE with 0.2884 in Table 3. However, there is no general decision for these results. Smallest epoch and mini batch values may not result in small RMSE values in all studies.

We also compared the performance of DNN with other traditional algorithms. Table 4 shows the accuracy metrics for different classification algorithms. According to results, DNN has better accuracy values than other algorithms. For example, the precision of DNN is 0.700 which is the highest value in the table. On the other hand, DNN has the longest execution time with 5.04.

5. CONCLUSIONS

The biomedical research aims to investigate unknown and useful knowledge to make contributions for healthcare. Drug satisfaction is one of the most important issue for the medical area. In this study, we carried out data analysis for the treatment of RA and mostly reviewed etanercept drug. We analyzed WebMD database and searched the patient satisfaction of etanercept. We implemented deep learning approach to predict the relationships between drug effectiveness and other features such as gender, age and the time on the drug. The performance of DNN was observed by epoch and mini batch size to find optimum parameters. A comparative experiment was also performed on classification algorithms to evaluate them. The results highlight that deep learning is promising technique and it can result in high accuracy of classification with optimum epoch and mini batch size parameters. In conclusion, our study can make contributions for both medical experts and data scientists.

ACKNOWLEDGEMENTS

We would like to thank Alkan Kaya for his help and contributions.

REFERENCES

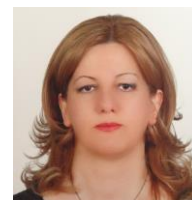
- [1] Yildirim P. Association patterns in open data to explore ciprofloxacin adverse events. *Applied Clinical Informatics* 2015, 6(4): 728-747.
- [2] <https://www.rheumatoidarthritis.org>.
- [3] Boshu R, Charles W-H, Yao L. Identifying serendipitous Drug Usages in Patient Forum Data-A Feasibility Study. *BIOSTEC 2017*, 2017, page 106-118.
- [4] Bordes JKA. des, Gonzales E., Lopez-Olivo M., Nayak P, Suarez-Almazor M.E. Assessing information needs and use of online resources for disease self-management in patients with rheumatoid arthritis:a qualitative study, *Clinical Rheumatology* 2018; 37;1791-1797.
- [5] Kanzaki H, Makimoto K, Takemura T, Ashida N. Development of web-based qualitative and quantitative data collection systems: study on Daily symptoms and coping strategies among Japanese rheumatoid arthritis patients, *Nursing and Health Sciences* 2004; 6, 229-236.
- [6] Ellis J, Mullan J, Worsley A, Pai N. The Role of Health Literacy and Social Networks in Arthritis Patients' Health Information-Seeking Behavior: A Qualitative Study, *International Journal of Family Medicine* Volume 2012, Article ID 397039, 6 pages.
- [7] Tobore I, Li J, Yuhang L, Al-Handarish Y, Kandwal A, Nie Z, Wang L. Deep Learning Intervention for Health Care Challenges: Some Biomedical Domain Considerations, *JMIR MHealth and UHealth* 2009, vol.7, iss.8, p1-36.
- [8] Taherkhani A, Cosma G, McGinnity T M, Deep-FS. A feature selection algorithm for deep Boltzmann Machines, December 2018, *Neurocomputing* Vol 322, page 22-37.
- [9] Cao C, Liu F, Tan H, Song D, Shu w, Li W, Zhou Y, Bo X, Zie Z. Deep Learning and its applications in biomedicine, *Genomics Proteomics Bioinformatics* 2018, 16, 17-32.
- [10] <https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/> [20 December 2021].
- [11] Maxwell A, Li R, Yang B, Weng H, Ou A, Hong H, Zhou Z, Gong P, Zhang C, Deep learning architectures for multi-label classification of intelligent health risk prediction, *BMC Bioinformatics*, 18(Suppl 14);523, 2017
- [12] Nusrat I, Jang SB. Comparison of Regularization Techniques in Deep Neural Networks, *Symmetry* 2018, 10, 648.

- [13] Koutsoukas A, Monaghan KJ, Li X, Huan J, Deep-learning: investigating deep neural networks hyper-parameters and comparison of performance to shallow methods for modeling bioactivity data, *J Cheminform*,9:42, 2017.
- [14] https://www.researchgate.net/figure/Deep-learning-diagram_fig5_323784695.
- [15] Han J Kamber M. *Data Mining Concepts and Techniques*.Morgan Kaufmann, 2011 .
- [16] Kalmegh SR. Comparative Analysis of WEKA Data Mining Algorithm RandomForest, RandomTree and LADTree for Classification of Indigenous News Data. *Int. Journal of Emerging Technology and Advanced Engineering* 2015; 5;1. 13.
- [17] Pfahringer B. Random model trees: an elective and scalable regression method. Working Paper Series, 2010, ISSN 1177-777X.
- [18] Breiman L, Friedman J, Stone CJ, Olshen RA, *Classification and regression trees* (1st ed.), Chapman and Hall/CRC, Belmont, CA (1984).
- [19] <https://www.javatpoint.com/machine-learning-naive-bayes-classifier>
- [20] <https://weka.sourceforge.io/doc.dev/weka/classifiers/lazy/KStar.html>
- [21] <https://www.statisticssolutions.com/free-resources/directory-of-statistical-analyses/what-is-logistic-regression>
- [22] Rastegar-Mojarad, Majid, Liu,Hongfang, Nambisan, Priya. (2016) “Using Social Media Data to Identify Potential Candidates for Drug Repurposing: A Feasibility Study.” *JMIR RESEARCH PROTOCOLS* ,vol. 5, iss. 2, e121 p.
- [23] <http://www.drugs.com>
- [24] <http://www.wikipedia.org>
- [25] Lang S, Bravo-Marquez, Beckham C, Hall M, Frank E, *WekaDeeplearning4j: A deep learning package for Weka based on Deeplearning4j*, *Knowledge-Based Systems*, Volume 178, 15 August 2019, Pages 48-50).
- [26] Kuçuksille EU, Selbas R, Şencan A. Prediction of thermodynamic properties of refrigerants using data mining. *Energy conversion and management* 2011; 52: 836-848.

AUTHORS

Pinar Yildirim is an associated professor at the Department of Computer Engineering of Istanbul Okan University in Istanbul.

She received her B.S. degree Yıldız Technical University, M.S. degree from Akdeniz University and PhD degree in the Department of Health Informatics of the Informatics Institute at Middle East Technical University in Turkey.Her research areas include biomedical data mining,.machine learning,.



GENERATIVE APPROACH TO THE AUTOMATION OF ARTIFICIAL INTELLIGENCE APPLICATIONS

Calvin Huang¹ and Yu Sun²

¹University High School, 4771 Campus Dr, Irvine, CA 92612

²California State Polytechnic University,
Pomona, CA, 91768, Irvine, CA 92620

ABSTRACT

In order to use the full power of artificial intelligence, many are required to navigate through a complex process that involves reading and understanding code. Understanding this process can be especially intimidating to domain experts who wish to use A.I to develop a project, but have no former experience with programming. This paper develops an application to allow for any domain expert (or normal person) to gather data, assign labels, and train models automatically without the use of software to do so. Our application, through a server, allows the user to send HTTP API requests to train models, upload images to the database, add models/labels, and access models/labels.

KEYWORDS

Tensorflow Lite, Flask, Flutter, Google Colab.

1. INTRODUCTION

With the rise of popularity of artificial intelligence throughout the last few decades, the world has seen an interweaving between A.I and certain academic domains [1]. Artificial Intelligence and its many applications have been used throughout a variety of critical and far-reaching projects. From medical research in cancer detection to blind assistance systems, image-detection has been used in so many impactful projects [2][3]. It soon became imperative for certain domain experts who want to combine image detection to their projects to have a thorough understanding of programming, training models, gathering data, navigating through Integrated Development Environments, and concepts in Convolutional Neural Networks [4].

Image detection models are quickly becoming an extremely powerful tool for domain experts to use [5]. By requiring them to understand a different subject entirely when they are focused on another academic interest may be time-consuming and inefficient. For example, in order to actually train a model on a system such as Google Colab, they would need to go through a lengthy chunk of code, import files filled with their training data, and export the finished model [6]. The process is too time consuming for non-experts and even general programmers to use. Furthermore, they may need to coordinate and hire machine-learning engineers, which convolutes the process and makes the overall project more complex. Additionally, many domain experts may already be incredibly invested in their own field, which could deter them from taking the time to learn the concepts of machine learning [7]. However, if there was a way to introduce an

abstraction that could allow them to train the models without any code, the process becomes significantly less challenging.

Some of the existing tools that have been used to make machine-learning more friendly and efficient for a non-experienced user are Google Colab Notebooks and an application called CreateML [8]. Google Colab, a hosted Jupyter notebook service developed by Google, gives anyone the ability to train a model simply by accessing a prewritten notebook on their site [9]. Without needing to install an IDE such as Jupyter, a user is theoretically able to add their own data-set into the notebook, and run the code in the prewritten notebook such that eventually the model is created. However, this method is not as efficient or user-friendly as our approach. Going through each section of code, and storing each image into a file one by one makes the process time-consuming and unappealing. In addition, some pieces of code may be confusing or unrecognizable to some users who want to use image-detection, which requires more programming knowledge.

Another tool that relates to this issue is CreateML, an application developed by Apple. CreateML allows the user to train an image-detection model without having to write a single line of code. However, this application is again quite limited as users do not have the ability to easily create their own datasets. Instead, they are forced to take potentially thousands of their own images, upload it all into its own folder, and then use their application to train the model. Thus, like Google Colab, the process of training is again very inefficient and time-consuming for a non-expert. Furthermore, CreateML, like many applications that attempt a Low-Code No-Code approach to image-detection, has a difficult User Interface for non-experts and therefore makes the process more difficult for them [10]. Although there are several existing approaches to making image-detection efficient for a non-expert, many of them are tedious, frustrating, and quite unfriendly for someone looking to utilize this tool but doesn't have much experience with it.

In this paper, we propose a solution that allows a non-expert to train a model and utilize it in an intuitive and straightforward way. The application gives the user both the ability to train a model and use that exact same model. In the admin page, they can either select an existing model or create a new model with any name they want. After, they can give the model different label names, and for each label they can take pictures that correspond to that label. Using this method, non-experts will be able to gather a data-set without having to upload pictures onto a file and then utilize existing tools. After pictures are taken for each label, they can train the model.

The application also allows the user to use the image-detection model. They are given the option to load models, and by choosing an existing model that has already been trained, they can take a screenshot of what they want to compute. This directs them to a page that gives them the probability of the image being a certain label within the model that they choose. Therefore, we believe that our application is not only more friendly for a non-user, but also less time-consuming and more efficient for any tasks that need to implement an image-detection model.

In order to prove that our mobile application would be less tedious, more efficient, and include similar functionality as the standard approach, we conducted three experiments to compare the functionalities, compare the end-to-end processes, and evaluate the accuracy and confidence of the image-detection model used in our mobile application. First, by comparing the functionality of both approaches, we create a checklist to determine if our application succeeds in carrying out certain tasks that are instrumental to the process of machine learning. However, we also analyzed the benefits of using our application as well, and show that not only do we carry out such tasks, but we do it in a more intuitive manner. Furthermore, we compared the end-to-end processes, which allowed us to illustrate the strength of our application: our efficiency. Our results showed that while most of the methods in the typical approach required lines of code and the organization

of images, our approach was able to simplify the process by only requiring users to make several clicks, type several words, and take shots with their phone camera. Finally, by evaluating our accuracy and confidence, we were able to show that the accuracy of our application would not differ from the accuracy of the typical approach. The goal of using these experiments in tandem is to demonstrate that our approach is not only the most efficient method for training and evaluating models, but it is also the most intuitive and user-friendly system for non-experts to use if they wish to create and organize a list of image-detection models.

The rest of the paper is organized as follows: Section 2 gives the details on the challenges that we met during the experiment and designing the sample; Section 3 focuses on the details of our solutions corresponding to the challenges that we mentioned in Section 2; Section 4 presents the relevant details about the experiment we did, following by presenting the related work in Section 5. Finally, Section 6 gives the conclusion remarks, as well as pointing out the future work of this project.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. How do we allow the user to create their own datasets in an efficient way

Perhaps the most tedious part of training an image-detection model is gathering the data. There are a couple of options. For instance, if a user prefers to create a model using existing datasets, then online platforms such as Kaggle can provide the user with thousands of images already organized in its specific labels [11]. There are certainly many websites that give users the ability to gather lots of data. However, if a user needed to customize their own data-set and execute a model based on their own images, they would need to take individual pictures for each label, export the images to a computer, store each image in its corresponding label file, and then export the file to be used to train the model on a notebook. This process is clearly very time-consuming and thus stops non-experts or other people without any experience in code to use image-detection in their projects or other initiatives.

2.2. How do we design a user-interface that is easy to navigate for a non-expert

Because the purpose of the application is to target those who do not have experience with machine learning, the user-interface must also be friendly and intuitive for them. However, because there are so many terms and concepts in Machine Learning, it is quite difficult to introduce a user-interface that doesn't require the user to have at least a basic understanding of the training process in machine learning. For example, in order to understand the process of training a model, the user must be able to understand terms such as labels, training set, and test set. This issue is accelerated even further by the fact that understanding machine learning requires an understanding of coding concepts. If a domain-expert who wants to utilize image-detection does not even know how to code, they would be forced to either work with another engineer or learn by themselves, which is more time-consuming and less efficient.

2.3. How do we introduce an approach on a mobile application that trains the dataset

In order to allow users to use an application on their phones to build models, the app must include an approach to not just organize the dataset, but also train the model. The difficulty lies in the fact that training the model on a phone is too computationally expensive to train. The process would

take more than one standard smartphone, which is unreliable and serves as poor user experience. Furthermore, the smartphone must also hold in potentially thousands of pictures to train the model. This is clearly not doable and thus the application must have some method in which a server is called in order to train the model.

3. SOLUTION

Our application provides an approach to train image-detection models, gather training data, and compute accuracy of testing data on a mobile device. In order to customize the model, we used the Tensorflow Lite Model Maker, a library that reduces the training time and amount of training data, as a means of customizing each image detection model [12]. Our application has three main components - a front-end consisting of an admin and consumer page, a back-end, and a database.

The user is first greeted with a splash screen, and then interacts with the UI on the main menu. The frontend consists of text, buttons, and a list which holds in the different routing pages. From here, the user can choose to either customize their model by selecting “Model Admin”, or test their model by selecting “Model Test”. By selecting “Model Test”, the UI will consist of a text field allowing the user to add a new model, and a list of past models that they can customize and train. Selecting a model will redirect the user to a new page where they can add labels to the model, or edit an existing label. By selecting a label, the user will need to capture images using their camera. The more images they take for each label, the more accurate the overall model. Once they take enough pictures for each label, they can select the “Training!” button to train the model.

If the admin chooses to go to the consumer page, the user will be greeted by a list of trained models. By selecting one, they will need to take a picture of whatever object they want. Once the picture is taken, the user will be shown a screen detailing the label that the image taken corresponds with and the likelihood of the model being correct.

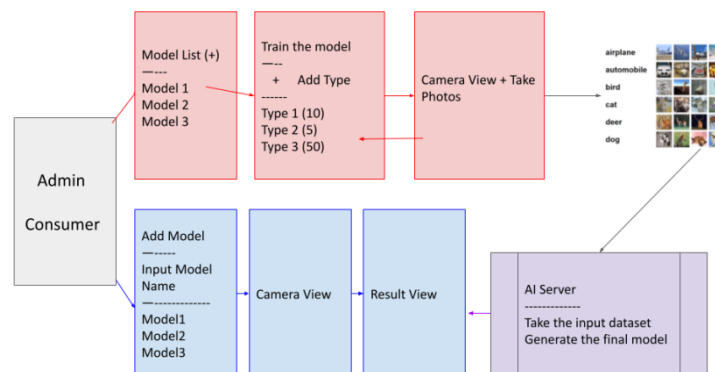


Figure 1. Overview of the solution

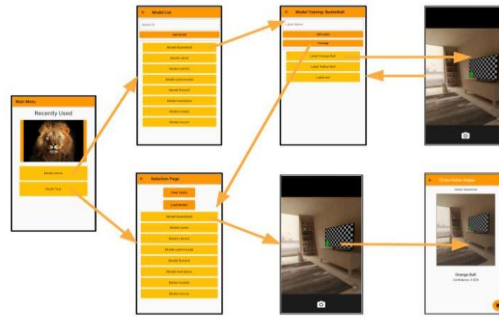


Figure 2. Screenshot of App process

The front-end of the application was developed using Flutter], a UI software development kit created by Google that supports both iOS and Android versions of the application [13]. The Main Menu page was built utilizing the ListView Class, which holds the Page Routers to either the Admin Page or the Consumer Page. Within the Admin portion, we used both the TextField Class to gather the names of any new Model IDs or labels inputted by the user, and the ListView Class to load any new Model IDs or labels. Within the Consumer portion, we also used the ListView Class to load any trained models for testing, and a button class that allowed the user to clear the cache. Once the user takes a picture on the Consumer side, a page is shown with an image of the label it computes to be most accurate, and a text displaying the accuracy.

```
Expanded(
  child: ListView.builder(
    // padding: const EdgeInsets.all(0),
    itemCount: entries.length,
    itemBuilder: (BuildContext context, int index) {
      return ListTile(
        onTap: () {
          if(index == 0){
            Navigator.push(
              context,
              MaterialPageRoute(builder: (context) => ModelListPage()),
            );
          }
          else{
            Navigator.push(
              context,
              MaterialPageRoute(builder: (context) => ModelSelectionNew()),
            );
          }
        },
        title: Container(
          height: 80,
          color: Colors.amber[500],
          child: Center(child: Text('${entries[index]}')),
        ), // Container
      ); // ListTile
    }, // ListView.builder
  ), // Expanded
```

Figure 3. Screenshot of code 1

There are two elements in the ListView: a page router to the admin section and a page router to the consumer section. Clicking on either element in the list will direct the user to that specific section.

The backend was made using a Python Flask server which holds 6 main HTTP APIs [14]. Flask is a web framework that allows for the routing of HTTP requests to the specified controller. The backend is connected to a Firebase database that stores the model names, model labels, and a url which consists of the labels text file and the tflite file of the trained model [15]. In addition, the database stores each image taken by the user for each label they select. By taking advantage of the HTTP APIs from the Flask server, we were able to access and edit the items within the Firebase database. Consequently, we were able to create changes on the front-end UI as well.

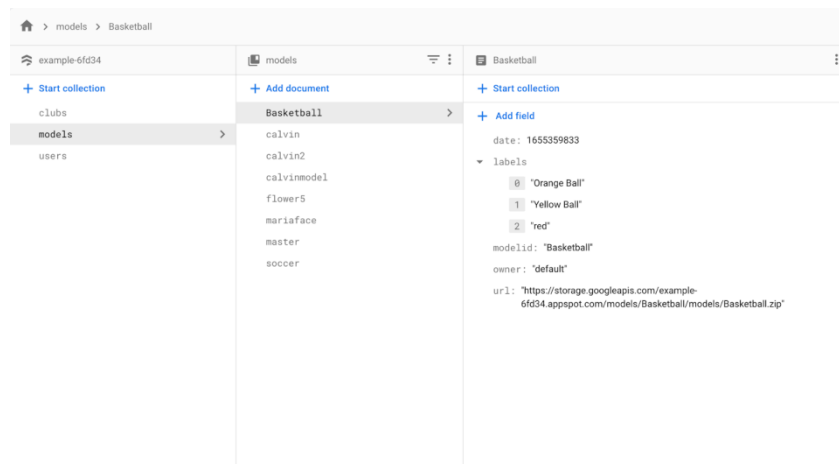


Figure 4. Firestore Database

This image shows the Firestore Database, which holds a variety of models, with its branches having properties such as label names, model ids, and a url containing the tflite file and the label file. This structure allows us to utilize the HTTP APIs to access and edit the properties.

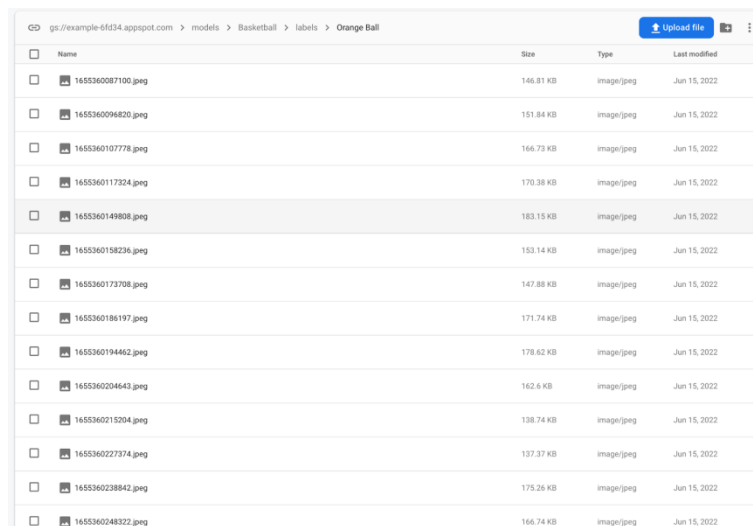


Figure 5. The storage for all the images contained for each label

This image shows the storage for all the images contained for each label. For the example above, the label Orange Ball is selected for the model Basketball. The storage will contain a list of all the pictures that will be taken by the user in the model admin.

Our application uses an HTTP API named `addmodel`, which when given the name as a parameter, will add a new model branch in our Firebase. This allows users to create as many models with different names as they want. Similarly, we used another HTTP API named `addlabel`, which has two parameters: the name of an existing model and a new name for the label. By providing the name of the existing model, the user is able to attach this new label to the branch of that model as a new property.

```

@app.route("/addmodel/<model_name>")
def add_model(model_name):
    data = {
        u'modelid': model_name,
        u'date': int(time.time()),
        u'owner': 'default'
    }
    db.collection("models").document(model_name).set(data)
    return "OK"

@app.route("/addlabel/<model_name>/<label>")
def add_label(model_name, label):
    ref = db.collection("models").document(model_name)
    ref.update({u'labels': firestore.ArrayUnion([label])})

    return "OK"

```

Figure 6. Screenshot of code 2

The Python Flask representation of the APIs for add_model and add_label. Both will access the Firestore Database and add specific values based on the user input.

```

TextField(
    controller: modelIdController,
    obscureText: false,
    decoration: InputDecoration(
        border: OutlineInputBorder(),
        labelText: 'Model ID',
    ), // InputDecoration
), // TextField
Container(
    width: double.infinity,
    child: ElevatedButton(
        onPressed: () {
            http.get(Uri.parse(LocalDB.baseUrl + "/addmodel/" + modelIdController.text)).then((response) {
                loadModels();
                setState(() {

                });
            });
        },
        child: Text(
            "Add Model"
        ) // Text
    ), // ElevatedButton
), // Container

```

Figure 7. Screenshot of code 3

```

TextField(
    controller: labelController,
    obscureText: false,
    decoration: InputDecoration(
        border: OutlineInputBorder(),
        labelText: 'Label Name',
    ), // InputDecoration
), // TextField
Container(
    width: double.infinity,
    child: ElevatedButton(
        onPressed: () {
            http.get(Uri.parse(LocalDB.baseUrl + "/addlabel/" + "/" + widget.modelId + "/" + labelController.text)).then((response) {
                loadLabels();
                setState(() {

                });
            });
        },
        child: Text(
            "Add Label"
        ) // Text
    ), // ElevatedButton
), // Container

```

Figure 8. Screenshot of code 4

The app also uses two different HTTP APIs to gain access to all the model branches and labels for each particular model branch - get_all_models and get_model_info respectively. get_all_models, when executed by an HTTP request, returns a list of the name properties of all

the models. This allows us to utilize the ListView class to linearly display each model with a text that holds the name property. `get_model_info` returns a Python dictionary that stores key-value pairs of objects. In order to gain access to the list of all labels, we set the key property to “labels”, allowing us again to display all the names of the labels as a ListView class.

```
@app.route("/getmodelinfo/<model_name>")
def get_model_info(model_name):
    doc_ref = db.collection("models").document(model_name)
    doc = doc_ref.get()
    return doc.to_dict()

@app.route("/getmodels")
def get_all_models():
    models = []
    doc_ref = db.collection("models").get()
    for doc in doc_ref:
        models.append(doc.id)
    return json.dumps(models)
```

Figure 9. Screenshot of code 5

```
var entries = [];
TextEditingController modelIdController = TextEditingController();

void loadModels(){
    http.get(Uri.parse(LocalDB.baseUrl + "/getmodels")).then((response) {
        print(response.body);
        entries = jsonDecode(response.body);
        setState(() {

        });
    });
}
```

Figure 10. Screenshot of code 6

```
Expanded(
  child: ListView.builder(
    // padding: const EdgeInsets.all(8),
    itemCount: entries.length,
    itemBuilder: (BuildContext context, int index) {
      return ListTile(
        onTap: () {
          Navigator.push(
            context,
            MaterialPageRoute(builder: (context) => ModelTrainingPage(entries[index])),
          );
        },
        title: Container(
          height: 50,
          color: Colors.amber[500],
          child: Center(child: Text('Model ${entries[index]}')),
        ), // Container
      ); // ListTile
    },
  ), // ListView.builder
) // Expanded
```

Figure 11. Screenshot of code 7

```

var entries = {};
var label_name;
TextEditingController labelController = TextEditingController();
void loadLabels(){
  http.get(Uri.parse(LocalDB.baseUrl + "/getmodelinfo" + "/" + widget.modelId)).then((response) {
    print(response.body);
    entries = jsonDecode(response.body);
    if(!entries.containsKey("Labels")){
      entries["Labels"] = [];
    }
    //print(entries);
    setState(() {
    });
  });
}
}
}
}

```

Figure 12. Screenshot of code 8

```

Expanded(
  child: ListView.builder(
    // padding: const EdgeInsets.all(8),
    itemCount: entries["Labels"].length,
    itemBuilder: (BuildContext context, int index) {
      return ListTile (
        onTap: () {
          label_name = entries["Labels"][index];
          getImage();
        },
        title: Container(
          height: 50,
          color: Colors.amber[500],
          child: Center(child: Text('Label ${entries["Labels"][index]}')),
        ), // Container
      ); // ListTile
    }, // ListView.builder
  ), // Expanded
)

```

Figure 13. Screenshot of code 9

The fifth HTTP API we used was called `train_model`, which trains the model based on the labels and then uploads the model file into the Firebase. This allows us to call the `get_model_info` HTTP API on the consumer side, where we can set the key value of the dictionary to “url” to gain access to the model file for testing. The final HTTP API, `upload_image`, saves the picture taken by the user and stores the file to the Firebase as a property of each label. This in turn will allow the `train_model` API to gain access to these images and train the model.

```

@app.route('/upload/<model_name>/<label>', methods=['GET', 'POST'])
def upload_image(model_name, label):
    f = request.files['file']
    f.save(f.filename)
    # upload the file to Firebase storage
    storage_client = storage.Client()
    bucket = storage_client.bucket(bucket_name)
    blob = bucket.blob('models/' + model_name + '/labels/' + label + '/' + f.filename)
    blob.upload_from_filename(f.filename)
    os.remove(f.filename)

    return 'file uploaded successfully'

@app.route("/train/<model_name>")
def train_model(model_name):
    download_from_bucket("models/" + model_name, "models/" + model_name)
    tf_lite_local.train_model(model_name)

    # upload to firestore
    storage_client = storage.Client()
    bucket = storage_client.bucket(bucket_name)
    blob = bucket.blob('models/' + model_name + "/models/" + model_name + ".zip")
    blob.upload_from_filename('models/' + model_name + "/tfmodels/" + model_name + ".zip")
    blob.make_public()
    final_url = blob.public_url
    print("Final URL: ", final_url)

    # update the model URL
    db.collection("models").document(model_name).update({
        "url": final_url
    })

    return "OK"

```

Figure 14. Screenshot of code 10

```

void _upload() async {
    var url = LocalDB.baseUrl + "/" + "upload/" + widget.modelId + "/" + label_name;
    print(url);

    http.MultipartRequest request = http.MultipartRequest('POST', Uri.parse('$url'));
    // print(widget.fileBytes);
    request.files.add(
        await http.MultipartFile.fromBytes(
            'file',
            _image!.readAsBytesSync(),
            filename: DateTime.now().millisecondsSinceEpoch.toString() + ".jpeg",
            contentType: MediaType('image', 'jpeg'),
        ), // http.MultipartFile.fromBytes
    );

    request.send().then((r) async {
        print(r.statusCode);
        if (r.statusCode == 200) {
            ScaffoldMessenger.of(context).showSnackBar(SnackBar(
                content: Text("Uploaded the image successfully"),
            )); // SnackBar
        } else {
            setState(() {});
            print("Failed to upload the image");
            print('error + $r.statusCode');
            ScaffoldMessenger.of(context).showSnackBar(SnackBar(
                content: Text("Failed to upload the image."),
            )); // SnackBar
        }
    });
}

```

Figure 15. Screenshot of code 11

```

Container(
  width: double.infinity,
  child: ElevatedButton(
    onPressed: () {
      print("Training");
      http.get(Uri.parse(LocalDB.baseUrl + "/train" + "/" + widget.modelId)).then((response) {
        print("Successfully triggered the training.");
      });
    },
    child: Text(
      "Training!"
    ) // Text
  ), // ElevatedButton
), // Container

```

Figure 16. Screenshot of code 12

4. EXPERIMENT

4.1. Experiment 1

Common functionality found in text-based script approach for ML Tasks	The Benefits of choosing the mobile application	Functionality that has been already implemented in <i>this</i> software
Training model	Convenient to train models without a computer or running code	Visual interface for training model
Making a prediction	Having a visual interface with the results is more intuitive and less intimidating.	Visual screen with prediction and percentage of error
Uploading a dataset	Way more convenient to create the dataset via images on the phone rather than uploading images to a computer	Intuitive design for the creation of datasets
Must include labels inside the dataset they upload	User does not need to organize the images into labels, reducing the time of training models	Intuitive, visual interface for creating Labels
Must run multiple programs to create multiple models	Easier to store multiple, different models at once.	Intuitive, visual interface for creating models
Large community ^[4]	Depends if the user/developers prefers it.	None; negligible.

Figure 17. A qualitative test on common functionality

Figure 17 depicts a qualitative test on common functionality that is found in the typical script approach to generating image-detection models. We list such functionality and compare the differences between the typical approach and our mobile approach. While the approach can slightly differ, the purpose of the test is to ensure that we check the boxes in the standard functionality, including training, making a prediction, and utilizing data.

In order to be effective for domain experts to use, the application must include features and certain functionality that must be present in the typical approach . These include the abilities to train a model and predict the results based on the labels. However, our application also includes

abilities that are generally not found in the standard text-based programming approach to machine learning, such as the ability to create custom datasets directly on the application. In this experiment, we attempt to compare our application's functionality with that found in a typical text-based script approach to image-detection. Here we can see that we have listed which functionalities are within both approaches and why our approach can be more beneficial for domain-experts with no experience in code. In every case, from training the model, making a prediction, uploading a dataset, and creating multiple models, our mobile application has a simple visual interface that makes the process significantly easier to navigate through.

4.2. Experiment 2

Figure 18 depicts a quantitative test comparing the lists of steps between the typical approach and the approach it takes to handle specific tasks within image-detection. In this experiment, we attempt to demonstrate that our approach is significantly less time-consuming, less tedious, and more intuitive. We will also remove any boilerplate code for the typical approach as implemented in the program is trivial.

Functionality	Typical Approach	Visual Programming (Common Number of Actions)
Saving Multiple Models	End to End Process: Take pictures Manually upload pictures to computer Name each label in ugly file interface Drag each image into its label file Upload dataset to Notebook ~15 LOC for end-to-end process 1 LOC for export for each model Store in file containing all models Repeat process for each model	End to End Process: 1 click to admin Type in name of model Type in names for label Add images (10+ for each label) 1 click for train 1 click to Consumer
Creating Label	Take new pictures Upload into dataset Drag new pictures into label Re-upload dataset to notebook ~15 LOC for end-to-end process 1 LOC for export for each mode 1 LOC for end-to-end process	1 click to admin 1 click to get selected model Type in name for label Add images 1 click to consumer Take picture to use model
Creating New Model	End to End Process: Take new pictures Manually upload pictures to computer Name each label in ugly file interface Drag each image into its label file Upload dataset to Notebook ~15 LOC for end-to-end process (including training) 1 LOC for export for each model	1 click to admin Type in name of model Type in names for label Add images (10+ for each label) 1 click for train
Training Model	End to End Process: Take pictures Manually upload pictures to computer	1 click to admin Type in name of model Type in names for label

	Name each label in ugly file interface Drag each image into its label file Upload dataset to Notebook ~15 LOC for training	Add images (10+ for each label) 1 click for train
Uploading images to dataset	Take pictures Manually upload pictures to computer Name each label in ugly file interface Drag each image into its label file Upload dataset to Notebook	1 click to admin Type in name of model Type in names for label Add images (10+ for each label)
Testing model	End to End Process: Take pictures Manually upload pictures to computer Name each label in ugly file interface Drag each image into its label file Upload dataset to Notebook ~15 LOC for end-to-end process 1 LOC for export for each model Store in file containing all models Repeat process for each model Evaluate Model	End to End Process: 1 click to admin Type in name of model Type in names for label Add images (10+ for each label) 1 click for train 1 click to Consumer Now, you can choose any model already saved to evaluate

Figure 18. A quantitative test comparing the lists of steps

The results show that traditional text-based programming to accomplish any standard functionality is likely to be far more tedious, and requires a more technical understanding of both machine learning and programming. While in the typical approach we would need to use another piece of technology to gather images and then upload, we offer the approach of taking pictures on the mobile phone they are using to train and evaluate the model, making for a significantly less time-consuming process. Furthermore, important functionality such as evaluation and training requires utilizing code in the typical approach; we include the ability to do so with only a couple of clicks, typing, and taking pictures on the phone. This clearly reduces the knowledge threshold required to create image-detection models.

Figure 19 shows a quantitative test depicting the accuracy of the image-detection model used in our mobile application. We attempt to show that the difference between using the model in the text-based approach and our approach is negligible.

	Dataset 1 (3 labels)	Dataset 2 (6 labels)	Dataset 3 (10 labels)
Accuracy	Orange Ball Label 1. Correct @ 90.2% Yellow Ball Label 1. Correct @ 85.1% Green Ball Label 1. Correct @ 84.3% Correct Rate: 3/3 Overall Average: 86.53%	Label 1 1. Correct @ 91.6% Label 2 1. Correct @ 90.9% Average: Label 3 1. Correct @ 92.4% Label 4 1. Correct @ 87.9% Label 5 1. Correct @ 91.1% Label 6 1. Correct @ 89.6% Correct Rate: 6/6 Overall Average: 90.58%	Label 1 1. Correct @ 92.1% Label 2 1. Correct @ 90.6% Average: Label 3 1. Correct @ 91.3% Label 4 1. Correct @ 87.9% Label 5 1. Correct @ 90.4% Label 6 1. Correct @ 89.5% Label 7 1. Correct @ 88.3% Label 8 1. Correct @ 92.3% Label 9 1. Correct @ 88.9% Label 10 1. Correct @ 90.6% Correct Rate: 10/10 Overall Average: 90.19%

Figure 19. A quantitative test depicting the accuracy

The results show that overall, each prediction made by the model has been correct for models that have 3 labels, 6 labels and 10 labels. For 3 labels, we had an overall confidence of 86.53%. For 6 labels, we had an overall confidence of 90.58%. And for 10 labels, we had an overall confidence of 90.19%. The overall median of our results was above 90%, and increasing the number of labels did not decrease our model's accuracy. Our data can prove that the model works the same, and will display the correct result for an overwhelming majority of the time.

5. RELATED WORK

kTrain is a low-code Python library that attempts to make the process of machine learning easier to program [16]. Using kTrain, tasks within the training that would normally require more lines of confusing code would be shortened using their libraries. Furthermore, the library makes each line of code more intuitive and allows the user to have an easier process when writing commands. kTrain, while simplifying the training, does not support users that don't know how to code. Our approach, on the other hand, gives the user the ability to train the model without having to write a single line of code. This gives an abstraction that opens up machine learning to everybody, not just those with a basic understanding of code.

Lobe AI is an application that allows users to gather testing data, train a model, and compute its results without having to write any code [17]. In addition, similar to our application, Lobe allows users to create their own dataset without having to export images. However, since Lobe AI is only supported on the computer, gathering such images using a webcam is not only inconvenient, but also limiting as some computers may not have a webcam that works. However, because our application is supported on smartphones, users can easily take pictures of the data using their phones and thus will have a better user experience.

Levity AI is a software that allows for the automation of images, text, and other documents [18]. By importing images or other pieces of training data, Levity is able to train a model based on such images. However, Levity is not only expensive, but it also does not give users the ability to create their own data-sets. This requires users to go through the time-consuming process of gathering images and exporting them. In contrast, our approach allows users to train models for free, increasing its usability and scope, and also allows users to create their own models based on the images they collect by their own phone camera.

6. CONCLUSIONS

In conclusion, my application allows the user to organize a list of models, train the models, create labels, and create datasets with a mobile phone. Using the images collected by the user, the application uses a server-side approach to train the model, allowing anyone to run tests using the models without having to train the model within the mobile app. Furthermore, we utilized HTTP APIs to add images to the database, get models/labels to load in our User Interface, and add models/labels to our database. We also designed a simple, easy-to-use UI that follows a simple procedure and isn't as convoluted as other similar applications. We conducted three experiments: one to evaluate the completeness of our approach, one to find the efficiency of our approach, and one to determine the accuracy of our approach. The results have shown that our application maintains the same accuracy and confidence as the typical text-based approach to train image-detection models. However, we have also concluded that we have a similar set of functionality with an easy-to-navigate UI and a dynamic structure that allows for easy modification of models. Similarly, we provide a simple approach to modify or add datasets so that the user can easily increase the amount of data used to train the model.

One limitation of our application is the time it would take to gather one image at a time. While taking photos on a phone and storing it to the dataset is already quite efficient, it will still be a tedious process to take pictures one at a time. Because an image-detection model generally requires hundreds of photos for it to be accurate, taking pictures would still be a time-consuming process. Furthermore, our approach only supports image-detection. As machine-learning encompasses other architectures such as data classification and object detection, our app can only be used to service a specific machine learning task.

We plan on creating a system that allows users to upload their own images to our app. Furthermore, we plan on introducing a video system that would allow the user to take pictures in batches at one time at high quantities. Both of these additions would allow the process of gathering data to be even more efficient and less tedious.

REFERENCES

- [1] Flasiński, Mariusz. Introduction to artificial intelligence. Switzerland: Springer International Publishing, 2016.
- [2] Bi, Wenya Linda, et al. "Artificial intelligence in cancer imaging: clinical challenges and applications." *CA: a cancer journal for clinicians* 69.2 (2019): 127-157.
- [3] Kumar, Ashwani, and Ankush Chourasia. "Blind navigation system using artificial intelligence." *International research journal of engineering and technology (IRJET)* 5.3 (2018): 601-605.
- [4] Albawi, Saad, Tareq Abed Mohammed, and Saad Al-Zawi. "Understanding of a convolutional neural network." 2017 international conference on engineering and technology (ICET). Ieee, 2017.
- [5] Srivastava, Shrey, et al. "Comparative analysis of deep learning image detection algorithms." *Journal of Big Data* 8.1 (2021): 1-27.
- [6] Carneiro, Tiago, et al. "Performance analysis of google colaboratory as a tool for accelerating deep learning applications." *IEEE Access* 6 (2018): 61677-61685.
- [7] Jordan, Michael I., and Tom M. Mitchell. "Machine learning: Trends, perspectives, and prospects." *Science* 349.6245 (2015): 255-260.
- [8] Thakkar, Mohit. "Custom core ML models using create ML." *Beginning Machine Learning in iOS*. Apress, Berkeley, CA, 2019. 95-138.
- [9] Kluyver, Thomas, et al. *Jupyter Notebooks-a publishing format for reproducible computational workflows*. Vol. 2016. 2016.
- [10] Sahay, Apurvanand, et al. "Supporting the understanding and comparison of low-code development platforms." 2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA). IEEE, 2020.
- [11] Bojer, Casper Solheim, and Jens Peder Meldgaard. "Kaggle forecasting competitions: An overlooked learning opportunity." *International Journal of Forecasting* 37.2 (2021): 587-603.
- [12] Louis, Marcia Sahaya, et al. "Towards deep learning using tensorflow lite on risc-v." *Third Workshop on Computer Architecture Research with RISC-V (CARRV)*. Vol. 1. 2019.
- [13] Windmill, Eric. *Flutter in action*. Simon and Schuster, 2020.
- [14] Grinberg, Miguel. *Flask web development: developing web applications with python*. " O'Reilly Media, Inc.", 2018.
- [15] Moroney, Laurence, Anglin Moroney, and Anglin. *Definitive Guide to Firebase*. California: Apress, 2017.
- [16] Maiya, Arun S. "ktrain: A low-code library for augmented machine learning." (2020).
- [17] García-Ortiz, Joselin, and Santiago Sánchez-Viteri. "Identification of the Factors That Influence University Learning with Low-Code/No-Code Artificial Intelligence Techniques." *Electronics* 10.10 (2021): 1192.
- [18] Hughes, Larry W., and James B. Avey. "Transforming with levity: Humor, leadership, and follower attitudes." *Leadership & Organization Development Journal* (2009).

PERFORMANCE EVALUATION FOR THE USE OF ELMo WORD EMBEDDING IN CYBERBULLYING DETECTION

Tina Yazdizadeh and Wei Shi

School of Information Technology,
Carleton University, Ottawa, Ontario, Canada

ABSTRACT

Communication using modern internet technologies has revolutionized the ways humans exchange information. Despite the numerous advantages offered by such technology, its applicability is still limited due to problems stemming from personal attacks and pseudo-attacks. On social media platforms, these toxic contents may take the form of texts (e.g., online chats, emails), speech, and even images and movie clips. Because the cyberbullying of an individual via the use of such toxic digital content may have severe consequences, it is essential to design and implement, among others, various techniques to automatically detect, using machine learning approaches, cyberbullying on social media. It is important to use word embedding techniques to represent words for text analysis, typically in the form of a real-valued vector that encodes the meaning of words. The extracted embeddings are used to decide if a digital input contains cyberbullying contents. Supplying strong word representations to classification methods is a key facet of such detection approaches. In this paper, we evaluate the ELMo word embedding against three other word embeddings, namely, TF-IDF, Word2Vec, and BERT, using three basic machine learning models and four deep learning models. The results show that the ELMo word embeddings have the best results when combined with neural network-based machine learning models.

KEYWORDS

Cyberbullying, Natural Language Processing, Word Embeddings, ELMo, Machine Learning.

1. INTRODUCTION

Cyberbullying is a real-life issue that comes from the development and global use of Information and Communication Technology (ICT) solutions in today's life. It endangers everyone's life, especially children, meaning the future psychological health of societies is at real risk. Cyberbullying detection owes its development to many Artificial Intelligence (AI)-based methods. This means a set of semantic and sentiment analysis through data pre-processing, word embeddings, and classification is performed to make sure that the toxic text-based concepts are accurately detected.

The Advancement of ICT has led to the explosion of online communication via social networks and other related applications. Communication enabled by internet technologies has revolutionized modern human interaction. People would like to connect to each other over social media for many reasons, including expressing their ideas and opinions, engaging in forums and discussions, and receiving feedback on their views via interactive media. Despite all the advantages made available by ICT, its applicability is limited due to the problems caused by

personal attacks or pseudo-attacks through the usage of toxic content. Therefore, it is crucial to design and implement various techniques to detect cyberbullying content on social media automatically and evaluate the effectiveness of various approaches.

The Semantic and Sentiment Analysis (SSA) technique[1] is frequently used for cyberbullying detection in texts. In semantic analysis, the meaning of a given text is drawn using computer programs that interpret sentences, paragraphs, or whole documents, by analyzing the grammatical structure and identifying relationships between individual words in a particular context. On the other hand, sentiment analysis employs Natural Language Processing (NLP) techniques, text analysis methods, and in general computational linguistics to systematically identify, extract, quantify, and study affective states and subjective information as what needs to be done to identify cyberbullying contents. Both of these two techniques usually employ supervised Machine Learning (ML) techniques to perform cyberbullying detections. It is essential to use rich datasets to perform training in Neural Networks (NN) and Deep Learning (DL) based solutions.

Word embedding techniques are used to represent the words for text analysis, typically in the form of a real-valued vector that encodes the meaning of the word such that they are closer in the vector space, expected to be similar in meaning. Word embedding paves the way for representing textual data ready to be fed to the ML tools for further analysis toward cyberbullying detection. It is a mapping from the words space with different dimensions to real numbers space with much lower dimensions. Word to Vector (Word2Vec) word embedding model was designed and presented in 2013 by researchers from Google[2]. Bidirectional Encoder Representations from Transformers (BERT)[3] were also proposed by a Google team in 2018.

Embeddings from Language Model (ELMo) was first introduced by Matthew E. Peters et al. as a new type of deep contextualized word representation that models both complex characteristics of words and the procedure through which these vary across linguistic contexts[4]. ELMo can analyze the syntax and semantics of the texts in a very prominent manner. It captures semantic relationships as well as syntactic relationships. That is why it achieves good results in solving the problem of polysemous words and outperform previously existing word embeddings. ELMo has been known as a very effective method for word embedding in many applications. In this paper, we employ ELMo as a word embedding technique that, in conjunction with deep learning models and MLP classifier, has provided us with a novel structure to perform cyberbullying detection on well-known datasets. The proposed structure benefits from the most important and influential tools for word embedding and classification that paves the way for more accurate results. The contributions of this paper are summarized as follows:

- (i) We combine ELMo with Multi-Layer Perceptron (MLP), Decision Tree, and Random Forest to achieve text-based cyberbullying detection. The combination of ELMo with MLP provided us with better results in terms of precision, recall, and F1-score in comparison to the previous research works using MLP with TF-IDF word embedding. To the best of our knowledge, the combination of ELMo with the Decision Tree has not been used previously.
- (ii) We conduct a comparative evaluation of the impact of ELMo word embedding on three basic machine learning models and four deep learning models. Six different datasets were used to evaluate the performance of the models using three metrics. Results demonstrate the advantage of ELMo on cyberbullying detection when combined with neural network-based machine learning models.
- (iii) Among the deep learning models, we combine ELMo with a modified Dense model that leads to further improvement compared to previous research works.

2. LITERATURE REVIEW AND BACKGROUND

In the past years, researchers have done several works on NLP and text analysis in social media for cyberbullying detection. They used a wide variety of Machine Learning (ML) algorithms such as Support Vector Machine (SVM), Ensemble Models, Linear Regression, and Naive Bayes by using Deep Learning (DL) models on different datasets such as Twitter, Facebook, FormSpring, and so on. In this section, we review the most recent and reputable references in the field.

Deep Learning (DL) technique has been used by the authors of [1] and [5]. The main goal of the papers is to ease online communication on textual platforms without being hurt by insults, harassment, and fake news. This is one step forward toward fully AI-based techniques for the detection and prevention toward the protection of a reader being hurt during online chatting. As a general drawback, the computational burden in DL-based techniques is a matter to be addressed. Bidirectional Encoder Representations from Transformers (BERT)[3], as a deep bidirectional, unsupervised language representation capable of creating word embedding (that represents the semantic of the words in the context that they are used) along with other methods is also used in this paper. The four employed deep learning models are Dense, Convolutional Neural Network (CNN), and Long-Short Term Memory (LSTM) layers to detect various levels of toxicity. As for word embedding techniques, the paper has examined Word2Vec[2] and BERT[3] algorithms. To show the performance of the proposed method, the authors have employed the dataset that was released by a Kaggle competition [6] collected from Wikipedia comments, which have been manually labeled into six different toxicity classes.

In another recently published survey paper, the authors have reviewed related works in the literature where word embeddings techniques based on deep learning techniques have been used[7]. Moreover, different types of word embeddings are categorized in this paper. These models need to understand how to pick out keywords that can change the emotion of a sentence. The popular models with the capability of solving such cases are ELMo, OpenAI-GPT, and BERT.

More related to the application discussed in this paper, the effectiveness of the pre-trained embedding model using deep learning methods for classification of emails is examined in [8]. Global Vectors (GloVe) and BERT pre-trained word embedding are employed to identify relationships between words for the categorization of the emails. Well-known datasets like Spam Assassin and Enron are used in the experimentation. In the evaluation phase, the confusion matrix, accuracy, precision, recall, F1-score, and execution time with 10-fold cross-validation are computed for each method. The results show that the CNN model with GloVe embedding gives slightly better accuracy than the model with BERT embedding and traditional machine learning algorithms.

A survey on embeddings in Clinical Natural Language Processing has been given in[9]. Various medical corpora and their characteristics and medical codes have been discussed in this paper. The paper also explores that ELMo generates context-dependent vector representations and hence accounts for the polysemy nature of word embeddings for Out of Vocabulary (OOV), misspelled, and rare words. The main disadvantage of ELMo is computationally intensive, and memory requirements increase with the size of the corpus. ELMo is different from other well-known embedding techniques as it makes use of all the three-layer vectors, i.e., the final representation of a word is obtained as a task-specific weighted average of all the three-layer vectors. ELMo vectors are deep because they come through three-layer vectors and are context-sensitive because they assign different representations to a word depending on its context, which makes it more accurate and versatile. Similar work for studying public opinions on Human Papilloma Virus (HPV) vaccines on social media has been discussed in [10].

Similar to cyberbullying detection, text summarizing has attracted the attention of researchers in the field of NLP[11]. This application is usually performed through two methods, namely, extractive text summarizer and abstractive text summarizer. The paper has focused on retrieving the valuable amount of data using the ELMo embedding in extractive text summarization.

In a recently published paper, the authors have shown the performance of ELMo, where it is applied on a multi-language platform[12]. Similar to other ELMo-based applications, the paper proposes pre-trained embeddings from the popular contextual ELMo model for seven languages, namely, Croatian, Estonian, Finnish, Latvian, Lithuanian, Slovenian, and Swedish. The proposed ELMo model's architecture has three neural network layers, where the first layer is a CNN layer, which operates on a character level. It is followed by two BiLSTM layers, each one consisting of two concatenated LSTMs. Based on the structure of ELMo which is trained on character level and has the ability of handling Out Of Vocabulary words, having a file containing the most common tokens can be useful for training and make the embedding generation more efficient[12]. This paper shows how a proposed method initially designed for a specific language like English may be used for other languages as well.

Toxic context detection has also been studied in [13]. The paper considers embeddings, including BERT and FastText, along with a group of Machine Learning (LR, SVM, DT, RF, XGBoost) and Deep Learning algorithms (CNN, MLP, LSTM). Tokenization, performing basic stemming, and lemmatization techniques are done in the preprocessing phase. In the second phase, various ML algorithms, including Logistic Regression, Support Vector Machine, Decision Trees, Random Forest, and Gradient Boosting, are performed. They merged HASOC'20 and ALONE datasets as one major dataset and performed the evaluations on that. It has been shown that a combination of BERT embedding with CNN gives the best results. It has also shown that CNN understands and efficiently identifies appropriate patterns in the case of small sequences of words and noise in the dataset.

As the importance of word embeddings in the combination of neural-based models, authors in [14] proposed a dense classifier with contextual representations using ELMo to use for classifying crisis-related data on social networks during a disaster. They used real-time Twitter datasets, and analyzed the performance using precision, recall, f1-score, and accuracy. The dense model that they used contains two dense layers, which are a dense layer with Rectifier Linear Unit (ReLU) activation function, and the other one is a dense layer with a softmax function. The proposed combination of the dense classifier with ELMo representations gives better accuracy than the traditional classifier, such as SVM and deep learning classifiers CNN and MLP.

Another use of ELMo in text mining, especially in biomedical text classification, may be referred to [15], where they proposed both deep and shallow network approaches, and their predictions are based on the similarity between extracted features from contextualized representations of the words in their dataset. As the word representations, they considered ELMo and BERT. In addition, they proposed transfer learning by adding a dense layer to the pre-trained ELMo model. Their dataset is from the PubMed repository, which has records including biomedical citations and abstracts in an XML format. As one of their results, the ELMo classifier, in combination with one dense layer, outperforms other methods.

It is normal to have noisy data in NLP as the data is mainly collected from crawling the social media where people write their opinions in different formats and languages. Different type of character and word level methods are used by authors in [16] to simulate setups in which input may be somewhat noisy or different from the data distribution on which NLP systems were trained. They evaluated the performance of well-known deep contextualized word embeddings such as ELMo, BERT, XLNet, and RoBERTa. They used BERT, RoBERTa, and XLNet as both

words embedding generators and classifiers, but the word representations provided by the ELMo were fed into one dense layer. The results suggest that some language models can manage specific types of noise more efficiently than other models. ELMo achieved higher scores than BERT, even XLNet, and RoBERTa on some character-level perturbations.

Deep learning algorithms, coupled with word embeddings in detecting cyberbullying texts, are the topic of much research work[17]. In a matrix of choices, three deep learning algorithms, namely GRU, LSTM, and BiLSTM, in conjunction with word embeddings models, including word2vec, GloVe, Reddit, and ELMO models, are used to examine the effectiveness and accuracy of a possible configuration for cyberbullying detection. Similar to many other research works, data preprocessing steps, including oversampling, is performed on the selected datasets related to social media. A typical dataset in the literature, namely, Formspring. me, has been used for performance evaluation. Form spring. me is basically a social site that provides a platform for users to ask any question to any other users. It consists of 12,772 posts. Based on extensive experimental results, BiLSTM performs best with ELMo in detecting cyberbullying texts. As another performance index, the average time taken for the training of each model has also been measured based on which GRU outperforms compared to other methods.

As another survey on the use of a deep learning model in combination with deeply contextualized word embeddings such as BERT, and ELMo, one may refer to [18]. In this paper, the authors conducted experiments to study both classic and contextualized word embeddings in text classification. As the encoder for the sequence of text, they employed CNN and BiLSTM. They selected four different benchmarking classification datasets with variable average sample lengths, which are 20NewsGroup, The Stanford Sentiment Treebank dataset, the arXiv Academic Paper dataset, and Reuters-21578 (Reuters). In addition, they considered both single-label classification and multi-label classification. This study claims that selecting CNN over BiLSTM for document classification tasks is better than for sentence classification datasets. As the second task in this study, they applied CNN and BiLSTM on both ELMo and BERT. Based on reported results, BERT surpasses ELMo, especially for lengthy datasets. As a comparison with classic embeddings, both achieve improved performance for short datasets, while the improvement is not observed in more extended datasets.

3. METHODOLOGY

In this section, the proposed methodology is described in detail in three stages: pre-processing steps for dataset preparation, then word embedding phase followed by various classification methods.

3.1. Required Pre-processing

One of the most important steps in cyberbullying detection is text pre-processing. The common techniques include stop words and punctuation removal, lemmatization, stemming, and emoticon and URL removal [19]. The stop words are referred to as the most commonly used words in any language, such as articles, prepositions, pronouns, and so on. The next step is to generate the text representation. The embeddings are generated following different feature engineering processes. In this study, some of the stop words are maintained because they can enrich the semantics of the text and make improvements to the results [5]. The two performed pre-processing steps are text conversion to lower case and padding and truncating the sentences to a certain number of words as the neural network models need to have input with the same shape and size.

3.2. ELMo Word Embedding

Having pre-processed text, the input is ready to be fed to the selected embedding model. In this study, we choose the ELMo word embedding proposed by [4]. By using Bi-directional Language Models (BiLM), this word embedding provides two passes in its structure, which are forward passed, and backward pass. Unlike the other word embeddings such as Glove and Word2Vec, ELMo uses the complete sentence for generating the representation for a word in the sentence. In this study, for the ML algorithms, the ELMo representations are generated separately using the AllenNLP ELMo library [20]. The ELMo word representations are fed to the ML models as the input. For the DL models, a function was defined for the embedding layer, which used the ELMo embedding function from the TensorFlow hub. The signature parameter of the ELMo function is selected as default because the input type is not tokenized. The output of ELMo word embedding is a tensor with the shape of [batch-size, max-length, 1024]. The max length in this study is selected as 100 words per sentence.

3.3. Classification Methods

In the classification phase, various ML classification techniques are used in this study. We briefly describe each classification method with related models in the next few paragraphs.

For the deep learning classification methods, we used the same models used in [1]. As a general description for all DL models, they all have the same number of layers and are structured with an embedding layer for mapping the input text to the word representations. The last layer for all models is a Dense layer, which provides a single binary label as the result of an input. The sigmoid function is used as the activation function.

The Dense model is comprised of three Dense layers with 1024, 64, and 1 neuron. They can reduce the input size of numerous nodes to a few nodes with weights that can be used to predict the label of the input. This is because they are densely connected layers. The difference between our Dense architecture with the ones in the literature [14], [15], and [16] is in the number of layers and the activation function. As mentioned before, the authors in [14] used two-layer of dense, while in this study, we used three dense layers with a different number of neurons. Moreover, in [14], researchers used softmax as the activation function while we used sigmoid as the activation function. Two other papers used only one dense layer in their studies.

The CNN model has two layers, which perform the filtering operation. With its configuration, it extracts the more important features of the text. The kernel size for the first layer is ten and for the second layer is 5. All the layers in this model have the same number of neurons as mentioned in Dense layers so that better comparison can be performed.

The LSTM model is an updated version of Recurrent Neural Networks (RNN). It uses two LSTM layers to perform the classification. This model uses memory blocks to keep the record of the computations. This can help the model to understand the semantic patterns of historical input data and use them in the currently processed data. As the development of the LSTM model, the BiLSTM model uses the bidirectional LSTM layers, which process the training data in two directions, forward and backward, and pass to LSTM hidden layer, and then the results are combined by a shared output layer.

The remaining ML algorithms investigated in this study are MLP, Decision Tree, and Random Forest. The MLP model is composed of a single layer with 100 nodes. The Decision Tree builds a model where the data is continuously split according to specific parameters. The algorithm starts with a root node and is divided into children nodes according to a given set of rules. The Random

Forest model is composed of multiple Decision Trees. By using the majority votes, it chooses the best output as the final label for the input. The number of decision tree estimators used in this study is 100.

4. COMPARATIVE EVALUATION

In this section, after a brief description of the dataset and the experimental setup, the results of ELMo embedding applied to different groups of ML models are reported. Thereafter, a comparative evaluation of the results obtained in this study and the results provided by [1] is presented.

4.1. Dataset Description

We used the dataset released by the Kaggle competition[6]. This dataset is gathered from Wikipedia comments, which have been manually labelled into six different toxicity classes. The dataset has more than 200K comments presenting the labels for six different toxicity classes, which are toxic, severe toxic, obscene, threat, insult, and identity hate. The original dataset is reported as a strongly unbalanced dataset, and it caused a biased training procedure. The authors in [1] provided balanced datasets for each toxicity class where the datasets have an equal number of toxicity examples and the number of non-toxicity examples. Table 1 shows the number of examples in each dataset.

Table 1. Distribution of six classes

Dataset	Toxic	Severe Toxic	Obscene	Threat	Identity Hate	Insult
Num. Records	42768	3924	24280	1378	22608	4234

4.2. Experiment Setup and Evaluation Metrics

The experiments were run on 5-fold cross-validation, and the selected batch size for each model is 8. The models are trained in 5 epochs, and a binary cross-entropy is selected as the loss function. The optimizer is Adam, with the default learning rate of 0.002 provided by the library. To implement the ML algorithms, we used the Scikit-learn library. All the other parameters are based on the model's performance and previous experiences in the competitor's work. The experiments have been done on Google Colab GPU with High RAM of 26 GB memory.

We report the Precision, Recall, F1-Score, and accuracy of the cyberbullying detection results in this study. The Precision, Recall, and F-score are computed according to Equations 1, 2, and 3, respectively. The parameters used in these equations are True Positive (TP) which shows the number of correct instances guessed by the implemented models, and False Positive (FP), which is the number of false predicted instances by models. Moreover, False Negative (FN), which shows the number of instances erroneously associated with a wrong class is used in Recall equations.

$$Precision(P) = \frac{TP}{TP + FP} \quad (1)$$

$$Recall(R) = \frac{TP}{TP + FN} \quad (2)$$

$$F - measure(f) = 2 \times \frac{P \cdot R}{P + R} \quad (3)$$

4.3. Results and Analysis

In this section, we discuss the results of the experiments that we have performed. The results are divided into three tables which contain the results of the baseline paper [1] and current research results on precision, recall, and F1 score. The authors of [1] compared the effect of TF-IDF word embedding on three ML models. In this study, the effect of ELMo word embedding is evaluated on four deep learning models and three basic machine learning models. We then compare the results against the combination of TF-IDF in the same three basic ML models and the effect of Word2Vec and BERT embeddings on the same four DL models. It is worth mentioning that, since authors in [1] used three different versions of Word2Vec(pre-trained, domain-trained, and Mimicked) and based on their result analysis, the mimicked Word2Vec achieved the best results. Therefore, in this study, we compare our results against the Mimicked Word2Vec.

Table 2. Comparison of precision of ELMo against TF-IDF using three basic ML algorithms on six different datasets

	Feature	Toxic	Sever Toxic	Obscene	Threat	Identity Hate	Insult
Decision Tree	TF-IDF	0.859	0.847	0.926	0.917	0.819	0.887
	ELMo	0.661	0.751	0.690	0.749	0.838	0.701
Random Forest	TF-IDF	0.860	0.888	0.945	0.954	0.847	0.929
	ELMo	0.800	0.890	0.821	0.865	0.861	0.822
MLP	TF-IDF	0.849	0.913	0.884	0.914	0.889	0.871
	ELMo	0.855	0.901	0.891	0.937	0.902	0.872

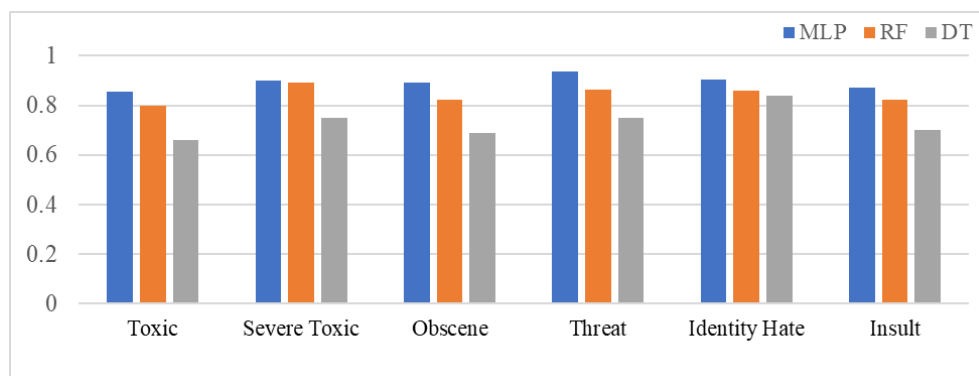


Fig.1. Comparison of ELMo based on precision using ML models

As shown in Table 2, we performed both TF-IDF and ELMo word embedding on MLP, Random Forest, and Decision Tree models. The obtained results show that ELMo outperforms TF-IDF when it is combined with the MLP model on precision. Moreover, we can see from Fig. 1 that ELMo word embedding has the best results on MLP compared to using Random Forest and Decision Tree models.

The reason behind it could be because the structure and functionality of the tree-based models. The tree-based models split features of a dataset and predict the labels in the leaf nodes. Having this fact in mind, the tree-based models can perform better on the datasets that have more features to split the tree based on that attribute. In our case, the only component of the dataset is the text of comments that converts to word representations. This way, the tree-based models do not get to use many attributes, however, still be able to calculate how much the representations are correlated to the labels.

Table 3. Comparison on the precision of ELMo against BERT and mimicked Word2Vec using DL models on six different datasets

	Feature	Toxic	Sever Toxic	Obscene	Threat	Identity Hate	Insult
<i>Dense Model</i>	Mimicked	0.868	0.926	0.880	0.933	0.881	0.873
	BERT	0.828	0.912	0.844	0.867	0.874	0.841
	ELMo	0.838	0.845	0.891	0.801	0.861	0.879
<i>CNN Model</i>	Mimicked	0.836	0.886	0.856	0.927	0.860	0.847
	BERT	0.801	0.899	0.819	0.842	0.824	0.831
	ELMo	0.874	0.912	0.859	0.900	0.888	0.860
<i>LSTM Model</i>	Mimicked	0.895	0.941	0.928	0.953	0.887	0.916
	BERT	0.866	0.927	0.889	0.916	0.880	0.874
	ELMo	0.681	0.962	0.943	0.970	0.943	0.961
<i>BiLSTM Model</i>	Mimicked	0.910	0.939	0.929	0.941	0.902	0.920
	BERT	0.875	0.933	0.892	0.913	0.900	0.889
	ELMo	0.680	0.951	0.944	0.974	0.951	0.937

Among the four DL models that are implemented using three-word embeddings, ELMo embedding outperforms Mimicked Word2Vec and BERT in most categories of CNN, LSTM, and BiLSTM models. Specifically, in the LSTM model, using ELMo word embeddings provided a good improvement in terms of precision with a minimum of 2% and a maximum amount of 5%. The ELMo model does not perform very well on the Toxic dataset among all the models. As mentioned before, the pre-processing steps did not act on these datasets because the punctuations and stop words have effects on the semantics of the sentence. Since the Toxic dataset is the largest, having more irrelevant words are unavoidable. Hence, these results are likely the result of having more stop words such as "the", "is", and so on in the Toxic dataset.

In Tables 4 and 5, we report results on the recall values obtained when different models and embeddings are combined. In general, the results obtained on recall are symmetrically better than precision ones in the combination of ELMo embeddings and all ML models. Based on the definition of precision and recall, higher recall means that the model predicts the most relevant results, and higher precision means that the model returns more relevant results than irrelevant ones. In other words, based on the definition of False Positive and False Negative, which are mentioned in section 4.2, getting a false negative has a much more significant impact than having a false positive in cyberbullying detection because the false negatives in cyberbullying detection mean the bullying comments are predicted as non-bullying ones while the false positives mean the non-bully instances are predicted as bullying contents. The goal of cyberbullying detection is to find and predict the correct bully instances and prevent the occurrence. Therefore, if the model predicts the bully instances as the non-bully ones, then the damage is bigger. Thus, having lower false negatives can help to have better recall due to the nature of this study.

Table 4. Comparison of recall of ELMo against TF-IDF using three basic ML models on six different datasets

	Feature	Toxic	Sever Toxic	Obscene	Threat	Identity Hate	Insult
Decision Tree	TF-IDF	0.855	0.947	0.929	0.891	0.927	0.891
	ELMo	0.657	0.770	0.691	0.791	0.820	0.690
Random Forest	TF-IDF	0.856	0.940	0.834	0.897	0.911	0.851
	ELMo	0.718	0.855	0.760	0.821	0.890	0.752
MLP	TF-IDF	0.857	0.918	0.895	0.916	0.897	0.880
	ELMo	0.859	0.927	0.871	0.921	0.978	0.895

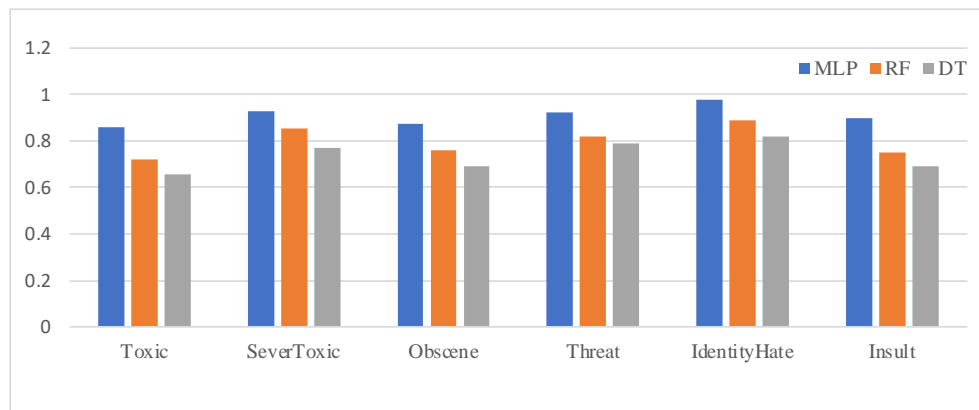


Fig 2. Comparison of ELMo based on Recall using three basic ML models

Similar to what is presented in Table 2, the TF-IDF performs better than ELMo when it is used in Random Forest and Decision Tree models. The combination of ELMo and MLP underperforms slightly compared to using TF-IDF on the Obscene dataset. The comparison between the combination of ELMo and three basic ML models is shown in Fig2. ELMo embedding demonstrated better results only when combined with MLP compared to the integration of ELMo with the other two basic ML models.

Table 5. Comparison of Recall of ELMo against BERT and Mimicked Word2Vec using DL model on six different datasets

	Feature	Toxic	Sever Toxic	Obscene	Threat	Identity Hate	Insult
Dense Model	Mimicked	0.844	0.914	0.877	0.932	0.882	0.857
	BERT	0.817	0.917	0.821	0.891	0.865	0.827
	ELMo	0.905	0.929	0.871	0.970	0.920	0.890
CNN Model	Mimicked	0.865	0.919	0.870	0.918	0.879	0.849
	BERT	0.812	0.911	0.832	0.872	0.842	0.821
	ELMo	0.857	0.920	0.863	0.899	0.884	0.880
LSTM Model	Mimicked	0.938	0.966	0.938	0.962	0.946	0.948
	BERT	0.851	0.932	0.861	0.899	0.895	0.870
	ELMo	0.889	0.949	0.952	0.951	0.982	0.952

<i>BiLSTM</i>	Mimicked	0.921	0.963	0.945	0.944	0.934	0.935
<i>Model</i>	BERT	0.841	0.941	0.852	0.900	0.857	0.866
	ELMo	0.869	0.984	0.959	0.953	0.971	0.960

The results of ELMo on Dense, LSTM, and BiLSTM models are better than Mimicked Word2Vec and BERT. Although authors stated in [1] that the Dense model had the worst results among the other deep learning models, in this study, we found out that the combination of the Dense model with ELMo improves the outcomes against the combination of this model with BERT, mimicked Word2Vec.

Table 6. Comparison of F1-score of ELMo against TF-IDF using three basic ML models on six different datasets

	Feature	Toxic	Sever Toxic	Obscene	Threat	Identity Hate	Insult
<i>Decision Tree</i>	TF-IDF	0.857	0.894	0.928	0.903	0.869	0.889
	ELMo	0.670	0.761	0.700	0.783	0.841	0.719
<i>Random Forest</i>	TF-IDF	0.858	0.913	0.913	0.924	0.877	0.888
	ELMo	0.761	0.871	0.791	0.846	0.879	0.790
<i>MLP</i>	TF-IDF	0.853	0.915	0.889	0.913	0.893	0.876
	ELMo	0.860	0.918	0.901	0.929	0.881	0.883

Table 7. Comparison of ELMo against BERT and Mimicked Word2Vec based on F1-score using DL models on six different datasets

	Feature	Toxic	Sever Toxic	Obscene	Threat	Identity Hate	Insult
<i>Dense Model</i>	Mimicked	0.844	0.914	0.919	0.931	0.880	0.863
	BERT	0.855	0.917	0.913	0.877	0.855	0.834
	ELMo	0.905	0.929	0.946	0.970	0.880	0.880
<i>CNN Model</i>	Mimicked	0.865	0.919	0.901	0.922	0.869	0.847
	BERT	0.812	0.911	0.904	0.849	0.832	0.826
	ELMo	0.857	0.920	0.918	0.881	0.883	0.870
<i>LSTM Model</i>	Mimicked	0.916	0.966	0.953	0.957	0.914	0.931
	BERT	0.858	0.932	0.929	0.907	0.886	0.872
	ELMo	0.760	0.949	0.955	0.960	0.961	0.970
<i>BiLSTM Model</i>	Mimicked	0.915	0.963	0.951	0.940	0.916	0.927
	BERT	0.856	0.941	0.937	0.905	0.874	0.877
	ELMo	0.760	0.980	0.970	0.960	0.960	0.950

Tables 6 and 7 show the results of the F1 score on different word embeddings on different DL models. The combination of MLP and ELMo outperforms all other DL models. From the DL perspective, the BiLSTM model, which has a complex architecture, gets the best results in combination with ELMo. This combination has outdone the others with a minimum improvement of 2% and a maximum improvement of 4%. It is interesting to observe that the combination of ELMo with the Dense model has the best results against BERT and Mimicked word2vec word embeddings in all six datasets. This combination obtains the same result as the combination of the Dense model and Mimicked word2Vec just on the Identity hate dataset, which is still the

highest outcome for this dataset. Again, ELMo does not provide good representations for the Toxic dataset.

From the results, we conclude that the TF-IDF algorithm is a good choice as a word embedding for resources to be parsed with ML models such as Random Forest and Decision Tree. Moreover, the results suggest that ELMo word embeddings could be a good choice for ML algorithms which has a neural network-based, such as MLP, because the structure of ELMo word embeddings is based on a two-layer bidirectional language model which has two passes, forward pass, and backward pass, which solves the problem of polysemy in word representation.

Surprisingly, between BERT and ELMo embeddings, BERT performs worse on this task. The authors in [1] think the reason that caused BERT's undesirable results is that assigning a different embedding to the same word is confusing to the training of the DL models. However, as mentioned above, the strength of ELMo is that it can take the entire input sentence into an equation when calculating the word embeddings. Therefore, the selected word would produce different ELMo vectors in different contexts.

5. CONCLUSION AND FUTURE WORK

In today's age of Information and Communication Technology, the availability of detection systems to prevent the spread of harassment and cyberbullying behaviour promotes a safer and healthier adoption of social media platforms. The core of cyberbullying detection systems is composed of word embedding and classification techniques, for which AI-based solutions are essential. In this paper, we considered ELMo-based methods as word embedding techniques combined with Dense, CNN, LSTM, and the BiLSTM methods as deep learning models and MLP, Random Forest, and Decision Tree as other machine learning classification techniques. The rich datasets from the Kaggle competition were used for performance and comparative evaluations. The practical results show that the combination of ELMo word embedding with most of the deep learning models outperforms other combinations of word embeddings and deep learning models. Moreover, it is interesting to observe that combining ELMo word embedding with MLP, which is a neural network-based model, produces better results than other machine learning algorithms. For future work and as a necessary step toward a real-life application of cyberbullying detection, we will investigate, in the immediate future, the use of an online scheme for ELMo word embedding and classification.

ACKNOWLEDGEMENT

We gratefully acknowledge the financial support from the Natural Sciences and Engineering Research Council of Canada (NSERC) under Grant No. RGPIN-2020-06482.

REFERENCES

- [1] D. Dessì, D. R. Recupero, and H. Sack, (2021) "An Assessment of Deep Learning Models and Word Embeddings for Toxicity Detection within Online Textual Comments," *Electronics*, vol. 10, no. 7, p. 779, doi: 10.3390/electronics10070779.
- [2] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, (2013) "Distributed Representations of Words and Phrases and their Compositionality," in *Advances in Neural Information Processing Systems*, vol. 26. Accessed: May 31, 2022. [Online]. Available: <https://proceedings.neurips.cc/paper/2013/hash/9aa42b31882ec039965f3c4923ce901b-Abstract.html>
- [3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, (2019) "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language*

- Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota, pp. 4171–4186. doi: 10.18653/v1/N19-1423.
- [4] M. E. Peters *et al.*, (2018) "Deep Contextualized Word Representations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, New Orleans, Louisiana, pp. 2227–2237. doi: 10.18653/v1/N18-1202.
- [5] H. H. Saeed, K. Shahzad, and F. Kamiran, (2018) "Overlapping Toxic Sentiment Classification Using Deep Neural Architectures," in *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, Singapore, Singapore, pp. 1361–1366. doi: 10.1109/ICDMW.2018.00193.
- [6] "Toxic Comment Classification Challenge." <https://kaggle.com/c/jigsaw-toxic-comment-classification-challenge> (accessed Sep. 22, 2021).
- [7] B. Wang, A. Wang, F. Chen, Y. Wang, and C.-C. J. Kuo, (2022) "Evaluating word embedding models: methods and experimental results," *APSIPA Trans. Signal Inf. Process.*, vol. 8, no. 1, 2019, doi: 10.1017/ATSIP.2019.12.
- [8] D. S. Asudani, N. K. Nagwani, and P. Singh, (2021) "Exploring the effectiveness of word embedding based deep learning model for improving email classification," *Data Technol. Appl.*, doi: 10.1108/DTA-07-2021-0191.
- [9] K. S. Kalyan and S. Sangeetha, (2020) "SECNLP: A survey of embeddings in clinical natural language processing," *J. Biomed. Inform.*, vol. 101, p. 103323, doi: 10.1016/j.jbi.2019.103323.
- [10] L. Zhang, H. Fan, C. Peng, G. Rao, and Q. Cong, (2020) "Sentiment Analysis Methods for HPV Vaccines Related Tweets Based on Transfer Learning," *Healthcare*, vol. 8, no. 3, p. 307, doi: 10.3390/healthcare8030307.
- [11] H. Gupta and M. Patel, (2020) "Study of Extractive Text Summarizer Using The Elmo Embedding," in *2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, Palladam, India, pp. 829–834. doi: 10.1109/I-SMAC49090.2020.9243610.
- [12] M. Ulčar and M. Robnik-Šikonja, (2020) "High Quality ELMo Embeddings for Seven Less-Resourced Languages," in *Proceedings of the 12th Language Resources and Evaluation Conference*, Marseille, France, pp. 4731–4738. Accessed: May 31, 2022. [Online]. Available: <https://aclanthology.org/2020.lrec-1.582>
- [13] P. Malik, A. Aggrawal, and D. K. Vishwakarma, (2021) "Toxic Speech Detection using Traditional Machine Learning Models and BERT and fastText Embedding with Deep Neural Networks," in *2021 5th International Conference on Computing Methodologies and Communication (ICCMC)*, Erode, India, pp. 1254–1259. doi: 10.1109/ICCMC51019.2021.9418395.
- [14] S. Madichetty and S. M., (2020) "Improved Classification of Crisis-Related Data on Twitter using Contextual Representations," *Procedia Comput. Sci.*, vol. 167, pp. 962–968, doi: 10.1016/j.procs.2020.03.395.
- [15] D. A. Koutsomitropoulos and A. D. Andriopoulos, (2022) "Thesaurus-based word embeddings for automated biomedical literature classification," *Neural Comput. Appl.*, vol. 34, no. 2, pp. 937–950, doi: 10.1007/s00521-021-06053-z.
- [16] M. Moradi and M. Samwald, (2021) "Evaluating the Robustness of Neural Language Models to Input Perturbations," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic, pp. 1558–1570. doi: 10.18653/v1/2021.emnlp-main.117.
- [17] M. Al-Hashedi, L.-K. Soon, and H.-N. Goh, (2019) "Cyberbullying Detection Using Deep Learning and Word Embeddings: An Empirical Study," in *Proceedings of the 2019 2nd International Conference on Computational Intelligence and Intelligent Systems*, Bangkok Thailand, pp. 17–21. doi: 10.1145/3372422.3373592.
- [18] C. Wang, P. Nulty, and D. Lillis, (2020) "A Comparative Study on Word Embeddings in Deep Learning for Text Classification," in *Proceedings of the 4th International Conference on Natural Language Processing and Information Retrieval*, New York, NY, USA, pp. 37–46. doi: 10.1145/3443279.3443304.
- [19] D. Dessì, G. Fenu, M. Marras, and D. Reforgiato Recupero, (2019) "Bridging learning analytics and Cognitive Computing for Big Data classification in micro-learning video collections," *Comput. Hum. Behav.*, vol. 92, pp. 468–477, doi: 10.1016/j.chb.2018.03.004.
- [20] "AllenNLP - ELMo — Allen Institute for AI." <https://allennlp.org/allennlp/software/elmo> (accessed May 29, 2022).

AUTHORS**Tina Yazdizadeh**

Tina is a Master of Information Technology with a specialization in Data Science student at Carleton University, Ottawa, Canada. Her current research is focused on the intersection of the very demanding fields, namely, "Text Mining" and "Cybersecurity". Before joining the Department of Information Technology at Carleton University, she had received her B.Sc. in Computer Engineering (Software) from the University of Tehran. Her B.Sc thesis was on Map Matching Using GPS Data, a research work which was supported by TAPSI Co. as a growing E-Taxi company very similar to Uber.

**Wei Shi**

Dr. Wei Shi is a Professor in the School of Information Technology, cross-appointed to the Department of Systems and Computer Engineering in the Faculty of Engineering & Design at Carleton University. She specializes in algorithm design and analysis in distributed environments such as Data Centers, Clouds, Mobile Agents, Actuator systems, and Wireless Sensor Networks. She has also been conducting research in data privacy and Big Data analytics. She holds a Bachelor of Computer Engineering from Harbin Institute of Technology in China and received her master's and Ph.D. in Computer Science from Carleton University in Ottawa, Canada. Dr. Shi is also a Professional Engineer licensed in Ontario, Canada.



© 2022 By AIRCC Publishing Corporation. This article is published under the Creative Commons Attribution (CC BY) license.

.

AN INTELLIGENT FOOD INVENTORY MONITORING SYSTEM USING MACHINE LEARNING AND COMPUTER VISION

Tianyu Li¹ and Yu Sun²

¹St. George's School, 4175 W 29th Ave, Vancouver, BC V6S 1V1, Canada

²California State Polytechnic University,
Pomona, CA, 91768, Irvine, CA 92620

ABSTRACT

Due to technological advancements, humans are able to produce more food than ever before. In fact, the food production level is so high that all population could be supported if the food resource is distributed correctly. Yet, it is more than common to see items left expiring on the supermarket shelves, wasting the food resource that could otherwise be useful. Neither are the adverse impacts on the climate due to food disposal in anyone's favor or interest. This paper proposes an application to identify the stock status of supermarket items, specifically food items, so that supermarket managers can react to the selling status and prevent oversupply. The key tool implemented in the application is computer vision, specifically YOLOv5, which uses convolutional neural networks [1]. The model automatically recognizes and counts the items in a taken picture. We applied our computer vision model to numerous supermarket shelf photos and conducted an evaluation of the model's precision and speed. The results show that the application is a useful tool for users to log supermarket stock information since the computer vision model, despite lacking slightly in object detection precision, can return a reliable count for well-taken photos. As a platform where such information is shared, the application is therefore a viable tool for store managers to import amounts of food accordingly and for the public to be informed and make smart buying choices.

KEYWORDS

Flutter, YOLOv5, Computer Vision, Inventory Management.

1. INTRODUCTION

Food waste is detrimental, especially in this environmentally stressed world: not only the food itself but the expended resources and energy – including irrigation and transportation – are wasted. One of the main culprits is the supermarket; among the 931 million tons of food waste each year, 13% comes from retail [2]. Exacerbating the problem, 44.7% of food waste in 2020 will be diverted to landfill, the least favorable disposal method [3]. Food landfills produce a large amount of methane, an extremely potent greenhouse gas; about 7% of global greenhouse gas is due to preventable food waste [4]. Besides, the estimated annual food waste cost in the North American economy is about \$278 billion [5]. As seen above, food waste is both an economically tolling and environmentally-damaging problem that affects virtually everyone in the world.

In order to eliminate the problem, it is therefore most favorable to limit the amount of food waste. There have been countless instances of fully loaded supermarket shelves of items not being sold anywhere before their expiration, by which time the retailers have no choice but to discard the

food products. This could be attributed to either managers losing track of consumers' demand or consumers not knowing where the items are available at. The proposed solution solves the problem by offering both sides foresight into a supermarket's supply. The application increases the transparency of food stocks in the supermarket and automates the process using computer vision. Managers can then monitor their inventories according to the demands and adjust to the data over time.

Many techniques have been implemented to limit the amount of food waste in grocery stores. One commonality among many of them is the usage of technology. Some supermarkets use technology as a means to track the expiration dates of products. This allows the retailers to discount the near-expired products and earn profits from that, and technology saves retailers time as it automates the process [6]. Others use technology to digitalize the layout of their stores. Information of products in the inventory are then more accessible, and retailers will no longer have to rely on intermediaries when importing from warehouses, decreasing the amount of perishables that previously exist due to inefficiencies in handling [7]. Yet, such ideas are still generally tentative as they are currently being experimented with in a few stores.

Besides technology, grocery stores can adopt alternative supply practices. To start with, they can partner with their suppliers by actively communicating consumers' demand for the different food items. Some agri-tech companies have further supported this collaboration by sharing different retailers' market information with farmers on application software [8]. This ultimately allows farmers to plan their production better and make use of potential wastes, such as producing energy [9]. However, the communication between stores and farms can sometimes be inefficient as there could be a time-lag when stores receive products, and some stores' tracking of items is inaccurate.

Another popular waste-reduction method is making use of all products. Instead of rejecting the imperfect-looking food, which are food that are not as good as others in appearance, supermarkets promote them. For instance, grocery stores like Morrison's sell these foods at discounted prices. In addition, instead of discarding surplus produce into landfill, supermarkets convert them into produce or distribute them to people who need the food [10]. These methods vitally take advantage of the food produced, but the key shortcoming is that they are short-term practices that require a high level of civilians' stewardship, which may be hard to achieve in some communities.

In this paper, our proposed solution is a real-time digital grocery stock tracking system that provides counts and images of supermarket items, which can be provided by the users of the application. The application shows a list of supermarkets and collects stock information of food items in the supermarkets. Users can see the exact location of a desired item by clicking a supermarket. In addition, users can plan their purchase by searching for certain items. The item stock information at different store locations will be retrieved, and users can select the number of items and where to purchase in their trip.

This proposed solution encompasses all the existing methods. The application is a technology that practically digitalize grocery stores. Consumers can have a more holistic perspective of items in different stores around them. They can also help the stores update inventories by simply taking a photo, which is then analyzed by AI. Store managers can have clear data on their items. This makes their communication with the suppliers much more efficient since the trends of the data allow their future imports of goods to be more oriented. Therefore, by providing food items at amounts that suit the consumers' needs, supermarkets can reduce the amount of perishables that would be left expired and wasted but make use of them fresh instead.

In two application scenarios, we demonstrate how implementing the computer vision model as an integral functionality into the application increases both the utility and usability of the application through its efficiency in updating stock information. First, the accuracy of the computer vision model is evaluated in two components – object detection and counting. The model labeled a set of validation and test images, and its performance in object detection was measured by four key metrics – precision, Mean Average Precision (mAP), recall, and General Intersection Over Union (GIOU). While the returned labels were slightly less accurate, the model could localize the items to operate on. Moreover, the model's counts in several cases grocery images were manually checked over. It accurately detected and counted items in large-scale, upright-angled, and high-resolution pictures. By taking proper item-specific photos, users can update the inventory accurately and more quickly for their local communities with the numbers returned by the embedded computer vision model.

Second, the speed of the computer vision model is gauged through the `timeit` module in Python. The model, which runs on a backend server, is tasked with 60 photos of various supermarket items, and the execution time of the model in counting items in each photo is recorded. Key statistical parameters, such as mean and standard deviation, are calculated and convincing that the computer vision model is quick in counting items. By providing a count in a timely fashion, the computer vision model makes updating the inventory convenient for users, and more people will, therefore, be inclined to download the application and limit food waste.

The rest of the paper is organized as follows: Section 2 outlines and details the challenges in developing the computer vision and application; Section 3 elaborates on how the proposed solution works; Section 4 presents experiments that test the efficiency of the solution and analyzes the solution; Section 5 lists related works that address the problem of grocery food waste; Section 6, finally, gives the conclusion remarks and points out future works of this project.

2. CHALLENGES

To develop a helpful, user-friendly application system, several challenges have been identified as follows.

2.1. Designing the layout and functions of the application

The key to the success of an application is its user-friendliness. This characteristic is often shown through how easily the users can navigate through the application and the functions that users need to benefit from the application. As a result, a lot of efforts have been dedicated to designing how all the stock information should be presented. This process is the most time-consuming as multiple possibilities of the layouts were tested before finalizing on one. Eventually, the dashboard shows all stores with elapsed information in connected pages. The camera option is embedded inside the item page so that the users can take a photo of the stock when looking for the items; this also makes the most sense because the information analyzed from the photo can then most easily be directed to the according location in the application. To further improve, perhaps including a business analysis for managers or a wishlist for consumers can build a community for the application.

2.2. Accommodating all the various grocery items in the dataset

In an average supermarket, there are about 40000 products on the innumerable shelves [11]. What differentiates the products are what the products are and the producing or manufacturing company, which sometimes is one of the consumers' decision factors. Theoretically, in order for

the machine to recognize and discern between the products, it has to be trained with all the products, which will require a large amount of data. Feeding in such a large dataset is already an almost impossible task, and the increased variety of products simply makes the model more prone to errors and inaccuracies when identifying and counting the items. As a result, it is impractical to attempt to apply a model to every item. Alternatively in the solution, most similar items are regrouped to one general group. For instance, all bread items – including whole wheat, white, or grain – are all under one category of bread, which is trained with images of the various sub-products. This ensures that while there is less clue as to the specifics of the items, the products can still be identified and counted.

2.3. Creating a secure server for the application

The computer vision model embedded in the camera function is a central feature that processes inventory photos, and it is hosted on an online server in the backend. Since the model should operate whenever a photo is taken, the server needs to be accessible at all times. An HTTP server was first deployed on Repl.it for the application because it leverages the Flask web framework, which comes with established libraries and is flexible for implementation. When testing with real-time photos, however, the server, despite running autosynchronously at first, had preserving connection and compatibility issues, which crashed the application and rendered it counterproductive. Consequently, the model was integrated into the Amazon Web Services (AWS) cloud server instead. The AWS server offered a stable connection once launched, and it efficiently returns object detection and counting outputs to the application user interface (UI) upon requests. This keeps the application functional as users can update inventories by handily taking photos using the camera.

3. SOLUTION

This application provides users with the amount, location, and image of different food items in grocery stores of close geographical proximity. As shown in Figure 1, the system is composed of three main pages to achieve the purpose: Stores, Wishlist, and Me.

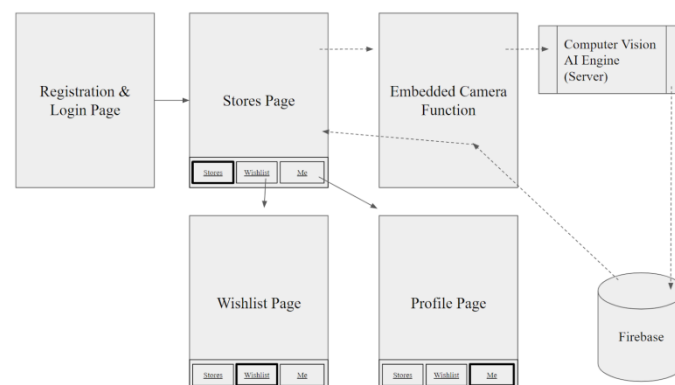


Figure 1. Application Overview

In the “Stores” page, users can access all the aforementioned information by clicking through each store or typing the desired items in the search bar above. Items of names related to the one typed in the search bar will be identified and included in the drop down menu for selection. The camera function embedded in the stores page is the only means to update the inventory. After taking a photo of one type of item at a time, the user specifies the item, and the AI counts and

returns the number of items in the photo to both the database and the application UI. If incorrect, users can edit the number. The finalized number is then updated.

The “Wishlist” page serves for planning purposes. For supermarket managers, the list could be filled with items that they wish to import from their suppliers; for consumers, the list could be filled with items that they wish to buy for daily usage. An image of the item’s stock is presented when populating the list.

The “Me” page displays the user’s registration information and allows editing of it. The credential to log into the system consists of an email address and a password, which could be set after clicking the “Register” option.

The entire application is developed in Flutter using the programming language Dart and is available in both Android and iOS devices. The key data that allows the functionality of the application – number and photo of food items and user profile – are all stored in the cloud database Google Firebase. The computer vision technology of the camera function hosts on a server and has been custom trained on a manually created dataset in Roboflow with YOLOv5, a helpful object detection algorithm that uses the convolutional neural network. The algorithm divides a given image into grids, or kernels, performs pooling to extract dominant features, and calculates an expected probability for each component. It repeats the above procedure and returns the final object detection output after non-maximum suppression, which prevents false-positive identifications.

In short, the “Stores” page contains the core feature of accessing and updating the stock information of food items in grocery stores to the backend database and the application; the camera function supports the update feature with computer vision that counts items in a taken photo. The “Wishlist” page documents users’ personal demands, users can manage their accounts on the “Me” Page.

The flow of the “Stores” page follows the stores’ physical layout order, as shown in Figure 2. A list of stores are first presented. In each store, there are different aisles, and each aisle houses several shelves, on which users can find the specific items with their quantities. A photo of an item can be found by clicking on the item on the list. Users direct through these components by clicking on the desired entry on each subpage. Alternatively, users can directly find an item by typing the item name in the search bar and clicking on it from the dropdown menu. The items’ stock information in each store is retrieved for users to view.

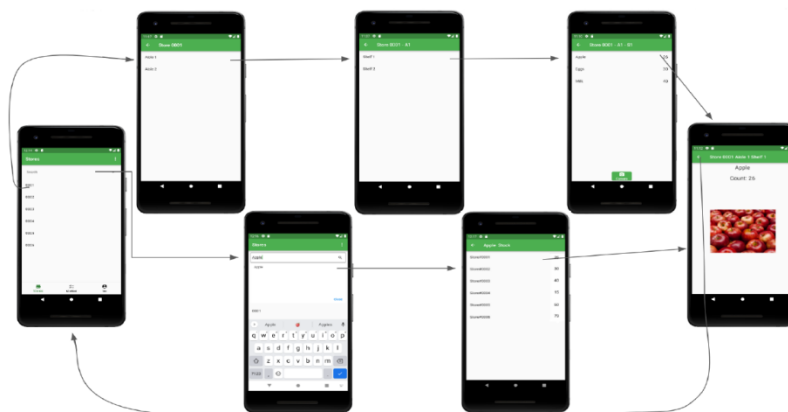


Figure 2. Stores Page UI

Clicking on the “Camera” button at the bottom of the “Items” subpage, users can activate the camera function. A count will be automatically returned once the user successfully takes a photo of items. To complete the update, users just have to confirm the food item and number of items the AI identified. After that, the data will be retrieved to Firebase and shown on the application UI.



Figure 3. Camera Function UI

Lastly, the “Wishlist” page mostly leverages the search bar functionality, as shown in Figure 4. Users can find the quantity of their desired item at different stores by typing them in the search bar. After selecting the item in a desired store, the item’s most recent photo will be displayed along with the quantity on a new page, where users can select the number of the item they want. The item can then be added to the wishlist and are still editable.

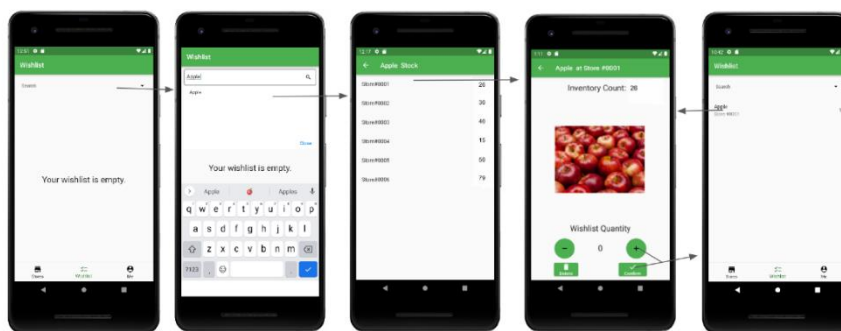


Figure 4. Wishlist Page UI

When a user edits a wishlist item, the stock of the item could be updated, and how many of the item the user wants is also constantly modified. As a result, these two pieces of information are asynchronous, requiring real-time updates, and thus, a StreamBuilder, shown in Figure 5, is implemented [12]. The StreamBuilder widget treats these real-time data as streams, and since these data are stored in Firebase, it takes a stream that constantly requests a snapshot of the user’s wishlist in the Firebase. The builder, which creates the UI, default returns “Loading” text when checking stream snapshot data. If the snapshot data is erroneous, an error message is returned. Otherwise, the builder does further checks. If the user already had a wishlist quantity for the item

in the designated store, the builder builds the page with the item's stock, which is retrieved in another StreamBuilder, and the user's wishlist quantity, which will be updated by constantly undergoing the same checks as the user edits it. If not, the builder defaults return the item's stock and 0 as the user's wishlist quantity for the user to edit.

```
StreamBuilder streamBuilderUserWishList(int inventoryCount) {
  return StreamBuilder(
    stream: FirebaseFirestore.Instance.doc("userShoppingList/${FirebaseAuth.Instance.currentUser!.uid}").snapshots(),
    builder: (context, AsyncSnapshot snapshot){
      if(snapshot.hasData){
        Map<String, dynamic> userWishListData = snapshot.data.data();
        Map<String, dynamic> userAllStoresWishList = userWishListData["storeWishList"];
        if(userAllStoresWishList.keys.contains(widget.storeID)){
          // Store in wishlist exists
          Map<String, dynamic> userStoreWishList = userAllStoresWishList[widget.storeID];
          if(userStoreWishList.keys.contains(widget.item)){
            // Item exists in wishlist
            int wishListCount = userStoreWishList[widget.item] as int;
            return body(inventoryCount, wishListCount);
          }
        }
        // return something here
        return body(inventoryCount, 0);
      } else if(snapshot.hasError){
        return const Text(...); // Text
      }
      return const Text(...); // Text
    }
  ); // StreamBuilder
}
```

Figure 5. Wishlist StreamBuilder Code Example

4. EXPERIMENT

Central to this application is the functionality of automatically recognizing item quantities for photos that users take. As a result, testing the model's efficacy in different types of photos, including varying scales and items, is essential.

4.1. Experiment 1: Evaluating the Accuracy of the Computer Vision Model

The first experiment was conducted to gauge the accuracy of the computer vision model. A high model accuracy ensures that users can share factual information efficiently with their local communities. In our application, the accuracy concerns identifying not only what an item is but also the quantity of an item. To test its item identification capability, the model counted items in a set of validation and testing images after training in more than 200 epochs. The four metrics – precision, mAP, recall, and GIOU – were then calculated for each epoch. Besides, a qualitative investigation of counting was done by manually checking the model's output counts for several shelf images.

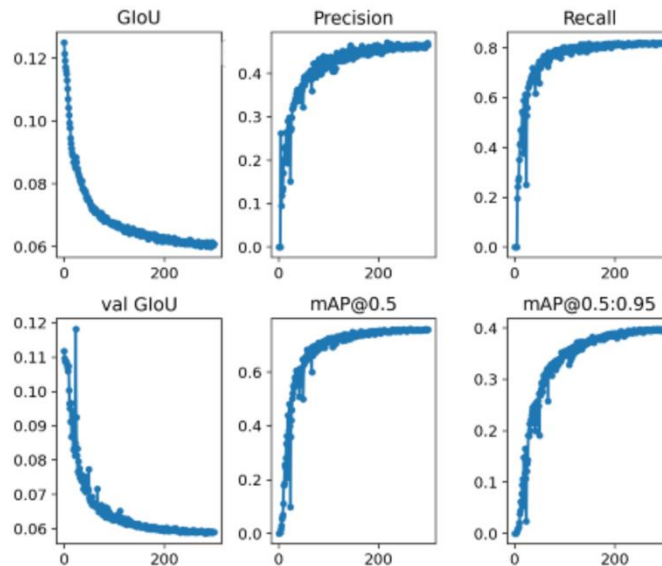


Figure 6. Model Results in Validation and Test sets

All the graphs in Figure 6 have the horizontal axis as the model's epochs and the vertical axis as the percentage. The shortcoming of the model is that its precision, which is around 50%, is slightly low, especially at a 95% Intersection Over Union (IOU) threshold, where the mAP maxed at 40%. Yet, at a 50% IOU, which indicates a decent bounding of items, the model's mAP is near 70%, a fairly high accuracy. Meanwhile, the model's recall is highly reliable; it is identified to operate on 80% of the grocery items it is trained with. Furthermore, the model localizes items well. As reflected by a 6% GIOU, which is a measure of bounding box loss, the model mostly makes correct bounds for items to identify.

In terms of counting, although the model did not count correctly in most angled photos, which are usually small-scale and of varying resolution as a result, these photos are not in consideration since the application is designed to have users take photos straight in front of the specific item of interest.



Figure 7. Example Test Images for Object Counting

Figure 7 contains two examples of large scale, well-angled, and high-resolution photos of supermarket stocks. For the photo of apples, which is an idealized, single-layered stock photo, the model correctly returns 24 apples. On the other hand, for the photo of drinks, which is a more realistic, depth-involved stock photo, the model returned 20 drinks. Even though there are clearly more drinks on the shelf, the model outputs a correct number for the first layer of the drinks, in which case the users can adjust the number.

With a high recall, the model eliminates users' need to type an item's name when updating inventory, except when the model mislabels an item, in which case the user can correct it in the review page. Similarly, the model's accurate counts in well-taken photos like ones shown above save users time with precise information. Yet, in case of miscounts due to negligence of shelf depth (illustrated in the drink example) or unclarity of a taken photo, which could be in practice, the application allows users to change the model output. Overall, the computer vision model offers helpful foundation data for users to work with, and the convenience ultimately translates to high productivity when users modify items' inventory status to be a reference for informed food purchase.

4.2. Experiment 2: Evaluating the Speed of the Computer Vision Model

The second experiment was conducted to measure the speed of the computer vision functionality. Being able to return an item count at a quick speed ensures efficiency and user experience with the application. To do this experiment, I implemented Python's `timeit` module, which measures the execution time of a code snippet. In an IDE, the model ran on 60 images of common grocery store items in different quantities, and the model's execution time on each image was measured. The average of all the measurements is calculated and used to represent the model's speed.

```
A total of 60 images were counted in 47.34 seconds.
count      60.000000
mean       0.788997
std        0.229070
min        0.615681
25%        0.690042
50%        0.736774
75%        0.779955
max        1.935442
dtype: float64
```

Figure 8. Timeit Console Output

As shown in the console output in Figure 8, the computer vision model finished counting all sixty images relatively quickly, averaging to less than 0.8 seconds per photo of items. Despite the fact that the longest time that the model has spent to count items in a photo is near 2 seconds, the model's counting speed is still considered to be highly consistent because of the small standard deviation, 0.23 seconds per photo of items, in counting times. Since the photos that the model is tested with are of various layouts, the model will most likely perform in a similar manner for all kinds of photos of items. Thus, the model is expected to generally finish counting a photo of food items at around 0.79 seconds.

Based on the result of this experiment, which indicates that users can receive a count of items immediately, it is reasonable to conclude that the application is easy to use and, therefore, acceptable among users. The model's capability to perform quickly fits people's fast-paced, modern lifestyle and facilitates people's effort in combating food waste.

5. RELATED WORK

Christensen, B. et al developed an application to limit food waste through charitable sales [13]. The application, Too Good to go, allows users to be informed of the food surplus in nearby grocery stores or restaurants, who provide the information on the platform. Users can then reserve the food at a discounted price for pickup at designated locations. This system effectively allows the public to make use of the potentially wasted food. Compared to this application, our

application augments the feature by specifically indicating what items the users can claim, as detected from pictures. This difference enables the users of our application to make purchasing decisions more easily.

Trax, a Singaporean company, has also developed an inventory tracking system using computer vision [14]. The cameras on shelves and ceilings hourly record the stock status, which are then analyzed by a cloud. After checking for the completeness of the photo, the computer vision system ratifies the photo quality and obtains a full picture of a shelf through panoramic stitching. It then detects the inventory's stock status, which consists of item number count, item placement, and pricing information. While this application's AI mechanism is much more robust and provides more retailing information, our application allows users to more easily retain the information they need. Users can look for the most recent photo of the stock of their desired items by selecting them in the app. This allows our application to be a handy tool for users to plan their shopping by determining where to buy an item.

Varghese, C. et al built a community between food suppliers and consumers during COVID-19 [15]. To alleviate the increasing food shortage during the pandemic, the application is a platform on which donors can post their donation information for people wanting food to pick up, and others' demand can be entered for donors to see. It fits into AI for Smart Living via Human Computer Interaction and ubiquitous computing. Our application shares the property of making use of otherwise wasted food and furthers it by easing the entire process. Suppliers do not have to specify where and when to pick up the products as the interface is set up by stores, and consumers are assumed to buy the products as soon as possible. The food quality is also guaranteed since the food comes directly from grocery stores instead of individual donors.

6. CONCLUSIONS

To combat food waste in supermarkets, this project proposes a digital, real-time supermarket inventory management system that uses computer vision. Both store managers and the public can inform others about where and how much of a food item there is by taking a photo of the items. The computer vision model, which is built in Python, counts the number of an item and returns the result to both the database Google Firebase and application UI, which is built using Flutter, to update the information. With a wishlist that documents users' personal demands, the application promotes smart buying decisions made based on trends in a user's wishlist.

Our solution relies on the data that users provide and share on the application, and the only way that users can do so is taking a photo and, if necessary, modifying the count that the built-in computer vision returns. Therefore, the computer vision model, a pivotal functionality, was experimented with, mainly through testing its accuracy and speed of counting items in photos, the two most tangible features. To test the accuracy, the model performed object detection on validation and test images and object counting on several picked images. The near 50% precision and 80% recall were accommodated by editable item names in the application. The model correctly returned single-layered counts for photos of shelf items. Meanwhile, the speed of the model is measured by its average time of counting the grocery items in sixty photos, which is 0.79 seconds per photo. Altogether, by promptly returning stock information that users can adjust and utilize, the computer vision model enhances the effectiveness and usability of the application, easing and encouraging the public's initiatives in informed food consumption.

Yet, the effectiveness of the solution could be further improved in several aspects. To start with, the computer vision model vocabulary is currently limited. While the model could explicitly identify several commonly seen items, including apples and bottled goods, it cannot perform an item count on others that the model does not recognize. The model either ignores the items or

miscounts with irrelevant items. In addition, the accuracy of the model could definitely undergo more optimizations. Currently, the model can correctly count items that are photographed well-angled. However, it is unrealistic that users can take photos upright every time as that would require too much effort.

To address these limitations, expanding the scope of training data will improve the model's capacity. The model can then identify and perform counting on the objects. In addition, experimenting with parameters of the model can help find the most accurate specs. Along the way, providing the model with more data can prevent the model from obtaining skewed results.

REFERENCES

- [1] O'Shea, Keiron, and Ryan Nash. "An introduction to convolutional neural networks." arXiv preprint arXiv:1511.08458 (2015).
- [2] Zeide, Anna. "Grocery garbage: food waste and the rise of supermarkets in the mid-twentieth century United States." *History of Retailing and Consumption* 5.1 (2019): 71-86.
- [3] Eriksson, Mattias, and Johanna Spångberg. "Carbon footprint and energy use of food waste management options for fresh fruit and vegetables from supermarkets." *Waste Management* 60 (2017): 786-799.
- [4] Aschemann-Witzel, Jessica, et al. "Consumer-related food waste: Causes and potential for action." *Sustainability* 7.6 (2015): 6457-6477.
- [5] Curry, Nathan, and Pragasen Pillay. "Biogas prediction and design of a food waste to energy system for the urban environment." *Renewable Energy* 41 (2012): 200-209.
- [6] Poyatos-Racionero, Elisa, et al. "Recent advances on intelligent packaging as tools to reduce food waste." *Journal of cleaner production* 172 (2018): 3398-3409.
- [7] Kor, Yasemin Y., Jaideep Prabhu, and Mark Esposito. "How large food retailers can help solve the food waste crisis." *Harvard Business Review* 19 (2017).
- [8] Chaudhary, Sanjay, and P. K. Suri. "Agri-tech: experiential learning from the Agri-tech growth leaders." *Technology Analysis & Strategic Management* (2022): 1-14.
- [9] Skaggs, Richard L., et al. "Waste-to-Energy biofuel production potential for selected feedstocks in the conterminous United States." *Renewable and Sustainable Energy Reviews* 82 (2018): 2640-2651.
- [10] Ehrlen, Johan. "Why do plants produce surplus flowers? A reserve-ovary model." *The American Naturalist* 138.4 (1991): 918-933.
- [11] Consumer Reports. "What to do when there are too many product choices on the store shelves?." *Consumer Reports* (2014).
- [12] Islam, Md Olioul. "A high embedding capacity image steganography using stream builder and parity checker." 2012 15th International conference on computer and information technology (ICCIT). IEEE, 2012.
- [13] Baglioni, Simone, Benedetta De Pieri, and Tatiana Tallarico. "Surplus food recovery and food aid: The pivotal role of non-profit organisations. Insights from Italy and Germany." *VOLUNTAS: International Journal of Voluntary and Nonprofit Organizations* 28.5 (2017): 2032-2052.
- [14] Coifman, Benjamin, et al. "A real-time computer vision system for vehicle tracking and traffic surveillance." *Transportation Research Part C: Emerging Technologies* 6.4 (1998): 271-288.
- [15] Varghese, Christina, Drashti Pathak, and Aparna S. Varde. "SeVa: a food donation app for smart living." 2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC). IEEE, 2021.

AN INTELLIGENT COMMUNITY-DRIVEN MOBILE APPLICATION TO AUTOMATE THE CLASSIFICATION OF PLANTS USING ARTIFICIAL INTELLIGENCE AND COMPUTER VISION

Yifei Tong¹ and Yu Sun²

¹Trinity Grammar School, 119 Prospect Rd, Summer Hill NSW 2130, Australia

²California State Polytechnic University,
Pomona, CA, 91768, Irvine, CA 92620, USA

ABSTRACT

How can the efficiency of volunteers be improved in performing bushcare in the limited amount of time able to be spent caring for each location every month [1]?

Bushcare is a volunteer activity with a high difficulty curve for volunteers just starting out as the crucial skill of distinguishing the native plants from the harmful invasive species only comes with experience and memorization [2]. The lack of ability to distinguish targeted plants will greatly reduce the efficiency of the volunteers as they work through the limited amount of time they have at each location each month while also discouraging newly joined volunteers from continuing this activity.

To assist newly joined volunteers, the majority of each would likely be from a younger demographic with a digital app that could help the user distinguish the species of plant, making it easier for them to start familiarizing themselves with both the native and invasive species in their area [3]. The user could simply have to take a picture of the plant they wish to identify and the software would use its image recognition algorithm trained with a database of different species of plants to identify the type of plant and whether it needs to be removed. At the same time, more experienced volunteers could continue to use this app, identifying errors in the app's identification to make it more reliable.

KEYWORDS

Flutter, Machine learning, Firebase, Image recognition.

1. INTRODUCTION

Bushcare is very important to the Australian ecosystem because its unique fauna and flora are being out-competed by invasive species from other ecosystems [4]. If left unchecked, the invasive species will spread and take on resources that the native species need to survive, causing them to die off and reduce the biodiversity of the entire ecosystem [5]. Bushcare works to remove the invasive plants from an area while caring for the native plant to try to restore the state of the ecosystem to its healthy original state [6]. Currently the issue is that there is a limited number of volunteers and an even more limited number of staff who can supervise the volunteers as they

work, meaning that each area will only get a few hours for volunteers to carry out their job each month.

The limited amount of time means that it is crucial for the team of volunteers to work efficiently in the time they are given. The majority of the volunteers are of the older demographic, who are very experienced in doing their job each time they work in the area. But there is a lack of new recruits as it is hard and time-consuming for new volunteers to remember that they need to be removed from the ecosystem and what to care for, causing there to be a lack of younger generation volunteers [7]. The process of learning what plants of native and what is harmful is frustrating as there is no clear rule to distinguish the two groups while the fear of accidentally removing native plants and causing more harm further discourages them.

To solve this issue, a mobile app could be used to identify the type of plants for the user while they are still inexperienced, which greatly helps new volunteers when they join, improving their experience and efficiency. This app would work by using a database of different plant species to train an algorithm to recognize the plant through image recognition, therefore allowing users to simply take an image of the plant with their phone and the app will identify the species of plant for them. This will help them work as they familiarize themselves with the types of plants in the ecosystem.

There are already many plant recognition software on the market also using image recognition to identify the types of plants captured by the camera of the phone and allowing the user to gain some information about the plant in front of them [8]. However, that software is all focused on gardening rather than bushcare, meaning there is a great desperation in need of new volunteers from what those apps can provide. Those apps focus on the needs for gardening, therefore the types of plants that these apps need to recognize are different from the types of plant that needed to be recognized for bushcare as the types of the plant being deliberately planted is different from the types of plant found in the wild. Those apps will have a wider range of plants worldwide to satisfy the needs of their targeted customers as gardening involves many plants from around the world that thrive in many different climates while only needing to focus on being able to recognize plants encountered when gardening. But bushcare has a more specific need in the types of plants that need to be recognized, only focusing on a specific area with a set climate but needing all plants in this area to be able to be recognized, therefore those apps might not be as accurate in recognizing the types of plants encountered in bushcare.

A bigger problem is that after the plant is recognized by existing plant recognition apps the information is not helpful in the situation of bushcare. When those gardening apps recognize the plant captured, it gives the species of the plant and some information about the plant and how to care for the plant, which is mostly redundant to new volunteers to bushcare [9]. Rather than the species of the plant, more useful information would be if the plant is invasive and needs to be removed and information that would be helpful is how to properly remove a plant in areas such as whether the roots need to be removed or if the seeds need to be bagged.

In this paper, the process of creating the app that would help improve the efficiency of bushcare volunteers is very similar to the other commercial apps used to identify plants for gardening. Our goal is to develop an app that would be easy to use for new volunteers doing Bushcare to help them to work more efficiently during their inexperienced phase when they are just starting out. Our method is inspired by many other image recognition algorithms that have gained popularity in recent years.

First, the basic structure of the app was made on android studio using the flutter software development kit. The app contains a camera page where an image of plants was taken for the

software to identify the plant, and the main page was made for users to navigate to different pages like the calendar for upcoming bushcare schedules and other special events or to navigate to the personal profile page.

We also used Firebase to allow users to make personal accounts with their email. This would allow the account to be tracked when sending feedback to errors in the algorithm and potentially be used to track which bush care site a user goes to for automatic reminders or to share the picture of the different species of plants captured.

Then lastly, the image recognition algorithm is put in to process the image captured by the camera. A preexisting database for different species of plants is used to train the algorithm which could be added to as an error in the images captured to could help the algorithm to become more accurate.

The use-ability of the app has been tested to ensure that the app would be effective in solving the problem with new volunteers. The app has to be convenient for users to make sure that it would be used easily for anyone to assist them when they don't have the necessary knowledge for bushcare. The convenience of the app provided with other features is also important in making the process of going to bush care easier which would help to attract younger volunteers into joining this activity. We tested the working of the different accounts and the working of the application with Firebase, making sure that it is storing the different accounts and communicating with the application properly so that when the algorithm runs into an error, the user can report it and Firebase would be able to record from which user the error came from [10]. We also tested to make sure the camera would work on any phone as different phones have different dimensions or sizes of images for the camera. To ensure that all phone cameras would work, the application crops the image to a set size the algorithm could process therefore uniforming the input. As we have run into some problems setting up the code for the camera we haven't had time to flash out the image recognition algorithm yet. We have run some tests with images saved on the computer and run it through the algorithm and the result of the image recognition is not accurate enough to be used effectively for its purpose.

The following part of this report is organized into the following sections: Section 2 contains challenges encountered during the process of designing and testing the product; Section 3 describes the solution used to solve the challenges listed in section 2 in order to finish the app; Section 4 presents the experiments we did and the relevant details; Section 5 presents related works. And finally, Section 6 includes concluding remarks as well as listing the further work to be done in this area.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Limit Amount of Time

It is currently hard for new volunteers to work efficiently in the limited amount of time spent caring for each area's plant life. There is no rule classifying what plant is native and which is invasive so the skill to distinguish what needs to be removed from the local environment can only be achieved by slowly familiarizing ourselves with the plants in the area and remembering the different types of plants present. This is a very slow process that could take up to a few years to fully familiarize oneself with the plant lives of casual volunteers. This means new volunteers are usually just told a few common plants that need to be removed each time and only focus on the

species they are shown. This means that many other invasive species are left unchecked and would just fill the space cleared out by the volunteers and reduce the efficiency of the work.

2.2. Lack of New Volunteers of a Younger Age

The demographic of bush care is mostly made up of volunteers of older age as there is a lack of new volunteers of a younger age. The difficult learning curve at the start discourages many new volunteers from continuing causing there to be a lack of interest from the younger generation as there is not a common sight for young people to be part-taking in it. Apart from the difficulty in familiarizing themselves with the plant and the frustratingly long amount of time, volunteers might also be afraid that they might be doing more damage than good. In areas where there is a more diverse range of plants, new volunteers can't be told just a few types of plants to focus on and often just feel like they are removing random plants without knowing whether they are removing harmful plants or native plants and causing more damage than good, which further disparages them from continuing.

2.3. Recognize All The Plants

In order to help new volunteers recognize the types of plants the algorithm needs to be able to recognize all the plants that could be found in the area. For the algorithm to do this, there needs to be a database of all the plants present in the area to train the algorithm to recognize them. But as we are not a commercial organization with a large number of resources, we do not have the resources to gather a big enough database of plants ourselves, and doing it manually would be way too time-consuming.

3. SOLUTION

This application is an image recognition software that would serve as a personal assistant to new volunteers, helping them classify the different species of plants while they work to develop their own knowledge about the plants. Apart from the main function of identifying plants from the camera page of the application, there are also a few other parts that are made to improve the experience of the user and to motivate more younger volunteers to stay in bushcare after they start. The application has a login and sign-up page to allow users to create their personal account of the app with their email which allows more personalization for each user to make their experience more convenient. The Application stores the different accounts by communicating with Firebase, powered by google. The application also has a calendar page that is going to show all the bush care times and locations as well as any special event that will be happening. There is also a function to share the plants a user has captured with other users of the app. These functions are made to try to make the bushcare experience more suited for the younger generation to encourage a younger demographic to join bushcare volunteering. The application is made with android studio using the google open-source development kit, flutter, and implementing dart to create sections of the application.

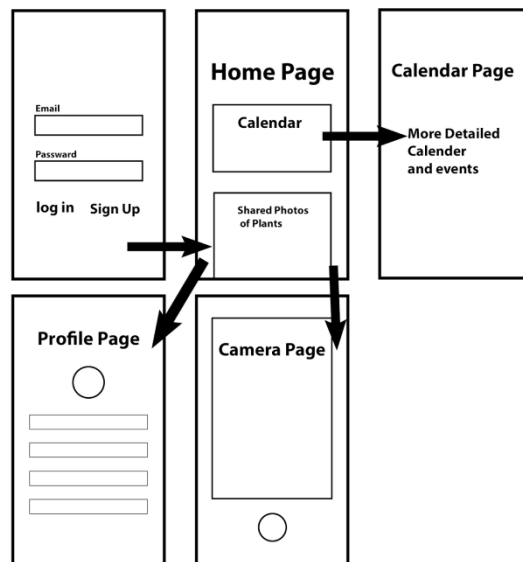


Figure 1. Overview of the solution

The log-in page works by communicating with the firebase platform with all the user information stored on it. As shown in the code snippet below, the application is linked to a firebase platform where all the information about already existing users is stored. The application checks if the email interns already exist on the firebase, and then it checks if the password is correct. If both are correct then it lets the user enter their own personalized account of the application where they can continue using the app.

```

if(_formKey.currentState!.validate()){

  try{

    await FirebaseAuth.instance.signInWithEmailAndPassword(email: email, password: password);
    showSnackBar("Login Successful ...");
    submitLock = false;

    Navigator.pushAndRemoveUntil(

      context,
      MaterialPageRoute(builder: (context) => HomePage()),
      (route) => false,
    );

  }
  on FirebaseAuthException catch(e){

    if (e.code == "user-not-found"){
      showSnackBar("Email is Incorrect");
    } else if (e.code == "wrong-password"){
      showSnackBar("Password is Incorrect.");
    }
    else{
      showSnackBar("Unknown Error.");
    }
  }
}
submitLock = false;

```

Figure 2. Screenshot of code 1

Similarly, the sign up page also communicates with the Firebase platform but as it is creating a new user and storing the information the sign up page uploads the information about the new user to Firebase rather than checking the information stored on Firebase. When the new user enters the email and password into the text boxes on the sign up page, the application communicates to

Firestore and checks if the email is already in use or valid before checking if the password meets the requirement. If everything is up to standard it stores that information as a new user on Firestore to allow them to log in the future and let the user into the home page.

```
Container(  
  child: TextFormField(  
    controller: confirmPasswordTextFieldController,  
    validator: validateConfirmPassword,  
    decoration: const InputDecoration(  
      labelText: "Confirm Password",  
      contentPadding: EdgeInsets.only(left: 20.0),  
    ),  
    obscureText: true,  
  ),  
  
  color: const Color.fromRGBO(200, 200, 200, 100),  
  margin: const EdgeInsets.only(left: 40.0, right: 40, bottom: 20,),  
),  
  
ElevatedButton(  
  onPressed: onSignupButtonPressed,  
  child: const Text('Get Started'),  
),  
]
```

Figure 3. Screenshot of code 2

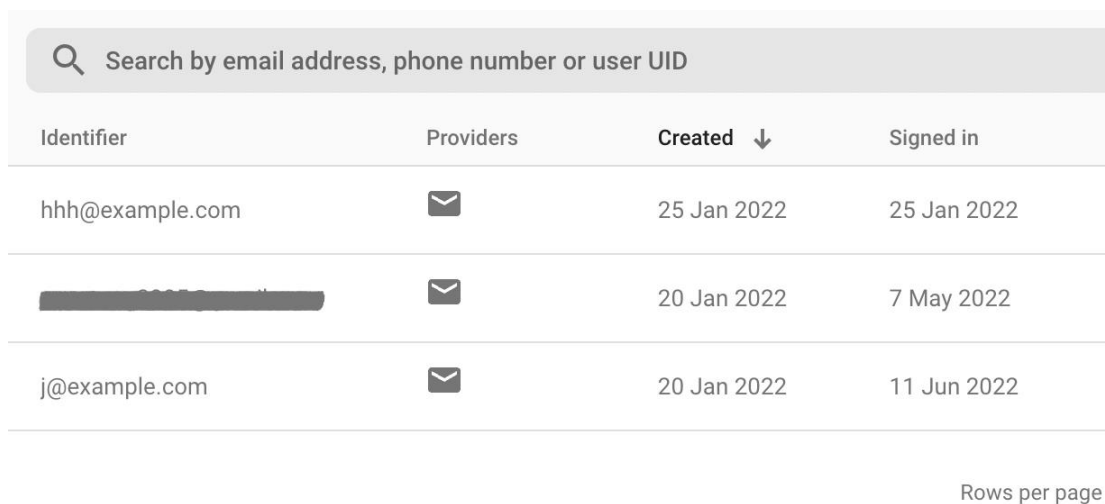
The camera page uses a dart in Flutter to capture the image with the camera of the phone. The dart package called image picker was first imported into the application at the top of the code to enable the application to capture images. When the image is captured it is first cropped to the right size for the application to process while the time when the image is captured is also recorded. The image is given three values: description, species, and whether it is invasive or not, and given to the algorithm to process.

4. EXPERIMENT

4.1. Experiment 1

To ensure that the users can log in to an account for better personalization of the app and so that we can track user information when information is sent in from the users we had to test whether the Firebase platform is storing user data and communication with the application properly. This would be important for the image recognition algorithm too as it would be the users report any incorrect output of species of plant and therefore increase the database we can train the algorithm with. To test this, We had several accounts to test that Firebase is able to store the accounts effectively and check the information stored through firebase directly. Then we checked the application would let users in smoothly while also preventing duplicate emails from signing up so causing problems with the system.

The application was able to successfully send the new accounts created to Firestore and access them when the information is required for logging in users. The wait time for users to load in isn't significant and fairly easy to use as it is very similar to any other login or the sign-up page for other programs. The application also was able to successfully check with Firestore to stop any email that already exists to create second accounts by signing up again and checking the strength of the password to ensure that it is strong enough.



The screenshot shows a table with a search bar at the top. The search bar contains the text "Search by email address, phone number or user UID". Below the search bar is a table with four columns: "Identifier", "Providers", "Created", and "Signed in". The table contains three rows of data. The first row has the identifier "hhh@example.com", one provider (represented by an envelope icon), a creation date of "25 Jan 2022", and a sign-in date of "25 Jan 2022". The second row has a redacted identifier, one provider, a creation date of "20 Jan 2022", and a sign-in date of "7 May 2022". The third row has the identifier "j@example.com", one provider, a creation date of "20 Jan 2022", and a sign-in date of "11 Jun 2022". At the bottom right of the table, there is a link that says "Rows per page".

Identifier	Providers	Created ↓	Signed in
hhh@example.com	✉	25 Jan 2022	25 Jan 2022
[REDACTED]	✉	20 Jan 2022	7 May 2022
j@example.com	✉	20 Jan 2022	11 Jun 2022

Figure 4. Screenshot of information

We tested the image capture function to ensure that both the camera of the phone could be accessed by the application to capture the image of the plants to be processed by the algorithm and also that the image could be sent to the Firebase with the correct information like time taken and its location. The ability to capture images is crucial to the application as it is the input that would enable the rest of its function to be carried out. We needed to check that the application could upload the images taken to the Firebase platform where its content was checked straight through Firebase. The application needed other data such as the time the image was taken, the location it was taken at and the user id that sent the image to Firebase while also leaving space for the algorithm to determine whether the image contained an invasive plant and the species of the plant.

We captured a large number of images with computers as they emulated the mobile app of the program and uploaded them to the Firebase platform to be stored and eventually communicated the algorithm to be processed. The information uploaded to Firebase was checked directly through Firebase storage where all the information was stored. All images were able to successfully have the time taken, the location it was taken from, and the user id of the account that uploaded the image to Firebase while leaving space for the algorithm to eventually fill in the species of the plant and whether it is invasive. In this process, we also deduced that we needed to crop the images to a uniform size to enable this process to function.

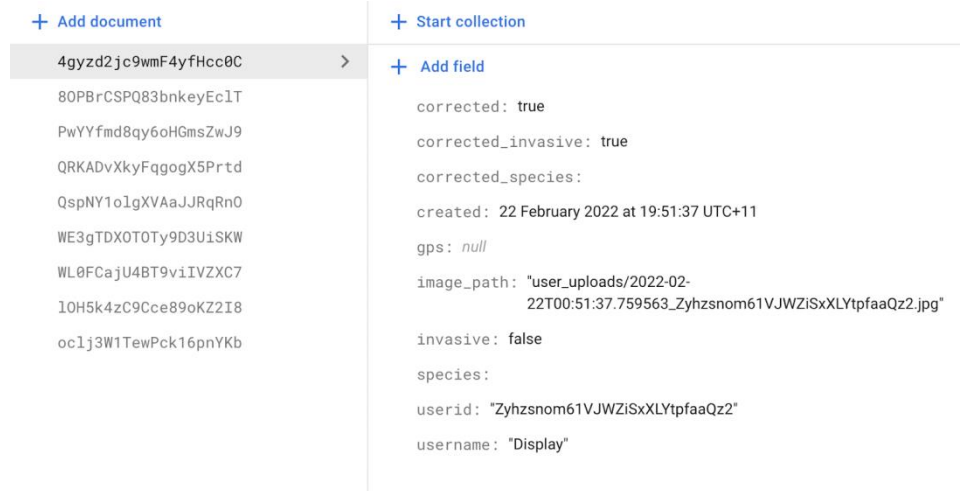


Figure 5. Screenshot of add document

As we have run into difficulties making sure that the capers of the application would function properly for the application to work and proper images to be uploaded to the algorithm to be processed and so we haven't had enough time to properly set up the image processing algorithm. We have tested the algorithm by manually uploading images to it and the result is not yet as accurate as needed for the application.

The result of the experiments conducted was mostly satisfactory as it confirms the result that we wanted to see. The experiment proves that the app is functioning properly, doing its intended job and therefore able to connect to other parts of the application and function smoothly to achieve its intended purpose. The app is convenient enough for users and therefore will improve the experience of the new volunteers when starting out bushcare. The algorithm is still not ready for use but it is still being perfected and it will still be worked on as the app is officially put out for people to use as users will upload more images to the database of the algorithm to develop and become more accurate.

5. RELATED WORK

This paper aims to produce an algorithm for plant recognition for the use of agriculture, aiming to improve the sustainability of producing crops by reducing laborious tasks from humans and to make a more accurate judgement [11]. This research is different from this project as it also focuses on the health of the plant and with a further step of determining how healthy the plant is and outputting the best environment and nutrient level for the plants to develop. The research focuses more on plants on the macro level while this project focuses more on individual plants. Furthermore, this research is lightly technical, providing a lot of information for the user while our application tries to simplify information for the convenience of everyday users.

This paper aims to use image recognition to detect plant disease to assist humans in caring for those plants [12]. This paper uses MATLAB as its primary processing tool and uses the linear progression model to detect whether the plant is diseased. This research is highly technical, using the coloration of the plant's leaf, isolating parts of the leaf with unnatural coloration, and determining if the plant is diseased. This research differs from our project as it focuses more on whether the plant is diseased rather than the species of the plant. Also, the algorithm focuses more on processing images in a more controlled environment while our application has to be able to process images taken on the field.

This paper researches image recognition algorithms to reduce human mistakes and improve efficiency in plant identification, disease detection, and diversity preservation [13]. It focuses on comparing the effectiveness of different methods of image recognition in the application of identifying the types of a plant from its leaves, listing out a great variety of methods to compare. This research has similar purposes to our application, used to improve efficiency and improve plant diversity in the environment but goes more in depth into the faults and merits of many different methods.

6. CONCLUSIONS

This project tries to solve the problem of it being very hard for new volunteers to pick up bushcare as an activity as there is a steep learning curve to distinguishing the different plants that need to be removed from the native plants that are being protected. This application aims to help new volunteers to classify the different plants with image recognition to improve their efficiency when first starting bushcare while improving their experience to keep more new volunteers from leaving from being discouraged by the steep learning curve. This app allows the user to take images of the plants they see and determine the species of the plant and whether it is an invasive plant with the image recognition algorithm [14]. The app allows users to sign in to their personal account and therefore allows them to share images of the plants they have captured or report any errors there are in the application or the results of the algorithm. The application also shows the calendar of all the volunteering/events for bushcare for the convenience of the user. All the extra functions serve to increase the experience of the users as an attempt to increase the number of volunteers of a younger demographic to join bushcare [15].

Currently, the app is still very limited and basic. The accuracy of the algorithm could be improved as there is only a very limited amount of images in the database and the algorithm is not optimized. But this could be improved as people start using the app, taking pictures of more plants to be added to the database and improving the algorithm by correcting any errors in results. The image-sharing function is also not very practical at the moment as it does a clear function, later could be a system of challenges asking the user to capture a number of plants to motivate them and give the image-sharing function a purpose.

Some future work includes adding more descriptions to the species of plant, namely how to effectively remove the different invasive species, including whether the roots need to be removed and seeds are a concern. Further work needs to be done to improve the image recognition algorithm by adding more images to the database and correcting any mistakes made by the algorithm to make it more accurate.

REFERENCES

- [1] González, Eduardo, and Antonio Alvarez. "From efficiency measurement to efficiency improvement: the choice of a relevant benchmark." *European Journal of Operational Research* 133.3 (2001): 512-520.
- [2] Shelef, Oren, Peter J. Weisberg, and Frederick D. Provenza. "The value of native plants and local production in an era of global agriculture." *Frontiers in plant science* 8 (2017): 2069.
- [3] García, Eugene E., Bryant T. Jensen, and Kent P. Scribner. "The demographic imperative." *Educational Leadership* 66.7 (2009): 8-13.
- [4] Dwyer, John M., Rod Fensham, and Yvonne M. Buckley. "Restoration thinning accelerates structural development and carbon sequestration in an endangered Australian ecosystem." *Journal of Applied Ecology* 47.3 (2010): 681-691.
- [5] Peltzer, Duane A., et al. "Understanding ecosystem retrogression." *Ecological Monographs* 80.4 (2010): 509-529.

- [6] Reidy, Margaret, Winkie Chevalier, and Tein McDonald. "Lane Cove National Park Bushcare volunteers: taking stock, 10 years on." *Ecological Management & Restoration* 6.2 (2005): 94-104.
- [7] Rostoft, Siri, and Tanya Petka Wildes. "Time to stop saying geriatric assessment is too time-consuming." *Journal of Clinical Oncology* (2017): 2871-2874.
- [8] Sharma, Sapna, et al. "A review of plant recognition methods and algorithms." *International Journal of Innovative Research in Advanced Engineering* 2.6 (2015): 111-116.
- [9] Abd Rahim, N., F. A. Zaki, and A. Noor. "Smart app for gardening monitoring system using iot technology." *system* 29.04 (2020): 7375-7384.
- [10] Khawas, Chunnu, and Pritam Shah. "Application of firebase in android app development-a study." *International Journal of Computer Applications* 179.46 (2018): 49-53.
- [11] Abdullahi, Halimatu Sadiyah, R. Sheriff, and Fatima Mahieddine. "Convolution neural network in precision agriculture for plant image recognition and classification." *2017 Seventh International Conference on Innovative Computing Technology (INTECH)*. Vol. 10. Ieee, 2017.
- [12] Sun, Guiling, Xinglong Jia, and Tianyu Geng. "Plant diseases recognition based on image processing technology." *Journal of Electrical and Computer Engineering* 2018 (2018).
- [13] Huixian, Jiang. "The analysis of plants image recognition based on deep learning and artificial neural network." *IEEE Access* 8 (2020): 68828-68841.
- [14] Richmond, Nicola J., Peter Willett, and Robert D. Clark. "Alignment of three-dimensional molecules using an image recognition algorithm." *Journal of Molecular Graphics and Modelling* 23.2 (2004): 199-209.
- [15] Pollak, Robert A., and Terence J. Wales. "Demographic variables in demand analysis." *Econometrica: Journal of the Econometric Society* (1981): 1533-1551.

CLASSIFICATION OF DEPRESSION USING TEMPORAL TEXT ANALYSIS IN SOCIAL NETWORK MESSAGES

Gabriel Melo, KaykeBonafé and Guilherme Wachs-Lopes

Department of Computer Science,
University Center of FEI, São Paulo, Brazil

ABSTRACT

In recent years, depression has gained increasing attention. As with other disorders, early detection of depression is an essential area of study, since severe depression can result in suicide. Thus, this study develops, implements, and analyzes a computational model based on natural language processing to identify the depression tendencies of Twitter users over time based on their tweets. Consequently, an F-measure of 83.58 % was achieved by analyzing both the textual content and the emotion of the papers. With these data, it is possible to determine whether constant fluctuation of emotions or the message in the text is a more accurate indicator of depression.

KEYWORDS

Depression, Natural Language Processing, Machine Learning.

1. INTRODUCTION

According to the WHO [1] more than 294 million people worldwide suffer from depression. Works, such as [2] and [3], states that early diagnosis is a crucial aspect in determining the efficacy of a treatment, therefore reducing the probability of the disease escalating to severe depression or even suicide.

In this sense, social networks can be a great field of study for the early detection of depressive behaviors. An example of this is the study of [4], which proposes to infer whether there is a relationship between the use of multiple social networks and the development of conditions such as anxiety and depression. The results of this study showed that there is a correlation between these factors. Another is the study of [5], which is a model of text classification that aims at early detection of depression in social media streams. The results of this paper show that the model proposed by the authors outperforms the most used machine learning models, like SVM.

Following this point, it is possible to create a model that can show the information related to the feelings contained in the text, in this case, the variation of feelings, so that it is possible to infer how much influence this variation affects the classification.

Therefore, the goal of this work is to propose, implement, and evaluate a computational model based on natural language processing to classify depressive tendencies of Twitter users through their posts over time. The second goal is to discuss how sentiment analysis and the textual content itself influence the classification score.

2. METHODOLOGY

The proposed methodology pipeline starts with dataset formatting and data pre-processing. Then, the pipeline proceeds to the vectorization step and sentiment analysis. Finally, the last step is the classification output. This pipeline can be seen in Figure 1.

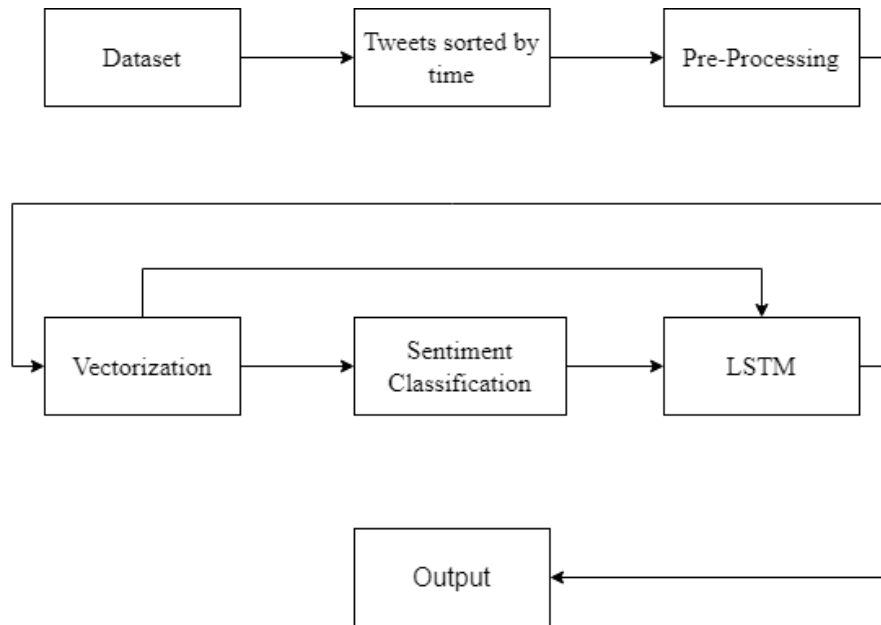


Figure 1. Methodology Pipeline

2.1. Dataset and Data Pre-Processing

The dataset used in this study was developed by [6]. It consists of textual data collected from Twitter from 2009 to 2016 and divided into 3 different categories: **D1**, which consists of users who explicitly have depression. This classification is done if the user has a post with a pattern similar to “(I’m/ I was/ I am/ I’ve been) diagnosed depression”; **D2** consists of users who have not been classified as depressed. This happens if the term ‘depress’ was not found in any publication; **D3** consists of unclassified data. This occurs if the term ‘depress’ is found, but it is not identified in the pattern used in D1.

Users with more than 15,000 followers were excluded from this dataset because they may flag accounts of organizations, notable persons, or bots. Tweets that describe a narrative or a narrative have also been deleted, since they may indicate someone else’s narrative or a fake story. Additionally, it was eliminated outdated information, such as “I was diagnosed with depression when I was 10.”

The data is in JSON (JavaScript Object Notation) format and divided into multiple folders, where each tweet can be viewed individually or the entire user timeline, as well as data associated to their accounts such as the user’s username and profile description (for security reasons, no personal data was revealed). The most significant data in each tweet structure are its creation date and textual content. The texts are not arranged in chronological order.

Considering the pre-processing step, the first goal is to increase the quality of the data obtained from the dataset, where some techniques can be applied and combined in order to make the texts

denser (word uniformity and with greater cooccurrence) so that the created model can process the data more accurately. In order to decrease textual complexity, we removed special characters, emojis, and nonlatin alphabet characters, like Japanese Kanji.

After the tokenization process, it is verified that some tokens have no semantic value, being useful only for the formalism and rules of the language, not adding any relevant information. These tokens are known as stop words and are part of the so-called stop list, which is a list of predefined words. These stop-lists are removed in order to improve data quality.

2.2. Vectorization and Feature Extraction

In the vectorization stage, the pre-processed texts are transformed into valued vectors using Doc2Vec. In addition, we also use sentiment analysis to study how this feature can contribute to the final classification.

During the vectorization step, text that has already been pre-processed is converted to value vectors by using the Doc2Vec algorithm. In addition to that, we also make use of sentiment analysis in order to investigate the potential role that this attribute plays in the overall categorization.

An SVM classifier is used to perform sentiment analysis. In the first step of the vectorization process, each document is processed through a word counter (BoW) and a standard scaler. We carried out an SVD reduction in conformity with the results shown in Table 1 to accomplish the dual goals of decreasing the matrix's dimensions and raising its density (Section 4.1). Following this step, the produced vectors are fed into a support vector machine (SVM) in order to classify the polarity of sentiment for the sentences that were collected from Twitter.

Since the dataset employed in this research lacks a sentiment ground truth, a second dataset is required. This dataset was created by [7], and all tweets were classified as having either a positive or negative sentiment polarity using many techniques, including SVM.

2.3. Training and Classification

During the training and classification stage, the collected preprocessed data is structured as input parameters for the SVM and LSTM.

SVM is used for sentiment analysis. However, as described in the vectorization section, this training cannot be done on the depression dataset, since there is no supervised information on feelings. As a result, the model is trained using the dataset from [7]. The texts from the Twitter dataset will be inferred from the vectorized documents using BoW and their dimensions will be decreased using SVD, as outlined in Section 2.2.

After entering the vectorized BoW data into SVM, the output will be a vector that represents how much positive feeling is present in the analyzed text. The trained algorithm is used in the dataset developed by [6] that classifies tweets as depressive and non-depressive, as presented above.

At this point, there are two vector representations for each tweet. The first is obtained by Doc2Vec and the second is the sentiment polarity classified by SVM. The final classification of the data occurs through the LSTM network that is fed with the vectorized sentences of the dataset (tweets). Each of these sentences enters the timestep dimension of the LSTM.

The result of this process is given by a numerical value between 0 and 1, which indicates the level of depression in the sentence, with 1 being the presence of depression and 0 the absence. This process is illustrated in Figure 2.

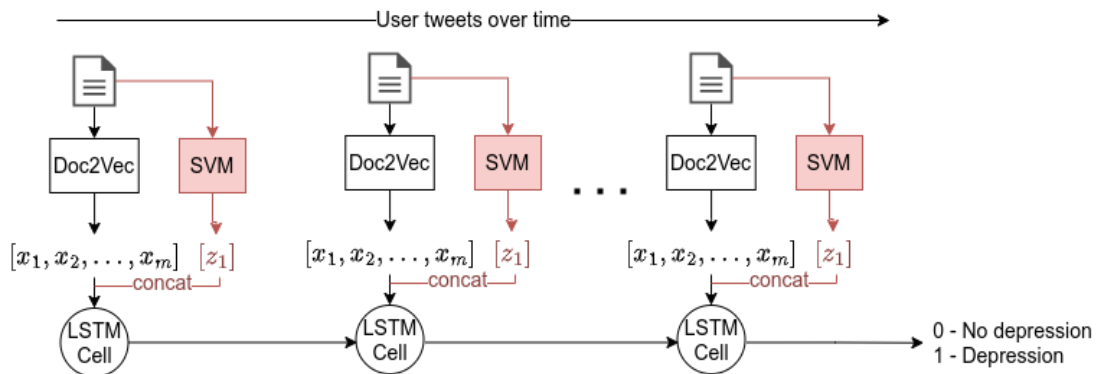


Figure 2. LSTM usage

3. EXPERIMENTS

3.1. Sentiment Classification

The SVM is trained independently using the dataset that has sentiment classifications of texts, described in Section 2.3, to be used in the classification of the LSTM. To carry out the training, 70% of the dataset is used as training data and 30% as input data for classification and subsequently tool verification. The input is the text of the post in Twitter and the output is the positive or negative value detected in the tweet. To check the score level, four metrics are used: precision, recall, F-Measure, and accuracy.

3.2. Vectorization

Since Doc2Vec has some tuning parameters, such as window size and embeddings vector dimension, this experiment consists of analysing how they contribute to describing document contents as vectors. For instance, if the embedding size is too high, this can lead to high memory usage, and LSTM network may receive inputs with higher dimensions. However, if the embedding size is too small, Doc2Vec could not have enough vector space to describe all the documents from the dataset.

Therefore, the goal of this experiment is to find a balance between the size of the embeddings and the discrimination of the documents. The first step is to choose n randomly tweets from the dataset D and store in a list L . Then, Doc2Vec is trained with different values of window size and embedding size. For each training, we choose a random generated database, $R = L \cup \{x \mid x \in s(D, z)\}$ where s is a function that returns z sample tweets.

Finally, for each document $d \in L$ we infer document embedding from Doc2Vec and compare it for each document in R dataset using cosine similarity. This process generates a list of tweets sorted by similarity with respect to document d . When the most similar document in R dataset is the document d it means that the Doc2Vec could generate discriminant vectors. However, the farther d is from first position, the lower is the discrimination between documents. Figure 3 illustrates the whole process.

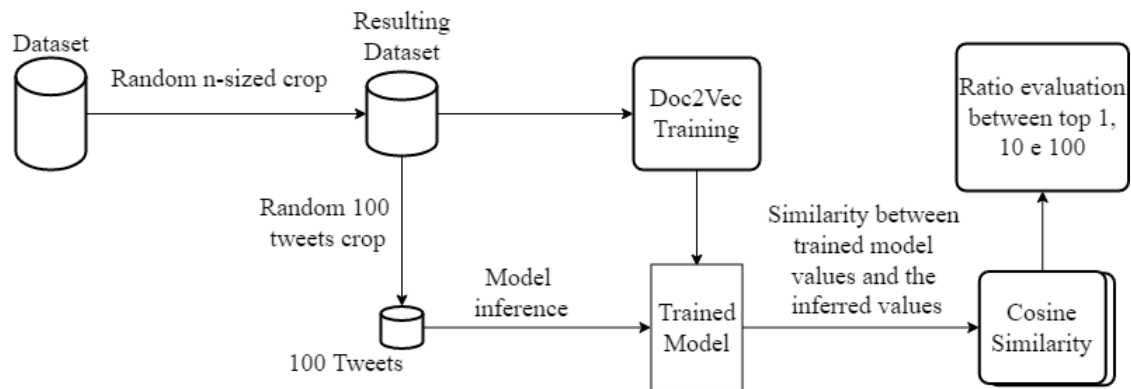


Figure 3. Visual representation of vectorization experiment

The results are computed counting how many documents were found into Top 1, 10 and 100 positions of distance sorted list.

3.3. Doc2Vec and SVM

As described in Section 2.3 and illustrated in Figure 2, there are two mechanisms to extract features from tweets: Doc2Vec and Sentiment Analysis. Doc2Vec is used for textual analysis in order to vectorize the tweets in the text dataset in order to comply with the input format expected by LSTM network. For sentiment analysis, SVM is trained to evaluate sentiment from texts analyzed in the dataset.

As one of the goals of this work is to study how sentiment analysis can influence depression classification, we propose an experiment that compares two classifiers: with both information (Doc2Vec + SVM); and with only information on document content (Doc2Vec).

After performing both training sessions, it is possible to measure whether the detection of depression is more related to the constant variation of emotions or the final message conveyed by the text.

3.4. Stemming Validation

In this step, an experiment is carried out in which the sentiment dataset are submitted to a classification evaluation through the SVM to validate the impact of using stemming on the result.

4. RESULTS

This paper achieved three main results: sentiment classification with SVM, vectorization with Doc2Vec, and depressive tendencies classification. The following sections discuss the results.

4.1. Sentiment Classification

The method used for the sentiment classification was the SVM. Several executions of the training have been performed by varying some parameters. Beyond that, two different datasets were used, with and without stemming. Another parameter on the execution is the words sparse matrix dimensionality, generated by the BoW. The starting dimensions were 25, going up to 300. When validating with the F-Score measure, the best results got a precision of 70.11%, a recall of 78.35%, an accuracy of 72.48, and an F-Score of 74%. The results are shown in Table 1.

Table 1. Sentiment classification results

Dataset	BoW Dimensions	Precision	Recall	Accuracy	F-Measure
With Stemming	25	60.96%	68.83%	62.24%	64.65%
	50	63.09%	73.82%	65.46%	68.04%
	75	65.60%	75.72%	68.01%	70.30%
	100	66.95%	75.65%	69.08%	71.04%
	200	68.75%	78.11%	71.32%	73.14%
	300	70.11%	78.35%	72.48%	74.00%
Without Stemming	25	57.18%	72.39%	59.04%	63.89%
	50	62.59%	72.35%	64.51%	67.11%
	75	63.06%	74.80%	65.63%	68.43%
	100	65.81%	76.64%	68.41%	70.82%
	200	68.01%	76.87%	70.33%	72.17%
	300	69.60%	77.77%	71.83%	73.46%

According to Table 1, we can observe that the bigger the dimensionality, the better the result. The conclusion here is that the higher dimensions the bigger is the vector space used to describe a word. Therefore, more details of the document are captured. The outcome would have been better if there had been additional dimensionality added, however due to time and hardware constraints, only 300 dimensions were used.

4.2. Vectorization

On the text vectorization, the method used was Doc2Vec. Seven runs were conducted during this stage, all with a dataset percentage, starting with 1000 tweets going up until 3,192,403. For this experiment, the used dimensionality was 300. When tested, the described model reached an average of 95.91% of similarity between the generated vectors on the training and the vectors generated for the validation.

The results of the experiments are shown in Figure 4, where the x axes represent the amount of data used, the left y axes represent the percentage of times that the model's first result is the same as the used validation vector and the second y axes represent the training time. On the first 6 experiments, all searched tweets were between the first 10 to 100 positions, only on the last experiment that this number goes to 91 on the first 10 positions and 98 on the first 100. In the first case, this happens because of the difference between the vectorial representation of the words. On the last two, this decrease is justified by the amount of data used.

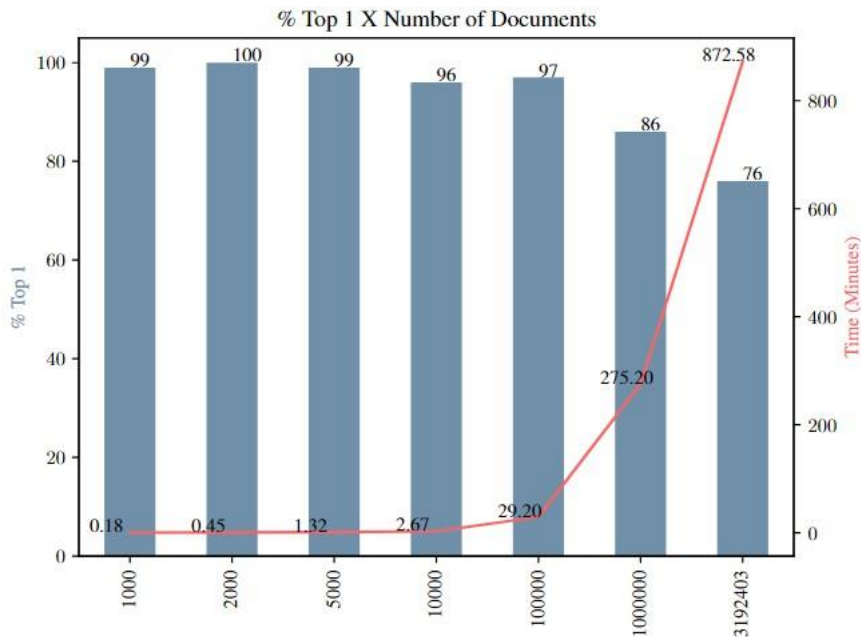


Figure 4. Number of documents versus the percentage of times that the first result of the model was identical to the result of the inference

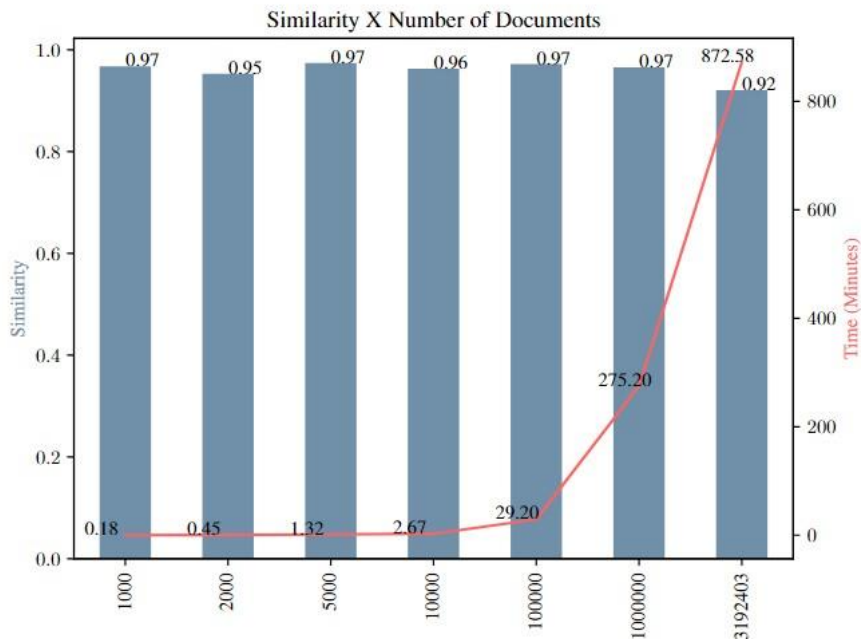


Figure 5. Similarity versus Number of Documents

Last, the same observation can be made if we analyze the average similarity between the tweet used for the validation and the existing in the model, which decreases slightly, as can be seen in Figure 5. In this Figure, the x axes represent the amount of data used, the first y axes represent the similarity and the second one represents the time used on the training.

4.3. Depressive Tendencies Classification

For this purpose, the LSTM was executed with two different strategies: using textual data from Doc2Vec (textual); and using both sentiment analysis and combined textual data (concatenated), this being a transfer learning approach. Two different executions of the experiments occurred, each with random data from the dataset, both on the training and on the validation. The hidden state size was one and two times the input size.

It is possible to see the variation of F-Score in the three strategies on image, as well as the hidden state sizes. It is noticeable that increasing the hidden size increases the F-Score, in most cases.

The best F-Score was obtained using the concatenated data as well as the hidden size being two times the input size. As shown in Table 2, the best result obtained a precision of 82.87%, recall of 84.31%, accuracy of 83.46%, and an F score of 83.58%. The threshold used for the metrics was found by maximizing the Geometric Mean using a ROC curve, where the best AUC was 0.90. The curve can be seen in Figure 6.

Initially, the entire dataset was being used, having more than 3 million documents. However, when validation was being done, the results had around 40% F-Score and 8% of accuracy, which was evidence of an unbalanced dataset. The initial proportion was around 9 nondepressive users to 1 depressive user. After the data were balanced, the dataset had 50% depressive and non-depressive users, totaling around 900 thousand documents. The final results can be seen in Table 2.

Table 2. LSTM Results

	Dataset	hidden size	Precision	Recall	Accuracy	F-Measure	Threshold
Execution 1	Textual	300	80.87%	82.02%	81.34%	81.44%	54.01%
		600	82.48%	84.15%	83.15%	83.31%	52.41%
	Textual + Sentiment	301	82.25%	81.93%	82.14%	82.09%	92.67%
		602	82.41%	84.04%	83.06%	83.22%	42.47%
Execution 2	Textual	300	81.62%	82.75%	82.12%	82.19%	62.35%
		600	82.65%	84.21%	83.30%	83.42%	72.41%
	Textual + Sentiment	301	81.89%	82.13%	81.99%	82.01%	80.44%
		602	82.87%	84.31%	83.46%	83.58%	60.93%
Execution 3	Textual	300	82.28%	82.63%	82.43%	82.45%	76.66%
		600	83.70%	84.36%	83.98%	84.03%	64.59%
	Textual + Sentiment	301	80.21%	82.47%	81.43%	81.64%	35.14%
		602	83.15%	84.96%	83.84%	84.04%	28.91%
Execution 4	Textual	300	82.08%	83.37%	82.56%	82.72%	60.15%
		600	83.69%	84.50%	84.00%	84.10%	74.49%

	Textual + Sentiment	301	82.27%	82.81%	82.46%	82.54%	59.68%
		602	83.66%	83.95%	83.73%	83.81%	61.08%
Execution 5	Textual	300	82.39%	83.52%	82.80%	82.95%	81.41%
		600	83.68%	84.20%	83.85%	83.94%	75.71%
	Textual + Sentiment	301	82.59%	82.83%	82.65%	82.71%	49.76%
		602	83.10%	84.56%	83.71%	83.83%	49.77%

When comparing the F1 scores of the two techniques using the Student T Test, we can see that there is no significant difference between the classification test that uses only vectorized textual data and the test that uses vectorized textual data and sentiment information.

The test results show a p-value of 65.37% for the LSTM with hidden size of 300 (for the textual data) and 301 (for the textual with sentiment analysis data). Furthermore, the p-value of 77.27% for the LSTM with hidden size of 600 (for textual data) and 602 (for the textual with sentiment analysis data). These test results mean a non-trustable confidence interval, confirming that the sentiment analysis does not make any significant improvement in the overall classification.

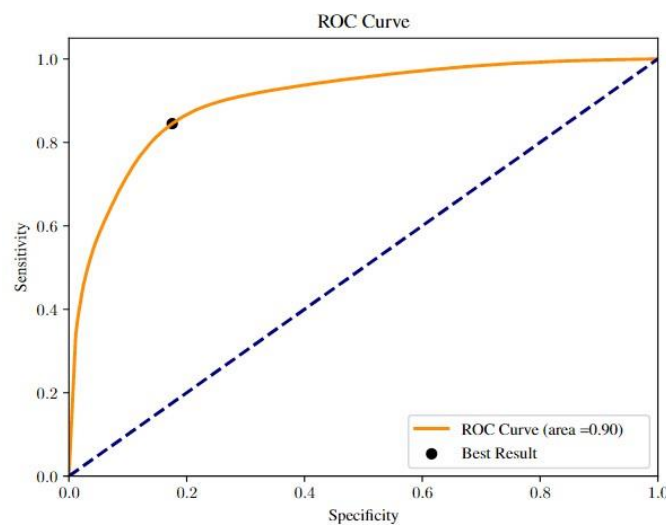


Figure 6. ROC Curve of the best result

5. CONCLUSIONS

This paper proposed a computational model based on natural language processing to classify depressive tendencies in tweets. For this goal, two different approaches were used: classification through text analysis; and classification using both described methods.

The results show that depressive tendencies can be detected using textual content and textual content combined with sentiment analysis. Furthermore, another finding was that textual models were more relevant to the classification than to the sentiment analysis. This indicates that the existing sentiment on the text is not a piece of discriminant information for classification.

Remarkably, both the textual data and both textual and sentiment together got similar results, making the use of sentiment classification unnecessary, when considering the computational cost.

As a contribution to this project, the proposed method got an F-Score of 83.58% using only textual information, in comparison, the state-of-the-art related to depression detection has an F-Score of 97% on [8], research that used not only textual content but images as well, which is not used in this paper. The model as well as the code can be found on Github, available at: <https://github.com/gabrielomelo/tcc-lstm>.

In future works, it is suggested to increase the density number of the data to improve the model precision as well as the sentiment analysis data precision as a way to aggregate the most accurate pieces of information to the concatenated model. Another suggestion in the sentiment analysis model could be the change from linear to the RBF Kernel in order to increase the classification accuracy.

In addition to this point, there is a possibility of doing a study on how the textual serialized information by time contributed to improving the classification quality from the proposed model. Thus, it is suggested to create a simpler model that considers only one tweet from the user. The main hypothesis here is that the textual serialized information by time contributes to the improvement of the F score.

To conclude, for more evidence about the model efficiency, it is expected that this line of study and investigate the model quality will be investigated through other statistical tests such as K-fold.

REFERENCES

- [1] WORLD HEALTH ORGANIZATION (2021) Depression. [Online]. Available: <https://www.who.int/news-room/fact-sheets/detail/depression>
- [2] W. Yu-Tseng, H. Hen-Hsen and H. Chen, (2018) A Neural Network Approach to Early Risk Detection of Depression and Anorexia on Social Media Text. Avignon, France: CLEF 2018.
- [3] Paul, S.; Jandhyala, S. K.; Basu, T. (2018) Early detection of signs of anorexia and depression over social media using effective machine learning frameworks. West Bengal, India: CLEF 2018.
- [4] B. A. Primack, S. Ariel, C. G. Escobar-Viera et al., (2017) Use of multiple social media platforms and symptoms of depression and anxiety: A nationally representative study among U.S. young adults. Pittsburgh, United States of America: Computers in Human Behaviour.
- [5] Burdisso, S.G., Errecalde, M.L., & Montes-y-Gómez, M. (2019). A Text Classification Framework for Simple and Effective Early Depression Detection Over Social Media Streams. San Luís: Argentina: Expert Syst. Appl. 2019.
- [6] G. Shen and J. Jia and L. Nie et al., (2017) Depression Detection via Harvesting Social Media: A Multimodal Dictionary Learning Solution. Hefei, China: IJCAI-17.
- [7] A. Go and R. Bhayani and I. Huang, (2009) Twitter sentiment classification using distant supervision. California, United States: CS224N.
- [8] R. Kumar and S. K. Nagar and A. Shrivastava, (2020) Depression Detection Using Stacked Autoencoder from Facial Features and NLP. Bhopal, India: SMART MOVES JOURNAL.

AUTHORS

Gabriel Melo has a Computer Science bachelor degree (University Center of FEI, 2021), is interested in the following topics: complex networks, neural networks, natural language



processing and cyber security. Currently works as an information security analyst developing cyber intelligence tools (Itaú Unibanco S.A.).

Kayke Bonafé has a degree in Computer Science (University Center of FEI, 2021), is interested in the following topics: artificial intelligence, neural networks, natural language processing and machine learning. Currently works as a data scientist developing reports and models (Monett Conteúdo Digital LTDA.).



Prof. Dr. Guilherme Wachs has a Computer Science bachelor degree, master in Artificial Intelligence and PhD in Signal Processing Area. Currently, is a researcher and professor of A.I. group of University Center of FEI, and interested in NLP, IoT and Computer Vision.



© 2022 By AIRCC Publishing Corporation. This article is published under the Creative Commons Attribution (CC BY) license.

LEARNING CHESS WITH LANGUAGE MODELS AND TRANSFORMERS

Michael DeLeo and Erhan Guven

Whiting School of Engineering, Johns Hopkins University, Baltimore, USA

ABSTRACT

Representing a board game and its positions by text-based notation enables the possibility of NLP applications. Language models, can help gain insight into a variety of interesting problems such as unsupervised learning rules of a game, detecting player behavior patterns, player attribution, and ultimately learning the game to beat state of the art. In this study, we applied BERT models, first to the simple Nim game to analyze its performance in the presence of noise in a setup of a few-shot learning architecture. We analyzed the model performance via three virtual players, namely Nim Guru, Random player, and Q-learner. In the second part, we applied the game learning language model to the chess game, and a large set of grandmaster games with exhaustive encyclopedia openings. Finally, we have shown that model practically learns the rules of the chess game and can survive games against Stockfish at a category-A rating level.

KEYWORDS

Natural Language Processing, Chess, BERT, Sequence Learning.

1. INTRODUCTION

One of the oldest board games, chess is also one of the most researched computational problems in artificial intelligence. The number of combinational positions is around 10^{50} according to [1] and this makes the problem ultimately very challenging for even today's computational resources. State of the art solution to learning the chess game by a computer has two parts, generating valid board positions and evaluating their advantage to win the game. Like an optimization problem, generating possible and promising positions is analogous to a feasible optimization surface and is built by a tree data structure representing each position reached from a previous position. Evaluating a position involves the chess game knowledge, such as how a piece moves, their values, the position of the king, opponent piece positions, and existence of a combination of moves that can lead to a forced mate, are among numerous calculations to find the winning move or a combination of moves.

Stockfish [2], one of the best chess engines today, uses the minimax search with alpha-beta pruning improvement [3] by avoiding variations that will never be reached in optimal play. The computationally infeasible search is avoided where it is possible to infer the outcome of the game, such as in a deterministic mating attack discovered by the search tree. IBM Deep Blue chess computer [4] used dedicated processors conducting tree searches faster on hardware to beat the world champion Garry Kasparov who is considered as one of the best human chess players ever. Alpha Zero [5] uses plain randomly played games without any chess knowledge but learns the moves from the game. A general-purpose reinforcement learning algorithm, and a general-purpose tree search algorithm are used to conduct combination of moves. The deep learning

engine learns the game as much as the hand-crafted knowledge injected in the Stockfish evaluation engine.

In this study, we evaluated and analyzed a second approach to learn chess, specifically using BERT transformer to extract the language model of chess by observing the moves between players. Language models extracted by the state-of-the-art models such as GPT-3 [6] are considered as few-shot learners. Among many other statistical information, a language model can involve rare patterns represented by only a few counts in a histogram which can be extracted by BERT transformers.

Our study evaluates the BERT language model starting from a simple game, then towards increasing the complexity of the game as in chess. First, we analyzed the application of a language model to a simpler game Nim due to its smaller size allowing a complete analysis. We conducted experiments of Nim games between a Guru player which knows winning solutions to Nim as a rule-based approach versus a random player blindly making valid Nim moves and evaluate the performance of the language model. Next, we analyzed the model between a random, a Guru, and a Q-learner player with a controlled number of random games. And finally, we applied the model to the chess game by grandmaster games covering all possible openings from the chess opening encyclopedia [7].

A literature survey shows only a few very recent papers [8, 9] have applied NLP methods to chess but none of them used board/move text based on a grammar pattern to encode the game. As a novelty, the method in this paper encodes the game positions and moves in a specific text pattern based on Forsyth-Edwards Notation [10] which is possibly easier to be learned than a full game in Portable Game Notation format [10]. Starting from the opening position, PGN conveys a position virtually between the moves without explicitly encoding. In a board game each position and move pair can be thought of a sentence passed to the other party. Thus, these sentences are learned by the language model, and they are somewhat order independent. The following sections will describe the NLP method and analyze its performance towards learning board games that use text representation of each position and move.

1.1. Chess State of the Art

Stockfish is under the GPL license, open source, and still one of the best chess programs making it a suitable candidate to teach a natural language model. There are two main generations of Stockfish which are Classical and NNUE. The latter is stronger of the two, and for the purposes of this research is what will be focused on. In this version, Stockfish relies on a neural network for evaluation rather than its previous method of relying on a search tree. NNUE stands for Efficiently Updateable Neural-Network [11]. This network is a lightweight fully connected neural network that gets marginally updated depending on the state space of the board, which is an optimization technique to improve its performance.

Alpha Zero is a deep reinforcement learning algorithm with a Monte Carlo Tree Search (MCTS) algorithm as its tree search algorithm [8]. MCTS is a probabilistic algorithm that runs Monte Carlo simulations of the current state space to find different scenarios [3]. An example of the MCTS being used for a game of tic tac toe is shown in Figure 1. Notice the tree branches off for various game choices. This is a critical component of the Alpha Zero model in that it allows it to project/simulate potential future moves and consequences. MCTS was chosen by the deep mind team as opposed to using an Alpha Beta search tree algorithm because it was more lightweight.

Alpha Zero is famous for beating Stockfish in chess with 155 wins out 1000 games. Stockfish won 6 games [8]. There is some debate however as to if more hardware would have helped Stockfish. Nonetheless, the main advantage of Alpha Zero to Stockfish is that it is a deep learning

model which can play itself millions of times over to discover how to best play chess. One of the impracticalities of it is that it is not open source however, and proprietary to DeepMind.

1.2. Chess Text Notation

In this study the chess notation is based on coordinate algebraic notation. This notation is based on the chess axes where {a, b, ..., h} is the x axis, and {1, 2, ..., 8} is the y axis and represents two coordinates {(x!, y!,), (x", y")}. The first coordinate set represents the initial position, and the second set represents the position the piece moves to [10]. This notation is chosen to represent move states rather than other notations is because of its uniformity and how many tokens it would take to represent a full game position.

The Forsyth-Edwards Notation (FEN) is a notation that described a particular board state in one line of text with only ASCII characters [10]. A FEN sequence can completely describe any chess game state while considering any special moves. We use this notation to describe our board state in our experiments. An example of the FEN sequence is shown in Figure 1, this record represents the initial chess state.

```
rnbqkbnr/pppppppp/8/8/8/PPPPPPPP/RNBQKBNR w KQkq - 0 1
```

Figure 1. Example FEN Sequence

1. Piece placement: each piece is represented by a letter r, n, b, q, etc. and the case indicates the player where uppercase is white, and lowercase is black. The rank of each section of piece is described by Standard Algebraic Notation (SAN) [10], and this describes the positions of those pieces.
2. Active color: represented by a “w” meaning white’s turn is next, and “b” meaning black’s turn is next.
3. Castling Availability: There are five tokens to represent castling availability, “-“ no one can castle, “K” white can castle king side, “Q” white can castle queen side, “k” black can castle king side, and “q” black can castle queen side.
4. En Passant.
5. Half Move Clock: Starts at zero and represents the number of moves since the last capture or pawn advance.
6. Full Move Clock: Starts at 1, increments after black’s move [10].

FEN is particularly useful because it provides a complete stateless representation of the game state in a single character sequence. This is possible because chess is a game where there are no unknowns, and everything represented visually is everything there is to the game space. These are the reasons for why we chose the game of chess and chose these notations for our experimentations.

1.3. Nim Game

Nim is a game of strategy between two players in which players remove items from three piles on each turn. Every turn the player must remove at least one item from exactly one of the piles. There are two different versions of the game goal: the player who clears the last pile wins or the player who has to take the last piece loses the game.

1.4. BERT Model

The BERT model (Bidirectional Encodings for Representations of Transformers) is a language model which is designed to pretrain on bidirectional representations on unlabeled text by jointly conditioning on context from both the right and left sides [12]

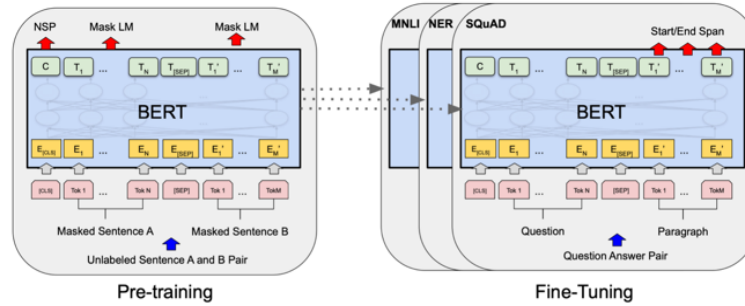


Figure 2. BERT Architecture [12]

The Bidirectional Encoder Representations from Transformers (BERT) model is a supervised model, that achieved state of the art on Q&A tasks before GPT. It's a lightweight, deep learning model that is trained to learn bidirectional representations of context in unlabeled text. The general architecture can be seen in Figure 2, and it should be noted that it is like GPT-1 in terms of its architecture and size [13].

2. METHODOLOGY

The objective of our study is to train a transformer model on text sequence datasets in such a way that it can learn to accurately play and understand the games. We apply the BERT model to both Nim and Chess. In this section we will lay our procedure for procuring the data for these experiments as well as our methodology for training the transformer.

2.1. Nim Data Collection

Our Nim experiment consisted of using three agents: a random player, a guru player, and a Q-learner. Each experiment is initialized to three piles, and ten items per pile (i.e. [10,10,10]). Equal number of games are played between each player taking equal number of times as the first player. The version of the Nim game in this study favors the first player. As a result, when two Guru players play against each other the first player wins roughly 95% of the time.

A variety of games positions are created by randomly creating the pile positions starting from [1,1,1] to [10,10,10] so that occasionally the random player can also win against a Guru just because the starting position was a lucky one. These played and stored games are used to train a Q-learner and a BERT learner later on in the experimentation.

Additionally, when Nim games are collected the pile positions of the game are randomized throughout each recorded game. This was to increase the level of difficulty for a model to learn from the sequences.

2.2. Chess Data Collection

The chess experiment uses Stockfish 14, and python 3.9. The Stockfish engine is configured to use NNUE, with one thread, default depth, and one for the value of MultiPV. The max depth that can be set is 20, however that slows the experiment down too much and so a value of 1 is chosen for the sake of getting a large dataset. Additionally, Stockfish is set to an ELO rating of 3900. To gather a large amount of data, one million games of chess are played. A timeout for moves is set for 200 to discourage runaway stalemate games. The data collection activity takes about 4-6 days.

```
rnbqkbnr/pppppppp/8/8/8/PPPPPPPP/RNBQKBNR w KQkq - 0 1 [MOVESEP] f2f4
```

Figure 3. Example Chess Sequence

1. The basic routine of the program is to initialize a fresh game with the Stockfish engine, and the stated configurations
2. Select the best move from Stockfish and submit for each player until the game is over
3. Record the moves (FEN and algebraic coordinate) as they are selected and store
4. At six moves end the game, delimit each set of moves with the next line tokens and perform post-processing

An example of the data returned by a game of chess is shown in Figure 3 where each line consists of a FEN position, the player, and the next move chosen. This example is the opening chess board, followed by a separator token and a move for F2 to F4.

2.3. Pretraining BERT

For each experiment, once the data is generated, the BERT Word Piece Tokenizer is trained on the entire set of data such that it can get a full scope of the sequences. Since information is encoded into words and letters being capitalized, the tokenizer must accommodate for this. Therefore, the vocabulary includes capitalization.

We utilize the datasets hugging face library to load all our datasets and delimitate by the end of line token. Those datasets are tokenized and collated in parallel then split into training and testing sets with a 20% split. For the training procedure, we use the hugging face trainer with a 15% MLM (masked language modelling, refer to [12]) probability.

To provide inference with the model, a hugging face pipeline is used where the state sequence is provided and a [MASK] token is placed at the end where the move token would be. For example, a sequence for Nim would be a10/b10/c10 G – [MASK], where the pipeline would fill in the MASK token for the move.

2.4. Initial Analysis of BERT Model on Nim

As a few shot learner and as an unsupervised learner [6, 14] BERT language model can extract patterns that are expressed only a few times and in midst of very high noise. The following experiment used the language model trained using the games between Guru and random players. A number of games are played between the Guru which is a rule based player, and a random player generating random but valid moves. Since the Nim game outcome heavily depends on the first player move (like Tic Tac Toe), an equal number of games are played by swapping the first player. Each game start from a non- zero number of pieces in three piles, so that a Guru player

can lose a game against a random player since it might be given a losing position in the first place. The number of possible positions or the feature space size is 113 (equals 1331) for three piles and 10 pieces to start the game. Theoretically one needs at least this many positions to fill up the feature space for a Guru to make a move so the game data would have at least one sample of every board position and winning (or the "right") move by Guru. Note that for the Q-learner, such a learning approach takes close to 300k games (against a random player) to be able to be on par with a Guru player [15].

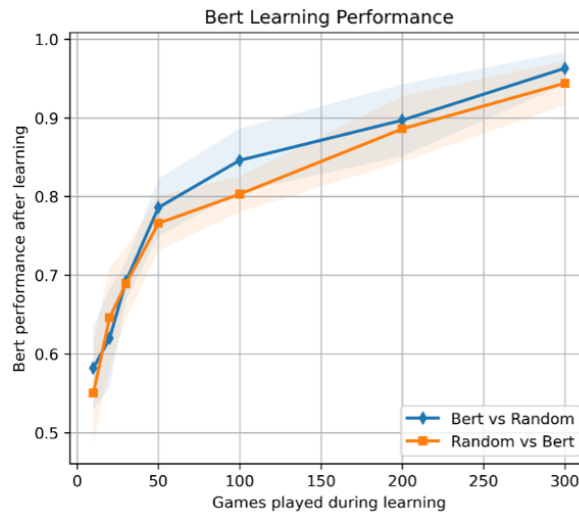


Figure 4. BERT Learning Nim Game from Random Player

The experiment trains a transformer by a certain number of games defined as a match, played between the Guru and the random player. Every training starts from reset and the trained model is used by the BERT player to make a move. An average game against Guru by the random player takes empirically ~ 6.5 many moves. Thus, 10- game match of Guru-random and an additional 10- game match of random-Guru would provide around 130 unique moves possibly. This space covers only 10% of the feature space (game board) presenting an almost impossible learning problem. Against all odds, as shown by the experimentation, the BERT player learns every move that Guru makes and plays accordingly when faced a random player. The range of number of games (match size) is changed from 10 to 300 where the latter makes the BERT player an excellent challenger for the random player. This is the direct result of the few-shot learning method presented by the transformers.

3. EXPERIMENTAL RESULTS

Following the methodology stated of performing collections of data into a standard dataset format, data was collected for both Nim and chess experiments. Some of the characteristics for these datasets are listed in Table 1 - Dataset Metrics.

Table 1. Dataset Metrics

Methodology Metric	Nim	Chess
Number of Games	30,000	30,000
Total Unique Game States	7973	2575
Total Unique Moves	30	892
Dataset Length	423,480	2657
Average Sequence Length	15.09	74.28
Dataset Size (MB)	8.1	167.9

Notably with our method of data collection and data format, choosing longer sequences to represent a system will cause the dataset's memory size to grow by an order of M where M is the current dataset length. This caused issues when we initially tried to generate a chess dataset that contained one million games and created an 8 GB text corpus. This is also one of the reasons we added the Nim experiment, to test our hypothesis on a smaller scoped dataset.

The hardware used for the experimentation is an RTX-A6000, an i9 processor, and 128 GB of RAM.

3.1. Results Nim

Following the data collection, two BERT word piece tokenizers were trained on each variant of the Nim datasets: X/W and Player ID. The vocabularies for each tokenizer were relatively small and are shown in Figure 5 and Figure 6 respectively. The vocabulary size maxed out around 60 tokens for each tokenizer because the game of Nim is not that complicated in sequence form.

```
{'a3': 43, 'a1': 33, '8': 15, 'b8': 59, 'b9': 61, 'b1': 35, '4': 11, '##9': 23, 'c8': 58, '6': 17, 'c4': 46, '##5': 31, '##6': 28, 'b3': 44, '##2': 32, 'c5': 48, 'c3': 42, '9': 16, '2': 9, 'c1': 34, 'a': 20, '6': 13, 'a10': 65, 'c0': 38, 'a8': 57, 'R': 19, '7': 14, 'b4': 45, 'c6': 51, 'a7': 56, 'c7': 54, '##3': 27, '[MAS K]': 4, 'c9': 60, '[PAD]': 0, 'a5': 50, '##1': 24, '[CLS]': 2, '-': 5, 'b5': 49, 'b10': 64, '##7': 26, 'a9': 62, '1': 8, '0': 7, 'c': 22, 'a0': 36, '3': 10, 'a6 ': 53, '5': 12, '0': 18, 'a4': 47, 'c10': 63, 'b0': 37, '##0': 25, 'a2': 41, '/': 6, '##4': 29, '[UNK]': 1, 'b6': 52, 'c2': 40, 'b2': 39, 'b7': 55, '##8': 30, 'b ': 21, '[SEP]': 3}
```

Figure 5. BERT Tokenizer Tokens for Nim with Player IDs' G, Q and R.

```
{'a0': 35, '/': 6, '9': 16, 'b1': 34, 'c1': 33, 'a6': 52, '7': 14, 'c9': 59, '##1': 24, '[UNK]': 1, '[CLS]': 2, 'c3': 42, 'b9': 61, 'a9': 60, 'c8': 56, '[MASK]': 4, 'b': 20, '##7': 30, 'c2': 39, 'a2': 40, 'b3': 41, '##8': 22, '1': 8, 'c0': 37, 'a1': 32, 'c4': 44, '-': 5, '##5': 26, 'b8': 57, '2': 9, 'W': 17, '5': 12, 'a10 ': 63, 'c7': 54, 'b0': 36, 'c10': 64, '6': 13, '##6': 27, '##0': 25, '[PAD]': 0, 'c': 21, '##4': 31, '0': 7, '4': 11, 'b5': 48, '3': 10, 'a4': 46, 'a3': 43, '[SE P]': 3, 'b2': 38, 'X': 18, '##2': 23, 'a': 19, 'b6': 50, 'a7': 55, '8': 15, 'b10': 62, 'c6': 51, '##3': 29, 'a8': 58, 'b4': 45, '##9': 28, 'a5': 49, 'b7': 53, 'c 5': 47}
```

Figure 6. BERT Tokenizer Tokens for Nim with Win States

We can verify the tokenizers captured the game state tokens and move tokens by inspecting their vocabulary. Since for Nim, the game state is being represented by a letter a, b, or c and a quantity we can see that those tokens do exist in the vocabularies.

Recall that two datasets were generated with partitions to designate artificial noise, the first had a special indicator for which agent made this move and the other had an indicator to as if this move won the game. We trained a fresh BERT model on each partition of each dataset and put each model into a roster where each agent played every single other agent, the results are below. The evaluation was performed with 1000 games of every permutation of every agent for each level of randomness for a total of 5000 games. The total wins for each partition were collected and that is what is shown in Figure 7, Figure 8, Figure 9, and Figure 10. For each level of randomness and for each game, it took 20 minutes to train the BERT model.

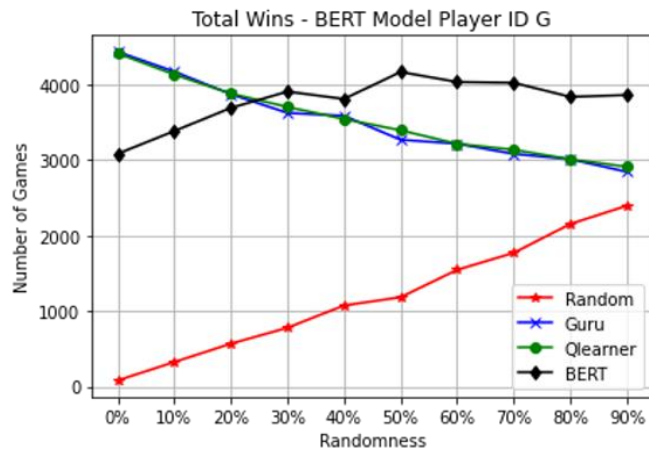


Figure 7. Nim Player ID G. The BERT model inferred and played with the Guru (G) ID token being specified.

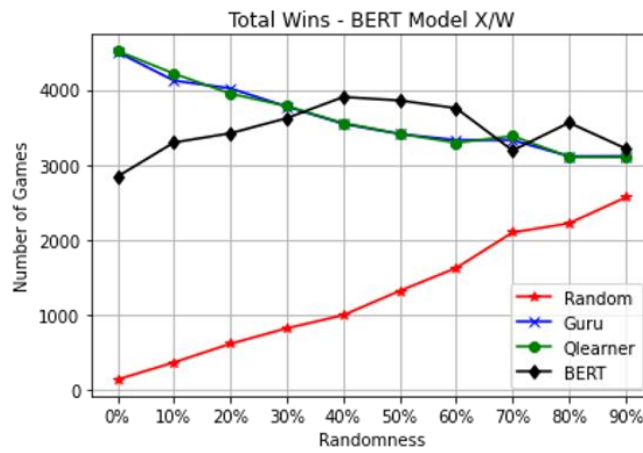


Figure 8. Nim X/W Win State

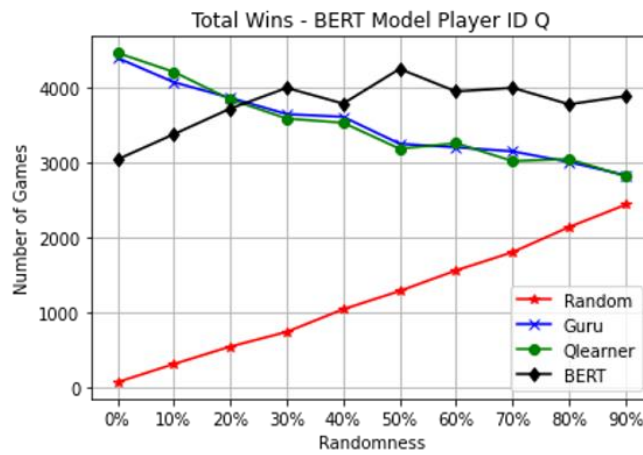


Figure 9. Nim Player ID Q. The BERT model inferred and played with the Q learner (Q) ID token being specified.

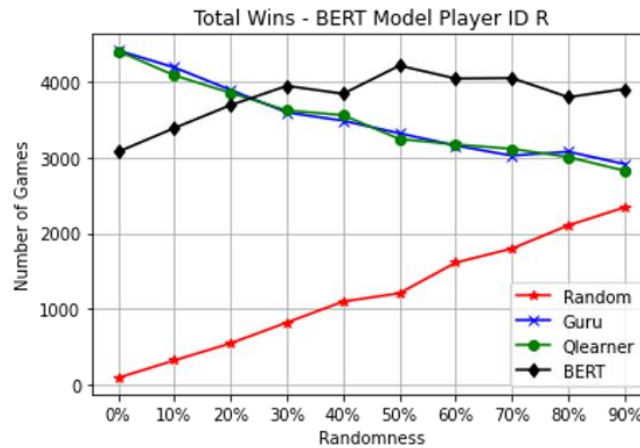


Figure 10. Nim Player ID R. The BERT model inferred and played with the random agent (R) ID token being specified.

The BERT model consistently beats the other agents as the level of randomness increases in the dataset. This supports our original hypothesis because it shows evidence that the BERT model can identify the strongest signal (Guru and Q-learner) despite the random noise. This is especially evident in the 90% index of the results. Despite learning from a dataset where the players were only making one out of every ten of their moves, the model performed better than them. This is shown in Figure 7, Figure 9, and Figure 10 where at randomness threshold of 30% the BERT model outperforms all the agents. This also held true up till 100% random. The model did not perform well however with a win/loss indicator system (graphed in Figure 8). In fact, the model somewhat follows the same performance trend as the agents.

The process of playing as one player or the other is defined within the state space of the text sequence. This is because the text sequence for a Nim sequence is generalized as the game space followed by an indicator token, and the corresponding move. For example, the sequence a10/b10/c10 G – [MASK] indicates that this should be a Guru agent move and a sequence such as a10/b0/c0 W – [MASK] indicates this is a winning move. These types of indicators are encoded into the dataset, and the transformer model learned these patterns.

The role of these indicators in the performance of the model is interesting. As it shows that one could perform additionally postprocessing on the dataset to add more attributes from which the model could learn from.

3.2. Results Chess

Only one BERT word piece tokenizer was trained on the Chess dataset. Its total vocabulary size is 16,000 tokens so it is not possible to show here like the Nim vocabularies.

The chess dataset was created with the first three moves (for each max level chess engine) per game. This is important to keep in mind as the BERT model seemed to perform well given that it had a very small subset of all possible chess moves.

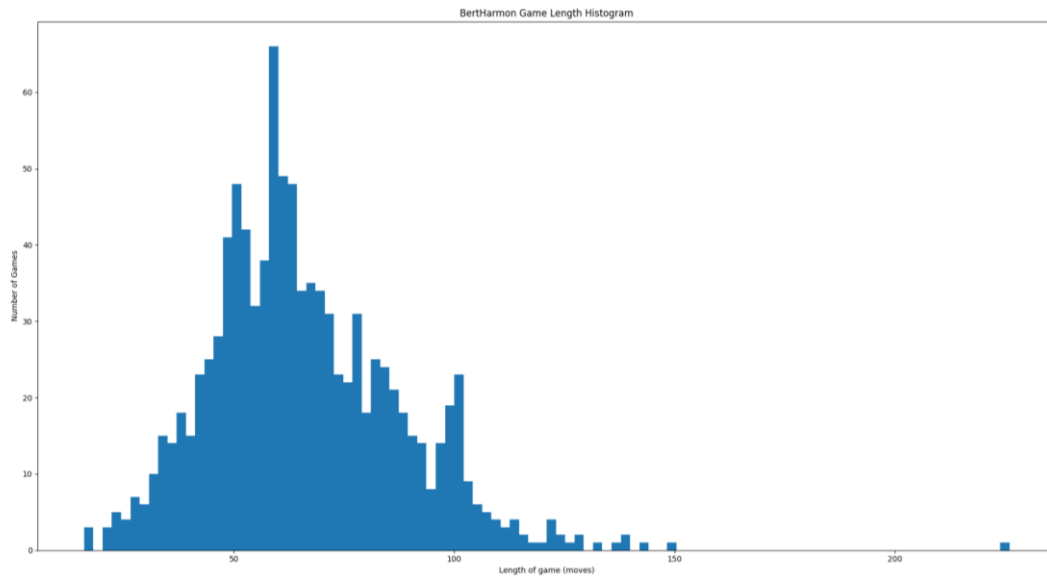


Figure 11. BERT Chess Game Length Distribution Versus Grandmaster Chess Engine

The BERT Chess model was not proficient enough to win chess games, so instead we show how well it did at playing chess against a grandmaster level chess player (Stockfish at max ELO). Additionally, it took around 3 days to train. Since the choice of a move given a board space is an open-ended answer, the model could technically answer with any text that it had in its vocabulary. We consider this as a feature of the model that it was given the option of answering in an incorrect format. As a result, we benchmarked its accuracy in terms of giving valid chess moves (shown in Figure 12). Given that it was only aware of the opening states of chess, it is impressive that at 35 moves into a game (35 moves for each player) it has an accuracy of 75%. When the model got a move wrong, we substituted a stockfish move in its place, and kept the game going.

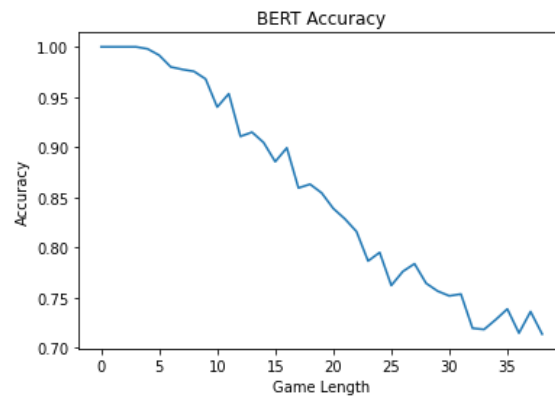


Figure 12. BERT Accuracy for Choosing Valid Chess Moves

In addition to measuring accuracy, we also measured the game length endurance of the model to understand how long it could play against a grandmaster stockfish engine till it lost. The results are graphed in Figure 11. Surprisingly BERT could survive for on average 32 moves (65 moves total for the game, ~32 per player), and at most we saw a game lasting for more than 200 moves.

4. CONCLUSIONS

In conclusion, we have shown that the BERT model is capable of learning both the games of Nim and Chess. We built text corpuses by representing game states and moves in a text sequence format. We have shown that the BERT transformer model is able to learn games in the context of very little information, in the presence of large quantities of noise, and in the presence of a large amount of data. The BERT model has been shown to learn the behaviors and patterns of primary game agents Guru, Q-learner, and stockfish such that the model can emulate their actions.

The results of our research should encourage BERT and other transformer models to be used as few-shot learners in situations where data is expensive to gather, difficult to clean, and in very high dimensional learning environments.

Transformer language models can represent input sequences efficiently through various autoencoding steps such as BERT, ALBERT, RoBERTa, ELECTRA, etc. Exploring the performance of these language models can help improving chess language models in this study. Second, these models can be used to cluster chess opening positions in order to compare and contrast to the ECO chess openings taxonomy. Future work will explore the clustering of chess positions to build taxonomy of openings, middle game positions and end game positions. This approach is analogous to text summarization where BERT approaches are known to be successfully applied. Third, future work will investigate player attribution in chess by analyzing various master games in chess databases as certain player styles are known to exist, such as Karpov likes closed and slow games, as Kasparov and Tal like open and sharp games.

Additionally, by representing a game space in our text sequence format, there are several interesting use cases with BERT such as authorship attribution, author playstyle and game space deduction. Given a dataset of games played by grandmasters, one could train this model and assess the probability that a given move has been made by Kasparov or another grandmaster by solving for the author/agent token instead of a move token. Additionally, if we solve for the move token then one could identify how a specific grandmaster would play given the board state. These two use cases allow for someone to prepare against a specific opponent. Given there are three spaces of the sequence, the last portion to solve for is the game space, and interestingly one could solve for the game space to suggest what is the most likely game space to precede this move for this player. All these use cases generalize for practical real-world problems that can be conceptualized into a state-independent text corpus such as predicting consumer behavior.

ACKNOWLEDGEMENTS

Special thanks to Booz Allen Hamilton, Johns Hopkins University, as well as our friends and family for their support.

REFERENCES

- [1] Chinchalkar, Shirish. "An upper bound for the number of reachable positions." *ICGA Journal* 19.3 (1996): 181-183.
- [2] Maharaj, Shiva, Nick Polson, and Alex Turk. "Chess AI: competing paradigms for machine intelligence." *Entropy* 24.4 (2022): 550.
- [3] Magnuson, Max. "Monte carlo tree search and its applications." *Scholarly Horizons: University of Minnesota, Morris Undergraduate Journal* 2.2 (2015): 4.
- [4] Hsu, Feng-hsiung. "IBM's deep blue chess grandmaster chips." *IEEE micro* 19.2 (1999): 70-81.
- [5] McGrath, Thomas, et al. "Acquisition of chess knowledge in alphazero." *arXiv preprint arXiv:2111.09259* (2021).

- [6] Brown, Tom, et al. "Language models are few-shot learners." *Advances in neural information processing systems* 33 (2020): 1877-1901.
- [7] Matanović, A., M. Molarović, and A. Božić. "Classification of chess openings." (1971).
- [8] Silver, David, et al. "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play." *Science* 362.6419 (2018): 1140-1144.
- [9] Stöckl, Andreas. "Watching a Language Model Learning Chess." *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*. 2021.
- [10] Edwards, Steven J. "Portable game notation specification and implementation guide." Retrieved April 4 (1994): 2011.
- [11] Nasu, Yu. "Efficiently updatable neural-network-based evaluation functions for computer shogi." *The 28th World Computer Shogi Championship Appeal Document* (2018).
- [12] Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." *arXiv preprint arXiv:1810.04805* (2018).
- [13] Radford, Alec, et al. "Improving Language Understanding by Generative Pre-Training." (2018)
- [14] Radford, Alec, et al. "Language models are unsupervised multitask learners." *OpenAI* (2019)
- [15] E. JÄRLEBERG, "Reinforcement Learning Combinatorial Game Nim (2011).pdf," KTH Royal Institute of Technology, 2011.

AUTHORS

Michael DeLeo is an engineer and researcher. He currently works at Booz Allen Hamilton as a Machine Learning Engineer. He graduated from Penn State with a BS in Computer Engineering (minors in Math and Computer Science). He is also currently studying for his masters in Artificial Intelligence at Johns Hopkins University where he is performing research on NLP. Email ID: mdeleo2@jhu.edu



Erhan Guven is a faculty member at JHU WSE. He also works at JHU Applied Physics Lab as a data scientist and researcher. He received the M.Sc. and Ph.D. degrees from George Washington University. His research includes Machine Learning applications in speech, text, and disease data. He is also active in cybersecurity research, graph analytics, and optimization. Email ID: eguven2@jhu.edu



A TRANSFORMER BASED MULTI-TASK LEARNING APPROACH LEVERAGING TRANSLATED AND TRANSLITERATED DATA TO HATE SPEECH DETECTION IN HINDI

Prashant Kapil and Asif Ekbal

Department of Computer Science and Engineering, IIT Patna, India

ABSTRACT

The increase in usage of the internet has also led to an increase in unsocial activities, hate speech is one of them. The increase in Hate speech over a few years has been one of the biggest problems and automated techniques need to be developed to detect it. This paper aims to use the eight publicly available Hindi datasets and explore different deep neural network techniques to detect aggression, hate, abuse, etc. We experimented on multilingual-bidirectional encoder representations from the transformer (M-BERT) and multilingual representations for Indian languages (MuRIL) in four settings (i) Single task learning (STL) framework. (ii) Transferring the encoder knowledge to the recurrent neural network (RNN). (iii) Multi-task learning (MTL) where eight Hindi datasets were jointly trained and (iv) pre-training the encoder with translated English tweets to Devanagari script and the same Devanagari scripts transliterated to romanized Hindi tweets and then fine-tuning it in MTL fashion. Experimental evaluation shows that cross-lingual information in MTL helps in improving the performance of all the datasets by a significant margin, hence outperforming the state-of-the-art approaches in terms of weighted-F1 score. Qualitative and quantitative error analysis is also done to show the effects of the proposed approach.

KEYWORDS

M-BERT, MuRIL, Weighted-F1, RNN, cross-lingual.

1. INTRODUCTION

The emergence of social media platforms like Facebook and Twitter has led to an exponential increase in user-generated content. The identification of hate speech within a large volume of posts on social media has posed a challenge and thus is a growing research area. There is a growing need to develop an automated classifier to detect different forms of hate speech such as offensive, profanity, abusive, and aggression that are prevalent on different social media platforms. The offensive posts which create social disability need to be restricted alongside maintaining the right to freedom of speech.

These incidents create mental and psychological agony for the users resulting in deactivating the account or in some cases committing suicide [1]. While research in this area is gaining momentum, there is a lack of research in the Hindi language. In multilingual societies like India usage of code-mixed languages is common for conveying any opinion. Code-mixing is a phenomenon of embedding linguistic units such as phrases, words, or morphemes of one language into an utterance of another [2]. In social media, Hindi posts are generally present in

either Devanagari script or Hindi-English code mixed pattern. To build an efficient classifier supervised learning on the labeled dataset is the most common approach. In India, native vernacular languages are spoken by a majority of the population. Mixed code language like Hinglish is most prevalent in social media conversations. [3] reported in 2015 that India ranked fourth in the social hostilities index with an index value of 8.7 out of 10., indicating the need to solve this problem of hate speech. Code-switched language presents challenges of randomized spelling variations in explicit words due to foreign script and ambiguity arising due to various interpretations of words in different contextual situations [4]. The detection of hate speech is very important for lawmakers and social media platforms to curb any wrong activity. Table 1 consists of the definition followed to collect the different sub class of hate. Table 2 enlists the laws on hate speech in some of the countries.

The significant contributions of this work are as follows:

Dataset: We utilized eight benchmark datasets related to the hate domain, aggressiveness, offensiveness, abuse, etc. To add cross-lingual information we also translated eleven English data to Devanagari script by leveraging Google Translate. The Devanagari tweets were also transliterated to the Roman script by using Indic-trans [19].

Model: We investigated the various state-of-the-art models such as M-BERT and MuRIL to design eight models. The first set of model is based on single task learning paradigm. The knowledge from the transformer encoder is transferred to the bidirectional long short term memory (Bilstm) in our second set of models. The third set of model is based on MTL paradigm and in the fourth set the encoder is first pre-trained with translated and transliterated data followed by leveraging the MTL on eight data.

Error Analysis: The results and errors on the experimented models were analyzed by presenting qualitative and quantitative analysis to highlight some of the errors that need to be rectified to improve the system performance.

The remaining structure of this paper is as follows.

A brief overview of the related background literature is presented in Section 2. In Section 3, the datasets used for the experiments are discussed. Section 4 discusses in detail the proposed methodology, experimental setup. Section 5 reports the evaluation results and comparisons to the state-of-the-art, and Error analysis containing qualitative and quantitative analysis of the obtained results. Finally, the conclusion and directions for future research are presented in Section 6.

Table 1. Definition of hate speech

Authors	Definition
[26]	The post contains hate, offensive, or profane content.
[25]	The posts contain covertly and overtly aggressive messages.
[4]	The tweets were labeled as hate speech if they satisfied one or more of the conditions: (i) tweet using sexist or racial slur to target a minority, (ii) undignified stereotyping or (iii) supporting a problematic hashtags such as #ReligiousSc*m.
[7]	It is a bias-motivated hostile speech aimed at a person or group of people with intentions to injure, dehumanize, harass, degrade and victimize targeted groups based on some innate characteristics.
[8]	It is defined as abusive speech containing a high frequency of stereotypical words.

Table 2: Laws of different countries on hate speech

Country	Law
USA	Hate speech is legally protected free speech under the First Amendment. However, speech that includes obscenity, speech integral to illegal conduct, and speech that incites lawless action or is likely to produce such activity are given lesser or no protection.
Brazil	According to the 1988 Brazilian constitution racism is an offense with no statute of limitations and no right to bail for the defendant.
Germany	Section 130 of the German criminal code states incitement to hatred is a punishable offense leading up to 5 years imprisonment. It also states that publicly inciting hate against some parts of the population or using insulting malicious slurs or defaming to violate their human dignity is a crime.
India	Article 19(1) of the constitution of India protects the freedom of speech and expression. However, article 19(2) states that to protect sovereignty, integrity, and security of the state, to protect decency and morality, defamation and incitement to an event, some restrictions can be imposed
Japan	The Hate speech act of 2016 does not apply to groups of people but covers threats and slander to protect.
New Zealand	Their Hate speech act follows Section 61 of the Human Rights Act 1993 that asserts that threatening, abusive content in any form, words that are likely to create hostility against a group of people based on race, color, or ethnicity is unlawful.

2. RELATED WORK

2.1. Hate speech detection in low resource languages

This section summarizes the works done on hate speech detection for low-resource languages.

Arabic: [5] investigated the religious hate speech detection on 6000 labeled data in Arabic from Twitter. They created and published three lexicons of religious hate terms. They investigated three different approaches namely lexicon-based, n-grams-based, and gated-recurrent unit (GRU)-based neural networks with word embeddings provided by AraVec [6]. [7] presented 3353 Arabic tweets tagged for five classification tasks. They analyzed the difficulties of collecting and annotating the Arabic data and determined 16 target groups like women, gay, Asians, Africans, immigrants, refugees, etc. The experiments showed that deep learning settings outperform the BOW (Bag of words) based method in all five tasks. [8] introduced the first Levantine Hate speech and abusive (L-HSAB) data comprising 5846 tweets tagged into three categories: normal, hate, and abusive. The results indicated the outperformance of naive Bayes (NB) over support vector machines (SVM) in both binary and multi-class classification experiments.

French: [9] described CONAN: as the first large-scale, multilingual, and expert-based hate speech/counter-narrative dataset for English, French and Italian. The data consist of other meta-data features such as expert demographics, hate speech sub-topic, and counter-narrative type. [7] created a hate speech dataset in English, French, and Arabic annotated for the five classification tasks: the directness of the speech, the hostility type of the tweet, the discriminating target attribute, the target group, and the annotator's sentiment.

German: [10] developed a dataset containing offensive posts by including their target. They implemented a two-step approach to detect the offending statements. The first step is a binary classification between offensive and not offensive. The second step classifies offensive into severity = 1 and severity =2. [11] released the pilot edition of the GermEval shared task on the Identification of Offensive Language comprising 8000 posts annotated for two layers. The first

layer is the coarse-grained binary classification between offensive and other. The second layer is fine-grained 4-way tagging of the offensive post between profanity, abuse, insult, and others. The popular features leveraged to solve the task were word embeddings, character n-grams, and lexicons of offensive words.

Italian: [12] created an Italian twitter corpus of 6000 tweets annotated for hate speech against immigrants and designed a multi-layer annotation scheme to annotate the post's intensity, aggressiveness, offensiveness, irony, and stereotypes. [13] proposed a shared task to solve the Hate Speech detection (HaSpeeDe) on Italian Twitter and Facebook. The teams utilized traditional machine learning approaches, such as support vector machine (SVM), logistic regression (LR), random forest (RF), and deep learning techniques such as convolution neural network (CNN), gated recurrent unit (GRU), and multi-layer perceptron (MLP), etc. The results also confirmed the difficulty of cross-platform hate speech detection.

There is little work done for other low-resource languages, which include Spanish ([14],[38]), Polish [15], Portuguese [45], Slovene [16], Turkish [17] and Indonesian [18].

2.2. Hate speech classification in Hindi

There has been little effort to solve the Hate speech detection in a low-resource language such as Hindi due to the scarcity of labeled data. The cost of generating labeled data is often time-consuming and tedious, limiting the further development of machine learning approaches. In recent years, shared tasks have been organized for low-resource languages, such as Hindi to solve the task of aggressive identification or hate classification. [20] released 15000 aggression annotated Facebook posts and comments in Hindi (Roman and Devanagari script). [21] conducted experiments with deep neural network models of varying complexity ranging from CNN, LSTM, BiLSTM, CNN-LSTM, LSTM-CNN, CNN-BiLSTM, and BiLSTM-CNN. To improve over the baseline, they also utilized data augmentation, pseudo labeling, and sentiment score as the feature. [22] explored the combination of passive-aggressive (PA) and SVM classifiers with character-based n-gram (1-5) TF-IDF for the feature representations. [23] uses LSTM, and CNN initialized with fast text word embeddings, and [24] uses BiLSTM with glove embeddings to solve the problem.

In recent times multi-layer annotated data to cover the different facets of a post has been released. [25] presented a shared task featuring two tasks: first is aggression identification to discriminate overtly, covertly, and non-aggressive posts and the second is gendered aggression identification. The approaches used by different teams were mostly based on neural networks such as CNN, LSTM, and BiLSTM initialized with word embeddings. The utility of M-BERT, XLM-RoBERTa, DistilRoBERTa, and transfer learning techniques based on universal sentence encoder (USE) embedding were also explored to solve the task. [26] and [27] developed 2 layer annotated data. The first is classified between Hate and Offensive (HOF) and non-hate (NOT). The second task is a fine-grained classification of HOF into hate, offensive, and profanity. [28] pre-trained the word vectors by 0.5 million in-domain unlabeled data to obtain task-specific embeddings. This knowledge is then transferred to CNN for classification. They observed that CNN outperforms LSTM when transfer learning through word vectors is utilized. [29] released the DHOT dataset in Devanagari script and developed a classifier based on FastText embeddings to classify offensive and non-offensive tweets. [30] explored IndicBERT, RoBERTa Hindi, and neural space BERT Hindi to solve the binary classification between Hate and Offensive (HOF) and NOT. [31] proposed to enhance the hate speech detection of code mixed Hind-English by incorporating social media-based features along with capturing profanity features into the model. They also proposed a novel bias elimination algorithm to mitigate any bias from the model. [32] experimented with two architectures, namely the sub-word level LSTM model and hierarchical

LSTM model with attention, based on phonemic sub-words for hate speech detection on social media code-mixed text. [2] presented an annotated corpus of 4575 tweets in Hindi-English code mixed text. To build the classification system, they utilized features such as character n-grams, punctuations, negation words, word n-grams, punctuations, negation words, and a hate lexicon.

3. DATA SETS

In this section, we will briefly describe all 8 Hindi datasets related to the hate domain used in this paper. The statistics of all the hate-related data are in Table 3.

Data 1 (D1) [29]: A lexicon of abusive words in Hindi were built. The 20 abusive terms collected serve as keywords that were assigned to a data acquisition program. The tweets were also mined from popular Twitter hashtags of viral topics, and popular public figures like politicians, sports personalities, and movie actors. The annotation of DHOT tweets is done by three language experts. The average value of cohen kappa for the inter-annotator agreement is 84%.

Data 2 (D2) [26]: The authors followed the heuristics approach to search for hate speech in an online forum by identifying the topics for which hate speech can be expected. Different hashtags and keywords were used to sample the posts from Twitter and Facebook. The inter-annotator agreement score obtained is 36%.

Data 3 (D3) [27]: The sampling of the dataset was done during the extremely hard COVID-19 second wave in India. Therefore during the sampling process, major topics in social media are influenced by COVID-19. To obtain potential hateful tweets, a weak classifier based on an SVM classifier with n-grams features to predict weak labels on the unlabeled corpus is used. The trending hashtags used to sample the tweets were *#resignmodi*, *#TMCTerror*, *#chinesevirus*, *#islamophobia*, *#covidvaccine*, *#IndiaCovidcrisis*, etc. The inter-annotator agreement score is 69%.

Data 4 (D4) [25]: The data is crawled from the public Facebook pages and Twitter. For Facebook, more than 40 pages were crawled which included news websites, web-based forums, political parties, student organizations, etc. For Twitter, the data was collected using some of the popular hashtags such as beef ban, election results, etc. The complete dataset contains 18K tweets and 21K Facebook comments annotated with aggression and discursive effects. The inter-annotator agreement for the top level is 72%.

Data 5 (D5) [46]: The dataset is collected from various social media platforms namely Facebook, Twitter, and Youtube. The actual sources of information ranged from public posts, tweets, videos, news coverage, etc. The annotation of data involves multiple human interventions and constant deliberations over the justification of assigned tags.

Data 6 (D6) [33]: They collected posts from various social media platforms like Twitter, Facebook, Whatsapp, etc. To collect hate speech data, the tweets encouraging violence against minorities based on race and religious beliefs were sampled. The timeline of the users with significant hate-related posts was also analyzed. The offensive posts are crawled by Twitter search API by employing the list of swear words used in the Hindi language released by [29]. The posts related to the defamation category are collected from viral news articles where people or a group are publicly shamed due to misinformation. The topic-wise search is performed to collect defamation tweets.

Data 7 (D7) [4]: The tweets were mined from popular Twitter hashtags of viral topics across the news feed. The tweets were collected from the Twitter handles of sportspersons, political figures, news channels, and movie stars. The annotation of tweets was done by three annotators having a background in NLP research. The Cohen kappa inter-annotator agreement score is 83%.

Data 8 (D1) [2]: presented an annotated corpus of 4575 tweets in Hindi-English code mixed text. To build the classification system, they utilized features such as character n-grams, punctuations, negation words, word n-grams, and a hate lexicon. The Kappa score is 98.20%.

Table 3. Statistics of dataset

Datasets	Labels and train/test set
D1	HOF: 403/81, NOT: 1200/316
D2	HOF: 2469/605, NOT: 2196/713
D3	HOF: 1433/483, NOT: 3161/798
D4	OAG: 6072/362, CAG: 6115/413 NAG: 2813/195
D5	OAG: 1118/669, CAG: 1040/215 NAG: 2823/316
D6	Hostile: 3054/780, Non-Hostile:3485/873
D7	Abusive: 1765 , Hate: 303 Neutral: 1121
D8	Hate: 1299, Neutral: 2249

Table 4: Translated and Transliterated sample

S1	We dont trust these n****s all these bitch.
Translation	हम इन सभी काले लोगों पर भरोसा नहीं करते हैं
Transliteration	ham in sabhi kutiya par bharosa nahi karte.
S2	your grammar is trash.
Translation	आपका व्याकरण कचरा है
Transliteration	aapka vyakaran kachra hai.
S3	you are irrelevant b****h.
Translation	तुम्हारे तुम अप्रासंगिक हो
Transliteration	aap aprasangik kutiya hai.

3.1 Cross-lingual Data

As there are abundant data available for English, we aim to determine if knowledge from one language can be used to improve the performance of another language. We utilized eleven English data [34], [35], [20], [36], [37], [26], [27], [38], [39], [40], and [41].

Translated Data: The Google translate API is used to translate approximately 2,50,000 tweets to the Devanagari script. We have selected 100 random samples to analyze the translation of the original post. The human evaluation found the translation to be satisfactory.

Transliterated Data: After obtaining the translated Devanagari posts, it is transliterated to Romanized form by Indic-trans [19]. Table 4 consists of some instances of translated and transliteration.

4. METHODOLOGY

4.1. Pre-Processing

Social media posts contain a lot of noisy text which is not considered a useful feature for the classification. We perform the following steps to remove the noise, and make it ready for experiments:

1. Words are reduced to lower case so that words such as "BI**H", "bi**h" and "Bi**h" will have the same syntax and will utilize the same pre-trained embedding values.
2. Word segmentation is being done using the Python-based word segment to preserve the important features present in hashtag mentions.
3. All the emoticons were categorized into 5 categories, namely प्रेम *love*, दुःख *sad*, खुशी *happy*, आश्चर्य *shocking*, and गुस्सा *anger*. The Unicode character of the emoticon in the text is substituted with one category.
4. All the @ (ex. @abc) mentions were replaced with the common token, i.e *user*.
5. The stop words were not removed due to the risk of losing some useful information, and this was also empirically found to be of little or no impact on the classification performance after removing them.
6. The maximum sequence length is set to 40. Post padding is done if any sentence is less than 40 and pruning is performed from the last if the sentence is greater than 40.

We experimented on 8 transformer-based approaches which are discussed in this section.

4.2. Models

Model 1(M1): Multilingual-BERT: [42] introduced {**M-BERT**} i.e Multilingual Bidirectional Encoder Representation from Transformers to pre-train deep bidirectional representations from unlabeled texts by jointly conditioning on both left and right contexts in all layers. There are two steps in the training framework: pre-training and fine-tuning. During pre-training, the model is trained on unlabeled data over different pre-training tasks. For fine-tuning, the BERT model is first initialized with the pre-trained parameters and all of the parameters are fine-tuned using labeled data from the downstream tasks. It follows two training objectives which are described as follows:

Masked language modeling (MLM): The model randomly masks 15% of the tokens from the input, and the objective is to predict the masked words based only on their context. The training data generator chooses 15% of the token positions at random for prediction. If the *i*th token is chosen it is replaced with (1) the [MASK] token 80% of the time (2) a random token 10% of the time (3) the unchanged *i*th token 10% of the time.

Next Sentence prediction (NSP): It jointly pre-trains text-pair representations, and the model is to predict whether two sentences are following each other or not.

The multi-lingual version of the BERT is capable of working with 104 languages. The first token of every sequence starts with a unique classification token ([CLS]). The final hidden state corresponding to this token is used as the aggregate sequence representation for the classification task.

Model 2(M2): MuRIL [43]: It is a multilingual language model specifically developed for the Indian languages by training on IN text corpora of 16 Indian languages. It utilizes two training

objectives: MLM and Translation language modeling (TLM). The MLM uses monolingual text only (unsupervised), and TLM uses translated and transliterated document pairs to train the model. The maximum sequence length is 512, global batch size of 4096, and trained for 1M steps. The total trained parameters are 236M that is optimized by Adam optimizer with the learning rate of $5e-4$. The general architecture of transformer encoder block is shown in Figure 1

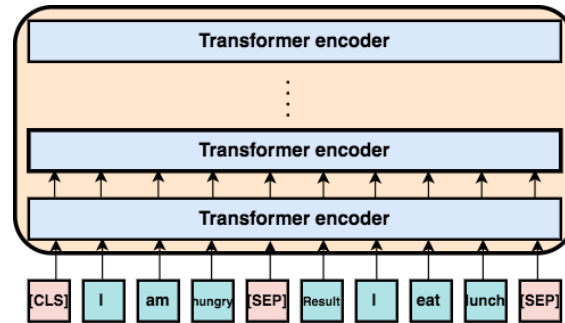


Figure 1. Transformer encoder

4.2.1. Knowledge Transfer

[42] compared different combinations of layers of BERT to conclude that the output of the last four layers combined encodes more information than only the last layer. In this work, we utilize the last 4 hidden layers output from pre-trained M-BERT and MuRIL models into Bilstm followed by the softmax activation function. Figure 2 shows the architecture.

Model 3 (M3): M-BERT-Bilstm: The concatenation of the last 4 hidden layers was passed into Bilstm.

Model 4 (M4): MuRIL-Bilstm: The concatenation of the last 4 hidden layers was passed into Bilstm.

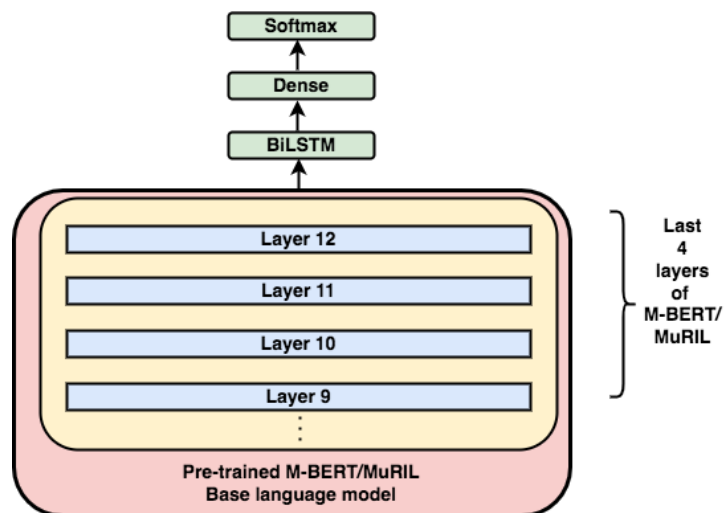


Figure 2. M-BERT/MuRIL-Bilstm architecture

4.2.2. Multi task learning (MTL)

Multi-tasking learning aims at solving more than one problem simultaneously. End-to-end deep multi-task learning has been recently employed in solving various problems of natural language processing (NLP). It enables the model by sharing representations between the related tasks and generalizing better by achieving better performance for the individual tasks.

[47] developed two forms of MTL, namely Symmetric multi-task learning (SMTL) and Asymmetric multi-task learning (AMTL). The former is joint learning of multiple classification tasks, which may differ in data distribution due to temporal, geographical, or other variations, and the latter refers to the transfer of learned features to a new task to improve the new task's learning performance.

[48] discussed the two most commonly used ways to perform multi-task in deep neural networks.

(i) **Hard Parameter Sharing:** Sharing the hidden layers between all tasks with several task-specific output layers.

(ii) **Soft Parameter Sharing:** Each task has its specific layers with some sharable parts.

Model 5(M5): This model leverages the M-BERT trained in the MTL paradigm.

Model 6(M6): This model leverages the MuRIL trained in the MTL paradigm.

The architecture of the MTL-DNN is shown in Figure 3. The lower layers are shared across all the tasks, while the top layers represent task-specific outputs. In our experiment, all the tasks are classified. The input X is a word sequence (either a sentence or a pair of sentences packed together) represented as a sequence of embedding vectors, one for each word in I_1 . Then the transformer encoder captures the contextual information for each word via self-attention and generates a sequence of contextual embedding in I_2 . The shared semantic representation is trained by the multi-task objectives. In the following, we will describe the model in detail.

Lexicon Encoder (I1): The input $X = \{x_1, x_2, \dots, x_m\}$ is a sequence of tokens of length m . Following [42] the first token x_1 is always the $\{CLS\}$ token. If X is packed by a sentence pair (X_1, X_2) , we separate the two sentences with a special token $[SEP]$. The lexicon encoder maps X into a sequence of input embedding vectors, one for each token, constructed by summing the corresponding word, segment, and positional embeddings.

Transformer Encoder (I2): It consists of a multi-layer bidirectional Transformer encoder [49] to map the input representation vectors (I1) into a sequence of contextual embedding vectors C belongs to $R(d*m)$. This will be the shared representation across different tasks. MT-DNN learns the representation using multi-task objectives, in addition to pre-training.

Single-Sentence Classification Output: Suppose that x is the contextual embedding (I2) of the token $[CLS]$ that can be viewed as the semantic representation of input sentence X . The probability that X is labeled as class c is predicted with softmax.

$$P_r(c|X) = \text{softmax}(W_{SST}^T \cdot x), \quad (1)$$

The training procedure of MT-DNN consists of two stages: pre-training and multi-task learning.

In the multi-task learning stage, mini-batch-based stochastic gradient descent (SGD) is used to learn the parameters of our model. In each epoch, a mini-batch b_i is selected among all the tasks

For the classification tasks, the loss function used is categorical cross-entropy loss.

$$-\sum_c \mathbb{1}(X, c) \log(P_r(c|X)), \quad (2)$$

Where $\mathbb{1}(X, c)$ is the binary indicator (0 or 1) if class label c is the correct classification for X

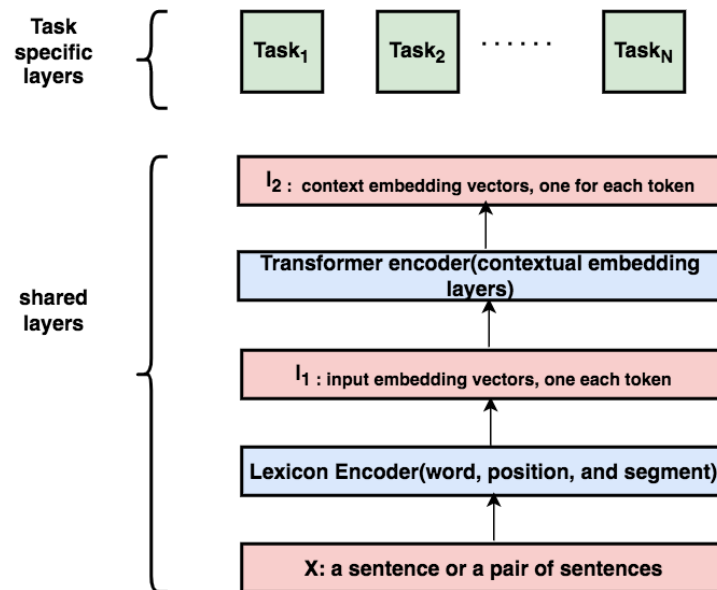


Figure 3. Multi task learning architecture with BERT/ALBERT as shared encoder

4.2.3. Pre-training with Cross lingual Information

Step 1: We utilize the 250K translated data and 250K transliterated data to pre-train the M-BERT and MuRIL.

Step 2: The trained parameters is used to initialize the weight of the shared encoder.

Step 3: The same procedure as of multi task learning is followed as in Figure 3.

Model 7 and **Model 8** utilizes the cross lingual information.

4.2.4. Experimental Setup

All the deep learning models were implemented using Keras, a neural network package [50] with Tensorflow [51] as the backend. Each dataset is split into an 80:20 ratio to use 80% in grid-search to tune the batch size and learning epochs using 5-fold cross-validation experiments and test the optimized model on 20% held-out data. The results are the mean of 5 runs with the same setup. For some data with a separate test set, the model is trained on train data, and performance is evaluated using test data. Categorical cross-entropy is used as a loss function, and Adam [52] optimizer is used for optimizing the network.

We use a learning rate of $2e-5$ for the transformer models. The batch size of 30 is used to train the shared encoder and an epoch of 2 is found to be optimal. The value for bias is randomly initialized to all zeros, the relu activation function is employed at the intermediate layer, and Softmax is utilized at the last dense layer. The transformers library is loaded from Hugging Face. It is a python library providing a pre-trained and configurable transformer model useful for various NLP tasks.

5. RESULTS, COMPARISON AND ANALYSIS

We report the weighted-F1 score of all the eight datasets in Table 5. Table 6 enlists comparison with the state-of-the-art approaches and the proposed approach over the weighted-F1 score. From the results it can be seen that pre-training with the translated and transliterated data followed by training in MTL outperformed the other methods. We are also presenting the statistical significance results between best and worst model in Table 7. We also did some qualitative analysis and presented different patterns of hate posts detected by the best model.

Table 5. Weighted-F1 scores of eight data sets

Data	M1	M2	M3	M4	M5	M6	M7	M8
D1	91.43	94.41	92.94	94.53	94.94	94.89	95.19	94.99
D2	74.47	81.64	77.11	82.57	79.55	82.27	82.78	82.39
D3	77.09	80.41	79.14	82.12	82.82	81.23	83.11	82.94
D4	61.80	59.80	62.67	61.80	65.63	63.89	65.97	64.23
D5	76.90	74.50	80.51	81.38	80.98	81.96	81.22	82.14
D6	82.32	80.94	83.45	81.28	85.98	83.65	86.14	83.78
D7	84.98	85.85	85.97	88.27	90.16	88.10	90.98	88.67
D8	89.10	89.10	89.58	90.51	90.10	91.78	90.86	92.17

5.1. Qualitative Analysis

In this section we are giving four types of hate posts with the explanation which were correctly classified by the model.

HATE IS TOXIC

GROUND VALUE: **HATE**

PREDICTED CLASS: **HATE**

1. अपनी औकात भूल गए हो तुम कुत्ते सुवर की औलाद

TRANSLITERATION: Apni aukaat bhool gaye ho tum suwar kii aulaad.

TRANSLATION: You have forgotten your real worth. You son of a pig.

2. पता लगा बे हराम कौन ट्रेंड कर रहा है

TRANSLITERATION: Pata laga be haram kaun trend kar raha hai.

TRANSLATION: Find out you scoundrel, who the hell is trending.

EXPLANABILITY: Both the tweets consists of slang term such as s***r , and h***m. As the training data consists of large number of tweets containing these terms it detected it successfully.

INDIRECT REFERENCES

GROUND TAG: HATE

PREDICTED CLASS: **HATE**

1. Kaun rapper aachha gaata hai. I hate all. Bas music kaa kachara karne aaye hai sab
TRANSLITERATION: Kaun rapper aachha gaata hai. I hate all. Bas music kaa kachara karne aaye hai sab.
TRANSLATION : No rapper is good enough, I hate all of them as they are just making the trash of music.
2. आखिर कब तक जनता उठाएगी निकम्मे कर्मचारियों का बोझ

TRANSLITERATION: Aakhir kab tak janta uthaegii nikamme karmachariyon kaa bojha.

TRANSLATION: After all, how long will the public bear the burden of the useless employees.

EXPLANABILITY: Here Indirect attack in a softer tone is being done which the model is able to detect.

CONTEXTUAL INFORMATION

GROUND TAG: HATE

PREDICTED CLASS: **HATE**

1. जो भी हो मुझे भी लगता है । दाल में कुछ काला
TRANSLITERATION :Jo bhi ho mujhe bhi lagta hai, daal me kuch kaala.
TRANSLATION: Whatever, even I think there is something fishy.
2. @INCINDIA ISHLIYE CORRUPTION KE JARIYE SAB KI KHOON CHOOS RAHE HEINE
TRANSLITERATION: Isliye corruption ke jariye sabi kii khoon choos rahe hain.
TRANSLATION: Thats why, sucking everyone's blood through corruption

EXPLANABILITY: These two tweets also needs the contextual information to get the true sentiment. As the model is also learning the cross-lingual information it is able to detect it.

HATE IS SARCASTIC

1. अभी तो कबीर सिंह फिल्म की वजह ये लोग पागल हो रखे है, जब RX100 का रीमेक आएगा तबतो चूड़ियां तोड़ेगी ये फेमिनिस्ट.

TRANSLITERATION: abhi to kabir singh film kii wajah ye log pagal ho rakhe hai, jab RX100 kaa remake aaega tab to churiyaan torengi ye feminist..

TRANSLATION: Right now these people are going crazy because of Kabir Singh movie, when the remake of RX100 comes, then these feminists will break bangles.

2. BOLLYWOOD FILM DEKHNE KE SAMAY LOGIC GHAR MEIN CHORKE ANA PARTA HAIN. PLEASE LOGIC MAT GHUSAO

TRANSLITERATION:. Bollywood film dekhne ke samay logic ghar mein chorke ana parta hai. Please logic mat gusao.

TRANSLATION: you have to leave your brain behind before watching any Bollywood movie. Please don't use any logic.

EXPLANABILITY: These tweets are sarcastic in nature. But as the encoder consists of all types of features, it is able to distinguish it.

Table 6. Comparison to the state-of-the-art systems and the proposed approach

Best Model (Weighted-F1)	Comparison (Weighted-F1)
D1 (95.19)	[2] 92.20
D2 (82.78)	[26] 80.30
D3 (83.11)	[27] 77.97 , [27] 77.48
D4 (65.97)	[25] 60.81
D5 (82.14)	[46] 80.0
D6 (86.14)	[33] 84.11 , [33] 83.98
D7 (90.98)	[4] 89.50 , [4] 89.30
D8 (92.17)	[29] 80

5.2. Statistical Significance Test

We also determine whether a difference between the M-BERT in STL (M1) and Model 7 is statistically significant (at $p \leq 0.05$), for this we run a bootstrap sampling test on the predictions of two systems. The test takes 3 confusion matrix out of 5 at a time and compares whether the better system is the same as the better system on the entire dataset. The resulting (p-) value of the bootstrap test is thus the fraction of samples where the winner differs from the entire data set.

Table 7. Bootstrapping Test

Data	Sample taken	p-value
D1	60%	≤ 0.03
D2	60%	≤ 0.01
D3	60%	≤ 0.03
D4	60%	≤ 0.05
D5	60%	≤ 0.03
D6	60%	≤ 0.04
D7	60%	≤ 0.05
D8	60%	≤ 0.05

6. CONCLUSIONS AND FUTURE WORK

In this paper, we leverage a deep multi-task learning framework to leverage the useful information of multiple related tasks. To deal with the data scarcity problem we utilize a multi-task learning approach that enables the model by sharing representations between the related tasks and generalize better by achieving better performance for the individual tasks. Detailed empirical evaluation shows that the proposed multi-task learning framework achieves statistically significant performance improvement over the single-task setting.

We have leveraged the labeled corpora for each tasks and experimented on single task learning and multi-task learning paradigm. The plausible extensions include the inclusion of more affective phenomenon correlated to hate speech such as sarcasm/irony [53], "big five" personality traits [54], and emotion role labeling [55].

REFERENCES

- [1] Patchin, Justin W., and Sameer Hinduja. "Cyberbullying and Online Aggression Survey." (2015).
- [2] Bohra, Aditya, Deepanshu Vijay, Vinay Singh, Syed Sarfaraz Akhtar, and Manish Shrivastava. "A dataset of Hindi-English code-mixed social media text for hate speech detection." In *Proceedings of the second workshop on computational modeling of people's opinions, personality, and emotions in social media*, pp. 36-41. 2018.
- [3] Liu, Joseph. "Religious hostilities reach six-year high." (2014).
- [4] Mathur, Puneet, Ramit Sawhney, Meghna Ayyar, and Rajiv Shah. "Did you offend me? classification of offensive tweets in hinglish language." In *Proceedings of the 2nd workshop on abusive language online (ALW2)*, pp. 138-148. 2018.
- [5] Albadi, Nuha, Maram Kurdi, and Shivakant Mishra. "Are they our brothers? analysis and detection of religious hate speech in the arabic twittersphere." In *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pp. 69-76. IEEE, 2018.
- [6] Soliman, Abu Bakr, Kareem Eissa, and Samhaa R. El-Beltagy. "Aravec: A set of arabic word embedding models for use in arabic nlp." *Procedia Computer Science* 117 (2017): 256-265.
- [7] Ousidhoum, Nedjma, Zizheng Lin, Hongming Zhang, Yangqiu Song, and Dit-Yan Yeung. "Multilingual and multi-aspect hate speech analysis." *arXiv preprint arXiv:1908.11049* (2019).
- [8] Mulki, Hala, Hatem Haddad, Chedi Bechikh Ali, and Halima Alshabani. "L-hsab: A levantine twitter dataset for hate speech and abusive language." In *Proceedings of the third workshop on abusive language online*, pp. 111-118. 2019.
- [9] Chung, Yi-Ling, Elizaveta Kuzmenko, Serra Sinem Tekiroglu, and Marco Guerini. "CONAN--COUNTER NARRATIVES THROUGH NICHE-SOURCING: A MULTILINGUAL DATASET OF RESPONSES TO FIGHT ONLINE HATE SPEECH." *arXiv preprint arXiv:1910.03270* (2019).
- [10] Bretschneider, Uwe, and Ralf Peters. "Detecting offensive statements towards foreigners in social media." In *Proceedings of the 50th Hawaii International Conference on System Sciences*. 2017.
- [11] Wiegand, Michael, Melanie Siegel, and Josef Ruppenhofer. "Overview of the germeval 2018 shared task on the identification of offensive language." (2018): 1-10.
- [12] Sanguinetti, Manuela, Fabio Poletto, Cristina Bosco, Viviana Patti, and Marco Stranisci. "An italian twitter corpus of hate speech against immigrants." In *Proceedings of the eleventh international conference on language resources and evaluation (LREC 2018)*. 2018.
- [13] Bosco, Cristina, Dell'Orletta Felice, Fabio Poletto, Manuela Sanguinetti, and Tesconi Maurizio. "Overview of the evalita 2018 hate speech detection task." In *EVALITA 2018-Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian*, vol. 2263, pp. 1-9. CEUR, 2018.
- [14] Álvarez-Carmona, Miguel Á., Estefania Guzmán-Falcón, Manuel Montes-y Gómez, Hugo Jair Escalante, Luis Villasenor-Pineda, Verónica Reyes-Meza, and Antonio Rico-Sulayes. "Overview of MEX-A3T at IberEval 2018: Authorship and aggressiveness analysis in Mexican Spanish tweets." In *Notebook papers of 3rd sepln workshop on evaluation of human language technologies for iberian languages (ibereval), seville, spain*, vol. 6. 2018.
- [15] Ptaszynski, Michal, Agata Pieciukiewicz, and Paweł Dytała. "Results of the poleval 2019 shared task 6: First dataset and open shared task for automatic cyberbullying detection in polish twitter." (2019).
- [16] Ljubešić, Nikola, Tomaž Erjavec, and Darja Fišer. "Datasets of Slovene and Croatian moderated news comments." In *Proceedings of the 2nd workshop on abusive language online (ALW2)*, pp. 124-131. 2018.
- [17] Çöltekin, Çağrı. "A corpus of Turkish offensive language on social media." In *Proceedings of the 12th language resources and evaluation conference*, pp. 6174-6184. 2020.
- [18] Alfina, Ika, Rio Mulia, Mohamad Ivan Fanany, and Yudo Ekanata. "Hate speech detection in the Indonesian language: A dataset and preliminary study." In *2017 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, pp. 233-238. IEEE, 2017.
- [19] Bhat, Irshad Ahmad, Vandan Mujadia, Aniruddha Tammewar, Riyaz Ahmad Bhat, and Manish Shrivastava. "Iiit-h system submission for fire2014 shared task on transliterated search." In *Proceedings of the Forum for Information Retrieval Evaluation*, pp. 48-53. 2014.
- [20] Kumar, Ritesh, Aishwarya N. Reganti, Akshit Bhatia, and Tushar Maheshwari. "Aggression-annotated corpus of hindi-english code-mixed data." *arXiv preprint arXiv:1803.09402* (2018).

- [21] Aroyehun, Segun Taofeek, and Alexander Gelbukh. "Aggression detection in social media: Using deep neural networks, data augmentation, and pseudo labeling." In *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, pp. 90-97. 2018.
- [22] Arroyo-Fernández, Ignacio, Dominic Forest, Juan-Manuel Torres-Moreno, Mauricio Carrasco-Ruiz, Thomas Legeleux, and Karen Joannette. "Cyberbullying detection task: the ebsi-lia-unam system (elu) at coling'18 trac-1." In *Proceedings of the first workshop on trolling, aggression and cyberbullying (TRAC-2018)*, pp. 140-149. 2018.
- [23] Modha, Sandip, Prasenjit Majumder, and Thomas Mandl. "Filtering aggression from the multilingual social media feed." In *Proceedings of the first workshop on trolling, aggression and cyberbullying (TRAC-2018)*, pp. 199-207. 2018.
- [24] Golem, Viktor, Mladen Karan, and Jan Šnajder. "Combining shallow and deep learning for aggressive text detection." In *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, pp. 188-198. 2018.
- [25] Kumar, Ritesh, Atul Kr Ojha, Shervin Malmasi, and Marcos Zampieri. "Benchmarking aggression identification in social media." In *Proceedings of the first workshop on trolling, aggression and cyberbullying (TRAC-2018)*, pp. 1-11. 2018.
- [26] Mandl, Thomas, Sandip Modha, Gautam Kishore Shahi, Hiren Madhu, Shrey Satapara, Prasenjit Majumder, Johannes Schäfer et al. "Overview of the hasoc subtrack at fire 2021: Hate speech and offensive content identification in english and indo-aryan languages." *arXiv preprint arXiv:2112.09301* (2021).
- [27] Mandl, Thomas, Sandip Modha, Anand Kumar M, and Bharathi Raja Chakravarthi. "Overview of the hasoc track at fire 2020: Hate speech and offensive language identification in tamil, malayalam, hindi, english and german." In *Forum for information retrieval evaluation*, pp. 29-32. 2020.
- [28] Bashar, Md Abul, and Richi Nayak. "QutNocturnal@ HASOC'19: CNN for hate speech and offensive content identification in Hindi language." *arXiv preprint arXiv:2008.12448* (2020).
- [29] Jha, Vikas Kumar, P. Hrudya, P. N. Vinu, Vishnu Vijayan, and P. Prabakaran. "DHOT-repository and classification of offensive tweets in the Hindi language." *Procedia Computer Science* 171 (2020): 2324-2333.
- [30] Velankar, Abhishek, Hrushikesh Patil, Amol Gore, Shubham Salunke, and Raviraj Joshi. "Hate and offensive speech detection in Hindi and Marathi." *arXiv preprint arXiv:2110.12200* (2021).
- [31] Chopra, Shivang, Ramit Sawhney, Puneet Mathur, and Rajiv Ratn Shah. "Hindi-english hate speech detection: Author profiling, debiasing, and practical perspectives." In *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 01, pp. 386-393. 2020.
- [32] Santosh, T. Y. S. S., and K. V. S. Aravind. "Hate speech detection in hindi-english code-mixed social media text." In *Proceedings of the ACM India joint international conference on data science and management of data*, pp. 310-313. 2019.
- [33] Bhardwaj, Mohit, Md Shad Akhtar, Asif Ekbal, Amitava Das, and Tanmoy Chakraborty. "Hostility detection dataset in Hindi." *arXiv preprint arXiv:2011.03588* (2020).
- [34] Davidson, Thomas, Dana Warmusley, Michael Macy, and Ingmar Weber. "Automated hate speech detection and the problem of offensive language." In *Proceedings of the international AAAI conference on web and social media*, vol. 11, no. 1, pp. 512-515. 2017.
- [35] Waseem, Zeerak, and Dirk Hovy. "Hateful symbols or hateful people? predictive features for hate speech detection on twitter." In *Proceedings of the NAACL student research workshop*, pp. 88-93. 2016.
- [36] Zampieri, Marcos, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. "Predicting the type and target of offensive posts in social media." *arXiv preprint arXiv:1902.09666* (2019).
- [37] Golbeck, Jennifer, Zahra Ashktorab, Rashad O. Banjo, Alexandra Berlinger, Siddharth Bhagwan, Cody Buntain, Paul Cheakalos et al. "A large labeled corpus for online harassment research." In *Proceedings of the 2017 ACM on web science conference*, pp. 229-233. 2017.
- [38] Basile, Valerio, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. "Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter." In *Proceedings of the 13th international workshop on semantic evaluation*, pp. 54-63. 2019.
- [39] De Gibert, Ona, Naiara Perez, Aitor García-Pablos, and Montse Cuadros. "Hate speech dataset from a white supremacy forum." *arXiv preprint arXiv:1809.04444* (2018).

- [40] Founta, Antigoni Maria, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. "Large scale crowdsourcing and characterization of twitter abusive behavior." In *Twelfth International AAAI Conference on Web and Social Media*. 2018.
- [41] Bhattacharya, Shiladitya, Siddharth Singh, Ritesh Kumar, Akanksha Bansal, Akash Bhagat, Yogesh Dawer, Bornini Lahiri, and Atul Kr Ojha. "Developing a multilingual annotated corpus of misogyny and aggression." *arXiv preprint arXiv:2003.07428* (2020).
- [42] Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding." *arXiv preprint arXiv:1810.04805* (2018).
- [43] Khanuja, Simran, Diksha Bansal, Sarvesh Mehtani, Savya Khosla, Atreyee Dey, Balaji Gopalan, Dilip Kumar Margam et al. "Muril: Multilingual representations for indian languages." *arXiv preprint arXiv:2103.10730* (2021).
- [44] Zhang, Yu, and Qiang Yang. "A survey on multi-task learning." *IEEE Transactions on Knowledge and Data Engineering* (2021).
- [45] Fortuna, Paula, Joao Rocha da Silva, Leo Wanner, and Sérgio Nunes. "A hierarchically-labeled portuguese hate speech dataset." In *Proceedings of the third workshop on abusive language online*, pp. 94-104. 2019.
- [46] Kumar, Ritesh, Atul Kr Ojha, Shervin Malmasi, and Marcos Zampieri. "Evaluating aggression identification in social media." In *Proceedings of the second workshop on trolling, aggression and cyberbullying*, pp. 1-5. 2020.
- [47] Xue, Ya, Xuejun Liao, Lawrence Carin, and Balaji Krishnapuram. "Multi-Task Learning for Classification with Dirichlet Process Priors." *Journal of Machine Learning Research* 8, no. 1 (2007).
- [48] Ruder, Sebastian. "An overview of multi-task learning in deep neural networks." *arXiv preprint arXiv:1706.05098* (2017).
- [49] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." *Advances in neural information processing systems* 30 (2017).
- [50] Chollet, François. "Keras: The python deep learning library." *Astrophysics source code library* (2018): ascl-1806.
- [51] Abadi, Martín, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado et al. "Tensorflow: Large-scale machine learning on heterogeneous distributed systems." *arXiv preprint arXiv:1603.04467* (2016).
- [52] Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980* (2014).
- [53] Reyes, Antonio, Paolo Rosso, and Davide Buscaldi. "From humor recognition to irony detection: The figurative language of social media." *Data & Knowledge Engineering* 74 (2012): 1-12.
- [54] Flek, Lucie. "Returning the N to NLP: Towards contextually personalized classification models." In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pp. 7828-7838. 2020.
- [55] Mohammad, Saif, Xiaodan Zhu, and Joel Martin. "Semantic role labeling of emotions in tweets." In *Proceedings of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pp. 32-41. 2014.

AUTHORS

Prashant Kapil is a PhD scholar in the Department of CSE at IIT Patna. The author would like to acknowledge the funding agency, the University Grant Commission (UGC) of the Government of India for providing financial support in the form of UGC NET-JRF/SRF.

Research interests: AI, NLP, and ML



Asif Ekbal is an Associate Professor in the Department of CSE, IIT Patna, India.

Research interests: AI, NLP and ML.



© 2022 By AIRCC Publishing Corporation. This article is published under the Creative Commons Attribution (CC BY) license.

WASSBERT: HIGH-PERFORMANCE BERT-BASED PERSIAN SENTIMENT ANALYZER AND COMPARISON TO OTHER STATE-OF-THE-ART APPROACHES

Masoumeh Mohammadi and Shadi Tavakoli

Department of Data Science & Machine Learning Telewebion, Tehran, Iran

ABSTRACT

Applications require the ability to perceive others' opinions as one of the most outstanding parts of knowledge. Finding the positive or negative feelings in sentences is called sentiment analysis (SA). Businesses use it to understand customer sentiment in comments on websites or social media. An optimized loss function and novel data augmentation methods are proposed for this study, based on Bidirectional Encoder Representations from Transformers (BERT). First, a crawled dataset from Persian movie comments on various sites has been prepared. Then, balancing and augmentation techniques are accomplished on the dataset. Next, some deep models and the proposed BERT are applied to the dataset. We focus on customizing the loss function, which achieves an overall accuracy of 94.06 for multi-label (positive, negative, neutral) sentences. And the comparative experiments are conducted on the dataset, where the results reveal the performance of the proposed model is significantly superior compared with other models.

KEYWORDS

Bidirectional encoder representations from Transformers (BERT), Bidirectional long short-term memory (Bi-LSTM), Comment classification, Convolutional neural network (CNN), Deep learning, Opinion mining(OM), Natural language processing (NLP), Persian language sentiment classification, Persian Sentiment analysis, Text mining.

1. INTRODUCTION

Watching movies is probably one of the most popular activities worldwide, and streaming movies online makes it more convenient. Furthermore, no one wants to waste their time on a film that is not worth watching [1]. Therefore, the Internet plays a crucial role in expressing opinions and sharing experiences about different movies. The goal of natural language processing (NLP) is to build a machine capable of understanding the contents of documents, including the contextual nuances of the language within them [2]. Sentiment analysis (SA) or opinion mining (OM) is a technique to determine the emotional tone. The SA models focus on polarity (positive, negative, neutral), feelings and emotions (angry, happy, sad, etc.), urgency, and even intentions (interested vs. not interested) [3].

Besides, it is widely applied to product reviews, social media, healthcare materials, etc. Many enhancements to SA models have been proposed in the last few years. In the next Section, we summarize and categorize some articles presented in this field that use various SA models such as machine learning (ML) algorithms or deep learning approaches. This paper aims to propose a

technique to classify reviews about movies depending on the sentiment they express, e.g., “The movie is surprising” (positive review), “I do not like cartoons” (neutral review), and “Crap, Crap and totally crap. Did I mention this film was total crap? Well, it’s total crap” (negative review). Our main contributions to this study are as follows:

- The reliability of an SA solution depends highly upon obtaining sufficient data in Persian NLP. In this regard, the movie comments are crawled from several Persian websites. Then, data augmentation techniques are applied to the texts as described in Section 3 to generate additional and synthetic data.
- The next challenge is to deal with the imbalanced dataset complicated by the size, noise, and distribution. Most ML algorithms perform poorly and must be modified to prevent simply predicting the bulk of the data. Furthermore, metrics such as classification accuracy no longer make sense, and it is crucial to develop alternative techniques to evaluate predictions from imbalanced samples. Thus, several methods are performed to determine the best way to balance datasets; under-sampling appears most promising.
- Another foundational aspect of this study is the preprocessing phase, which among others, transforms comments, including emojis and emoticons, into plain text, using language-independent conversion techniques that are general and proper also to the Persian language.
- A customized list of stop words is devised to eliminate commonly used words. They carry very little helpful information, which improves the learning of the model keywords extracted as a reference for the global sentiment. Then, the attached label is transferred into Persian words as label embedding.
- We also conduct a comparative analysis of existing and proposed machine learning models and novel deep learning models regarding the recall, f1 score, precision, and accuracy.
- Our model adopts BERT-based word embedding to obtain each partial feeling and learn Persian sentences’ complex and changeable structures. Finally, we use a custom loss function which results in our method outperforming the traditional and state-of-the-art models.

This paper organizes as follows. Section 2 includes the related works and a summary of the articles. A brief survey of comparable and benchmark methods is presented in Section 3, followed by the structure of the proposed approach. Consequently, the experimental setup and the results and evaluation are given in Section 4. Section 5 concludes and discusses future research.

2. RELATED WORKS

Many methods have been developed and tested around SA. They can be categorized as follows:

technique-based, text-oriented, level-based, rating level, etc. Collomb et al. [4] compared different points of view. From a technical viewpoint, they identified ML, lexicon-based, statistical, and rule-based approaches:

- The ML techniques perform learning algorithms to find the sentiment by training on a specific dataset.
- The lexicon-based method calculates sentiment polarity for a comment using the semantic sense or the semantic orientation of words and phrases in the review [4].

- The rule-based approach considers opinion words in content and then sorts them based on the number of positive and negative comments [4].
- Statistical models show reviews as a combination of hidden sights and ratings.

Another categorization based on the text structure includes document level, sentence level, or word/feature level classification. Reference [4] revealed that most techniques centralize a document-level classification. Also, the most current methods can identify sentiment strength for various aspects of a product/service and processes that intend to rate a review on a global level. Figure 1 depicts these details.

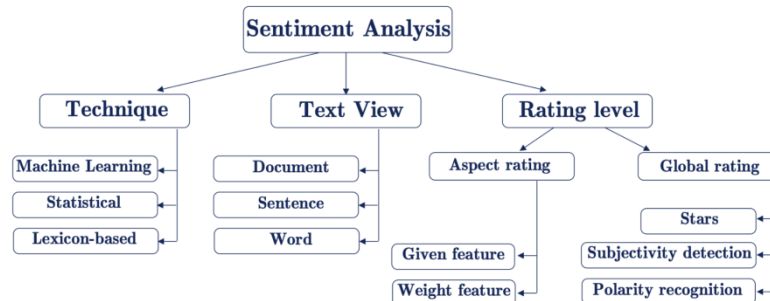


Figure 1. Types of sentiment classification, an overview of the classification techniques that have been used to answer the sentiment analysis questions.

Huifeng and Songbo [5] define several problems related to sentiment detection and discuss its different applications. They introduce semantic-based techniques, present ML methods, and mention two classification forms: binary (negative and positive) and multi-class (negative, neutral, and positive) sentiment classification.

Jakob & Gurevych [6] have focused on opinion extraction based on conditional random field (CRF). They apply a supervised methodology to a “movie review”. Toprak et al. [4] offer a scheme of annotation which contains two levels: sentence level and expression level. M. Hajmohammadi and R. Ibrahim [8] perform some ML techniques on a dataset of online Persian movie reviews to automatically classify them as either positive or negative. On this supervised classification task, they attain up to 82.9% accuracy.

Meanwhile, F. Amiri et al. [9] manually created a lexicon with sentiment scores and some rules on hand-coded grammar due to existing complexity, such as specific features, wrapped morphology, and the context-sensitivity of the script in the Persian language. They designed and developed a linguistic pipeline based on the framework and graphical development environment for robust NLP applications and named it GATE [10]. Their evaluation of the GATE pipeline reveals its overall accuracy of 69%.

González et al. [11] design the BERT emotion detection tasks for TASS 2020 (an Albert-like model). It turns the highest accuracy in almost all the Spanish variants at three levels. Then Palomino and Ochoa [12] obtain the second-best result based on the BERT model. They apply an additional step of unsupervised data augmentation to improve their previous results for most variants of the Spanish language.

In [13], for Persian movie reviews, the deep learning model achieves 82.86% accuracy using the CNN model, obtaining significantly better results compared with previous models. Study [14] manually creates sentiment seeds to determine the polarity of a new lexicon. Their best accuracy

is 81%. The proposed bidirectional LSTM network learning in [15] is considered the state-of-the-art model in Arabic SA. Their work improvement was 2.39% on average on the utilized datasets.

Amiri et al. [9] achieve an accuracy of 69% by SVM classifier for developing a lexicon to detect polarity on multi-domain products and movie reviews in Persian. Alimardani et al. [16] further improve this idea by proposing approaches that collected hotel reviews using an SVM classifier, achieving an accuracy up to 85.9%. Dos et al. [17] created a CharSCNN with two convolution layers to extract features and address SA. Wang et al. [18] developed a model based on LSTM to predict the sentiment polarities of tweets by composing word embeddings. Wu Xing et al. [19] demonstrated the subjective characteristics of the stock market by gated recurrent unit (GRU).

Nevertheless, the RNN can not be used in parallel calculations because of developing a gradient explosion. Vaswani et al. [20] offer a transformer to solve this problem and gain sustainable results in many NLP applications, including SA. Catelli et al. [21] use a multi-lingual technique based on BERT, performed a Named Entity distinction task for de-identification. Yu et al. [22] perform a BERT model to get state-of-the-art ancient Chinese sentence segmentation results.

3. Methodology

3.1. SVM

SVMs are the supervised learning methods for classification, regression, and outlier detection. This article uses an SVM from the Scikit-learn library as the first proposed model. It has been shown that the implementation of Gaussian kernels for SA is more performant than other nonlinear kernels.

3.2. BI-LSTM

We preprocess our dataset before feeding it to BI-LSTM. First, we normalize all comments using the Hazm normalizer [23]. The process of normalizing tokens returns them to their original form. Second, we separate each sentence into meaningful unit forms such as words, phrases, or subwords using the Keras tokenizer. Meanwhile, Hazm lemmatization is employed to merge two or more words into one by removing stop words from the penalties. The purpose of this step is to restore the roots of words or lemma, like می روم converted to رفت #رو. The Word2Vec training process vectorizes texts to help the system learn them. Fast-text is an NLP library developed by Facebook to use classification and word embedding [24]. Gensim Fast-text supports 157 languages. For the LSTM-based Persian SA, BI-LSTM is applied for the multi-label classification of movie reviews.

Since textual data are categorical variables, we need to convert them into numbers to feed the model [30]. One-hot encoding is an option to convert them into numbers. However, this approach is not viable due to its high memory demand. Meanwhile, the embedding layer is applied here to convert a word into a vector shape in multidimensional space and create a fixed-length vector to increase model efficiency. By using the max-pooling and dropout layer, we avoid overfitting problems. Global max-pooling reduces the dimension of the feature maps detected anywhere in this filter. For building the model, we compile the model with categorical cross-entropy loss function and Adam optimization. The model contained 5,535,003 trainable parameters. With 20 epochs, we run the BI-LSTM model and achieve the best mean accuracy of %87.01.

3.3. CNN

A CNN can extract multidimensional features (nonlinear features) without considering the probability of occurrence. There are 100 filters with a kernel size of 4, so each filter looks at a window of 4-word embeddings. It normalizes the previous layer's activation at each batch (batch normalization) by applying a transformation that maintains the mean activation close to zero and the activation standard deviation close to one [31]. After the activation function, a max-pooling layer is added.

3.4. BERT

Bidirectional encoder representations from transformers equip dense vector representations for NLP by using a deep, pre-trained neural network with the transformer architecture [16]. The original English language BERT has two models [16]:

1. the BERT-base: 12 encoders with 12 bidirectional self-attention heads.
2. the BERT-large: 24 encoders with 16 bidirectional self-attention heads.

There are also some other BERT models available:

- Small BERT: this model is a sample of the original BERT with a smaller number of layers [25].
- ALBERT: this is the “A Lite” version of BERT in which some of the parameters are reduced.
- BERT experts: setting off on a pre-trained BERT model and fine-tuning the downstream role produces efficient NLP tasks. It can increase the performance by starting from the BERT model that better aligns or transfers to the task at hand [32]. This collection is called “BERT expert” trained on different datasets and functions to perform better downstream tasks like SA, question answering, and all jobs requiring natural language inference skills.
- Electra: This is a pre-trained BERT-like model that plays a role as a discriminator in a setup resembling a generative adversarial network (GAN).
- ParsBERT: This model is pre-trained on large Persian corpora with more than 3.9M documents, 73M sentences, and 1.3B words [26].

3.4.1. The proposed BERT-based Model

We base our model on ParsBERT [26]. In this regard, ‘HooshvareLab/BERT-FA-Base-uncased’ was created, including 12 hidden layers and 12 attention heads. One dropout and a linear classifier with 768 hidden sizes. The whole model is displayed in Fig.2 Moreover, the parameters used in the proposed model are summarized in Table 1:

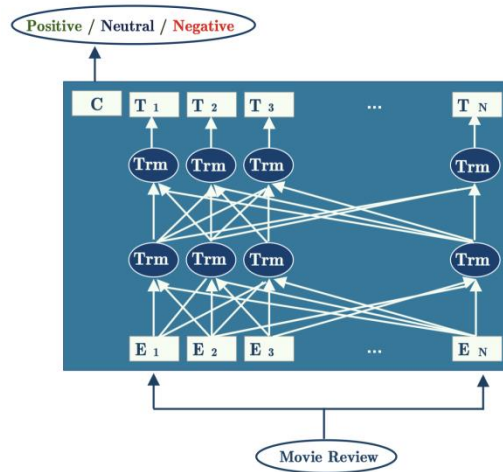


Figure 2. The BERT performs DL-based NLP tasks. It provides a model to understand the semantic meaning using NLP. The model uses movie comments as input and determines whether they are positive, neutral, or negative.

Table 1. The hyper-parameters that affect our purpose (feature importance) are empirically tested. Our experiments suggest that population-based training is the most efficient method for tuning the transformer model's hyper-parameters.

Parameter	Value
Epochs	3
Learning rate	2E-05
Train-batch-size	16
Valid-batch-size	16
Test-batch-size	16

3.4.2. Text classification using BERT

The following steps are followed in this investigation: Set up the Adam optimizer from transformers. Import and preprocess the dataset: The comments have different lengths. Detecting the most normal range could help us find the maximum length of the sequences for the preprocessing step. Create a BERT tokenizer: Tokenization separates a sentence into individual words. Besides, the inputs (users' movie reviews and comments) must be changed to numeric token ids and arranged in tensors before inputting to BERT [25]. It is a pre-trained model that has its input data format. Its structure contains two parts:

- The BERT summarizer that includes a BERT encoder and a summarizing classifier,
- The BERT classifier.

Figure 3 depicts both summarizing sectors.

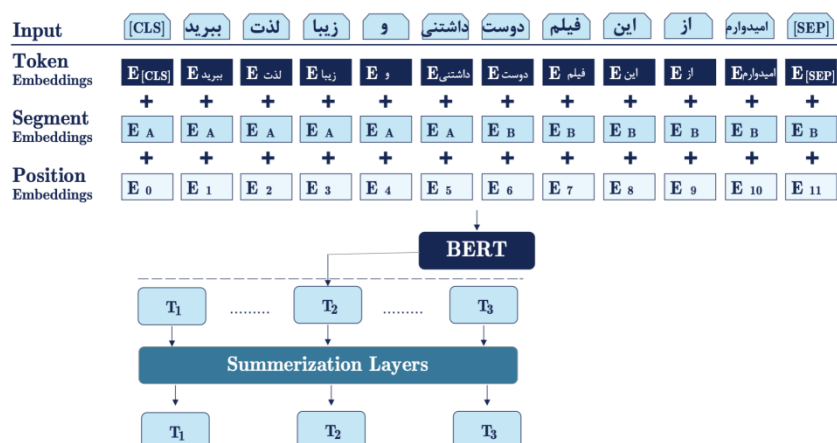


Figure 3. The BERT makes multiple embedding by a word to detect and report the content. Its input embedding includes the token, segment, and position components. The encoder gains the knowledge of interactions between tokens in the context, while the summarizing classifier learns the interactions between sentences.

The encoder learns the interactions among tokens in the document, while the summarization classifier learns the interactions among sentences. The BERT classifier has input and output. As figure 3 illustrates, [CLS] and [SEP] tokens separate two parts of the input. Each sentence is modeled as a sequence where the [CLS] token shows the beginning and [SEP] is a token to separate a sequence from a subsequent one [25]. After splitting words into tokens and converting the list of strings into a vocabulary index list, i.e., output for classification, we use the outcome of the first token, i.e., the [CLS] token. For more complicated results, we can use all the other token outputs. Figure 4 shows three outputs from the preprocessing that a BERT model would use. After this step, data is ready to convert to torch tensors and input to the BERT model. Figure 4 details the process: For NLP models to function, they need input in numerical vectors. Therefore, part of the process involves translating features such as vocabulary and parts of speech into numerical representations. Words can either be presented as uniquely indexed values (one-hot encoding) or as results from models such as Word2Vec or Fast-text, which match words with fixed-length feature embeddings. Each word has a fixed representation in these techniques regardless of the context; the words around them dynamically inform BERT representations of words. For example, consider the following two sentences: 1) “آخرش که تیتراژ” “پایانی تمام میشه خیلی خنده داره”, which means: the ending is funny, and 2) “خنده ام میگیره” which means: it makes me laugh. Word2Vec produces the same word embedding for the word “خنده”(meaning laugh) in both sentences, while BERT’s word embedding for “خنده” would be different for each sentence. In addition to taking apparent differences such as polysemy, the context-informed word embeddings capture other forms of information that result in more accurate feature representations, making a better conclusion in model performance [27]. We use this advantage of BERT and some data augmentation techniques to increase the accuracy of this study. The dataset was divided into 22829 training, 2537 validation, and 2819 test sentences.



Figure 4: Text inputs need to be transformed into numeric token ids and ordered in multiple tensors before being fed into the BERT; tokenization refers to assigning a sentence to single words.

There were three sentiment labels in the dataset (positive, negative, and neutral). The sample dataset is given in Table 2 below.

Table 2. The dataset contains 30000 user reviews which are balanced. A third of it owns negative comments labeled (-1), one third has the positive comments with the label (1), and the last part includes the neutral reviews tagged by (0).

Comments	Sentiment
بهترین فیلمی که تا حالا دیدم	1
شروع ضعیفی داشت امیدوارم در ادامه بهتر بشه	-1
تو رو خدا دوبله کنید	0

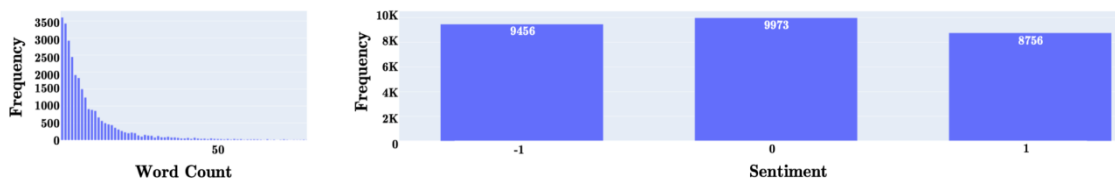


Figure 5. This chart illustrates the placement of the dataset before balancing. Balancing can be performed by over-sampling, under-sampling, class weight, or threshold. We use the under-sampling method to balance the dataset.

For the imbalanced dataset, two methods are applied: over-sampling and under-sampling. We observed better predictions in all deep models using the under-sampling technique [25]. Moreover, the clean dataset is augmented in two ways: random insertion and random swapping. The types of distribution of comments are demonstrated in Figs. 5.

The random swap does not work well in models due to existing particular characteristics in Persian, such as informal and conversational words, declension suffixes, various writing types, and word spacing. As a result, these traits affect Persian text accuracy. We also empirically observed that a delicately-crafted combination of Wasserstein and cross-entropy loss functions

would result in significantly better model training. Consider the $X = \{x_0, x_1, \dots, x_n\}$ to denote the possible outcomes or categories from the discriminator. Also, suppose $p : X \rightarrow [0, 1]$ and $q : X \rightarrow [0, 1]$ respectively denote the distributions for predicted and target values. The cross-entropy loss function (CLF) is then defined by:

$$H(p, q) = - \sum_{i=0}^n p(x_i) \log(q(x_i)) \quad (1)$$

It is widely adopted as the loss function and a metric for the performance of classifiers. Recently, the Wasserstein metric has been showing excellent results, particularly in generative adversarial networks (GANs). This approach is often based on the Wasserstein-1 or Earth mover distance (EMD) between the two distributions, which basically measures the amount of mass needed to be transported to convert one distribution to another. Based on our notation, this distance is defined by:

$$W(p, q) = \inf_{\gamma \in \Pi(p, q)} E_{(x, y) \sim \gamma} [\|x - y\|] \quad (2)$$

where $\Pi(p, q)$ is the set of all joint distributions having p and q as their marginal distributions. It can be shown using the Kantorovich-Rubinstein duality that this metric can be transformed to simply calculating the mean of a classifier's output [29]. Here, we propose to linearly combine the cross-entropy and Wasserstein loss function. The final loss function is of the form:

$$Finalloss = H(p, q) + \lambda W(p, q) \quad (3)$$

where the λ is the combination coefficient, which can be considered as a hyper-parameter. We empirically observed that the $\lambda = n$ would be a good choice and set it to 3 for all our experiments. Table 3 below compares the results:

Table 3. The loss functions' comparison; combining cross-entropy and Wasserstein grants the best prediction compared with other Persian studies.

Loss function	Train-loss	Train-acc	Valid-loss	Valid-acc
Cross entropy	0.0373	0.989	0.284	0.927
Cross entropy+Wasserstein	0.0317	0.991	0.219	0.940

4. EXPERIMENTAL RESULTS

Adjust the learning rate to about $2e^{-5}$ over three epochs. Using 16 GB of RAM and a Samsung SSD 870 500GB under the Ubuntu 64-bit operating system, the model was implemented in 11 minutes and 34 seconds under SA using an Intel Core i7 3.80GHz CPU with 16 GB of RAM. We developed machine learning and artificial intelligence projects with the visual intelligence model and Python libraries such as Numpy, Pandas, and Scikit-learn in Python 3.8.10. The data was collected from Persian movie review websites over 80 days, from 20 January 2020 to 25 April 2020.

4.1. Classifiers' measurement of this study

Different techniques are employed to extract the features from the movie reviews, and several opinions are applied to label the sentiments in the sentences. A balancing and augmentation method is used to carry the resulting dataset out, and each affected accuracy individually. As a result of completing various classifiers, performance metrics such as precision, recall, f1-measure, and accuracy [12] are calculated and reported in Fig. 6 and table 4. This figure reveals that BERT results are significantly higher than other algorithms.

Table 4. We compare several metrics to decide if a model performs well. The table shows the final result; the WassBERT gained the best scores.

Model	Accuracy	Recall	F1-score	Precision
SVM	75.5	75.5	75.5	80
BI-LSTM	87.01	82	85	82
CNN	83.38	80	81	83
WassBERT	94	93	93	93

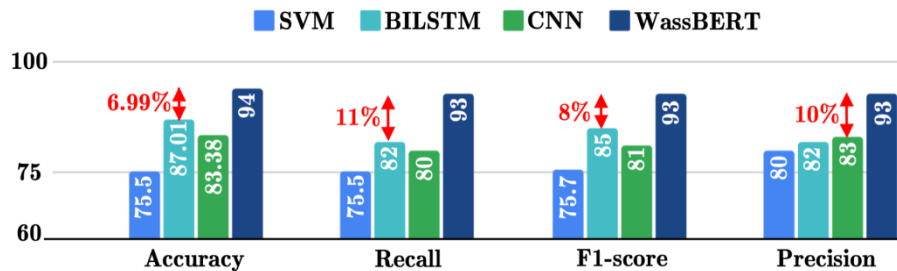


Figure 6. A comparison of WassBERT's performance metrics with those of other machine learning algorithms and deep models can be seen here where WassBERT is comparable to other Machine Learning algorithms and deep models in terms of its performance metrics. BERT yields an accuracy of 88.48 percent.

5. CONCLUSIONS

It is essential to recognize the sentiment of a movie comment in online reviews. However, the available Persian datasets are limited, and the existing models need to be improved. The proposed BERT model with a combination of Wasserstein and cross-entropy loss function is proved to achieve the best performance for the gathered Persian movie comments dataset. In a competitive study of deep learning models, proposed BERT's performance stands out (94%) among the deep learning models.

In future work, we address the dataset development in low resource languages, the balancing techniques, and augmentation methods that affect the model accuracy. We can also use explainable AI to Persian datasets with leading companies' data. Due to the lack of previous work on Persian datasets, our work cannot be compared to any previous ones and can now serve as a baseline for future work in this field.

REFERENCES

- [1] Movie reason. [Online]. Available: <https://www.everymoviehasalesson.com/blog/2021/9/4-reasons-to-read-movie-reviews>
- [2] The evolution of Natural Language Processing and its impact on the legal sector. [Online]. Available: <https://www.lexology.com/library/detail.aspx?g=0facd988-1702-4850-92e2-2f4cd25ab9db>
- [3] Sharma, Ritu; Gulati, Sarita; Kaur, Amanpreet; and Chakravarty, Rupak, (2021) "Users' Sentiment Analysis toward National Digital Library of India: a Quantitative Approach for Understanding User perception". *Library Philosophy and Practice (e-journal)*. 6372.
- [4] A. Collomb, C. Costea, D. Joyeux, O. Hasan, and L. Brunie, (2014) "A study and comparison of sentiment analysis methods for reputation evaluation," *Rapport de recherche RR-LIRIS-2014-002*.
- [5] H. Tang, S. Tan, and X. Cheng, (2009) "A survey on sentiment detection of reviews." *Expert Systems with Applications*, vol. 36, no. 7, pp. 10 760–10 773.
- [6] N. Jakob and I. Gurevych, (2010) "Extracting opinion targets in a single-and cross-domain setting with conditional random fields. *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*," pp. 1035–1045.
- [7] C. Toprak, N. Jakob, and I. Gurevych, (2010) "Sentence and expression level annotation of opinions in user-generated discourse. *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*," pp. 575–584.
- [8] M. S. Hajmohammadi and R. Ibrahim, (2013) "An SVM-based method for sentiment analysis in Persian language. *International Conference on Graphic and Image Processing (ICGIP)*," vol. 8768, p.876838.
- [9] F. Amiri, S. Scerri, and M. Khodashahi, (2015) "Lexicon-based sentiment analysis for Persian text. *Proceedings of the International Conference Recent Advances in Natural Language Processing*," pp. 9–16.
- [10] H. Cunningham, (2002) "GATE: A framework and graphical development environment for robust NLP tools and applications. *Proc. 40th Annual Meeting of the Association for Computational Linguistics (ACL)*," pp. 168–175.
- [11] J. Á. González-Barba, J. Arias-Moncho, L. F. Hurtado Oliver, and F. Pla Santamaría, (2020) "Elirf-upv at tass: Twilbert for sentiment analysis and emotion detection in spanish tweets. *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2020)*," pp. 179–186.
- [12] D. Palomino and J. O. Luna, (2020) "Palomino-Ochoa at TASS 2020: Transformer-based Data Augmentation for Overcoming Few-Shot Learning. *IberLEF@ SEPLN*," pp. 171–178.
- [13] K. Dashtipour, M. Gogate, J. Li, F. Jiang, B. Kong, and A. Hussain, (2020) "A hybrid Persian sentiment analysis framework: Integrating dependency grammar-based rules and deep neural networks." *Neurocomputing*, vol. 380, pp. 1–10.
- [14] N. Sabri, A. Edalat, and B. Bahrak, (2021) "Sentiment Analysis of Persian-English Code-mixed Texts. *2011 26th International Computer Conference, Computer Society of Iran (CSICC)*," pp. 1–4.
- [15] H. Elfaik et al., (2021) "Deep bidirectional lstm network learning-based sentiment analysis for Arabic text." *Journal of Intelligent Systems*, vol. 30, no. 1, pp. 395–412.
- [16] S. Alimardani and A. Aghaie, (2015) "Opinion mining in Persian language using supervised algorithms. *Journal of Information Systems and Telecommunication (JIST)*,".
- [17] C. Dos Santos and M. Gatti, (2014) "Deep convolutional neural networks for sentiment analysis of short texts. *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*," pp. 69–78.
- [18] X. Wang, Y. Liu, C.-J. Sun, B. Wang, and X. Wang, (2015) "Predicting polarities of tweets by composing word embeddings with long short-term memory. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*," pp. 1343–1353.
- [19] Wu, Xing and Chen, Haolei and Wang, Jianjia and Troiano, Luigi and Loia, Vincenzo and Fujita, Hamido, (2020) "Adaptive stock trading strategies with deep reinforcement learning methods., *Information Sciences*, vol. 538, pp. 142–158.
- [20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser and I. Polosukhin, (2017) "Attention is all you need." *Advances in Neural Information Processing Systems. 31st Conference on Neural Information Processing Systems (NIPS)*, vol. 30.

- [21] R. Catelli, F. Gargiulo, V. Casola, G. De Pietro, H. Fujita, and M. Esposito, (2020) “Cross-lingual named entity recognition for clinical de-identification applied to a covid-19 Italian data set,” *Applied Soft Computing*, vol. 97, p. 106779.
- [22] J. Yu, Y. Wei, and Y. Zhang, (2019) “Automatic ancient Chinese texts segmentation based on BERT.” *Journal of Chinese Information Processing*, vol. 33, no. 11, pp. 57–63.
- [23] Sobhe. Hazm. [Online]. Available: <https://www.sobhe.ir/hazm>
- [24] Facebook. Fasttext. [Online]. Available: <https://www.fasttext.cc>
- [25] J. D. M.-W. C. Kenton and L. K. Toutanova, (2019) “Bert: Pre-training of deep bidirectional transformers for language understanding. Proceedings of NAACL-HLT,” pp. 4171–4186.
- [26] M. F. M. M. Mehrdad Farahani, Mohammad Gharachorloo, (2019) “Parsbert: Transformer-based model for Persian language understanding,” *Neural Processing Letters*.
- [27] K. K. Mnih A, (2013) “Learning word embeddings efficiently with noise-contrastive estimation. Proceedings of the Annual Conference on Advances in Neural Information Processing Systems (NIPS).”
- [28] S. I. C. K. C. G. D. J. Mikolov, T., (2013) “Distributed representations of words and phrases and their compositionality. Proceedings of the 26th International Conference on Neural Information Processing Systems (NIPS), vol. 2, pp. 3111–3119. Curran Associates Inc., Lake Tahoe.”
- [29] M. Arjovsky, S. Chintala, and L. Bottou, (2017) “Wasserstein generative adversarial networks.” in *International conference on machine learning*. PMLR, pp. 214–223.
- [30] Data Handling. [Online]. Available: <https://towardsdatascience.com/data-handling-using-pandas-machine-learning-in-real-life-be76a697418c>
- [31] Gokhan Ciflikli, (2018) “Learning Conflict Duration: Insights from Predictive Modelling.” A thesis submitted to the International Relations Department of the London School of Economics for the degree of Doctor of Philosophy.
- [32] Bert Expert. [Online]. Available: https://www.tensorflow.org/hub/tutorials/bert_experts

AUTHORS

Masoumeh Mohammadi is the Co-Founder of Thumb Zone, a mobile usability testing platform company. A data scientist and application developer with over ten years of experience working with leading companies in social media, e-commerce, and online TV activities. She graduated with an M.S.C. in Artificial Intelligence and a B.S. in software engineering. Her interests include computer vision, natural language processing, and recommendation systems.



Shadi Tavakoli earned a Bachelor of Science in Electrical Engineering from Bu-Ali Sina University, Hamedan, Iran, in 2016. Currently, she is studying at the Islamic Azad University Central Tehran Branch for her Master's degree. Deep learning, natural language processing, and recommender systems are among her current research interests.



GRASS: A SYNTACTIC TEXT SIMPLIFICATION SYSTEM BASED ON SEMANTIC REPRESENTATIONS

Rita Hijazi^{1, 2}, Bernard Espinasse¹ and Núria Gala²

¹Aix-Marseille Univ.,
Laboratoire Informatique et Systèmes (LIS UMR 7020), Marseille, France
² Aix-Marseille Univ., Laboratoire Parole et Langage (LPL UMR 7309),
Aix-en-Provence, France

ABSTRACT

Automatic Text Simplification (ATS) is the process of reducing a text's linguistic complexity to improve its understandability and readability while maintaining its original information, content, and meaning. Several text transformation operations can be performed such as splitting a sentence into several shorter sentences, substitution of complex elements, and reorganization. It has been shown that the implementation of these operations essentially at a syntactic level causes several problems that could be solved by using semantic representations. In this paper, we present GRASS (GRaph-based Semantic representation for syntactic Simplification), a rule-based automatic syntactic simplification system that uses semantic representations. The system allows the syntactic transformation of complex constructions, such as subordination clauses, appositive clauses, coordination clauses, and passive forms into simpler sentences. It is based on graph-based meaning representation of the text expressed in DMRS (Dependency Minimal Recursion Semantics) notation and it uses rewriting rules. The experimental results obtained on a reference corpus and according to specific metrics outperform the results obtained by other state of the art systems on the same reference corpus.

KEYWORDS

Syntactic Text Simplification, Graph-Based Meaning Representation, DMRS, Graph-Rewriting.

1. INTRODUCTION

Automatic Text Simplification (ATS) transforms a complex text into an equivalent version that would be easier to read and/or understand by a target audience without significantly changing the input original meaning [1]. Simplification has been shown useful both as a pre-processing step for Natural Language Processing (NLP) tasks such as machine translation [2], relation extraction [3], text summarization [4], and for developing reading aids, e.g., for people with dyslexia [5], individuals with low vision [6], or non-native speakers [7]. Traditionally, two different tasks are considered in ATS: *lexical simplification* and *syntactic simplification*. Roughly speaking, *lexical simplification* (LS) consists of complex word identification and substitution by a simpler synonym or adding definitions. *Syntactic simplification* (SS) aims to transform sentences containing syntactic constructions that may hinder readability and comprehension into more readable or understandable equivalents. Several text transformation operations can be performed such as *division*, consisting of splitting a sentence into multiple shorter sentences, *deletion*, *reorganization*, and *morpho-syntactic substitutions*.

In this paper, we present GRASS (GRaph-based Semantic representation for syntactic Simplification), an automatic syntactic simplification system, and we focus on sentence splitting and passive to active voice transformations using graphs as semantic representations¹. GRASS implements a specific syntactic simplification method based on rewriting rules that exploit a semantic representation. This semantic representation of the text is expressed in Dependency Minimal Recursion Semantics notation (DMRS) [8]. Both semantic and syntactic information are expressed in the text, which simplifies the splitting operation. The simplification process in GRASS is done according three steps: (i) semantic representation of the complex sentence by a DMRS graph; (ii) transformation of this DMRS graph into one or several DMRS graphs by applying a set of transformation rules; and (ii) generation of simplified sentence(s) from the transformed DMRS graph(s).

GRASS system is automatically evaluated on the HSplit corpus [9] according to a set of reference metrics (BLEU, SARI, SAMSA) used in automatic text simplification. We compare the results obtained with GRASS with two state-of-the-art syntactic semantic-based simplification systems, HYBRID [10] and DSS [11]. We show that our system outperforms both HYBRID and DSS in syntactic simplification of the targeted structures.

The paper is organized as follows: section 2 introduces ATS main current approaches, with a special focus on semantic-based ATS systems. Section 3 presents GRASS, its theoretical foundations, and its software architecture. The experimental setup is detailed in section 4. Section 5 presents the results obtained by our tool, that we compare with the results obtained by other systems on the same reference corpus. We finally conclude with some perspectives of this work.

2. RELATED WORK

In this section we first present some mainstream approaches of automatic text simplification, and we then focus on semantic-based syntactic simplification.

2.1. Automatic Text Simplification

Text simplification mainly concerns two main linguistic levels of simplification: lexical and syntactic. To perform these simplifications, three main approaches can be identified: *rule-based approaches*, *machine learning-based approaches*, and a combination of both, known as *hybrid approaches*.

Rule-based approaches were the first to appear. Concerning syntactic simplification, specific hand-crafted sentence splitting rules were first proposed by [12] and [13]. Rule-based approaches are generally used for specific applications and for a well-targeted populations [14][15]. They rely on a study of corpora to identify linguistic phenomena affecting readability or comprehensibility. The idea here is to isolate a set of complex structures, and to create transformation rules to paraphrase. According to [16], manual rules are used in the field of text simplification when a system focuses on very specific linguistic structures and phenomena that are relatively easy to manage with a limited set of rules. However, their compilation and validation are laborious [17], i.e., they require expert human involvement and lead to linguistically accurate simplification systems.

In many cases, syntax transformation rules are implemented using synchronous grammars [18], which specify transformation operations between syntax trees using many rules. For example,

¹ The system code and results can be found on GitHub: <https://github.com/RitaHijazi/Semantic-based-Text-Simplification>

[19] used 111 rules for appositions, subordination, coordination, and relative clauses. [20] presented a rule-based system to automatically simplify Brazilian Portuguese text for people with low literacy. They proposed a set of operations to simplify 22 syntactic constructions. [14] followed a similar approach for French syntactic simplification, using manually constructed rules based on a typology of simplification rules manually extracted from a corpus of simplified French. [21] described a simplification of Spanish text that can simplify relatives, coordination, and participles. These rule-based systems often face several problems when dealing with long sentences, e.g., identifying the splitting points, rewriting shared elements, and deleting verb arguments which are needed for comprehension [10].

Machine Learning-based approaches, also called *corpus-based approaches*, have more recently been proposed in search of more robustness and coverage and to reduce the human involvement of the previous approach. The ATS systems developed based on these approaches generally use deep learning techniques (neural networks and word embeddings) and exploit large parallel corpora, i.e., original texts having simpler variants, e.g., Newsela [22] [23] and Wikipedia-Simple English Wikipedia [24] [25].

These approaches mainly consider the simplification task as a monolingual variant of a machine translation (MT) task. However, most of the simplified sentences are very similar to the complex sentence, and as such they are not suitable for the evaluation of full-fledged sentence simplification systems performing more complex sentence splitting and rewriting operations. That's why these models do not address sentence splitting.

The ATS systems developed according to this approach are generally efficient for lexical simplification but still present important limitations for syntactic simplification. The main drawback of these approaches is that the simplifications are not straightforwardly interpretable to humans (these models are often called 'black boxes') which can undermine trust in those models when it comes to evaluation of the results (i.e., when parallel corpora are not big enough).

Hybrid approaches try to take advantage of the benefits of the two previous approaches, mostly by combining rule-based syntactic simplifications, and lexical simplifications with learning-based approaches [10][11][26]. However, in this combination, to resolve limitations of rule-based systems for syntactic simplification, syntactic structures do not always capture the semantic arguments of a frame, which may result in wrong splitting boundaries [10]. To solve this problem, the authors working on hybrid approaches have proposed to take advantage of the semantic structures for sentence division.

2.2. Semantic-Based Syntactic Simplification

To our knowledge, [10] [26] are the first to propose to use semantic structures for sentence division in syntactic simplification. The operations of division and deletion are driven by semantics: the division is determined by the semantic roles that are associated with an element while the deletion of a node is determined by its semantic relationships with the divided events. Hence, their deletion model distinguishes between arguments and modifiers using a small number of rules. [10] proposed HYBRID, a supervised system that uses semantic structures, the Discourse Representation Structure [28] for sentence splitting and deletion. Splitting candidates are pairs of event variables associated with at least one core thematic role (e.g., agent or patient). Semantic annotation is used on the source side in both training and test of the system.

A little later, [26] proposed an unsupervised pipeline, where sentences are split based on a probabilistic model trained on the semantic structures of Simple Wikipedia, as well as a language model trained on the same corpus. [29] proposed the Split and Rephrase task, focusing on

sentence splitting. For this purpose, they presented a specialized parallel corpus, derived from the WebNLG dataset [30]. The latter is obtained from the DBpedia knowledge base [31] using content selection and crowdsourcing. It is annotated with semantic triplets of subject-relation-object, obtained semi-automatically.

More recently, [11] have combined structural semantics with rules for syntactic simplification and neural methods for lexical simplification. They presented Direct Semantic Splitting (DSS), an algorithm (based on rules) using a semantic parser which supports the direct decomposition of the sentence into its main semantic constituents. They use the UCCA semantic notation for semantic representation of the sentence [32]. UCCA aims to represent the main semantic phenomena in the text, without taking into consideration the syntactic forms. After splitting, NMT-based simplification system [33] is performed for lexical simplification.

While taking into account semantics is paramount, a system that would be only based on semantics does not seem appropriate for syntactic simplification. The argument-predicate relation is not enough to detect all the syntactic structures, both semantic and syntactic information are needed. Our research adopts the same approach as [11], focusing on syntactic simplification. It is based on a rule-based approach, but it uses the DMRS notation, which unlike UCCA combines the semantic and the syntactic representation of a sentence.

3. THE GRASS SYSTEM

GRASS for GRAPh-based Semantic representation for syntactic Simplification, is a rule-based automatic syntactic simplification system that uses semantic representations. It allows the syntactic transformation of complex sentences with syntactic constructions, such as subordination clauses, appositive clauses, coordination clauses and transformation from passive to active form into simpler constructions. As GRASS performs only syntactic simplification, as HYBRID and DSS systems do, it can be coupled with existing lexical simplification systems such as neural systems NTS [33].

In this section, we first present GRASS theoretical foundations, particularly the DMRS semantic graph representation and the DMRS-based simplification method. We then describe the GRASS software architecture with its components. We finally present a simplification example of appositive sentence transformed with GRASS.

3.1. GRASS Theoretical Foundations

GRASS uses the DMRS scheme for semantic representation [8]. DMRS differs from UCCA and DRS respectively used by [11] and [10] in the way the information is expressed. DMRS semantics are rooted in the superficial form of sentences and in the syntactic links between constituents. DMRS, as most semantic representations, rely on syntactic analyses: there is a strong overlap between semantic and syntactic constituents. DMRS semantics are anchored in the surface form of the sentences and in the syntactic links between the constituents. Syntactic information is explicitly marked, e.g., subordination, apposition, etc.

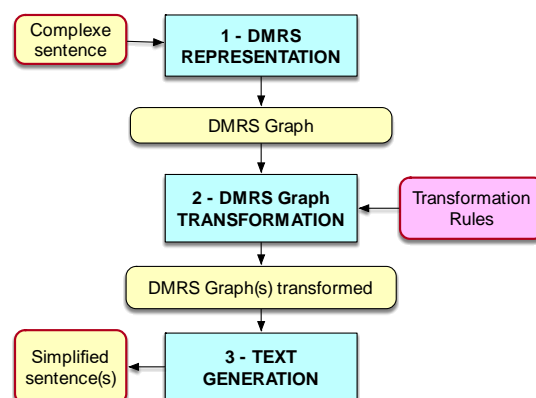


Figure 1. Steps of the syntactic simplification method.

GRASS implements a specific syntactic simplification method based on DMRS semantics and structured in three main steps as illustrated in Figure 1. The first one aims at representing the complex sentence by a DMRS graph-based meaning representation. The second step is to transform this DMRS graph into one or several DMRS graphs by applying a set of transformation rules defined manually (simplification rules). The third step consists of generating the simplified sentences from these transformed DMRS graphs.

GRASS is based on the English Resource Grammar (ERG) [34], a broad-coverage, symbolic grammar of English, developed as part of DELPH-IN² initiative and LinGO³ project. The ERG uses Minimal Recursion Semantics (MRS) [35] as semantic representation. The MRS format can be transformed into a more readable DMRS graph, which represents its dependency structure. The nodes correspond to predicates; edges, referred to as links, represent relations between them.

The ERG grammar is a bidirectional grammar which supports both parsing and generation. Several processors exist to parse sentences into MRSs and generate surface forms from MRS representations using chart generation. In our experiments, we used ACE⁴ to obtain DMRSs graphs and to generate other graphs from them. Parsing and generation are thus performed using already existing DELPH-IN tools. DMRS has already been used in other systems for prepositional phrase attachment disambiguation [36], for machine translation [37], for question generation [38], for evaluating multimodal deep learning models [39], and for sentiment analysis [40].

The DMRS notation considers both semantic and syntactic annotations of sentences. This enables to detect the syntactic constructions that has to be transformed. The semantically shared elements are kept to be able to rewrite them into the split sentence. This allows to have a simpler output which is both grammatical (syntactic information from DMRS) and to preserve the meaning (information related to semantics in DMRS). DMRS provides information about the thematic roles which are necessary to reconstruct the shared elements, and to detect complex syntactic constructions.

DMRS graphs can be manipulated using two existing Python libraries. The pyDelphin⁵ library is a more general MRS-dedicated library. It allows conversions between MRS and DMRS representations but internally performs operations on MRS objects.

² <http://moin.delph-in.net/wiki/>

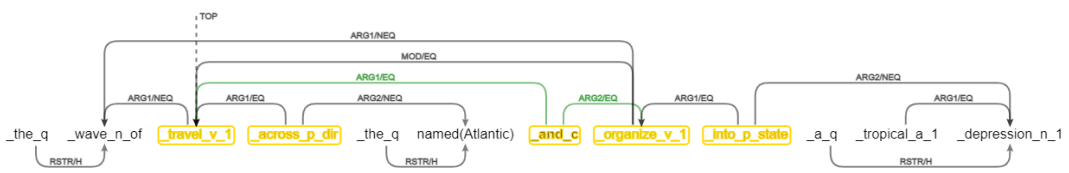
³ LINGuistic Grammars Online, <https://www-csli.stanford.edu/groups/lingo-project>

⁴ <http://sweaglesw.org/linguistics/ace/>

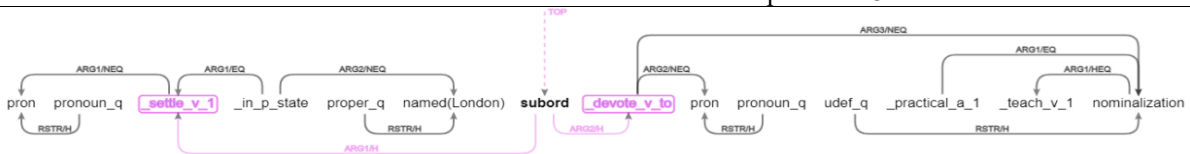
⁵ <https://github.com/delph-in/pydelphin>

We developed our simplification rules by examining data in raw texts and by transforming structural patterns into DMRS graphs. Currently, GRASS permits the syntactic simplification of 5 grammatical constructions: coordination (1), subordination (2), appositive clauses (3), relative clauses (4), passive forms (5). The DMRS representation of these sentences is showed in Figure 2. For the sake of clarity, we have modified the DMRS by deleting some elements in the sentences.

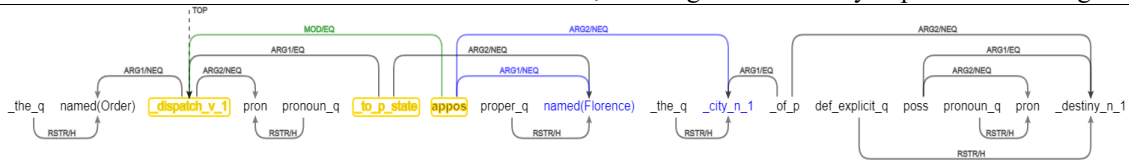
- (1) The wave traveled across the Atlantic, and organized into a tropical depression off the northern coast of Haiti on September 13.
- (2) He settled in London, devoting himself chiefly to practical teaching.
- (3) Finally, in 1482, the Order dispatched him to Florence, the city of his destiny.
- (4) It is located on an old portage trail which led west through the mountains to Unalakleet.
- (5) Most of the songs were written by Richard M. Sherman and Robert B. Sherman.



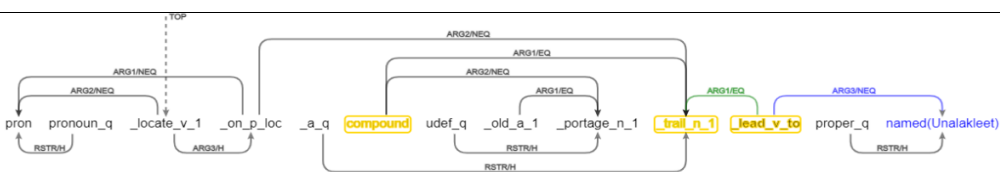
a. DMRS of sentence 1. “The wave traveled across the Atlantic, and organized into a tropical depression off the northern coast of Haiti on September 13”.



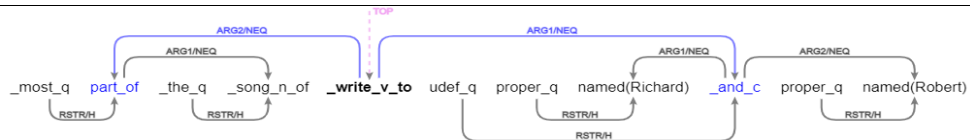
b. DMRS of sentence 2. “He settled in London, devoting himself chiefly to practical teaching”.



c. DMRS of sentence 3. “Finally, in 1482, the Order dispatched him to Florence, the city of his destiny”.



d. DMRS of sentence 4. “It is located on an old portage trail which led west through the mountains to Unalakleet”.



e. DMRS of sentence 5. “Most of the songs were written by Richard M. Sherman and Robert B. Sherman”.

Figure 2. DMRS graphs for sentences 1 to 5

To simplify these constructions, we extract triggering indicators (the arguments of conjunctions or prepositions). For each segmentation, we identify a splitting point that acts as a trigger, i.e., its

presence indicates the possibility of a segmentation. The development of the rules depends on the structure of the sentences in English. This involves studying each of the syntactic constructions to be processed, drawing up the “patterns” of constructions’ forms and translating them into manual rules.

3.2. GRASS Software Architecture

As illustrated in Figure 3, the software architecture is made of the following components: *Text Preparation*, *Semantic Parsing*, *Simplification* and *Text Generation*. In addition, there is a *DMRS graph visualization* component.

3.2.1. Preparation Component

This component prepares the corpus for simplification. The first operation is to put it in an interpretable format for the "Semantic Parsing" component (it transforms each sentence of the corpus into a DMRS semantic graph). In particular, the corpus to be processed must be divided into sentences. It is important to preserve the position of the sentences in the original corpus to be able to generate them in the right place.

3.2.2. Semantic Parsing Component

Semantic parsing is performed by the ACE component, developed by the DELPH-IN Consortium. ACE is an efficient processor for DELPH-IN HPSG grammars: ACE allows both to translate a sentence into a DMRS graph (ACE parser) and to generate a sentence from a DMRS graph (ACE generator). A sentence is taken as input and the output is an associated MRS format file describing the semantic information. MRS format cannot be handled by the tools that we have chosen to use for visualization and transformation. Therefore, it has to be transformed into a DMRS graph using a DELPH-IN utility.

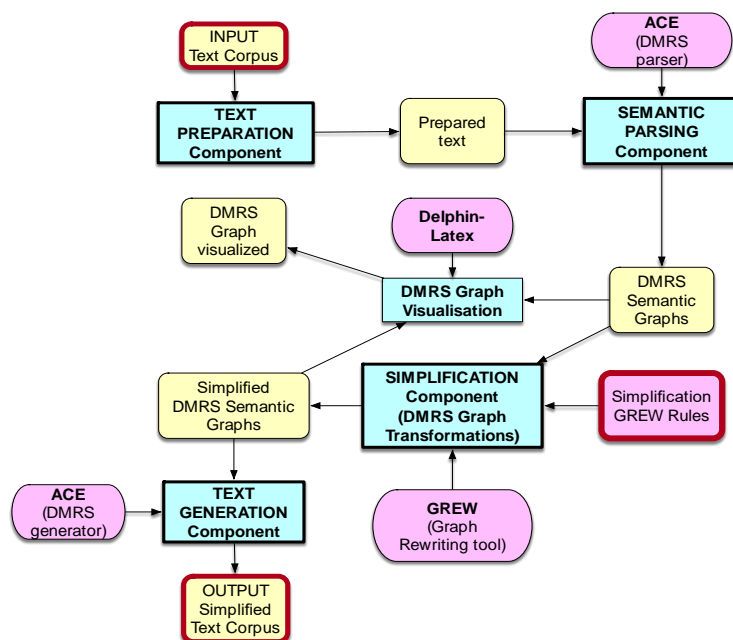


Figure 3. Software Architecture of GRASS system with its main components.

3.2.3. Simplification Component (DMRS Graphs Transformation)

This component simplifies the corpus, sentence by sentence, at the level of the DMRS graphs associated with each sentence of the corpus. It is based on GREW⁶ [41] [42] [43] developed at the LORIA laboratory of INRIA.

GREW is a Graph REWriting tool for applications in NLP that can manipulate syntactic and semantic representations. It is used on POS-tagged sequences, surface dependency syntax analysis, deep dependency parsing, and semantic representation (AMR, DMRS). It can also be used to represent any graph-based structure. As such, GREW permits to transform graph-based semantic representations in DMRS according to a set of rules.

Hand-crafted rules can be defined and applied on a DMRS graph. The rules are structured into three sections: (i) *pattern*: describes the part of graph to match, allowing the selection of nodes or edges thanks to their features, relations or positions in the graph; (ii) *without*: filters out unwanted occurrences of the pattern giving the possibility to exclude elements from a previous selection; (iii) *commands*: allows to apply structural transformations on the graph, such as the deletion, the creation or the reordering of the nodes and edges as well as the modification of their features in the graph. Each simplification operation transforming a DMRS graph is associated with a set of GREW rules (cf. section 3.3).

3.2.4. Generation Component

From the DMRS representations of the sentences of the corpus transformed by the GREW rules, this component generates the text associated with each sentence and places each generated sentence in the order of the original corpus. This component is based on the ACE tool that we already used for Semantic Parsing.

3.2.5. DMRS Graph Visualization Component

Delphin-Latex component, developed by the DELPH-IN Consortium [44], is a tool that takes as input a representation expressed in DMRS and visualizes the associated DMRS graph. This tool is very useful for the development of GREW simplification rules. It allows to visualize the DMRS representation before and after simplification.

3.3. Syntactic Simplification Rules

Our work enabled us to create simplification rules to transform DMRS graphs into other graphs. As regards to sentence splitting, we dealt with coordination, subordination, apposition, and relative clauses. We also worked on transformation from passive to active voices. These transformation rules, presented at an abstract level, are implemented in GREW. Our system contains 11 rules: 3 for apposition clauses, 3 for coordination clauses, 1 for passive to active voice transformation, 2 for relative clauses and 2 for subordination clauses. An example of GREW rule for rewrite one type of appositive clauses is presented in Figure 4.

3.3.1. Rules for Coordination Clauses

Coordination is formed by two or more elements linked by a conjunction such as “and”, “or”, etc. In DMRS, coordinations are identified by any relationship that has a *_c_* suffix, such as *_and_c_*

⁶ <https://www.grew.fr/>

and *_or_c_*. Coordination between propositions (not two nouns or adjectives) is our goal in splitting coordination. There are two types of coordinations between two clauses: clauses that share the same subject and clauses that do not share the same subject. We deal with these two cases. Sentence 1 is an example of coordination clause that sharing subject (*the wave*). The conjunction node C (*_and_c_*) takes the two verbs (*travel* V1 of the first clause and *organize* V2 of the second clause) of the two clauses as Arguments. The goal is to delete the conjunction C and to rewrite the shared subject (*the wave*) labeled ARG1/NEQ before the second verb adding edges between V2 and the rewritten subject. Sentence 1 can be transformed into two simpler sentences: *The wave traveled across the Atlantic. The wave organized into a tropical depression off the northern coast of Haiti on September 13.*

3.3.2. Rules for Subordination Clauses

In DMRS, subordination is marked by the label *_subord_*. The ARG1 of the subordinate clause refers to the main clause while the ARG2 refers to the subordinate clause (sentence 2). Thus, the splitting rule extracts all nodes linked to ARG1/2 separately and builds two new DMRSs. The goal is to transform a subordinate into a main and rewrite the shared subject. Sentence 2 can be transformed into two simpler sentences: *He settled in London. He devoted himself chiefly to practical teaching.*

3.3.3. Rules for Appositive Clauses

Apposition is formed by two adjacent nouns describing the same reference in a sentence. In DMRS, apposition in sentences can be captured precisely: it is identified by the label *appos* that takes the two adjacent nouns as arguments (sentence 3). The apposition splitting rule first duplicates the ARG1 of the node *appos*, removes it to form the first DMRS, then it builds the other DMRS by replacing *appos*' ARG1 with its ARG2. The second step is to add the verb to be in present simple after the reproduced subject. The last step is to add links between the verb to be, the subject and the object. Sentence 3 can be transformed into two simpler sentences: *Finally, in 1482, the Order dispatched him to Florence. Florence is the city of his destiny.*

3.3.4. Rules for Relative Clauses

Although relative pronouns indicate relative clauses, in a DMRS structure these relative pronouns are not explicitly represented: there is not a node for the relative pronoun “that”. However, the verb *lead* governs its subject by an /EQ relation. This indicates that *lead* and *trail* share the same tag and have the same scope. After splitting the sentence, this constraint of the same scope must be resolved. Sentence 4 can be transformed into two sentences: *It is located on an old portage trail. The trail led west through the mountains to Unalakleet.*

3.3.5. Rules for Transformation from Passive to Active Voices

A sentence in its active or passive form has two syntactic analyses, but the same semantic representation, hence the ease of the task by reversing the two arguments of the verb. In DMRS, the ARG1 the passive voice is the subject and ARG2 is the object. The goal is to reverse them to have ARG1 and ARG2 as object and subject respectively. Sentence 5 can be transformed into: *Richard M. Sherman and Robert B. Sherman wrote most of the songs.*

```

rule appos {
  pattern { %pattern of the graph to transform node appos and
the arguments
  R [gpred="appos"];
  m: R -[ARG1:NEQ]-> N;
  p: R -[ARG2:NEQ]-> M;}

  commands { %delete node appos, add verb to be, rewrite the
ARG1 node of the appos
  del_node R;
  add_node N1 :> N; append_feats N ==> N1;
  add_node V :>N ;
  V.lemma=be;V.pos="v";V.TENSE=pres; V.MOOD=indicative;
  add_edge V -[ARG2:NEQ]-> M;
  add_edge V -[ARG1:NEQ]-> N }}

```

Figure 4. Example of GREW rule for one case of appositive clause

4. EXPERIMENTAL SETUP

In this section we define the reference corpus and metrics used to evaluate GRASS.

4.1. Corpus

All systems including ours are tested on the HSplit⁷, the test corpus of [9] (the authors highlight that existing English Wikipedia-based datasets did not contain sufficient instances of sentence splitting). To overcome this problem, they collected four reference simplifications of this kind of transformation for all 359 original sentences in the Turkcorpus test set [22]. TurkCorpus⁸ comprises 359 sentences from the PWKP corpus [24] with 8 references collected by crowdsourcing for each of the sentences. In HSplit, each reference was created in only operating sentence splitting on the original complex sentence, so this is a data set for evaluating sentence splitting, but it does not generalize to sentence simplification in general.

For our evaluation, we used a parsing and regeneration procedure: each graph was transformed into sub-graphs. We fed the top parse for each sub-graph as input to the ACE generator, to finally recombine the sentences.

4.2. Evaluations Metrics

For the automatic evaluation of GRASS according to the following state-of-the art metrics we used the EASSE package [45]:

- (1) BLEU [48] relies on the proportion of n-gram matches between a system's output and references.
- (2) SARI [22] compares the n-grams of the system output with those of the input and the human references, separately evaluating the quality of words that are added, deleted, or kept by a system.
- (3) SAMSA [49] measures structural simplicity (i.e., sentence splitting), in contrast to SARI, which is designed to evaluate simplifications involving paraphrasing.

In addition, Quality Estimation Features leverages both the source sentence and the output simplification to provide additional information on simplification systems, in particular: (4) the

⁷ <https://github.com/eliorsulem/HSplit-corpus>

⁸ <https://github.com/cocoxu/simplification/tree/master/data/turkcorpus>

average number of sentence splits performed by the system, (5) the proportion of exact matches (i.e., original conserved sentences).

5. EXPERIMENT RESULTS

Applying GRASS to the 359 sentences of the TurkCorpus, as others syntactical simplification systems have done, we obtain 91 transformed sentences by our transformation rules. On these 359 sentences, 268 sentences were not changed when applying our rules. First, 265 sentences are not transformed because they are syntactically simple and cannot be simplified any further. Example: *Admission to Tsinghua is extremely competitive*. Finally, three other sentences that are syntactically complex are not transformed due to different reasons: (i) no rule has been applied on one sentence; (ii) a sentence has not been parsed by ACE parser, and (iii) a sentence that has been parsed and transformed but not generated by ACE generator.

As our system cannot transform sentences that do not contain the targeted syntactical constructions, we can consider that our system performs the transformation of 91 out of 94 sentences. We compared the transformed 91 sentences to the same ones obtained by the following systems. The outputs of these systems are collected from EASSE⁹ [45].

- Two semantic-based syntactic simplification DSS [11] and HYBRID [10].
- Phrase-based Machine Translation (PBMT-R) [46]. The outputs are collected from DRESS repository¹⁰.
- Sentence Simplification with Deep Reinforcement Learning (DRESS-LS) [47].
- Unsupervised Neural Text Simplification UNTS [25].

Results presented in Table 1 show that for these specific metrics computed by EASSE, GRASS obtains higher BLEU, SARI and SAMSA scores than semantic-based, Phrase-based MT and Neural-based text simplification systems. GRASS gets lower additions and deletions proportions because it doesn't deal with lexical simplification and other rewriting operations.

While recent improvement in text simplification has been achieved by the use of neural MT (NMT) approaches, sentence splitting operation has not been addressed by these systems, potentially due to the rareness of this operation in the training corpora [22]. Indeed, experimenting with a neural system [47][25], these systems present the higher score of unchanged input sentences (conservatism) and lower score of splitting sentences (0.13 and 0.12 for DRESS-LS and UNTS respectively), comparing to semantic-based systems.

Table 1. Automatic evaluation for text simplification systems for the 91 transformed sentences.

Metrics	GRASS	DSS	HYBRID	PBMT-R	DRESS-LS	UNTS
BLEU	63.85	62.49	25.65	60.23	43.06	48.0
SARI	48.81	48.03	25.04	36.24	38.10	32.4
SAMSA	51.44	48.13	30.86	33.54	25.445	26.69
Sent. splits	2.01	2.53	0.98	1.04	0.99	1.01
Exact copies	0.0	0.01	0.04	0.08	0.13	0.12

Table 2 and 3 give two examples of these systems outputs of the test corpus. Each system splits the original sentence in a specific manner (e.g., DSS splits “more” but not “better”).

⁹ <https://github.com/feralvam/easse>

¹⁰ <https://github.com/XingxingZhang/dress/tree/master/all-system-output/WikiLarge/test>

DSS and GRASS split the first sentence into 3 fragments. The second sentence is split into 5 fragments by DSS, while GRASS system splits it into 3 fragments. As we can see, the sentences obtained by DSS are not simpler than the original one, they are not semantically correct, and they are agrammatical. GRASS splits sentences into semantically and syntactically correct constructions. HYBRID did not split the sentences; it rewrote them by removing parts making the sentences linguistically incorrect and changing their original meanings. Finally, the translation-based system (PBMT-R) is conservative for the two sentences. Neural-based systems simplify sentence privileging the lexical simplification and deletion operation but not splitting operation.

Table 2. System outputs for example 1 of the test sentences.

EXAMPLE 1	
Original	<i>The tarantula, the trickster character, spun a black cord and, attaching it to the ball, crawled away fast to the east, pulling on the cord with all his strength.</i>
Hybrid	The tarantula, the trickster character, a black spun cord, and it attaching, crawled, pulling all.
DSS	the tarantula the trickster character spun a black cord . attaching it to the ball . character crawled away fast to the east . character pulling on the cord with all his strength .
PBMT-R	The Spider, the trickster character, made a black cord and attached to the ball, crawled away fast to the east, pulling on the cord, with all his strength.
DRESS-LS	The tarantula, the trickster character, spun a black cord and, holding it to the ball.
UNTS	The spider, the trick character, spun a black cord,
GRASS	The tarantula is the trickster character. The tarantula spun a black cord. Attaching it to the ball, the tarantula crawled away fast, to the east. The tarantula pulled on the cord, with all of his strength.

Table 3. System outputs for example 2 of the test sentences.

EXAMPLE 2	
Original	<i>Following the drummers are dancers, who often play the sogo (a tiny drum that makes almost no sound) and tend to have more elaborate — even acrobatic — choreography.</i>
Hybrid	Dancers, play the sogo (a drum that no and to .
DSS	the drummers are . dancers often play the sogo (a tiny drum makes almost no sound) . drum makes almost no sound) . the sogo tend to . the sogo have more elaborate even acrobatic choreography .
PBMT-R	Following the drummers are dancers, who often play the sogo (a small drum that makes almost no sound) and tend to have more elaborate -- even acrobatic -- choreography.
DRESS-LS	Following the drummers are dancers, who often play the sogo (a small drum that makes almost no sound).
UNTS	Following the musicians are dancers, who often play the Sogo (a tiny drum that makes almost no sound) and tend to have more happy even - .
GRASS	Dancers, which, play the sogo, often, are following the drummers. The sogo is a tiny drum, which, makes almost no sound. The dancers tend to have more elaborate, even acrobatic choreography.

To compare the semantic-based operation and while Hybrid and DSS deal essentially the coordination and relative clauses, we see that passive forms, appositive and subordination clause are not handled. As we can see, GRASS covers a wider range of syntactic structures and that is

due to the choice of semantic representation formalism. DMRS is suited for Natural Language Understanding tasks: unlike UCCA, DMRS has a specific label for proper name; so, in generation, proper names are recognized, and the first letter is capitalized. DMRS gives information about verb mode and tense, our rules are defined in a way that they enable to conjugate the verb in the right tense after splitting.

Finally, while DSS does “more” sentence splitting than other systems, that does not mean that it splits them “better”. One of the disadvantages of automatic measures like SAMSA or the average number of sentence splits is that they count the number of ending points in an output without considering the syntactic and semantic aspects in the sentence. DSS has high score for SAMSA and for the number of splitting. However, the meaning is not always kept, and the output does not preserve the Subject-Verb-Object (SVO) order. The important number of splitting doesn’t mean that the system performs better, yet it is considered as such following the automatic metrics.

6. CONCLUSIONS

In this paper, we have presented GRASS, an automatic syntactic simplification system for English based on semantic representations. To implement our system, we used different available NLP tools performing parsing, graph generation, visualization, and sentence rewriting. After a comparison with established state-of-the-art similar methods, our system outperforms particularly on rewriting shared elements on the 359 sentences of TurkCorpus as other existing syntactic simplification systems. Our system also provides a better coverage of syntactic constructions and provides interpretability of the syntactic transformations. We have run an automatic evaluation that shows that GRASS has better scores on BLEU, SARI and SAMSA scores as regards to other existing systems. On this TurkCorpus corpus reduced to 359 sentences we are currently running a human evaluation campaign that will provide a more fine-grained linguistic analysis of the data obtained with our system. However, the evaluation of our system should be done on a larger corpus than the TurkCorpus limited to 359 sentences, in which only 94 sentences are concerned by the transformations defined in GRASS. We hope to be able to evaluate our system, mainly automatically, on a larger corpus: the complete TurkCorpus, but also other corpora like the Newsela corpus.

In the future we would also like to couple our syntactic simplification system with an existing lexical simplification system based on neural techniques, which would allow us to compare our system with other simplification systems, and to measure the impact of combining these two levels of simplification.

ACKNOWLEDGEMENTS

The authors would like to thank Bruno Guillaume, and Guy Perrier for their support on GREW, and Bastien Gastinel, Hamza Ghorfi and William Domingues for their technical contributions to the development of GRASS.

REFERENCES

- [1] Saggion, H. (2017). Automatic text simplification: Synthesis lectures on human language technologies, vol. 10 (1). *California, Morgan & Claypool Publishers*.
- [2] Štajner, S., & Popović, M. (2016). Can text simplification help machine translation? In *Proceedings of the 19th Annual Conference of the European Association for Machine Translation* (pp. 230-242).
- [3] Niklaus, C., Bermeitinger, B., Handschuh, S., & Freitas, A. (2017). A sentence simplification system for improving relation extraction. *arXiv preprint arXiv:1703.09013*.

- [4] Vanderwende, L., Suzuki, H., Brockett, C., & Nenkova, A. (2007). Beyond SumBasic: Task-focused summarization with sentence simplification and lexical expansion. *Information Processing & Management*, 43(6), 1606-1618.
- [5] Rello, L., Baeza-Yates, R., Bott, S., & Saggion, H. (2013, May). Simplify or help? Text simplification strategies for people with dyslexia. In *Proceedings of the 10th International Cross-Disciplinary Conference on Web Accessibility* (pp. 1-10).
- [6] Sauvan, L., Stolowy, N., Aguilar, C., François, T., Gala, N., Matonti, F., ... & Calabrese, A. (2020, May). Text simplification to help individuals with low vision read more fluently. In *Proceedings of the 1st Workshop on Tools and Resources to Empower People with READING Difficulties (READI)* (pp. 27-32).
- [7] Siddharthan, A. (2002, December). An architecture for a text simplification system. In *Language Engineering Conference, 2002. Proceedings* (pp. 64-71). IEEE.
- [8] Copestake, A. (2009, March). Invited Talk: Slacker semantics: Why superficiality, dependency and avoidance of commitment can be the right way to go. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)* (pp. 1-9).
- [9] Sulem, E., Abend, O., & Rappoport, A. (2018). BLEU is not suitable for the evaluation of text simplification. *arXiv preprint arXiv:1810.05995*.
- [10] Narayan, S., & Gardent, C. (2014, June). Hybrid simplification using deep semantics and machine translation. In *The 52nd annual meeting of the association for computational linguistics* (pp. 435-445).
- [11] Sulem, E., Abend, O., & Rappoport, A. (2018). Simple and effective text simplification using semantic and neural methods. *arXiv preprint arXiv:1810.05104*.
- [12] Chandrasekar, R., Doran, C., & Bangalore, S. (1996). Motivations and methods for text simplification. In *COLING 1996 Volume 2: The 16th International Conference on Computational Linguistics*.
- [13] Siddharthan, A. (2002, December). An architecture for a text simplification system. In *Language Engineering Conference, 2002. Proceedings* (pp. 64-71). IEEE.
- [14] Brouwers, L., Bernhard, D., Ligozat, A. L., & François, T. (2014, April). Syntactic sentence simplification for French. In *Proceedings of the 3rd Workshop on Predicting and Improving Text Readability for Target Reader Populations (PITR)@ EACL 2014* (pp. 47-56).
- [15] De Belder, J., & Moens, M. F. (2010). Text simplification for children. In *Proceedings of the SIGIR workshop on accessible search systems* (pp. 19-26). ACM; New York.
- [16] Siddharthan, A. (2014). A survey of research on text simplification. *ITL-International Journal of Applied Linguistics*, 165(2), 259-298.
- [17] Shardlow, M. (2014). A survey of automated text simplification. *International Journal of Advanced Computer Science and Applications*, 4(1), 58-70.
- [18] Shieber SM, Schabes Y. Synchronous tree-adjointing grammars.
- [19] Siddharthan, A., & Mandya, A. A. (2014). Hybrid text simplification using synchronous dependency grammars with hand-written and automatically harvested rules. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2014)*. Association for Computational Linguistics.
- [20] Candido Jr, A., Maziero, E. G., Specia, L., Gasperin, C., Pardo, T., & Aluisio, S. (2009, June). Supporting the adaptation of texts for poor literacy readers: a text simplification editor for brazilian portuguese. In *Proceedings of the Fourth Workshop on Innovative Use of NLP for Building Educational Applications* (pp. 34-42).
- [21] Candido Jr, A., Maziero, E. G., Specia, L., Gasperin, C., Pardo, T., & Aluisio, S. (2009, June). Supporting the adaptation of texts for poor literacy readers: a text simplification editor for brazilian portuguese. In *Proceedings of the Fourth Workshop on Innovative Use of NLP for Building Educational Applications* (pp. 34-42).
- [22] Xu, W., Napoles, C., Pavlick, E., Chen, Q., & Callison-Burch, C. (2016). Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4, 401-415.
- [23] Scarton, C., & Specia, L. (2018, July). Learning simplifications for specific target audiences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)* (pp. 712-718).

- [24] Zhu, Z., Bernhard, D., & Gurevych, I. (2010, August). A monolingual tree-based translation model for sentence simplification. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)* (pp. 1353-1361).
- [25] Surya, S., Mishra, A., Laha, A., Jain, P., & Sankaranarayanan, K. (2018). Unsupervised neural text simplification. *arXiv preprint arXiv:1810.07931*.
- [26] Narayan, S., & Gardent, C. (2015). Unsupervised sentence simplification using deep semantics. *arXiv preprint arXiv:1507.08452*.
- [27] Todirascu, A., Wilkens, R., Rolin, E., François, T., Bernhard, D., & Gala, N. (submitted) HECTOR: A Hybrid TExt SimplifiCation TOol for Raw text in French. Current submission to LREC 2022.
- [28] Kamp, H. (2013). A theory of truth and semantic representation. In *Meaning and the Dynamics of Interpretation* (pp. 329-369). Brill.
- [29] Narayan, S., Gardent, C., Cohen, S. B., & Shimorina, A. (2017). Split and rephrase. *arXiv preprint arXiv:1707.06971*.
- [30] Gardent, C., Shimorina, A., Narayan, S., & Perez-Beltrachini, L. (2017, July). Creating training corpora for nlg micro-planning. In *55th annual meeting of the Association for Computational Linguistics (ACL)*.
- [31] Mendes, P. N., Jakob, M., & Bizer, C. (2012). *DBpedia: A multilingual cross-domain knowledge base* (pp. 1813-1817). European Language Resources Association (ELRA).
- [32] Abend, O., & Rappoport, A. (2013, August). Universal conceptual cognitive annotation (UCCA). In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 228-238).
- [33] Nisioi, S., Štajner, S., Ponzetto, S. P., & Dinu, L. P. (2017, July). Exploring neural text simplification models. In *Proceedings of the 55th annual meeting of the association for computational linguistics (volume 2: Short papers)* (pp. 85-91).
- [34] Flickinger, D. (2000). On building a more efficient grammar by exploiting types. *Natural Language Engineering*, 6(1), 15-28.
- [35] Copestake, A., Flickinger, D., Pollard, C., & Sag, I. A. (2005). Minimal recursion semantics: An introduction. *Research on language and computation*, 3(2), 281-332.
- [36] Emerson, G., & Copestake, A. (2015). Leveraging a semantically annotated corpus to disambiguate prepositional phrase attachment. Association for Computational Linguistics.
- [37] Horvat, M. (2017). *Hierarchical statistical semantic translation and realization* (No. UCAM-CL-TR-913). University of Cambridge, Computer Laboratory.
- [38] Yao, X., Bouma, G., & Zhang, Y. (2012). Semantics-based question generation and implementation. *Dialogue & Discourse*, 3(2), 11-42.
- [39] Kuhnle, A., & Copestake, A. (2017). Shapeworld-a new test methodology for multimodal language understanding. *arXiv preprint arXiv:1704.04517*.
- [40] Kramer, J., & Gordon, C. (2014, August). Improvement of a naive Bayes sentiment classifier using MRS-based features. In *Proceedings of the Third Joint Conference on Lexical and Computational Semantics (*SEM 2014)* (pp. 22-29).
- [41] Guillaume, B., Bonfante, G., Masson, P., Morey, M., & Perrier, G. (2012, June). Grew: un outil de réécriture de graphes pour le TAL (Grew: a Graph Rewriting Tool for NLP)[in French]. In *Proceedings of the Joint Conference JEP-TALN-RECITAL 2012, volume 5: Software Demonstrations* (pp. 1-2).
- [42] Bonfante, G., Guillaume, B., & Perrier, G. (2018). *Application of Graph Rewriting to Natural Language Processing*. John Wiley & Sons.
- [43] Guillaume, B. (2021, April). Graph Matching and Graph Rewriting: GREW tools for corpus exploration, maintenance and conversion. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations* (pp. 168-175).
- [44] Goodman, M. W. (2019, October). A Python library for deep linguistic resources. In *2019 Pacific Neighborhood Consortium Annual Conference and Joint Meetings (PNC)* (pp. 1-7). IEEE.
- [45] Alva-Manchego, F., Martin, L., Scarton, C., & Specia, L. (2019). EASSE: Easier automatic sentence simplification evaluation. *arXiv preprint arXiv:1908.04567*.
- [46] Wubben, S., Van Den Bosch, A., & Kraemer, E. (2012, July). Sentence simplification by monolingual machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 1015-1024).
- [47] Zhang, X., & Lapata, M. (2017). Sentence simplification with deep reinforcement learning. *arXiv preprint arXiv:1703.10931*.

- [48] Papineni, K., Roukos, S., Ward, T., & Zhu, W. J. (2002, July). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics* (pp. 311-318).
- [49] Sulem, E., Abend, O., & Rappoport, A. (2018). Semantic structural evaluation for text simplification. *arXiv preprint arXiv:1810.05022*.

AUTHORS

Rita Hijazi is a PhD student at Aix-Marseille University, France, since 2019, in co-direction between the Laboratoire Parole et Langage (Speech and Language Laboratory), LPL UMR7309 and the Laboratoire Informatique et Systèmes (Computer Science and Systems Laboratory) LIS UMR7020. She has a Bachelor's degree in Linguistics and a Master's degree in Natural Language Processing from the Lebanese University, Lebanon. Her research interests involve NLP tasks like Automatic Text Simplification.



Bernard Espinasse obtained his PhD in 1981 from the University of Aix-Marseille (AMU) after an Engineer diploma from the Ecole Nationale Supérieure des Arts et Métiers of Paris in 1977. He was Assistant Professor at Laval University in Quebec (Canada) from 1983 to 1987. He is currently Full Professor at AMU and researcher at LIS UMR CNRS 7020 lab., where he was team leader for more than fifteen years. He is the author of numerous publications in various fields of computer science, particularly in text mining.



Núria Gala is Assistant Professor at Aix Marseille Univ. (AMU, France) since 2004 and researcher at the Laboratoire Parole et Langage (LPL UMR 7309) since 2017. She is interested in analyzing linguistic complexity and in building resources to help struggling readers improve reading and vocabulary learning. Her research projects are oriented towards the use of language technologies in computer-assisted language learning applications, and towards populations with special reading-comprehension needs (low-readers, dyslexic readers, illiterates, etc.). She is the author of numerous publications in computational linguistics, and natural language processing.



COMPARISON OF VARIOUS FORMS OF SERIOUS GAMES: EXPLORING THE POTENTIAL USE OF SERIOUS GAME WALKTHROUGH IN EDUCATION OUTSIDE THE CLASSROOM

Xiaohan Feng¹ and Makoto Murakami²

¹Graduate School of Information Sciences and Arts,
Toyo University, Kawagoe, Saitama, Japan

²Dept. of Information Sciences and Arts,
Toyo University, Kawagoe, Saitama, Japan

ABSTRACT

The advantages of using serious games for education have already been proven in many studies, especially narrative VR games, which allow players to remember more information. On the other hand, game walkthrough can compensate for the disadvantages of gaming, such as pervasiveness and convenience. This study investigates whether game walkthrough of serious games can have the same learning effect as serious games. Use game creation (samples) and questionnaires, this study will compare the information that viewers remember from game walkthrough and actual game play, analyze their strengths and weaknesses, and examine the impact of the VR format on the results. The results proved that while game walkthrough allows subjects to follow the experiences of actual game players with a certain degree of empathy, they have limitations when it comes to compare with actual gameplay, especially when it comes to topics that require subjects to think for themselves. Meanwhile game walkthrough of VR game is not a medium suitable for making the receiver memorize information. For prevalence and convenience, however, serious games walkthrough is a viable educational option outside the classroom.

KEYWORDS

Serious game, multimedia, educational game, virtual reality, narratology, Education Outside the Classroom (EOTC).

1. INTRODUCTION

In recent years, there has been a growing body of research on the gamification of education. Some studies use the elements and aesthetics of games to motivate students to learn [1], while others claim the advantage of gamification that games are a familiar medium for students and are more interesting than written media [2]. In addition to this, there is already abundant evidence that video games have a more or less positive impact on the cognitive domain. For example, games can improve visual spatial resolution [3], reaction time [4], spatial awareness [5], probabilistic reasoning [6], and visual short-term memory [7]. Overall, games have been shown to lead education into a new realm.

However, it is worth mentioning that games are a comprehensive medium. Especially when compared to written media, which are still the most used in education. This means that the results obtained are also comprehensive. If the importance of a particular attribute in a game needs to be examined, the results can be made clearer by focusing the sample on that attribute of the game. In previous studies, we have chosen "narrative" and "immersion". In other words, we created a sample of serious VR games with narrative content.

The reason for this choice is that serious and educational games are closely related. Both are trying to use some aspect of the game to achieve some goal other than entertainment [8]. On the other hand, according to neurologist Michael Smith, when people watch narrative images that induce empathy, the brain automatically filters out external influencing factors to focus on learning and cognition, indicating that the human mind wants to know the unknown from a narrative perspective. He believes that without a narrative connection, people cannot stay in the brain for long [9]. There is also considerable research showing that memories of events with a strong narrative component are more likely to be remembered [10]. And many studies have revealed that an immersive virtual reality system can better facilitate situational memory performance [11], [12].

Consequently, we reinforce memory through narrative and immersion, and bring players into an independent worldview in the form of VR games to increase the target ability of information transfer and reduce the psychology of distributed responsibility. Our previous research proposes that using virtual reality technology to gamify narrative Contents is a way to make information more deeply memorable.

Previous studies have confirmed the above. In the case of a VR game that presents the story in images versus a game that presents the story in text, that presents the story in images is a better way to remember the information. The study also found a positive correlation between subjects' experience and their self-reported confidence in their memory. The researchers believe that gaming experience and short text reading experience are the reasons for the subjects' better self-perceived confidence [13].

This result and the information provided in the subject interviews have attracted the attention of researchers. In previous experiments, subjects in the non-VR group watched game walkthrough more often than they played the games themselves. Game walkthrough is when players record their own gameplay, and the video content is often uploaded to video sharing sites (such as YouTube) or streamed live on streaming sites (such as Twitch). There are millions of potential recipients for these contents [14].

The researchers conducted additional interviews with all groups of subjects and found that 95% of the subjects had watched game walkthrough, 74.16% watched at least 7 hours per week, and they were in the habit of watching them while eating. Their reasons for watching game walkthroughs were as follows:

- Watching game walkthroughs allows them to be exposed to games anytime and anywhere.
- They don't have the hardware for a particular game, or the hardware doesn't meet the requirements of the game. So they can only get to know a game by watching it.
- They have some game walkthrough players who are particularly fond of.
- By watching the game, they can find the hidden elements of the game quickly, or see the hidden ending without having to play the game for long time by themselves.

- Finally, the most common reason was that the videos can be viewed at any time on cell phones. From these reasons, it is clear that video media is certainly more accessible to users' fragmented time compared to game media.

From the above survey, we found that game walkthrough is convenient and accessible, so why not use it in the educational field.

The main theoretical idea of this study is based on Self-Determination Theory (SDT) [15]. This means that it is argued that the essence of behavior is motivated by interest and enjoyment [16]. Through our survey, we found that most college student gamers are interested in game walkthrough. In addition, the data on the time spent watching game walkthrough suggests that they are not just a way to pass the time, but that they enjoy them. Against the above background, This study investigates whether game walkthrough of serious games can have the same learning effect as serious games. Use game creation (samples) and questionnaires, this study will compare the information that viewers remember from game walkthrough and actual game play, analyze their strengths and weaknesses, and examine the impact of the VR format on the results.

2. METHODS

We create a VR game and a not VR game and record the game walkthroughs. And we divide the subjects into four groups: who play a not VR game (NVR group), who watch the walkthrough of the not VR game (group GW), who play a VR game (group VR), and who watch the walkthrough of the VR game (group GWVR). A questionnaire will be used to find out which group's subjects remember the game story. The sample is based on actual events. More than 80% of the text is taken directly from the news interviews. References are to newspaper articles about the incident from 2006 to 2015. These news stories were distributed across diverse media platforms over a long period of time. They were selected as scenarios for the serious game because of their integrative nature and the narrative nature required for this experiment.

2.1. Sample

The game is divided into four stages in total, each with 1-2 enemies and 2 key items, and players must collect key items while avoiding enemy pursuit. The key items in each stage are related to the storyline of that stage.

First, Design the game characters and draw the front and side views of the characters for modeling. After completing the design of the game's characters and scenes, modeling is done in 3ds Max. Before modeling, import the front and side views into the background to build the character model more accurately. Once the modeling is complete, create the texture. In order to make the texture position correspond to the model position, the model is introduced into UNFOLD3D and the texture is expanded. The next step is to import the model into Mudbox and draw the texture. Back in 3ds Max again, bound the bones, moved the bones in the model, and used keyframe animation to animate the character walking, running, and attacking. Finally, import the characters and scenes into Unity to create the game. Screenshots of the game are shown in Figure 1.

Also, the game walkthrough is actually divided into various genres. For example, "Speedrun" [17], which aims to finish a game as quickly as possible. "Longplay" [18] focuses on completely documenting the gameplay process, with little commentary from the player. "Let's play" [19] focuses on the player's in-game experience and sharing.

Considering the possibility of experimentation, research first consider eliminating the live broadcast type of play. Next, according to the interviews with the subjects, most of the game walkthrough they watch are of the “let’s play” type. And in “let’s play” type the player’s comments and reactions will be more sympathetic to the audience than in the other types. So in the “let’s play” type game walkthrough the viewers can be able to relate to the players more than in the other types. And the researchers expect that the “let’s play” type can reduce the lack of immersion which is one of the problems of using game walkthrough. Therefore, the game walkthrough used in this project is of the “let’s play” type. The game walkthrough will be recorded by inviting two female game walkthrough players who post videos, and they have never been exposed to the sample information until the game walkthrough is recorded.



Figure 1. Screenshots of the game

2.2. Questionnaire

All subjects were recruited through convenience sampling and snowball sampling. The original plan was for some subjects to experience the VR face-to-face with the researchers, but this was all changed to online for the new Corona. Subjects were asked to download the samples themselves through a link provided by the researcher, experience the samples on their respective devices, and mail in a questionnaire and recording. All subjects were aware of and consented to the experiment before it was conducted.

The questionnaire is divided into four written questionnaires and one recorded questionnaire. The written questionnaire consists of basic information, recognition check, correctness check, and empathy check, while the recorded questionnaire consists of subjects telling a story and giving their impressions of the sample. The basic information questionnaire asks the subjects their age, gender, major, gaming experience, experience using VR, and the theme of the sample. The recognition check and the correctness check use the same 10 questions. The purpose of the recognition check is to find out how much the subjects themselves think they know about the sample story, and their actual cognitive status is not important in this research. In the empathy check, the strength of the empathic emotion for each subject is investigated through a self-report questionnaire. Questionnaire as shown in Figure 2.

Basic information			
<i>Age</i>	<i>Gender</i>	<i>Game experience</i>	<i>major</i>
Have you ever used VR?		No	Yes
Have you paid attention to the news of female population sales?		No	Yes
Have you read related articles on this game?		No	Yes

Recognition check					
Question	<i>Get it</i>	<i>Maybe Get it</i>	<i>Not sure</i>	<i>Maybe don't get it</i>	<i>Don't get it</i>
Where the protagonist was kidnapped?					
Who bought the protagonist?					
What is the attitude of the protagonist's husband towards her?					
What are the protagonists' means of suicide?					
Why the protagonists left the village?					
What reminds the protagonist of life's hopes?					
How is the protagonist known to the public?					
Why doesn't the local government want the public to know about the protagonist?					
What is the attitude of the villagers towards this?					
Did you understand the ending?					
Correctness check					
Full score 100	<i>Correct answer +10</i>			<i>score:</i>	

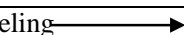








Empathy check	
Stage	No feeling  felt a strong emotion
1	
2	
3	
4	
<p> : No feeling</p> <p> : Feeling emotions, but not enough to take action.</p> <p> : Feels emotional and takes short-term/single/simple actions.</p> <p> : Feels emotional and takes long-term/ continuous/ /complex actions.</p>	

Figure 2. Questionnaire

3. EXPERIMENTAL RESULTS AND ANALYSIS

3.1. Basic Information

The subjects were 120 university students (62 females, 59 males, mean age 21.183 years, age range 18-24 years, SD=1.402). 120 subjects were divided into 4 groups of 30 each:

Group NVR: 19 females, 11 males, age range 19-23 years, mean age 20.733 years, SD=1.263.

Group VR: 13 females, 17 males, age range 19-23 years, mean age 20.833 years, SD=1.240.

Group GW: 15 females, 15 males, age range 18-24 years, mean age 21.3 years, SD=1.486.

Group GWVR: 14 females, 16 males, age range 19-24 years, mean age 21.866 years, SD=1.309).

The overall gender ratio is basically the same, but the gender ratio of each group is different; As can be seen from the gender ratios of the four groups, the sex ratio of the game walkthrough group appears to be average when compared to the large number of females in the group and the large number of males in the group. In a gender survey on VR and AR device Ownership rate conducted in 2017 [20], 43% of device owners were female and 31% of those planning to purchase a device were female. According to data from another U.S. gamer gender survey, the highest percentage of female gamers has only been 48% since 2006 [21]. Combining the data from this survey and its collection process, we can assume that although the number of women who own VR devices is smaller than men, the number of women interested in gaming is not small.

The majors of the 120 subjects included 21 in Science and Engineering, 20 in Education, 18 in Arts, 14 in Management, 12 in Architecture, 12 in Literature, 9 in Economics, 6 in Sociology, 3 in Physical Education, 3 in Philosophy, and 2 in History. The diversity of the subjects' majors is beneficial in obtaining more diverse perspectives in the recording survey.

When the subjects self-reported their gaming experience, 71 said they had "a lot of gaming experience," 15 said they had "normal gaming experience," and 34 said they had "little gaming experience. Overall, the subjects had a lot of gaming experience.

3.2. Recognition Check

Figure 3 shows the percentage for each option chosen by each group. Overall, Group NVR is the most confident in their own memory, followed by Group GW, then Group VR, and finally Group GWVR.

As can be seen from Figure 3, the selection tendencies of Group NVR and Group GW are very close. More than half of the respondents chose "Get it", and around 30% of the respondents chose "Maybe Get it". "Not Sure" was selected slightly more often by Group GW, but the difference between the two groups was not large. Group VR and Group GWVR not only differed from these two groups, but even in the same VR-related sample, their selection trends were not consistent.

the options for Group GWVR are distributed on average compared to the other groups. The percentage of times Group GWVR chose "Not Sure" was the highest among all options for this group, reaching 3%. The number of times they chose "Get it" and "Maybe get it" was equal, both accounting for 25%. "Get it" was chosen the most by the other three groups, with 18%, three times as many as the group VR chose the same item.

Studies have shown that the more experienced a person is, the more confident he or she is in identifying memories [22]. This may explain the greater confidence in self-recognition of memories in Group NVR and Group GW. Although the subjects in this experiment had more gaming experience, in daily life, visual media is more accessible than gaming media and does not require more energy or two hands. In addition, Chinese college students, the age group of the subjects, spend less time in contact with game media than video media. By combining the data of group NVR, group GW and group VR, it was proved that the memory discrimination and experience of the test takers are positively correlated [23].

The data of group GWVR shows different characteristics from this. The researchers made an additional visit to the group on this issue. "More than half of the subjects who chose "Not Sure" said that they could not concentrate on the GWVR sample throughout. There are two main reasons for this:

1. They are not used to VR game walkthroughs from the beginning. These subjects found it more distracting to watch a VR game walkthrough than to watch a regular game walkthrough.
2. At first, they were able to enjoy the VR game walkthrough, but later in the video, they felt uncomfortable due to the constant shaking of the camera.

Researchers believe that these two cases, especially the former one, reaffirm the positive correlation between media experience and subjects' memory.

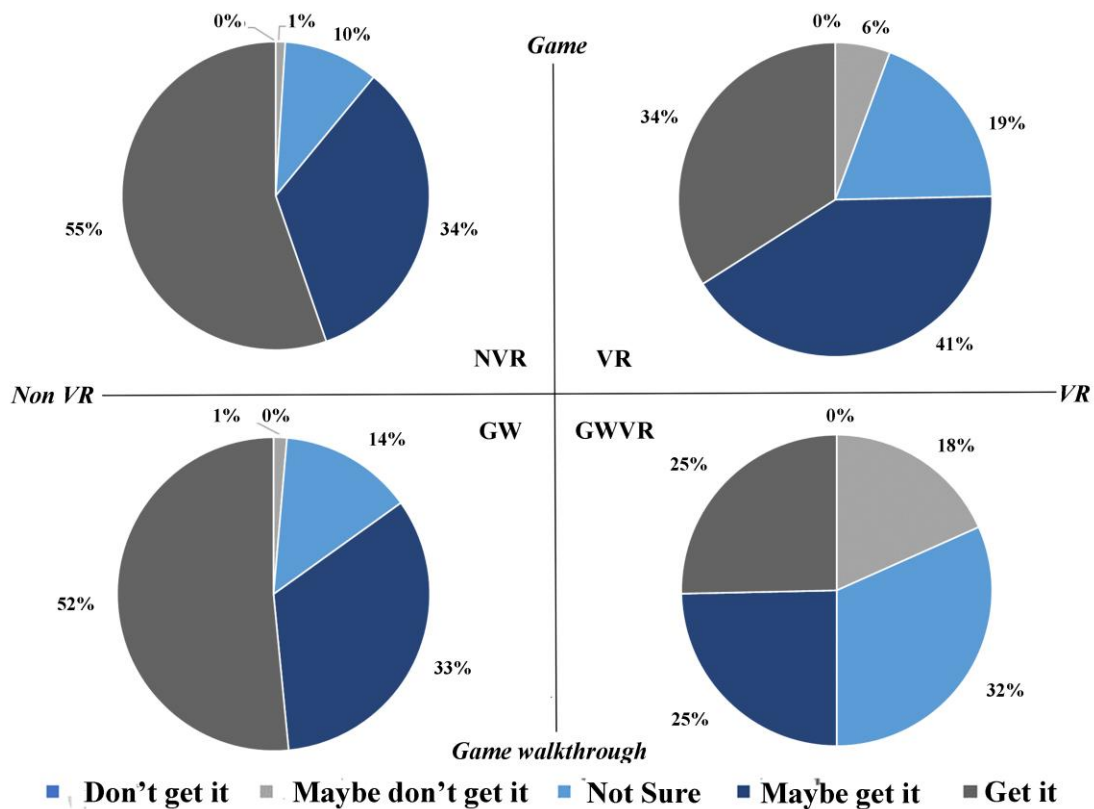


Figure 3. The percentage of times each option was selected in each group

3.3. Correctness Check

The same question is used for the recognition check and the correctness check. There are five levels of correctness depending on the subject's answer. The score for each level is: "More than correct": 5, "Correct": 4, "Mostly correct": 3, "Somewhat correct": 2, and "Incorrect": 1. The total score is 1500.

As Figure 4 shows, the total score of each group is Group NVR: 1153 points, Group GW: 1141 points, Group VR: 1248 points, Group GWVR: 1060 points. The scores are, in order from highest to lowest, VR>NVR>GW>GWVR.

It is clear that the overall score of the game sample is higher than the game walkthrough, but contrary to expectations, the difference in scores between group NVR and group GW is only 2 points. The highest and lowest scores were significantly different from those of adjacent groups, with a difference of 95 points between group VR and group NVR, and 81 points between group GW and group GWVR.

Taken together with the recognition check in the previous section, this indicates that Group NVR and Group GW are not only similar in their tendency to select for memory recognition, but also have very similar levels of the actual memory component.

The score ranking for the subjective questions again matches the overall score ranking: VR>NVR>GW>GWVR. Higher scores on subjective questions require more empathy from the subjects. If the genre is one that stimulates empathy, it may be better to have people play the game directly rather than deliver it using a game walkthrough to achieve the goal. Therefore, if subjective feedback from viewers is needed, the VR sample can provide the most feedback among the four groups. Next is NVR games. In terms of Ownership rate of VR and pc devices, NVR may be a more suitable medium. On the other hand, Game walkthroughs did not provide comparable subjective responses, especially for VR games.

In other words, if the subject matter requires empathy, it may be better to have the audience play the game directly rather than deliver it through a game walkthrough to achieve the goal.

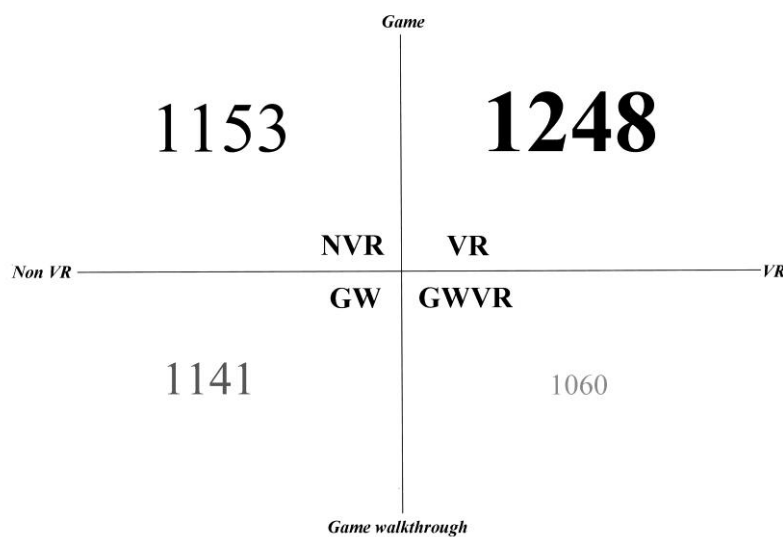


Figure 4. The total score a for each group

3.4. Empathy Check

As shown in Figure 5, overall, group NVR tends to start high and end low. Group VR and Group GW have very similar fluctuating trends, both starting high and ending low until stage 3, but increasing in empathy until stage 4. Group GWVR shows a steep decrease from stage 1 to stage 2, and a slower decrease in the other stages. The scores for NVR and VR do not change so much, whereas the scores for GW and GWVR are more variable.

With the exception of stage 3, group VR consistently led the other three groups in empathy, followed by group NVR, then group GW, and finally group GWVR. This is the same ranking as for the correctness check. The ranking of the overall score for the empathy survey is VR>NVR>GW>GWVR.



Figure 5. Empathy for each stage

4. LIMITATIONS AND CONCLUSION

The self-report of the four groups in the recognition check was NVR>GW>VR>GWVR in order from highest to lowest. In the correctness check, the total score ranking of the four groups was: VR>NVR>GW>GWVR. The ranking for the empathy check was the same as the correctness check: VR>NVR>GW>GWVR. Negative emotional keywords such as "scary" appear frequently in the recordings. A number of studies have confirmed that empathy plays a role in the construction of our memories [24, 25]. Coupled with the fact that it is an experimental result, it further proves that empathy is positively correlated with memory correctness.

In other words, Group VR, which ranked first in both the empathy and correctness surveys, and the VR serious game with a storyline are the means by which recipients can remember the most information.

Group NVR's sample narrative VR serious game is also a good medium for environments where VR conditions are not feasible or where VR device Ownership rate is a consideration. Although

slightly inferior to Group NVR, the Group GW data was very close to Group NVR in all surveys and the selection trends were similar. If the recipients want to memorize a single message or multiple messages, or if they value the convenience of game walkthroughs, GW is considered more suitable than NVR. However, group GW scored lower in questions that required subjective thinking, if the receiver is required to think subjectively, GW is less suitable.

Finally, GWVR, which is also a game walkthrough sample, does not perform well. In particular, compared to the similarity between group NVR and group GW, group GWVR and group VR have large differences in each survey. According to additional interviews with subjects in this group, the major reason for this is that the shaking lens distracts the subjects' attention and the subjects are not used to the walkthrough of VR game. Therefore, the walkthrough of VR game is not a suitable medium to make the receiver memorize the information.

In the case of game media, it is known from previous studies that empathy is positively correlated with memory correctness[26]. Based on the above data, this rule is not completely true for game walkthrough media. Although the group GW game walkthroughs can empathize to some extent with the game walkthrough players about their gaming experiences, they are limited compared to actual game plays, especially when it comes to the questions they have to think about.

Researchers believe that this is because the quality of empathy that the receiver feels differs from the empathy that is guided by the game walkthrough players. Game walkthroughs often allow the receiver to experience the game by empathizing with the medium of the game walkthrough players. A game walkthrough, on the other hand, allows the viewer to empathize directly with the game's protagonist; in a game walkthrough, the player is constantly verbalizing his or her feelings as the game progresses, thus reducing the need for the GW viewer to perceive his or her own feelings. Therefore, group GW is inferior to group NVR in terms of the need to think and construct their own words, which has lost its empathic mediation.

There are also limitations to this study. One is that the data reads that while the percentage of correct answers for GW group is not bad, it is slightly lower than NVR group. This is most likely due to the fact that as a Video medium, games walkthroughs are still less interactive than game media. If the comments were used to increase the participation and interaction of the recipients, a higher percentage of correct answers might be obtained. At the same time, however, some of the comments undermine the convenience of the game walkthrough, as recipients who are eating have to put down their cutlery and use both hands to control the game. Therefore, the viewers of the game walkthrough are not able to empathize in a positive way through the game walkthrough players, or the accuracy and precision of the game content is not guaranteed due to the lack of interactivity of the visual media. In order to increase interactivity, the game walkthrough needs to sacrifice convenience. There may be ways to increase interactivity while maintaining the convenience of the game walkthrough, such as voice input of comments. However, it is difficult to guarantee the interactivity and convenience of the game walkthrough in the current mainstream way. Second, the sample in this study is narrative oriented and has a weak gaming aspect in comparison. Different samples may yield different results, and subsequently, a sample that focuses more on gaming than narrative will be the next research topic.

The results of this study indicate that serious games walkthrough is a viable educational option outside the classroom, especially given the prevalence and convenience. With the rapid development of technology, students' interests are constantly changing, According to self-determination theory, education can be constantly updated to stimulate students' interest in learning. On the other hand, game walkthrough, a byproduct of educational gamification, give students more options. Students will be able to experience educational games in class, and also learn outside the classroom by watching game walkthrough anytime, anywhere.

ACKNOWLEDGEMENTS

The authors would like to thank everyone, just everyone!

REFERENCES

- [1] Zimmerling, E.; Höllig, C.E.; Sandner, P.G.; Welp, I.M. Exploring the Influence of Common Game Elements on Ideation Output and Motivation. *J. Bus. Res.* 2019, 94, 302–312.
- [2] Loganathan, P.; Talib, C.; Thoe, N.; Aliyu, F.; Zawadski, R. Implementing Technology Infused Gamification in Science Classroom: A Systematic Review and Suggestions for Future Research. *Learn. Sci. Math.* 2019, 14, 60–73.
- [3] Green, C. S., and Bavelier, D. (2007). Action video game experience alters the spatial resolution of vision. *Psychol. Sci.* 18, 88–94. doi: 10.1111/j.1467-9280.2007.01853.x
- [4] Dye, M. W. G., Green, C. S., and Bavelier, D. (2009). Increasing speed of processing with action video games. *Curr. Dir. Psychol. Sci.* 18, 321–326. doi: 10.1111/j.1467-8721.2009.01660.x
- [5] Feng, J., Spence, I., and Pratt, J. (2007). Playing an action videogame reduces gender differences in spatial cognition. *Psychol. Sci.* 18, 850–855. doi: 10.1111/j.1467-9280.2007.01990.x
- [6] Green, C. S., Pouget, A., and Bavelier, D. (2010). Improved probabilistic inference, as a general learning mechanism with action video games. *Curr. Biol.* 20, 1573–1579. doi: 10.1016/j.cub.2010.07.040
- [7] Boot, W. R., Kramer, A. F., Simons, D. J., Fabiani, M., and Gratton, G. (2008). The effects of video game playing on attention, memory, and executive control. *Acta Psychol. (Amst.)* 129, 387–398. doi: 10.1016/j.actpsy.2008.09.005
- [8] Hu, J. Gamification in Learning and Education: Enjoy Learning Like Gaming. *Br. J. Educ. Stud.* 2020, 68, 265–267
- [9] Michael Smith.” From Theory To Common Practice: Consumer Neuroscience Goes Mainstream”.(2016)<https://www.nielsen.com/us/en/insights/article/2016/from-theory-to-common-practice-consumer-neuroscience/>
- [10] Tom Trabasso, Paul van den Broek, Causal thinking and the representation of narrative events, *Journal of Memory and Language*, Volume 24, Issue 5, 1985, Pages 612-630, ISSN 0749-596X.
- [11] Harman, J., Brown, R., & Johnson, D. (2017). Improved memory elicitation in virtual reality: New experimental results and insights. In R. Bernhaupt, G. D. Anirudha, J. Devanuj, K. Balkrishan, J. O’Neill, & M. Winckler (Eds.), *IFIP Conference on Human–Computer Interaction* (pp. 128–146).
- [12] Ruddle, R. A., Volkova, E., Mohler, B., & Bühlhoff, H. H. (2011). The effect of landmark and body-based sensory information on route knowledge. *Memory & Cognition*, 39(4), 686–699.
- [13] Xiaohan Feng, Makoto Murakami, “COMBINING OF NARRATIVE NEWS AND VR GAMES: COMPARISON OF VARIOUS FORMS OF SERIOUS GAMES” *Signal & Image Processing : An International Journal (SIPIJ)*, Volume 12, Number 5, 2021-10
- [14] Stark, Chelsea. "Who Wants to Watch Other People Play Video Games? Millions on Twitch". *Mashable*. Retrieved 2017-05-10.
- [15] Richter, G.; Raban, D.R.; Rafaeli, S. Studying Gamification: The Effect of Rewards and Incentives on Motivation. In *Gamification in Education and Business*; Springer: Cham, Switzerland, 2015; pp. 21–46
- [16] Ryan, R.M.; Deci, E.L. Intrinsic and Extrinsic Motivations: Classic Definitions and New Directions. *Contemp. Educ. Psychol.* 2000, 25, 54–67.
- [17] Snyder, David (2017). *Speedrunning: Interviews with the Quickest Gamers*. McFarland Publishing. p. 19. ISBN 978-1-4766-7080-5.
- [18] "RAG-Longplay Announcement". *Recorded Amiga Games*. Archived from the original on July 24, 2011. Retrieved March 3, 2008.
- [19] White, Patrick (2013-04-18). "Fan fiction more creative than most people think". *Kansas State Collegian*. Retrieved 2013-04-21.
- [20] Clement ,Virtual reality (VR) and augmented reality (AR) device Ownership rate and purchase intent among consumers in the United States as of 1st quarter 2017, by gender.(2021)
- [21] Clement ,Distribution of video gamers in the United States from 2006 to 2020, by gender.(2021)

- [22] Cichoń, E., Gawęda, Ł., Moritz, S. et al. Experience-based knowledge increases confidence in discriminating our memories. *Curr Psychol* 40, 840–852. <https://doi.org/10.1007/s12144-018-0011-8>, (2021)
- [23] Xiaohan Feng, Makoto Murakami, “COMBINING OF NARRATIVE NEWS AND VR GAMES: COMPARISON OF VARIOUS FORMS OF SERIOUS GAMES” *Signal & Image Processing : An International Journal (SIPIJ)*, Volume 12, Number 5, 2021-10
- [24] Spreng, R. N., & Grady, C. L. Patterns of Brain Activity Supporting Autobiographical Memory, Propection, and Theory of Mind, and Their Relationship to the Default Mode Network. *Journal of Cognitive Neuroscience*, 22(6), 1112–1123. <https://doi.org/10.1162/jocn.2009.21282>, (2009)
- [25] Spreng, R. N., Mar, A. R., & Kim, A. S. N. The Common Neural Basis of Autobiographical Memory, Propection, Navigation, Theory of Mind, and the Default Mode: A Quantitative Meta-analysis. *Journal of Cognitive Neuroscience*, 21(3), 489–510. <https://doi.org/10.1162/jocn.2008.21029> ,(2009)
- [26] Xiaohan Feng, Makoto Murakami, “COMBINING OF NARRATIVE NEWS AND VR GAMES: COMPARISON OF VARIOUS FORMS OF SERIOUS GAMES” *Signal & Image Processing : An International Journal (SIPIJ)*, Volume 12, Number 5, 2021-10

3DHERO: AN INTERACTIVE PUZZLE GAME PLATFORM FOR 3D SPATIAL AND REASONING TRAINING USING GAME ENGINE AND MACHINE LEARNING

David Tang¹ and Yu Sun²

¹Irvine High School, 4321 Walnut Avenue, Irvine, CA 92604

²California State Polytechnic University,
Pomona, CA, 91768, Irvine, CA 92620

ABSTRACT

The well-known puzzle game Tetris, where arrangements of 4 squares (tetrominoes) fall onto the field like meteors, has been found to increase the brain's efficiency [1]. Many variations came into existence ever since its invention. Sometimes, the leveling can become a double-edged sword, so this game is essentially a Zen mode without a leveling system. This game is built for people who want to play a 3D version of Tetris at a speed they themselves have set. This paper designs a game to exercise spatial visualization. This study uses a Unity/C++-based game [2]. This game will be tested by kids on the autism spectrum, and we will conduct a qualitative evaluation of the approach.

No results have been shown yet, and that is due to the fact that this study is still a work in progress. I am trying to make the game comply with the latest Tetris design guidelines that I can find online (that is, the 2009 guideline).

KEYWORDS

Tetris, Spatial visualization, 3-Dimensional Perception.

1. INTRODUCTION

Tetris is a puzzle game created by the famous Soviet-American game developer Alexey Pajitnov and released in 1984 [3]. Developed in the Soviet Academy of Sciences in Moscow, Tetris was based on the famous pentomino puzzles Pajitnov liked to play with when he was a child [4]. He adapted the game to Cold War-era hardware and tweaked the game by reducing pentominoes to tetrominoes (hence the name, a portmanteau of “tennis”, one of Pajitnov’s favorite sports, and “tetra”, meaning “four” in Latin) and creating a playing field where tetrominoes would fall like meteorites [5]. Pajitnov and his team realized that the game would end too quickly without a key feature: making rows disappear whenever players filled them up. People that Pajitnov had worked with were attracted to the game, and the game is still popular today, even leading some to make variants with special twists in them, including but not limited to “Not Tetris”, with a physics engine and free-rotating tetrominoes; and a 3D version developed by T&E Soft for the Virtual Boy. It has inspired competitions to see who can earn the highest score. People have placed tetrominoes in specific arrangements at the beginning of their game to score more points.

Research from Mind Research Lab in Albuquerque on the original game has led to research on variations of the game, and its effects on autism [11]. This game incorporates features found in numerous other games before it. The only feature setting it apart from other games is a “central cube rotation system”.

There already exists the aforementioned Virtual Boy version, and another game called Blockout. Though Blockout has an indication for the height of the playing field, it only provides a top view of the playing field [10]. However, this Unity remake has a full 3D view of the playing field, allowing people to strategize where their piece will land using the ghost piece.

The pre-existing Tetris research involves the original 2D game [6].

In this paper, we follow the same line of research by ... Our goal is for players of this game to visualize arrangements of cubes spatially. Our method is inspired by Alexey Pajitnov’s Tetris and some other 3D Tetris-based games. There are some good features of Unity and C++. First, Unity is the second-most used game engine on Steam products. Second, we added more and more Unity plugins to the game.

The differences between my method and the other Tetris research project are that this paper focuses on Tetris as it relates to autism, and that this project uses an unofficial self-made 3D version.

The rest of the paper is organized as follows: Section 2 gives the details on the challenges that we met during the experiment and designing the sample; Section 3 focuses on the details of our solutions corresponding to the challenges that we mentioned in Section 2; Section 4 presents the relevant details about the experiment we did, following by presenting the related work in Section 5. Finally, Section 6 gives the conclusion remarks, as well as pointing out the future work of this project.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Implementing the Ghost Piece

The Ghost piece is a prediction of the landing position of a Tetromino if allowed to drop into the playing field. It is intended to reduce misdrops, especially for beginners and high-speed players.

According to the Tetris Wiki, the Ghost piece, or ghost for short, also called shadow or (in Arika games) Temporary Landing System (TLS), is a representation of where a tetromino or other piece will land if allowed to drop into the playfield [7]. It is generally colored fainter than the falling piece and the blocks in the playfield. As the player moves the falling piece, the ghost piece moves below it; when the piece falls far enough that it overlaps the ghost piece, the falling piece is always drawn in front. Older games did not have a ghost piece, but all games that conform to the Tetris Guideline allow the player to use a ghost piece at all times, and Dr. Mario for Nintendo 64 has a ghost piece as well. The ghost piece reduces the number of misdrops, especially for beginners or for high-speed players who use hard drop, but some players who are migrating from games without a ghost piece have trouble adjusting to the ghost piece when they fail to distinguish it from blocks in the playfield.

2.2. The Free-Rotating Camera

Blockout's camera is not free-rotating, and only gives a top view of the playing field. If implemented, it may block players' vision of the space below overhang. Instead, this Unity game features a plugin called Cinemachine. The camera rotates around an orbit point (which in this case refers to the center of the playing field)

3. SOLUTION

All scripts for this 3D Tetris game (working title) were coded in C++ [8].

The game is set on a black background with a white floor. Like in 3DT and Blockout, the goal is to clear planes [9]. Since the tetracubes rotate around one singular mino, T-Spin singles and doubles are possible, but not T-Spin triples and mini T-spins.

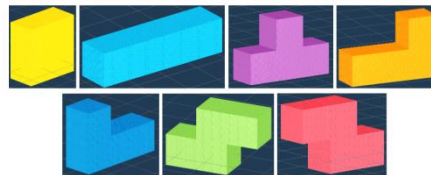


Figure 1. For the seven one-sided tetrominoes, I kept the colors for their tetracube counterparts. Top (left to right): O, I, T, and L. Bottom (left to right): J, S, and Z. L and J are chiral pairs in 2D, but not in 3D. Same applies to S and Z

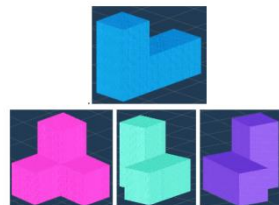


Figure 2. Left to right: The tripod, and the 3D chiral pair [left arm and right arm]. Those are new to the game because they do not exist in 2D

To create these arrangements, start with one cube, then clone it three times and move the clones until the shape is made.

Shown below is the code of the current tetracube spawner.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.Events;
using System;
using Random = UnityEngine.Random;
using UnityEngine.InputSystem;

public class RandomPieceSpawner : MonoBehaviour
{
    [Header("Internal Info")]
```

```

[SerializeField] int current = 0;
[SerializeField] bool isRandomized = true;
public List<TetraminoGeneratorUpdater> piecesList = new();
[SerializeField] float waitTimer = 0.5f;
[SerializeField, Tooltip("The amount to divide speed by to make the timer not decrease as
fast.")]
float speedInverseMultiplier = 2;
[SerializeField] float minimumWaitTime = 0.5f;

[Header("Extrenal Info")]
public GameObject piecesParent = null;
public TetraminoGenerator tetGen;

public TetraPieceScript currentPiece = null;

public GameObject hintPrefab;
// public PieceKeyboardControl defaultControl = new();
[SerializeField] InputActionAsset controls;
[Header("Events")]
public UnityEvent OnGameOver = new ();
public UnityEvent OnPieceSpawn = new ();

// Private Variables
private bool gameEnded = false;

void Start()
{
    Random.InitState(Random.Range(Int32.MinValue, Int32.MaxValue));
    if(piecesList.Count == 0)
    {
        Debug.LogError("Pieces List is Empty!");
        return;
    }
    tetGen?.UpdateGenerator(piecesList[current]);
}

// Update is called once per frame
void Update()
{
    if (gameEnded) return;

    if(currentPiece == null || currentPiece.CheckIfStopped)
    {
        SpawnPiece();
        OnPieceSpawn?.Invoke();
    }
}

public int RandomIndex()
{
    return Random.Range(0,piecesList.Count);
}

public void SpawnPiece()

```

```

{
    current = isRandomized ? RandomIndex(): current;
    tetGen?.UpdateGenerator(piecesList[current]);

    var piece = tetGen?.GenerateTetramino();
    piece.transform.position = Tetra3DGrid.ForceIntoGridPosition(transform.position);
    if(!Tetra3DGrid.CheckMovementOK(piece.transform.position, Vector3.zero))
    {
        gameEnded = true;
        currentPiece = null;
        OnGameOver?.Invoke();
        return;
    }

    piece.transform.parent = piecesParent != null ? piecesParent.transform : this.transform;

    Rigidbody rb = piece.AddComponent<Rigidbody>();
    rb.useGravity = false;
    rb.isKinematic = true;

    TetraPieceScript pieceScript = piece.AddComponent<TetraPieceScript>();
    // pieceScript.OverrideControls(defaultControl);
    currentPiece = pieceScript;
    pieceScript.SetControls(controls);
    pieceScript.OverrideTimings(newDropTimer: waitTimer);

    pieceScript.OnUnableToRegister.AddListener(this.GameOver);

    TetraPieceHints pieceHint = piece.AddComponent<TetraPieceHints>();
    pieceHint.pieceRef = pieceScript;
    pieceHint.hintPrefab = hintPrefab;
}

public void GameOver()
{
    OnGameOver?.Invoke();
}

// Increase speed by decreasing waitTime
public void IncreaseSpeed(float increaseAmount){
    waitTimer -= increaseAmount/speedInverseMultiplier;
    waitTimer = Mathf.Max(waitTimer, minimumWaitTime);
}

// Set wait timer by level
public void SetSpeedByLevel(int level){
    waitTimer = Mathf.Pow((0.8f - ( level - 1) * 0.007f ), level - 1);
}

public void SetWaitTimer(float timerAmount){
    waitTimer = Mathf.Max(timerAmount, minimumWaitTime);
}
}

```

This segment of code always returns a random integer from 0 to the length of the piece list. Problem is, this algorithm generates piece droughts.

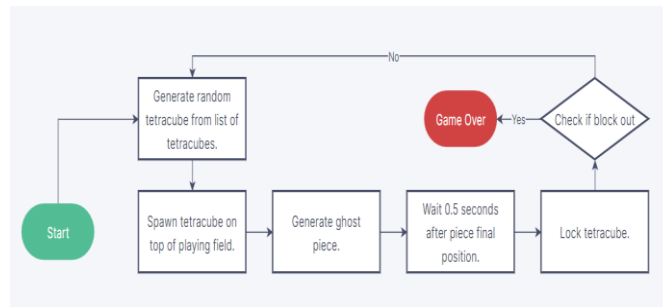


Figure 3. A flow chart describing how the randomizer and spawner in RandomPieceSpawner.cs works

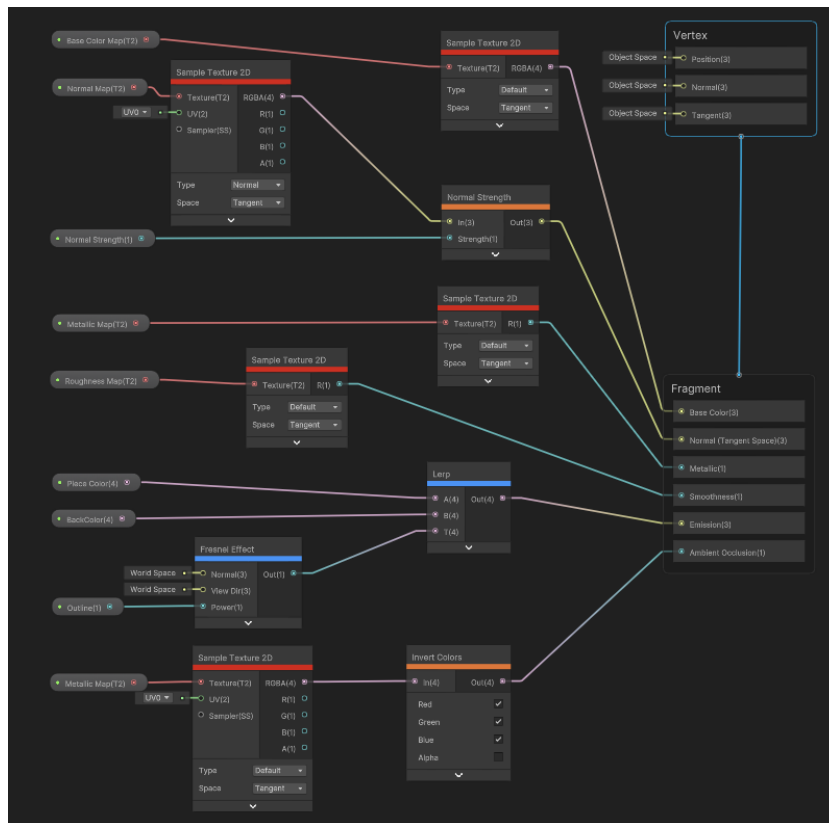


Figure 4. The Shader Graph for the Tetracubes

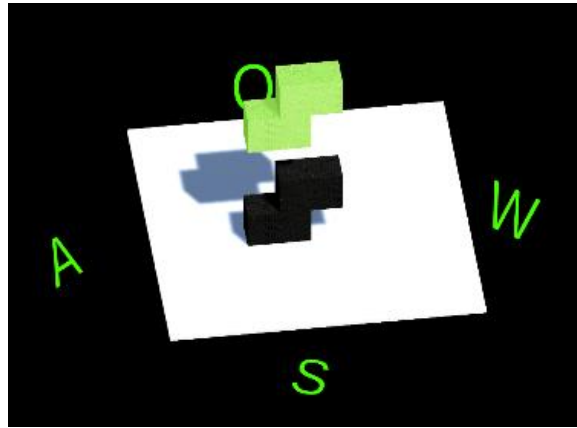


Figure 5. The S-tetrahedron, otherwise known as the N or the Z, with the shader as mentioned in Figure 4. Below the actual S-tetrahedron (in green) is a ghost piece (in black). The gray shadows are from the light source

Unlike in BlockOut, however, the camera is free-rotating, allowing players to view a full 3D view.

The code for the camera is as follows, and consists of two scripts: CameraFixedRotator.cs, which handles the rotation of the camera around a singular point; and CameraOrbitControls.cs, which handles the controls required to rotate the camera.

```
(CameraFixedRotator.cs)
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.InputSystem;

public class CameraFixedRotator : MonoBehaviour
{
    public Key cwRotationKey = Key.Comma;
    public Key acwRotationKey = Key.Period;

    [SerializeField]
    private int xRemain = 1;
    [SerializeField]
    private int zRemain = -1;

    // Update is called once per frame
    void Update()
    {
        if(Keyboard.current[cwRotationKey].wasPressedThisFrame)
        {
            transform.rotation = Quaternion.Euler(transform.eulerAngles + Vector3.up*90);
            transform.position = new Vector3(transform.position.x * xRemain, transform.position.y,
transform.position.z * zRemain);
            xRemain *= -1;
            zRemain *= -1;
        }
        else if(Keyboard.current[acwRotationKey].wasPressedThisFrame)
        {
            transform.rotation = Quaternion.Euler(transform.eulerAngles + Vector3.down*90);
            transform.position = new Vector3(transform.position.x * zRemain, transform.position.y,
transform.position.z * xRemain);
            xRemain *= -1;
            zRemain *= -1;
        }
    }
}
```

```
(CameraOrbitControls.cs)
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.InputSystem;
using Cinemachine;

public class CameraOrbitControls : MonoBehaviour
{
    [SerializeField] CinemachineFreeLook orbitCamera;
    [SerializeField] float orbitSpeed = 1f;
    [SerializeField] bool invertYValue = false;

    private void Awake()
    {
        if(orbitCamera == null){
            orbitCamera = GetComponent<CinemachineFreeLook>();
        }
    }

    public void OnOrbitMove(InputAction.CallbackContext context)
    {
        Vector2 rotation = context.ReadValue<Vector2>().normalized;

        rotation.y = invertYValue ? -rotation.y : rotation.y;
        rotation.x = rotation.x * 180;

        orbitCamera.m_XAxis.Value = rotation.x * orbitSpeed * Time.deltaTime;
        orbitCamera.m_YAxis.Value = rotation.y * orbitSpeed * Time.deltaTime;
    }
}
```

The ghost piece is integrated into the game as a script called TetraPieceHints.cs.


```

using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class TetraPieceHints : MonoBehaviour
{
    public float transparency = 0.25f;
    public TetraPieceScript pieceRef;
    public GameObject hintPrefab;

    private List<GameObject> hintHolder = new List<GameObject>();
    private bool isStopped = false;
    void Start()
    {
        pieceRef = GetComponent<TetraPieceScript>();
        foreach(Transform child in transform)
        {
            hintHolder.Add(Instantiate(hintPrefab, child));
        }
    }

    void Update()
    {
        if(isStopped) return;

        isStopped = pieceRef.CheckIfStopped;
        if(!isStopped && pieceRef != null && hintHolder.Count > 0)
        {
            UpdateAllPieces();
        }
        if(isStopped)
        {
            foreach(GameObject child in hintHolder)
            {
                Destroy(child);
            }
            hintHolder.Clear();
        }
    }

    public void UpdateAllPieces()
    {
        for(int moveAmount = Tetra3DGrid.gridHeight+1; moveAmount > 0; moveAmount--)
        {
            bool allWorks = true;
            foreach (Transform child in this.transform)
            {
                if(!Tetra3DGrid.CheckMovementOK(child.position, Vector3.down*moveAmount))
                {
                    allWorks = false;
                    break;
                }
            }
            if(allWorks)
            {
                foreach(GameObject hint in hintHolder)

```

```
        {
            hint.transform.position = (Vector3.down * moveAmount)
                + hint.transform.parent.position;
        }
        break;
    }
}

public Vector3 FindCollisionPoint(Transform child)
{
    Vector3 direction = Vector3.down;
    Vector3 highestHit = Vector3.negativeInfinity;
    foreach(var collision in Physics.RaycastAll(child.position, direction,
Tetra3DGrid.gridHeight*2))
    {
        if(collision.collider.GetComponentInParent<TetraPieceScript>() != pieceRef
|| collision.collider.CompareTag("Finish"))
        {
            if(Mathf.Ceil(collision.point.y) > highestHit.y)
            {
                highestHit = collision.point;
            }
        }
    }
    return highestHit;
}
```

4. EXPERIMENT

4.1. Experiment 1

To test whether our games are really useful in helping children with autism focus, we found 9 children with autism of various ages from California. Divide them into 3 different groups. We separately calculated the playtime of children of different ages when playing this game. It was found that after a period of practice, children's attention time was significantly longer when playing the game. The results show that the game is most effective for 7-10-year-olds. 7-10-year-olds can only play the game continuously for 15.1 seconds at first, After a period of time, it can reach more than 30 seconds.

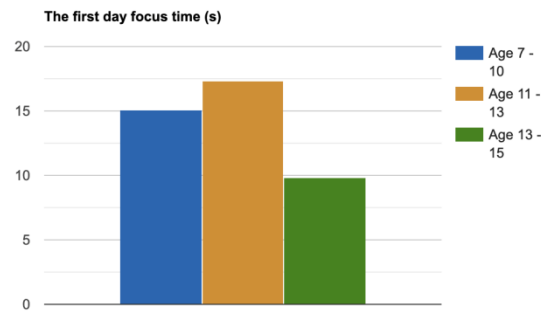


Figure 6. The first day focus time graph

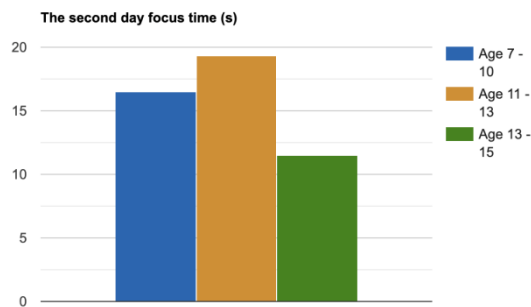


Figure 7. The second day focus time graph

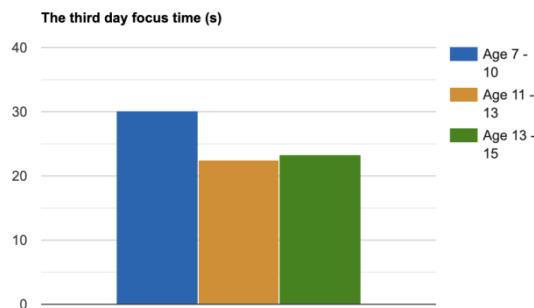


Figure 10. Result of experiment 2

5. RELATED WORK

The idea of a 3D version of Tetris wasn't new. There exists a Microsoft DOS game called Blockout, which implements polycubes of various orders, a ghost piece, and a top-view camera [12]. This project is similar to Blockout in the way that there is no next queue and that it also has the ghost piece implemented. The size of the playing field, and also the set of polycubes used, can be chosen by the player.

However, this project restricts the pieces available to the 8 one-sided tetracubes, and the camera is free-rotating. Also, the pieces are not colored based on stack height in this incarnation. They are rather colored based on what shape they are and what orientation they spawn in.

On to 3D Tetris, the Virtual Boy port by T&E Soft. The polycubes are only red due to the Virtual Boy only being able to display in black and red [13]. It, like Blockout, involves polycubes of orders 1-5, but also includes arrangements of pseudo-polyominoes extruded by 1 unit, and even arrangements where the cubes themselves are not connected at all. Whenever players max out, the bottom-most row gets cut and the playing field gets shorter. There is a layer-by-layer view of the playing field in 3D Tetris.

This project, however, has an 8x12x8 playing field in contrast to 3D Tetris' 5x5x5 playing field, and stays true to its title by restricting the pieces to the 8 one-sided tetracubes.

6. CONCLUSIONS

This paper talks about a 3D remake of Tetris (and by extension, BlockOut) while trying to adhere to the design guidelines as much as possible. It also involved elements from the already-existing Virtual Boy game. The game teaches people how to deal with piece droughts.

Interestingly, there exists a maxout TAS of NES Tetris without any I pieces [14].

The game was made by the Tetris design guidelines of 2009. There is no next queue. There is no visible grid. There is no soft dropping. Hard dropping does not generate the next piece instantly.

As for the future work, we plan to do or add the following: a warning system that warns players whenever they are about to "top out"; an awards system; T-spins and Mini T-spins; and a symmetrical SRS rotation system [15].

We are currently in the process of implementing these features into the game.

Here is a segment of pseudocode for the Random Generator and the Next queue, though they have not been implemented into the game itself.

- Make the entire next queue an ArrayList of tetracubes.
- Generate a random permutation of the bag of tetra cubes and add them to the next queue once there are (visible length of next queue) tetracubes left in the next queue.

REFERENCES

- [1] Demaine, Erik D., Susan Hohenberger, and David Liben-Nowell. "Tetris is hard, even to approximate." International Computing and Combinatorics Conference. Springer, Berlin, Heidelberg, 2003.
- [2] Mattingly, William A., et al. "Robot design using Unity for computer games and robotic simulations." 2012 17th International Conference on Computer Games (CGAMES). IEEE, 2012.
- [3] Flom, Landon, and Cliff Robinson. "Using a genetic algorithm to weight an evaluation function for Tetris." Colorado State University, Tech. Rep. Luke (2004).
- [4] Fortescue, Stephen. "The Russian Academy of sciences and the Soviet Academy of sciences: Continuity or disjunction?." *Minerva* 30.4 (1992): 459-478.
- [5] Sears, Derek WG, and Robert T. Dodd. "Overview and classification of meteorites." *Meteorites and the early solar system* (1988): 3-31.
- [6] Ak, Oguz, and Birgul Kutlu. "Comparing 2D and 3D game-based learning environments in terms of learning gains and student perceptions." *British Journal of Educational Technology* 48.1 (2017): 129-144.
- [7] Zhao, Tianqu, and Hong Jiang. "Landing system for AR. Drone 2.0 using onboard camera and ROS." 2016 IEEE Chinese Guidance, Navigation and Control Conference (CGNCC). IEEE, 2016.
- [8] Nitsche, Michael. *Video game spaces: image, play, and structure in 3D worlds*. MIT Press, 2008.

- [9] Murdock, Calvin, et al. "Blockout: Dynamic model selection for hierarchical deep networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
- [10] Prabhakar, Balaji, Nick McKeown, and Jean Mairesse. "Tetris models for multicast switches." Proc. of the 30th Annual Conference on Information Sciences and Systems, Princeton. 1996.
- [11] Jackson, Wallace. "Implementing Game Audio Assets: Using the JavaFX AudioClip Class Audio Sequencing Engine." Beginning Java 8 Games Development. Apress, Berkeley, CA, 2014. 323-341.
- [12] Agah, Afrand, and Sajal K. Das. "Preventing DoS attacks in wireless sensor networks: A repeated game theory approach." Int. J. Netw. Secur. 5.2 (2007): 145-153.
- [13] Li, Yuzhe, et al. "SINR-based DoS attack on remote state estimation: A game-theoretic approach." IEEE Transactions on Control of Network Systems 4.3 (2016): 632-642.
- [14] Gibbons, William. "Blip, bloop, Bach? Some uses of classical music on the Nintendo entertainment system." Music and the Moving Image 2.1 (2009): 40-52.
- [15] Bourne, S., and P. Bruggen. "Examination of the Distinctive Awards System." Br Med J 1.5950 (1975): 162-165.

FRAME SIZE OPTIMIZATION USING A MACHINE LEARNING APPROACH IN WLAN DOWNLINK MU-MIMO CHANNEL

Lemlem Kassa¹, Jianhua Deng¹, Mark Davis² and Jingye Cai¹

¹School of Information and Software Engineering, University of Electronic Science and Technology China (UESTC), Chengdu 610054, China

²Communication Network Research Institute (CNRI), Technological University, D08 NF82 Dublin, Ireland

ABSTRACT

IEEE 802.11n/ac introduced frame aggregation technology to accommodate the growing traffic demand and increases the performance of transmission efficiency and channel utilization by allowing many packets to be aggregated per transmission which significantly enhances the throughput of WLAN. However, it is difficult to efficiently utilize the benefits of frame aggregation as stations in the downlink MU-MIMO channel have heterogeneous traffic demand and data transmission rate. As a result of this, wasted space channel time will be occurred that degrade transmission efficiency. In addressing these challenges, the existing studies have proposed different approaches. However, most of these approaches did not consider a machine-learning based optimization solution. The main contribution of this paper is to propose a machine-learning based frame size optimization solution to maximize the system throughput of WLAN in the downlink MU-MIMO channel. In this approach, the Access Point (AP) performs the maximum system throughput measurement and collects the “frame size-system throughput patterns” which contain knowledge about the effects of traffic condition, channel condition, and number of stations (STAs). Based on these patterns, our approach uses neural networks to correctly model the system throughput as a function of the system frame size. After training the neural network, we obtain the gradient information to adjust the system frame size. The performance of the proposed ML approach is evaluated over the FIFO aggregation algorithm under the effects of heterogenous traffic patterns for VoIP and Video traffic applications, channel conditions, and number of STAs.

KEYWORDS

Frame Size Optimization, Downlink MU-MIMO, WLAN, Network Traffic, Machine Learning, Neural Network, Throughput Optimization.

1. INTRODUCTION

Due to the advancement of wireless technologies, IEEE 802.11 based networks are becoming more popular, and different technologies have been introduced to improve throughput performance. Multi-user multiple-input multiple-output (MU-MIMO) is among the ones at the physical layer introduced by IEEE 802.11ac standard to accommodate the increasing demand of high data transmission rate by allowing a single Access Point (AP) supports simultaneous transmission up to a maximum of eight users at a time [1,2]. This is one of the most crucial technologies that has driven wireless local area networks (WLANs) toward the gigabit era. Moreover, the wireless medium has a high overhead in terms of bytes that can be higher than the

actual payload. To amortize these overheads such as the Medium Access control (MAC) and physical (PHY) headers, acknowledgments (ACK), backoff time, and inter-frame spacing, the standard also introduced a frame aggregation scheme which has contributed to a high data throughput by combining multiple frames, also known as MAC Service Data Units (MSDUs), into a single transmission unit [1]. The performance of WLAN depends on different performance factors such as frequency channel, modulation, and coding schemes, transmitter power, etc. at the PHY layer, and retry limit, frame size, contention window size, maximum number of backoffs, etc. at the MAC layer have a significant impact on the performance of WLAN. Optimizing these parameters would improve the system performance of WLAN. Frame size optimization is the main concern of this study. If a wireless frame size is large, a bit error would destroy the whole frame thus the frame success rate decreases, and also the throughput performance degrades [3]. On the other hand, if the frame size is shorter, the overhead frames such as MAC and PHY headers occupy a large portion of the transmitted frame thus degrade the transmission efficiency [3]. IEEE 802.11 standard specified a constant length aggregation strategy regardless of the traffic pattern and channel conditions. This contributes to the reduction of channel access overhead. However, utilizing the maximum aggregation size may not be optimal in all channel conditions and traffic patterns because it may lead to an increase in the delivery of error frames and retransmissions [4]. This phenomenon particularly degrades the performance of WLAN in downlink MU-MIMO channel when streams have heterogeneous traffic demands such that variable transmission rate among spatial streams causes space channel time. Therefore, determining the optimal frame size is significant to improve the system throughput of WLAN [3,4].

The development of smart devices, mobile applications, and wireless users' interaction with wireless communication systems are significantly increased and caused a massive amount of traffic being generated in the communication network [5]. These challenges have pushed the wireless networking industry to seek innovative solutions to ensure the required network performance. On the other hand, the release of the new IEEE 802.11 standards such as IEEE 802.11ax and IEEE 802.11ay, and 5G technologies expand the set of available communication technologies which compete for the limited radio spectrum resources in pushing for the need of enhancing their coexistence and more effective use of the scarce spectrum resources. In response to these performance demands, more recently, Machine Learning (ML) approaches have started to attract significant attention and are increasingly used in the context of wireless communication systems to generate a self-driven networks that can configure and optimize themselves by reducing human interventions [6-10]. The developments of mobile edge caching and computing technologies also made it possible for base stations (BSs) to store and analyze human behavior for wireless communication [8-10]. Therefore, the evolution toward learning-based data-driven network systems helps to develop and realize many of the promising benefits obtained from ML. ML is used to develop advanced approaches that can autonomously extract patterns and predict trends based on environmental measurements and performance indicators as input. Such patterns can be used to optimize the parameter settings at different protocol layers, e.g., PHY, and MAC layers [5, 6]. Several frame size optimization schemes are proposed to improve the throughput performance of WLAN. For instance, [11] proposed the adaptive frame size estimation scheme depending on the channel condition to improve the throughput performance of WLAN in the error-prone channel based on the Extended Kalman Filter. By studying the relationship between throughput and frame size, [12] illustrated that throughput is a monotonically increasing function of the frame size, i.e., the larger the frame size, the better the throughput. However, these approaches do not provide a machine-learning based optimization solution and the algorithms are not applicable in IEEE 802.11 MU-MIMO enabled WLAN. In considering the channel condition and contention effects in WLAN, [13] proposed a machine learning-based adaptive approach for frame size optimization, however, this approach is not applicable in MU-MIMO enabled WLAN. The main contribution of this paper is to propose a machine-learning based adaptive algorithm to

optimize the frame size that would maximize the system throughput in the WLAN downlink MU-MIMO channel considering the effect of channel condition heterogeneous traffic patterns and number of stations. Thus, our proposed ML approach is significant as it can autonomously extract patterns and predict trends based on environmental measurements and performance indicators as input.

The rest of the paper is organized as follows, in Section 2, we introduce related works about the frame aggregation schemes and the performance challenges of multi-user transmissions in the WLAN downlink MU-MIMO channel. A detailed problem description of the proposed machine-learning approach is given in Section 3. In Section 4, results and discussions are presented to evaluate the performance of the proposed approach under various channel conditions, traffic models, and number of stations. Finally, the conclusions are given in Section 5.

2. RELATED WORK AND OUR MOTIVATION

In this section, some previous works and the effects of frame size determination approaches on the performance of WLAN are discussed mainly focusing on the downlink MU-MIMO channel.

2.1. Related Work

Frame size optimization problem has been studied by several researchers for IEEE 802.11 networks. For instance, employing a specific procedure of dynamically adjusting the frame size, [11] proposed a method that deals with frame size estimation based on the extended Kalman Filter for saturated networks. They derive the mathematical equation of throughput, which is a function of the frame size. The optimal frame size is obtained using differential calculus. Bianchi's Markov chain model studied the relationship between the throughput and frame size, in IEEE 802.11 WLANs [12]. However, the assumption of this work is ideal channel which is unrealistic. According to the simulation results, the throughput increases with the frame size, i.e., the larger the frame size, the better the throughput. A machine learning-based frame size optimization approach in considering both channel conditions and contention effects of users is proposed by [13]. According to the simulation results, the frame size optimization is effectively achieved to maximize the throughput performance of WLAN. However, this approach does not support the frame aggregation mechanism and the algorithm is not suitable for IEEE 802.11 MU-MIMO enabled WLAN. An adaptive algorithm for frame size optimization is proposed by [14] which allows an ARQ protocol to dynamically optimize the packet size based on estimates of the channel bit-error. The main strategy of this study is, to make estimates of the channel bit-error-rate, they consider the acknowledgment history, thus based on that the optimal packet size can be determined. However, this approach is not suitable for IEEE 802.11 WLAN environments.

Moreover, some studies contributed frame size aggregation schemes in WLAN downlink MU-MIMO channels to enhance the throughput performance [15-22]. The algorithm in [15] proposed a new approach aiming to enhance the system throughput performance of WLAN employing a dynamic adaptive aggregation selection scheme to determine the optimal length of the frame size in downlink MU-MIMO transmission. The effects of heterogeneous traffic demand among spatial streams are considered under the assumption of ideal channel. According to the simulation results, the maximum performance of system throughput performance and channel utilization is achieved. By extending the work of [15], an adaptive frame aggregation algorithm is proposed by [16] in considering the effect of transmission error. Moreover, a data frame construction scheme called DFSC [17] proposed to find the length of a Multi-User (MU) frame aiming to maximize the transmission efficiency by considering the status of buffers and transmission bit rates of stations in both uplink and downlink multiuser transmissions. However, this work did not

consider the effect of channel error that could reduce the transmission performance due to excessive retransmissions of frames received in error. A frame size-based aggregation scheme is proposed by [18] where the authors demonstrated that both the queueing length and number of active nodes have significant impacts on the system throughput performance. The main approach of this paper is to generate the same frame length in all spatial streams that could maximize the system throughput performance. [19] proposed a novel method to determine frame aggregation size in MU-MIMO channel to improve channel utilization in considering delay data frames wait in transmission queues. Some works in the literature have also been studied focusing on the padding problem. According to [20,21], they improved transmission efficiency in the downlink MU-MIMO channel by replacing padding bits with data frames from other users in one stream to fill the space of frame padding violating the rules of MU transmissions. However, these approaches increased the complexity of both the transmission and reception process in wireless communication which requires modification of the standard to allow the transmission to multiple destinations within a special stream. A frame duration-based frame aggregation scheme is proposed in [22] by employing user selection criteria by providing high priority to the MT expecting high throughput in the next MU-MIMO transmission and having a large amount of data while reducing signaling overhead. The main approach of this study is, by equalizing the transmission time of all spatial streams in all MTs according to their Modulation and Coding (MCS) level they could achieve the maximum system throughput and minimize space channel time in WLAN the downlink MU-MIMO channel. Although all the above proposals contributed several schemes to enhance the performance of WLAN, none of them has proposed a machine-learning based optimization solutions. To the best of our knowledge, there is little research explored with the use of ML techniques to tackle frame size optimization problems in WLAN. In contrast to these approaches our work attempt to propose a machine learning-based adaptive approach for frame size optimization in the WLAN downlink MU-MIMO channel.

2.2. Motivation for this Work

The dynamic adaptive frame aggregation selection scheme can maximize the system throughput performance of WLAN while enhancing system throughput performance by minimizing space channel time. However, this approach does not consider a machine learning-based optimization solution. The motivation of this work comes with the aim of extending the previous work [16] to contribute a machine learning base adaptive approach for frame size optimization to maximize the system throughput performance of WLAN in the downlink MU-MIMO channel. Thus, we can generate a self-driven networks that can configure and optimize themselves by reducing human interventions. Moreover, for the growing diversification of services, users, and the constantly changing channel and traffic dynamics in a networking system, a ML solution is relevant and should be adopted in more effective ways to speed up the decision-making process [4–7].

3. PROPOSED APPROACH

In this topic, the problem definitions and the proposed machine learning approach are discussed.

3.1. Problem Definition

In this paper, we tackle the frame-size optimization problem using a machine-learning-based adaptive approach in considering the effects of traffic patterns, channel conditions, and number of stations in WLAN downlink MU-MIMO. In this approach, the simulation environment proposed by [16] is used to collect the “frame size–system throughput “patterns. The frame size represents the average offered traffic load in [Mbps] generated in the system by employing different traffic

models (Pareto, Weibull, or fractional Brownian Motion (fBM)) [15, 16]. System throughput is defined as the average system data rate at which the AP can successfully transmit to all receiving stations. The collected patterns contain knowledge about the effects of traffic patterns, channel conditions, and number of stations. Suppose frm is the frame size and Thr is the corresponding throughput. Based on these patterns, our approach uses neural networks to build the knowledge and accurately model the throughput Thr as a function with respect to the frame size frm . The neural network is a good approach to model a system effectively that may contain some noise [13, 23]. Thus, after the knowledge building, we obtain the gradient information from the neural networks and adaptively adjust the frame size based on the gradient information. In the formation of frame-size optimization problem, the throughput Thr is a complex function of the frame size frm under some channel conditions and traffic patterns and number of stations, i.e., $Thr = f(frm)$. The function f varies with the channel conditions, traffic pattern and number of stations in the network. Therefore, how the throughput Thr can be maximized by optimizing the frame size frm is the main focus of the problem in this study.

$$frm_{opt} = \underset{frm}{\operatorname{argmax}} Thr = \underset{frm}{\operatorname{argmax}} f(frm) \quad (1)$$

Therefore, the goal of this approach is to choose the optimal frame size that would maximize the objective function Thr . The objective function of this optimization problem defined as the throughput $Thr = f(frm)$. However, due to the dynamic effects of channel conditions and traffic patterns, it is difficult to analyze and obtain an accurate throughput function $f(frm)$ in all network conditions. Thus, we solved such an optimization problem by adopting the well-known gradient ascent algorithm [24]. Such that the local maximum of the throughput function $Thr = f(frm)$ can be found by adaptively adjusting frame size frm using gradient ascent, by taking steps that are proportional to the gradient. Suppose that, at the n^{th} time of adjustment, the frame size is $frm(n)$, and the throughput is $Thr(n)$. At the next time of adjustment, the frame size frm is set as:

$$frm(n+1) = frm(n) + \Delta frm(n) \quad (2)$$

Where $\Delta frm(n)$ depends on the gradient of the estimated throughput $Thr(n)$ with respect to $frm(n)$, i.e.,

$$\Delta frm(n) = \mu \frac{\partial Thr(n)}{\partial frm(n)} \quad (3)$$

The parameter μ is a variable adjustment rate heuristically selected for different network scenarios. Then, to solve the gradient problem $(\partial Thr(n)) / (\partial frm(n))$, a machine-learning-based adaptive approach is elaborated in the following sub section.

3.2. The Proposed Machine- Learning-based Adaptive Solution

Machine Learning (ML) is an innovative solution that can autonomously extract patterns and predict trends based on environmental measurements and performance indicators as input to provide self-driven intelligent network systems that can configure and optimize themselves. Under the effects of heterogeneous traffic demand among users and varying channel conditions in WLAN downlink MU-MIMO channels, achieving the maximum system throughput performance is challenging. Online learning (also called incremental learning) and offline learning (or batch learning) are types of learning strategies in machine learning [26]. In online learning, the algorithm updates its parameters after learning from each individual training instance i.e., it is feeded with individual data or mini-batches [25, 26]. This allows the learning algorithm keep learning on the fly, after being deployed as new data arrives. The weight changes in online

learning made at a given stage depend specifically only on the current training instance being presented and possibly on the current state of the model. When an online model has learned from new data instances, it no longer needs to use them and can therefore discard them. This can save a huge amount of memory space.

Whereas traditional machine learning is performed offline using offline learning which is the opposite of online learning [26]. On the contrary, in offline learning, the learning algorithm updates its parameters after consuming the whole batch, and the weight changes depending on the whole (training) dataset, defining a global cost function [26]. Therefore, in this study to cope with the effects of time-varying channel conditions and heterogeneous traffic patterns, the online machine learning strategy is employed to achieve the data collection, knowledge building, and frame-size adjustment kept online.

The proposed Multi-Layer Perception (MLP) ML approach consists of one hidden layer with four neurons and an output layer. The backpropagation algorithm only consists of two passes: 1) a forward pass and 2) a backward pass [27]. To obtain the gradient information $\frac{\partial Thr(n)}{\partial frm(n)}$ which is used to adjust the frame size, we add a third pass, i.e., the tuning pass as shown in Figure 1. In the proposed MLP approach, the backpropagation algorithm is used to adjust the network and minimize the error between the actual response and the desired (target). The detailed description of the tuning pass is provided as follows including a summary of notations in Table I.

3.2.1. Tuning Pass Strategies

The diagram shown in Figure 1 illustrated the signal flow of the tuning pass in the machine learning model to estimate the gradient $\frac{\partial Thr(n)}{\partial frm(n)}$, and the key to adjusting the frame size to maximize the throughput. The initial weight is denoted as w_{ij}^i in the neural network is randomly chosen. The synaptic weights that have been well adjusted in the backward pass are set as fixed in the tuning pass. An adaptive learning rate is adopted to improve the convergence speed [25].

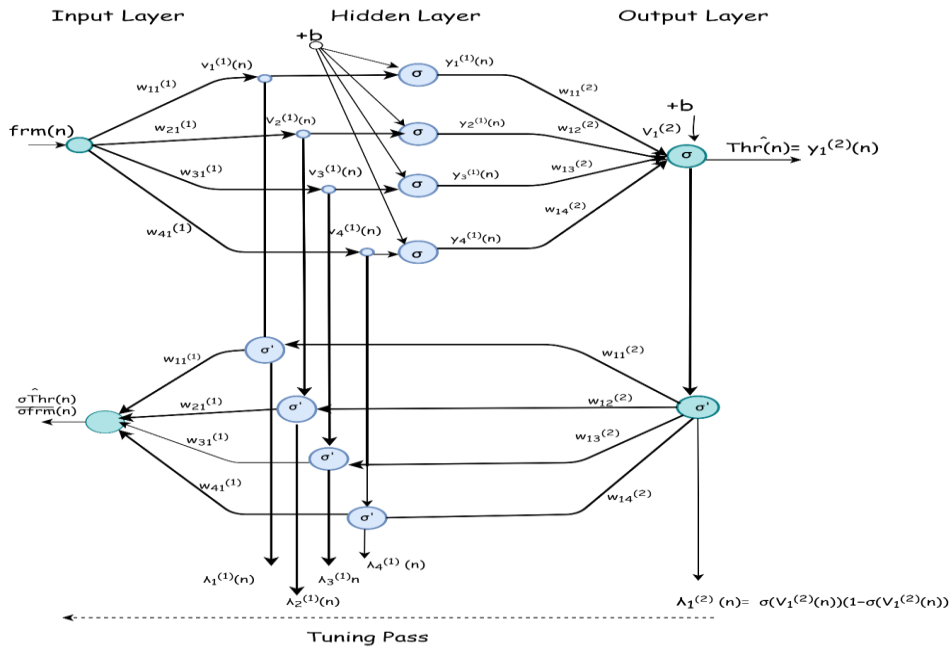


Figure 1. Flow chart of the proposed machine learning approach consisting of tuning pass, which depicts the derivation of the local gradients and the gradient for frame size adjustment.

In the following discussion, the procedure how the estimated gradient $\frac{\partial \widetilde{Thr}(n)}{\partial frm(n)}$ can be obtained is presented. Considering the hidden layer, the local gradient $\lambda_j^{(l)}(n)$ for the tuning pass is defined as follows:

$$\lambda_j^{(l)}(n) = \frac{\partial \widetilde{Thr}}{\partial v_j^{(l)}(n)} \quad (4)$$

Where $v_j^{(l)}$ in equation (4) is the weight sum of synaptic input plus bias of neuron j in layer l .

Similarly, considering the output layer, the local gradient $\lambda_1^{(2)}(n)$ is defined as follows:

$$\lambda_1^{(2)}(n) = \frac{\partial \widetilde{Thr}(n)}{\partial v_1^{(2)}(n)} = \partial'(v_1^{(2)}(n)) = \partial(v_1^{(2)}(n))(1 - \partial(v_1^{(2)}(n))) \quad (5)$$

While considering the hidden layer, the local gradient $\lambda_j^{(l)}(n)$ can be expressed as follows using the chain rule:

$$\begin{aligned} \lambda_j^{(1)}(n) &= \frac{\partial \widetilde{Thr}(n)}{\partial v_j^{(1)}(n)} = \frac{\partial \widetilde{Thr}(n)}{\partial v_1^{(2)}(n)} \cdot \frac{\partial v_1^{(2)}(n)}{\partial y_j^{(1)}(n)} \cdot \frac{\partial y_j^{(1)}(n)}{\partial v_j^{(l)}(n)} \\ &= \lambda_1^{(2)}(n) \cdot w_{1j}^{(2)}(n) \cdot \partial'(v_j^{(1)}(n)) \quad (6) \end{aligned}$$

Therefore, using the results (5) and (6), the gradient can be written as follows:

$$\begin{aligned} \frac{\partial \widetilde{Thr}(n)}{\partial frm(n)} &= \frac{\partial \widetilde{Thr}(n)}{\partial v_1^{(2)}(n)} \cdot \frac{\partial v_1^{(2)}(n)}{\partial frm(n)} \\ &= \lambda_1^{(2)}(n) \cdot \frac{\partial v_1^{(2)}(n)}{\partial frm(n)} \quad (7) \end{aligned}$$

Where $v_1^{(2)}(n)$, can be defined as $v_1^{(2)}(n) = \sum_{i=1}^4 w_{1j}^{(2)}(n) \cdot y_j^{(1)}(n)$. Thus, the second term at the rightmost side of equation (7) can be written as:

$$\begin{aligned} \frac{\partial v_1^{(2)}(n)}{\partial frm(n)} &= \sum_{i=1}^4 w_{1j}^{(2)}(n) \cdot \frac{\partial y_j^{(1)}(n)}{\partial frm(n)} \\ &= \sum_{i=1}^4 w_{1j}^{(2)}(n) \cdot \frac{\partial y_j^{(1)}(n)}{\partial v_j^{(1)}(n)} \cdot \frac{\partial v_j^{(1)}(n)}{\partial frm(n)} \\ &= \sum_{i=1}^4 w_{1j}^{(2)}(n) \cdot \partial'(v_j^{(1)}(n)) \cdot w_{j1}^{(1)}(n) \\ &= \sum_{i=1}^4 \lambda_1^{(2)}(n) \cdot w_{1j}^{(2)}(n) \cdot \partial'(v_j^{(1)}(n)) \cdot w_{j1}^{(1)}(n) \\ &= \sum_{i=1}^4 \lambda_j^{(1)}(n) \cdot w_{j1}^{(1)}(n) \quad (8) \end{aligned}$$

Therefore, the gradient $\frac{\partial \widetilde{Thr}(n)}{\partial frm(n)} = \sum_{i=1}^4 \lambda_j^{(1)}(n) \cdot w_{j1}^{(1)}(n)$ (9)

The derivation of the local gradients at each layer and the gradient $\frac{\partial \overline{Thr}(n)}{\partial frm(n)}$ is depicted in Figure 1. Based on the result from equation (9), the frame size frm is adjusted as shown in the equations in (2) and (3).

In general, Figure 2 illustrates the basic components and flow of the proposed ML approach. As shown in the figure, once the AP collects the instant learning dataset from the simulation experiment [16] as a pattern of frame size-system throughput, the neural network performs the training and adjusts the weight by employing the collected data set. Then, the AP performs the knowledge-building task. The gradient information obtained from the neural network is adopted by the Tuning pass to adjust the frame size. Finally, the optimal frame size and corresponding throughput are recorded to analyze the results.

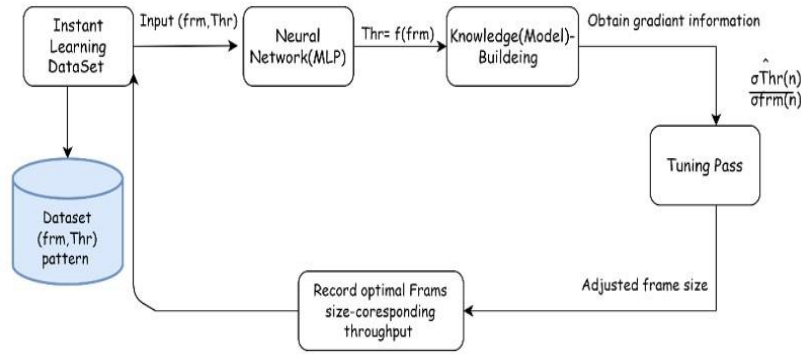


Figure 2. Basic components and flow of the proposed ML model.

Table 1. Simulation Parameter and Notation Summary

Parameters	Symbol	Value
# Of Antenna at AP	NAnt	4
# Of Stations	NumSTA	2–4
VoIP traffic payload size		100Byte
Video traffic payload size		1000Byte
Learning Rate	η	0.5
Mean Square Error Threshold	MES	0.00001
Epoch Threshold		1000 times
Activation Function	Sigmoid (σ)	
Number of training patterns	n	
Indices of neurons in different layers	i, j	
Frame size(input) of nth training patten	frm(n)	
Target response for neuron j	Thr(n)	
Actual response of the nth training patten	$\overline{Thr}(n)$	
Synaptic weight in layer l connecting the output neuron of i to the input neuron j at iteration n	$w_{ji}^{(l)}$	
Weight sum of all synaptic inputs plus bias of neuron j in layer l at iteration n.	$v_j^l(n)$	
Signal of output of neuron j in layer l at iteration n	$y_j^l(n)$	
Local gradient of neuron j in layer l in the tuning pass of hidden layer	$\lambda_j^l(n)$	
Local gradient of neuron j in layer l in the tuning pass of the output layer	$\lambda_1^2(n)$	
Adjustment rate	μ	

4. RESULTS AND DISCUSSION

In this section, we evaluate the performance of the proposed machine learning-based adaptive approach to optimize the system frame size in the WLAN downlink MU-MIMO channel that aims to maximize the system throughput performance by considering the effects of channel conditions, heterogeneous traffic patterns, and number of stations.

4.1. Experimental Procedure

The training data set is collected by adopting the simulation environment proposed by [16] as a pattern of “frame size - system throughput”. The training data set is collected once every 50 seconds. Thus, 50 samples will be collected for each training in considering different network scenarios such as channel conditions, traffic patterns, and number of stations to train the neural network. The system throughput in the data set is the maximum system throughput values obtained from the maximum system throughput achieved by the adaptive aggregation algorithm in [16]. Similarly, the frame size which is used as the input data set in this experiment represents the average offered traffic load generated in the network to obtain the corresponding output i.e., the target system throughput. The weight is updated using these data following the procedure in the backward pass. Forward and backward passes are iteratively performed until the stopping criteria of Mean Square Error (MES) fall below 0.00001 or when the training epoch exceeds 1000 times. The error threshold and the maximum number of iterations determine the accuracy of the function and the computing cost. Then, the tuning pass is executed to adjust the frame size frm by adopting the gradient information from the neural network.

The performance of the proposed approach is evaluated by comparing with the system throughput performance achieved by FIFO (Baseline Approach) which was used as a baseline approach to evaluate the performance of the adaptive aggregation approach in [16]. FIFO (Baseline Approach) is an aggregation algorithm which does not consider adaptive aggregation approach [15,16]. Likewise, in this work, we compare the performance of the proposed machine learning approach denoted as Proposed ML Approach in this experiment with the baseline FIFO (Baseline Approach) obtained from [16]. Moreover, we considered the Maximum Throughput achieved by the adaptive aggregation algorithm in [16] to compare it with the Proposed ML Approach to examine how much the Proposed ML Approach effectively optimized the frame size to the maximum system throughput comparably.

In general, the proposed machine-learning based adaptive approach will be evaluated under the following performance factors. The performance of the proposed ML approach is evaluated under the effects of different traffic models such as Pareto, Weibull, and fBM in Section (4.2). Then the performance of the proposed approach under the effect of channel conditions considering SNR= 3, 10, and 20 dB is evaluated in Section (4.3). The performance of the proposed approach under a varying number of STAs (2,3,4) is evaluated in Section (4.4). Finally, the performance of the proposed ML approach is evaluated in terms of system throughput versus optimal system frame size in Section (4.5). All experiments are conducted with a traffic mix of 50% VoIP and 50% video with a constant frame size of 100 Byte and 1000 Byte, respectively.

4.2. Performance Under the Effect of Various Traffic Models

In this experiment, the proposed approach is evaluated under the effects of different traffic models such as Pareto, Weibull, and fBM [16], SNR = 10 dB, and Num_{STA}= 4. This experiment demonstrates how heterogeneous traffic patterns affect the optimal throughput performance in the WLAN downlink channel. Table 2 illustrates quantitative comparative results of the average

maximum system throughput obtained by optimizing the frame size for the proposed ML approach, the baseline FIFO approaches, and Maximum Throughput under the conditions of different traffic models.

Table 2. Quantitative results achieved by the Proposed ML Approach, Maximum Throughput, and FIFO (Baseline Approaches) for average system throughput performance in Mbps under the effects of different traffic models.

Comparative Approaches	Traffic Models		
	Pareto	Weibull	fBM
FIFO (Baseline Approach)	511.3145	760.629	497.7865
Maximum Throughput	708.9975	820.52775	728.33775
Proposed ML Approach	708.724	820.4445	728.74575

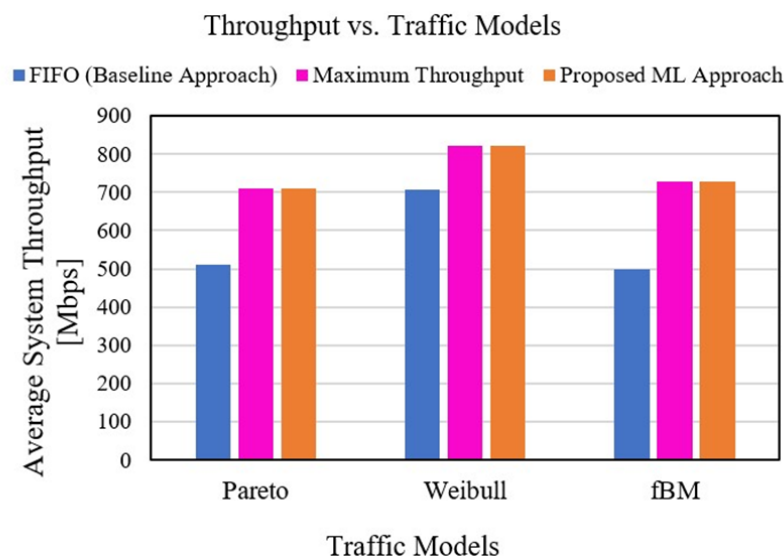


Figure 3. Performance of average system throughput under the effects of heterogenous traffic models when SNR = 10dB.

As the result shows in Figure 3, the proposed ML approach achieved the maximum performance for all traffic models. For instance, for the Weibull traffic model, the maximum performance of 820Mbps is achieved compared to the Pareto and fBM models. Whereas the lowest performance of 708Mbps is achieved for the Pareto traffic model. This indicates that the performance of the proposed ML Approach better copes with the Weibull traffic which is less bursty compared to the other traffic models according to [15]. Thus, these results indicate that traffic patterns in the network determine the system performance. Moreover, the result also demonstrated that the proposed ML approach achieved a compatible result with the maximum system throughput achieved by the adaptive aggregation algorithm i.e., Maximum Throughput proposed by [16]. The FIFO (Baseline Approach) is the worst performance of all traffic models compared to the proposed approach due to its non-adaptive aggregation policy employed in it [15].

4.3. Performance Under the Effects of Channel Conditions

In this section the performance of the proposed approach under different channel conditions when SNR = 3, 10, and 20dB, and NumSTA =4 is evaluated as shown in Figure 4 (a), (b), and (c) for the case of different traffic models such as Pareto, Weibull, and fBM. According to the results,

the system throughput performance increases when the channel quality improved from 3dB to 20dB better than that of the FIFO (Baseline Approach) due to the adaptive aggregation approach adopted in the Proposed ML Approach. In this regard, the Proposed ML approach achieved the lowest performance of 125Mbps in fBM traffic model as shown in Figure 4 (c), and the maximum of 143Mbps is achieved in the Weibull traffic, when the traffic condition is worst i.e., SNR=3dB. In the contrary, under the near-ideal channel condition, e.g., in SNR of 20dB in the figure, the system throughput performance is almost optimal in all approaches due to lower frame error rate occurred under the near-ideal channel condition. However, the proposed approach achieved the maximum performance of 892Mbps using the Weibull traffic model and the lower 732Mbps is achieved in the Pareto traffic model.

In general, from these results, we can conclude that the performance of the proposed approach is affected by the conditions of traffic patterns and channel conditions. The FIFO (Baseline approach) aggregation policy is the worst compared to the proposed approach in all scenarios because of the non-adaptive aggregation strategy it employs. Moreover, the results also demonstrated that the Proposed ML Approach always archived the maximum performance close to the maximum system throughput achieved by the adaptive aggregation algorithm proposed by [16]. Table 3 illustrates quantitative performance results of the average system throughput performances achieved by the Proposed ML Approach, FIFO (Baseline Approach), and Maximum Throughput under the effects of different channel conditions and traffic models.

Table 3. Quantitative results achieved by the Proposed ML Approach, Maximum Throughput, and FIFO (Baseline Approach) for average system throughput performance in Mbps under the effects of different traffic models and channel conditions.

Comparative Approaches	Traffic Models	SNR (dB)		
		3(dB)	10(dB)	20(dB)
FIFO (Baseline Approach)	Pareto	78.569725	511.3145	587.10275
Maximum Throughput		139.19	708.9975	732.4925
Proposed Approach		139.21475	708.724	732.435
FIFO (Baseline Approach)	Weibull	99.8366	706.629	806.74175
Maximum Throughput		143.7976	820.52775	892.26925
Proposed Approach		143.5908	820.4445	892.46475
FIFO (Baseline Approach)	fBM	86.87635	497.7865	566.02125
Maximum Throughput		124.09275	728.33775	785.82175
Proposed Approach		125.05782	728.74575	786.1

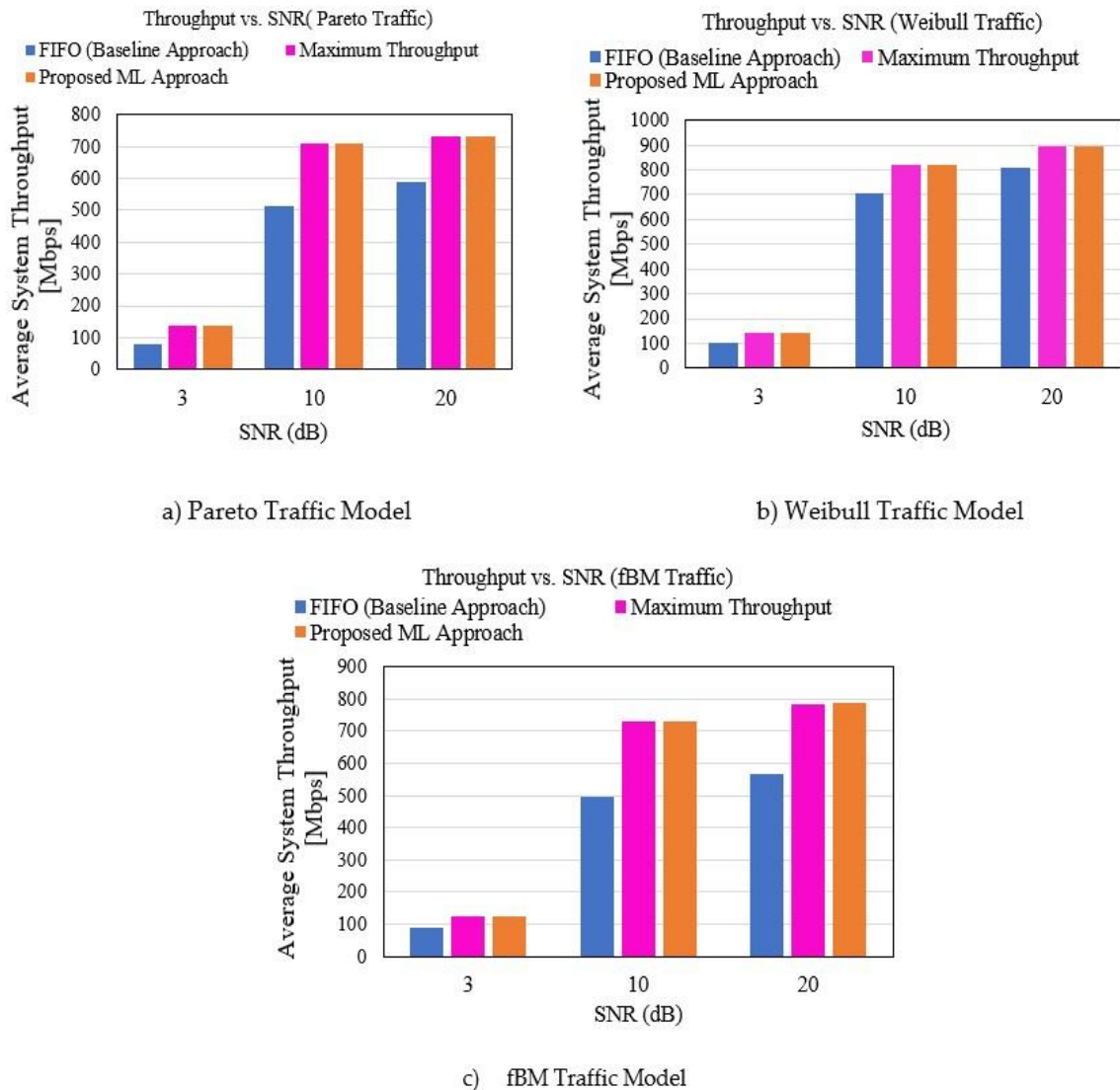


Figure 4. Illustrates System throughput versus SNR for different traffic models such as Pareto, Weibull and fBM when $\text{Num}_{\text{STAs}}=4$.

4.4. Performance Under the Effects of Number of Stations

The performance of the proposed approach is evaluated under the effect of different number of stations ($\text{Num}_{\text{STA}}=2, 3$ and 4), and when the channel condition is $\text{SNR}=10\text{dB}$ for the case of Weibull, Pareto and fBM traffic models. As the results show in Figure 5 (a), (b), and (c), when the number of stations ranges from 2 to 4, the system throughput performance significantly increases in all traffic models as the traffic rate increase with increasing number of stations. However, due to the effect of heterogeneous traffic patterns in different traffic models the performance of the Proposed ML Approach achieved varies even under the same number of stations. Table 4 illustrates quantitative comparative results of the average optimal system throughput achieved by the Proposed ML Approach, Maximum Throughput, and FIFO (Baseline Approaches) under the effects of variable number of STAs.

Table 4. Quantitative results achieved by the Proposed ML approach, Maximum Throughput, and FIFO (Baseline Approach) for average system throughput in Mbps under the effects of variable number of stations in Weibull, Pareto, and fBM traffic models.

Comparative Approaches	Traffic Models	Number of STAs		
		2	3	4
FIFO (Baseline Approach)	Weibull	431.467	507.87425	706.629
Maximum Throughput		438.04875	620.19375	820.52775
Proposed Approach		437.902	620.18075	820.4445
FIFO (Baseline Approach)	Pareto	338.65	357.08	511.3145
Maximum Throughput		425.56325	554.08825	708.9975
Proposed Approach		425.377	553.0695	708.724
FIFO (Baseline Approach)	fBM	311.0525	417.13275	497.7865
Maximum Throughput		396.22	566.40875	728.33775
Proposed Approach		396.0355	566.284	728.74575

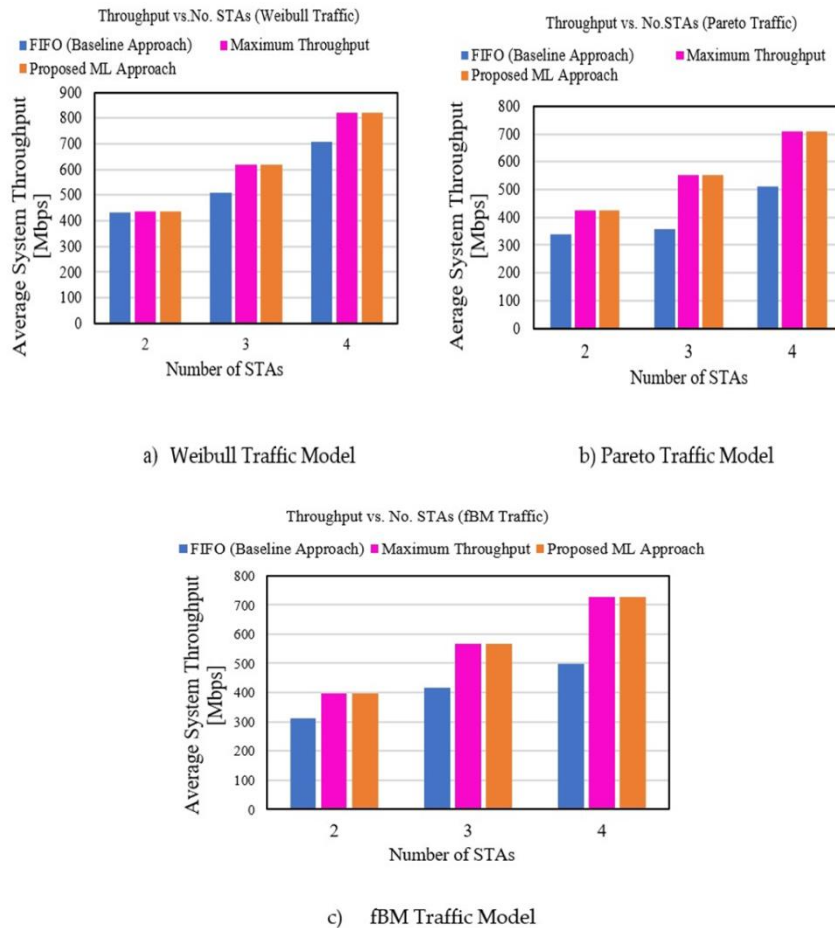


Figure 5. Performance of system throughput versus number of stations when the channel condition is SNR =10dB for the Weibull, Pareto, and fBM traffic models.

As shown in the results in Figure 5, the proposed approach always outperforms the FIFO (Baseline Approaches) in all scenarios due to the adaptive aggregation strategy it adopts. In this regard, the proposed approach achieved the maximum performance of 820Mbps in the case of Weibull traffic whereas the lower performance of 708Mbps is achieved in the Pareto traffic with the same number of STAs. Likewise, when the number of stations equals 2, the worst performance of 396Mbps is achieved by the fBM traffic. These results show that number of stations affects the performance of the system throughput behavior under the conditions of heterogeneous traffic patterns among streams in the downlink MU-MIMO channel. However, the proposed approach always achieved the maximum system throughput performance better than the FIFO (Baseline Approach) closest to the Maximum Throughput of the adaptive aggregation algorithm [16].

4.5. Performance of System Throughput Vs. Optimal Frame Size

The results in Figure 6 (a), (b), and (c) show the performance of system throughput behavior with increasing optimal frame size considering SNR= 10 dB, Num_{STA} = 4, under the effects of different traffic models Weibull, Pareto, and fBM. This experiment examines the optimal frame size and the corresponding system throughput achieved by the Proposed ML Approach under the effect of different traffic models compared with the FIFO (Baseline Approach).

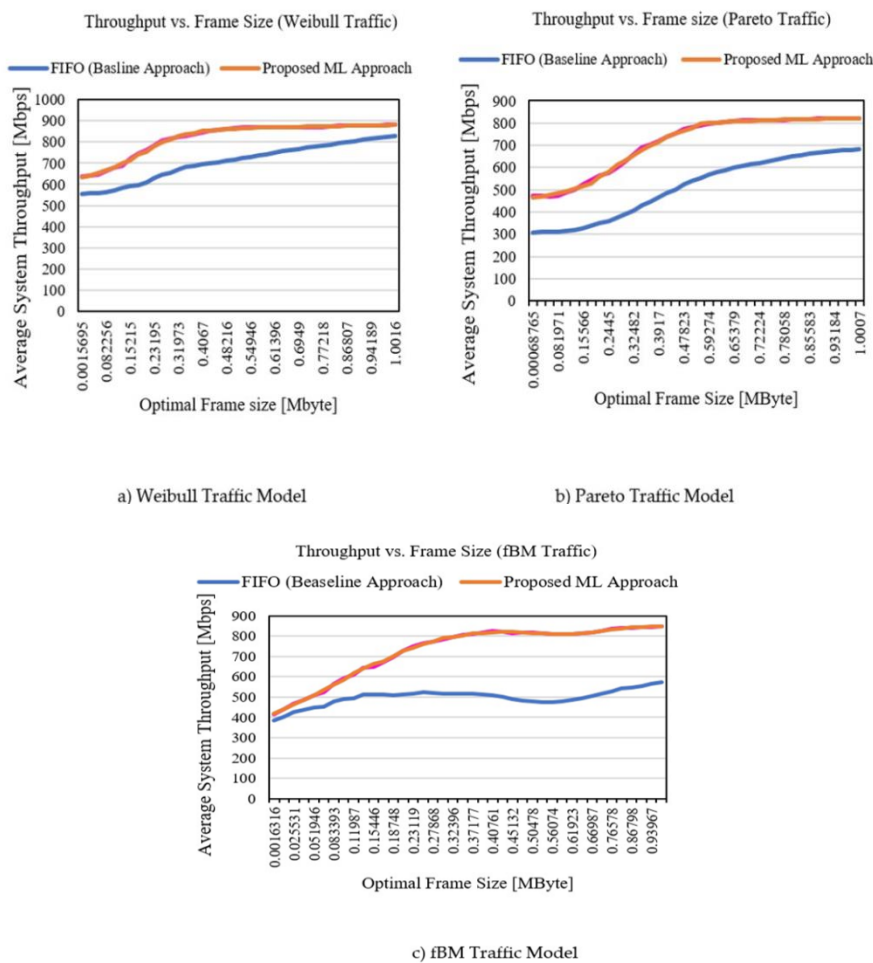


Figure 6. Performance of system throughput versus optimal System frame size when Num_{STAs} = 4 and SNR =10dB for the Weibull, Pareto, and fBM traffic models.

According to the results shown in Figure 6 (a), (b), and (c), the proposed ML approach achieved the maximum performance in all traffic models because of the adaptive aggregation approach employed in it in considering channel conditions, traffic patterns, and number of stations. For instance, in the Weibull traffic model, the performance increases with increasing frame size thus achieved the maximum system throughput 880Mbps at the optimal frame size of 1Mbyte. In the case of the Pareto traffic model, the proposed ML approach achieved a maximum system throughput performance of 820Mbps at the optimal system frame size of 1Mbyte. Moreover, as the result shows, the system throughput performance in the Weibull traffic model achieved 630Mbps better than that of the Pareto 470Mbps at the beginning of the result and when the frame size increases. In fBM traffic model the proposed approach achieved the maximum performance of 846Mbps at the optimal system frame size of 0.93Mbyte. However, FIFO Baseline approach achieves the lowest performance in all scenarios because it does not allow adaptive aggregation approach. These results demonstrated that the optimal system frame size achieved is affected by the traffic condition in the network. In this regard, the proposed adaptive ML approach achieved a significant performance by efficiently optimizing the frame size that would maximize the system throughput of WLAN in the downlink MU-MIMO channel taking into account the traffic conditions better than that of the FIFO (Baseline Approach) non-adaptive aggregation approach.

5. CONCLUSIONS

IEEE 802.11n/ac introduced frame aggregation technology to accommodate the growing traffic demand and increases the performance of transmission efficiency and channel utilization by allowing many packets to be aggregated per transmission which significantly enhances the throughput of WLAN. The performance of WLAN depends on different performance factors such as frequency channel, modulation, and coding schemes, transmitter power, etc. at the PHY layer, and retry limit, frame size, contention window size, maximum number of backoffs, etc. at the MAC layer have a significant impact on the performance of WLAN. Optimizing these parameters would improve the system performance of WLAN. Frame size optimization is the main concern of this study. However, it is difficult to efficiently utilize the benefits of frame aggregation in downlink MU-MIMO channel as stations have heterogeneous traffic demand and data transmission rates. As a consequences, wasted space channel time will be occurred that degrades transmission efficiency. Moreover, the release of the new IEEE 802.11 standards such as IEEE 802.11ax and IEEE 802.11ay, 5G technologies, and the massive amount of traffic generated in the communication network, allow to expand the set of available communication technologies to compete for the limited radio spectrum resources in pushing for the need of enhancing their coexistence and more effective use of the scarce spectrum resources and to speed up decision-making process. In response to these performance demands, Machine Learning (ML) is the recent innovating solution to maintain a self-driven network that can configure and optimize itself by reducing human interventions and it is capable of overcoming the drawbacks of traditional mathematical formulations and complex data analysis algorithms. However, most of the existing approaches did not consider a machine-learning-based optimization solution. The main contribution of this paper is to propose a machine-learning-based frame size optimization solution to maximize the system throughput in WLAN downlink MU-MIMO channel by considering the effect of channel conditions, heterogeneous traffic patterns, and number of stations. In this approach, the AP performs the system throughput measurement and collects the “frame size – throughput” patterns as a data set. To cope with the effects of time-varying channel conditions and heterogeneous traffic patterns, we use online training and iteratively operate the three passes (forward pass, backward pass, and tuning pass) to model the instantaneous (frm, Thr) relationship and optimize the frame size. The neural network is used to train these training datasets to accurately model the system throughput with respect to the frame size. Frame size is adjusted according to gradient information which is abstracted from the neural network after the

knowledge building. We have performed a simulation experiment to validate that the proposed approach can effectively optimize the system frame size under various channel conditions, traffic patterns, and number of STAs to maximize the system throughput performance of WLAN as compared to the baseline FIFO baseline aggregation algorithm. Moreover, the proposed ML approach can achieve the maximum performance close to the Maximum Throughput of our earlier adaptive aggregation algorithm.

Future work will be conducted by considering the real traffic scenarios. Moreover, the cost of delay and the effects in different channel models such as Rayleigh and Rician on both uplink and downlink WLAN channels will be studied.

ACKNOWLEDGEMENTS

We would like to thank all authors for their contributions and for the success of this manuscript. Moreover, we would like to thank the entire authors team for their supportive participation, editors, and anonymous reviewers of this manuscript.

REFERENCES

- [1] IEEE Computer Society. Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 4: Enhancements for Very High Throughput for Operation in Bands below 6 GHz IEEE Computer Society. 2013.
- [2] Liao, Ruizhi, Boris Bellalta, Miquel Oliver, and Zhisheng Niu, (2014) "MU-MIMO MAC protocols for wireless local area networks: A survey." *IEEE Communications Surveys & Tutorials*, Vol. 18, No. 1: 162-183.
- [3] Yin, Jun, Xiaodong Wang, and Dharma P. Agrawal, (2004) "Optimal packet size in error-prone channel for IEEE 802.11 distributed coordination function." In 2004 IEEE Wireless Communications and Networking Conference (IEEE Cat. No. 04TH8733), Vol. 3, pp. 1654-1659. <https://doi.org/10.1109/wcnc.2004.1311801>.
- [4] Coronado, Estefanía, Abin Thomas, and Roberto Riggio, (2020) "Adaptive ml-based frame length optimization in enterprise SD-WLANs." *Journal of Network and Systems Management*, Vol. 28, No. 4, pp 850-881.
- [5] Kulin, Merima, Tarik Kazaz, Eli De Poorter, and Ingrid Moerman, (2021) "A survey on machine learning-based performance improvement of wireless networks: PHY, MAC and network layer." *Electronics* Vol. 10, No. 3, pp 318.
- [6] Sun, Yaohua, Mugen Peng, Yangcheng Zhou, Yuzhe Huang, and Shiwen Mao, (2019) "Application of machine learning in wireless networks: Key techniques and open issues." *IEEE Communications Surveys & Tutorials*, Vol. 21, No. 4, pp 3072-3108.
- [7] Shea, T. "O and Hoydis J., (2017) " An introduction to deep learning for the physical layer," *IEEE Transactions on Cognitive Communications and Networking*, Vol 3, No. 4, pp 563-575.
- [8] Joo, Er Meng, and Yi Zhou, eds. (2009). *Theory and novel applications of machine learning*. BoD—Books on Demand.
- [9] Luong, Nguyen Cong, Dinh Thai Hoang, Ping Wang, Dusit Niyato, Dong In Kim, and Zhu Han, (2016) "Data collection and wireless communication in Internet of Things (IoT) using economic analysis and pricing models: A survey." *IEEE Communications Surveys & Tutorials*, Vol. 18, No. 4, pp 2546-2590.
- [10] Wang, Cheng-Xiang, Marco Di Renzo, Slawomir Stanczak, Sen Wang, and Erik G. Larsson, (2020) "Artificial intelligence enabled wireless networking for 5G and beyond: Recent advances and future challenges." *IEEE Wireless Communications*, Vol. 27, No. 1, pp 16-23.
- [11] Ci, Song, and Hamid Sharif, (2002) "Adaptive optimal frame length predictor for IEEE 802.11 wireless LAN." In *Proceedings of the 6th International Symposium on Digital Signal Processing for Communication Systems (IEE DSPCS'2002)*.
- [12] Bianchi, Giuseppe, (2000) "Performance analysis of the IEEE 802.11 distributed coordination function." *IEEE Journal on selected areas in communications*, Vol. 18, No. 3, pp 535-547.

- [13] Lin, Pochiang, and Tsungnan Lin, (2009) "Machine-learning-based adaptive approach for frame-size optimization in wireless LAN environments." *IEEE transactions on vehicular technology*, Vol. 58, No. 9 pp 5060-5073.
- [14] Modiano, Eytan, (1999) "An adaptive algorithm for optimizing the packet size used in wireless ARQ protocols." *Wireless Networks*, Vol. 5, No. 4, pp 279-286.
- [15] Kassa, Lemlem, Mark Davis, Jingye Cai, and Jianhua Deng, (2021) "A New Adaptive Frame Aggregation Method for Downlink WLAN MU-MIMO Channels." *J. Commun.* Vol. 16, No. 8, pp 311-322.
- [16] Kassa, Lemlem, Mark Davis, Jianhua Deng, and Jingye Cai, (2022) "Performance of an Adaptive Aggregation Mechanism in a Noisy WLAN Downlink MU-MIMO Channel." *Electronics*, Vol. 11, No. 5, pp 754.
- [17] Kim, Sanghyun, and Ji-Hoon Yun, (2019) "Efficient frame construction for multi-user transmission in IEEE 802.11 WLANs." *IEEE Transactions on Vehicular Technology*, Vol. 68, No. 6, pp 5859-5870.
- [18] Bellalta, Boris, Jaume Barcelo, Dirk Staehle, Alexey Vinel, and Miquel Oliver, (2012) "On the performance of packet aggregation in IEEE 802.11 ac MU-MIMO WLANs." *IEEE Communications Letters*, Vol. 16, No. 10, pp 1588-1591.
- [19] Moriyama, Tomokazu, Ryo Yamamoto, Satoshi Ohzahata, and Toshihiko Kato, (2017) "Frame aggregation size determination for IEEE 802.11 ac WLAN considering channel utilization and transfer delay." *ICETE 2017 - Proc 14th Int Jt Conf E-Bus Telecommun.* Vol. 6, pp 89-94.
- [20] Lin, Chi-Han, Yi-Ting Chen, Kate Ching-Ju Lin, and Wen-Tsuen Chen, (2018) "FdoF: Enhancing channel utilization for 802.11 ac." *IEEE/ACM Transactions on Networking*, Vol. 26, No. 1, pp 465-477.
- [21] Lin, Chi-Han, Yi-Ting Chen, Kate Ching-Ju Lin, and Wen-Tsuen Chen, (2017) "acPad: Enhancing channel utilization for 802.11 ac using packet padding." In *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*, pp. 1-9.
- [22] Nomura, Yoshihide, Kazuo Mori, and Hideo Kobayashi, (2016) "High-Efficient Frame Aggregation with Frame Size Adaptation for Downlink MU-MIMO Wireless LANs." *IEICE Transactions on Communications*, Vol. 99, No. 7, pp 1584-1592.
- [23] Haykin, Simon, (1999). *Neural Networks: A Comprehensive Foundation*. Englewood Cliffs, NJ: Prentice-Hall.
- [24] Snyman, Jan A., and Daniel N. Wilke, (2005) *Practical mathematical optimization*. Springer Science+Business Media, Incorporated.
- [25] Hoi, Steven CH, Doyen Sahoo, Jing Lu, and Peilin Zhao, (2021) "Online learning: A comprehensive survey." *Neurocomputing* 459 (2021): 249-289.
- [26] Online-vs-offline-machine-learning. Available online: <https://www.qwak.com/post/> (accessed on 12 June 2022).
- [27] Behera, Laxmidhar, Swagat Kumar, and Awahan Patnaik, (2006) "On adaptive learning rate that guarantees convergence in feedforward networks." *IEEE transactions on neural networks*. Vol. 17, No. 5, pp 1116-1125.

AUTHORS

Lemlem Kassa has been a senior lecturer in Addis Ababa Science and Technology university in Addis Ababa, Ethiopia since February 2013. She is currently pursuing her PhD study in School of Information and Software Engineering in University of Electronic Science and Technology of China (UESTC) in China. She received the B.S. degree from Micro link Information Technology college in Addis Ababa Ethiopia in 2006 in software engineering and M.S. degree from University Putra Malaysia (UPM), Malaysia in 2009. Her research interests are in the area of wireless communications, artificial intelligence, mobile computing, and software designing.



Dr. Jianhua Deng graduated in information security from the University of Electronic Science and Technology of China (UESTC), China, in 2006. After graduated, he joined the School of Computer Science and Engineering at UESTC as a staff. From 2009 to 2014, he was a Ph.D. student in Dublin Institute of Technology (DIT) in Ireland and received Ph.D. degree in electrical engineering from DIT in 2014. Now, he is a vice professor in the School of Information and Software Engineering at UESTC. His research interests are in the area of wireless communication, statistical machine learning, artificial intelligence, deep learning. He is the reviewer of some SCI journals (e.g. wireless personal communications).



Prof. Mark Davis received his BE, MEngSc, and PhD degrees from University College Dublin in 1986, 1989 and 1992 respectively. He is currently the director of the Communications Network Research Institute at Technological University Dublin (TU Dublin). His research interests are in the area of radio resource management techniques for wireless networks, specifically IEEE 802.11 WLANs.



Prof. Jingye Cai is Professor and Associate Dean at University of Electronic Science and Technology of China (UESTC). He received BS from Sichuan University in 1983 with a major in radio electronics and Phd from the University of Electronic Science and Technology of China in 1990, with major in signal and information processing. His research interests are in the area of intelligent computing, information engineering, digital Information Processing, and signal processing.



A SIMPLE NEURAL NETWORK FOR DETECTION OF VARIOUS IMAGE STEGANOGRAPHY METHODS

Mikołaj Płachta and Artur Janicki

Warsaw University of Technology, Warsaw, Poland

ABSTRACT

This paper addresses the problem of detecting image steganography based in JPEG files. We analyze the detection of the most popular steganographic algorithms: J-Uniward, UERD and nsF5, using DCTR, GFR and PHARM features. Our goal was to find a single neural network model that can best perform detection of different algorithms at different data hiding densities. We proposed a three-layer neural network in Dense-Batch Normalization architecture using ADAM optimizer. The research was conducted on the publicly available BOSS dataset. The best configuration achieved an average detection accuracy of 72 percent.

KEYWORDS

Steganography, deep machine learning, detection malware, BOSS database, image processing.

1. INTRODUCTION

As the Internet evolves, so do the threats lurking in it. Therefore, cyber security is playing an increasingly important role in our world. Threats are becoming more sophisticated and less obvious, making them more difficult to identify and detect. One such threat is transmissions that do not transmit data overtly. This type of method is called steganography, which aims to hide classified information in unclassified material. In other words, it is possible to hide a message in data that is publicly transmitted without revealing the fact that a secret communication exists. These are very dangerous methods for this reason, as it is hard to protect against them, and they can be used to spread malware or can be exploited by such software, known as stegomalware [1].

Most steganographic methods use multimedia data as a carrier of information, such as images. They are called digital media steganography and image steganography, respectively. Examples of such techniques include the Vawtrak/Neverquest method [2], whose idea was to hide URLs in favicon images, or the Invoke-PSImage tool [3], where developers hid PowerShell scripts in image pixels using the commonly used least significant bit (LSB) approach. Another variation can be hiding information in the structure of GIF files [4], which is quite innovative due to the binary complexity of the GIF structure.

As there are already a lot of ways to hide information in this way and it is a big threat to the ordinary user, there is a great need to develop effective, reliable and fast methods to detect hidden content. For this reason, a number of projects have been set up to improve the ability to warn and prevent attacks of this type. One project aimed at stegomalware detection was Secure Intelligent Methods for Advanced RecoGnition of malware and stegomalware (SIMARGL), which was carried out under the EU's Horizon 2020 program.

The experiments presented in this paper are a continuation of this initiative. The goal of this research was to find the most effective automatic methods for detecting digital steganography in JPEG images. JPEG compression is commonly used to store and transmit images, so it can be easily exploited for malicious purposes. For this purpose, different variants of neural networks and shallow learning methods have been studied. Detailed studies and obtained results for such methods are described in [5]. This paper mainly focuses on continuing the search for the best predictive model that would work best for the detection of various steganographic algorithms. The research continues exclusively in the area of deep machine learning. This type of detection method can be integrated with antimalware software or any other system that performs file scanning for security purposes (such as a messaging system).

The first part of the paper will recall the theory of steganography and the algorithms and detection methods used in the research, while the second part will present further research along with a comparison to the original research path.

2. RELATED WORK

Our paper focuses on JPEG images as data storage media for image steganography. The popularity of this file format has resulted in many methods of hiding data, as well as various detection methods. This section will briefly review the basics of JPEG-based image steganography, including the most commonly used algorithms.

2.1. Steganographic Methods in JPEG Images

While many steganographic algorithms operate in the spatial domain, some introduce changes at the level of Discrete Cosine Transform (DCT) coefficients stored in JPEG files. Moreover, some algorithms aim to minimize the probability of detection by exploiting content adaptivity: they embed data mainly in less predictable regions, where changes are harder to identify. Such modifications are the most difficult to detect, which is why they were chosen as the leading ones at the beginning of the ongoing research. After analyzing image collections, e.g. [6], we selected three algorithms: nsF5 [7], JPEG Universal Wavelet Relative Distortion (J-Uniward) [8] and Uniform Embedding Revisited Distortion (UERD) [9]. They are briefly characterized in the following subsections.

2.2. nsF5

The nsF5 algorithm embeds data by modifying the least significant bits of the AC (having at least one non-zero value) of the DCT coefficients of unmodified JPEG objects. The data is hidden using syndrome coding. Having an m p -bit message to embed using n values of AC DCT our task is to obtain a vector y . This vector must satisfy the equation:

$$Dy = m \quad (1)$$

where D is a binary matrix p by n , which is shared by the sending and receiving parties. The embedder must find a solution to the above equation that does not require modification of the zero-value coefficient bits. The solution must minimize the Hamming weight between the modified and unmodified vectors of the least significant bit. The above version is a simple syndrome coding idea, but a more sophisticated coding scheme, Syndrome Trellis Coding (STC) [19], using a parity check matrix instead of D , is usually used. The y -vector represents the path through the trellis built based on the parity check matrix.

2.3. J-Uniward

J-Uniward is a method for modeling steganographic distortion caused by data embedding. The goal is to provide a function that determines which areas of an unmodified object are less predictable and more difficult to recognize. Changes introduced during steganographic data embedding in these areas are more difficult to detect than if they were introduced uniformly across the media. By calculating the relative changes in value based on the directional decomposition of the filter bank, the method is able to detect smooth edges that are easy to recognize. By analyzing in this way which areas may be more susceptible to detection, this method gives a very high efficiency in little-noticed data hiding. As with nsF5, the STC coding scheme is used to create a data hiding algorithm that adapts to the content.

2.4. UERD

UERD is another steganographic embedding model that aims to minimize the probability of detecting the presence of hidden information by reducing the impact of embedding on the statistical parameters of cover information. It achieves this by analyzing the parameters of the DCT coefficients of individual mods, as well as entire DCT blocks and their neighbors. It can then determine whether a region can be considered "noisy" and whether embedding will affect statistical features such as file histograms. "Wet" regions are those where statistical parameters are predictable and where embedding would cause a risk of information detection. The use of values during embedding such as DC mode coefficients or zero DCT coefficients are not excluded. This is because their statistical profiles can make them appropriate from a security perspective. The UERD algorithm evenly distributes the relative changes in statistics resulting from embedding. UERD, like nsF5 and J-Uniward, uses STC to hide message bits in desired values.

3. STEGANOGRAPHY DETECTION

Image steganography is an important topic in cyber security, and so far one can read in the literature about a very large number of attempts to detect it. These methods usually extract certain parameters from analyzed images, and then classification algorithms are applied. They are usually based on machine learning approaches, so shallow or deep methods can be used. The research described in this paper focuses only on the deep ones, so this section first describes the features most commonly used in steganalytic algorithms, and then briefly describes typical examples of detection algorithms based on deep learning.

3.1. Feature Space Extraction

While many feature space analysis methods for image steganalysis have been described in the literature, three of the most effective were selected. The first one analyzed was Discrete Cosine Transform Residuals (DCTR) [10], which analyzes the data resulting from obtaining DCT values for a given image. In the first step, a random 8x8 pixel filter is created, which will be applied later to filter the entire image. Then, iterating step by step over the analyzed image, a histogram is created using the spline function with the previously mentioned filter. The article [11] proposes an example of using DCTR parameters in combination with a multilevel filter. Another variation of this approach is a method based on Gabor filters, or Gabor Filter Residuals (GFR) [12]. It works in a very similar way to DCTR, but instead of a random 8x8 filter, Gabor filters are used. The article [13] describes a successful application of the GFR function in JPEG steganography detection. A third approach to parameterizing the feature space is to use the PHase Aware

pRojection Model (PHARM) [14]. Applying various linear and nonlinear filters, a histogram is created from the projection of values for each residual portion of the image.

3.2. Steganography Detection Methods

Recently, neural networks have been among the most popular machine learning methods used in various task automation applications. Detecting steganographically hidden data in digital images is one of them. Extracted image parameters based on decompressed DCT values, which were pre-filtered and fed into the first wave layer of the network, were usually used as input data.

Proprietary variants of convolutional networks, such as XuNet [15] or ResNet [16], are most commonly used for this purpose. A common feature of these networks is the combination of Convolution-BatchNormalization-Dense (C-CB-D) structures, i.e. a spline function, a normalization layer and a base layer of neurons with an appropriate activation function. Functions such as Sigmoid, TLU (Threshold Linear Unit) and Gaussian are used, but the most common are Rectified Linear Unit (ReLU) or TanH. Steganography detection models based on feature extraction and the C-BN-D scheme are shown in Figure 1.

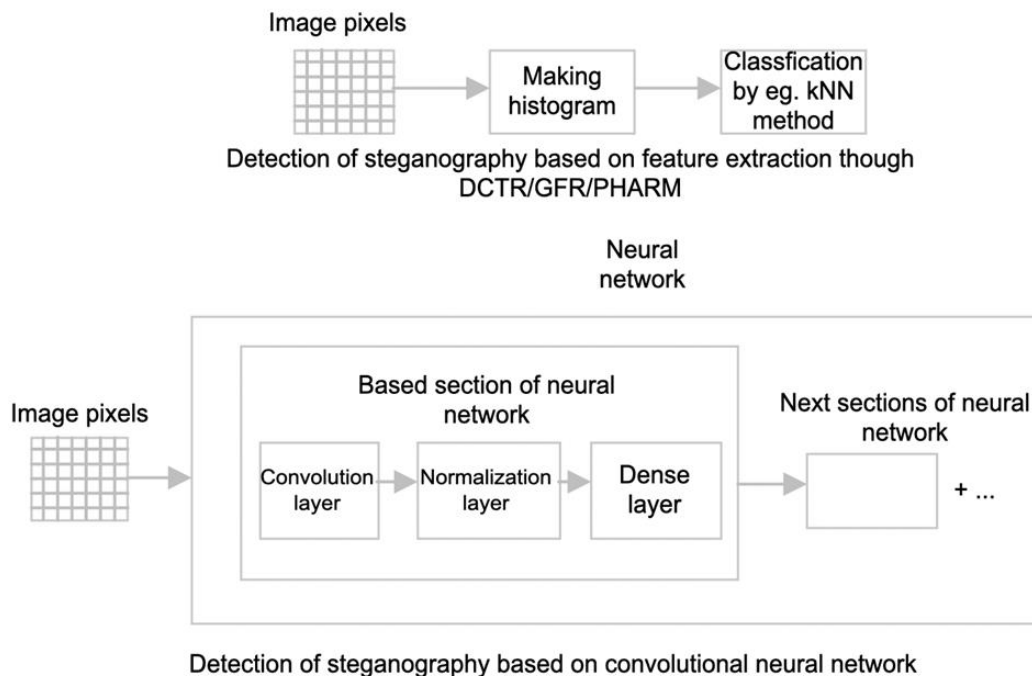


Figure 1. Examples of prediction models

4. RESEARCH METHODOLOGY AND MATERIALS

4.1. Data Set Under Study

The "Break Our Steganographic System" (BOSS) image set [17], which contains 10,000 black-and-white images (without hidden data), was used for the study. The images were converted to JPEG format with a quality factor of 75. Three other sets of images were then generated, hiding there random data with a density (bpnzac, i.e. the number of bits for each non-zero AC co-factor) of 0.4 or 0.1, using the previously mentioned three steganographic algorithms: nsF5, J-Uniward

and UERD. Each dataset was then divided in parallel into training and test subsets, in a 90:10 ratio.

4.2. The Proposed Detection Method

The neural network environment was based on the Keras library and Tensorflow due to the easy definition of the model. The proposed network architecture was mainly based on the Dense-BatchNormalization structure but did not use the spline part as described in the available literature. A schematic of this concept is shown in Figure 2.

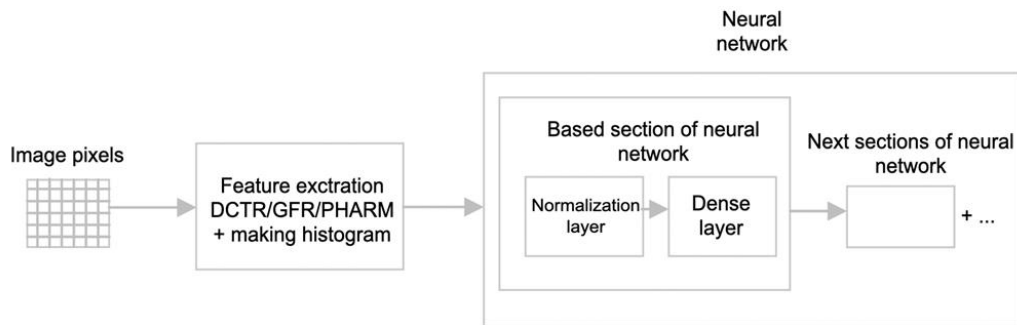


Figure 2. Proposed model for research

We also tested different activation functions for the dense layer, but the best results were obtained for the ReLU function. After extensive research, one optimizer was selected: Adaptive Moment Estimation (ADAM) [18], which gave better results than the others like Stochastic Gradient Descent etc. The last parameter that significantly affected the model's learning efficiency was the learning rate. During the study, it turned out that lowering it gave very promising results without changing the network architecture and optimizer.

4.3. Neural network learning environment

The research consisted of two parts. The first part was similar to the preliminary research described in the article [5]. First, two neural network model architectures were selected:

- The first with three layers with ReLU activation function, with 250 neurons in the first, 120 in the second and 50 in the third, used in four neural network models;
- The second with two layers also with ReLU function, having 500 neurons in the first layer and 250 in the second, used in the last (fifth) neural network model.

No spline layers were used, while additional normalization layers (BatchNormalization) were applied between the simple layers. All models use the ADAM optimizer. The SGD optimizer was also tested in a previous phase of research described here [5], but it gave relatively poor results and can be ignored. The learning rate used values of $1e^{-4}$ or $1e^{-5}$ for ADAM. In this way, three configurations were prepared. The results are shown in Table 1

Table 1. Results of accuracy for all three configurations for matching conditions, i.e., detection model was dedicated to given steganographic algorithm (a dash means that network learning did not successfully converge)

Network Architecture	Learning Rate	Parameters	J-Uniward		nsF5		UERD		Average
			0.1	0.4	0.1	0.4	0.1	0.4	
250 x BN x 120 x BN x 50 (3 layers)	1e ⁻⁴	DCTR	-	83.1	76.3	98.8	66.5	94.5	78.3
		GFR	-	86.5	68.3	95.5	63.4	92.9	76.1
		PHARM	-	74.7	62.3	95.9	51.4	88.5	70.5
	1e ⁻⁵	DCTR	-	83.0	74.2	99.7	64.7	93.1	77.5
		GFR	-	88.4	68.0	98.2	62.6	92.5	76.6
		PHARM	-	76.1	66.1	93.4	55.5	89.4	71.8
500 x BN x 250 (2 layers)	1e ⁻⁵	DCTR	-	80.8	73.5	99.6	61.9	93.5	76.6
		GFR	53.6	86.4	67.6	97.4	64.2	91.9	76.9
		PHARM	-	75.0	54.1	94.2	54.0	87.9	69.2

Next, the best model configuration was selected, i.e., a three-layer model with the ADAM optimizer at a learning rate of 1e⁻⁴, and six separate models were taught for this configuration, one for each version of the set. Next, cross-testing was carried out, that is, each model was tested to see how it performed in detecting all six harvests. These results are shown in Table 2.

4.3. Evaluating the Effectiveness of Models

To assess the effectiveness of the resulting models, popularly used metrics were applied. The first type is accuracy, which defines what percentage of the entire examined data set is correctly classified. The second metric is precision, which defines what proportion of the images indicated by the classifier as belonging to a given class actually do.

Table 2. Accuracy results of cross-testing DCTR model. Best results in columns are shown in bold.

DCTR model version	J-Uniward		nsF5		UERD		Average
	0.1	0.4	0.1	0.4	0.1	0.4	
J-Uniward 0.1	50.2	51.4	50.4	54.3	50.4	52.3	51.5
J-Uniward 0.4	53.6	83.1	64.2	88.7	57.6	84.5	72.0
nsF5 0.1	52.9	70.7	76.3	85.9	56.7	82.7	70.9
nsF5 0.4	50.2	55.5	53.5	98.8	50.6	63.0	61.9
UERD 0.1	53.3	71.2	63.3	76.9	66.5	76.8	68.0
UERD 0.4	50.8	66.1	54.9	95.8	54.1	94.5	69.3

The next metric is recall, which determines what fraction of images of a given class will be detected by the model. The fourth metric analyzed is F1-score, which is the harmonic mean of precision and recall. The last metric we used to test the effectiveness of the model is the area under the ROC curve (AUC). ROC curves will also be presented, as they can show the effectiveness of a given model very well. The larger the area under the ROC curve (i.e., AUC), the more effective the model is. In the first and second phases of the study, the main metric used was accuracy, while for the best model, which was selected after cross-testing, the other metrics will also be calculated. Since the test set is perfectly balanced, the accuracy score is not biased and reflects well the detection ability of a given classifier.

5. OBTAINED RESULTS

Table 1 shows the accuracy results for each configuration tested in the first phase of the study. The two-layer model was noticeably worse than the three-layer versions. The best results were obtained for the ADAM optimizer at a learning rate of $1e-4$ with the three-layer neural network model. For the matching conditions (detection model trained on data generated with the same image steganographic method) the worst accuracy results were obtained for J-Uniward sets, better on UERD, and the best for nsF5 sets. When analyzing feature spaces, PHARM-based models were the least accurate, while GFR and DCTR were very close to each other with a slight advantage for DCTR. Therefore, a model with the DCTR feature space was selected for further testing.

In the second part of the study as for cross-testing, one can notice that obviously the best scores are mostly on the diagonal of Table 2, meaning the matching condition. However, it can be seen that the model trained on the J-Uniward 0.4 set performed best as for classification of images. Additional metrics shown in Table 3 were calculated for this configuration.

Table 3. All metrics for best cross-testing DCTR model

Metrics	J-Uniward		nsF5		UERD		Average
	0.1	0.4	0.1	0.4	0.1	0.4	
Accuracy	53.6	83.1	64.2	88.7	57.6	84.5	72.0
Precision	57.1	80.2	69.9	82.1	63.0	80.8	72.2
Recall	28.8	87.7	50.1	99.1	36.8	90.7	65.5
F1-score	38.3	83.8	58.4	89.8	46.5	85.4	67.0
AUC	57.3	91.6	70.2	99.1	63.2	93.3	79.1

As one can observe the precision and recall parameters, for sets with a density of 0.4 the results are close to the precision, which means that the model is well balanced. There is a larger difference for a density of 0.1 due to the greater hiding of data in the files. In Figure 3, we can see that the values of accuracy, F1-score and AUC are close to each other at each harvest. It can also be visually observed that the weakest results were obtained for J-Uniward, and the best for nsF5. Figure 4 shows the ROC curves, which illustrate the efficiency in detection of a given harvest. They are consistent with previous results for this model.

6. SUMMARY

Comparing the results obtained with previous results from the article [5], it can be seen that using a single model instead of separate six models for universal detection is feasible. The difference in the average accuracy score between the two concepts is on the order of 10 percent relative, which, with the possible complexity of interpreting data from six separate models plus a high chance of false-positive cases, is a very good and promising result.

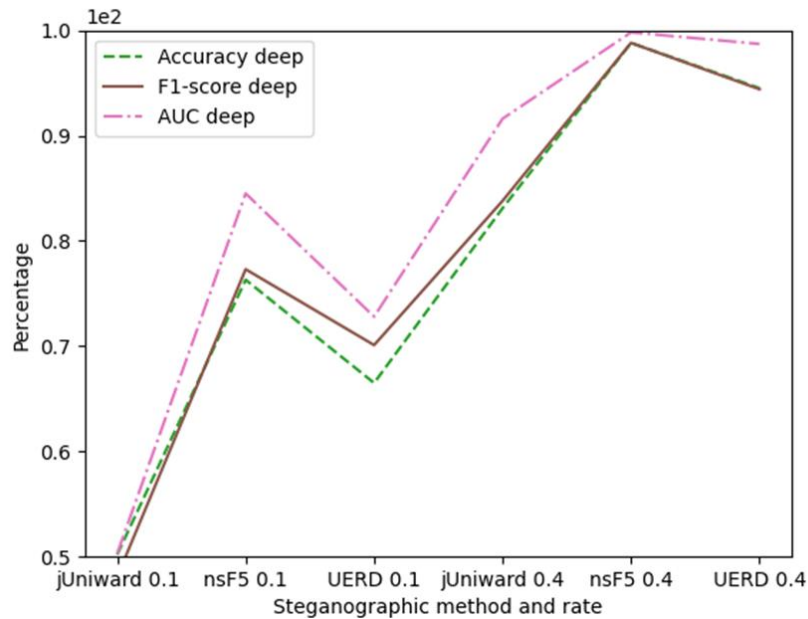


Figure 3. Accuracy for the best DCTR model

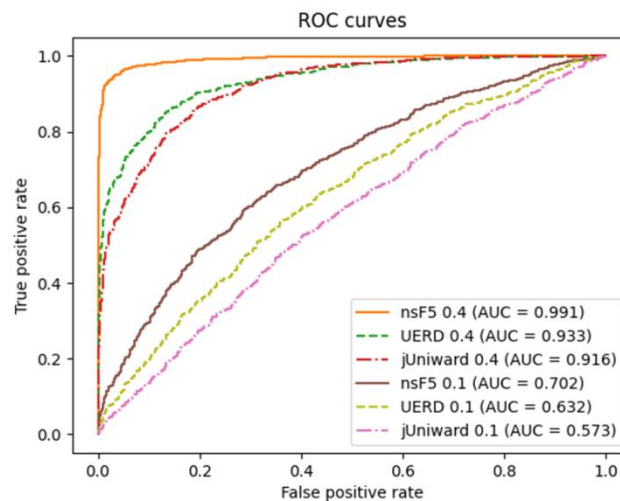


Figure 4. ROC curves for the best DCTR model

During the study, it was also noted that adding a normalization layer was able to improve the results significantly. For example, for the nsF5 sets, the difference was on the order of 15-20 percent relative. This means that the normalization layer from the C-BN-D model is indispensable over the splicing part, which can be dispensed with. Also, it was noted that the difference between two and three layers of dense networks is practically imperceptible, and there is no good reason to use much more complicated networks. Regardless of the phase one or phase two tests conducted, the J-Uniward algorithm was the most difficult to detect, and the easiest was nsF5. Also, for sets of 0.1 there is noticeably worse detection than for sets of 0.4, which means that the less data we hide in an image, the lower the chance of discovering it.

This paper analyzed the effectiveness of image steganography detection based solely on a single neural network model. The effectiveness depended on the algorithm used as well as the data density used. Analyzing the results, we can see that one network model using the DCTR feature

space did quite well in detecting most threats. This gives hope for further potential improvements to this model using some combination of two or three base models. Further research on this issue will be conducted in the next iteration.

ACKNOWLEDGEMENTS

The study has been partially supported by the SIMARGL Project with the support of the European Commission and the Horizon 2020 Program, under Grant Agreement No. 833042 and also by the IDUB program from the Warsaw University of Technology.

REFERENCES

- [1] Caviglione, L.; Choraś, M.; Corona, I.; Janicki, A.; Pawlicki, M.; Mazurczyk, W.; Wasielewska, K. Tight Arms Race: Overview of Current Malware Threats and Trends in Their Detection. *IEEE Access* 2021, 9, 5371–5396
- [2] Cabaj, K.; Caviglione, L.; Mazurczyk, W.; Wendzel, S.; Woodward, A.; Zander, S. The New Threats of Information Hiding: The Road Ahead. *IT Professional* 2018, 20, 31–3
- [3] Encodes a PowerShell script in the pixels of a PNG file and generates a oneliner to execute. <https://github.com/peewpw/Invoke-PSImage>. Accessed: 2022-01-18
- [4] Puchalski, D.; Caviglione, L.; Kozik, R.; Marzecki, A.; Krawczyk, S.; Choraś, M. Stegomalware Detection through Structural Analysis of Media Files. *Proc. 15th International Conference on Availability, Reliability and Security; Association for Computing Machinery: New York, NY, USA, 2020; ARES '20*.
- [5] Płachta, M.; Krzemień, M.; Szczypiorski, K.; Janicki, A. Detection of Image Steganography Using Deep Learning and Ensemble Classifiers. *Electronics* 2022, 11, 1565. <https://doi.org/10.3390/electronics11101565>
- [6] Yang, Z.; Wang, K.; Ma, S.; Huang, Y.; Kang, X.; Zhao, X. *IStego100K: Large-scale Image Steganalysis Dataset*. International Workshop on Digital Watermarking, Springer, 2019
- [7] Fridrich, J.; Pevný, T.; Kodovský, J. Statistically undetectable JPEG steganography: Dead ends, challenges, and opportunities. *the 9th ACM Multimedia & Security Workshop*. Association for Computing Machinery, 2007, p. 3–14.
- [8] Holub, V.; Fridrich, J.; Denemark, T. Universal distortion function for steganography in an arbitrary domain. *EURASIP Journal on Multimedia and Information Security* 2014
- [9] Guo, L.; Ni, J.; Su, W.; Tang, C.; Shi, Y.Q. Using Statistical Image Model for JPEG Steganography: Uniform Embedding Revisited. *IEEE transactions on information forensics and security* 2015
- [10] Holub, V.; Fridrich, J. Low-complexity features for JPEG steganalysis using undecimated DCT. *IEEE Transactions on Information Forensics and Security* 2015
- [11] Wang, C.; Feng, G. Calibration-based features for JPEG steganalysis using multi-level filter. *2015 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, 2015,
- [12] Song, X.; Liu, F.; Yang, C.; Luo, X.; Zhang, Y. Steganalysis of adaptive JPEG steganography using 2D Gabor filters. *Proceedings of the 3rd ACM Workshop on Information Hiding and Multimedia Security; Association for Computing Machinery: New York, NY, USA, 2015; IH&MMSec '15*,
- [13] Xia, C.; Guan, Q.; Zhao, X.; Xu, Z.; Ma, Y. Improving GFR Steganalysis Features by Using Gabor Symmetry and Weighted Histograms. *Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security; Association for Computing Machinery: New York, NY, USA, 2017; IH&MMSec '17*, p. 55–66.
- [14] Holub, V.; Fridrich, J. Phase-aware projection model for steganalysis of JPEG images. *Media Watermarking, Security, and Forensics 2015; Alattar, A.M.; Memon, N.D.; Heitzenrater, C.D., Eds. International Society for Optics and Photonics, SPIE, 2015, pp. 259 – 269*
- [15] Xu, G.; Wu, H.Z.; Shi, Y.Q. Structural Design of Convolutional Neural Networks for Steganalysis. *IEEE Signal Processing Letters* 2016, 23, 708–712
- [16] He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [17] Break Our Steganographic System Base webpage (BossBase). <http://agents.fel.cvut.cz/boss/>. Accessed: 2022-01-18

- [18] Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization, 2014.
- [19] Filler, T.; Judas, J.; Fridrich, J. Minimizing Embedding Impact in Steganography using Trellis-Coded Quantization. Media Forensics and Security II; Memon, N.D.; Dittmann, J.; Alattar, A.M.; III, E.J.D., Eds. International Society for Optics and Photonics, SPIE, 2010

AUTHORS

Mikolaj Plachta doctoral student at the Warsaw University of Technology, mobile application developer by profession, research area mainly focused on the study of the operation of neural networks in the application of steganography detection.

Artur Janicki university professor at the Cybersecurity Division of the Institute of Telecommunications, Warsaw University of Technology. His research and teaching activities focus on signal processing and machine learning, mostly in cybersecurity context. Member of technical program committees of various international conferences, reviewer for international journals in computer science and telecommunications. Author or co-author of over 80 conference and journal papers.

© 2022 By AIRCC Publishing Corporation. This article is published under the Creative Commons Attribution (CC BY) license.

EARLY DETECTION OF PARKINSON'S DISEASE USING MACHINE LEARNING AND CONVOLUTIONAL NEURAL NETWORKS FROM DRAWING MOVEMENTS

Sarah Fan¹ and Yu Sun²

¹Sage Hill School, 20402 Newport Coast Dr, Newport Beach, CA 92657

²California State Polytechnic University,
Pomona, CA, 91768, Irvine, CA 92620

ABSTRACT

Parkinson's disease (PD) is a progressive neurodegenerative disorder that causes uncontrollable movements and difficulty with balance and coordination. It is highly important for early detection of Parkinson's disease in order for patients to receive proper treatment. This paper aims to aid in the early detection of Parkinson's disease by using a convolutional neural network for PD detection from drawing movements. This CNN consists of 2 convolutional layers, 2 max-pooling layers, 2 dropout layers, 2 dense layers, and a flattened layer. Additionally, our approach explores multiple types of drawings, specifically spiral, meander, and wave datasets hand-drawn by patients and healthy controls to find the most effective one in the discrimination process. The models can be continuously trained in which the test data can be inputted to differentiate between healthy controls and PD patients. By analyzing the training and validation accuracy and loss, we were able to find the most appropriate model and dataset combination, which was the spiral drawing with an accuracy of 85%. With a proper model and a larger dataset for increased accuracy, this approach has the potential to be implemented in a clinical setting.

KEYWORDS

Machine Learning, Deep Learning, Parkinson Disease.

1. INTRODUCTION

Parkinson's Disease (PD) is a progressive disorder of the nervous system marked by tremors, muscular rigidity, and slow, imprecise movement, chiefly affecting middle-aged and elderly people [1]. It is associated with degeneration of the brain's basal ganglia and a deficiency of the neurotransmitter dopamine. Worldwide, around 7-10 million people have Parkinson's Disease [2], making it highly important to diagnose PD accurately in the early stage so that patients can receive proper treatment [3]. Parkinson's disease (PD) is difficult to diagnose, particularly in its early stages, because the symptoms of other neurologic disorders can be similar to those found in PD. Meanwhile, early non-motor symptoms of PD may be mild and can be caused by many other conditions. Therefore, these symptoms are often overlooked, making the diagnosis of PD at an early stage more challenging [4]. To address these difficulties and refine the early detection of PD, different neuroimaging techniques (such as magnetic resonance imaging (MRI), computed tomography (CT) and positron emission tomography (PET)) and deep learning-based analysis methods have been developed [5].

The rest of the paper is organized as follows: Section 2 describes our research background, direction, and the crucial elements needed in our research in detail; Section 3 presents our approach to solve the problem and relevant details about the experiment we did; Section 4 presents the results and analysis; Section 5 gives a brief summary of other work that tackles a similar problem; finally, Section 6 gives the conclusion remarks and discusses the future work of this project.

2. CHALLENGES

Figure 1 shows a chart of simplified machine learning applications. The research presented in this paper focused on the dark blue boxes as our research direction.

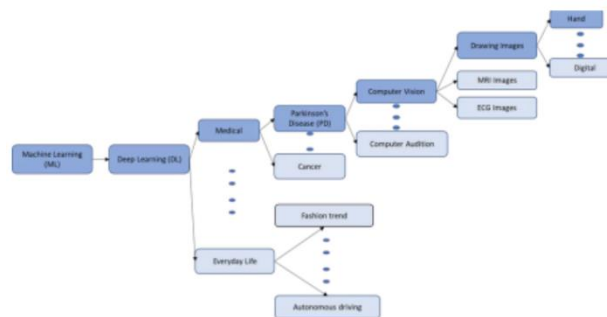


Figure 1. Machine Learning Application

2.1. Machine learning

Machine Learning is the technology of developing systems that can learn and draw inferences from patterns in data which can be applied to many different fields, from data analytics to predictive analytics, from service personalization to natural language processing, and so on [6].

According to Shalev-Shwartz, S. et al. [7], Machine learning can be defined as “using the experience to gain expertise.” The learning could be supervised learning, unsupervised learning, etc. Supervised learning is the most common approach and is the approach we utilize in our research.

Supervised learning algorithms try to model relationships between the target prediction output and input features to predict output values for new data based on the relationships learned from the prior data sets. This type of learning is normally related to classification tasks, which is the process of teaching a classifier the relationship between the model’s input and output to use this expertise later for un-seen input [8].

2.2. Deep Learning

Because machine learning is unable to meet the requirements due to the complexity of the problems in certain areas, Deep Learning (DL) is gaining popularity due to its supremacy in terms of accuracy. It is an advanced level of machine learning which includes a hierarchical function that enables machines to process data with a nonlinear approach. The deep learning networks are built with neuron nodes connected like the human brain and have many layers, each layer receiving information from the previous layer, trained to perform the desired tasks, and then passing on the information to the next layer [9].

Figure 1 shows that Deep Learning (DL) can be applied to fashion trend forecasting and autonomous driving, as examples for everyday life. Additionally, Deep Learning (DL) has also been applied to pharmaceutical research, such as cancer diagnosis [10] and Parkinson's Disease diagnosis [11] [12].

Within Parkinson's Disease diagnosis, there is research that focuses on computer audition while others focus on computer vision. Within the computer vision domain, the computer can help recognize and visualize electroencephalogram (EEG) signals automatically and help recognize and visualize brain scan images [5]. Some research focuses on the computer analyzing human drawn images [11] [12].

While human drawings can be hand drawn on paper or digitally, this paper's research interest is focused on using computer vision Convolutional Neural Network (CNN) to read/process hand-drawn drawings to help diagnose Parkinson's Disease.

2.3. Convolutional Neural Network

A convolutional neural network (CNN) can be made up of many layers of models, where each layer takes input from the previous layer, applies a filter to the data, and outputs it to the next layer. CNNs run much faster on GPU, and the huge stockpiles of data that have been collected can improve the accuracy of computer vision and NLP algorithms. A CNN consists of several convolutional layers, with each layer including three major stages: convolution, non linear activation (non linearity transform), and pooling (sub-sampling) [13].

2.4. Datasets

Datasets are fundamental in a deep learning system. An extensive and diverse dataset is a crucial requirement for the successful training of a deep neural network. In our research, we explore different CNNs using datasets we downloaded from HandPD [14] and Kaggle [15].

3. SOLUTION

The problem this research trying to solve can be summarized as the following

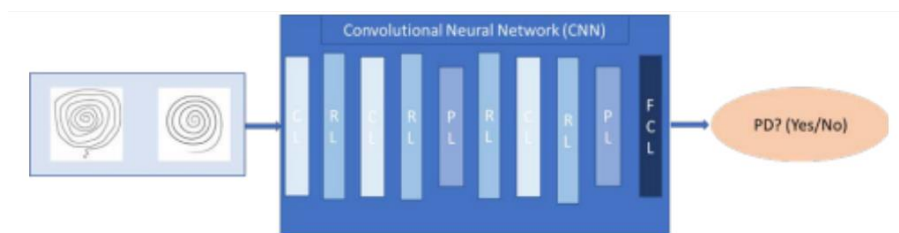


Figure 2. Problem Definition CL: Convolutional Layer, FCL: Fully Connected Layer, PL: Pooling Layer, RL: ReLU Layer

The dataset consists of hand drawn images (spiral/meander/wave) drawn by healthy people and Parkinson's disease patients. The model learns through training and uses a CNN to predict whether the person who drew the image has Parkinson's disease or not. The CNN model has one convolutional layer in front, a fully connected layer at the end, and a variable number of convolutional layers, max-pooling layers, and ReLU layers in between.

We downloaded HandPD dataset from [14]. The dataset contains 92 individuals, divided into 18 healthy people (Healthy Group) and 74 patients (Patients Group). Some examples are shown below. The brief description is the following:

- Healthy Group: 6 male and 12 female individuals with ages ranging from 19 to 79 years old (average age of 44.22 ± 16.53 years). Among those individuals, 2 are left-handed and 16 are right-handed.
- Patient Group: 59 male and 15 female individuals with ages ranging from 38 to 78 years old (average age of 58.75 ± 7.51 years). Among those individuals, 5 are left-handed and 69 are right-handed.

Therefore, each spiral and meander dataset is labeled in two groups: the healthy group containing 72 images, and the patient group containing 296 images. The images are labeled as follows: ID_EXAM-ID_IMAGE.jpg, in which ID_EXAM stands for the exam's identifier, and ID_IMAGE denotes the number of the image of the exam.

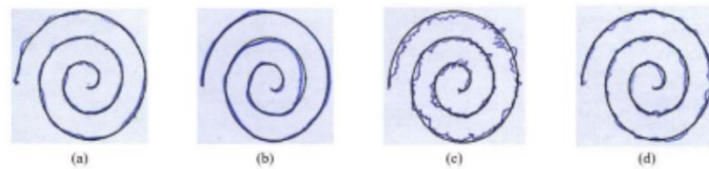


Figure 3. Some Examples of Spirals Extracted from the HandPD dataset [11]

Figure 3 shows (a) 58-year-old males (b) 28-year-old female individuals of the control group, (c) 56-year-old males, and (d) 65-year old female individuals of the patient group.

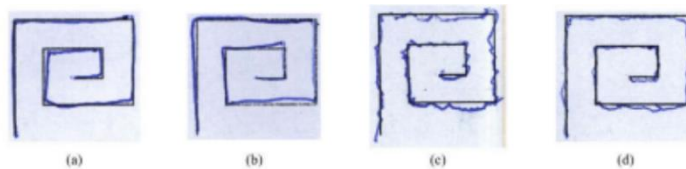


Figure 4. Some Examples of Meanders Extracted from HandPD Dataset[11]

Figure 4 shows (a) 58-years old male (b) 28-years old female individuals of a control group and (c) 56-years old mail and (d) 65-years old female individuals of a patient group.

All the data in HandPD dataset is in *.jpg format. For the exploration, we did some pre-processing including resizing, blurring, eroding, diluting, and color space converting (cv2.cvtColor() method).

The second dataset is downloaded from Kaggle [15]. The dataset has two patterns: wave and spiral. They are all in *.png format. The dataset is split into training and testing data. No personal information such as age and gender are available. We did not do any pre-processing for the data.

Wave drawing: there are 72 total wave drawings in the training data -- 36 drawn by Parkinson's disease patients and 36 drawn by healthy people. There are 30 total wave drawings in the testing data -- 15 drawn by Parkinson's disease patients and 15 drawn by healthy people. Figure 5 shows example drawings by Parkinson's disease patients and Figure 6 shows example drawings by healthy people.

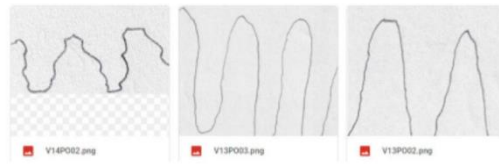


Figure 5. Wave Drawing Sample by Parkinson's Disease Patients from Kaggle [15]



Figure 6. Wave Drawing Sample by Healthy People from Kaggle [15]

Spiral drawing: there are 72 total spiral drawings in the training data -- 36 drawn by Parkinson's disease patients and 36 drawn by healthy people. There are 30 total spiral drawings in the testing data -- 15 drawn by Parkinson's disease patients and 15 drawn by healthy people. Figure 7 shows example drawings by Parkinson's disease patients and Figure 8 shows example drawings by healthy people.



Figure 7. Spiral Drawing by Parkinson's Disease Patients from Kaggle [15]



Figure 8. Spiral Drawing by Healthy People from Kaggle [15]

Our model consists of 2 convolutional layers, 2 max-pooling layers, 2 dropout layers, 2 dense layers, and one flattened layer. All the activation functions are ReLU. Figure 9 shows our model. We chose this particular CNN architecture since it gives good results [11] [27].

The dropout layer is a technique introduced by Srivastava et al. [31]. This layer aims to avoid overfitting by randomly ignoring randomly some neurons from the previous layer. We inserted the dropout layers to improve the performance of our model.

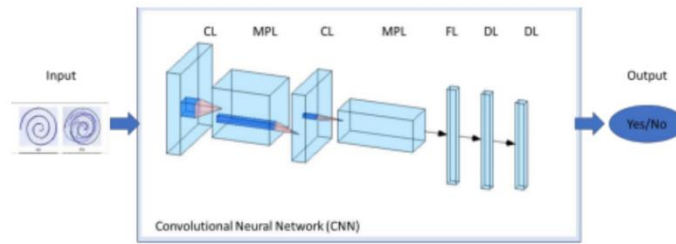


Figure 9. The Proposed CNN Architecture

4. EXPERIMENT

Figure 10 shows the training and validation accuracy and loss. The dataset we used is the spiral dataset downloaded from HandPD [14] without pre-processing. The CNN we used is the one shown in Figure 9. As we can see, it has a severe overfitting problem. To resolve this issue, we added dropout [31] after max-pooling. Figure 11 shows the validation accuracy and loss after dropout was added, preventing the model from overfitting and minimizing validation loss.

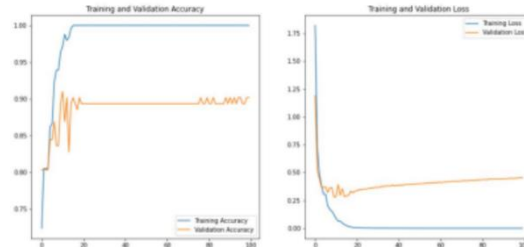


Figure 10. Pre-processing Spiral Data from HandPD [14] Using the Model Shown in Figure 9

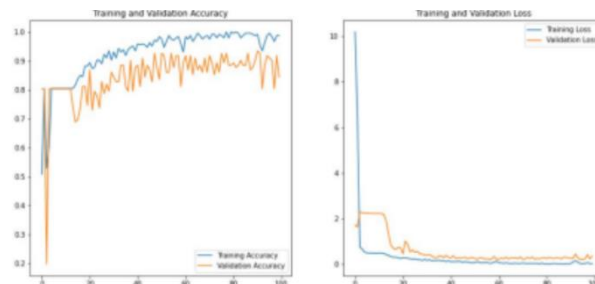


Figure 11. Spiral Data from HandPD [14] with Two Dropout Layers Added after Max-pooling

To see the effects of different drawing patterns, we used two different patterns from the same dataset with the same CNN, with dropout added after max-pooling. Figure 11 uses spiral data from HandPD while Figure 12 uses meander data from HandPD. Comparing Figure 11 and Figure 12 we can see that both patterns generate similar validation accuracy and validation loss results, with the spiral slightly more accurate.

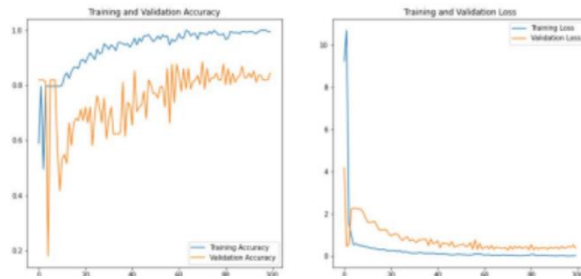


Figure 12. Meander Data from HandPD [14] with Two Dropout Layers Added after Max-pooling

Figure 13 shows a CNN model proposed by M. Alissa [27]. It consists of 6 convolutional layers, three max-pooling layers, three dense layers and one flatten layer. It's much more complicated and the training/validation time is much longer. We ran both meander data from HandPD (shown in Figure 14) and spiral data from HandPD (shown in Figure 15).

The comparison shows that even though our proposed CNN model is much simpler, it generates better results. This shows that we need a suitable CNN, not necessarily one that is more complicated.

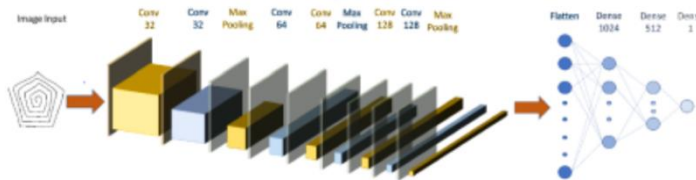


Figure 13. CNN Model Proposed by M. Alissa [27]

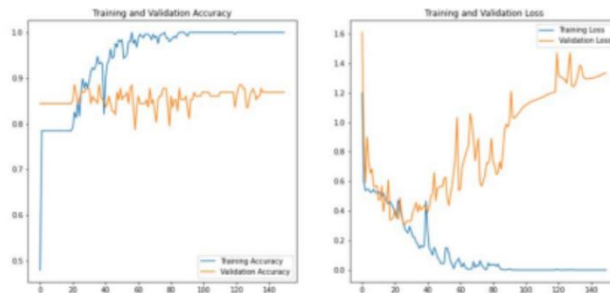


Figure 14. Meander Data from HandPD [14] using CNN Shown in Figure 13

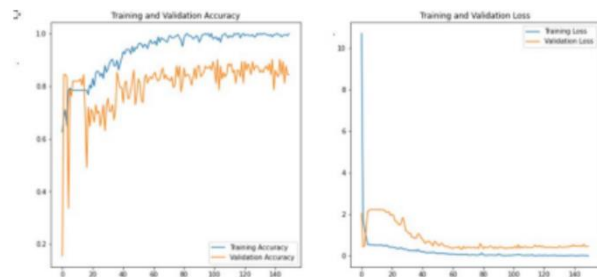


Figure 15. Meander Data from HandPD [14] using Our Proposed CNN shown in Figure 9

Using the same CNN shown in Figure 9, Figure 15 uses pre-processed meander data from HandPD, with added post processing. The results with added post processing are shown in Figure 16. Comparing Figure 15 and Figure 16, the post-processed data did not generate better results.

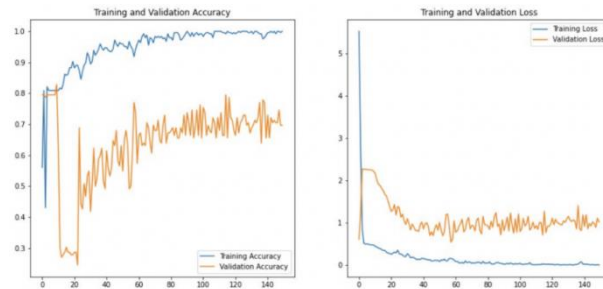


Figure 16. Post-processed Meander Data from HandPD [14] using CNN shown in Figure 9

To compare different datasets, we ran experiments with wave and spiral data from Kaggle using the CNN shown in Figure 9. The wave data results are shown in Figure 17, while the spiral data results are shown in Figure 18. Because the data from Kaggle is in *.png format, the dataset itself is much smaller and not much pro-processing could be done. Therefore, the results are not as accurate as when we use the dataset from HandPD.

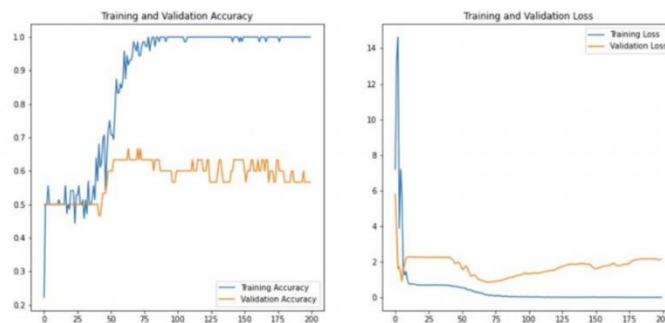


Figure 17. Wave Data from Kaggle [15] using CNN Model in Figure 9

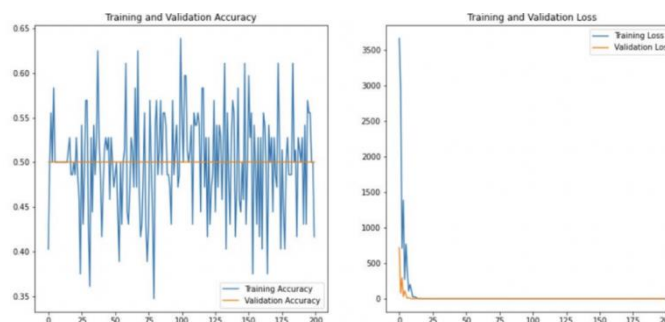


Figure 18. Spiral Data from Kaggle [15] using CNN Shown in Figure 9

5. RELATED WORK

Several researchers have worked on the diagnosis of Parkinson's Disease by using machine learning methods, e.g. diagnosis using voice, diagnosis using brain scan images, diagnosis drawings such as meander patterns, spirals, waves, etc.

J. Mei et al. [16] did a review of literature on machine learning for the diagnosis of Parkinson's disease, using sound, MRI images, and hand-drawn images. It searches IEEE Xplore and PubMed. It reviewed research articles published from the year 2009 onwards and summarized data sources and sample size.

The public repositories and databases include HandPD [14], Kaggle dataset [15], the University of California at Irvine (UCI) Machine Learning Repository [17], Parkinson's Progression Markers Initiative (PPMI) [18], PhysioNet [19], etc.

Quite a few researchers use magnetic resonance images (MRI) or their variations as their research dataset. Noor et al. [5] surveyed the application of deep learning in detecting neurological disorders from magnetic resonance images (MRI) in the detection of Parkinson's disease, Alzheimer's disease, and schizophrenia.

E. Huseyn et al. [20] [21] used MRI images as their dataset. S. Chakraborty [22] and X. Zhang [23] has used a dataset from Parkinson's Progression Markers Initiative (PPMI). Z.Cai et al. [24] used an enhanced fuzzy k-nearest neighbor (FKNN) method for the early detection of Parkinson's Disease based on vocal measurements. L. Badea et al. [25] explored the reproducibility of functional connectivity alterations in Parkinson's Disease based on resting-state fMRI scans images.

Pereira et al. did a series of research on automatic detecting Classify Parkinson's disease for many years. At first, they used non-deep learning algorithms in diagnosing PD [26]. They collected/constructed a public dataset called "HandPD" [14]. Based on this dataset, they compared the efficiency of different hand drawn patterns in the diagnosis of PD [11]. Their results show that the meander pattern generates more accurate results compared to the spiral pattern. However, in our research, the spiral and meander patterns generate similar results when they are trained and tested through the same CNN.

Later, they explored the use of CNN on the images extracted from time-series signals and used three different CNN architectures, ImageNet, CIFAR-10, and LeNet as baseline approach [12]. In her master project, M. Alissa [27] used non-public datasets (spiral pentagon dataset) to evaluate the efficiency of two different neural networks (Recursive Neural Networks(RNN) and Convolutional Neural Networks (CNN)). We built a CNN similar to hers and used the dataset from HandPD [14] and Kaggle [15] to evaluate different CNNs and different datasets.

Gil-Martin et al. [28] presented a method to detect Parkinson's Disease from drawing movements using Convolutional Neural Networks. He used the dataset from the UCI machine learning repository as input data, applied signal-processing (sampling with 100 Hz and 140 Hz, resampling with 110 Hz, perform Hamming windowing and FFT) to generate preprocessed data, and used this data to train/validate the CNNs.

M.E. Isenkul et al. [29] designed an improved spiral test dataset using a digitized graphics tablet for monitoring Parkinson's Disease. Digitized graphics have more information, including timestamps, grip angles, and hand pressure, etc. The significance of that can be investigated in future work.

P.Zham [30] presented a dataset at Kaggle [15] with waves and spirals. He used a composite index of speed and pen-pressure to distinguish different stages of Parkinson's Disease.

6. CONCLUSIONS

Our results show that to get the best results from a deep learning system, we need a good dataset and a suitable CNN, rather than a complicated one. Furthermore, not all pre-processing led to better results.

Bigger datasets: Both HandPD and Kaggle datasets are too small, containing not enough data. To train and make a better model, we need to collect more data, possibly with an app or a collaboration with a hospital/organization [31].

Imbalanced datasets: The data from HandPD and Kaggle are imbalanced as we described in section 3.2. This might mislead the classifier where our models classified all the test sets as patients' [32]. The solution is to increase the number of images drawn by healthy people using augmentation or downsampling the images drawn by patients. However, both ways have their limitations: the augmentation makes the data not real anymore, while the downsampling makes the dataset smaller.

Model improvement: In the future we can explore other deep learning techniques, such as k-fold cross-validation, multiple-stage deep CNN architecture, etc. [33].

REFERENCES

- [1] M. Bhat, "Parkinson's Disease prediction based on hand tremor analysis," IEEE 2017 International Conference, 2017.
- [2] "Parkinson's Disease Statistics," Parkinson's News Today, 2021. [Online]. Available: <https://parkinsonsnewstoday.com/parkinsons-disease-statistics/>. [Accessed 3rd September 2022].
- [3] Editorial Team, "Diagnosis-Early Symptoms & Early Diagnosis," parkinsonsdisease.net, 8 March 2017. [Online]. Available: <https://parkinsonsdisease.net/diagnosis/early-symptoms-signs/>. [Accessed 3rd September 2022].
- [4] Blog post, "You could have Parkinson's disease symptoms in your 30s or 40s and not know it," health direct, 11 April 2019. [Online]. Available: <https://www.healthdirect.gov.au/blog/parkinsons-disease-symptoms-in-your-30s-40s>. [Accessed 3 September 2022].
- [5] M. Noor, "Application of deep learning in detecting neurological disorders from magnetic resonance images: a survey on the detection of Alzheimer's disease, Parkinson's disease and schizophrenia," Brain Informatics, 9th October 2020.
- [6] T. blog, "Guide to machine learning applications: 7 Major Fields," The APP solution, 2021. [Online]. Available: <https://theappsolutions.com/blog/development/machine-learning-applications-guide/>. [Accessed 3rd September 2022].
- [7] S. Shalev-Shwartz, "Understanding machine learning: From theory to algorithms," Cambridge University Press, 2014.
- [8] I. H. Witten, "Data Mining: Practical Machine Learning Tools and Techniques, Third Edition," Morgan Kaufmann Series in Data Management Systems, 2011.
- [9] Trending Blog, "What is Deep Learning? and What are its Significance Deep Learning Trends," ALPHA Information Systems INDIA PVT LTD, 15 September 2019. [Online]. Available: <https://www.aalpha.net/blog/what-is-deep-learning-and-what-are-its-significance/>. [Accessed 3rd September 2022].
- [10] A. Cruz-Roa, "A deep learning architecture for image representation, visual interpretability, and automated basal-cell carcinoma cancer detection.," In International Conference on Medical Image Computing and Computer-Assisted Intervention, 2013.
- [11] C. Pereira, "Deep Learning-Aided Parkinson's Disease Diagnosis from Handwritten Dynamics.," Graphics. Patterns and Images (SIBGRAPI) 2016 29th SIBGRAPI Conference IEEE, 2016.
- [12] C. Pereira, "Convolutional neural networks applied for Parkinson's disease identification," Machine Learning for Health Informatics, 2017.
- [13] H. Wang, "On the origin of deep learning," arXiv preprint arXiv, 2017.

- [14] C. Pereira, "Welcome to the HandPD dataset," HandPD, 2017. [Online]. Available: <https://www.fc.unesp.br/~papa/pub/datasets/Handpd/>. [Accessed 3rd September 2022].
- [15] K. Mader, "Parkinson's Drawings," Kaggle.com, 2019. [Online]. Available: <https://www.kaggle.com/kmader/parkinsons-drawings>. [Accessed 27 September 2021].
- [16] J. Mei, "Machine Learning for the Diagnosis of Parkinson's disease: a review of literature," *Frontiers in Aging Neuroscience*, 2021.
- [17] D. Graff, "UCI Machine Learning Repository," University of California, Irvine.
- [18] K. Marek, "The Parkinson Progression Marker Initiative (PPMI)," *Progress Neurobiol*, 2011.
- [19] A. L. Goldberger, "PhysioBank, Physio Toolkit and PhysioNet: Components of a new research resource for complex physiologic signals," *Circulation*, 2000.
- [20] E. Huseyn, "Deep Learning Based Early Diagnostis of Parkinsons Disease," Cornell University arXiv.org, August 2020.
- [21] S. Shinde, "Predictive markers for Parkinson's disease using deep neural nets on neuromelanin sensitive MRI," bioRxiv the preprint server for biology, 2019.
- [22] S. Chakraborty, "Detection of Parkinson's Disease from 3T T1 weighted MRI scans using 3D convolutional neural network," *diagnostics (MDPI)*, 2020.
- [23] X. Zhang, "Multi-View graph convolutional network and its applications on neuroimage analysis for parkinson's disease," *Amia annual symposium proceedings archive*, 2018.
- [24] Z. Cai, "An Intelligent Parkinson's Disease Diagnostic," *Hindawi.com, Computational and Mathematical Methods in Medicine*, 2018.
- [25] L. Badea, "Exploring the reproducibility of functional connectivity alterations in Parkinson's disease," *National Institute for Research and Development in Informatics, Bucharest, Romania*.
- [26] C. Pereira, "A step towards the automated diagnosis of parkinson's disease: Analyzing handwriting movements," in *28th International Symposium on Computer-Based Medical Systems (Sao Carlos: IEEE)*, 2015.
- [27] M. Alisssa, "Master's Project: Parkinson's Disease Diagnosis Using Deep Learning," *Heriot Watt University*, 2018.
- [28] M. Gil-Martin, "Parkinson's Disease Detection from Drawing movements using convolutional neural networks," in *Electronics (MDPI)*, 2019.
- [29] M Isenkul, "Improved Spiral Test using a Digitized Graphics Tablet for Monitoring Parkinson's Disease," in *International Conference on e-Health and Telemedicine*, 2014.
- [30] P. Zham, "Distinguishing Different Stages of Parkinson's Disease Using Composite Index of Speed and Pen-Pressure of Sketching a Spiral," in *Frontiers in Neurology*, 2017.
- [31] N. Srivastava, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting.," *The Journal of Machine Learning Research.*, 2014.
- [32] Y. Xue, "Application of Deep Learning in Automated Analysis of molecular images in cancer: a survey," in *Hindawi Computational and Mathematical Methods in Medicine*, 2017.
- [33] R. Ruizendaal, "Deep Learning #3: More on CNNs & Handling Overfitting," *Towards data science*, 12 May 2017. [Online]. Available: <https://towardsdatascience.com/deep-learning-3-more-on-cnns-handling-overfitting-2bd5d99abe5d>. [Accessed 3rd September 2022].

BANTU SPELL CHECKER AND CORRECTOR USING MODIFIED EDIT DISTANCE ALGORITHM (MEDA)

Boago Okgetheng, Gabofetswe Malema,
Ariq Ahmer, Boemo Lenyibi and Ontiretse Ishmael

Department of Computer Science,
University of Botswana, Gaborone, Botswana

ABSTRACT

Automatic spelling correction for a language is critical since the current world is almost entirely dependent on digital devices that employ electronic keyboards. Correct spelling adds to textual document accessibility and readability. Many NLP applications, such as web search engines, text summarization, sentiment analysis, and so on, rely on automatic spelling correction. A few efforts on automatic spelling correction in Bantu languages have been completed; however, the numbers are insufficient. We proposed a spell checker for typed words based on the Modified minimum edit distance Algorithm (MEDA), and the Syllable Error Detection Algorithm (SEDA). In this study, we adjusted the minimal edit distance Algorithm by including a frequency score for letters and ordered operations. The SEDA identifies the component of the word and the position of the letter which has an error. For this research, the Setswana language was utilized for testing, and other languages related to Setswana will use this spell checker. Setswana is a Bantu language spoken mostly in Botswana, South Africa, and Namibia and its automatic spelling correction are still in its early stages. Setswana is Botswana's national language and is mostly utilized in schools and government offices. The accuracy was measured in 2500 Setswana words for assessment. The SEDA discovered incorrect Setswana words with 99% accuracy. When evaluating MEDA, the edit distance algorithm was utilized as the baseline, and it generated an accuracy of 52%. In comparison, the edit distance algorithm with ordered operations provided 64% accuracy, and MEDA produced 92% accuracy. The model failed in the closely related terms.

KEYWORDS

Bantu Spell Checker, Edit Distance algorithm, morphologically rich, Syllable Error Detection Algorithm.

1. INTRODUCTION

The written form of a language is a significant aspect in communication. We do not need to know right spelling of a word when speaking, and communications can be done even without proper grammatical attention, but perfect spelling is critical when writing. Spelling errors are prevalent, especially in a complex language especially in Bantu languages which are morphologically rich. Most of Bantu languages are agglutinative with heavy use of affixes, suffixes, and prefixes. The most notable aspects involving Bantu syllable structure, consonant/vowel inventories, and phonological processes are firmly established throughout the Bantu region [1], despite the enormous number of languages and vast geographic expanse that they span. The notion of what constitutes a word is therefore different from European languages like English.

Spelling mistakes are widespread in Bantu languages due to the difficult word formation. Spelling mistakes are not necessarily caused by their complexity; they can also arise because of rapid typing or a lack of expertise and some are generated from other close terms. Spelling errors could be divided into two types: typographical errors and cognitive errors [2]. Cognitive errors include phonetic errors as well as errors associated with homonyms, which can result in a valid word that is incorrect in context. When developing a spell-checking tool, these two types of errors must be considered.

Since spelling mistakes are so common, automatic spelling corrective systems are critical. The practice of detecting improperly written words in a particular language is known as spell checking [3]. The spell-checking process may also make suggestions for improperly misspelled terms in the language. Many potential NLP applications, like text summarization, sentiment analysis, and machine translation, require spelling correction. [4]. There has been a lot of research on this area, although there aren't many significant publications on Spell checkers in Bantu languages. Some studies employed the minimal edit distance approach in conjunction with a considerably bigger dictionary/database.

We present a multilingual Bantu spell checker and corrector that employ a modified Edit distance algorithm (MEDA) and Syllable error detection Algorithm (SEDA), with a focus on spell application usability. The method employed SEDA to detect the error in the term and MEDA to fix it. The MEDA scores the frequency of the letters in the word using a mathematical method. The frequency approach calculates the likelihood of letter *a* being followed by letter *b*. MEDA additionally incorporates the optimal operation ordering from the edit distance for efficient and effective outcomes. This proposed paradigm was tested using Setswana, a Bantu language. This study is divided into the following sections: literature review, methodology, implementation, results, results analysis, and conclusions.

2. LITERATURE REVIEW

With the digitization of almost everything becoming the norm, in this case, text data, it is important to have spell checker tools in word processors or software programs put in place to validate and verify textual information. Although error detection and correction are the basis of a professional spell checker, thorough investigations indicate that for the Bantu languages this has not been conducted and is still in its primitive stages due to the languages' rich morphology and high agglutination. Spellchecking entails both error correction and detection. Text error correction has mostly concentrated on three areas: non-word error detection, isolated-word error detection, and context-dependent word correction. Error detection has been successfully implemented; however, error repair is currently being worked on incrementally.

2.1. Existing Systems

There are some existing techniques that are used for spell checking. These include R-rule based technique system – where a number of associated rules captures frequent spelling errors and typos and uses them to on the misspelled words, Dictionary lookup – where the word is checked against a dictionary and N-gram technique [5]. Other techniques include using Clustering Algorithm (PAM) to identify errors in the spelling [6] and using statistical model to check for spelling errors [7].

2.2. Related Work

A study in the development of a Turkish spell-checking and error-correcting application by Ince [8] also suggests that the development of a spell checker and error corrector for an agglutinative language proves to be challenging and different. The author emphasizes that for this to take place; mathematical preliminaries and morphological analysis are required for such languages. Ince developed a spell checker and error corrector for the Turkish language using the n-gram and edit distance algorithms using the C#.Net and Microsoft Visual Studio platform. An nZembrek file was then used as the Turkish dictionary's sustainability of the word roots and compliance with suffixes. The file contained a text file, a suffix file, a letter file, and an optional syllable finder. Ince says the n-gram was used to learn the rules of the letters and the word sequences and was dependent on the word length the user input. The n-gram was used to determine the distance of a misspelled word as n , where $(n-1)$ and $(n-2)$ are the misspelled word's properties. The edit distance algorithm was used to add the nearest 2 distance words to the suggestion list. The results obtained showed an accuracy of 95% and a success rate of 95% for suggestions for spelling errors.

Mjaria [9] conducted a study similar and came to the same conclusion that stated although it has become a common norm for the use of spellcheckers due to the increasing usage of text-based communication, be it at the workplace or on social media, it has been very gruesome when it comes to support for spellchecking of Bantu languages. Spellcheckers are designed such that the body consists of text representing the language, basically the "corpus", an error model, and a language model. The language model is used for determining how frequently a word occurs in a dictionary. The error model is just used for the modelling of spelling errors, whose accuracy is liable to the corpus.

Bantu languages are mostly known to be ambiguous, thus making it difficult for error correction when it comes to spell checking. Mashiane [10] pointed that there are yet efficient algorithms and tools that are yet to be found when it comes to error correction of Bantu languages. In this work a minimum edit distance algorithm was used for error detection, where the algorithm searches for the smallest number of insertions or deletions of a word. The structure of African languages differs significantly from that of the languages supported by conventional spell checker software. It has been easier for non-word error detection compared to error correction. A binary spell checker has been used to achieve this. Spelling errors in languages with complex morphology and structure are common and could either be cognitive or typographical in nature, so error corrections are required.

A discovery was made by Islam [11], who conducted research on the Bangla language, which was prone to errors due to its complex structure. Islam created a model for correct word prediction in Bangla using the fractional accuracy function for non-error words and the direct dictionary method and edit distance algorithm to check correctness and word suggestions and compute the accuracy. Islam further states that in handling the typographical non-word errors, direct dictionary lookup was used to detect erroneous words, and if the typed word was not available, it was an error. The edit distance algorithm was then used to correct the word with an estimate of the similarity of its structure to the misspelled word and possible correct words. The model produced an accuracy of 98.83% from a suggestion list of length 9 and an accuracy of 65.17% from a suggestion list of length 1. Contrary to the use of the edit distance algorithm, the reverse edit distance algorithm may be used in spell checkers as it uses fewer word comparisons and has proved to be efficient.

Iqbal [12], in his study of creating a spell checker for the Urdu language, used the reverse edit distance algorithm. Iqbal indicated that the reverse edit distance algorithm minimized the

comparisons between misspelled words and words in the dictionary for error correction. The reverse edit distance proposed for the Urdu language uses the edit distance permutation, which is generated and compared to the lexicon word in alphabetical order. With the Urdu language comprised of 42 alphabets, a word length of n resulted in a total of $86n+41$ comparisons. The edit distance algorithm was then used on the generated permutation to avoid the occurrence of non-word errors that resulted in a zero match in the lexicon. Some of the difficulties faced in building the Urdu spell checking model, according to Iqbal, were its complicated morphological structure, diction problems, word separators, and loan words.

3. METHODOLOGY

Our method for spell Checker is to extend the existing edit distance algorithm to make it work for Bantu languages, based on how words are formed or written. The proposed has been described in this section.

3.1. Syllable Error Detection Algorithm (SEDA)

In our methodology, we first detect the dissimilarities between the words using our error detection algorithm. We firstly input documents containing the words of that language in the model. Then the model learns the different types of syllables present in the language and stores them in a file. Then we take our input words and pass them through our SEDA. This model then extracts the syllables and looks them up the file of syllables to look for any inconsistencies. An inconsistency represents that the model detected an error in the word and then outputs that error to the next phase.

3.2. Edit Distance Algorithm (EDA)

We now consider the Edit Distance Algorithm (EDA). This algorithm attempts to determine the difference between two words and the fewest number of operations required to change one word to another. [13]. The error that is identified from the SEDA is then passed to this phase of the Edit Distance Algorithm. The EDA takes this error into account and performs its operations on the word to be transformed. We propose a system of ordering of the fundamental operations of EDA: insert, delete, substitute and transpose. The ordering of these operations is done to optimize the discovery of the smallest number of operations needed to change the word. Section 3.3 shows the orderings of the operations that are proposed for the transformation.

3.3. EDA with Ordering

In this step, take into account the order of operations that are available for the general EDA. We run the proposed ordering of operations (as shown in below) and run each order against the EDA to get a general performance of which ordering to take. An average score for each of the ordering is calculated which gives the performance of the ordering. This score determines which operations ordering will be used as we assume that this performs the best on average for every word. We determine the average performance for each operation and then arrange the results based on the scores. This is done to investigate how the ordering of operations affects the spell checker in Bantu languages. The best operation ordering will then be used in our MEDA.

Insert Ordering

1. insert , replace , delete, transpose
2. insert, transpose, replace, delete

3. insert, delete, transpose , replace
4. insert, replace, transpose, delete
5. insert, transpose , delete , replace
6. insert, delete, replace, transpose

Transpose Ordering

1. transpose, insert , delete, replace
2. transpose, insert, replace, delete
3. transpose, replace, delete, insert
4. transpose, replace, insert, delete
5. transpose, delete ,insert, replace
6. transpose, delete, replace, insert

Delete Ordering

1. delete, insert , transpose , replace
2. delete, insert, replace, transpose
3. delete, replace, transpose, insert
4. delete, replace, insert, transpose
5. delete, transpose, insert, replace
6. delete, transpose, replace, insert

Replace Ordering

1. replace , delete, transpose, insert
2. replace, delete, insert, transpose
3. replace, insert, delete, transpose
4. replace, insert, transpose, delete
5. replace, transpose, delete, insert
6. replace, transpose, insert, delete

3.4. Modified Edit Distance Algorithm (MEDA)

MEDA is a component of our suggested solution since it employs algorithms based on the structure of Bantu languages. This algorithm incorporates operations ordering and a mathematical model together with the general EDA. The mathematical model is used to find the score of each alphabet in the given word relative to its position in that word. This allows us to find the probable letter that may be the error in the word. Once this score is found, we employ the ordered operations of the general EDA to the “incorrect” word to find the minimum number of operations to be taken to transform the word. The operations of the EDA are ordered to explore which set of operations are optimized for the error correction of any given word in time and to see whether ordering said operations actually influences the final yield.

3.4.1. Mathematical Model

Let $W = \{c_0, c_1, \dots, c_n\}$ be the set of characters in a given word W .

Let $D_{js} = \{\{a_{00}, a_{10}, \dots, a_{j0}\}, \dots, \{a_{0s}, a_{1s}, \dots, a_{js}\}\}$ be the dictionary that consists of words whose characters are denoted by a_{js} of a word D_s .

Let $p = |W|, q = |D_s|, r = |D|$ where D_s a word in the dictionary is at any given time.

Let $O = \{O_0, O_1, \dots, O_n\}, 0 \leq n \leq p$ be the set consisting of objects that hold the frequency of each character c_n of the word W . Then the frequency of the characters relative to their position is calculated as follows.

$$O_i = \frac{1}{r} \sum_{k=0}^r \Phi_i$$

Where,

$$\Phi_i = \begin{cases} 1, & \text{if } c_i = D_{js}, i = j, 0 \leq j \leq q, 0 \leq s \leq r \\ 0, & \text{Otherwise} \end{cases}$$

The function Φ basically keeps track of how many times the letter c_i appears in the position i in the set of words in D . If there is a character that is common in that position, then a 1 is added to Φ and 0 if the characters don't match. The figure is then normalized by taking the average of the set Φ over the total number of words in the dictionary and place it in the i -th position of set O . Note that the position of the elements in the set W match with that of set O and $|W| = |O|$.

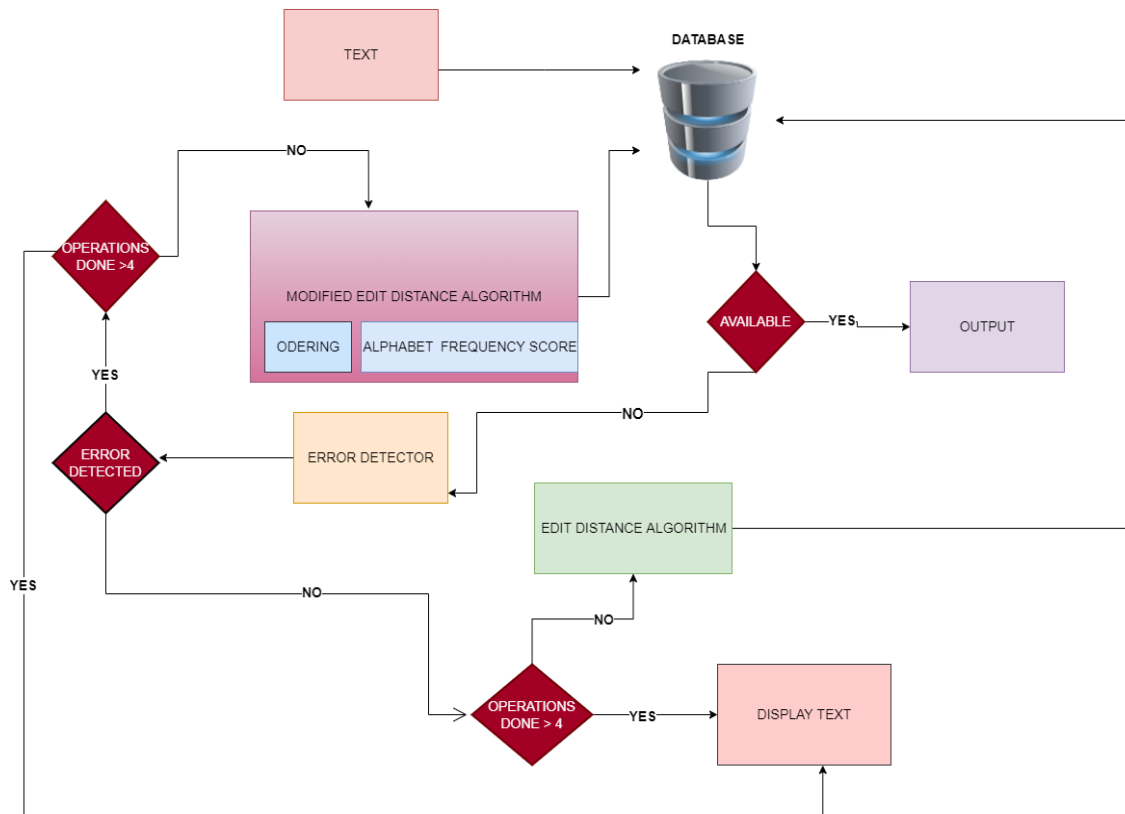


Figure 1. Proposed Solution

We start off by getting a word input. The word is then checked against a dictionary of known words. If the word appears in the dictionary, we assume that it is accurate, therefore we display it right away. In the case that it is not found, the word is passed through an error detector. If there are no errors detected and we also know that the word is non-existent in the dictionary, then the word is passed through the Edit Distance algorithm (EDA) in four passes. The operations in each

pass are accounted for before passing to EDA. Each pass consists of an operation of the following: Insert, Delete, Replace and Transpose. The operations are ordered such that the optimized operation is performed on average. If, after executing the aforementioned operations, the word is still not located in the dictionary, we conclude that it does not exist and display it directly. However, if there is (are) error(s) detected after passing through the error detector, then the word is passed to check if the said number of operations (as mentioned above) are done before passing it to “Modified Edit Distant” algorithm (MEDA). In this method, the algorithm detects where the error is and then splits the syllables and characters. Each character is then evaluated with a frequency score (see section 3.4.1) relative to its position in the word. The character with the least score is taken to be the anomaly. Then, the four standard operations are applied to this character in an attempt to correct the word. After the operations are done, the word is checked against the dictionary to see if it exists. If the word does not exist and we exceed the number of operations, then we conclude that the word is non-existent and display it accordingly.

4. IMPLEMENTATION

4.1. Data Pre Processing

Test Data

We created a json file including Setswana words with right Setswana words and different erroneous words next to each, for example "bolela": "bolel blela bulela" the first is the proper word 'bolela' and the following are alternative mistake terms. The target word is then used to construct a list of spelling mutations within a distance of two, yielding the wrong Setswana terms. This file served as an input. We used 2500 correct terms and 7728 wrong words. These were nouns and verbs in Setswana.

```

1  ["abang": "abing", "aba": "abn", "amogela":
2  "amogella amogel", "atla": "atha", "adima":
3  "adim adema", "adimela": "ademela", "apaya": "apaa apya",
4  "apayang": "apayng", "agisa": "agiisa", "arabisisa": "arbisis",
5  "amusa": "amisa amsa", "akanya": "aknya", "akela": "akle",
6  "atlana": "atlna atlna", "bolotsa": "bulotsa", "bapatsa": "baphatsa",
7  "betla": "beta", "betsega": "btsega", "bidile": "bedele", "bintsha":
8  "bintsa bitsha", "bidikamisa": "bedikamisa bidekamesa bidekamisa bidikama bidikamasa", "botsosolosa": "botsoslosa botsosolo",
9  "bina": "bena", "baka": "bak",
10 "bidisa": "bidis bedesa", "bankanya": "bankana", "bakisa": "bkisa",
11 "biletsa": "biltsa bilets billetsa",
12 "boifa": "biofa", "boifisisa": "boifsisa", "bokegelwa": "bokegelw",
13 "bolotsana": "bolotsna", "batola": "batlola", "budusa": "bodusa",
14 "belega": "bilega", "belegisa": "belegsa", "bitisiwa":
15 "btisiwa bitlisiwa botisiwa", "balela": "balila balla", "dikologa": "dkologa dekologa", "dirisiwa":
16 "deresewa deresiwa", "dipa": "depa", "dirolola": "dirolola", "dia": "dhia", "emelela": "emella", "epa": "epha", "eta": "etla",
17 "etsa": "etsha", "ega": "eha", "ema": "emha",
18 "eletsa": "eiletsa", "emisa": "emsa"

```

Figure 2. Test Data Sample

Dictionary/database

This is a list of Setswana terms. The dictionary is used as a lookup to see if the term is in the dictionary; if it is, we presume it is the proper targeted words. If the entered word is not accessible, it is considered an error.

4.2. Module Implementation

We used python as a programming language for implementation. Python comes with a support for advanced text manipulation that is easy to code which in our case meant that it helped in pre-processing our text input easily.

4.2.1. Frequency score Module

We require a corpus of words from a certain language that has most of the forms, or a method that understands how to generate all of the forms of a word. We next performed our frequency score method, which gave us the probability of specific alphabets following each other. To train the model, we utilized a big dataset containing Setswana phrases to score our alphabets in Setswana. Figure 3 illustrates the findings of the frequency score for the Setswana word "bontshantse."

```
The word is: bontshantse
The relative frequencies are:
[{'char': 'b', 'index': 0, 'frequency': 0.08}, {'char': 'o', 'index': 1, 'frequency': 0.13}, {'char': 'n', 'index': 2, 'frequency': 0.05}, {'char': 't', 'index': 3, 'frequency': 0.07}, {'char': 's', 'index': 4, 'frequency': 0.06}, {'char': 'h', 'index': 5, 'frequency': 0.02}, {'char': 'a', 'index': 6, 'frequency': 0.1}, {'char': 'n', 'index': 7, 'frequency': 0.06}, {'char': 't', 'index': 8, 'frequency': 0.04}, {'char': 's', 'index': 9, 'frequency': 0.04}, {'char': 'e', 'index': 10, 'frequency': 0.04}]
```

Figure 3. Frequency score

4.2.2. EDA, EDA by ordering and MEDA Modules

We implemented the three approaches, and they were all tested with the same dataset of 2500 Setswana words. The results of each approach were recoded as indicated in section 5.

5. RESULTS

A set of words with varying morphologies from a text were used as input to test the performance of the three techniques. The accuracy was measured by comparing the correct number of words to the total number of words provided, as shown in equation 1 below.

Equation 1. Accuracy [14]

$$\text{Accuracy} = \frac{\text{No. of correct words}}{\text{No. of words}} \times 100$$

To evaluate the spelling behaviour of Setswana words, the three methodologies were explored. In this experiment, 2500 words were analysed, and the accuracy of each approach was measured. Except for overlapping terms, the SEDA detector 99 percent accurate (Words which are wrong but having correct syllables). The edit distance method by Levenstein has 52 percent accuracy in correcting misspelled words, the edit distance algorithm by ordering has 64 percent accuracy, and the modified edit distance algorithm has 92 percent accuracy. The failure causes are described in the section on outcomes.

Edit Distance Algorithm by ordering

In our test input, we ordered the operations. In this case, we modified the Edit distance Algorithm to avoid using the shortest distance and instead complete the processes in chronological order. The key is **I-Insert, R-Replace, T-Transpose, and D-Delete**. We wanted to see the order in which Setswana words may be corrected so that we determine the most prevalent Setswana word error (it be swapping, omission or addition). The accuracy of all orders was calculated, as well as the average from each operation, as shown in tables 1, 2, 3, and 4. The average accuracy of the four operations is 64.3 percent for transpose, 59.9 percent for insert, and 56.9 percent for replace, with delete being the least accurate at 56.6 percent. The precision is depicted in figure 4 below.

Table 1. Accuracy of Transpose Ordering

Ordering	No. of words	No. correct words	No. incorrect words	Accuracy rate (%)
1. T,I,R,D	2500	1617	883	64.7
2. T,I,D,R	2500	1610	830	64.4
3. T,R,D,I	2500	1597	903	63.9
4. T,R,I,D	2500	1610	890	64.4
5. T,D,I,R	2500	1605	895	64.2
6. T,D,R,I	2500	1597	903	63.9
Average Accuracy				64.3

Table 2. Accuracy of Insertion ordering

Ordering	No. of words	No. correct words	No. incorrect words	Accuracy rate (%)
1. I,R,D,T	2500	1595	705	63.8
2. I,T,R,D	2500	1607	893	64.3
3. I,D,T,R	2500	1425	1075	57.0
4. I,R,T,D	2500	1460	1040	58.4
5. I,T,D,R	2500	1412	1088	56.5
6. I,D,R,T	2500	1485	1015	59.4
Average Accuracy				59.9

Table 3. Accuracy of Delete ordering

Ordering	No. of words	No. correct words	No. incorrect words	Accuracy rate (%)
1 D,I,T,R	2500	1460	1040	58.4
2 D,I,R,T	2500	1460	1040	58.4
3 D,R,T,I	2500	1365	1135	54.6
4 D,R,I,T	2500	1375	1125	55.0
5 D,T,I,R	2500	1450	1050	58.0
6 D,T,R,I	2500	1378	1122	55.1
Average Accuracy				56.6

Table 4. Accuracy of Replace ordering

Ordering	No. of words	No. correct words	No. incorrect words	Accuracy rate (%)
1 R,D,T,I	2500	1378	1122	55.1
2 R,D,I,T	2500	1390	1110	55.6
3 R,I,D,T	2500	1460	1040	58.4
4 R,I,T,D	2500	1463	1037	58.5
5 R,T,D,I	2500	1425	1075	57.0
6 R,T,I,D	2500	1413	1087	56.5
Average Accuracy				56.9

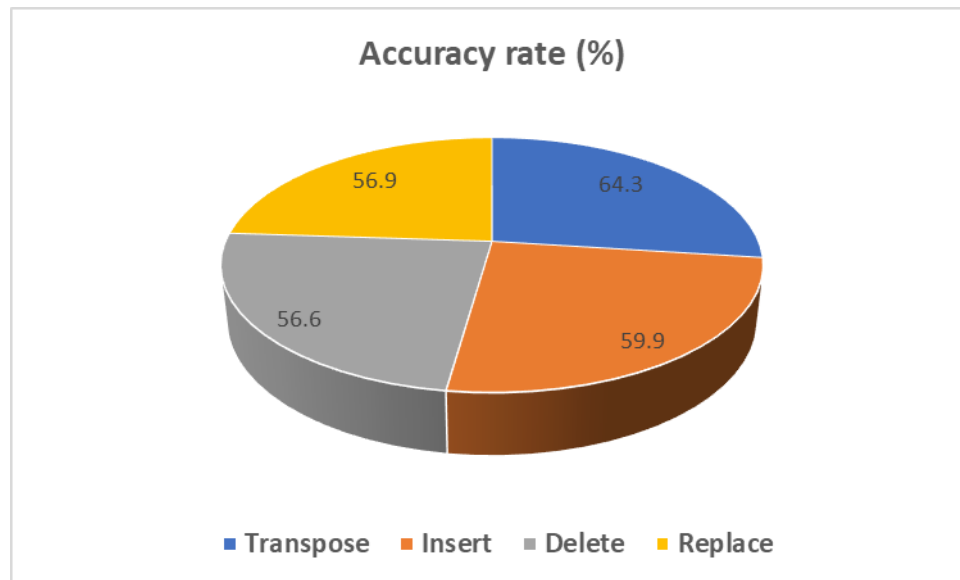


Figure 4. Accuracy per operation

From figure 4, it is clear that starting with the transpose operation yielded the best result. And in order of accuracy the order of operations {T, I, R, D} yielded best result which is in accordance with the average results obtained per operations order set. As this order was the best performing, we adopted this in our proposed solution (MEDA) which gave us optimal results

Edit Distance Algorithm

The **edit distance** is a measure of how similar two strings are. This checks the minimum number of editing operations. The accuracy of this algorithm given Setswana words gave 52% as indicated in table 5.

Table 5. Accuracy of the three Methods

No. of Words	Accuracy (%)		
	Edit Distance Algorithm	Edit Distance Algorithm by Ordering	Modified Edit Distance Algorithm (MEDA)
2500	52%	64.3%	92%

Modified Edit Distance Algorithm

This recommended option outperformed the other two described approaches above. Table 5 shows that the Modified algorithm performed at 92 percent. As the approach employed the score of the alphabets in terms of following each other, this is a clear depiction that the Setswana words are constructed in a consistent pattern. The MEDA also employed the syllables i , $i+1$ where i represents the position of the word's incorrect syllable and the syllable at. Because Setswana has few syllables, this made it easy for the algorithm to correct the words.

The algorithm corrects the syllable and checks to see if the word is valid; if not, it employs a mathematical method that scores each letter in a word. In this example, it begins with the order of operations from the ordering Edit distance algorithm. In this instance, for example, we have "thlola" as the input word (incorrect spelling), and the correct word is "tlhola." The term has three syllables: "th", "lo", and "la". And the error detector found a mistake in the initial syllable "th," which is not a proper syllable in Setswana. The MEDA then attempted to remedy this by using the nearest syllable, "tho," and it produces a correct Setswana word, "Tholola," which is in the dictionary but not the intended term. The frequency score will then be used by the MEDA to perform the operations in the ordering. In this situation, we have "th" and "lo," which are syllables at i and $i+1$, respectively. Because the scoring of l following h is low, we perform operations on l. We begin with transpose, which means we swap h and l to get "Tlhola," which is the proper and predicted word. If the term is incorrect, we proceed with the other operations in the order listed above.

6. RESULTS DISCUSSIONS

Different factors influenced the MEDA performance. Setswana is morphologically rich, which means that most of the words are derived from other words, therefore there is a good probability that if someone misses the intended word they wanted to write, they will receive a legitimate Setswana term. Also, because most words are derived from other words, utilizing the edit distance algorithm to do different operations may result in the word being rectified by any operation to yield an acceptable Setswana word when it was not the desired word. The 8% was from the words which were added as correct while were not the desired one. The following are examples of Setswana word errors which affected the performance of MEDA.

6.1. Omission and Addition of a Character

Botswana has various ethnic groupings in different demographics northern and southern regions. This influences how they speak Setswana. People in the north commonly omit the 'l' letter while pronouncing some Setswana words, which influences how they write it. This affected MEDA performance, because some words, when 'l' is omitted, they fall into a different term which is legitimate and most of the time having a different meaning. E.g.

+beta 'beta' betla
 +thapa 'thapa' tlhapa
 +fapa 'fapa' fapha
 +ikgantse 'ikgantse' ikgantse

According to the examples above, 'beta' is an acceptable Setswana term, however some people pronounce 'betla' as beta, and beta was found in the dictionary whereas we wanted it to give us betla. The edit distance is the shortest distance and represents the number of operations required in a word to produce a valid word. Then there could be 0 operations, indicating that the word was already in the dictionary. As though we expected 'tlhapa' and the user typed 'thapa' because most individuals who have a problem with this 'l' character sometimes add it when it is not supposed to be included. And certain Setswana words have the letter 'l,' which will confirm the term as correct. This can also happen with the character 'h,' as shown in the list above.

6.2. Swapping of characters

This is mainly a keyboard mistake; there are characters near to one other in the "QWERTY" keyboard, and when one of them is used, it can result in a legitimate word. For example, u and l are on the same row of the keyboard. We anticipated the term amusa in the example below, but because the user made an error by inputting u, it becomes legal because amisa is a word in Setswana.e.g

+amisa amisa amusa

6.3. Morphology

Morphology is the study of words, how they are formed, and their relationship to other words in the same language. It analyzes the structure of words and parts of words, such as stems, root words, prefixes, and suffixes. So, there are words in Setswana which has suffixes/ prefixes, which when removed the remaining sub-string becomes valid.

For example if we perform delete operation, it affects most words which are coming from the other ones, usually words starting with *n,i,m* in Setswana, when we delete the first character, it gives us a valid Setswana word.

+ fitesa 'itesa' 'Fitisa'
 +nkgakololo 'kgakololo' 'nkgakolole'

6.4. Closed Words

Since most words are closed to each other, and even though inserting ordering performing better, it has bad effects. E.g., Setswana words are almost the same but not meaning the same thing, we can say they are too close to each other. When we give insertion priority, we can end up in totally different words. E.g., the word 'lametsa' was misspelled and expecting the word 'lamatsa' and inserting with change the first letter with different letter and it will give us nametsa which is correct.

+ lametsa 'nametsa' 'lamatsa'
 + deresewa 'keresega' 'dirisiwa'

7. DISCUSSIONS

Due to the fact that the bulk of words in Setswana are loanwords, there is a high probability of accidentally writing a term that is both acceptable and not the one intended. This study revealed that all four edit distance operations are essential because they affect how prospective words are mutated, but the findings also indicated that the order in which the operations are carried out matters. MEDA is a morphologically rich language edit distance method. By utilizing the syllable algorithm and the frequency score of each letter in the word, this technique efficiently locates errors in words. The algorithm was tested in Setswana words and received a 92 percent, and it might be tested in additional languages which uses the same writing style.

REFERENCES

- [1] Hyman, Larry M. , "Segmental Phonology" , in *The Bantu Languages* ed. Derek Nurse and Gerard Philippson (Abingdon: Routledge, 03 Jul 2003) , accessed 07 Jul 2022 , Routledge Handbooks Online.
- [2] F. J. Damerau, "A technique for computer detection and correction of spelling errors," *Commun. ACM*, vol. 7, pp.171-176, 1964
- [3] Walfish, M., Hachamovitch, & Andrew, F. (2000). Patent No 6047300. United States of America.
- [4] Yonatan , B., and Yonatan B. (2017). Synthetic and natural noise both break neural machine translation. arXiv preprint arXiv:1711.02173.
- [5] Patil, C., Rodrigues, R., & Ron, R. (2020). Auto-Spelling Checker using Natural Language Processing.
- [6] Cordeiro De Amorim, R. and Zampieri, M., 2022. Effective Spell Checking Methods Using Clustering Algorithms. [online] Uhra.herts.ac.uk. Available at: <<https://uhra.herts.ac.uk/handle/2299/16916>> [Accessed 7 July 2022].
- [7] M. S. Rasooli, O. Kahefi and B. Minaei-Bidgoli, "Effect of adaptive spell checking in Persian," 2011 7th International Conference on Natural Language Processing and Knowledge Engineering, 2011, pp. 161-164, doi: 10.1109/NLPKE.2011.6138186.
- [8] İnce, E. Y. (2017). Spell checking and error correcting application for Turkish. *International Journal of Information and Electronics Engineering*, 7(2), 68-71.
- [9] Mjaria, F., & Keet, C. M. (2018, May). A statistical approach to error correction for isiZulu spellcheckers. In *2018 IST-Africa Week Conference (IST-Africa)* (pp. 1-of). IEEE.
- [10] Mashiane, N. An overview of digitization of African languages, spellchecking techniques & the progress of spellcheckers globally.
- [11] Islam, M. I. K., Meem, R. I., Kasem, F. B. A., Rakshit, A., & Habib, M. T. (2019, May). Bangla spell checking and correction using edit distance. In *2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT)* (pp. 1-4). IEEE
- [12] Iqbal, S., Anwar, W., Bajwa, U. I., & Rehman, Z. (2013, October). Urdu spell checking: Reverse edit distance approach. In *Proceedings of the 4th workshop on south and southeast asian natural language processing* (pp. 58-65).
- [13] Haldar, Rishin & Mukhopadhyay, Debajyoti. (2011). Levenshtein Distance Technique in Dictionary Lookup Methods: An Improved Approach. *Computing Research Repository - CORR*

AUTHORS

B. Okgetheng: MSc, BSc Computer Science.



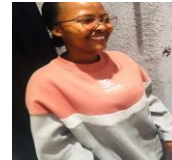
G. Malema: PhD Computer Engineering, MS Electrical and Computer Science, BS Computer Engineering.



Ariq Ahmer: Bsc Computer Science Student.



Boemo Lenyibi: Bsc Computing with Finance Student.



O. Ishmael: MSc Computer Science, BSc Computer Systems Engineering.



AUTHOR INDEX

<i>Ali Retha Hasoon Khayeat</i>	49
<i>Ariq Ahmer</i>	303
<i>Artur Janicki</i>	281
<i>Asif Ekbal</i>	191
<i>Bernard Espinasse</i>	221
<i>Brigitta Nagy</i>	01
<i>Boago Okgetheng</i>	303
<i>Boemo Lenyibi</i>	303
<i>Calvin Huang</i>	115
<i>David Tang</i>	249
<i>Dorián László Galat</i>	01
<i>Erhan Guven</i>	179
<i>Gabofetswe Malema</i>	303
<i>Gabriel Melo</i>	167
<i>Guilherme Wachs-Lopes</i>	167
<i>Jae-Hyung Koo</i>	13
<i>Jean-Pierre Corriveau</i>	75
<i>Jianhua Deng</i>	263
<i>Jingye Cai</i>	263
<i>KaykeBonafé</i>	167
<i>Kristóf Csorba</i>	01
<i>Kyung-Yup Kim</i>	13
<i>Lemlem Kassa</i>	263
<i>Makoto Murakami</i>	237
<i>Mark Davis</i>	263
<i>Masoumeh Mohammadi</i>	209
<i>Michael DeLeo</i>	179
<i>Mikolaj Plachta</i>	281
<i>Mohamed Azouz Mrad</i>	01
<i>Núria Gala</i>	221
<i>Pinar Yildirim</i>	105
<i>Prashant Kapil</i>	191
<i>Qinqin Guo</i>	37
<i>Ontiretse Ishmael</i>	303
<i>Radhwan Adnan Dakhil</i>	49
<i>Rita Hijazi</i>	221
<i>Sang-Wook Kim</i>	13
<i>Sarah Fan</i>	291
<i>Saranyanath K P</i>	75
<i>Shadi Tavakoli</i>	209
<i>Shin-Hwan Kim</i>	13
<i>Tianyu Li</i>	145
<i>Tina Yazdizadeh</i>	131
<i>Tony Zheng</i>	95
<i>Wei Shi</i>	75,131

<i>Xiaohan Feng</i>	237
<i>Xingyu Zheng</i>	25
<i>Xuanxi Kuang</i>	65
<i>Yaoshen Yu</i>	25
<i>Yifei Tong</i>	157
<i>Yongchao Wang</i>	25
<i>Yu Sun</i>	37,65,95,115
<i>Yu Sun</i>	145,157,249,291
<i>Zhiqiu Huang</i>	25
<i>Zsombor Kristóf Nagy</i>	01