Well-Posed Learning Problems and Designing Learning System

Dr. Kothuri Parashu Ramulu[1], Dr. Jogannagari Malla Reddy[2]
[1]*Associate Professor, Indur Institute of Engineering & Technology*
[2]*Professor, Mahaveer Institute of Science & Technology*

***Abstract*—Machine learning is the study of computer algorithms that can improve automatically through experience and by the use of data. It is seen as a part of artificial intelligence. A computer program is said to learn from experience with respect to some set of tasks and performance measure , if its performance at set of tasks improves with experience. A well-defined learning problem will have the features like class of tasks, the measure of performance to be improved, and the source of experience examples. To get a successful learning system, it should be designed properly, for a proper design several steps may be followed for perfect and efficient system.**

***Index Terms— Direct feedback, Indirect feedback,* Estimating Training Values, LMS, Performance System, Generalizer, Critic, Experiment Generator.**

I. INTRODUCTION

A computer program is said to learn from experience E with respect to some set of tasks T and performance measure P, if its performance at set of tasks in T, as measured by P, improves with experience E. A well-defined learning problem, have to identify three features: The class of tasks, the measure of performance to be improved, the source of experience Examples [3]. The various examples are Checkers game**:** A computer program that learns to play checkers might improve its performance as measured by its ability to win at the class of tasks involving playing checkers games, through experience obtained by playing games against it.

A checkers learning problem: Task T→playing checkers, Performance measure P→ percent of games won against opponents, Training experience E→ playing practice games against itself. The checkers game board will be as in figure 1.1.

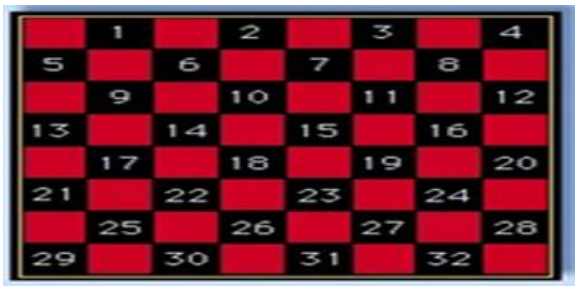

Figure 1.1

A handwriting recognition learning problem: Task T→recognizing and classifying handwritten words within images, Performance measure P→ percent of words correctly classified, Training experience E→a database of handwritten words with given classifications.

A robot driving learning problem: Task T→ driving on public four-lane highways using vision sensors [2], Performance measure P→average distance travelled before an error (as judged by human overseer) Training experience E→ a sequence of images and steering commands recorded while observing a human driver

II. DESIGNING LEARNING SYSTEM

To get a successful learning system, it should be designed properly, for a proper design several steps may be followed for perfect and efficient system [1].

The basic design issues and approaches to machine learning are illustrated by designing a program to learn to play checkers, with the goal of entering it in the world checkers tournament

1. Choosing the Training Experience
2. Choosing the Target Function
3. Choosing a Representation for the Target Function
4. Choosing a Function Approximation Algorithm
   a. Estimating training values
   b. Adjusting the weights
5. The Final Design

Diagrammatical representation of designing learning system is as in figure 2.1.

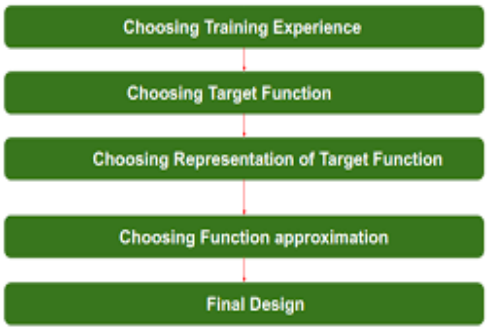

Figure 2.1

III. CHOOSING THE TRAINING EXPERIENCE

The first design step is to choose the type of training experience from which the system will learn. The type of training experience which is available can have significant impact on success or failure of the learning system.

A. Whether the training Experience provides direct or indirect feedback regarding the choices made by the performance system

To do a task T, performance P needs to be good which also requires good experience E.

Good training will give good experience; training is of two types, direct feedback training and indirect feedback training. Direct feedback will give the result of the move simultaneously (driving a car with the help of coach sitting besides), indirect feedback will give the hints on moves (driving a car by listening recorded sessions). Learner can select direct feedback or indirect feedback.

B. The degree to which the learner controls the sequence of training examples

Up to what extent the learner can control the sequence of training examples. Example the learner driving the car with full help of the trainer or partial help of the trainer or learner driving the car without the help of the trainer.

C. How well it represents the distribution of examples over which the final system performance P must be measured

The diagrammatical representation of the attributes of choosing the training experience is as in figure 2.2

IV. CHOOSING THE TARGET FUNCTION

The next design step is to determine exactly what type of knowledge will be learned and how this will be used by the performance program.

Consider the checkers problem

Let us therefore define the target value V(b) for an arbitrary board state b in B, as follows:

1. if b is a final board state that is won, then V(b) = 100
2. if b is a final board state that is lost, then V(b) = -100
3. if b is a final board state that is drawn, then V(b) = 0
4. if b is a not a final state in the game, then V(b) = V(b'), where b' is the best final board state that can be achieved starting from b and playing optimally until the end of the game (assuming the opponent plays optimally, as well).

While this recursive definition specifies a value of V(b) for every board state b, this definition is not usable by our checkers player because it is not efficiently computable. Except for the trivial cases (cases 1-3) in which the game has already ended, determining the value of V(b) for a particular board state requires (case 4) searching ahead for the optimal line of play, all the way to the end of the game! Because this definition is not efficiently computable by our checkers playing program, we say that it is a nonoperational definition. The goal of learning in this case is to discover an operational description of V; that is, a description that can be used by the checkers-playing program to evaluate states and select moves within realistic time bounds. Thus, we have reduced the learning task in this case to the problem of discovering an operational description of the ideal target function V [4]. It may be very difficult in general to learn such an operational form of V perfectly. In fact, we often expect learning algorithms to acquire only some approximation to the target function, and for this reason the process of learning the target function is often called function approximation. In the current discussion we will use the symbol $\hat{v}$ to refer to the function that is actually learned by our program, to distinguish it from the ideal target function V.

V. CHOOSING A REPRESENTATION FOR THE TARGET FUNCTION

Let us choose a simple representation: for any given board state, the function c will be calculated as a linear combination of the following board features:

1. x_1: the number of black pieces on the board
2. x_2: the number of red pieces on the board
3. x_3: the number of black kings on the board
4. x_4: the number of red kings on the board
5. x_5: the number of black pieces threatened by red (i.e., which can be captured on red's next turn)
6. x_6: the number of red pieces threatened by black

Thus, our learning program will represent $\hat{v}(b)$ as a linear function of the form

$$\hat{v}(b) = w_0+w_1x_1+w_2x_2+w_3x_3+w_4x_4+w_5x_5+w_6x_6$$

Where W_o through W_6 are numerical coefficients, or weights, to be chosen by the learning algorithm.

Learned values for the weights W_1 through W_6 will determine the relative importance of the various board features in determining the value of the board, whereas the weight W_o will provide an additive constant to the board value.

To summarize our design choices thus far, we have elaborated the original formulation of the learning problem by choosing a type of training experience, a target function to be learned, and a representation for this target function [8]. Our elaborated learning task is now

Partial design of a checkers learning program:

Task T: playing checkers

Performance measure P: percent of games won in the world tournament

Training experience E: games played against itself

Target function: V: Board →R

Target function representation

$$\hat{v}(b) = w_0+w_1x_1+w_2x_2+w_3x_3+w_4x_4+w_5x_5+w_6x_6$$

The first three items above correspond to the specification of the learning task, whereas the final two items constitute design choices for the implementation of the learning program. Notice the net effect of this set of design choices is to reduce the problem of learning a checkers strategy to the problem of learning values for the coefficients w_0 through w6 in the target function representation.

VI.CHOOSING A FUNCTION APPROXIMATION ALGORITHM

Set of training examples are required to train the target function $\hat{v}$, each training example describe a specific board state b and the training value $V_{train}(b)$ for b. Each training example is an ordered pair of the form (b, $V_{train}(b)$). For example, the following training example describes a board state b in which black has won the game, i.e. $X_2 = 0$ indicates that red has no remaining pieces then the target function $V_{train}(b)$ value will be +100.

$((X_1=3, X_2=0, X_3=1, X_4=0, X_5=0, X_6=0), +100)$

Now, we describe a procedure that first derives such training examples from the indirect training experience available to the learner, then adjusts the weights w_i to best fit these training examples.

A. Estimating Training Values

It is easy to assign a value to board states that correspond to the end of the game (0, -100 or +100) [5], but estimating training values of intermediate board states b is $V_{train}(b)$. Assign $\hat{v}(Successor(b))$for V_{train} (b), where $\hat{v}$ is the learner's current approximation to V and *Successor(b)* denotes the next board state following the b for which it is again it is again the program turn to move.

Rule for estimating the training values.

$V_{train}(b) \leftarrow \hat{v}(Successor(b))$

B. Adjusting the weights

All that remains is to specify the learning algorithm for choosing the weights wi to best fit the set of training examples {(b, Vtrain(b))}. As a first step we must define what we mean by the bestfit to the training data. One common approach is to define the best hypothesis, or set of weights, as that which minimizes the squared error [6] E between the training values and the values predicted by the hypothesis $\hat{\mathbf{v}}$.

$$E \equiv \sum_{\langle b, V_{train}(b)\rangle \in \text{ training examples}} (V_{train}(b) - \hat{V}(b))^2$$

Thus, we seek the weights, or equivalently the $\hat{v}$, that minimize E for the observed training examples. The LMS algorithm is defined as follows:

LMS weight update rule.

For each training example $\langle b, V_{train}(b)\rangle$

- Use the current weights to calculate $\hat{V}(b)$
- For each weight w_i, update it as

$$w_i \leftarrow w_i + \eta\ (V_{train}(b) - \hat{V}(b))\ x_i$$

VII. THE FINAL DESIGN

The final design of checkers learning system can be described by four distinct program modules that

represent the central components in many learning systems

These four modules, summarized in Figure 7.1are The Performance System, is the module that must solve the given performance task, in this case playing checkers, by using the learned target function(s)[7].

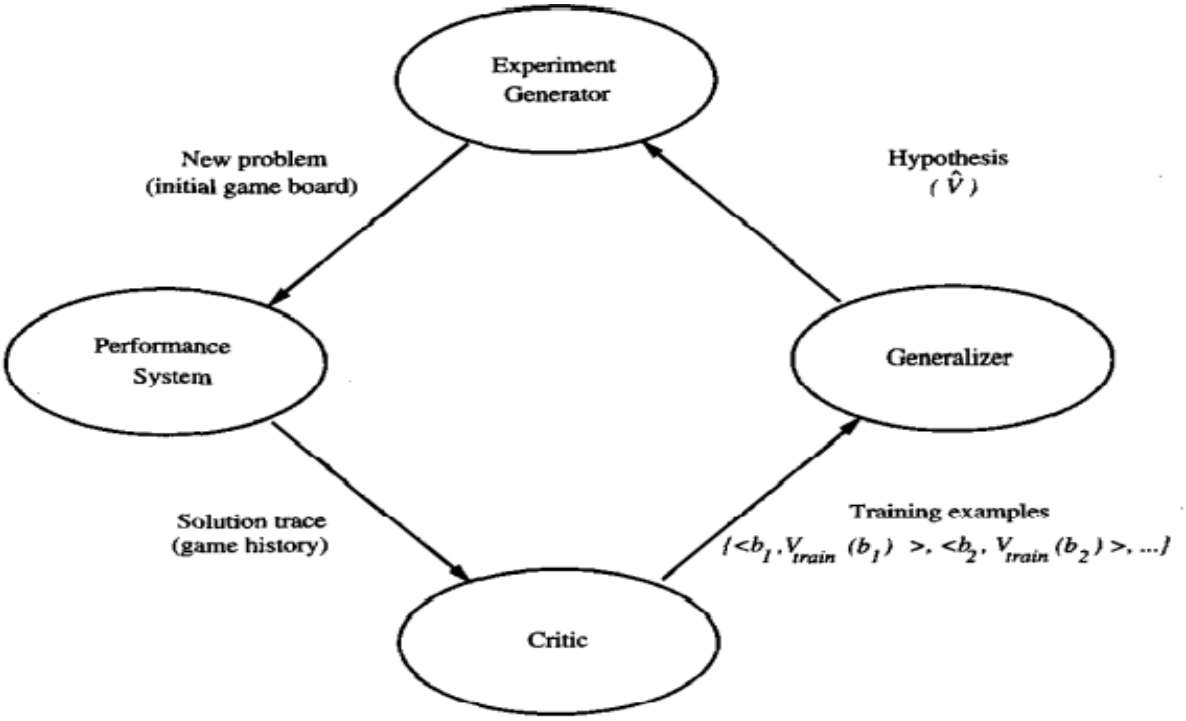

Figure 7.1

The Critic takes as input the history or trace of the game and produces as output a set of training examples of the target function. The Generalizer takes as input the training examples and produces an output hypothesis that is its estimate of the target function. The Experiment Generator takes as input the current hypothesis (currently learned function) and outputs a new problem (i.e., initial board state) for the Performance System to explore. Its role is to pick new practice problems that will maximize the learning rate of the overall system.

VIII. CONCLUSION

Machine learning is the part of Artificial Intelligence, is the study of computer algorithms that can improve automatically through experience and by the use of data. A computer program is said to learn from experience with respect to some set of tasks and performance measure, if its performance at set of tasks improves with experience. A well-defined learning problem will have the features like class of tasks, the measure of performance to be improved, and the source of experience examples. To get a successful learning system, it should be designed properly, for a proper design several steps may be followed for perfect and efficient system.

ACKNOWLEDGMENT

We are thankful to our friends, faculty members and management for their help in writing this paper.

REFERENCES

[1] Luis Alfaro1, Claudia Rivera2, Jorge A Luna-Urquizo, “Using Projectbased Learning in a Hybrid e-Learning System Model”, International Journal of Advanced Computer Science and Applications, Vol. 10, No.10, pp. 426-436, 2019.

[2] Markowska-Kaczmar U., Kwasnicka H., Paradowski M., ”Intelligent Techniques in Personalization of Learning in e-Learning Systems”, Studies in Computational Intelligence, Springer, Berlin, Heidelberg, Vol. 273, pp. 1–23, 2010.

[3] Etienne, Wenger. "Artificial intelligence A.and tutoring systems." Computational and Cognitive Approaches to the Communication of Knowledge. Morgan Kauffmann, Los Altos, San Francisco, CA USA, 1987.
[4] Ohlsson, Stellan. "Some principles of intelligent tutoring", Instructional Science, Vol.3, No.4, pp. 293-326, 1986.
[5] Mark Nichols, “A theory for eLearning,” Journal of Educational Technology & Society, Vol. 6, No. 2, pp. 1-10, 2003.
[6] M. Alavi. “Computer mediated collaborative learning: An empirical, evaluation”, MIS Quart., Vol.18, No.2, pp.159-174, 1994.
[7] Nicola Henze, Peter Dolog and Wolfgang Nejdl, “Reasoning and Ontologies for Personalized E-Learning in the Semantic Web,” Journal of Educational Technology & Society, Vol. 7, No. 4, pp. 82-97, 2004.
[8] Jeroen J. G., van Merriënboer and Paul A. Kirschner, “Ten Steps to Complex Learning A Systematic Approach to Four-Component Instructional Design,” Taylor & Francis Group, 2017.
[9] Zhanga, Ye Jun, and Bo Songb. “A Personalized e-Learning System Based on GWT.” In: International Conference on Education, Management, Commerce and Society, pp. 183-187, 2015.