Thèse n° 9923

From event-based surprise to lifelong learning. A journey in the timescales of adaptation

Présentée le 6 juillet 2023

Faculté des sciences de la vie
Laboratoire de calcul neuromimétique (SV/IC)
Programme doctoral en neurosciences

pour l'obtention du grade de Docteur ès Sciences

par

Martin Louis Lucien Rémy BARRY

Acceptée sur proposition du jury

Prof. C. Petersen, président du jury
Prof. W. Gerstner, directeur de thèse
Dr G. Keller, rapporteur
Prof. H. Sprekeler, rapporteur
Prof. M. Mathis, rapporteuse

École
polytechnique
fédérale
de Lausanne

2023

Les hommes n’ont plus le temps de rien connaître.
— Antoine de Saint-Exupéry, Le petit prince

To Noé

Acknowledgements

I will start these acknowledgments by thanking my family. My parents Nadine Ruolt and Pierre-Michel Barry—in probable order of meeting—for all the support they gave me over the years. All the Discussions and mental support from them helped me go through this Ph.D. My siblings Juliette Barry, Nicolas Barry, and Ugo Philippe-in alphabetical order—for the joy they bring me on the rare occasions our paths may cross. I take this opportunity to send all my love to my grandparents Julienne, Daniel, Julien, and Monique. I am grateful to my three nephews Marcel, Lucien*, and José Bardet Barry – In order of appearance— for being the cutest. I also want to thank Laure Gourgue and Simon Bardet without whom this family would be incomplete. I also want to give special thanks to my regretted Tatie and all the family Bottin that, even though I don't see much anymore, holds a special place in my life. I also want to thank the close family friends Lune Diallo, Mustapha Ayad, and Régis Broy that have always been around since I can remember.

Moving on, I want to express my deepest gratitude to the Wattignies group, including Pierre-Antoine Lemaître, Armand Rémy, Flavien Wallart, Théo Boitel, Alexandre Fernandez$^{\heartsuit}$, Thibaud Dorchies, Quentin Barbry, Thomas Guilloton, and Morgan Sarpaux – in estimated height order – for all the great times and memories for so long now. I would also like to extend my thanks to Adrien Brière, with whom I have shared countless unforgettable moments and holidays.

No acknowledgment would be complete without mentioning my two closest Ph.D. friends and flatmates Paul Rolland and Matthieu Broisin —in order of blondness— with whom I had the best of times during these 10 years at EPFL, even if Paul fled during the lockdown... Special thanks also to Ivan Perret for quitting his Ph.D. when I wanted to leave mine, unknowingly forcing me to continue.

I am truly grateful to my Ph.D. supervisor, Wulfram Gerstner, for trusting me and giving me the opportunity to make a Ph.D in his wonderful lab. I am thankful for his unwavering patience and guidance throughout my academic journey. I also want to thank all the people I had the pleasure to share the last years of my Ph.D. with Valentin Schmutz, Louis Pezon, Berfin Simsek, Flavio Martinelli, Johanni Brea, Georgios Iatropoulos, Guillaume Bellec, Shuqi Wang, Ariane Delrocq, and Sophia Becker – ordered by topics – and the ones that left before Dane Cornell, Samuel Muscinelli, Marco Lehman, Vasiliki Liakoni, Chiara Gastaldi – ordered by departure. I must add special thanks to Christos Sourmpis for the gains, Alireza Modirshanechi for the meaningful times and discussions, Florian Colombo for all the amazing moments on the boat and around, and Bernd Illing for not kIlling me when we shared an office.

Also, all the people that I can't see enough anymore but to whom I am still thankful for all the great times Loïc Veyssière, Louise Naudin, Nitya Duella, Guillaume Pernollet, Edoardo Remelli, Adrien Doerig, and Alban Bornex – randomly ordered.

Finally, I want to thank Elsa Lovat for all the support and for taking such good care of me through the difficult times of finishing this thesis. Your presence has been a constant source of strength, and I am truly grateful.

Lausanne, June 20, 2023 M. B-R.

Abstract

Humans and animals constantly adapt to their environment over the course of their life. This thesis seeks to integrate various timescales of adaptation, ranging from the adaptation of synaptic connections between spiking neurons (milliseconds), rapid behavioral adjustments in response to situations like unexpected encountering someone on the street (seconds), and the process of lifelong learning such as learning a language (years).

Our work comprises two primary narratives: the development of a bio-inspired Spiking Neural Network (SpikeSuM) for surprise-based learning, and the creation of an artificial continual learning model (GateON).

To examine the role of surprise in rapid adaptation to unforeseen events, we developed SpikeSuM. SpikeSuM is a biologically plausible model using three-factor Hebbian rules for fast adaptation to unexpected events. Our findings indicate that brain-like surprise can be extracted from the neuronal activity of a prediction-error-minimizing network, and that SpikeSuM achieves Bayesian performance in online adaptation tasks. However, rapid adaptation may be susceptible to unwanted memory loss, or "catastrophic forgetting". To address this issue, we developed a context-driven prediction circuit that enables SpikeSuM to not only learn quickly in new contexts, but also retain memories of previous ones.

Our research resulted in the formulation of key hypotheses for context detection in the brain. These include a Surprise-driven context boundary detection using dis-inhibitory networks, as well as the hypothesis of locally inhibited neuromodulation for preventing unwanted learning. The context-driven approach of SpikeSuM inspired the development of our GateON method. GateON is an artificial neural network method that emphasizes the importance of context-based prediction and the ability of neurons to retain information to solve continual learning problems. The method is designed to solve the limitations of SpikeSuM, namely the modular approach preventing generalization across contexts. GateON achieves state-of-the-art results on large-scale continual learning problems, including continual learning on 100 MNIST tasks and large language models. Furthermore, GateON's potential for refinement through local learning rules suggests a path toward a more biologically plausible implementation, making it a promising approach for future research.

By synthesizing these approaches and insights, our work aims to advance the neuroscience and machine learning fields and contribute to a deeper comprehension of brain mechanisms underlying surprise, context-based learning, and continual learning.

***keywords* —** Surprise, spiking network, fast adaptation, context, continual learning, Artificial network

Résumé

Les êtres humains et les animaux s'adaptent en permanence à leur environnement tout au long de leur vie. Cette thèse vise à intégrer diverses échelles temporelles d'adaptation, allant de la modification des connexions synaptiques entre les neurones à impulsion (millisecondes), les ajustements comportementaux rapides face à des situations imprévues – comme rencontrer quelqu'un dans la rue (secondes) – et le processus continu d'apprentissage, tel qu'apprendre une langue (années).

La thèse se compose de deux récits principaux : le premier se consacre au développement d'un réseau de neurones à impulsion bio-inspiré (SpikeSuM) dédié à un apprentissage basé sur la surprise, le second s'axe sur la création d'un modèle artificiel d'apprentissage continu (GateON).

Pour étudier le rôle de la surprise afin de permettre une rapide adaptation aux événements imprévus, nous avons élaboré SpikeSuM, un modèle biologiquement plausible utilisant des règles de Hebb à trois facteurs. Nos résultats montrent qu'un signal lié a la surprise, similaire à celui observé dans le cerveau, peut être extrait de l'activité neuronale d'un réseau minimisant les erreurs de prédiction. De plus, SpikeSuM atteint des performances Bayésiennes dans les tâches d'adaptation en ligne. Cependant, l'adaptation rapide peut induire une perte de mémoire non souhaitée, ou "oubli catastrophique". Pour résoudre ce problème, nous avons développé un réseau de neurones adaptant ses prédictions selon le contexte qui permet à SpikeSuM d'apprendre rapidement dans de nouvelles tâches, tout en conservant les souvenirs des précédentes.

Nos recherches ont conduit à la formulation de diverses hypothèses clés relevant de la détection de contexte dans le cerveau. Ces dernières incluent la détection de contexte basée sur la surprise par le biais de réseaux désinhibiteurs, ainsi qu'une neuromodulation localement inhibée pour prévenir d'apprentissages indésirables. Au-delà de ces hypothèses, l'approche utilisée dans SpikeSuM nous a inspiré pour le développement de la méthode GateON.

GateON est une méthode de réseau neuronal artificiel qui met l'accent sur l'importance de la prédiction basée sur le contexte ainsi que la capacité des neurones à conserver des informations. Cependant, GateON vise à rester aussi proche que possible d'une compatibilité biologique. Contrairement à SpikeSuM, qui utilise une approche modulaire limitant la généralisation entre les tâches, GateON est conçu pour surmonter cette limitation. Cette nouvelle architecture nous a permis d'obtenir des résultats de pointe dans des problèmes d'apprentissage continus à grande échelle, tel que sur l'apprentissage continu de 100 tâches MNIST ainsi que sur de grands modèles de langage. De plus, l'inspiration biologique initiale de GateON suggère une capacité d'adaptation de la méthode qui permettrait une implémentation bio-plausible. Cette observation en fait une approche prometteuse pour de futures recherches en neuroscience.

En synthétisant ces approches et ces idées, notre travail a pour objectif faire progresser les domaines de l'adaptation rapide et de l'apprentissage continu. En outre, notre recherche vise à contribuer à une compréhension plus approfondie des mécanismes cérébraux sous-jacents à l'apprentissage basé sur la surprise et le contexte. Cette thèse met en avant l'intégration des informations sur diverses échelles temporelles, allant des millisecondes aux années, et fournit une base pour les recherches futures et les applications pratiques en neurosciences.

---

mots clés — Surprise, réseau à impulsion, adaptation rapide, contexte, apprentissage continue, réseau artificiel

Contents

Acknowledgements **i**

1 Introduction **1**

2 SpikeSuM: A spiking neural network for surprise-based learning **11**

2.1 Introduction 12
2.2 Results 13
2.2.1 Building expectations in a sequence task with rule switching 13
2.2.2 A mismatch of excitation and inhibition yields an intrinsically generated surprise signal 15
2.2.3 Modulation of plasticity by surprise supports rapid re-adaptation 18
2.2.4 Continual learning across rule switches is supported by the surprise signal 20
2.2.5 A modular network architecture avoids the stability-plasticity dilemma . 22
2.3 Discussion 23

Appendices **29**

2.A Material and methods 30
2.A.1 Two volatile sequence tasks 30
2.A.2 Spike trains of sensory neurons 31
2.A.3 Transmission from sensory neurons to prediction error neurons 31
2.A.4 Spiking neuron model 32
2.A.5 Two connectivity patterns onto the prediction error layer: random and regular 33
2.A.6 SpikeSuM Network architecture 33
2.A.7 SpikeSuM learning rule: Derivation of Hebbian factors 34
2.A.8 SpikeSuM learning rule: third factor 35
2.A.9 After convergence, synaptic weights of SpikeSuM reflect transition probabilities 36
2.A.10 Predicted next stimuli with learning rate modulated by surprise 42
2.A.11 Population activity in prediction error layer represents the present state and consistent alternatives to the present state, given the previous state 42
2.A.12 One-shot learning after a rule switch 43
2.A.13 Benchmark algorithms 43
2.A.14 Simulation parameters and comparison of algorithms 44

2.A.15 Context Modules: architecture of SpikeSuM-C 44
2.A.16 Systematic results for SpikeSuM-C 47
2.A.17 SpikeSuM-C parameters 47
2.B Supplementary figures 50

3 GateON: an unsupervised method for large scale continual learning 57
3.1 Introduction 58
3.2 The GateON Model 60
3.3 Simulations 63
3.3.1 Classical continual learning 64
3.3.2 Large-scale Continual learning. 64
3.3.3 Continual learning with large language models 66
3.4 Discussion 69

Appendices 71
3.A Methods 72
3.A.1 Relevance mapping 72
3.A.2 Availability forgetting time 72
3.A.3 Task inference 73
3.A.4 Intuition behind the GateON method 74
3.A.5 Alternative relevance mappings approximations for n-GateON 77
3.A.6 n-GateON for CNN 79
3.A.7 Metrics for Continual learning 79
3.A.8 Network Generalization and neuron sharing 80
3.A.9 Task-locked accuracy 80
3.A.10 Context correlation matrix 82
3.A.11 Network architectures 82
3.B Supplementary tables 84

4 Discussion & Conclusions 89

Bibliography 111

Curriculum Vitae 113

List of Figures

1.1 Ramón y Cajal pyramidal neurons 3
1.2 Illustration of the 2008 market crisis through the lens of surprise theory 5
1.3 experimental results of surprise in sequences 6
1.4 Introduction to continual learning 9

2.1 Volatile sequence task generation 15
2.2 Extracting surprise in SpikeSuM 17
2.3 Fast-adaptation results for SpikeSuM 19
2.4 Simulated compared to behavioral surprise 20
2.5 Continual learning across rule switches 26
2.6 Extinction of synaptic plasticity 27
2.A.1 Spike response model 32
2.A.2 Context selector module 46
2.B.1 Deterministic transitions have a different signature than stochastic ones 51
2.B.2 Population response profile before and after switch-point 52

3.1 GateON schematic 62
3.2 Leaky availability mapping and network saturation 65
3.A.1 Context Detection in MNIST 75
3.A.2 Speed of decay in network availability for different approximation of relevance mapping 78
3.A.3 Neuron sharing in GateON 81
3.A.4 Task-locked accuracy in GateON 82

List of Tables

2.A.1 Summary parameters SpikeSuM for fast adaptation 44
2.A.2 SpikeSuM-C task-switch detection accuracy 48
2.A.3 SpikeSuM-C parameters for 2 rooms stochasticity 48
2.A.4 SpikeSuM-C parameters for 4 rooms stochasticity 49

3.1 Results MNIST 10 64
3.2 Results MNIST 100 67
3.3 Results for large language model 68
3.4 Forward transfer in large language model 69
3.5 Backward transfer in large language model 69
3.A.1, GateON parameters for MNIST 84
3.A.2, GateON parameters for NLP 84
3.B.1 results MNIST 100, task fed 84
3.B.2 results MNIST 100, task fed, with convolutions 84
3.B.3 results MNIST 100, task inferred 85
3.B.4 results MNIST 100, task inferred, with convolutions 85

1 Introduction

In the past two decades, artificial intelligence (AI) and machine learning (ML) have become increasingly popular topics, paving the way for numerous scientific breakthroughs. While these cutting-edge technologies, from the classic feed-forward neural network[1] to transformers[2], have received much attention, it is noteworthy that the human brain has and still serves as the primary source of inspiration[3–5].

Although the first studies on the human brain trace back to Ancient Greece, modern neuroscience, particularly computational neuroscience, emerged in the mid to late 20th century with the development of new measurement devices, such as calcium imaging[6], fMRI[7], two-photon imaging[8], etc. Today, neuroscience is a growing interdisciplinary field that involves professionals ranging from medical doctors and biologists to physicists and computer scientists. This thesis takes place at the interface of theoretical neuroscience and applied machine learning.

In biological neural networks, like in their artificial counterparts, even the smallest changes can have a significant impact on the overall system. The way neurons communicate (information transmitted through spikes[9]) impacts the way the synapses between two neurons are strengthened (synaptic update rule[10, 11]). The way the synapses are strengthened impacts the way the neurons organize (neural network). The way the neurons organize impacts the cortical column structure (from the input to output layers), all of these impacting human behavior. The first chapter focuses on the impact that spikes can have on global signals in the brain influencing learning, called neuromodulatory signals.

Since the 2000s, it has become evident that neuromodulatory signals play a crucial role in influencing human and animal behavior, memory, and responses, among other aspects[12–16]. One such signal is the Surprise. Over recent decades, extensive research has been conducted on the detection and modeling of surprise, primarily in the fields of behavioral studies and information theory[17, 18]. In recent years, however, there has been a growing interest in exploring the mechanistic underpinnings of these signals.

We propose to investigate 3 different time scales of learning and show their interdependency. First, the use of spike signals, with a duration of approximately 1 millisecond, for the purpose of detecting surprise signals over a longer integration time period of around 200–600 milliseconds[19]. Second, we explore how these surprise signals can be employed to detect contextual changes that can occur in the environment over varying time spans ranging from seconds to hours. Third, we highlight the potential of context-driven models as a means of facilitating lifelong learning, also called continual learning. By utilizing these models, individuals can continually adapt and improve their behavior as they encounter novel situations and challenges.

The following sections introduce the general topics of our work. First, we delve into the origins and properties of theoretical **bio-plausible neural networks**. Second, we describe the concept of **surprise in information theory** to gain a better understanding of its significance for learning. Furthermore, we elucidate how surprise can be linked to **context boundary detection** to prevent memory overwriting, also known as catastrophic forgetting. Finally, we explore the **concept of continual learning** in machine learning to illustrate how our framework of surprise and context detection holds the potential to facilitate human-like learning.

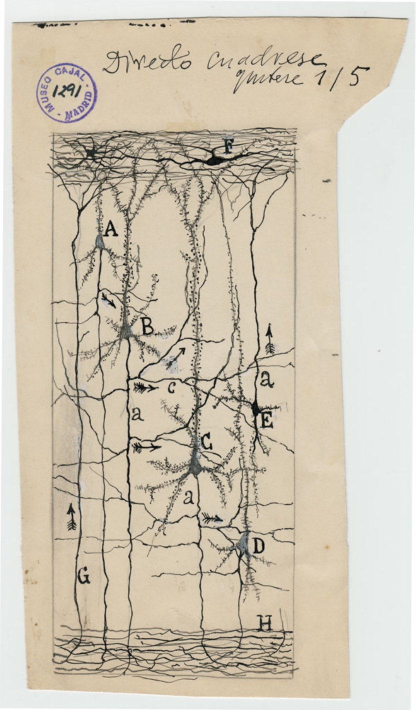
Biological model of neural networks

The cerebral cortex is a complex network of neurons and synapses (Figure 1.1) that enable the brain to process information and produce behavior.

Neurons play a dual role in the brain, acting as both information receivers and transmitters. Although there is a multitude of neuronal subtypes, they can be generally categorized based on their structural or functional attributes. Structural classifications include pyramidal, unipolar, bipolar, and multipolar. While it is easy to define functional distinctions of neurons in the spinal cords comprising sensory, motor, and interneurons[21], it is more complicated for cortical neurons. One main classification of brain neurons is based on their neurotransmitter type (Glutamate^{t}[22], GABA^{t}[23], ...). For most computational neuroscientists, this practically means a model composed of Excitatory (Glutamate receptors) and inhibitory (GABA receptors) neurons only. Excitatory neurons increase the electrical potential of adjacent neurons, while inhibitory neurons reduce it.

The synapses are the channels of communication, allowing the information to flow from the presynaptic neuron to the postsynaptic one. Together, these neurons and synapses form a neural network that is capable of producing macroscopic behavior.

Figure 1.1: First reproduction of pyramidal neurons in the cerebral cortex, observed by Santiago Ramón y Cajal in 1888[20]



The properties of biological neural networks differ significantly from those of artificial neural networks introduced earlier. In particular, artificial neural networks do not follow the three key biological constraints that we focus on: neurons emit information through spikes, the evolution of a synapse is driven only by the two neurons it connects (locality), and neurons can only be either excitatory or inhibitory, not both (Dale's law).

The limitations imposed on bio-inspired neural networks can significantly hinder their capabilities when compared to artificial neural networks. One of the earliest models of synaptic evolution was introduced by Donald Hebb in his 1949 publication, "The Organization of Behavior"[24]. Hebb proposed that changes in synaptic strength (w_{ik}) are only dependent on the activities of the pre-and post-synaptic neurons (i.e., neuron i and neuron k). This led to the weight update:

$$\Delta w_{ik} = f(\text{pre}_k, \text{post}_i, w_{ik}) \tag{1.1}$$

Recent research suggests that neuromodulators such as dopamine, serotonin, etc. act as global modulatory (not neuron-specific) signals for synaptic evolution. In particular, a generalization of Hebbian rules also known as the three-factor learning rule, suggests that the variation in synaptic strength between pre-neuron k and post-neuron i is influenced by these neuromodulators (also called 3^{rd} factor):

$$\Delta w_{ik} = f(\text{pre}_k, \text{post}_i, w_{ik}, 3^{rd}) \tag{1.2}$$

The first experimental evidence of neuromodulated learning in the brain was highlighted in reward-based experiments, where dopamine was found to be the crucial third factor[25]. Ever since, the theory of three-factor learning rules has allowed for many breakthroughs in neuroscience[26] and further research has indicated that brain signals, such as reward, surprise, and novelty, stimulate the release of neuromodulators, influencing learning and behavior in animals. In this thesis, we formulate the hypothesis that the surprise signal should act as the introduced third factor.

Surprise: from information theory to neural implementation

In the field of information theory, the notion of surprise emerges when there is a contrast between a prior (or prior prediction), which reflects an individual's internal model of the world, and an actual observation in the external world. The extent to which an observation is deemed surprising is contingent on an individual's expectations. As an illustration, the 2008 financial crisis may have been astounding for someone who firmly believed in the stability of the market (low-risk prior), whereas an anti-capitalist activist who predicted the collapse of the system (high-risk prior) might not have been surprised (see Figure 1.2).

As nicely summarized in Modirshanechi et al.[27], surprise can be quantified using various methods, including Bayesian surprise[28], which is calculated based on the magnitude of update in predictions. Shannon Surprise[29], which is based on the logarithm of the probability of the internal model given an observation, and the direct distance between predicted and observed outcomes. The feeling of surprise in neuroscience has been extensively studied in terms of behavior, including observable changes such as startle and pupil dilation. However, the underlying mechanisms of surprise remain poorly understood.

One of our goals was to investigate the origins of surprise and its effects on human learning. We focus on neutral surprise, not related to reward or any other feelings. To exemplify our approach and later test our model, we designed a real-life action-free task for surprise detection tested on 100 subjects in Fig.1.3.

Prior to observation, looking for surprises allows for maximizing information gain. Such reinforcement algorithms are called surprise-seeking. Posterior to observation, surprise-based updates allow for fast adaptation of the model towards the new data. Fast adaptation allows for preventing negative consequences. Minimizing surprise is critical for reliable

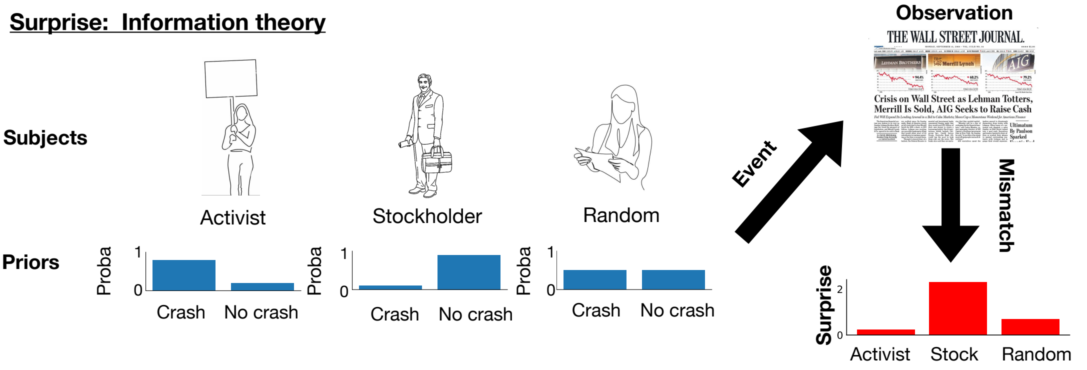


Figure 1.2: **Illustration of the 2008 market crisis through the lens of surprise theory**. Each individual constructs their own world model, or prior belief, based on previous life experience. In the example of the stock market: the anti-capitalist activist anticipates a high risk of a crash ($P(\text{crash}) = 0.8$), the stockholder has a low expectation of a crash ($P(\text{crash}) = 0.1$), and the random citizen has no particular belief about the likelihood of a crash ($P(\text{crash}) = 0.5$). The external observation is the actual market crash in 2008. Surprise is the result of the discrepancy between the prior belief and the observation.

prediction-making, as demonstrated by the financial crisis example, and is also vital for human development activities such as food acquisition and danger avoidance. Additionally, multiple studies have demonstrated the impact of surprise on learning speed[17, 18, 30, 31]. Therefore, we propose that surprise may modulate synaptic plasticity through a global third factor.

Our mechanistic approach began with the observation that surprise is a highly energy-consuming experience, showing also an increase in brain activity. From this, we hypothesized that minimizing surprise would be a primary goal in our model. We focus on action-free tasks to avoid any interfering properties. Chapter 1 provides a detailed account of the development of SpikeSuM, a fast-adaptive spiking neural network, building on the insights gained from these earlier investigations.

Context boundary detection: another property of the surprise signal

In the field of surprise-based learning, it is widely accepted that surprise serves as a crucial mechanism to identify significant changes in the environment's structure, commonly known as switch points, which in turn facilitates swift adaptation[18, 32]. We focus on delta rule updates such that for a given prediction $\hat{y}$, an observation y, and a surprise factor S the model update is as follows:

$$\Delta\hat{y} = S(\hat{y} - y) \tag{1.3}$$

In essence, the model is barely updated if S is small, while a larger value of S results in a stronger update toward the observation. Regrettably, the use of Eq. (1.3) entails overwriting

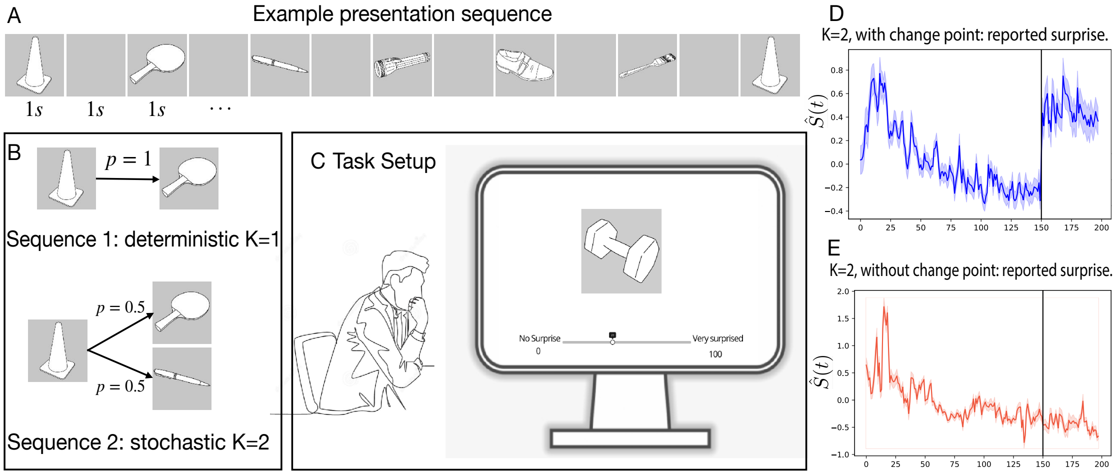


Figure 1.3: **Change points are linked to the subjective feeling of surprise.** **A** Example of an image sequence shown to participants. Each image is presented for $1s$ followed by a $1s$ gray screen. **B** Sequence 1 is deterministic and used to familiarize the subject with the task. Sequence 2 has stochastic transitions so that each given image can be followed by one of 2 other images, with equal probability. **C** Experimental setup: Participants sit in front of a screen and observe the image sequence. Participants are informed to predict the image following the shoe and asked to report their feeling of surprise continuously throughout the task by moving a 'Surprise slider' horizontally. Participants are randomly assigned to two different groups. **D** Reported surprise $\hat{S}$ (dark blue) as a function of time (image presentation number) for the stochastic sequence, averaged over all 65 participants in the first group. Immediately after the change point at step 150 (vertical bar) the reported surprise increases. Before averaging, the reported surprise $\hat{S}$ (dark blue) is, for each participant, normalized to mean zero and unit variance before the switch-point. Light blue: standard deviation across participants. **E**. 25 other participants in the second group see a different sequence without change-point. For time steps >150, differences between D and E are significant (see Fig. 2.4). *Data collected on the platform 'prolific', courtesy of Michael Herzog and Wei-Hsiang Lin, EPFL.*

the previous model with each new switch-point, which can result in an unwanted overwrite called catastrophic forgetting.

To address this challenge, our proposed solution involves incorporating context-dependency into the model learning process. Specifically, we suggest introducing the concept of the probability of a particular context C, represented as P_C, alongside a context-dependent model $\hat{y}_c$. By making these adjustments, we can establish an updated rule that accounts for context, which can be expressed as:

$$\Delta \hat{y}_c = P_c S(\hat{y}_c - y) \tag{1.4}$$

The objective of the second part of Chapter 1 is to incorporate a rule into a spiking neural network, with the goal of enabling rapid adaptation without disrupting prior knowledge. It is worth highlighting that this model is contextual, which allows for efficient adaptation without overwriting previous information.

Continual learning and context-gated neural networks

Continual learning represents a critical challenge in the field of machine learning. Despite widespread interest, defining continual learning remains an elusive task, with divergent interpretations among experts. This introduction aims to establish a comprehensive definition of continual learning, with the ultimate objective of clarifying the underlying objectives of this rapidly-evolving field. Specifically, continual learning may be understood as the capacity of an autonomous agent, such as a human, robot, or artificial intelligence system, to **learn and integrate** an increasing number of **complex functions** within a continuously **changing environment**. While this definition provides a broad framework, the precise goals of continual learning remain subject to ongoing refinement and exploration.

The primary objective unanimously agreed upon in the field is to prevent catastrophic forgetting. It is imperative that an agent, once it has learned a function, task, or language, should not forget it when presented with a novel environment or trained on a new function. However, if we limit our scope to this objective, the most optimal approach would be to train n agents on n tasks and consider the task complete.

In comparison to training n independently functioning agents, what benefits can be reaped from continual learning? Timo Flesh et al.[33] defined continual learning as "*[...] learn[ing] to perform multiple distinct tasks over their lifetimes while avoiding mutual interference among them*". Even though not intended that way, this definition might overlook the primary advantage of continual learning, i.e., positive mutual interference. Continual learning aims to generalize the learned knowledge to be applicable to novel functions, also called forward transfer. Forward transfer allows for the attainment of favorable solutions even with limited available data, which is typically the case in real-life scenarios or datasets. Pre-trained models fine-tuned for specific tasks were initially thought to be the solution to this problem, but they are highly susceptible to catastrophic forgetting. Backward transfer complements forward transfer, and allows agents to enhance previously learned functions when exposed to new tasks.

So far, our discussion on the learning process has not yet incorporated the formal introduction of the terms "functions" and "environment" employed in our definition. In the context of machine learning, functions refer to the objectives of the agent, such as classification, reward optimization, regression, translation, etc. It is important to note that the same dataset can be used to accomplish different functions, with the agent exhibiting different behavior. Mathematically, we can represent this as:

$$y = f(x) \neq f'(x) = y' \tag{1.5}$$

Where f and f' are two distinct functions of the agent, yielding different outputs despite receiving the same input x. For instance, when applied to a dataset composed of images of integers (such as the MNIST dataset), an agent can be trained to perform number classification (each number has to be classified from 0 to 9), or even numbers versus odd numbers classification.

Now, the environment can be defined mathematically as the dataset or data distribution from which the inputs x and x' are drawn, where $\mathscr{D}$ and $\mathscr{D}'$ are two distinct environments if $\mathscr{D} \neq \mathscr{D}'$. The challenge faced by continual learners lies in distinguishing between two environments that may appear very similar. For instance, when confronted with two mazes that possess identical room arrangements but differ in only two rooms, one may opt to learn two separate models for each maze, or instead, a single model except for the specific rooms in question. More complex problems arise with time-dependent environments $\mathscr{D}(t)$ that slowly shift without distinct boundaries. The English lexicon, for instance, is continually evolving, yet there exists no objective way of defining the boundaries that could separate old from new English. To circumvent this issue, we focus solely on environments with sharp changes that can be differentiated according to a distance measure d. Specifically, we consider two environments $\mathscr{D}$ and $\mathscr{D}'$ to be distinct if their distance $d(\mathscr{D},\mathscr{D}')$ is greater than a pre-defined threshold value θ, denoting the minimum distance required to classify two environments as separate.

In essence, continual learning would be a valuable tool for dealing with partially observable datasets that are observed sequentially, with limited or no repetitions. This approach enables the training of multi-functional agents with minimal examples. By mitigating catastrophic forgetting, continual learning facilitates model fine-tuning and upgrading without requiring a complete retraining process from scratch. This feature is particularly beneficial given the ever-increasing size and training time of new AI technologies, which can have a significant ecological impact[34, 35].

We previously discussed the use of surprise signals to facilitate learning and identify shifts in environmental structures. Additionally, we explored the concept of contextual predictions, which enables adaptation to novel environments without erasing prior knowledge. However, life-long learning (or continual learning as referred to within the machine learning community) encompasses more than just preventing memory loss. A comprehensive algorithm should be able to persistently train with minimal forgetting, transfer knowledge to new contexts (forward transfer), and refine previous models with the incorporation of new pertinent information (backward transfer). In comparison to machines, humans excel at operating within such a paradigm, as the natural inputs we encounter are predominantly sequential. We summarize Multi-task learning (access to all data at all times) versus continual learning in Fig 1.4.

In the longstanding interplay between neuroscience and machine learning, we chose to develop a machine learning implementation of a continual learning algorithm, drawing inspiration from neuroscience, called GateON. Building upon the idea of context-based prediction modules from SpikeSuM, we aimed to create non-independent models using multiplicative gating of activity—a concept inspired by observations of dendritic inputs in the brain[36, 37].

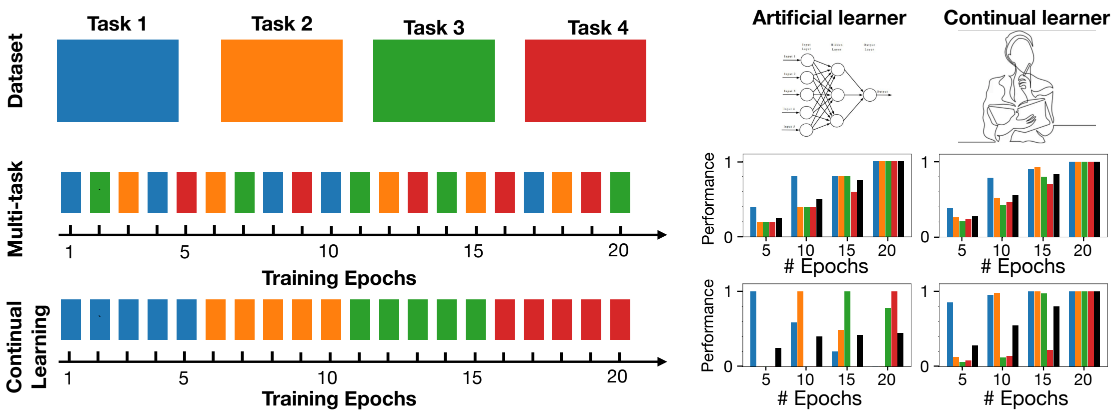


Figure 1.4: **Schematic of Continual Learning with 4 tasks**. **Left**: The datasets comprise the data from which the agent learns. For example, task 1 could be learning French, and the dataset composed of French books, task 2 Italian, and so on. Multi-Task randomly draws examples (books) from any task at each epoch, while continually learning samples from the same task for a certain time (here 5 epochs) before going to the next one without revisiting earlier tasks. **Right**: Results comparison between artificial and continual learners. The results presented are synthetic and for comprehension purposes. The bars' colors indicate test performance for corresponding tasks, with black bars representing average performance across tasks. In multitask learning, both artificial learners excel, achieving a performance of 1 for all tasks at the end of training. Conversely, in the continual learning paradigm, artificial learners struggle to maintain performance for earlier tasks when exposed to new tasks, while continual learners enhance performance over time. This includes forward transfer (increased performance for unobserved tasks) and backward transfer (improved performance for previously observed tasks). In our example, knowing French helps a lot to learn Italian (forward transfer), and learning Italian can improve already known French (Backward transfer), see bottom right histograms.

Additionally, we considered an unsupervised, online algorithm, which is typically favored for biologically plausible models, for determining neuron importance. This approach aids in fixing representations to prevent catastrophic forgetting. In chapter 2, we examine the GateON method in detail, exploring its ability to continuously learn new tasks while preserving existing knowledge.

2 SpikeSuM: A spiking neural network for surprise-based learning

Fast Adaptation to Rule Switching using Neuronal Surprise

Authors: *Martin Barry, Wulfram Gerstner*
Contribution: All theories and implementations have been made by Martin Barry. The writing has been shared across authors.

2.1 Introduction

An event is surprising if it does not match our expectations [27, 38–40]. The unexpected punchline of a joke [40], the unexpected continuation of a sequence of tones [41], harmonies [42, 43] or images [44–46], as well as rule switching such as shift of escape platform in the Morris water maze [47] or meaning of cues [48–51] induce measurable physiological and behavioral reactions in humans and animals. Without expectations arising from previous experiences, an event such as the observation of a new image may be perceived as 'novel' but cannot be 'surprising' [52, 53].

Surprise is a well-studied phenomenon in the neurosciences [38, 39] and has also been formally analyzed in the mathematical literature [27]. In the neurosciences, startle responses [54], delayed responses [39] and pupil dilation [55, 56] are measurable physiological manifestations in response to surprising events. Moreover, EEG, fMRI, MEG, and electrophysiological studies show an increase in brain activity shortly after a surprising event [38, 45, 57–61]. Apart from its potential role for intrinsic motivation [62], surprise plays a crucial role in learning: surprising events are more memorable [39, 63, 64] and allow quick adaptation to a changing environment [65, 66]. In this chapter, we study the role of surprise in building expectations, modulating learning, and detecting rule switches. Specifically, we focus on two aspects. First, surprising events significantly increase the speed of learning [17, 30, 31, 52], presumably by increasing synaptic plasticity. Second, surprise is involved in the creation and consolidation of memories [39, 67, 68], presumably including the memory of rules.

In contrast to the numerous mathematical studies that start from a normative framework of surprise [27, 32, 69–78], we take a constructive approach based on a network of spiking model neurons with plastic connections. We consider two aspects of spiking neural networks as crucial requirements for biological plausibility. First, all information about expected and observed events, and an occasional mismatch between the two, needs to be *communicated via spikes*; thus a comparison of subthreshold membrane potentials across different neurons – as required in some existing models [79–81] – is not possible. Second, synaptic plasticity rules should be expressed as *NeoHebbian three-factor learning rules* [82–87] where the changes of a synapse from neuron A to neuron B can only depend on the spikes of neuron A and the state of neuron B (the two 'local' factors') plus one (or several) neuromodulators that play the role of a global feedback signal (third factor) broadcasted to large groups of neurons; in our approach, a detailed synapse-specific feedback as used in the BackProp algorithm [88] and variants thereof [89–92] is not needed.

Our main hypothesis is that surprise manifests itself in a spiking neural network as a *mismatch between excitation and inhibition* in a layer of hidden neurons that represent the current observation and compare it to the expectation arising from earlier observations. Our approach is intimately linked to both the theory of excitation-inhibition balance (E-I balance) [93–95] and the theory of predictive coding [96–99].

Predictive coding is an influential theory in the fields of neurosciences [60, 96, 100–102] and bio-inspired artificial neural networks [103–106]. In contrast to the classic framework of predictive coding that emphasizes sparsity of activity as a means to minimize redundancy of codes [107], we emphasize the advantage of predictive codes for generating a surprise signal in spiking neural networks. Importantly, we propose in this chapter that *an intrinsic spike-based surprise signal can modulate biologically plausible synaptic plasticity rules to achieve fast adaptation and continual learning across rule switches.*

We focus on two related tasks, both involving sequences of observations. The first task illustrates the well-known problem of re-adaptation to abrupt switches in the stimulus statistics where the same rule of stimulus generation is unlikely to occur twice [31, 32, 75]; the second one exemplifies the problem of continual learning across rule switches where each rule should be memorized since it is likely to re-appear [47, 51, 108]. In both tasks, expectations ('predictions') must be built by self-supervised learning, and change points ('rule switches') must be inferred from the observation sequence since they are not indicated by a cue. Our model links observations in the neurosciences at the level of single neurons or circuits to psychological phenomena of surprise and provides an alternative to algorithmic approaches to the stability-plasticity dilemma [109, 110], continual learning [49, 108, 111, 112], context-dependent prediction [113–115], or context buffers in artificial neural networks [104].

2.2 Results

2.2.1 Building expectations in a sequence task with rule switching

Imagine the following sequence of numbers

$$\underbrace{1 \to 2 \to 3 \to 4 \to 1 \to 2 \to 3 \to 4}_{\text{rule } A} \to \underbrace{1 \to 3 \to 2 \to 4 \to 1 \to 3 \to 2 \to 4}_{\text{rule } B}\,. \tag{2.1}$$

The complete sequence is composed of **transitions** (e.g., $2 \to 3$) and **switches** between two different rules (rule $A \to$ rule B). We refer to the moment of rule switching as a change point. The "volatile sequence tasks" generalizes switches between elementary deterministic rules, as in (2.1), to more complex probabilistic sequences of images generated by the following procedure. We have a total of $\mathscr{R}$ images. After the presentation of image number i, the next image is one of K images that are possible as successors of i. For example, for $K = 2$ the possible successors after the image 'rubber cone' could be 'pen' or 'table tennis bat' **(See introductory Figure 1.3 A, B)**. When the rule of sequence generation changes, participants watching the sequence of images report the subjective feeling of surprise (**See introductory Figure 1.3 C, D)** consistent with earlier experiments [116]. Thus, in such a framework, a change point triggers a surprise signal.

To generalize the above procedure to the case of $K = 4$ possible successor images, we may think of a video taken in an empty apartment of $\mathscr{R}$ square rooms, each room recognizable by a specific wallpaper. The video camera takes one static view of a room before it is moved to one of the K neighboring rooms (Fig. 2.1 **A**). In total, $\mathscr{R}x\mathscr{R}$ transitions would be possible, but because of the specific layout of the apartment, not all of these are observed. An observer watching the recorded sequence would see transitions of images ('rooms') 1 → 2 with probability $T^*_{2,1}$ or 1 → 3 with probability $T^*_{3,1}$, etc. The hidden 'rule' of sequence generation arises from the transition matrix $T^*_{i,j}$ (Fig. 2.1 **B**). However, at unknown moments in time the rule changes (with switch probability $\mathscr{H}$, called 'volatility'), akin to the switch from rule $A \rightarrow$ rule B in (2.1). Note that the set of images remains the same after a change point, while the transition matrix changes [72, 79]. The above probabilistic task with rule switching is a generalization of established tasks in the cognitive neuroscience of surprise [31, 116–118].

As opposed to an agent that selects actions to collect information, our observer is passively watching the image sequence. From the observed transitions between images, the observer learns which image (or images) to expect given the current one, i.e., estimate transition probabilities $T_{i,j}$. This passive mode is ideal for a study of surprise because, in the context of neuroscience, it avoids any confounding factors arising from action selection [119] or reward [12] and, in the context of reinforcement learning theory, it avoids any complex interaction with models of curiosity, action selection policy or questions of model-based versus model-free reinforcement learning [120, 121] - simply because our observer does not choose actions. Once the set of possible transitions under a given rule has been learned, this knowledge could, of course, be used in model-based reinforcement learning, but this is not part of the tasks that we consider (see **Supplementary Discussion**).

A typical sequence of rule switches is shown in Fig. 2.1 **C** where different rules correspond to different transition matrices $T^*_{i,j}$. Inspired by experimental observations for passive learning in humans and animals [46, 60, 118, 122, 123], we assume that the (potentially unconscious) goal of observers is to predict possible next observations, i.e., estimate transition probabilities $T_{i,j}$ that are as close as possible to the real probabilities $T^*_{i,j}$. Our spiking neural network model (introduced in the next paragraph) implicitly encodes expectations about possible next stimuli in the set of synaptic weights. From this set of weights, we extract the expectations at time t in the form of a learned transition matrix $T_{i,j}(t)$ that can be compared to the currently active rule $T^*_{i,j}$ (Fig. 2.1 **B**). The expectations built during exposure to the sequence are a prerequisite to extracting a surprise signal.

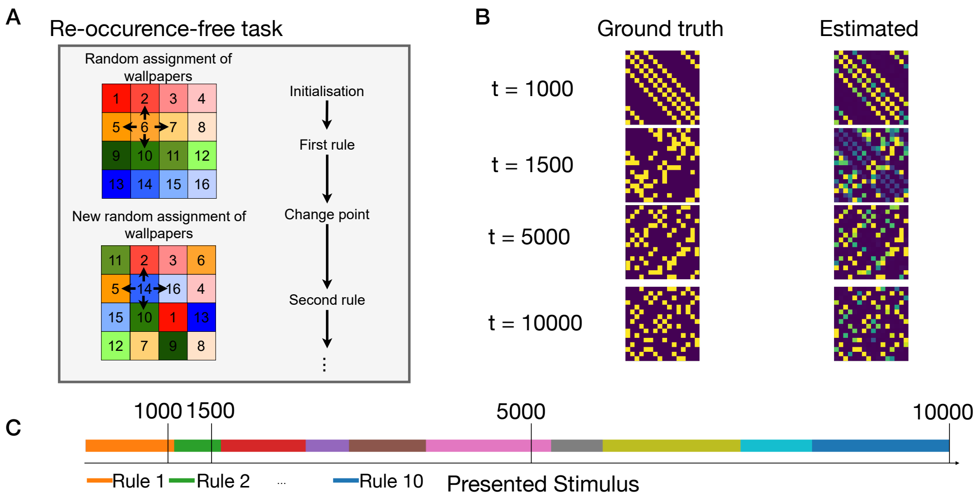


Figure 2.1: **Expected transitions in a volatile sequence task. A.** At each presentation step, the stimulation is caused by the wallpaper image (indicated by different colors) in one of the rooms of an apartment with $\mathscr{R}$ rooms (here $\mathscr{R} = 16$). The image sequence reflects transitions (arrows) from the current room to one of the K neighboring rooms (here $K = 4$). On rare occasions (change points), the transition rule is changed by a new random assignment of wallpapers to rooms. The same rule is unlikely to return. **B.** The ground truth transition matrix for different rules (left), compared to the transition matrix estimated by the model (right) at different time points of a simulation run. **C.** Switching of rules over time in the simulation of B. Each rule (Rule 1, Rule 2, ...) only appears once. Vertical lines indicate the time points in B.

2.2.2 A mismatch of excitation and inhibition yields an intrinsically generated surprise signal

The **Spike**-based **Su**rprise-**M**odulated (SpikeSuM$_{rand}$) network model (Fig. 2.2 **A**) consists of an input layer with random projections onto excitatory and inhibitory neurons in a prediction error layer, and a deep nucleus (i.e., a cluster of neurons in the central nervous system located below cortex [100, 124]), e.g., the locus Coeruleus [59], the ventral tegmental area [125] or higher-order thalamus [126, 127]. Neurons in the prediction error layer receive spikes from a first pool of N neurons encoding the currently observed stimulus and from another pool of N neurons in a memory buffer that encodes information on the previously observed stimulus (**Material and methods**). Synapses onto excitatory and inhibitory neurons have different weights. Two populations of pyramidal neurons P_1 and P_2, putatively located in cortical layers 2/3 [60], compare the weighted inputs of the current observation with the weighted inputs arising via connections from the memory buffer that we interpret as 'predictions'. Population P_1 is inhibited by the current observation and excited by the prediction coming from the buffer, whereas population P_2 is excited by the current observation and inhibited by the prediction. Both populations project to a group of pyramidal tract (PT) neurons, putatively located in layer 5b [87, 128], which output a low-pass filtered version $\bar{A}$ of the summed neuronal activity. Since $\bar{A}$ reflects the combined outputs of populations P_1 and P_2, the output of PT neurons can be interpreted as a symmetric measure of 'distance' between prediction and observation (**Material and methods**). If a prediction is correct, excitation and inhibition balance each other so that the total activity $\bar{A}$ of all pyramidal neurons is close to zero.

In our model, the PT-neurons send the filtered network activity information $\bar{A}$ to an unspecified nucleus (Fig. 2.2**A**) which sends back a neuromodulatory signal $3^{rd}(\bar{A})$ that is broadcast across the prediction error layer. We have checked that a large activity $\bar{A}$, caused by positive or negative prediction errors [57, 60, 122, 129, 130], indicates an unexpected transition. A transition is *unexpected* ('surprising') if the network has for example learned that after image '6', the next possible images are 2,5,7 or 10 (Fig 2.1 **A**), but the observed input corresponds to image '3', indicating that a switch point has occurred. Indeed, we find that the amplitude $3^{rd}(\bar{A}(t))$ of the 3rd factor increases after a switch of rules (Fig. 2.2 **A**, inset). We therefore interpret $3^{rd}(\bar{A})$ as a 'surprise signal'. Note that the surprise signal is a function of activity in the prediction error layer—and therefore implicitly a function of the mismatch between excitation and inhibition.

To achieve E-I balance for *expected* transitions, we assume that activated *excitatory* synapses onto neurons in population P_1 change according to an anti-Hebbian three-factor plasticity rule, modulated by the surprise signal,

$$\Delta w = -3^{rd}(\bar{A})\, h^{\text{post}}\, \overline{\text{EPSC}}^{\text{pre}}, \tag{2.2}$$

where $\overline{\text{EPSC}}^{\text{pre}}$ is the filtered sequence of excitatory postsynaptic currents (EPSCs) caused by the presynaptic spike train and h^{post} is the input potential of the postsynaptic neuron (for details, see **Material and methods**). Analogously, we assume that activated *inhibitory* synapses onto neurons in population P_2 change according to a Hebbian three-factor rule modulated by the surprise signal,

$$\Delta w = 3^{rd}(\bar{A})\, h^{\text{post}}\, \overline{\text{IPSC}}^{\text{pre}} \tag{2.3}$$

where $\overline{\text{IPSC}}^{\text{pre}}$ is the filtered sequence of inhibitory postsynaptic currents (IPSCs). For convergence properties of the two learning rules, see **Material and methods**.

Earlier theories have established that both Hebbian learning of inhibitory synapses [95] and anti-Hebbian learning of excitatory synapses [131] lead, for predictable inputs, to a stabilization of the firing rate of postsynaptic neurons at a low value. To check whether this holds also true for the above three-factor rules, we focus on a long stimulation sequence of 3000 presentation steps containing a single switch between two equal-length sequences generated each with a fixed but different rule. Consistent with earlier Hebbian theories, we observe that the SpikeSuM$_{rand}$ network converges after about 500 presentation steps to a stationary state of low activity (Fig 2.2 **D1**). Moreover, the switch between rules causes a sharp peak in the activity $\bar{A}$ (Fig 2.2 **D1**, inset). Thus, the activity $\bar{A}$ of PT-neurons can indeed be used to extract a surprise signal that is large for *unexpected* observations.

The predictability of the next stimulus is higher in a volatile sequence task with $K = 2$ possible

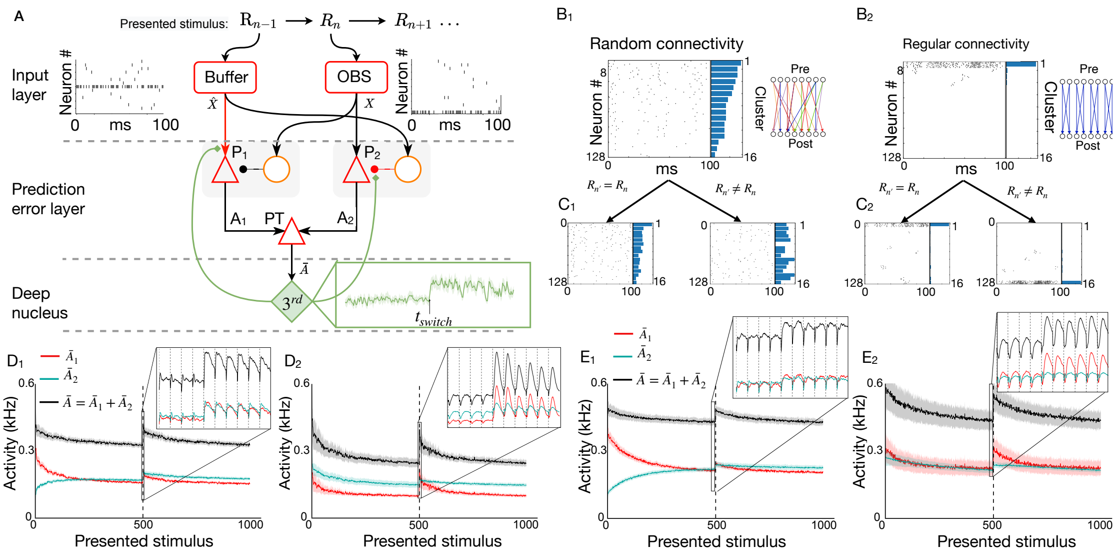


Figure 2.2: **Neurons in the prediction error layer exhibit high activity for unexpected transitions. A.** Spiking network model (SpikeSuM). From top to bottom: Every 100ms, stimuli change gives rise to a sequence $R_{n-1}, R_n, R_{n+1}\ldots$ representing images of the volatile sequence task. The presently observed stimulus (wallpaper image of room R_n, red box 'OBS') and the previous stimulus (R_{n-1}, 'Buffer') are encoded with spike trains of 128 neurons each (16 sample spike trains shown). These spike trains are transmitted to an antisymmetric excitation-inhibition network (prediction error layer) composed of pyramidal neurons (red triangles) and inhibitory neurons (orange circles). Pyramidal neurons in population P_1 are excited (arrowheads) by the inputs representing the prediction $\hat{X}$ from the previous presentation step and inhibited (round heads) by the current observation X whereas neurons in P_2 are inhibited by the prediction $\hat{X}$ and excited by the current observation X. The activity A_1 and A_2 of populations P_1 and P_2 is transmitted to a population of pyramidal tract neurons (PT), which conveys a low-pass filtered global activity $\tau\dot{\bar{A}} = -\bar{A} + A_1 + A_2$ to a group of neuromodulatory neurons in a deep nucleus (green, labeled 3^{rd}) which send a neuromodulatory surprise signal back to the prediction error layer. Poorly predicted stimuli increase activity in the prediction error layer and indirectly accelerate, via the 3^{rd} factor, learning in the plastic connections (red lines). Inset: Time course of the 3^{rd} factor (green) over 4s before and after a rule switch at time t_{switch}. **B:** Spike trains of neurons in P_2 during a specific stimulus R_n representing one of the $\mathscr{R} = 16$ images. For visualization, the neurons have first been ordered from highest to lowest firing rate and then clustered into groups of 8 neurons, with neurons 1 to 8 forming the first cluster, 9 to 16 second etc. The histogram of the average firing rate per cluster is shown on the right. **B**$_1$: Random sparse connectivity from presynaptic neurons in the input layer to neurons in the prediction error layer. Inset: schematics, continuous weights with colors indicating strength from red (weak) to blue (strong). **B**$_2$: Connectivity with binary connections. Inset: schematics, nonzero connections (blue) are organized in clusters of independent neurons. Random connectivity is potentially closer to biology, but regular connectivity has algorithmic advantages. **C**$_1$ **and C**$_2$**:** To compare the two networks, we show the spikes generated in response to a new image $R_{n'}$ while keeping the same order of neurons. For random connectivity (C1) spike plots are different if $R_{n'} \neq R_n$ but similar if $R_{n'} = R_n$. The same holds for regular connectivity (C2), but the structure is more obvious to the human eye. **D**$_1$ **and D**$_2$**:** Filtered activity of pyramidal neurons in populations P_1 (red), P_2 (cyan), and the total filtered activity $\bar{A}$ (black) as a function of time-averaged over 100 different sequences with a change point (switch of rule) after 500 presentation steps, for random (D1) or regular (D2) connectivity. Both networks indicate a surprising transition (dashed vertical line) by increased activity. Insets show the activity before and after the rule switch. **E**$_1$ **and E**$_2$**:** Same as in D$_1$ and D$_2$, but for the case of $K = 4$ possible next images. Since predictions are less reliable, the activity $\bar{A}$ converges to higher levels.

transitions from a given observation (Fig 2.2 **D1**) than in a task with $K = 4$ (Fig 2.2 **E1**). Hence, the next stimulus becomes 'more expected', the prediction error is lower, and the population activity converges to a significantly lower value in Fig 2.2 **D1** than in **E1**; mean activity averaged over the last 100 presentation steps is 375Hz in Fig. 2.2 D1 versus 461Hz in Fig. 2.2 E1 ($p < 10^{-10}$). This observation leads to experimentally testable predictions (**Supplementary Fig. S2.B.1**).

We consider two different architectures for the connectivity from the input spike trains to the pools P_1 and P_2 of pyramidal neurons. The first one, SpikeSuM$_{rand}$ (Fig. 2.2 **B1**), uses sparsely connected random projection weights from the input layer to the prediction error layer, whereas the second one (SpikeSuM) has a simplified connectivity matrix with hand-wired binary weights implementing a direct representation of input stimuli by non-overlapping subsets of pyramidal neurons in the prediction error layer (See **Material and methods**). We find that the qualitative features of the population activity in the simpler network (Fig. 2.2 **B2-E2**) are similar to those of the randomly connected network (Fig. 2.2 **B1-E1**). Since results are comparable for the two connectivity patterns, we focus in the following on SpikeSuM with the simple binary connectivity as a reference because it is faster to simulate. Moreover, the interpretation of activity patterns is easier with binary connectivity, since states that are predicted as potential observations given the previous stimulus are identifiable by simple visual inspection of the spiking activity (**Supplementary Figure S2.B.2**).

2.2.3 Modulation of plasticity by surprise supports rapid re-adaptation

To understand whether the modulation of plasticity by surprise is necessary for the rapid re-adaptation after rule switches, we compare the results of SpikeSuM over a long sequence of 10'000 presentation steps with those of two simpler networks with the same architecture but different modulation (Fig. 2.3**A**). In our reference model, the third factor $3^{rd}(\bar{A})$ has two components that yield a slow modulation of plasticity for small $\bar{A}$ and a rapid one for large $\bar{A}$ (Fig. 2.3**A**, inset, red line). The two components enable rapid learning after an unexpected rule switch, but slow learning during a long phase with a fixed rule with a residual level of 'expected uncertainty' [17] caused by stochastic transitions compatible with the rule. In comparison, a network with a constant learning rate (no modulation, SNN$_{nm}$) either converges after a switch of rules with a short delay (for a fixed high learning rate, an example is shown in Fig. 2.3**A**) towards a high-error solution or converges (for a fixed low learning rate, example not shown) to a low-error solution, but only slowly and with a much longer delay. Moreover, a network with a simpler modulation SNN$_{sm}$ shows fairly good convergence but adapts more slowly immediately after a switch. We find that, within the family of tested functions, a 3^{rd} factor built of two components as in SpikeSuM is necessary to reach adaptation that is both fast and precise, but adding a third component does not further improve learning.

A systematic comparison shows that SpikeSuM and SpikeSuM$_{rand}$ outperform SNN$_{sm}$ and SNN$_{nm}$ across various instantiations of the volatile sequence task (Fig. 2.3 **B**). Moreover, the performance of SpikeSuM is only slightly worse than that of the variational Bayesian algorithm

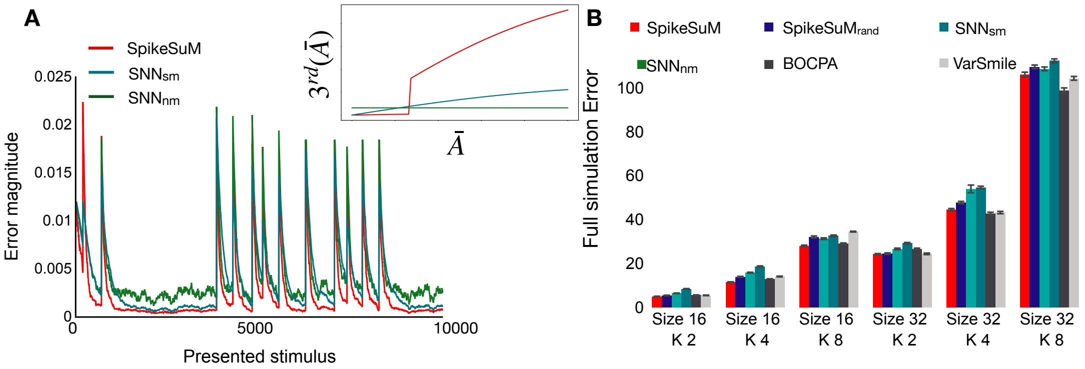
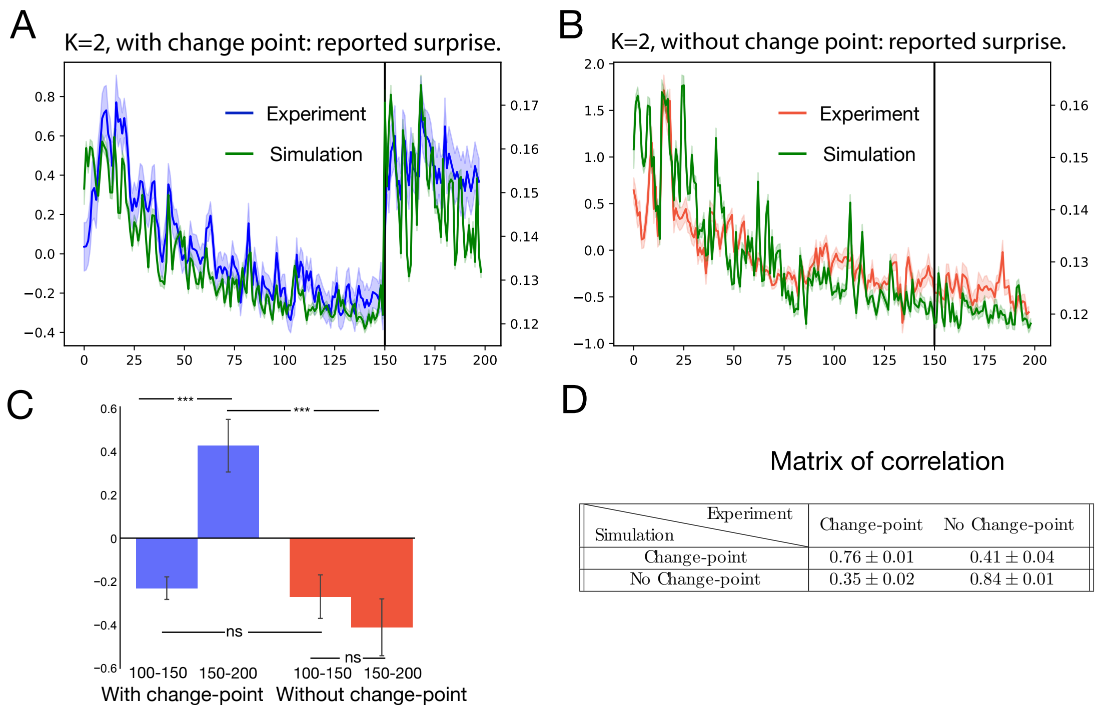


Figure 2.3: **Rapid adaptation enabled by surprise-modulated three-factor plasticity. A** Inset: The surprise signal transmitted by the 3rd factor as a function of the activity $\bar{A}$ in the prediction error layer for three cases (red: SpikeSuM rule; cyan: simplified modulation rule; green: constant learning rate, no modulation). The parameters of all three rules have been optimized. Main graph: Error magnitude of the transition matrix (Frobenius norm between the true transition matrix T^* and the estimated matrix T) as a function of time for the SpikeSuM model (red), and a Spiking Neural Network model (SNN) with the same architecture and number of neurons as SpikeSuM, but simple modulation (cyan SNN$_{sm}$) or no modulation (green SNN$_{nm}$), in a volatile sequence task with $\mathscr{R} = 16$ images and $K = 4$ possible transitions. The occasional abrupt increases in error are caused by rule switches. The SpikeSuM network exhibits faster learning immediately after the switch, as well as better convergence during periods when the rule stays fixed; volatility H=0.001. **B** Average error over 10'000 presentation steps with volatility $H = 0.001$ for different values of $\mathscr{R}$ (size) and K. The performance of SpikeSuM is comparable to that of the Bayesian Online Change Point detection algorithm (BOCPA) and better than SNN$_{nm}$ or SNN$_{sm}$. The results with random connectivity SpikeSuM$_{\text{rand}}$ are shown in dark blue.

varSMiLe [32] or the online Bayesian change point detection algorithm BOCPA [75] which are both surprise-based machine learning algorithms designed for near-optimal change-point detection (**Material and methods**).

In summary, on the volatile sequence task without re-occurrence of the same rule, our spiking network with surprise-modulated learning shows faster relearning after a rule switch than the one without, which suggests an important role of surprise-modulation in rapid, yet precise, adaptation to changes in the stimulus statistics. Importantly, the surprise signal is not some external variable but is extracted from the spiking activity of the network itself. Moreover, the feeling of surprise reported by human participants is strongly correlated with the simulated surprise signal in the SpikeSuM network if the model is stimulated with the same sequence (Fig. 2.4). Thus, the surprise signal in the SpikeSuM network links the notion of surprise in a behavioral experiment with a functional role for modulating synaptic plasticity at the level of neuronal circuits.



Experiment / Simulation	Change-point	No Change-point
Change-point	0.76 ± 0.01	0.41 ± 0.04
No Change-point	0.35 ± 0.02	0.84 ± 0.01

Figure 2.4: **Simulated compared to behavioral surprise**. We ran 60 simulations of SpikeSuM using the specific sequence with change-points from the behavioral experiments of **Fig. S1**, and 30 further simulations using the sequence without change points. **A** Scaled normalized surprise (cf. Fig. S1) as a function of time in simulations with SpikeSuM (green line: mean; shaded green: variance) and from the 65 participants in group 1 (blue line: mean; shaded blue: variance) for sequences **with** change-points. **B** Same as A, but for sequences **without** change-points; experimental data in red colors. **C** Differences in the experimental data of participants are significant (t-test) in A between the 50 steps before and 50 steps after the change point (blue bars in C); not significant in B between the 50 steps before and 50 steps after step 150 in the absence of change point (red bars in C); and significant for the time steps 150-200 between A and B (blue vs. red bar in C). The symbol *** indicates $p < 10^{-5}$, and 'ns' not significant. **D** Bootstrapped (20 random draws of 10 subjects) correlation between simulation and experiment. Simulation in the presence of a change point strongly correlates with the experimental data in the presence of a change point, but much less with the experimental data in the absence of a change point, and analogous for simulations without a change point. The table shows bootstrapped correlation values plus standard deviation.

2.2.4 Continual learning across rule switches is supported by the surprise signal

So far, each switch of a rule introduced a sequence of observations with a new transition matrix so that the previously learned transition matrix could be overwritten at no loss. To explore continual learning across rule switches, we now consider a task where the same rules ('apartments') reappear several times (Fig. 2.5 **A**). We study a meta-network composed of M SpikeSuM modules, each acting as one of the rule memories (Fig. 2.5 **B**). We call this enlarged network SpikeSuM-C (for SpikeSuM with Context). Note that the sets of stimuli (e.g., the different wallpapers) are the same for all rules so the context needs to be inferred from the observed sequences. Ideally, each module $m \in M$ should focus on one of the contexts, i.e., a single set of rules. We postulate that in a well-functioning network, only predictions within the currently active rule are updated while multiple other contexts that were memorized before are left untouched and can be reused later when the same context reappears.

To implement this idea, we assume that a set of 'context selector modules' (CSMs) selects the specific module that should learn the observed transition (**Material and methods**). The indirect coupling of context memories via the CSMs gives rise to a *Best-Predictor-Learns* (BPL) architecture, such that only the context module m with the *lowest* activity in the prediction-error layer updates its weights. Importantly, the prediction-error module with the lowest activity is the one with the best prediction for the currently observed transition.

All CSMs compete with each other via standard Winner-Take-All dynamics [132], such that all CSMs are silent except one. However, none of the prediction error neurons is shut down by the competitive dynamics between CSMs, so an arbitrary population p in module m has a non-zero activity. To restrict synaptic plasticity to the prediction error module with the lowest activity, we hypothesize that the nucleus that broadcasts the third factor is organized in several segments, such that segment m sends a neuromodulatory signal $3^{rd}(\bar{A}^m)$ to the corresponding prediction-error module m. Such a structure with localized feedback loops is compatible with the anatomy of the higher-order thalamus [100, 126, 127] or the ventral tegmental area [125]. More specifically, in our model the activity of populations P_1 and P_2 in the SpikeSuM module m excites segment m of the nucleus. In parallel, high activity of another CSM $m' \neq m$ (i.e., m' is the winner) inhibits PT neurons in the prediction-error layer of module m and hence suppresses segment m of the nucleus. But without neuromodulatory activity, plasticity does not occur in module m. Taken together, excitation and inhibition ensure that only the module m' with the lowest prediction error updates its weights (**Material and methods**)

To illustrate the function of the network, we initialize it with 5 empty context modules and stimulate it with a stochastic sequence generated by switches between four different rules. Figure 2.5 **D** shows that SpikeSuM-C learns a new rule as fast as SpikeSuM (equivalent to SpikeSuM-C with 1 module). Moreover, if a known rule reappears it re-activates an existing module instead of learning from scratch. Switches to a previously learned rule trigger a rapid switch of the network to the correct module. Finally, we find that if the number of learned rules is smaller than the number of allocated modules, empty modules stay untouched and therefore remain available for later use (Fig. 2.5 **E**).

The amplitude of the surprise signal after a switch to a previously encountered rule is smaller than that after a switch to a completely new rule (Fig. 2.5 **D**). In the first case, surprise leads to a switch to an existing module while in the second case to the recruitment of a previously untouched module. Thus, the surprise signals that are generated in the network are used by the same network to trigger learning or switching between context modules - all in an unsupervised manner (**Material and methods** for more details).

2.2.5 A modular network architecture avoids the stability-plasticity dilemma

In both SpikeSuM and SpikeSum-C, neurons in the prediction-error layer exhibit low activity for known transitions and large transient activities for surprising ones (Figs. 2.2**D-E** and 2.5 **D**). This finding is in agreement with experimental studies using visual [122, 133, 134], auditory [58, 118, 135], or tactile [117, 136] tasks. The two model networks contain separate populations of neurons, P1 and P2, that respond to negative and positive prediction error, respectively, consistent with in-vivo data [57, 60, 122, 129, 130]. Here we ask how the plasticity of synapses onto prediction-error neurons succeeds to avoid the stability-plasticity dilemma [109] which stipulates that learning is either slow or leads to overwriting of earlier memories.

We find in the volatile sequence task with re-occurrence of rules that the 'effective modulation' of prediction-error neurons is radically different between SpikeSuM and SpikeSuM-C model networks (Fig. 2.6 **A1-A2**). In the network with multiple modules, a large activity of prediction error neurons in layer-2/3 of module k does not cause the emission of neuromodulator in module k since another module is the winner. The magnitude of the neuromodulator signal influences the amount of Hebbian learning. The magnitude of synaptic changes in populations P1 and P2 of the SpikeSuM network *without* modular architecture (equivalent to a SpikeSuM-C network with a single module) depends monotonically on the total Hebbian drive, i.e. on the combination of presynaptic activity with an elevated postsynaptic membrane potential (Fig. 2.6 **B1**). While in the network with a single module, the increase is monotonic, this is not the case for the network with multiple modules (Fig. 2.6 **B2**). For small and medium values of the Hebbian drive, the update magnitude increases with the Hebbian drive whereas, for a Hebbian drive above a value of 0.6 (a.u.), the magnitude decreases back to negligible values (Fig. 2.6 **B2**).

We observe the same qualitative difference between the SpikeSuM and SpikeSuM-C model networks if we study the total incoming plasticity of a single neuron in the prediction error layer, i.e., the update magnitude of all synapses onto a given neuron as a function of the membrane potential averaged over all prediction error neurons (Fig. 2.6 **C1-C2**). Since the membrane potential encodes positive (in population P2) or negative (in population P1) prediction errors, the graph in Fig. 2.6 **C1-C2** can be interpreted as the total amount of synaptic plasticity (vertical axis) as a function of prediction error (horizontal axis). The small magnitude of synaptic changes for very large prediction errors (Fig. 2.6 **C2**) is functionally important because it leads to the protection of existing modules after a switch of context. The two regimes, i.e., memory modification for small prediction errors and memory protection for large prediction errors in a given module, separated by an inverted-U-shaped curve, have been hypothesized by Gershman et al. as a model for memory modification and formation [68]. Therefore, our model has translated a hypothesis at the cognitive level [68] into specific experimental predictions for synaptic plasticity at the circuit level. In an in-vivo experiment involving multiple contexts, presynaptic activation and postsynaptic membrane potential of putative prediction-error neurons should be monitored while the size of the synaptic connection is measured, e.g., by spine size estimation from optogenetic experiments. We speculate that in primary sensory areas future experimental observations might resemble the qualitative features of SpikeSuM whereas in frontal cortex or subcortical areas those of SpikeSuM-C.

2.3 Discussion

Our network of spiking model neurons enables rapid formation of context-dependent expectations in a paradigm of continual learning where rule switching occurs at unknown moments in time. Importantly, rapid adaptation becomes possible by surprise-modulated learning. In contrast to earlier implementations of surprise in cognitive neuroscience models [17, 30, 31, 52, 71, 73], surprise manifests itself in our spiking neural model by increased population activity caused by a momentary imbalance of excitation and inhibition [60, 95]. The surprise signal has two different roles in our model. First, it triggers the release of feedback signals (e.g., neuromodulators) that serve as 'third factors' in an unsupervised neoHebbian learning rule [82, 83, 86]. Second, it initiates switches between modules and avoids overwriting old memories [68, 111, 112], since synaptic plasticity is disinhibited only in the module representing the current rule.

In our approach, predictive coding is a consequence of our aim to extract a surprise signal from spiking activity—as opposed to classic approaches where predictive coding is a consequence of redundancy-reducing or energy-minimizing codes [96, 107]. Surprise requires expectations that arise from earlier experience. In our model, the sensory experience of the previous presentation step is represented in the buffer population, while predictions are encoded in the connection weights. Our model does not specify whether the buffer population is located in the same area (e.g., cortical L5 cells [60]) or in some other area (e.g., prefrontal cortex [137, 138]). The antisymmetric architecture of the prediction-error circuit in each module requires two separate excitatory and inhibitory pathways onto model neurons that extract positive and negative error prediction, similar to putative prediction error neurons in layer 2/3 of the sensory cortex [123, 139]. We propose that the activity of these neurons is summed, and potentially low-pass filtered, by layer 5b PT neurons [87, 128] which would then transmit the aggregated signal ('surprise') to other areas or nuclei that eventually trigger a feedback signal such as the release of a neuromodulator. While positive or negative prediction errors can be assigned to *single* neurons, surprise in our model is extracted from the *aggregated* unsigned prediction error—available by a summation over large groups of neurons.

Our model is a conceptual one and makes no specific predictions on the type or origin of these feedback signals. However, candidate sources for such feedback signals could be acetylcholinergic neurons in nuclei of the basal forebrain and brain stem, potentially linked to arousal and plasticity [100, 124, 140]; noradrenergic neurons in locus coeruleus linked to cognition, attention, and network reorganization [59, 141]; serotonergic neurons in the Raphe nuclei linked to surprise [50]; dopaminergic neurons in the ventral tegmental area linked to reward [142]; or populations of neurons in the higher-order thalamus potentially linked to consciousness or predictive processing [100, 126, 127]. At least for the last two, it is known that the population is not homogeneous but structured, [125, 127] which is a necessary condition for the proposed model of switching between different rules encoded in different modules. Even though dopamine is largely correlated with reward and reward prediction error [12, 142], dopamine has also been linked to novelty and potentially surprise [142, 143]. On the other hand, dopamine can also be triggered by activity in locus coeruleus [144, 145], a nucleus that is traditionally associated with noradrenaline [141]. Hence, a one-to-one mapping between neuromodulators and functional roles should not be expected [84].

Predictions in our model are encoded at two levels, i.e., in the weights of synaptic connections and the activity pattern of excitatory neurons in the prediction-error layer (**Supplementary Fig. S2.B.2**). While the model was not designed to reproduce experimental data of frontal cortex neurons, several aspects of the activity patterns in the SpikeSuM-C model are qualitatively consistent with delay activity [138], implicit encoding of associations [137], and mixed activity profiles [146] which enables to decode from the population activity the current rule, the present input, the previous state, and alternative observations consistent with the previous state but inconsistent with the present input.

A distinction between expected and unexpected uncertainty has been proposed in the literature on reward-based learning [17, 147]. Analogously, we can define expected and unexpected uncertainty in the absence of rewards. In our volatile sequence task, the expected uncertainty depends on the number K of possible next stimuli whereas the unexpected uncertainty corresponds to unpredictable switches between rules. For $K = 1$, the expected uncertainty vanishes. For $K > 1$, the level of expected uncertainty is, after learning, represented in our model by the remaining activity of excitatory neurons in the prediction error layer which could be tested in experiments (**Supplementary Fig. S2.B.1**). Expected uncertainty can also be visible as a non-zero tonic level of the surprise signal (i.e., the 3rd factor). The unexpected uncertainty is represented by sharp peaks in the activity of the prediction error neurons (Fig. 2.5 **D**).

Detecting unpredictable switches in the rules governing the momentary environment is a challenge for both artificial neural networks [108] and biological brains [50, 138]. If rule switching is not detected, for example, because of reduced serotonergic signaling, behavior exhibits reduced adaptation speed [50] or even obsessive-compulsive signatures [50, 148]. Surprise in our model is putatively related to mismatch negativity in EEG signals. Interestingly, schizophrenia patients exhibit a reduced mismatch negativity[149] and a reduced capacity to make valid prediction [150–152]. In our model, missing surprise signals lead to an impairment of memory formation and consolidation, potentially linked to deficits in schizophrenia patients [153–156].

Definitions of surprise in a probabilistic framework [27] have previously been used to explain adaptation to rule switching [75–78]. However, these definitions cannot be directly applied to spiking neural networks since a correct normalization of probability distributions is difficult to maintain within spiking networks [157, 158] and the calculation of a distance, or Kullback-Leiber divergence, between two probability distributions [27, 159] is even harder. Surprise-driven neural networks for adaptive decision making [160] or neural particle filters for adaptive perception [161] are not easily extendable to networks of spiking neurons. Our approach extracts from the activity of spiking neurons a qualitative surprise signal that can be interpreted as a measure of observation-mismatch surprise [27] without a direct link to probability distributions. In summary, surprise, i.e., a response of the brain to a stimulus that occurs against expectations [27, 38–40], is a phenomenon of relevance similar to that of reward. Similar to reward and reward expectations [12], surprise must be detected by neuronal networks in the brain and transformed into modulatory signals that influence synaptic plasticity. Our conceptual model study shows how surprise detection and modulation of plasticity can be implemented in spiking neural networks and how these networks can be used for memory formation, memory protection, and prediction of upcoming inputs, in the absence of reward.

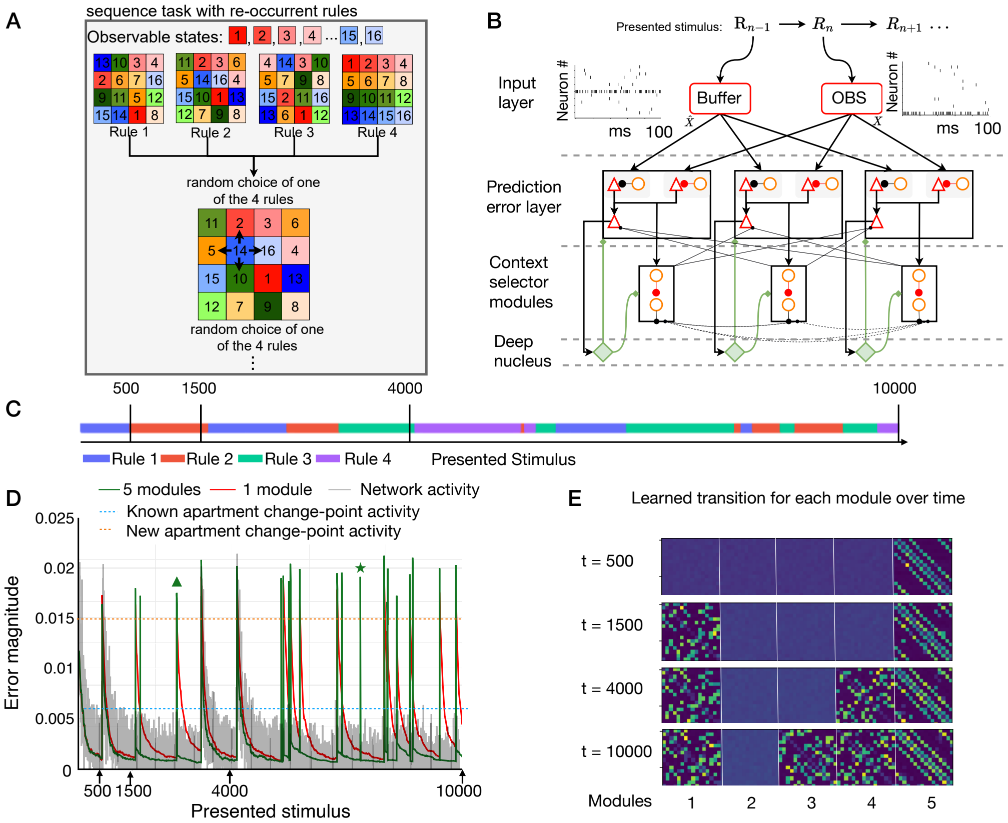


Figure 2.5: **Continual learning across rule switches. A**: Sequence task with re-occurring rules. At the beginning, M rules (here represented as $M = 4$ rules) are created by a random assignment of different wallpapers to $\mathscr{R} = 16$ rooms. At each change point, a rule is randomly picked from these M rules. **B**: The SpikeSuM-C network is composed of four layers. The input layer receives the stimulus and connects to the prediction-error layer which is composed of several SpikeSuM modules (see section 2.2.2); only three are shown. A set of context selector modules (CSM) composed of dis-inhibitory networks is bidirectionally connected with the prediction-error layer. Each SpikeSuM module excites its corresponding CSM. A Winner-Take-All circuit in the CSM layer selects the least excited module. The inhibitory feedback weights from the CSM to the prediction-error layer inhibit the PT neurons of unselected SpikeSuM modules, but not the prediction-error neurons. (see **Material and methods**). Red weights are plastic. Non-plastic weights are shown in black for feedforward, solid blue for feedback, and dashed blue for lateral inhibitory connections. **C**: Sequence of rule switches as a function of time. **D**: Summed activity of all PT-cells (gray, arbitrary units) in a SpikeSuM-C network with 5 modules and error magnitude (green, mismatch between transition matrix in currently selected module and ground truth) during learning. When the SpikeSuM-C is exposed to the second rule for the second time, the error exhibits a short spike (green triangle) before it instantaneously goes down, indicating successful switching between modules. At rare moments (green star marks one of the examples) module switching is initiated at an inappropriate moment but stops immediately thereafter. The activity generated by the switch to an unknown rule is much stronger (gray bars exceed the horizontal orange dashed line) than that of a previously observed one (gray bars never reach the cyan dashed line). SpikeSuM-C can create new memories or reactivate memories previously learned if a known transition rule reappears. Red line: behavior of SpikeSuM (control, 1 single module). **E** Evolution of synaptic weight matrices over time for each of the five modules. After 500 time steps, the transition matrix of rule 1 has been stored in module 5, and transition matrices of other rules are added as they appear.

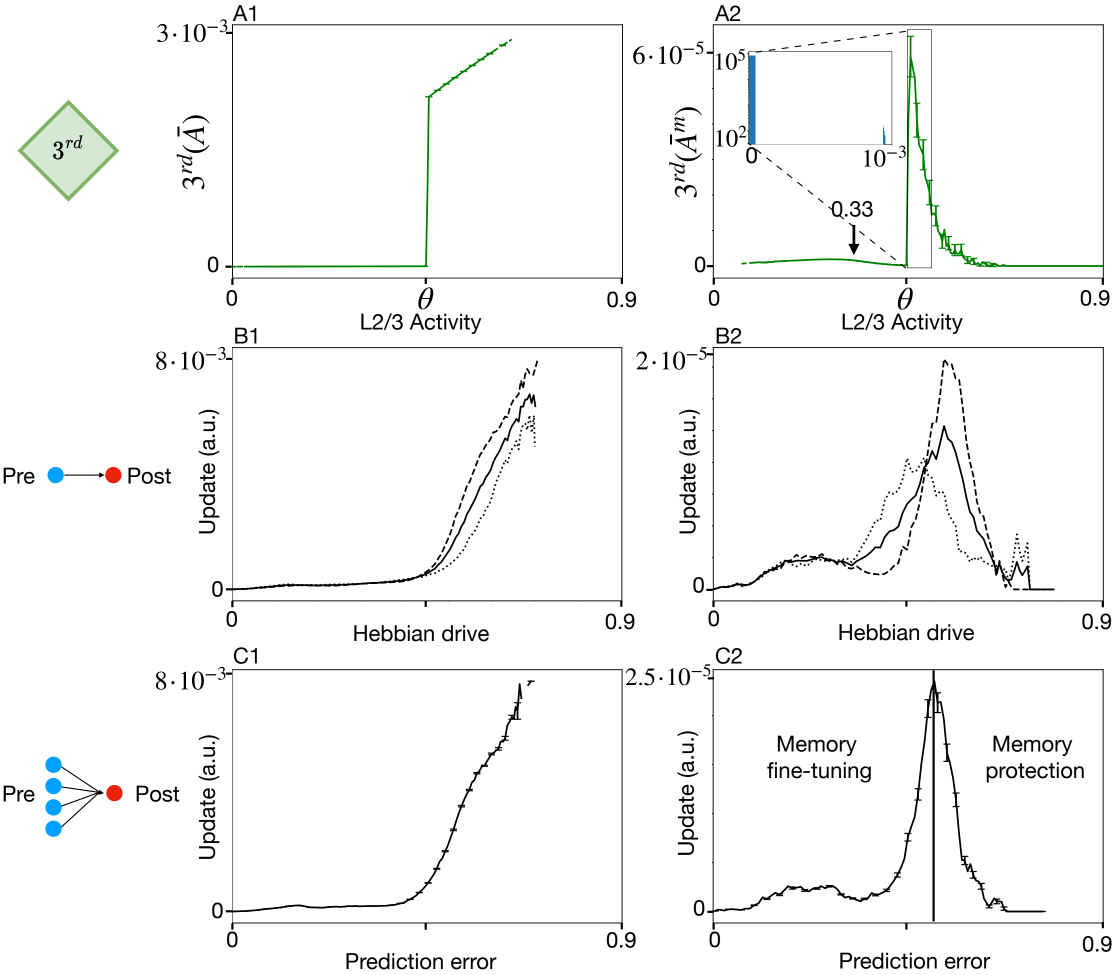


Figure 2.6: **Synaptic plasticity as a function of prediction error has two regimes in SpikeSuM-C. A1-A2**: The magnitude of modulation (3^{rd} factor) is shown as a function of the total activity A of layer-2/3 neurons for a SpikeSuM-C network with a single module (A1; equivalent to SpikeSuM; cf. inset of Fig. 2.3A) and for a SpikeSuM-C network with three modules (A2). Bars: standard error of the mean. The difference between the two curves (A1-A2) arises from the inhibition of model PT-neurons if they are not located in the winning module: in A1, the activity $\bar{A}$ of PT neurons always reflects the activity A of layer-2/3 neurons, in A2 it does not. Inset: Histogram of modulation amplitudes $3^{rd}(\bar{A}^m)$ for values slightly above θ: the distribution of modulation amplitudes is bimodal with rare events of large modulation. Arrow: the peak is due to known transitions that remain after a rule change. **B1-B2**: Update magnitude $|\Delta w_{ik}|$ of a specific synapse is shown as a function of the Hebbian drive $|\tanh(h_i)\text{EPSC}_k|$ i.e., the combination of postsynaptic membrane potential and the current influx caused by presynaptic spike arrival. The short-dashed and long-dashed lines show the average over the population P1 and P2, respectively. The solid line is the average over both populations.**C1-C2**: The total amount of synaptic plasticity, represented by the update magnitude $\sum_k |\Delta w_{ik}|$ summed over all synapses onto an arbitrary neuron i is shown as a function of the prediction error, represented by the scaled membrane potential $|\tanh h_i|$. In a network with a single module (C1), plasticity increases with prediction error so that large prediction errors after a context change lead to overwriting of existing memories. In the network with multiple modules (C2), the plasticity in the SpikeSuM-C network exhibits two regimes: prediction errors between 0.1 and 0.4 generate small but non-negligible changes, and induce a refinement of existing memories, whereas for prediction errors above 0.6 existing memories are protected since other memories are created or changed. The error bars represent the 90% confidence interval of the mean. The two regimes agree with a prediction of Gershman et al.[68].

Appendix

2.A Material and methods

2.A.1 Two volatile sequence tasks

In the volatile sequence tasks (Fig. 2.1 **A** and 2.5 **A**), a sequence of stimuli is generated by a doubly stochastic Markov chain. At each presentation step, a stimulus with index q is chosen from a finite set of $\mathscr{R}$ different inputs, $1 \leq q \leq \mathscr{R}$. Given a stimulus q at presentation time step n, a stimulus k at presentation step $n+1$ is chosen with probability $T^*_{k,q}$ where T^* is the transition matrix that summarizes a given rule. At each presentation step, rules switch stochastically with probability $H \ll 1$, called the volatility of the rule. We often refer to the moment of rule switch as a 'change point'. From the point of view of the observer, switches are unexpected and potentially cause a high surprise.

While the theory is more general, we often visualize stimuli as static wallpaper images collected by a video camera that is moved randomly across an apartment composed of $\mathscr{R}$ rooms (Fig. 2.1A), each enabling K possible transitions to other rooms. Rooms have distinct wallpapers. The stimulus R_n stands for the wallpaper in the room seen at presentation step n. The transitions are stochastic and follow:

$$P(R_{n+1}|R_n) = \begin{cases} \frac{1}{K} \text{ if stimuli } R_{n+1} \text{ and } R_n \text{ refer to connected rooms} \\ 0 \text{ otherwise,} \end{cases} \tag{2.4}$$

We assume periodic boundary conditions, e.g., room 4 in Figure 2.1A (top) is a neighbor to rooms 1,3,8, and 16. Thus, the layout of the apartment defines the hidden rule of allowed transitions between stimuli. In particular, a transition matrix generated from a given apartment has the property that for each starting stimulus q, the elements $T^*_{k,q}$ either vanish or take a value $T^*_{k,q} = 1/K$ with constraints $\sum_k T^*_{k,q} = 1$ and $T^*_{q,q} = 0$. In the theory below, we do not assume that the transition matrix is symmetric, even though whenever we simulate an apartment with two-dimensional layout and $K = 4$ (or a 1-dimensional apartment with $K = 2$), then the matrix is symmetric $T^*_{k,q} = T^*_{q,k}$.

We design two tasks with different switching patterns. For the first task ('volatile sequence task without re-occurrence of rules'), at each change point, all wallpapers are randomly shuffled. Thus, at each change point, a new transition rule is generated. For the second task ('volatile sequence task with re-occurrence of rules'), we first randomly assign the set of $\mathscr{R}$ different wallpapers M times to different apartments, each of $\mathscr{R}$ rooms. At each change point, we randomly choose one of the $M-1$ possible other apartments. Thus, the number of potential transition rules is finite. The first task implies that having a memory of past apartments is vain, as there is a very low probability to observe the same apartment multiple times. Hence, an adaptive algorithm with rapid forgetting is suited to solve this task. For the second task, a suitable algorithm should memorize context dependent predictions and quickly re-activate the correct context after each rule switch.

In the simulations in the main text, we use apartments with either $\mathscr{R} = 16$ or $\mathscr{R} = 32$ rooms and vary the number K of allowed transitions per room between $K = 2$ and $K = 8$. The terms 'apartment', 'room', 'wallpaper' are for illustration purpose only, since each stimulus is represented in the model by a unique neuronal input pattern (see below).

2.A.2 Spike trains of sensory neurons

To simulate the volatile sequence task with $\mathscr{R}$ discrete stimuli ('wallpapers of rooms'), we translate the wallpaper images (introduced in 2.A.1) into spiking patterns of abstract 'sensory' neurons: each stimulus is represented by a distinct cluster of $m = 8$ neurons with elevated firing rate of 100Hz. (We think of these 'sensory' neurons as the output of a multi-layer network with wallpaper images as input and 8-hot coding as output; but we do not implement such a preprocessing network; see **Supplementary Discussion**). Each stimulus presentation lasts for 100ms (=1 presentation step), and thereafter a new input stimulus is presented to the network. The network input layer is composed of two populations of 'sensory' neurons: a population of observation neurons and a population of buffer neurons (see Fig. 2.2). Both populations consist of $m \times \mathscr{R}$ Poisson neurons. Note that we use $m = 8$ neurons per cluster to have a good estimation of the firing rate; however, for a network of rate neurons it would be sufficient to use a single neuron per stimulus (1-hot coding).

In a network of $8 \times \mathscr{R}$ presynaptic neurons per sensory population, the first cluster of 8 neurons represents the first stimulus ($q = 1$) of the volatile sequence task, the second cluster consisting of neurons 9 to 16 the second one and so forth. For each observation, neurons in one of the cluster will spike with firing probability 0.1 at each time step of $dt = 1$ms (firing rate 100Hz), whereas all other neurons fire with probability $\epsilon \ll 0.1$ at each time step. Note that, sensory neurons in the buffer population have the same behavior as those in the observation population except that active neurons encode the stimulus number of the previous observation.

2.A.3 Transmission from sensory neurons to prediction error neurons

Each spike z_k in a neuron k of one of the sensory populations triggers a square EPSC of length $l = 4$ms, which is transmitted to neurons in the prediction error layer consisting of two populations $p \in \{P_1, P_2\}$. The total input current I_i into neuron i of the prediction error layer is

$$I_i^{P_1} = \hat{x}_i^{P_1} - x_i^{P_1}, \tag{2.5}$$

if neuron i is in population P_1 and

$$I_i^{P_2} = x_i^{P_2} - \hat{x}_i^{P_2}, \tag{2.6}$$

if neuron i is in population P_2. Here,

$$x_i^p(t) = \sum_k w_{ik}^{p,o} \mathrm{EPSC}_k^o(t), \tag{2.7}$$

is the input from sensory neurons in the observation population to neuron i in population p of the prediction error layer where $\mathrm{EPSC}^o_k(t)$ is 1 if neuron k in the observation population has fired in the last 4ms and $w^{p,o}_{ik}$ are fixed observation weights. Similarly,

$$\hat{x}^p_i(t) = \sum_k w^{p,b}_{ik}\mathrm{EPSC}^b_k(t). \tag{2.8}$$

is the input from sensory neurons in the buffer population to neuron i in population p of the prediction error layer, where $\mathrm{EPSC}^b_k(t)$ is 1 if neuron k in the buffer population has fired in the last 4ms and $w^{p,b}_{ik}$ are plastic weights driven by the plasticity rule of Eq. 2.19. We refer to $\hat{x}^p_i(t)$ as the (learned) prediction and to $x^p_i(t)$ as the (representation of the present) observation. To simplify the notations, we drop in the following the time argument t and replace

$$\begin{aligned} x^p_i &\triangleq x^p_i(t) \\ \hat{x}^p_i &\triangleq \hat{x}^p_i(t) \end{aligned} \tag{2.9}$$

2.A.4 Spiking neuron model

Neurons in the prediction error layer are described by the Spike Response Model SRM_0 [132, 162]. Each prediction error neuron i receives an input current I^p_i where p stands for P_1 or P_2; cf. Eqs. 2.5 and 2.6. The input current is then integrated to obtain the input potential (Fig. 2.A.1)

$$\tau\frac{dh^p_i}{dt} = -h^p_i + I^p_i \quad \longrightarrow \quad h^p_i(t) = \frac{1}{\tau}\int_{-\infty}^{t} e^{\frac{s-t}{\tau}} I^p_i(s)\mathrm{ds} \tag{2.10}$$

Combining the input potential with a refractory kernel η leads to the membrane potential

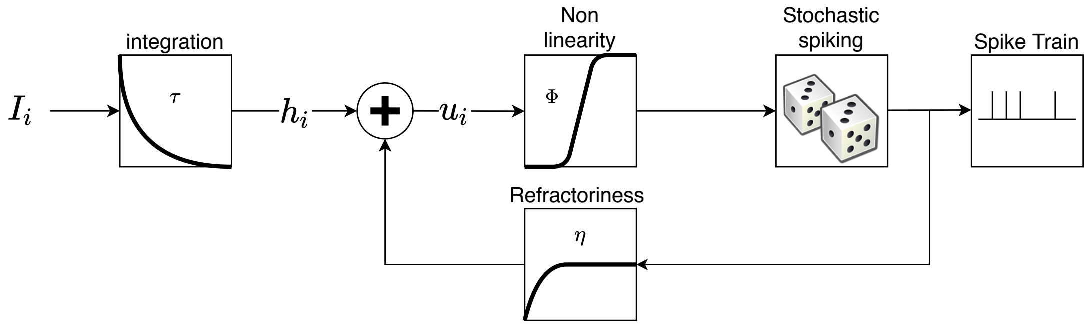


Figure 2.A.1: **Spike Response Model of neurons in the prediction error layer.** Each postsynaptic neuron receives an input current I_i. This current is integrated, with membrane time constant τ, to obtain the input potential h_i. The actual membrane potential of the neuron u_i is the combination of both the input potential and a refractory function η, where η is a strong negative potential activated after a spike, forcing the neuron to stay silent for a while. The spike times are then randomly drawn with probability $\phi(u_i)$ generating the spike train of neuron i. Redrawn from [132].

$$u_i^p(t) = \eta(t - t_i') + h_i^p(t), \tag{2.11}$$

where t_i' stands for the *last* firing time of post-synaptic neuron i, and $\eta(t - t_i') = -e^{-\frac{(t-t_i')}{\tau}}$ is an exponential refractory function, preventing the neuron to fire again right after a spike. Spikes are generated stochastically with probability

$$P(z_i = 1) = \phi(u_i) = [\tanh(u_i)]_+ \tag{2.12}$$

per time steps of $dt = 1ms$ where ϕ is the activation function of the neurons and $[x]_+ = x$ for $x > 0$ and zero otherwise. Eqs. (2.10), (2.11) and (2.12) define the Spike Response Model of the prediction error neurons.

2.A.5 Two connectivity patterns onto the prediction error layer: random and regular

The first projection pattern (SpikeSuM$_{rand}$) is sparse random connectivity (with density 0.1) and weights uniformly drawn between 0 and 1. In other words, for each of the 256 postsynaptic neurons in the prediction error layer, we draw an input connection to a specific presynaptic neuron with a probability of 10 percent and then connect the two neurons with a random weight (Fig. 2.2). Since in our standard simulations we have 16 different stimuli and each stimulus is represented by a distinct cluster of m =8 presynaptic neurons, the average number of input connections to a neuron in the prediction error layer is $0.1 \cdot m \cdot \mathscr{R} = 12.8$ with a mean weight of 0.5. Thus, in the prediction error layer stimuli (wallpapers) are represented by overlapping groups of neurons of different firing rates (coarse coding).

The second projection pattern is a regularly structured connectivity pattern (SpikeSuM). Presynaptic neurons are, as before, separated in $\mathscr{R}$ clusters of $m = 8$ neurons each, but each cluster projects (with binary weights) to a different group of 8 neurons in the prediction error layer. In other words, both pre-and postsynaptic layers are composed of $8\mathscr{R}$ neurons such that different stimuli are represented in the prediction error layer by distinct, non-overlapping groups of neurons (Fig. 2.2).

2.A.6 SpikeSuM Network architecture

Eqs. (2.5) and (2.6) show that $I_i^p = 0$ if and only if $\hat{x}_i^p = x_i^p$, for $p \in \{P_1, P_2\}$. Note that $\hat{x}_i^p$ is the prediction arising from the activity of the buffer population whereas x_i^p is the present observation. Hence, the total input is minimal if the prediction coincides with the observation. A wrong prediction increases the activity in at least one of the two populations in the prediction error layer: if $\hat{x}_i^p > x_i^p$ for many neurons in $p = P_1$, then many neurons in population P_1 have a positive input current and nonzero spiking activity; on the other hand, if $\hat{x}_i^p < x_i^p$ for many neurons in $p = P_2$, then many neurons in population P_2 have a positive input current and nonzero spiking activity. Because of the rectification at the transition from neuronal input I_i^p

to output spikes (Eq. 2.12), the two populations P_1 and P_2 complement each other. A natural way to estimate the overall prediction error of the network is therefore to collect the spikes of both populations P_1 and P_2. We assume that the population of PT-neurons acts as a linear filter and transmits a mean activity $\bar{A}$ defined as

$$\tau_{\bar{A}} \frac{d\bar{A}}{dt} = -\bar{A} + c \sum_{p=P_1,P_2} \sum_i \mathrm{EPSC}_i^p, \tag{2.13}$$

where EPSC_i^p denotes the square excitatory postsynaptic current of neuron i of population p and $\tau_{\bar{A}}$ and c are constants. In our model, $\bar{A}$ provides the total drive of neurons in a deep nucleus that receives dense input connections from PT cells. The neurons in the deep nucleus send back a broadcast signal that measures the total surprise

$$'\mathrm{surprise}' = \eta_1 \tanh(\bar{A}) + \eta_2 \tanh(\bar{A})\Theta(\bar{A}-\theta) = 3^{rd}(\bar{A}) \tag{2.14}$$

where $3^{rd}(\bar{A})$ is a non-linearly increasing function of $\bar{A}$, Θ is the Heaviside step function and η_1, η_2, θ are fixed hyperparameters. Since this surprise signal modulates learning, we refer to it as a 3^{rd} factor that gates plasticity in neoHebbian three-factor learning rules [86].

The third factor, composed of two non-linear components, could either be interpreted as a single neuromodulator with a complex nonlinearity or alternatively as the combined action of two neuromodulators involved in surprise-based learning [17]. Following the terminology of [17], the adaptation to the *expected* uncertainty (e.g., stochastic transitions to one of the possible next stimuli under a fixed rule) could be controlled by the action of acetylcholine [described in our model by the term $\eta_1 \tanh(\bar{A})$], whereas the adaptation speed to the *unexpected* uncertainty (i.e. a rule switch) could be controlled by the action of norepinephrine [turned on in our model if $\bar{A} > \theta$].

2.A.7 SpikeSuM learning rule: Derivation of Hebbian factors

We aim for a neoHebbian plasticity rule with three factors [86], i.e., a rule that combines traces of pre-and postsynaptic activity with a modulation of the learning rate. As indicated above, a good prediction of the present observation is indicated by a small current in Eqs. (2.5) and (2.6) or, similarly, by a small value of the input potential h_i^p of all pyramidal neurons in populations P_1 and P_2; cf. Eq. (2.10). We, therefore, minimize the loss function

$$\mathscr{L} = \int_{t^{start}}^{t^{stop}} \hat{\mathscr{L}}(t)\, dt = \int_{t^{start}}^{t^{stop}} \frac{1}{2} \sum_{p=P_1,P_2} \sum_i [h_i^p(t)]^2\, dt \tag{2.15}$$

where t is time and runs from the beginning t^{start} to the end t^{stop} of the experiment. Optimization is implemented as online gradient descent with respect to the weights $w_{ik}^{p,b}$ that project from neuron k in the buffer population to neuron i in population p of the prediction error network. We recall that weights $w_{ik}^{p,o}$ from observation neuron k to neuron i are fixed.

We present here all the calculations for $p = P_1$ only. For the population P_2, one just needs to add a minus sign. The integral over time corresponds to a batch rule; for stochastic gradient descent (online rule) we can focus on an arbitrary point in time and apply the chain rule of differentiation

$$\frac{\partial \hat{\mathscr{L}}(t)}{\partial w_{ik}^{P1,b}} = h_i(t) \frac{\partial h_i^{P1}(t)}{\partial w_{ik}^{P1,b}} \tag{2.16}$$

where we can evaluate the derivative using Eqs. (2.5) and (2.8)

$$\frac{\partial h_i^{P1}(t)}{\partial w_{ik}^{P1,b}} = \int_{-\infty}^{t} e^{\frac{s-t}{\tau}} \frac{\partial I_i^{P1}(s)}{\partial w_{ik}^{P1,b}} ds \underset{Eq.(2.8)}{=} \int_{-\infty}^{t} e^{\frac{s-t}{\tau}} \mathrm{EPSC}_k^b(s) ds \tag{2.17}$$

Since EPSCs have a rectangular shape with duration l we can evaluate further

$$\frac{\partial h_i^{P1}(t)}{\partial w_{ik}^{P1,b}} = \sum_{t'_k \leq t} e^{\frac{-t}{\tau}} \int_{t'_k}^{t'_k+\Delta_k} e^{\frac{s}{\tau}} ds = \sum_{t'_k \leq t} e^{\frac{-(t-t'_k)}{\tau}} [e^{\frac{\Delta_k}{\tau}} - 1] \triangleq \overline{\mathrm{EPSC}}_k^b, \tag{2.18}$$

where $\overline{\mathrm{EPSC}}_k^b$ is a low-pass filtered version of the EPSC, t'_k are the spike times of neuron k and $\Delta_k = min(t - t'_k, l)$. We now apply online gradient descent with an update amplitude proportional to the variable 3^{rd} ('learning rate') and the step size dt

$$\begin{aligned} \frac{dw_{ik}^{P1,b}}{dt} &= -3^{rd}\, h_i^{P1} \overline{\mathrm{EPSC}}_k^b \quad \text{in population } P1, \\ \frac{dw_{ik}^{P2,b}}{dt} &= +3^{rd}\, h_i^{P2} \overline{\mathrm{EPSC}}_k^b \quad \text{in population } P2. \end{aligned} \tag{2.19}$$

The above neoHebbian rule combines a trace of the incoming EPSC_k^b (presynaptic factor) with the momentary input potential h_i^p (rather than the spike time) of the postsynaptic neuron (postsynaptic factor): these are the two Hebbian factors. In standard stochastic gradient descent, the learning rate 3^{rd} could be fixed or slowly decrease over time as learning proceeds ('freezing'), and also depend (via a momentum term) on the recent history. However, in our model, the learning rate increases whenever the prediction fails (indicated by a large prediction error), so that we refer to the learning rate 3^{rd} as a surprise-driven neuromodulator. To summarize, we have a three-factor learning rule with the following properties: (i) EPSC_k limits the weight update to active connections; (ii) h_i is the local signed prediction error and goes to zero if the prediction for neuron i is correct; (iii) finally, 3^{rd} is a function of the global unsigned prediction error which is sent back as 'surprise' to the full network; see main text and section 2.A.8.

2.A.8 SpikeSuM learning rule: third factor

There is no fundamental reason that a learning rate should be fixed as long as each update step (in the batch rule) decreases the loss [163]. However, in an online gradient descent rule, we have to make sure that all observations get an appropriate statistical weight during the update. In particular, we have to ensure that none of the observations is systematically ignored. This could happen if the learning rate 3^{rd} vanished whenever a specific stimulus appears. Such a problem is not a hypothetical one, because of the rectification of the neuronal gain function; cf. Eq. 2.12. Suppose that, for a given stimulus, the observation $x_i^{P_1}$ is larger than the prediction $\hat{x}_i^{P_1}$ for all neurons in population P_1. In this case, none of the neurons in population P_1 would respond. If we were to use a third factor that is proportional to the activity A_1 of population P_1 (e.g., if we set $3^{rd} = \beta A_1$), then this stimulus would never lead to an update.

However, the dependence of the third factor $3^{rd}(\bar{A})$ upon the total population activity $\bar{A}$ of the prediction error layer together with the anti-symmetric architecture avoids this problem. Whenever the observation does not match the prediction, at least one of the populations, either P_1 or P_2, will be turned on. This is true throughout the simulation because (i) there are many plastic weights that code for each stimulus (e.g., with regular connectivity and $\mathscr{R} = 16$ different stimuli, we have 64 weights coding for each stimulus in each of the two populations); (ii) all synaptic weights in both populations are initialized in the range $[0, 1]$; (iii) the update rule Eq. (2.19) is symmetric for both populations (i.e., if the excitatory weights onto a neuron in P_1 increase, then the inhibitory weights onto a neuron in P_2 decrease) which ensures that the symmetries at initialization remain throughout learning.

Thus, whenever predictions and observations do not match, the total activity $\bar{A}$ conveys a prediction error signal which leads to a non-zero learning rate $3^{rd}(\bar{A})$ that is identical for *all* weights.

2.A.9 After convergence, synaptic weights of SpikeSuM reflect transition probabilities

We claim that the anti-symmetric architecture of the prediction error layer together with the three-factor learning rule makes the weights converge to a solution that reflects the main features of the hidden rule implemented by the transition matrix $T^*_{k,q}$ between stimuli. For example, if we imagine an apartment with $K = 4$ possible transitions from a given room, then all four transitions will be encoded in the weight matrix. More generally, we want to show that, for a transition matrix $T^*_{k,q}$ with K entries of value $1/K$ per column and zero entries otherwise, the weights onto the pyramidal neurons in the prediction error layer are adjusted such that all possible transitions are predicted proportional to their statistical probabilities. We will also show how to decode the predictions of the network. To stay concrete, we illustrate the abstract mathematical concepts with the words ’room’ and ’apartment’, but the calculations are valid for arbitrary stimulus sets and transition matrices.

Preliminaries: Encoding of rooms and decoding of predicted transitions

As an abstract encoding of rooms, we use a 1-hot encoding. If the total number of rooms is $\mathscr{R}$, then room q (with $1 \leq q \leq \mathscr{R}$) is encoded by an $\mathscr{R}$-dimensional vector $\mathbf{R}_q \in \{0,1\}^{\mathscr{R}}$ with the qth component equal to 1 and all other components equal to zero. The transition matrix $T^* \in \mathscr{R} \times \mathscr{R}$ that describes the probability of a transitions from room $\mathbf{R}_q$ to room $\mathbf{R}_k$ has elements $T^*_{k,q}$ defined as

$$T^*_{k,q} = \text{Prob}(\mathbf{R}_k|\mathbf{R}_q) \tag{2.20}$$

with $\sum_k T^*_{k,q} = 1$ for all q. The set of rooms $\{\mathbf{R}_1, ..., \mathbf{R}_{\mathscr{R}}\}$ represented by 1-hot coding vectors defines an orthonormal basis in an $\mathscr{R}$-dimensional vector space, which gives rise to the following properties of the transition matrix T^*. First, the multiplication of the matrix with the room vectors from both sides gives back the transition

$$\mathbf{R}_k T^* \mathbf{R}_q = T^*_{k,q} \tag{2.21}$$

and, second, one-sided multiplication with a room $\mathbf{R}_q$ gives a vector $\mathbf{R}_{.|q}$ with non-zero elements for all those rooms that are reachable from $\mathbf{R}_q$

$$\mathbf{R}_{.|q} = T^* \mathbf{R}_q = \sum_k \mathbf{R}_k T^*_{k,q}. \tag{2.22}$$

We interpret $\mathbf{R}_{.|q}$ as the code of 'reachable rooms' starting from room q. It represents the qth column of the transition matrix T^* and can be expressed as a linear sum over the one-hot-coded rooms $\mathbf{R}_k$. In particular, for $K = 4$ neighboring rooms, the vector on the left-hand-side of Eq. (2.22) contains four non-zero entries (with a value of 1/4 each) that represent the four rooms reachable from the room with index q.

The actual encoding of rooms in the input layer of the SpikeSuM network corresponds to m-hot encoding, since a room R_q is represented in the input layer by a cluster of m neurons that fire at a high rate (ν=100Hz); cf. **Spike trains of sensory neurons**. For the sake of simplicity of the arguments below, we assume that the neurons representing other rooms $R_k \neq R_q$ are inactive ($\epsilon \to 0$) when room R_q is observed. Thus, we can think of the input representation of room q as an m-hot encoding $\mathbf{V}_q = P^{1\to m}\mathbf{R}_q \in \{0,1\}^{m\mathscr{R}}$, where $P^{1\to m}$ is the rectangular expansion matrix from 1-hot encoding to m-hot encoding transforming the $\mathscr{R}$-dimensional space of rooms into a $m\mathscr{R}$-dimensional space of input neurons.

We now turn to the representation of stimuli in the prediction error layer. For the SpikeSuM network with *regular* connectivity, the representation in the prediction error layer is also an m-hot encoding in each of the two populations P_1 and P_2. However, to keep our arguments general, we will also include the case of *random* connectivity. From Eq. (2.7) we know that the input neurons in the observation population drive neuron i in population p of the prediction error layer with a current $x^p_i(t) = \sum_k w^{p,o}_{ik}\text{EPSC}^o_k(t)$. We collect the set of neurons i in population p into a vector $\mathbf{x}^p$, and the weights $w^{p,o}_{ik}$ into a matrix $W^{p,o}$ and write the vector equation

$$\mathbf{x}^p(t) = W^{p,o}\mathbf{EPSC}^o(t)\,. \tag{2.23}$$

Let us consider a time point t_n located close to the end of the nth presentation step. Furthermore, let us suppose that during the nth presentation step room $R_{q(t_n)}$ was observed. Here $q(t_n)$ denotes the index of the room in presentation step n.

We exploit the m-hot encoding to write for the mean activity pattern in population p

$$\mathbb{E}_S[\mathbf{x}^p(t_n)] = W^{p,o}P^{1\to m}\mathbf{R}_{q(t_n)}\,v\,l \tag{2.24}$$

where $\mathbb{E}_S[\mathbf{x}]$ denotes the expectation over stochastic spiking of the Poisson neurons in the input layer, $W^{p,o}$ is the matrix of fixed connectivity weights to the pyramidal neurons in the prediction error layer, v is the firing rate of the active neurons, and l is the duration of the rectangular EPSC. Similarly, the expected prediction generated by connections from neurons in the buffer population to those in population $p \in \{P_1, P_2\}$ is

$$\mathbb{E}_S[\hat{\mathbf{x}}^p(t_n)] = W^{p,b}P^{1\to m}\mathbf{R}_{k(t_{n-1})}\,v\,l. \tag{2.25}$$

Since we would like to interpret activity patterns in terms of the rooms, we introduce hypothetical decoding weights D^p from the space of neuronal activities (in one of the pyramidal populations in the prediction error layer, $p \in \{1,2\}$) to the space of room labels in 1-hot coding. We choose decoding weights such that encoding followed by decoding forms an auto-encoder for arbitrary rooms R_q:

$$\mathbf{R}_q = D^p\,W^{p,o}P^{1\to m}\mathbf{R}_q \tag{2.26}$$

With these decoding weights fixed, the read-out with the matrix D^p enables us to interpret the momentary activity $\mathbf{x}^p(t_n)$ of neurons in the prediction error layer in terms of room labels; to see this compare the right-hand side of Eq. (2.26) with Eq. (2.25). Note that the decoding weights are an interpretation tool, but not implemented in the network (even though it would be easy to learn them, for example with the perceptron learning rule).

In order to interpret the *predicted* activity $\mathbb{E}_S[\hat{\mathbf{x}}^p(t_n)]$ in terms of room labels, we use the *same* decoding weights D as for the observed activity

$$\hat{\mathbf{R}}_{.|k(t_{n-1})} = D^p\,\underbrace{W^{p,b}P^{1\to m}\mathbf{R}_{k(t_{n-1})}}_{\mathbb{E}_S[\hat{\mathbf{x}}^p(t_n)]/(v\,l)}, \tag{2.27}$$

where $k(t_{n-1})$ is the index of the room during presentation step $n-1$ and $\hat{\mathbf{R}}_{.|k(t_{n-1})}$ is the prediction of rooms in step n, given room with index $k(t_{n-1})$ in step $n-1$. These predicted room labels enable not only the decoding of predictions in the figures of the results section but are also at the core of the following theorem.

Weights after convergence

We now show the following:

Theorem 1. For a large number of input neurons ($m \to \infty$), a small fixed learning rate $3^{rd} = \eta \ll 1$, presentation steps longer than the membrane time constant ($\Delta T \gg \tau$), and a large dwell time in a given apartment ($H \to 0$), the synaptic weights connecting the buffer population to the prediction error layer converge under the plasticity rule of Eq. (2.19) to a fixed point such that (if the input from the momentary observation is blocked) the activity of rectified linear neurons in the prediction error layer can be decoded as

$$\hat{\mathbf{R}}_{.|q} = \mathbb{E}_{tr}[\mathbf{R}_k|\mathbf{R}_q] = \sum_k \mathbf{R}_k T^*_{k,q} = T^*\mathbf{R}_q \tag{2.28}$$

where $\mathbb{E}_{tr}$ denotes expectations over transitions conditioned on the index q of the previous room.

Thus, given that the room in the previous time step t_{n-1} was $\mathbf{R}_q$, the predictive input from the buffer population can be decoded and represents the average of reachable next rooms; cf. Eq. (2.22)

Notes:

(i) For the situation with $K = 4$ transitions per room, the theorem implies that the network activity of the prediction error layer reflects all possible next rooms (with equal weights) if there is no input from the current observation.

(ii) The condition of a small and constant learning rate ensures a separation of time scales. If learning is slow enough to keep fluctuations of weights small, then learning becomes self-averaging after many presentation steps [164].

(iii) The condition of $m \to \infty$ where m is the number of neurons in the input layer coding for the same room ensures that fluctuations due to spikes, and in particular those correlations between input-and-output spikes that are not accounted for by correlations of firing rates, become negligible [131].

(iv) We only need to calculate the stationary state because for the plasticity rule of Eq. (2.19) the local stability of the stationary state is guaranteed by [95, 131].

Proof. According to Eq. (2.19) the update of the weights from an input neuron k to the set of neurons in population p is proportional to the product of the membrane potential and the postsynaptic current PSC(t) (EPSC or IPSC), so that at the end of a single presentation time step of duration $\Delta T \gg \tau$

$$\begin{aligned}\Delta \mathbf{w}_k^p(\Delta T) &= \eta \int_0^{\Delta T} dt' \mathbf{h}^p(t') \overline{\mathrm{PSC}}_k^b(t'),\\ &= \pm\eta \int_0^{\Delta T} dt' \int_{-\infty}^{t'} \mathrm{d}s\, e^{\frac{s-t}{\tau}} (\mathbf{x}^p(s) - \hat{\mathbf{x}}^p(s))\, \overline{\mathrm{PSC}}_k^b(t)\end{aligned} \tag{2.29}$$

where η is a small constant learning rate and $\overline{\mathrm{PSC}}_k$ are the filtered PSCs from the presynaptic neuron k. The plus sign applies to population $p = P_1$ and the minus sign to $p = P_2$. We exploited that the presentation time step (ΔT= 100 ms) is long compared to the membrane time constant τ so that the transients of neuronal activities after the transition between rooms can be neglected.

We study the network at the end of presentation step n and assume that during the previous presentation step $n-1$ the room $\mathbf{R}_q$ with index q was observed. There are two levels of stochasticity in equation (2.29), stochasticity of transitions and stochasticity of spike firing. We first take the average over the stochasticity of spiking $\mathbb{E}_S$, by taking the expectation over the Poisson distribution of input spikes

$$\mathbb{E}_S\left[\Delta \mathbf{w}_k^P \,|\, \mathbf{R}_q\right] = \pm\eta\, \mathbb{E}_S \left[\int_0^{\Delta T} dt' \int_{-\infty}^{t'} \mathrm{d}s\, e^{\frac{s-t'}{\tau}} ((\mathbf{x}^p(s) - \hat{\mathbf{x}}^p(s))\, \overline{\mathrm{PSC}}_k^b(t') \,|\, \mathbf{R}_q\right]. \tag{2.30}$$

Under the condition $m \to \infty$, correlations between input spikes and membrane potential can be neglected [131]. We can therefore separate the conditioned expectations into two independent terms and write

$$\mathbb{E}_S\left[\Delta \mathbf{w}_k^P \,|\, \mathbf{R}_q\right] = \pm\eta \int_0^{\Delta T} dt' \int_{-\infty}^{t'} \mathrm{d}s\, e^{\frac{s-t'}{\tau}} \left(\mathbb{E}_S\left[\mathbf{x}^p(s) \,|\, \mathbf{R}_q\right] - \mathbb{E}_S\left[\hat{\mathbf{x}}^p(s) \,|\, \mathbf{R}_q\right]\right) \mathbb{E}_S\left[\overline{\mathrm{PSC}}_k^b(t') \,|\, \mathbf{R}_q\right]. \tag{2.31}$$

We define the expected input current originating from neuron k of the buffer population as, $\mathbb{E}_S[\overline{\mathrm{PSC}}_{k|q}^b(t') \,|\, \mathbf{R}_q] = \overline{J}_k$ which is constant after an initial transient; this simplifies the notation of the last factor on the right-hand side of Eq. (2.31). Furthermore, we use Eqs. (2.24) and (2.25) to evaluate the two remaining expectations in Eq. (2.31).

$$\mathbb{E}_S\left[\Delta \mathbf{w}_k^P \,|\, \mathbf{R}_q\right] = \pm\eta \int_0^{\Delta T} dt' \int_{-\infty}^{t'} \mathrm{d}s\, e^{\frac{s-t'}{\tau}}\, \nu\, l \left(W^{p,o} P^{1\to m} \mathbf{R}_{j(t_n)\,|\,q} - W^{p,b} P^{1\to m} \mathbf{R}_q\right) \overline{J}_{k|q} \tag{2.32}$$

where $\mathbf{R}_{j(t_n)\,|\,q}$ is the room observed in presentation step n, given that room $\mathbf{R}_q$ was observed in step $n-1$. Note that the index $j(t_n)$ depends on the specific realization of the stochastic transition starting from room $\mathbf{R}_q$.

Exploiting that $H \to 0$, we now compute the average over a long observation sequence (expectation $\mathbb{E}_n$ over presentation steps t_n) in the same apartment. We can decompose this average into multiplication with the probability $P(q)$ of observing room $\mathbf{R}_q$ and the expected transitions $\mathbb{E}_{tr}[\mathbf{R}_{j(t_n)|q}]$ from room $\mathbf{R}_q$ to other rooms $\mathbf{R}_j$. We exploit that the rooms reachable from room $\mathbf{R}_q$ are given by the transition matrix T^*.

After convergence, the change of weight $\Delta\mathbf{w}_k^P$ averaged over many presentation steps and realizations of spike trains is zero. Hence, we will set $\mathbb{E}_n\left[\mathbb{E}_S\left[\Delta\mathbf{w}_k^P\right]|\mathbf{R}_q\right] = 0$. Since the filtering operations induced by the two integrations in Eq. (2.32) are linear, they yield a fixed factor which can—just like the fixed multiplicative parameters $\eta\, \nu\, l$ —be dropped after convergence. We exploit that the only term that depends on transitions is $\mathbf{R}_{j|q}$ so that we can pull the transition average inside and find

$$0 = \mathbb{E}_n\left[\mathbb{E}_S\left[\Delta\mathbf{w}_k^P\right]|\mathbf{R}_q\right] = \sum_{q=1}^{\mathscr{R}}\left(W^{p,o}P^{1\to m}\mathbb{E}_{tr}[\mathbf{R}_{j|q}] - W^{p,b}P^{1\to m}\mathbf{R}_q\right)\overline{J}_{k|q}\,P(q). \tag{2.33}$$

Using Eq. (2.22) we rewrite $\mathbb{E}_{tr}[\mathbf{R}_{j|q}] = T^*\mathbf{R}_q = \sum_j \mathbf{R}_j T^*_{j,q}$.

Since decoding is linear, stationary, and deterministic, we can multiply Eq. (2.33) with the decoding weights D^p from the left. From Eq. (2.26) and Eq. (2.27) we obtain

$$0 = \sum_{q=1}^{\mathscr{R}} P(q)\,\overline{J}_{k|q}\left(\sum_j \mathbf{R}_j T^*_{j,q} - \hat{\mathbf{R}}^p_{.|q}\right). \tag{2.34}$$

Note that because of the presynaptic factor proportional to $\overline{J}_{k|q}$, only those weights will be changed that receive input from a neuron k coding for room $\mathbf{R}_q$. However, since in the long sequence, all rooms appear with non-zero probability $P(q) = 1/K$, for each choice of k the synaptic input current $\overline{J}_{k|q}$ is non-zero during some presentation steps so that all weights are eventually adapted during the presentation sequence and the terms inside the parenthesis must be zero. Hence,

$$\hat{\mathbf{R}}^p_{.|q} = \sum_j \mathbf{R}_j T^*_{j,q}. \tag{2.35}$$

Eq. 2.35 shows that the synaptic rule with a fixed learning rate has a stationary solution, where the weight pattern predicts possible next rooms according to the probabilities of the transition matrix. This ends the proof. □

Notes:

(i) The stationary solution is locally stable both for the plastic excitatory weights in P_1 [131] and for the plastic inhibitory weights in P_2 [95].

(ii) In the proof, we decode rooms from the membrane potential of neurons in the prediction error layer. If neurons in the prediction error layer are rectified linear and the input from the observation pool is blocked, then their output is either zero or proportional to their potential. Neurons in at least one of the populations, P_1 or P_2, have a positive potential and can therefore be decoded.

(iii) The predictions reported in the Results section are the average across the readouts from two populations P_1 and P_2

$$\hat{\mathbf{R}}_q = \frac{1}{2}(\hat{\mathbf{R}}^{P_1}_{.|q} + \hat{\mathbf{R}}^{P_2}_{.|q}). \tag{2.36}$$

2.A.10 Predicted next stimuli with learning rate modulated by surprise

In the previous subsection, the learning rate was a constant, η whereas in our model the learning rate is modulated by the third fact $3^{rd}(\bar{A})$. Let us consider a transition from room q to one of the reachable rooms. If the reachable rooms have different transition probabilities, e.g, $T^*_{1,q} = 0.7$ and $T^*_{2,q} = T^*_{3,q} = T^*_{4,q} = 0.1$, then the transition to a room $j = 1$ is less surprising than the transition to one of the other available rooms. Since the amount of activity depends on the surprise level, the third factor will be a function of the room j that is reached from room q: $3^{rd}(\bar{A}) = 3^{rd}_{j|q}$. We need to include this dependence in our calculations and modify Eqs. (2.33) and (2.34) accordingly. Multiplication on the right-hand-side of Eq. (2.33) with $3^{rd}_{j|q}$ gives a weighted average

$$0 = \mathbb{E}_n\left[\mathbb{E}_S\left[\Delta\mathbf{w}^P_k\right]|\mathbf{R}_q\right] = \sum_{q=1}^{\mathscr{R}} \mathbb{E}_{tr}[3^{rd}_{j|q}\left(W^{p,o}P^{1\to m}\mathbf{R}_{j|q} - W^{p,b}P^{1\to m}\mathbf{R}_q\right)\bar{J}_{k|q}]P(q). \tag{2.37}$$

As before, we now use linear decoding weights

$$0 = \sum_{q=1}^{\mathscr{R}} P(q)\bar{J}_{k|q}\left(\sum_j 3^{rd}_{j|q}\mathbf{R}_j T^*_{j,q} - \sum_j 3^{rd}_{j|q}\hat{\mathbf{R}}^p_{.|q}\right). \tag{2.38}$$

which gives a weighted average for the predicted rooms,

$$\hat{\mathbf{R}}^p_{.|q} = \left(\sum_j 3^{rd}_{j|q}\mathbf{R}_j T^*_{j,q}\right)/(\sum_j 3^{rd}_{j|q}) \tag{2.39}$$

Thus, for surprise-modulated learning rate and inhomogeneous transition probabilities, the code of the predicted rooms does not correctly reflect the statistical weights, and rare transitions are slightly amplified.

2.A.11 Population activity in prediction error layer represents the present state and consistent alternatives to the present state, given the previous state

To illustrate the theoretical results in simulations, we run the volatile sequence task without re-occurrence of rules (Fig 2.1 **A**) with $\mathscr{R} = 16$ different stimuli ($K = 4$) for 3000 presentation steps in a SpikeSuM network. In the beginning, the spike pattern across the population of pyramidal neurons in the prediction error layer looks noisy (**Supplementary Fig. S2.B.2** middle left), but after a few hundred presentation steps with a fixed transition rule $T^*_{k,q}$, the prediction error layer exhibits four active groups of neurons. These four groups represent the four possible transitions predicted from the *previous* stimulus, including the currently observed one (**Supplementary Fig. S2.B.2** middle, second panel). Note that the predictions from the memory buffer of the previous stimulus excite neurons in population P_1, whereas the current stimulus mainly excites neurons in population P_2 (Fig. 2.2). Therefore, with $K = 4$ possible transitions, the currently observed stimulus is represented by a single group of neurons in population P_2 whereas the three alternatives are consistent with the previous stimulus by groups of neurons in population P_1. Thus, for the SpikeSuM network with hand-wired connectivity, the activity in the prediction error layers reflects a column of the transition matrix $T^*_{k,q}$ where the fixed q denotes the previous stimulus (stored in the buffer) and the index k runs over the groups of neurons in the prediction error layer coding for stimulus k. Immediately after a switch to a new rule, a fifth cluster of active pyramidal neurons is observed. The five clusters correspond to the four wrong predictions (that have been learned with the previous rule and now give rise to negative prediction errors) and the currently observed (unexpected) state under the new rule (which gives rise to a 'positive prediction error', in the sense that the current sensory input is stronger than the prediction [60]).

2.A.12 One-shot learning after a rule switch

As predicted by the theory in the previous paragraphs, the learned weights implicitly encode the full transition matrix of the current rule (**Supplementary Fig. S2.B.2** bottom). The switch between rules causes a large activity (**Supplementary Fig. S2.B.2**, top). In turn, the large activity induces an increase of the neuromodulatory surprise signal $3^{rd}(\bar{A})$ that leads to a fast update of the weights. Hence, the new transition appears in the transition matrix already at the end of the first presentation step, i.e., after only 100ms (red circle in the graph). Thus, a single novel transition is sufficient to change the matrix (learning in 'one shot'). After spending some time with stimulus presentations under the new rule, the activity $\bar{A}$ of PT-neurons returns to a low value and the new transition matrix can be extracted from the weights onto pyramidal neurons in populations P_1 and P_2 (**Supplementary Fig. S2.B.2**, right, labeled 4).

2.A.13 Benchmark algorithms

We compare the results of our network to several state-of-the-art algorithms (Fig. 2.3). For fairness of comparison, each of these algorithms uses surprise-based online adaptation to detect change points induced by rule switching. BOCPA [75] is a Bayesian online algorithm for exact inference of the most recent change point. It is a message-passing algorithm that infers the probability distribution over the run time since the last switch. It is known to be optimal on average for long simulations. VarSMiLe [32] is a Variational approximation [165] of BOCPA that uses the Bayes Factor surprise S_{BF} [32] to detect change points. VarSMiLe does not need message-passing (as implemented in BOCPA) and has a closed-form update rule similar to the SMiLe rule[72].

We also compare SpikeSuM with networks of the same architecture but with a simplified function for the third factor. The function 3^{rd}, introduced in Eq. (2.3), scales the amount of plasticity. The first line in the following equation gives the definition, while the next ones the simplifications considered

$$\begin{aligned}
3^{rd}(\bar{A}) &= \eta_1 \tanh(\bar{A})\Theta(\bar{A}) + \eta_2 \tanh(\bar{A})\Theta(\bar{A}-\theta), && \text{for SpikeSuM} \\
3^{rd}(\bar{A}) &= \eta_1 \tanh(\bar{A}), && \text{for SNN}_{sm} \\
3^{rd}(\bar{A}) &= \eta_1 && \text{for SNN}_{nm}\ .
\end{aligned} \tag{2.40}$$

This comparison of 3^{rd} factors allows us to investigate the impact of modulation on learning. The differences in networks parameters can be found in table 2.A.1

2.A.14 Simulation parameters and comparison of algorithms

The results are obtained by running networks composed of, $8\times\mathscr{R}$, presynaptic neurons (so that 8 neurons have sustained spiking for each stimulus), and 128 postsynaptic neurons (256 for random connectivity). The presynaptic neurons have a firing rate of 100Hz if representing the observed stimulus, and the squared EPSCs last for 4ms. The integration time of the input potential $\tau = 10ms$. See the summary in table 2.A.1

Table 2.A.1: Networks parameters for simulations with $\mathscr{R}$ stimuli.

	Presynaptic neurons	Postsynaptic neurons	3^{rd} hyperparameters
SpikeSuM	$8\times\mathscr{R}$	128	3
SpikeSuM$_{rand}$	$8\times\mathscr{R}$	256	3
SNN$_{sm}$	$8\times\mathscr{R}$	128	1
SNN$_{nm}$	$8\times\mathscr{R}$	128	1

Hyperparameters (η_1, η_2, θ) of SpikeSuM as well those for SMiLe and BOCPA have been optimized using the python library *scikit-optimize* [166] minimizing $\sum_n ||T_n - \hat{T}_n||_F^2$, where T_n is the transition matrix of the apartment at time step t and $\hat{T}_n$ the estimated transition matrix extracted from the weights and $||.||_F$ the Frobenius norm.

2.A.15 Context Modules: architecture of SpikeSuM-C

SpikeSuM-C is an extension of the original SpikeSuM network and is composed of M SpikeSuM modules (Fig. 2.5). Each context selection module (CSM) m outputs (in its second layer, L2) an inhibitory current

$$Inh^m = J \sum_{m' \neq m} \sum_j \mathrm{EPSC}^{m'}_{j,L2}, \tag{2.41}$$

inhibiting the PT neurons,

$$\tau_{\bar{A}^m} \frac{d\bar{A}^m}{dt} = -\bar{A}^m + c \sum_{p=P_1^m, P_2^m} \sum_i \mathrm{EPSC}_i^p - Inh^m. \tag{2.42}$$

The sum run over all neurons j of all CSM other than the one with index m. We chose J=20 for strong inhibition. This way if m is the active module $3^{rd}(\bar{A}^m) \geq 0$ and $3^{rd}(\bar{A}^{m'}) = 0$.

The neurons in SpikeSuM module m are updated following

$$\begin{aligned} \Delta w_{ik}^m &= -3^{rd}(\bar{A}^m)\, h_i^m\, \overline{\mathrm{EPSC}}_k^m \quad \text{in population } P_1, \\ \Delta w_{ik}^m &= 3^{rd}(\bar{A}^m)\, h_i^m\, \overline{\mathrm{EPSC}}_k^m \quad \text{in population } P_2. \end{aligned} \tag{2.43}$$

for each SpikeSuM module. 3^{rd} is the same as in Eq. (2.14).

The selection of the learning module is done in the CSM network (Fig. 2.A.2) which has a Best-Prediction-Learns (BPL) dynamics. It is a combination of dis-inhibitory feedforward connectivity within a module and lateral Winner-Take-ALL (WTA) dynamics between modules. Each module contains two layers of N_{CSM} spiking neurons each, described by the Spike Response Model in 2.A.4.
The first layer (L1) of CSM m receives excitatory input from SpikeSuM module m and sends inhibitory input to the second layer (L2). L2 is a WTA circuit where the least inhibited module stays selected, whereas the others are silenced. The overall network architecture has a BPL-dynamics since the least active SpikeSuM module (i.e., the best prediction) is chosen by the CSM network as the learning module. The inputs to neurons in layers L1 and L2 are

$$\begin{aligned} I_{\mathrm{L1},i}^m &= a_1 \left[\sum_k W_{i,k}^{FFp,m} \mathrm{EPSC}_k^m - \beta \right], \\ I_{\mathrm{L2},i}^m &= a_2 \left[-a_3 \sum_k W_{ik}^{Inh,m} \mathrm{EPSC}_{\mathrm{L1},k}^m - a_4 \sum_{m' \neq m} \sum_k W_{ik}^{WTA,m'} \mathrm{EPSC}_{\mathrm{L2},k}^{m'} + a_5 I_{\mathrm{ext}} \right]. \end{aligned} \tag{2.44}$$

where a_1, a_2, a_3, a_4, a_5 and β are fixed positive parameters. Here, a_3 governs the overall strength of the feedforward dis-inhibition drive from the SpikeSuM module. The connection strength a_4 controls the strength of lateral inhibition in the WTA circuit. $I_{\mathrm{ext}} = \sum_m \sum_k \mathrm{EPSC}_k^m$, the sum over all possible spikes in the prediction error populations (and all context). I_{ext}^m drives the necessary amount of excitation so that the WTA dynamics take place and account for the variability in the network activity, in other words, this noise-dependent input allows stabilizing learning and adapting the WTA threshold. In fact, if the overall activity of the prediction error layer increases due to randomness (not misprediction) in the network, the threshold of the WTA would also increase. The parameter a_2 is a scaling parameter that we found useful in setting up the simulations. The choice of parameters is discussed below in **SpikeSuM-C parameters**.

While the weights $(W^{FF})^{p,m}$ and $W^{WTA,m'}$ are fixed at a value of one, the connections $W_{ik}^{Inh,m}$ from L1 to L2 are plastic. The inhibitory connections $W_{ik}^{Inh,m}$ are potentiated by a Hebbian rule modulated by a third factor and depressed by an unspecific decay term with decay rate, α

$$\Delta W_{ik}^{Inh,m} = \eta\, g(3^{rd,m})\, \mathrm{EPSC}_{L2,i}^m\, \mathrm{EPSC}_{L1,k}^m - \alpha(W_{ik}^{Inh,m} - W_0^{Inh}). \tag{2.45}$$

We call these weights the confidence weights. Indeed, $\eta \mathrm{EPSC}^m_{L2,i}\mathrm{EPSC}^m_{L1,k} > 0$ drives the potentiating as long as neurons in both L1 and L2 are active; i.e., when the module m is selected. During this phase, we consider that the observer builds their confidence up.
After learning a task for some time, $W^{Inh,m}_{ik} \gg W^{Inh}_0$ its initial value.
Suppose now that suddenly a higher prediction error in SpikeSuM module m occurs. This causes an increase of activity in L1 and leads to strong inhibition (because the weights had been potentiated earlier) of neurons in L2 so that the WTA mechanism rapidly 'unselects' this module in favor of another one. Note that postsynaptic neurons in L2 of a CSM that lost the WTA competition are silenced so that connections onto these neurons are no longer potentiated. The net result of the plasticity rule is that modules that have never been chosen in the past have connection weights that are still close to their initial values—and these modules can then be later selected by the WTA dynamics for new tasks. The function $g(3^{rd,m})$ in Eq. 2.45 allows influencing the direction of learning. For the simulations we use, $g(3^{rd,m}) = \big(1-2\Theta(\bar{A}-\theta)\big)\eta$ where θ is the surprise threshold of SpikeSuM so that weights are depressed during phases of surprise and potentiated otherwise. Finally $\alpha(W^{Inh,m}_{ik} - W^{Inh}_0)$, $\alpha \ll 1$, implements weight decay on a long timescale so that a module that is not used for a long time is slowly forgotten. Note that we could also add a similar decay term to the standard SpikeSuM learning rule in the prediction error layer, to allow the weights to be slowly forgotten.

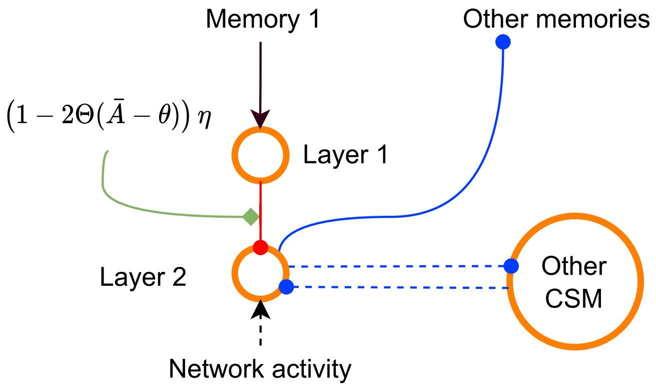


Figure 2.A.2: **Context selector module (CSM).** Each CSM is composed of two layers of inhibitory neurons. Layer 1 receives excitatory input from the corresponding SpikeSuM module. Layer 2 receives inhibition from layer 1 and lateral inhibition from layer 2 of other CSMs. Because of the inhibitory connection from layer 1 to layer 1, the more excitation a CSM receives, the lower the activity in layer 2. Because of WTA dynamics implemented by lateral inhibition, the CSM module with the lowest excitation is selected, inhibits other CSM, and shuts down the plasticity of other SpikeSuM modules. Red weights are plastic and can be interpreted as a 'commitment' of the selected module. The network activity is driven by all SpikeSuM modules and supports the WTA dynamic.

2.A.16 Systematic results for SpikeSuM-C

To systematically evaluate the performance of the SpikeSuM-C network, we ran a series of simulations for three volatilities H = (0.002,0.001,0.0005). We ran 100 simulations that run over 10 000 presentations, each with 4 randomly drawn apartments and $K = 2$ (or 4) transitions per room. To describe the efficiency of the network we use four different measures of performance:

1. Detection success rate: Number of simulation runs (out of 100) that detect *all* switch points within less than 50 steps and that made less than 10 context switches not linked to apartment switching.
2. Module usage success rate: the match $\frac{\sum_i [N^i_{used} == N^i_{observed}]}{100}$ between the number of context modules used in simulation run i, N^i_{used}, and the number of apartments observed $N^i_{observed}$.
3. Complete success rate: Number of simulation runs (out of 100) that have detected all switch points in less than 50 steps and that fulfill $N^i_{used} == N^i_{observed}$.
4. Detection time: mean number of presentation steps between a switch to a different apartment in the input sequence and a switch to a different memory module in the model network.

Note that a simulation where a network uses more memory modules than the objective number of apartments does not necessarily imply a complete failure. Indeed, one can conceive that an observer splits a single apartment into two different memories due to the unsupervised nature of the task. This is why we consider the results section the detection success rate as the most valuable measure (it is not so important how many memories are used as long as one can detect the switch points).

For sequences that switch apartments on average every 2000 time steps ($\frac{1}{H}$=2000), the switch detection success rate is 100 percent for apartments with both $K = 2$ and 4 transitions, but it is slightly lower for simulation paradigms with more frequent context switches (Tables 2.A.2a and 2.A.2b). We expected that for $K = 8$ transitions the performance breaks down since it implies that from each room and in each apartment 8 out of 16 possible transitions are allowed so that a large fraction of transitions is compatible with several apartments. This was indeed the case (results not shown). However, for $K = 2$ already, the first or second transition after a switch between apartments is with high probability a good indicator of the switch. Indeed, for $K = 2$ the SpikeSuM-C model detects switches within less than three presentation steps if switches occur on average every 1000 or 2000 time steps (Table 2.A.2a).

2.A.17 SpikeSuM-C parameters

The results are obtained by running networks with the parameters summarized in tables 2.A.3 and 2.A.4. The presynaptic neurons have a firing rate of 100Hz if representing the

(a) Results for 2 transitions apartment

	Detection success rate	module usage success rate	Complete success rate	Detection time
H: 0.002	95.0 ± 2.2	90.0 ± 3.2	90.0± 3.2	1.4 ± 0.2
H: 0.001	100.0 ± 0.0	93.0 ± 2.6	93.0 ± 2.6	2.7 ± 0.2
H: 0.0005	100.0 ± 0.0	96.0 ± 2.0	96.0 ± 2.0	2.3 ± 0.2

(b) Results for 4 transitions apartment

	Detection success rate	module usage success rate	Complete success rate	Detection time
H: 0.002	74.0 ± 4.4	59.0 ± 4.9	48.0 ± 5.0	5.7 ± 0.7
H: 0.001	98.0 ± 1.4	91.0 ± 2.9	91.0 ± 2.9	6.1 ± 0.5
H: 0.0005	100.0 ± 0.0	92.0 ± 2.72	92.0 ± 2.72	7.3 ± 0.83

Table 2.A.2: Summary of SpikeSuM-C accuracy on the re-occurrence-based task with different stochasticities and volatilities.

observed room, and the squared EPSCs last for 4ms. The integration time of the input potential, $\tau = 10$ms. The code is available on GitHub (https://github.com/martinbarry59/SpikeSuMNet).

Context selection module parameters	1/H = 500	1/H = 1000	1/H = 2000
a_1	0.256	0.22	0.22
a_2	0.05	0.05	0.05
a_3	4	4	4
a_4, a_5	20	20	20
α	1e-06	1e-06	1e-06
β	0.045	0.045	0.045
η	0.0003	0.00025	0.0003
$W_{max}^{Inh,m}$	0.11	0.08	0.09
SpikeSuM parameters			
η_1	1e-05	1e-05	1e-05
η_2	0.005	0.005	0.005
θ	0.45	0.45	0.45
W_{init}	60	60	60
c	40	40	40
EI-neurons	128	128	128

Table 2.A.3: Summary of SpikeSuM-C parameters for results in table 2.A.2a

Context selection module parameters	1/H = 500	1/H = 1000	1/H = 2000
a_1	0.07	0.07	0.07
a_2	0.05	0.05	0.05
a_3	4	4	4
a_4, a_5	20	20	20
α	1e-06	1e-06	1e-06
β	0.055	0.055	0.055
η	0.0008	0.0007	0.0006
$W_{max}^{Inh,m}$	0.1	0.11	0.13
SpikeSuM parameters			
η_1	1e-06	1e-06	1e-06
η_2	0.002	0.002	0.002
θ	0.5	0.5	0.5
W_{init}	9	9	9
c	40	40	40
EI-neurons	512	512	512

Table 2.A.4: Summary of SpikeSuM-C parameters for results in table 2.A.2b

2.B Supplementary figures

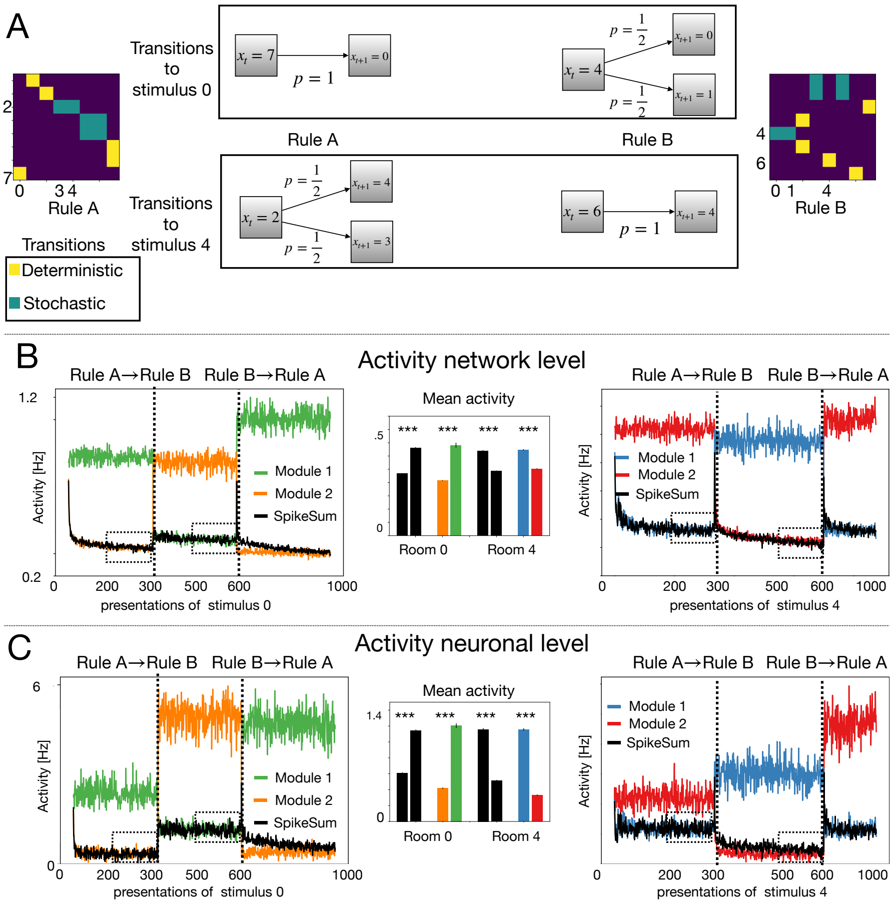


Figure 2.B.1: **Deterministic transitions have a different signature than stochastic ones**. The paradigm uses a volatile sequence task with re-occurrence of rules but restricted to $\mathscr{R} = 8$ different (auditory or visual) stimuli and two different transition rules (A and B) and could be tested in rodent experiments. **A** Transition matrix corresponding to rule A (left) and rule B (right). The transition to stimulus '0', $T_{7\to 0} = 1$, is deterministic (yellow square, lower left corner) under rule A, but stochastic with a value of $T_{4\to 0} = 0.5$ (light blue) under rule B, and vice versa for the transitions to stimulus '4'. **B** Population activity averaged over network neurons in populations P1 and P2 during all presentations of stimulus $x_{t+1} = 0$ (left) or $x_{t+1} = 4$ (right). Black lines: SpikeSuM without context. Green/blue lines: population of neurons in module 1 of SpikeSuM-C (i.e., the module responding to rule A). Orange/red lines: population of neurons in module 2 (i.e., responding to rule B) of SpikeSuM-C. Horizontal axis: count of occurrences of the stimulus '0' or '4', respectively. Inset, middle: histogram of average activity after 200 presentation time steps under a given rule. Black bars: comparison of activity under rules A and B in SpikeSuM without context. Colored bars: The activity of neurons in module 1 of SpikeSuM-C during stimuli under rule A is compared with that of neurons in module 2 during stimuli under rule B. A stochastic transition causes more activity than a deterministic one. **C**, same as in **B**, but only the activity of those neurons responsive to stimulus '0' (left) or '4' (right) is shown. In contrast to the simple SpikeSuM network without context, neurons in the SpikeSuM-C network that respond to stimulus '4' in module 1 under rule A (blue) respond even stronger in the context of rule B but this does not affect their plasticity. Thus, the same experimental paradigm also differentiates between models with and without context modules

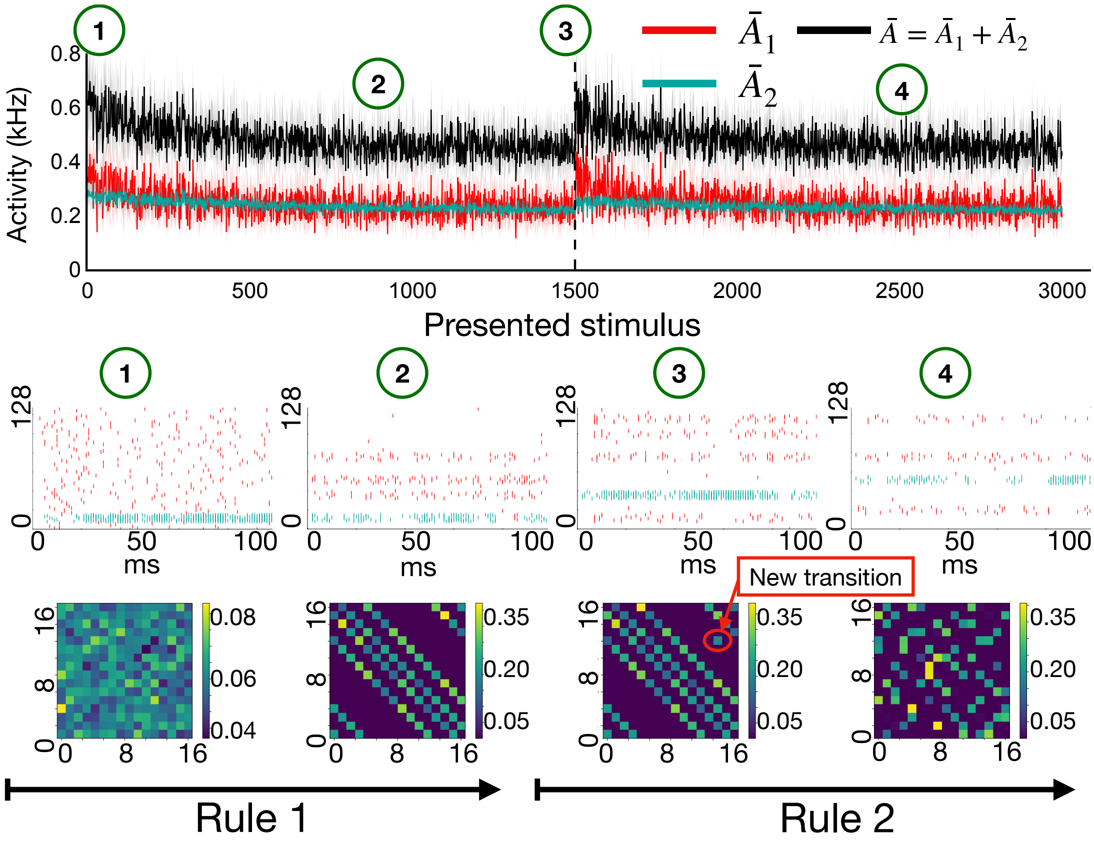


Figure 2.B.2: **Mixed response profile where neuronal responses depend on the present state, the previous state, and consistent alternatives to the present state** in a task with $\mathscr{R} = 16$ stimuli and $K = 4$ transitions possibilities and two rules. **Top**: Activity (arbitrary units) of populations P_1 (green) and P_2 (red) as well as the total activity $\bar{A}$ (black) of all pyramidal neurons. After 1500 presentation steps, the transition rule switches from rule 1 to rule 2. Each presentation step corresponds to the exposure to one stimulus for 100ms. **Middle**: Spike trains of pyramidal neurons during one presentation step, at different points during learning (from left to right): at the beginning and end of the first episode with rule 1 and the beginning and end of the first episode with rule 2. Initially, (label 1) all pyramidal neurons have approximately the same activity, whereas after 1000 presentation steps (label 2) only a few pyramidal neurons are active, i.e., those that correspond to the four possible next states reachable from the previously observed state. Out of the four reachable states, one is presently observed, and, since the observation is stronger than the prediction, neurons in population P_2 fire (blue dots) whereas for the three other predicted states the observation is weaker than the prediction so that neurons in population P_1 fire (red dots). After the rule change (label 3), a fifth cluster of neurons is activated corresponding to the presently observed, but unpredicted, state visible as spikes in population P_2. Pyramidal neurons (16 per state, 8 neurons each from P_1 and P_2) have been sorted according to state numbers for visual clarity. **Bottom**. Matrix of transitions between states decoded from the weights onto pyramidal neurons. At the end of the first presentation step after a change point (label 3), a new element (red arrow) has appeared in the transition matrix corresponding to the newly observed transition, $R_{n-1} \rightarrow R_n$. After some time with the novel rule, the new transition matrix is learned (label 4) and the old one is suppressed.

Supplementary Discussion

We comment on the relations of our approach to Artificial Neural Network approaches in neuroscience.

Predictions and Model-Based Reinforcement learning

In our model, the surprise signal arises from a mismatch between the representation of the present state and predictions based on the previous state. These learned predictions could be used in several ways. First, the learned predictions generate a synaptic transition matrix that reflects the temporal neighborhood relations of the input stimuli ('states'). These neighborhood relations correspond to the adjacency matrix of states in an abstract or, in the case of apartments, a two-dimensional space. Thus, our approach learns an autoencoder of transitions rather than states[167]. Second, in the absence of input from the next state (e.g., while closing the eyes and waiting), the predictions could be used by an agent to 'imagine' the next state—and, if this imagined next state is fed back into the network as surrogate input, it would become possible to plan forward in time over several states, similar to forward replay observed in animals and humans [168, 169] and used in model-based reinforcement learning [120]. Interestingly, since the speed of forward replay would be given by the intrinsic dynamics of the network when closing the loop, it is faster than the original experience [168, 169].

The unsupervised paradigm for training SpikeSuM modules can be complemented by a reward-based paradigm[12], for example in an actor-critic network[120]. Even though detecting surprise requires a model of possible transitions, recent evidence from human behavior suggests that surprise is mainly used to detect rule changes. In contrast, reward-based learning in complex environments is dominated by model-free reinforcement learning[52].

Representation of States and Artificial Neural Networks

In the model in the main text, the representation of states is simple and predefined. However, we view our surprise-detection networks as modules that can be integrated into bigger networks able to solve more complicated tasks. For example, the input layer of our SpikeSuM-C network can be replaced by the output of a model of the ventral stream[170]. Moreover, the backprop learning rules of a multi-layer artificial network could be replaced by local predictive learning rules[105], and SpikeSuM modules can be used in each layer of the network (area of the processing stream) to detect area-specific surprise and locally modulate plasticity.

Contribution

In this chapter, we addressed the questions of how to measure surprise in a biologically constrained neural network and how to utilize it for learning. Unlike prior research, we use a network architecture and dynamics where high neuronal activity serves as a natural measure of surprise, rather than relying on an explicitly defined surprise signal.

In order to extract a surprise signal, we developed a paradigm that is in line with experimental findings. Our approach involved the implementation of positive and negative prediction error neuron populations, which sought to balance excitatory and inhibitory inputs, akin to putative L2/3 neurons. Notably, the design of these two-population prediction errors (which were already hypothesized in Keller et al.[60]) was influenced by the biological constraints of our model, specifically the asymmetry in neuronal activity between an over-inhibited and over-excited population. The activity of these populations is then integrated by projection neurons, which estimate the total prediction error and act as putative L5 neurons. This integrated activity is then sent to the deeper locus coeruleus area to estimate the surprise scenario, where we use a two-neuromodulator model (acetylcholine and norepinephrine) to distinguish between intrinsic volatility of the environment and structural changes for computing surprise. This two-neuromodulator approach allows us to reach Bayesian performance in our scenario. Finally, the complete surprise signal is used as a neuromodulator for learning, guiding the speed of adaptation of our error populations. Unlike many other models, SpikeSuM is fully self-supervised and does not focus on reward-based learning that would ultimately interfere with the surprise signal. To our knowledge, there are no other existing surprise-based spiking models working on fully local three-factor learning rules capable of extracting Surprise and learning from it. We also are the first to implement with biological constraints the two-neuromodulator model introduced in Yu et al.[17].

In addition to this new model of surprise, we realized that the model could be extended to avoid forgetting the previous environment at every structural change. Our approach of module-based surprise and dis-inhibitory networks for self-supervised detection of context changes and reactivation has shown great potential. Context-based learning is becoming increasingly popular in the neuroscience community, and we hope that our novel model serves as an inspiration for follow-up experimental studies, such as the existence of dis-inhibitory networks and activity-driven context detection in possibly L5 neurons, and the existence of local inhibition of neuromodulation.

Contribution

This chapter emphasizes the crucial role of multiple time scales for adaptation. SpikeSuM integrates 1ms spike signals over 100ms windows to detect and adapt to unexpected inputs rapidly and efficiently. Moreover, SpikeSuM features a context detector that leverages long-term integrated confidence in the environment, enabling the creation and learning of models over longer time scales, up to minutes (hundreds to thousands of image presentations). By harnessing these capabilities, SpikeSuM can learn and adapt to varying time scales of input, underscoring its versatility and efficacy as a learning system.

3 GateON: an unsupervised method for large scale continual learning

GateON: an unsupervised method for large scale continual learning

Authors: *Martin Barry, Guillaume Bellec, Wulfram Gerstner*
Contribution: All theories and implementations have been made by Martin Barry. The writing has been shared across authors.

3.1 Introduction

Artificial neural networks (ANNs) have made significant strides in recent years, showcasing remarkable capabilities that surpass human performance in several domains. However, traditional neural network models (multi-layer perceptrons[171], LSTM[172], CNN[173], Transformers[2]), unlike humans, are plagued by a major drawback – the inability to continually integrate new data without overwriting previously acquired knowledge[174]. This phenomenon, commonly known as catastrophic forgetting, raises a major challenge for real-life applications such as training robots or understanding human development[175–178]. Unlike traditional ANNs, humans can easily generalize their internal models to new paradigms (forward transfer[179]), update previous knowledge (backward transfer[180]), and prevent catastrophic forgetting. These remarkable capabilities have intrigued the community of theoretical neuroscience[33, 111, 112, 181–183] as well as that of Machine Learning (ML)[184–187].

An ideal continual learning (CL) method would leverage prior knowledge acquired during a first task for new task acquisition (forward transfer). Forward transfer results in accelerated training and enhanced generalization, since prior knowledge from a first task acts as a useful initialization for the following tasks. It has been suggested[188] that forward transfer could (at least partially) address the problem of small or biased datasets, that pose challenges for traditional ML paradigms. Forward transfer during CL is similar in spirit to adapting pre-trained models, such as Bert or ResNet50[189, 190], to novel tasks. In addition to forward transfer, CL could also cause positive backward transfer across similar tasks, i.e., knowledge gained in later tasks could potentially facilitate improvement of the model on the original task, without reactivation of stimuli of the original task.

There are three primary types of continual learning methods, summarized in Kaushik et al.[191]. The first type is the replay-based method[184, 192–194], where the algorithm intelligently stores crucial samples in memory for reuse in later training epochs. The second type is the regularizer-based method[112, 195, 196], where the loss function or gradient is adjusted to prevent deviation from the solution obtained in previous tasks. The third type is formed by parameter isolation models[185, 197–199], which protect a subset of parameters to prevent forgetting in subsequent tasks. Our method, though closer to parameter-isolation models, also shares aspects with regularizer-based methods. Our method avoids using replay because replay is expensive in terms of memory since it requires to keep an increasing amount of data in a buffer when going from a few to a hundred tasks.

Contributions. We present a novel CL method (called GateON) and test it on established tasks for CL, such as permuted MNIST and language tasks with a pre-trained BERT model[189]. We show that

- GateON is compatible with various neural network architectures, including fully connected networks, convolutional neural networks (CNNs), and transformers.
- GateON combines aspects of regularization-based and isolation-based methods by monitoring the relevance of each parameter (or each neuron) and adjusting the learning rate on a parameter-specific basis. Importantly, the learning rate can decrease or increase, which prevents the 'freezing' of parameters that are never re-used.
- GateON does not require separate inputs to indicate task switching but detects task switches automatically.
- Going beyond the standard CL paradigm (limited to 10 tasks), GateON also works across 100 MNIST CL tasks. It also surpasses the State-of-the-Art for CL language tasks with pre-trained BERT with a measurable forward and backward transfer of knowledge, outperforming recent state-of-the-art approaches, such as Ramesh et al.[184], Ke et al.[200], and Sun et al.[186].

Related work

Our approach uses online relevance estimation, in combination with gating and unsupervised context detection.

(i) *relevance estimation* is used in several other studies, either in combination with gradient obstruction (e.g., HAT[185], and RMN[199]) or in combination with regularization to prevent large parameter drifts (EWC[112], SI[111]). HAT uses an attention mask as relevance mapping, which may overestimate the true importance of a parameter, while RMN uses an estimation of relevance learned during training. By contrast, we compute online an analytical estimate of the impact of each parameter on the loss of the *current* task which defines the instantaneous relevance of a parameter.

(ii) *Leaky availability mapping.* In contrast to existing availability control methods which fixate neurons of high relevance [185, 199], GateON integrates the relevance gradually into an availability mapping, which determines which parameters are allowed to update and which are not. A key advantage of our design over Hat or RMN [185, 199] is that it enables the introduction of a meta-parameter, denoted by ϵ, that controls the reactivation rate of parameters that have not been relevant for a long time while hedging against catastrophic forgetting.

(iii) *Gating approaches*, such as HAT[185] and Active dendrites[182], have been successful in the field of continual learning. In contrast to HAT[185] which employs a binary attention mechanism for gating and a multi-head output, our approach gates every unit (or parameter) with a task-specific control variable and does not require a multi-head output. Our gating approach shares similarities with the learnable context masking in the Active dendrites model [182], but in contrast to their approach, we do not enforce a strict number of active neurons per context nor do we need to estimate a task embedding.

(iv) Whereas standard CL models give explicit information on task numbers[199], we introduce a simple context detector that enables unsupervised inference of tasks. While the fields of machine learning and signal processing have incorporated context switches via out-of-distribution detection[201–205], our proposed detector is, in the context of our CL tasks, computationally cheap since it focuses on deviations in the loss that is evaluated anyway for each sample.

3.2 The GateON Model

In this section, we explain the GateON method for fully-connected networks (FC) but the GateON method can also be used for CNNs (**Appendix** 3.A.6). We study CL with a sequence of tasks $1 \leq t \leq T$ where tasks are also called 'context'.

Network architecture and contextual gating We study a network of L layers. To simplify notation, all hidden layers are assumed to have N neurons. For the activity of a neuron i in layer l, we use the notation

$$x_i^l = g_i^l \cdot f(\sum_j w_{ij}^l x_j^{(l-1)}), \quad \text{with } g_i^l = \sigma(\sum_j w_{ij}^{g,l} C_j), \tag{3.1}$$

where w_{ij}^l are the connection weights from layer $l-1$ to l and $w_{ij}^{g,l}$ the context weights to layer l. The input is denoted by $x_i^{(0)}$ and the output by $y_i = x_i^L$. Contextual gating, denoted by g_i^l, modulates the output activity x_i^l of a neuron i and σ is a rectified hyperbolic tangent, so that $g_i^l(t) \in [0,1]$. Each context $\vec{C}$ is a one-hot encoded vector (Fig. 3.1) where the vector contains (at least) as many elements as the number of tasks. GateON can work in two modes: either the task ID is given in the dataset, otherwise, we provide in (**Appendix** 3.A.3) an algorithm for change point detection. In this unsupervised mode (task ID is not given), the algorithm can also reactivate previously encountered contexts (aka tasks). The gating mechanism learns through backprop which neurons should participate in the feedforward processing network and which should not. While the learning rate of the feedforward weights w_{ij}^l is modulated by the availability variable (see next paragraph), the learning rate of the contextual weights $w_{ik}^{g,l}$ is not.

Relevance mapping and gradient obstruction Parameter isolation methods[197, 206] demonstrate that fixing a neuron's representation during learning can prevent catastrophic forgetting. However, fixing all weights would prevent future training on later tasks and is similar to setting the learning rate of gradient descent to 0. To circumvent this issue, GateON uses a 'flexible isolation' by online detection of relevance. Here, relevance can be defined at two different levels: either the relevance of each specific parameter (i.e., weights and biases) or the relevance of a neuron as a whole (i.e., all its input weights and bias together). In both cases, a high relevance turns down the parameter-specific learning rate ('gradient obstruction') and eventually fixes the corresponding parameter. We call the two variants parameter GateON (p-GateON) and neuron GateON (n-GateON) respectively.

For the rest of the chapter, θ denotes a parameter (feedforward weights and biases) if we use p-GateON or the neuronal activity x_i^l for n-GateON. We measure the relevance of θ as the mean effect that it has on the loss function for the current task t

$$\mu_\theta = (\mathscr{L}_\theta - \mathscr{L})^2, \tag{3.2}$$

where $\mathscr{L}_\theta$ denotes the loss for $\theta = 0$ in the forward path. Since this term is computationally expensive, we assume θ small and approximate the relevance by its first-order Taylor expansion around θ

$$\mu_\theta \approx \left(\frac{\partial \mathscr{L}}{\partial \theta}\theta\right)^2, \tag{3.3}$$

details in **Appendix** 3.A.1. The vector $\boldsymbol{\mu}$ denotes the relevance of all parameters (with the loss evaluated for the current batch of data) and is called the **relevance mapping**. We normalize the relevance $\mu_\theta \rightarrow \mu_\theta^{norm}$ so that the average is 1 and that $0 \leq \mu_\theta^{norm} \leq N'$ where N' is the number of parameters (or neurons) per hidden layer (**Appendix** 3.A.1).

The gradient is flexibly obstructed by modulating the learning rate with the *availability* parameter $A_\theta \geq 0$. After each batch update, we estimate the relevance mapping and integrate it in an **availability mapping**:

$$A_\theta(t+1) = Clip[A_\theta(t)(1 - \eta_A(\mu_\theta^{norm} - \epsilon))], \tag{3.4}$$

where $A_\theta(t)$ denotes the availability of the parameter θ at time t, and η_A and ϵ are non-negative constants. The function $Clip[x] = x$ for $0 < x < 1$ and $Clip[x] = 1$ for $x \geq 1$ is clipping $A_\theta(t)$ in the range $[0,1]$. Eq. 3.4 has two stable fixed points, at $A_\theta = 0$ and $A_\theta = 1$. For a relevance $\mu_\theta > \epsilon$, the availability decays exponentially to zero, and for $\mu_\theta < \epsilon$ it increases exponentially until it stops at $A_\theta = 1$. Hence, ϵ acts as a threshold to separate relevant from non-relevant parameters. $A_\theta(t)$ together with $\epsilon > 0$, define a **leaky availability mapping**.

Finally, gradient updates are controlled with a parameter-specific learning rate that is controlled by the availability,

$$\Delta w_{ij}^l = \begin{cases} \eta A_{w_{ij}^l} \frac{\partial \mathscr{L}}{\partial w_{ij}^l}, & \text{if p-GateON} \\ \eta A_{x_i^l} \frac{\partial \mathscr{L}}{\partial w_{ij}^l}, & \text{if n-GateON} \end{cases} \tag{3.5}$$

where η is the nominal learning rate (SGD, Adam, ...). As shown in Eq. (3.5), the smaller

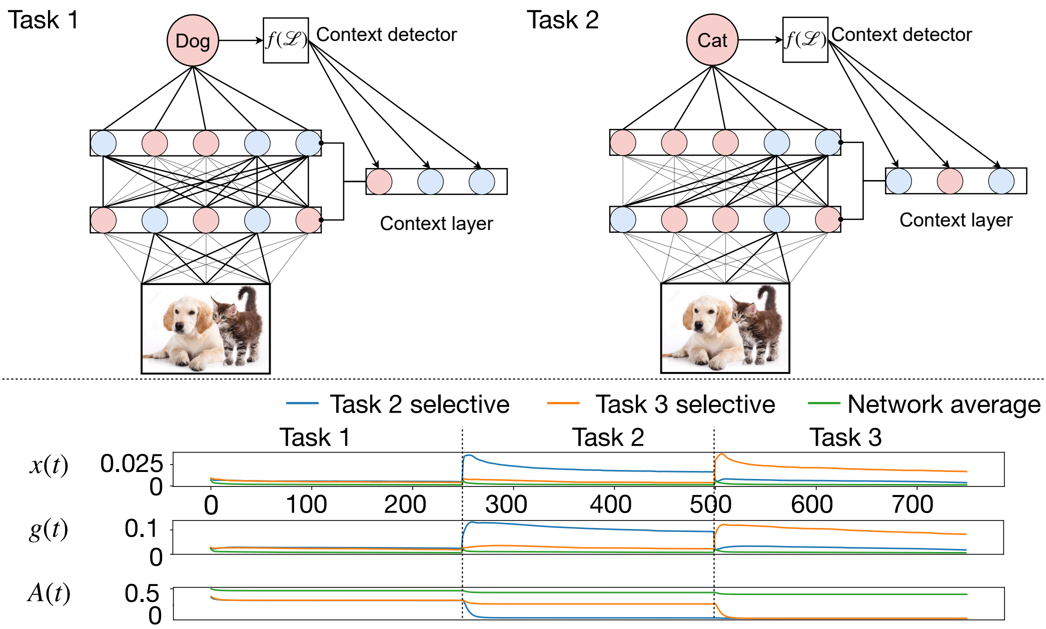


Figure 3.1: **Top**: Schematic. GateON during 2 successive tasks. Assume task 1 and task 2 classify what is on the left or right side of the picture, respectively. Blue neurons in the network are silenced by the context layer, and red ones are active. Dashed connections are connections with low availability by the end of the task. Neurons can be shared across tasks, and neurons with low availability keep low availability for the following task. **Bottom**: Toy simulation across three permuted MNIST tasks. The activity $\mathbf{x}$, gating $\mathbf{g}$, and availability $\mathbf{A}$ evolve over batch updates for three consecutive tasks, each lasting 250 batch presentations. Blue and yellow lines indicate an average over the subset of neurons with mean activity higher than a threshold during tasks 2 and 3, respectively. Neurons that are selective for task 2 (or 3) have a higher gating value during task 2 (or 3) than the network average (green). The availability of neurons selective for task 2 (or 3) decreases towards zero during task 2 (or 3). Note that the average availability of neurons remains high (green) while the availability of neurons selective for task 2 remains low during the presentation of task 3 showing that only a few neurons are considered relevant for the task.

the availability A^l_θ, the larger the obstruction of the gradient. In other words, our method regularizes learning speed.

We will show that the hyperparameter ϵ balances the trade-off between preserving representations ($\epsilon \to 0$) and maintaining the flexibility of the network for future tasks. In n-GateON learning is obstructed on a per-neuron basis, i.e., N parameters per layer. This method is therefore coarse compared to p-GateON ($N' = O(N^2)$ parameters per layer) which enables a more fine-grained obstruction. The consequences of these choices will be analyzed in the results section. Figure 3.1 bottom illustrates an example of the availability of task-selective neurons in an n-GateON network across three tasks. We discuss the network inference of the context in **additional figure** 3.A.1.

3.3 Simulations

In this study, we employ four distinct metrics to differentiate between various aspects of continual learning. The *immediate test accuracy (A_{cc}^{t})*, measures the test accuracy on a single task t immediately after training this task. The *accuracy deviation (ΔA_{cc}^{t})*, computes the relative difference between the immediate test accuracy of GateON and the accuracy of task-specific isolated networks, averaged over all tasks (a positive ΔA_{cc}^{t} implies forward transfer). The *continual accuracy ($A_{cc}^{cont,t}$)*, measures the averaged test accuracy of a task t from the end of its training until the completion of all tasks. The *forgetting rate (FR^{t})*, quantifies the difference between the immediate test accuracy and the continual accuracy averaged only on $t' > t$ (excluding the immediate test accuracy). A $FR^{t} < 0$ implies that on average, previously trained tasks increase their test accuracy more than they forget, showing positive backward transfer. All these metrics M^{t} averaged over all tasks are denoted M (e.g., $A_{cc} = \frac{1}{T}\sum_{t} A_{cc}^{t}$), we only report the average in the following.

Continual MNIST datasets Our investigation into the properties of the GateON method involves its application to networks trained on four CL problems constructed on the MNIST dataset. The first three have been defined previously: Permuted MNIST[207] (*Permuted*), Rotated MNIST[208] (*Rotated*), Split MNIST[209] (*Split*); and we add a fourth one: Shuffled-label MNIST task (*Shuffled*). For each problem, the network receives the image pixels as inputs and outputs a digit label. Then for each problem, we apply a change in the input or output convention after each task change, i.e., the input image (or output) is either rotated or shuffled. In *Permuted*, the input pixels are randomly permuted, thereby rendering it impossible for a classical fully-connected network to generalize from the previous task. The tasks in *Permuted*are orthogonal to each other, which means that knowledge gained from one task cannot be used to perform well on any other task. On the other hand, in *Rotated*, if the rotation angle between two tasks is small, the network might be able to generalize using its learned feature space. However, correlations could cause negative interference, making *Rotated* more challenging to learn than *Permuted* for some methods[210, 211]. *Split* consists in separating the classes present in the dataset and presenting these sub-datasets sequentially. For MNIST instead of showing all digits, we split the dataset into five tasks each containing two random digits. *Split* is interesting as it impacts the output. A limitation of the existing 'Split' approach for MNIST is that you can only get 5 tasks with non-overlapping labels. This is why we also introduce *Shuffled*, which replaces *Split* when working on 100 tasks. In *Shuffled* each new task randomly shuffles the labels. This task is particularly challenging for classical networks (without multi-head) as it completely overrides in the final layer the previous knowledge at each new task. Together with *Rotated*, *Permuted*and *Split*, this collection of problems enables us to evaluate three key properties of the GateON method: its ability to prevent catastrophic forgetting (measured by *FR* and A_{cc}^{cont}), its saturation limit that eventually prevents learning new tasks sequentially (A_{cc} and A_{cc}^{cont}), and its ability in forward (ΔA_{cc}) and backward transfer (*FR*).

Table 3.1: Continual accuracy results (A_{cc}^{cont}) for the three classical CL-tasks (*Permuted*, *Rotated*, and *Split* MNIST) over 10 tasks. We compare the performance of our model with other 'online' models (i.e., no replay), such as RMN, EWC, OGDT, and HAT. We also select state-of-the-art methods with replay (e.g., Zoo, Active dendrite), indicated by $^{\mathscr{R}}$. GateON 0 refers to our model with $\epsilon = 0$. Isolated is the averaged accuracy for 10 vanilla networks trained on each task separately. Methods that use multi-head outputs are marked by $^{\mathscr{H}}$. The values in bold indicate the best performance. The *Rotated*value for Zoo is in parentheses because they used only 5 fixed small rotations as opposed to random ones.

Methods	*Permuted*	*Rotated*	*Split*
OGDT$^{\mathscr{H}}$ [212]	86.44	88.32	98.84
EWC[112]	96.9	84	-
HAT$^{\mathscr{H}}$[111]	**98.6**	-	99.0
RMN[199]	97.73	-	99.5
n-GateON 0 (OURS)	97.8	99.2	**99.98**
p-GateON 0 (OURS)	97.29	99.27	**99.98**
Active dendrite$^{\mathscr{R}}$[182]	97.2	-	-
Zoo$^{\mathscr{R}}$ [184]	97.71	(**99.66**)	99.97
Isolated	98.0	99.4	99.99

3.3.1 Classical continual learning

Most of the literature study only *continual accuracy* after CL on 10 tasks. To compare to the state-of-the-art, we first limit to classic CL with 10 tasks before studying large-scale CL with 100 tasks. The architectures of our networks are detailed in **Appendix** (3.A.11).

We show in Tab. 3.1 CL results for 10 MNIST tasks. We observe that GateON performs well across *all* three classic continual tasks, as it is close to the performance of 'Isolated' models, optimized for each task separately. Moreover, it beats the state-of-the-art for *Split*. On *Permuted*and *Rotated*, GateON is the best, or amongst the best *online* methods (no replay) of all 15 online models reported in Ramesh et al.[184]. The only exception is HAT, on *Permuted*. On *Permuted*, HAT performs even better than our 'Isolated' model, which is usually considered a baseline for *Permuted*. This performance might come from the multi-head output applied to a common representation in the previous layer. Notably, Zoo, the only model that beats GateON on *Rotated*, uses Replay and reports only results on 5 fixed, and relatively small rotations which might make transfer across tasks easier.

3.3.2 Large-scale Continual learning.

Although our method has achieved state-of-the-art results on classical CL, we aim to show that there is a network saturation problem that emerges on long-term CL (up to 100 MNIST tasks). This problem has not been addressed in previous literature and is typical for parameter-fixing methods.

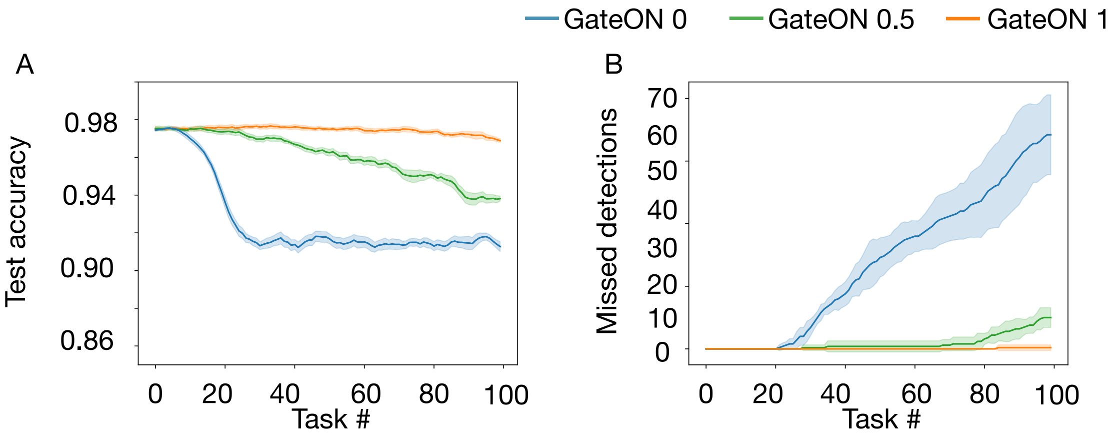


Figure 3.2: **Network saturation**. **A**: Immediate test accuracy across tasks for *Permuted*on n-GateON (with task identity given, not inferred). The test accuracy shows that the network saturates after about 10 - 20 tasks in its learning performance for $\epsilon \to 0$ (blue). Yet, for GateON 1, this saturation does not occur. **B**: Missed detections of tasks switches for n-GateON with task-inference activated. Comparing **A** with **B** shows a clear correlation between the saturation in immediate test accuracy and the missed detections. Missed detections strongly harm CL. The number following GateON stands for the ϵ used.

Saturation when scaling to 100 tasks. Parameter-fixing models can quickly reach *saturation*, meaning that they are unable to learn anything new once all their parameters are fixed. To address this issue, GateON uses a leaky availability mapping, Eq. (3.4). In our analysis, we focus specifically on n-GateON, as it saturates much faster than p-GateON. Fig. 3.2A shows the evolution of *immediate test accuracy* over the sequential training of 100 tasks, revealing a distinct drop in accuracy after 10 tasks for n-GateON 0 (where 0 stands for $epsilon$ = 0), consistent with the idea that strict fixing of parameters ($\epsilon = 0$) limits network capacity. However, we observe that this issue is mitigated when ϵ is set to 0.5 or 1. Moreover, Fig. 3.2B highlights that saturation significantly impairs the model's capacity to detect task switches. This incapacity of detecting switches worsens the CL capacity of our method. However, we find that saturation does not occur for more correlated tasks (*Rotated*, *Shuffled*), which is discussed in more detail in **Appendix** 3.A.8. These results suggest that saturation induced by parameter-fixing presents a critical limitation, which the leaky availability mapping helps to address, preventing network fixation and incapacity of context detection.

Continual learning across 100 tasks. Although we showed that a leaky availability mapping prevents saturation, the 'un-fixing' of parameters with an $\epsilon > 0$ might harm the CL performance of GateON when tested across different tasks. To answer this question, we turn now to learning 100 MNIST tasks across three types of task variants (the *Split* task has been replaced by *Shuffled*). Going to 100 tasks has rarely been attempted before (Table 3.2). We compare the results of GateON with four other models: Vanilla, a classically trained ANN; the Active Dendrites[182] algorithm as a baseline (data only available for *Permuted*); and two ablation models named Gating and Obstruction. The ablated model 'Gating' employs contextual gating

(Eq. 3.1) but without obstruction (Eq. 3.5); on the contrary, 'Obstruction' does not use gating. The results of Table 3.2 show that both gating and obstruction are important components of our model, and none of them works on their own.

Studying A_{cc}^{cont} across the three task variations indicates that the traditional fully-connected model (Vanilla) underperforms for all tasks, a phenomenon previously observed in studies on catastrophic forgetting[176, 213]. The GateON method leads to a significant increase in mean continual accuracy across all competing models and all tasks. Both n-GateON (with $\epsilon \geq 0.5$) and p-GateON (for arbitrary ϵ) surpass Active Dendrites[182] by a large margin.

For n-GateON, a parameter choice $\epsilon \geq 0.5$ positively impacts continual accuracy, making it a valuable tool for large-scale continual learning. While there were only small differences between n- and p-GateON for their *best* results, $\epsilon \to 0$ affects n-GateON more strongly than p-GateON. Our interpretation is that the parameter ϵ is less important for p-GateON since the latter makes better use of the parameters and probably does not yet reach saturation for 100 tasks. For the sake of completeness, we also report the results for GateON fed with explicit context information in Supplementary 3.B, which sheds light on the impact of context inference on the final results.

For the *Permuted*MNIST task, ΔA_{cc}^{t} is negative for all GateON models, indicating the absence of forward transfer. However, for the two other CL MNIST variants, we find positive ΔA_{cc}^{t} for all models, suggesting that our model does promote slight forward transfer. Since the parameter *FR* is always positive we conclude that there is no backward transfer on the MNIST tasks. However, p-GateON demonstrates a significant resilience against forgetting, with all *FRs* below 1% across all 100 tasks. Note that for MNIST, the amount of training data is large. We anticipate that backward transfer may be more relevant in tasks with limited training data.

Our findings demonstrate excellent performance for both n-GateON and p-GateON. The use of a leaky availability map enables the resolution of network saturation while maintaining exceptional continual learning performance. To understand the performance, we have mathematically analyzed forgetting in the leaky availability map: the forgetting time for highly relevant neurons is much longer than their learning time (see **Appendix** 3.A.2) whereas less relevant neurons learn as fast as they forget. This ensures that the relevant neurons do not forget the first task over time, while other neurons become reusable, thereby significantly increasing the network saturation limit.

3.3.3 Continual learning with large language models

To evaluate the GateON method on interesting large datasets, we conducted tests on three natural language processing (NLP) continual learning (CL) tasks: Aspect Sentiment Classification (ASC)[214], Document Sentiment Classification (DSC)[215], and Text Classification using 20News data[216] (20News).

Table 3.2: Performance results on *Permuted*, *Rotated*, and *Shuffled* **for 100 tasks**. Results show that a leaky relevance mapping drastically improves the CL performance of n-GateON on *Permuted*. Results for GateON with CNNs can be found in Supplementary 3.B. The values in bold indicate the best performance. For A_{cc}^{cont}, green shows all the values that are within 1% of the best performance. For ΔA_{cc} and *FR* the green values show forward (positive) and backward (negative) transfer, respectively. On MNIST we do not observe backward transfer (no green).

	A_{cc}^{cont}			ΔA_{cc}			*FR*		
Models	*Permuted*	*Rotated*	*Shuffled*	*Permuted*	*Rotated*	*Shuffled*	*Permuted*	*Rotated*	*Shuffled*
Vanilla	26.29	69.68	13.41	-	-	28.53	71.57	28.53	77.44
Active dendrites$^{\mathscr{R}}$[182]	91.2	-	-	-	-	-	-	-	-
Obstruction	18.69	48.29	11.15	-34.15	-63.85	-13.67	45.82	29.36	72.16
Gating	55.07	71.63	62.07	**0.24**	**0.94**	**0.6**	41.96	25.33	35.11
n-GateON 0	73.64	97.62	**97.72**	-5.61	0.59	0.26	14.82	0.35	**0.12**
n-GateON 0.5	92.86	97.6	97.35	-1.57	0.66	0.39	2.06	0.33	0.62
n-GateON 1	95.75	97.7	96.89	-0.48	0.7	0.4	1.12	0.24	1.09
p-GateON 0	96.34	**97.87**	97.69	-0.6	0.64	0.24	**0.48**	**0.14**	0.13
p-GateON 0.5	**96.48**	97.76	97.51	-0.05	0.55	0.29	0.84	0.16	0.37

The ASC CL task involves sentiment analysis on product reviews, with the network required to classify each aspect of the product, such as the picture or sound, as positive, negative, or neutral. The CL task includes 19 products, with each product considered a separate task. The aspect term and sentence are concatenated using the [SEP] token, and the [CLS] token is used to predict the opinion polarity. For example, the network receives "[CLS] torchlight [SEP] this is a good torchlight" and should output 'positive' for the aspect torchlight. Some reviews can have multiple aspects. *The DSC CL task* requires sentiment analysis on the full customer review, with the network tasked with classifying the overall tone as positive or negative. The CL task comprises 10 products (tasks), and the input comprises the token [CLS] followed by the text review. The first output token is used as the sentiment estimate. *The split 20News CL task* is a topic classification task on the 20News dataset. The original task has 20 topics, which were randomly split into ten tasks, each containing two topics. The input consists of the token [CLS] followed by text. The first output token is used as the topic estimate.

All these are CL-tasks, meaning that their tasks are shown sequentially without recall. ASC and DSC are similar to *Permuted* and *Rotated* in the sense that each task has the same classification objectives but with different datasets, while split-20News changes also the labels similarly as for split-MNIST. We use both A_{cc}^{cont} and Macro-F1 (MF1) score for comparison. MF1 is the F1 score averaged over the different tasks and is more relevant than accuracy on biased datasets such as *ASC*. To solve these tasks we use one of the most used large language models, pre-trained Bert[189] to which we applied the GateON method on each module, attention, intermediate, output, for all 12 layers as well as pooling and embedding.

GateON performance on large language models Remarkably n-GateON emerges as a versatile method that achieves state-of-the-art or near state-of-the-art for all tasks (including other models in Ke et al.[200]) see Tab. 3.3. Additionally, our consistent use of the same parameters across tasks demonstrates its adaptability. Only LAMOL significantly outperforms it for the

Table 3.3: The table presented above compares the results for the NLP continual learning CL-tasks. Multi-task learning (MTL) is not considered as continual learning since it addresses all tasks simultaneously and can be used as a maximum baseline. We compare our results with the best models currently known. The GateON method appears to be competitive with the state-of-the-art and outperforms all other known models summarized in[200]. The values in bold indicate the best performance.

Models	ASC		DSC		20News	
	A_{cc}^{cont}	MF1	A_{cc}^{cont}	MF1	A_{cc}^{cont}	MF1
BERT MTL (max baseline)	91.91	88.11	89.77	89.28	96.77	96.77
BERT naive	49.60	43.08	73.08	71.81	52.50	39.22
HAT[111]	86.74	78.16	87.29	86.14	93.51	92.93
LAMOL[186]	88.91	80.59	**92.11**	**91.72**	66.13	45.74
B-CL[187]	88.29	81.40	79.76	76.51	95.07	95.04
CTR[200]	**89.47**	83.62	89.31	88.75	95.25	95.23
n-GateON 0	89.35	**84.11**	89.29	88.79	95.85	95.86
p-GateON 0	83.09	76.71	87.1	85.34	**95.93**	**95.96**

DSC CL task but it performs poorly on the other tasks. It is worth noting that both LAMOL and CTR are designed for BERT models and are not generalizable to other architectures. p-GateON achieves a relatively low performance on ASC and DSC in comparison with n-GateON. We believe that the parameter approach might require more data to be able to identify and tune all relevant parameters individually. Additionally, unlike n-GateON which fixes all the parameters into the neuron p-GateON only fixes the relevant ones. Since BERT is very sensitive to slight changes in representation, the difference between p-GateON and n-GateON might have a big impact.

Forward and backward transfer The NLP datasets exhibit a wide range of sizes, with some tasks characterized by limited and biased datasets. We report the ΔA_{cc}^{t} to study forward transfer in Tab. 3.4. Both models on all tasks show positive deviations emphasizing the method's capacity for forward transfer. It is most obvious in ASC and DSC, reaching 6.3% for ASC's MF1 for n-GateON, which is understandable since ASC possesses small and biased datasets. The detailed values of *FR* are reported in table 3.5. Remarkably, n-GateON FR scores on *ASC* are negative, suggesting that the network not only retains high accuracy levels but also enhances the performance of previous tasks through backward transfer. p-GateON appears to have more difficulties on these datasets showing a significant forgetting rate explaining its low performance. The dataset Split 20news has consistent dataset sizes and highly independent tasks, which contribute to a reduced impact on both forward and backward transfer (small ΔA_{cc}^{t} and positive FR) for both n- and p-GateON.

Table 3.4: **Forward transfer in large language model**. The deviation accuracy (ΔA_{cc}) and MF1 deviation ($\Delta MF1$) are the performance improvement against the baseline where tasks are trained in isolation. All tasks show systematic positive deviation implying a forward transfer of knowledge for both n- and p-GateON. With an MF1 deviation up to 6.3% for ASC. ASC comprises many small and biased datasets that seem to profit from this knowledge transfer.

Models	ASC		DSC		20News	
	ΔAcc	$\Delta MF1$	ΔA_{cc}	$\Delta MF1$	ΔA_{cc}	$\Delta MF1$
n-GateON	3.52	6.28	2.34	2.99	0.16	0.17
p-GateON	2.24	1.99	2.78	3.17	0.12	0.13

Table 3.5: **Backward transfer in large language model**. The forgetting rate (FR) is a measure of the difference in accuracy between a model's performance immediately after training on a particular task and its performance after training on all subsequent tasks. For ASC FR is negative, showing that on average previous tasks learn more than they forget, showing the backward transfer capacity of n-GateON. In contrast, the p-GateON model has generally high FR values, which may explain its poor performance.

Models	ASC		DSC		20News	
	$FR_{A_{cc}}$	FR_{MF1}	$FR_{A_{cc}}$	FR_{MF1}	$FR_{A_{cc}}$	FR_{MF1}
n-GateON	-0.21	-0.09	-0.02	0.03	0.08	0.08
p-GateON	2.33	1.16	1.56	2.32	0.12	0.12

3.4 Discussion

In conclusion, the GateON method is a powerful continual learning approach that can be applied to many types of neural networks by incorporating a context-gating layer and gradient flow obstruction. Unlike previous models [111, 185, 199], we showed that our model is general enough to be used either with the parameter or neuron relevance mappings. With minimal algorithmic complexity, the method can prevent catastrophic forgetting by analytically computing the relevance mapping of each neuron online and adjusting their learning rate.

On MNIST CL tasks p-GateON performed better than n-GateON, because of the large amount of available training data. On the other hand, n-GateON performed exceptionally well on NLP continual learning tasks using pre-trained transformers. It has also shown positive forward transfer capacity for all tasks. Our experiments on the ASC dataset revealed a negative forgetting rate, indicating that the method possesses backward transfer capabilities for incomplete datasets. In the optic of a complete approach, we also implemented a computationally cheap task detection algorithm allowing for unsupervised context switches.

Overall, the GateON method has the potential to revolutionize the field of continual learning, particularly in areas such as robotics[217], where the ability to generalize from just a few real-world examples is crucial; or NLP where new datasets arise that might not be available at training time. As an online method, unlike replay-based methods, GateON does not require memory of previous datasets and avoids catastrophic remembering[218]. Our method resonates with the idea of preserving neuron selectivity[219, 220] in neuroscience, as well as possibly explaining paradigms such as representational drifts[221]. Indicating that our method could have broad applications.

Appendix

3.A Methods

3.A.1 Relevance mapping

In order to implement the GateON approach, it is necessary to compute the relevance μ_θ^l of the parameter (or neuron) θ on output loss. We define an analytical expression of the relevance μ_θ as the distance between the loss and the loss if the parameter theta were to be removed:

$$\mu_\theta = (\mathscr{L}_\theta - \mathscr{L})^2 \tag{3.6}$$

To simplify the computation of this term, we use the Taylor expansion around $\theta = 0$:

$$\mu_\theta \approx \left(\frac{\partial \mathscr{L}}{\partial \theta}\theta\right)^2. \tag{3.7}$$

To avoid scaling issues we normalize μ_θ with respect to the number of parameters N (or neurons) within the layer

$$\mu_\theta^{norm} = \frac{\mu_\theta}{\frac{1}{N}\sum_\theta \mu_\theta}. \tag{3.8}$$

The higher the causal effect the more important the neuron is to the task. Hence, we use this causal effect for availability adaptation

$$A_\theta(t+1) = Clip\left(A_\theta(t) \cdot (1 - \eta_A(\mu_\theta^{norm} - \epsilon))\right). \tag{3.9}$$

With $Clip$ as saturating function $Clip(x) = x$ if $x \in [0,1], Clip(x) = 0$ if $x \leq 0$ and $Clip(x) = 1$ otherwise. $Clip$ clips the availability A_θ between 0 and 1. The term ϵ allows neurons to recover availability over time if their causal effect decreases.

3.A.2 Availability forgetting time

An important theoretical question is how the introduction of the threshold ϵ in equation (3.9) impacts the temporal evolution of the availability. Indeed, with $\epsilon = 0$, the availability A can only decrease or stay fixed. To study the evolution of A for $\epsilon > 0$, we consider the following toy scenario: we assume that a given neuron is relevant for the first T update steps (T could, for example, represent all updates during the first task) where relevance is indicated by a high level $\mu = a > \epsilon$; and thereafter the same neuron is irrelevant for the next T' steps, as indicated by $\mu = b < \epsilon$ (T' could represent all updates during all other tasks). From our assumptions and Eq. (3.9) with $A(0) = A_0$ we have :

$$A(T+T') = A_0(1 - \eta_A(a-\epsilon))^T(1 - \eta_A(b-\epsilon))^{T'}. \tag{3.10}$$

We ask ourselves how big T' needs to be for the availability to return to its original value. Solving $A(T+T') = A_0$ we obtain

$$T^{*} = -T\frac{\log\big(1-\eta_A(a-\epsilon)\big)}{\log\big(1-\eta_A(b-\epsilon)\big)}. \tag{3.11}$$

Eq. (3.11) shows that the bigger the relevance a, the longer it will take to grow back to A_0. For the further interpretation, we suppose now that $\epsilon = 1$. Since $0 < b < \epsilon$, we have $\epsilon - b < 1$ during the 'irrelevant' phase. The relevance a, however, is bounded by the number of neurons N per layer (for n-GateON). Hence, during the 'relevant' phase of a neuron, the difference $a-\epsilon$ can take a value much larger than 1. The behavior described by Eq. (3.11) is exactly what we expect from the threshold term. It shows that very relevant neurons ($a \gg \epsilon$) have $T^{*} \gg T$ meaning that, for relevant neurons, the time scale of forgetting is much slower than the time scale of learning.

3.A.3 Task inference

In this section, we present an unsupervised algorithm capable of identifying change points and inferring the prevailing context (i.e., the current task) for the network. While our approach employs a single context for the entire network, it is adaptable for application to individual network layers. Initially, we define the concept of network confidence in the current context, prior to addressing change-point detection. Denoted as confidence_k, the confidence quantifies the network's level of expertise regarding context k. This confidence is updated as follows:

$$\Delta\text{confidence}_k = \text{confidence}_k + \eta_C C_k, \tag{3.12}$$

where η is a hyperparameter and C_k is a one-hot neuron layer that represents whether the context k is active during the backward step. In other words, the confidence over the context k increases if the said context is active during the backward pass.

Next, we describe the change-point detection algorithm in ALG. 1.

1. In line 11, we compare the minimum loss over the last k steps to a low-pass filtered loss. A change point is detected if the error exceeds a predefined threshold Θ and the network exhibits sufficient confidence; otherwise, the network remains in its current context. We compare to the minimum of the last k steps to be sure that the current loss is not an outlier. The larger k the longer to detect a change-point but the more sure we get.

2. Upon detecting a change point, lines 12-18 assess if a prior context aligns with the present observation. If such alignment is found, the previous context is reactivated; if not, a new context is generated.

3. This procedure is carried out for every batch step.

We detect change points by comparing the current loss to the average one in this context.

Algorithm 1 Change-point detection algorithm for GateON

```
procedure CPD
    Change-point ← $false$
    $\Theta, \eta, k$ ← set hyper parameters
    active-context ← 0
    Confidence(active-context) ← 0
    $N_{contexts}$ ← 1
    for data in datasets do
        $x^0 \leftarrow data$
        $\mathscr{L}^t \leftarrow forward(x^0,$ active-context$)$                // forward pass network
        $\bar{\mathscr{L}} = \bar{\mathscr{L}} + \eta_L(-\bar{\mathscr{L}} + \mathscr{L}^t)$
        if $\min_{t-k\le t'\le t} \mathscr{L}^{t'} > \Theta\cdot\bar{\mathscr{L}}$ and Confidence(active-context) $> 0.9$ then
            reactivated ← false                // Trying to reactivate previous context
            for test_context in $N_{contexts}$ do
                $\mathscr{L}^t \leftarrow forward(x^0,$ test_context$)$
                if $\mathscr{L}^t < \Theta\cdot\bar{\mathscr{L}}$ then
                    active-context ← test_context
                    reactivated ← True
                    Break
        if not reactivated then                // if no previous context suitable
            $N_{contexts} \leftarrow N_{contexts} + 1$
            active-context ←$N_{contexts}$
            Confidence(active-context)← 0
        Change-point ← false
        Confidence(active-context) ←Confidence(active-context) $(1-\eta_C)+\eta_C$
        Continue the training steps...
```

The use of inferred context can have benefits, as it allows us to set our tolerance for change points. For instance, in the case of *Rotated*, the representational drift depends on the angle between task k and k', which means that we can choose a tolerance Θ at which we consider the tasks to be similar or dissimilar. In Fig. 3.A.1 we study the properties of the task detection with MNIST CL tasks.

3.A.4 Intuition behind the GateON method

We know that the feedforward activity of the network if context k' is active follows

$$\begin{aligned} x^l_{i,k'} &= \sigma(w^{g,l}_{ik'}) \cdot f(\sum_j w^l_{ij} x^{(l-1)}_j + b_i) \\ &= g^l_i(k') \cdot f_i(x^{(l-1)}) \end{aligned} \tag{3.13}$$

In this equation, the input to the network is influenced by two factors: the task-driven input $g^l_i(k')$ and the data-driven one $f_i(x^{(l-1)})$. The data-driven input determines the representation of a neuron given the data, whereas the task-driven input determines the average

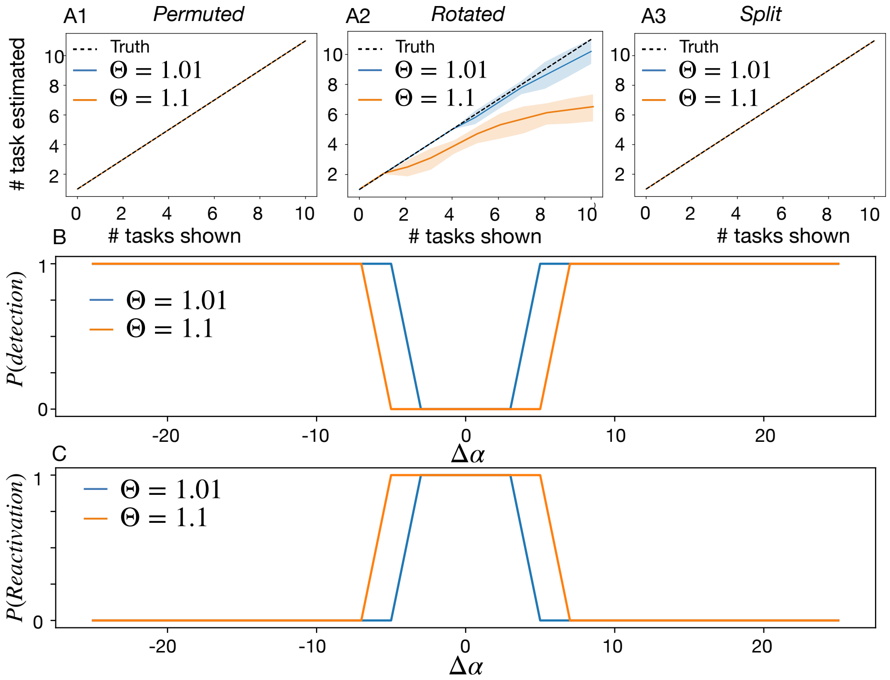


Figure 3.A.1: **Context Detection in MNIST**. **A1-3** We executed 10 continual learning (CL) tasks and tracked the number of tasks identified by GateON compared to the number of tasks presented. The results are plotted for two Θ values, where Θ modulates task detection stringency as explained in Algorithm (1). The dashed black line represents perfect task estimation accuracy. Results are averaged over 10 trials, with standard deviation indicated by the shaded region. **A1** and **A3** exhibit parity between the number of tasks presented and detected for both Θ parameters. In contrast, **A2** reveals that for *Rotated*, GateON underestimates the actual task count, with larger Θ values leading to fewer tasks detected. Specifically, tasks with closely spaced angles may be misconstrued as a single task, leading to context collapse. **B** To elucidate the reasons for the incomplete task detection in **A3**, we executed two consecutive *Rotated*tasks with an angle difference $\Delta\alpha \in [-25, 25]$, and charted the switch-point detection probability for GateON (averaged over 10 trials for each $\Delta\alpha$). The results indicate a binary behavior: if $\Delta\alpha$ exceeds a certain threshold, tasks are distinctly identified with a probability of 1; otherwise, tasks are amalgamated with equal certainty. This suggests a 100 percent switch-point detection rate if the angle difference is sufficiently large; otherwise, GateON perceives the tasks as identical. To confirm that similar tasks can reactivate prior ones, we ran in **C** a sequence of three tasks such that angles 1 and 3 differ by $\Delta\alpha$ and angle 2 is always far from the other two to be detected as switch-point. **C** demonstrates that the angle pairs leading to context collapse in **B** will also trigger reactivation, while those that resulted in distinct tasks do not reactivate. In summary, our context detector appears proficient at identifying switch-points between significantly dissimilar tasks and can also reactivate or collapse tasks when they are sufficiently alike.

representation of the neuron over the task.

Our objective is to explore methods for preserving the results of a trained task. To answer this we study the variation of the activity of a neuron i before (time t) and after update (time $t+1$), expressed as $x^l_{i,k'}$ and $x^l_{i,k(t+1)}$, respectively, with k' the active context and $k(t+1)$ is the active context at time $t+1$. In the following, we denote the variations of the parameters $\Delta\theta$, with $\theta \in \{w_{ij}, b_i\} \, \forall i, j$.

$$\begin{aligned}
\Delta x^l_i &= x^l_{i,k(t+1)} - x^l_{i,k'} \\
&= g^l_i(k(t+1)) \cdot f_i(x^{(l-1)}(t+1)) - g^l_i(k') \cdot f_i(x^{(l-1)}) \\
&\underset{\text{Taylor}}{\simeq} {\color{red} \Delta w^{g,l}_{ik(t+1)} (\sum_j \Delta w^l_{ij} + \Delta b^l_i) f'_i(x^{(l-1)}) g'(k(t+1))} \\
&{\color{red} + \Delta w^{g,l}_{ik(t+1)} f_i(x^{(l-1)}) g'(k(t+1))} \\
&+ (\sum_j \Delta w^l_{ij} + \Delta b^l_i) \cdot f'_i(x^{(l-1)}) g(k(t+1)).
\end{aligned} \tag{3.14}$$

In Eq. (3.14) the black part comes from the update of the forward weights, whereas the red one is from the update of the context weights, hence only impacting activity if k' is active. From a catastrophic forgetting point of view, if we want to assess the impact of the weight updates on the representation of a context $k(t+1) \neq k'$, $\Delta w^{g,l}_{ik(t+1)} = 0$ because it was not the active context, Eq. (3.14) reduces to:

$$\Delta x^l_i = (\sum_j \Delta w^l_{ij} + \Delta b^l_i) \cdot f'_i(x^{(l-1)}) g(k(t+1)). \tag{3.15}$$

Eq. (3.15) represents the change in activity (and hence in representation) that suffers the feed-forward path for any non-active context $k(t+1) \neq k'$. Meaning that this would be the difference observed in activity if we were to re-active an old task after the update. To prevent catastrophic forgetting we want $\Delta x^l_i = 0$. The common approach in the literature is to look at the relevant parameters and fix them hoping that it is sufficient to prevent catastrophic forgetting. This is what we do in p-GateON. Another more restrictive way is to ensure that $\Delta w^l_{ik}, \Delta b^l_i \to 0 \, \forall k$. One way to force the variation of all parameters connected to the neuron i to be 0 is the obstruction method we use in n-GateON, which ensures that the network prioritizes learning new information on available neurons while preserving the non-available ones. Finally depending on the approach we obstruct the gradient as follows for p-GateON

$$\begin{aligned}
\Delta w^{g,l}_{ik} &= -\eta_g \frac{\partial \mathscr{L}}{\partial w^{g,l}_{ik}} \\
\Delta w^l_{ik} &= -\eta \mathbf{A}_{w^l_{ij}} \frac{\partial \mathscr{L}}{\partial w^l_{ik}} \\
\Delta b^l_i &= -\eta \mathbf{A}_{b^l_i} \frac{\partial \mathscr{L}}{\partial b^l_i}.
\end{aligned} \tag{3.16}$$

and this version for n-GateON

$$\begin{aligned}\Delta w_{ik}^{g,l} &= -\eta_g \frac{\partial \mathscr{L}}{\partial w_{ik}^{g,l}} \\ \Delta w_{ik}^{l} &= -\eta \mathbf{A}_{x_i^l} \frac{\partial \mathscr{L}}{\partial w_{ik}^{l}} \\ \Delta b_i^l &= -\eta \mathbf{A}_{x_i^l} \frac{\partial \mathscr{L}}{\partial b_i^l}.\end{aligned} \tag{3.17}$$

Interestingly, some neurons have small but non-zero availability, which acts as a gradient regularizer. Due to their very low learning rate, these neurons cannot drift too far from the previous representation, helping to preserve previously learned information and prevent catastrophic forgetting.

3.A.5 Alternative relevance mappings approximations for n-GateON

In the scope of n-GateON Eq. (3.7) writes as

$$\mu_{x_i^l} \approx \left(\frac{\partial \mathscr{L}}{\partial x_i^l} x_i^l \right)^2 . \tag{3.18}$$

However, computing the term $\frac{\partial \mathscr{L}}{\partial x_i^l}$ can be expensive or not accessible, especially for convolutional layers. We propose two approximations. The first one is a local definition of causality that computes how much removing neuron i in layer l impacts neurons j in layer $l+1$

$$\begin{aligned}\mu_{x_i^l} &= \frac{1}{N} \sum_j (x_j^{l+1,-i} - x_j^{l+1})^2 \\ \mu_{x_i^l} &= \frac{1}{N} \sum_j g_j \cdot (f(I_j^{l+1,-i}) - f(I_j^{l+1}))^2 \\ &\underset{Taylor}{\approx} \frac{1}{N} \sum_j g_j \cdot (\nabla_{\vec{I}}^T f(I_j^{l+1}) \vec{w}_j x_i^l)^2,\end{aligned} \tag{3.19}$$

where $-i$ implies that we removed the neuron i in the forward path. For example, for ReLU Eq. (3.19) reduces to $\mu_{x_i^l} = \frac{1}{N} \sum_j g_j (w_{ji} x_i^l)^2$. This local variation allows computing a good approximation of the global causality if we don't have access to the loss function.

The last approximation, $\mu_{x_i^l} = (x_i^l)^2$, is the least precise but easiest to implement on any model. We show in Fig 3.A.2 that in practice using the approximations solely impacts the speed of availability reduction leading to faster network saturation. The problem of saturation is compensated to some extent by ϵ.

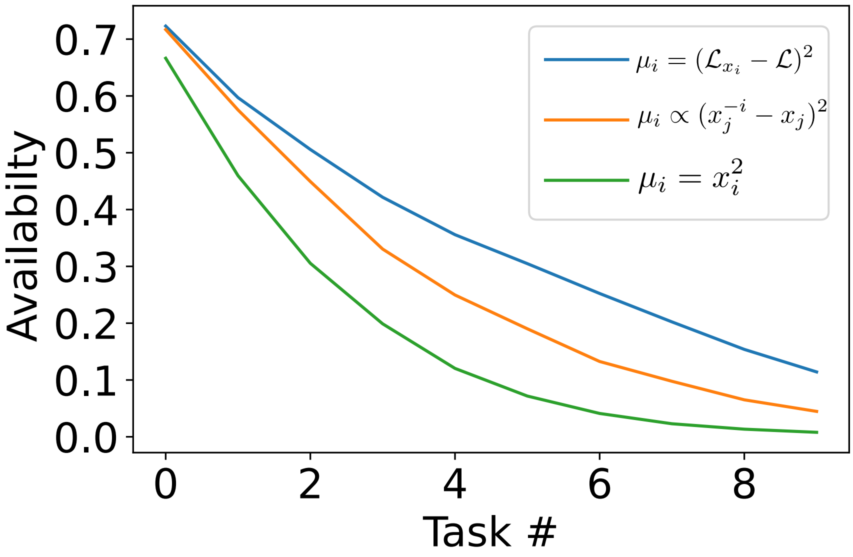


Figure 3.A.2: Network availability decay over 10 tasks for the three different versions of μ_i for n-GateON 0 on *Permuted*. Blue is the full causality, orange is its local approximation, and green weakest approximation. As the approximation weakens the availability decreases faster, meaning that an activity approximation obstructs the gradient of more neurons that its true counterpart. All reach 97.7% continual accuracy on 10 tasks, meaning that the CL capabilities of the network stays the same. In practice, the approximated relevance mapping only impacts the saturation of the network but not directly its CL properties.

3.A.6 n-GateON for CNN

In the context of a convolutional neural network (CNN), it is necessary to define the gating and availability operations at the level of filters as opposed to individual neurons. Building upon the notation established in the previous section, we refer to the filters in the l-th layer of channel c as $w^{c,l}_{a,b}$, where a and b correspond to the indices of the filter weights. By applying gating and availability operations to each filter in every channel c of layer l, we can compute the corresponding gated activity as follows:

$$\begin{aligned} x^{l,c}_{ij} &= g^l_c f\Big(\sum_{a=0}^{K}\sum_{b=0}^{K} w^{c,l}_{ab} x^{l-1,c}_{(i+a)(j+b)}\Big) \\ \text{with } g^l_c &= \sigma\Big(\sum_j w^{g,l}_{cj} C_j\Big), \end{aligned} \tag{3.20}$$

For p-GateON, the relevance mappings are exactly the same. Shamefully here the relevance in Eq.(3.6) is ill-defined in n-GateON for channels. More specifically, we discuss in math met 3.A.5 that the causal effect can be approximated by the mean activity of the channel outputs.

$$\mu^{l,c}_x = \frac{1}{N_i}\frac{1}{N_j}\sum_i\sum_j (x^{l,c}_{ij})^2 \tag{3.21}$$

3.A.7 Metrics for Continual learning

We train the tasks $t \in \{1, T\}$ sequentially. For any task t' larger than t we can measure the accuracy on task t right after training on task t', this test accuracy is denoted $A^t_{cc}|t'$. The immediate test accuracy after training task t being, $A^t_{cc} = A^{t|t}_{cc}$.
After training on all tasks we measure the continual accuracy

$$A^{cont}_{cc,t} = \frac{1}{T-t}\sum_{t' \geq t}^{T} A^{t|t'}_{cc}. \tag{3.22}$$

The continual accuracy can only expose the catastrophic forgetting property. To get an estimation of the capacity of sequential training we measure the ratio between the average immediate test accuracy of the GateON method and isolated networks called the relative immediate test accuracy ratio (ΔA^t_{cc}).

$$\Delta A^t_{cc} = \frac{1}{\bar{A}^{\text{Isolated}}_{cc},}\left(\frac{1}{T}\sum_t^T A^t_{cc} - \bar{A}^{\text{Isolated}}_{cc}\right), \tag{3.23}$$

with $\bar{A}^{\text{Isolated}}_{cc}$ the average accuracy over all tasks if networks are trained separately for each task. This measure allows for estimating the ability for forward transfer. A positive ΔA^t_{cc} shows that on average the immediate test accuracy is better with continual learning (forward transfer) than without. We also compute how much accuracy is lost compared to the immediate test accuracy, the forgetting rate (FR)

$$FR_t = \left(A_{cc}^{t} - \frac{1}{T-t} \sum_{t'>t}^{T} A_{cc}^{t|t'} \right). \tag{3.24}$$

A positive FR_t shows catastrophic forgetting, while negative FR_t represents backward transfer as it means that on average you have better accuracy later compared to right after training. The three measures in Eqs (3.22),(3.23),(3.24) summarize well all the aspects of catastrophic forgetting. The continual accuracy, the relative continual accuracy to the vanilla method, and the absolute forgetting. In the results section we show all the measures in %, i.e. multiplied by 100.

3.A.8 Network Generalization and neuron sharing

GateON utilizes over-parameterization in both fully-connected and convolutional models. Specifically, our method employs sub-networks, to solve each task, with typically only 2% to 10% of neurons per layer being used for a specific task. Yet, we show in Fig 3.A.3 that for correlated tasks (2 tasks with small angle difference on *Rotated*) neurons are shared while for un-correlated tasks (any 2 tasks in *Permuted*) the network decorrelates used neurons as much as possible until saturation. GateON capitalizes on the resemblances between tasks to enhance generalization, as opposed to merely dissociating tasks in order of appearances.

Figure 3.A.4 presents the task-locked average accuracy (see details in material method 3.A.9) during the 500 batch steps of task presentations, along with the vanilla first task test accuracy evolution in black (baseline). For orthogonal tasks, the network struggles to generalize and shows some loss in final accuracy for $\epsilon = 0$, resulting in the network not re-learning faster than the initial speed. For $\epsilon = 0$ we observe a better start test accuracy, this is not because it behaves better but because it does not detect switch points anymore. Conversely, for *Rotated*, GateON methods facilitate greatly faster learning of new tasks, suggesting that the method can generalize to correlated tasks, as expected from the correlation matrices. Notably, *Shuffled* proves to be counterintuitive to fully-connected models, significantly impeding the speed of re-training for non-gated networks (bottom right figure). GateON re-trains faster than vanilla but still can not generalize very well. The parameters of GateON appear to have no significant impact on correlated tasks. The shaded area in the figure corresponds to the standard deviation of the mean over (99 tasks * 5 simulations).

3.A.9 Task-locked accuracy

In the classical learning paradigms, we plot the evolution of the test accuracy of the training epochs or batch steps. For MNIST we train each task for 500 batch steps (around 9 epochs) and then switch to the next one. During training, we save the test accuracy of task t as $A_{cc}^{t}(step)$ for $step \in \{1, 500\}$.

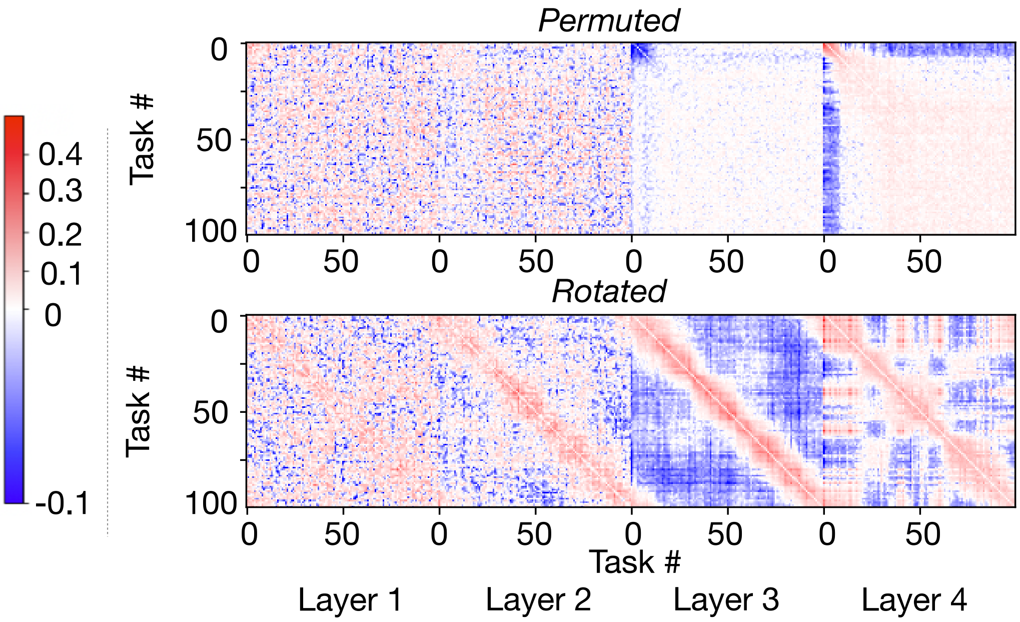


Figure 3.A.3: **Context correlation matrices**. This figure showcases the Pearson correlations of context weights across all layers of n-GateON$_{CNN}$ 0 after, trained on 100 tasks for *Permuted* (top) and *Rotated* (bottom). We use n-GateON$_{CNN}$ to show the effect on both FC and convolutional layers. For *Permuted* there is no clear structure in the CNN layers. For the fully connected ones, the first tasks are highly decorrelated until network saturation (around 10,15 tasks). This is due to *Permuted* having orthogonal tasks, meaning that you can not generalize and always use new neurons. For *Rotated* for visualization, we re-ordered after training the tasks by increasing angles. The context weights clearly correlate with their neighboring angles and decorate with the others. This correlation facilitates generalization across tasks. The figure highlights GateON's ability to discover existing structures or generate new selective neurons as needed. We can also explain the propensity of GateON to saturate in *Permuted*as at each new task a new set of neurons must be recruited while for *Rotated*neurons are reused depending on the angle.

To avoid showing redundant plots (100 tasks trained) and highlight the average convergence speed and accuracy during training we compute the average tasked-locked accuracy,

$$A_{cc}(step) = \frac{1}{T-1}\sum_{t>1}^{T} A^{t}_{cc}(step). \tag{3.25}$$

$A_{cc}(step)$ highlights the speed of re-training ($t > 1$) on average after a task switch and its average convergence. A fast re-training compared to $A^{1}_{cc}(step)$ shows the capacity of the network to generalize previous knowledge to new tasks. Similar convergence to the $A^{1}_{cc}(step)$ shows that the method does not harm the final accuracy either. In some cases, the final accuracy can even be better as the generalization might also benefit.

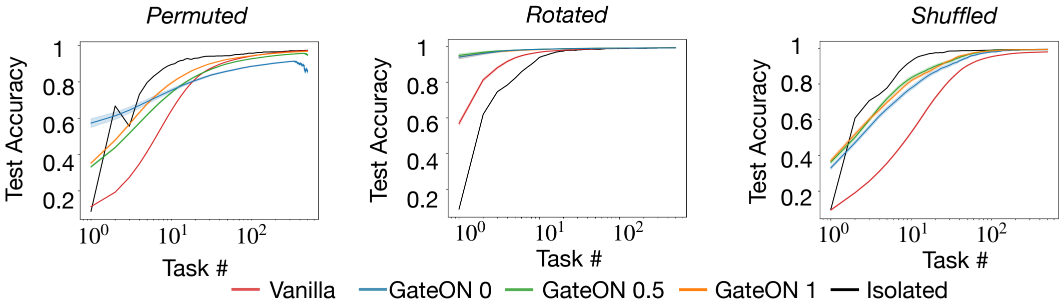


Figure 3.A.4: **Network generalization**. The three figures show the task-locked accuracy (**material methods** 3.A.9) over 500 batch presentations for all three MNIST tasks. The task-locked accuracy reveals the average re-training speed and final accuracy of the network. A network that generalizes well should learn faster and achieve higher accuracy if possible. Isolated indicates the average speed of training using a new network for each task. The number following GateON in the legend corresponds to the value of epsilon in Eq. (3.4) used for the GateON method.

3.A.10 Context correlation matrix

To be sure that sub-networks generated from the GateON method are not mutually exclusive we want to measure how the context weights that gate the activity of the neurons correlate between each task. To do so we binarise the context weight vector from context i to all neurons in layer l

$$w_{ik}^{l,binarised} = \begin{cases} 1 & \text{if } w_{ik}^{l} > 0 \\ 0 & \text{otherwise.} \end{cases} \tag{3.26}$$

We chose to binarise since we want to focus only on whether this neuron is used or not and hence if the neuron feature is re-used. Then we compute the Pearson correlation ([222])

$$C_{ij} = \frac{COV(\vec{w}_i^{binarised}, \vec{w}_j^{binarised})}{\sigma_i \sigma_j}. \tag{3.27}$$

Strong correlation means that similar neurons are used between the two tasks and hence the sub-networks are not separated. We expect this kind of behavior in correlated tasks like *Rotated* with two close angles. In orthogonal tasks we would expect the neurons to be anti-correlated, i.e. the network actively chooses other neurons than the one previously used.

3.A.11 Network architectures

10 Tasks

For *Permuted*, the network is composed of one input layer of size 728, 2 hidden layers with 2000 neurons, and an output layer of 10 neurons. We added two layers of conv with size 32 for *Split* and 256,512 for *Rotated*. We Trained for 17 epochs each. We use the Adam optimizer with a learning rate of $5e-3$, reset after each task switch. $\eta_A = 0.01$ for all layers. The gating activation function is a rectified hyperbolic tangent, ReLU for the neurons and softmax for the output. To make the learning faster we also add a batch norm at each layer. We use cross-entropy loss. Everything is implemented with Pytorch [223]

100 Tasks

Fully-connected architecture The architecture used for the three meta-tasks is the same with the same hyper-parameters using Pytorch [223]. It is composed of one input layer of size 728, 4 hidden layers with 5000 neurons, and an output layer of 10 neurons. So many neurons are necessary for solving 100 tasks but it can be decreased if fewer tasks require to be learned. We use the Adam optimizer with a learning rate of $5e-3$, reset after each task switch. $\eta_A = 0.01$ for all layers. The gating activation function is a rectified hyperbolic tangent, ReLU for the hidden neurons and softmax for the output. To make the learning faster we also add a batch norm at each layer, and we re-normalize the output layer activity at each layer. We use a cross-entropy loss.

Convolutional architecture The architecture used for the three meta-tasks is the same with the same hyper-parameters using Pytorch [223]. It is composed of one input layer of size 728, 2 convolutional layers with 256 and 512 channels followed by one max pool of kernel 4, 2 hidden layers with 5000 neurons, and the output layer of 10 neurons. We use the Adam optimizer with a learning rate of $5e-3$, reset after each task switch. $\eta_A = 0.004$ for all conv layers, $\eta_A = 0.01$ all hidden layers. The gating activation function is a rectified hyperbolic tangent, ReLU for the neurons and softmax for the output. To make the learning faster we also add a batch norm at each hidden layer, and we re-normalize the output layer activity. We use a cross-entropy loss.

BERT network architecture

We use the BERT base model (uncased) similarly as in Devlin et al.[189]. The same parameters are used for all tasks for the training. The parameters for context detection differ between tasks. The only adaption we did is to add the context layers. We also put the availability of the embedding layer and self-attention layers to 0. We do this because we did not have access to the pre-training to implement our relevance mapping and availability fixing. Since Attentions and embedding are projections in small dimensions it has a very dense representation leading to most of the neurons being relevant. We expect that for these layers our model would fix most of the neurons during training. Hence we set the availability of the embedding layer and each attention layer to 0. Note that they still have the gating that allows task-driven adaptation. The context parameters w are initialized so that $\sigma(w) = 1$ for all neurons and contexts. It prevents gating to change too much the Bert representation before training. We use $\eta_A = 0.05$ except if the first dataset has less than 50 batch updates in this case the first task uses $\eta_A = 0.1$.

	k	Θ	η_A	η_C
Permuted	1	1.01	0.01	0.02
Rotated	1	1.01	0.01	0.02
Shuffle	1	1.01	0.01	0.02

Table 3.A.1: GateON parameters for MNIST

	k	Θ	η_A	η_C
ASC	3	1.1	0.05	0.02
DSC	3	5	0.05	0.02
newsgroup	3	5	0.05	0.02

Table 3.A.2: GateON parameters for NLP

3.B Supplementary tables

Results GateON, task-fed

	Permuted			*Rotated*			*Shuffled*		
Models	A_{cc}^{cont}	ΔA_{cc}^{t}	FR.	A_{cc}^{cont}	ΔA_{cc}^{t}	FR	A_{cc}^{cont}	ΔA_{cc}^{t}	FR
Vanilla	26.29	0.0	71.57	69.68	0.0	28.53	13.41	0.0	77.44
n-Gating	55.07	**0.24**	41.96	71.63	0.94	25.33	62.07	0.6	35.11
n-GateON 1	95.96	0.2	1.5	97.46	**1.18**	0.49	96.38	**1.05**	1.52
n-GateON 0.5	95.88	-1.39	0.25	97.6	1.08	0.19	97.66	1.01	0.19
n-GateON 0	92.38	-5.0	0.25	97.39	0.82	**0.05**	**97.72**	0.93	0.06
p-GateON 0.5	96.54	-0.05	0.79	**97.83**	0.67	0.14	97.52	0.31	0.36
p-GateON 0	**96.56**	-0.62	**0.23**	97.8	0.64	0.06	97.69	0.25	**0.15**

Table 3.B.1: Performance results on the three CL-tasks (*Permuted*, *Rotated*, and *Shuffled*) for 100 tasks with task fed in the context layer.

	Permuted			*Rotated*			*Shuffled*		
Models	A_{cc}^{cont}	ΔA_{cc}^{t}	FR.	A_{cc}^{cont}	ΔA_{cc}^{t}	FR	A_{cc}^{cont}	ΔA_{cc}^{t}	FR
Vanilla$_{CNN}$	41.32		55.91	94.06		5.33	40.15		58.22
Gating$_{CNN}$	50.03	**-0.07**	47.15	88.76	-0.04	10.1	48.63	-0.14	50.65
n-GateON$_{CNN}$ 1	96.16	-0.58	0.9	98.83	**0.43**	0.37	98.5	**-0.1**2	0.78
n-GateON$_{CNN}$ 0.5	**96.57**	-0.72	**0.37**	98.62	0.38	0.36	98.97	-0.15	0.28
n-GateON$_{CNN}$ 0	89.43	-7.91	0.78	**99.11**	0.25	**0.06**	98.97	-0.26	**0.2**
p-GateON$_{CNN}$ 0.5	94.75	-1.4	1.62	99.1	0.11	0.21	98.94	-0.14	0.34
p-GateON$_{CNN}$ 0	95.02	-2.34	0.48	99.05	0.09	0.12	**99.06**	-0.14	0.22

Table 3.B.2: Performance results on the three CL-tasks (*Permuted*, *Rotated*, and *Shuffled*) for 100 tasks with task-fed in the context layer using CNN.

Results GateON, task inferred

	Permuted			*Rotated*			*Shuffled*		
Models	A_{cc}^{cont}	ΔA_{cc}^{t}	FR.	A_{cc}^{cont}	ΔA_{cc}^{t}	FR	A_{cc}^{cont}	ΔA_{cc}^{t}	FR
Vanilla	26.29	0.0	71.57	69.68	0.0	28.53	13.41	0.0	77.44
Gating	55.07	**0.24**	41.96	71.63	**0.94**	25.33	62.07	**0.6**	35.11
n-GateON 1	95.75	-0.48	1.12	97.7	0.7	0.24	96.89	0.4	1.09
n-GateON 0.5	92.86	-1.57	2.06	97.6	0.66	0.33	97.35	0.39	0.62
n-GateON 0	73.64	-5.61	14.82	97.62	0.59	0.35	**97.72**	0.26	**0.12**
p-GateON 0.5	**96.48**	-0.05	0.84	97.76	0.55	0.16	97.51	0.29	0.37
p-GateON 0	96.34	-0.6	**0.48**	**97.87**	0.64	**0.14**	97.69	0.24	0.13

Table 3.B.3: Performance results on the three CL-tasks (*Permuted*, *Rotated*, and *Shuffled*) for 100 tasks with task inferred by the network.

	Permuted			*Rotated*			*Shuffled*		
Models	A_{cc}^{cont}	ΔA_{cc}^{t}	FR.	A_{cc}^{cont}	ΔA_{cc}^{t}	FR	A_{cc}^{cont}	ΔA_{cc}^{t}	FR
Vanilla$_{CNN}$	41.32		55.91	94.06		5.33	40.15		58.22
Gating$_{CNN}$	50.03	**-0.07**	47.15	88.76	-0.04	10.1	48.63	**-0.14**	50.65
n-GateON$_{CNN}$ 1	89.82	-2.61	4.03	98.92	**0.11**	0.45	98.71	-0.16	0.55
n-GateON$_{CNN}$ 0.5	93.22	-1.95	**1.64**	98.79	0.03	0.5	98.92	-0.18	0.32
n-GateON$_{CNN}$ 0	76.8	-8.79	10.56	98.89	-0.01	**0.37**	99.02	-0.21	**0.2**
p-GateON$_{CNN}$ 0.5	**93.4**	-1.59	2.51	98.92	0.07	0.41	98.91	-0.15	0.37
p-GateON$_{CNN}$ 0	93.36	-2.17	1.71	**98.97**	0.07	0.36	**99.04**	-0.16	0.22

Table 3.B.4: Performance results on the three CL-tasks (*Permuted*, *Rotated*, and *Shuffled*) for 100 tasks with task inferred by the network.

Contribution

In our first chapter, we developed a module-based approach to context-based learning using SpikeSuM. While this approach provided valuable insights, we recognized that the human and animal brain exhibits neuron sharing [224] and the ability to generalize across tasks[225], far from our modular approach. To address these features, we developed GateON. Drawing on studies on complex neurons[37], task-specific dendritic input[36], and our work on SpikeSuM, we employed context-based multiplicative gating of activity in a bio-inspired model. Our research has demonstrated that GateON is a performative model with many novel ideas for the machine learning community, such as neuron-based relevance mapping instead of parameters, which has proven effective even for large language models. We also developed leaky relevance mapping to prevent network fixing and implemented contextual gating using multiplicative gating based on one-hot encoded tasks, which simplifies complex embedding of context that typically requires binarized gating to avoid interferences across tasks.

We anticipate that GateON will have many applications in neuroscience. In fact, most of GateON's properties were inspired by our previous work on spiking networks, such as unsupervised context detection and context-based predictions. GateON is also inspired by experimental results on possible multiplicative gating of activity through dendritic inputs. All the computations are done online, and most can be at least approximated with local computations (for n-GateON). In fact, only the contextual gating needs to use the backpropagation algorithm to be updated, yet we have good hope that even this could be solved by the use of predictive coding-like architecture where the error is computed layer-wise.

4 Discussion & Conclusions

In this thesis, we studied how surprise signals can be extracted from the spike activity of an error-minimizing network, and how this process triggers the release of neuromodulators to promote rapid adaptation to unexpected events. Our model proposes that the three-factor Hebbian rules play a critical role in promoting efficient model rehabilitation.

However, our findings also indicate that our prime model exhibited strong catastrophic forgetting, which is the tendency to forget previously learned information. While humans are adept at re-adaptation, they are also skilled at multitasking and integrating new information without erasing old ones.

In order to address the problem of catastrophic forgetting resulting from rapid adaptation, we proposed a solution that involves contextualizing predictions. Specifically, we have developed a bio-plausible circuit that builds on top of our surprise detector network, which allows for detecting context switches and context-driven predictions. This approach enables our network to make more precise predictions based on the current context, while also retaining memory of previous contexts for reactivation later.

Furthermore, our new approach to understanding the role of surprise and prediction error has led to the development of new hypotheses for context detection in the brain. Such has the putative Layer 5 integration of prediction error coming from layer 2/3 leads to both computations of surprise signal and context boundary detection. We also propose global/local neuromodulation where the global signal is broadcast from deep areas such as locus coeruleus and our model of competitive inhibitory populations fine-tunes the effect of neuromodulation locally. We hope that these testable hypotheses can be further explored and confirmed or refuted by experiments.

However, one major drawback of SpikeSuM is its independent modular architecture, which prevents the reuse of old knowledge for new tasks or the update of previous models based on new observations. These limitations are known as forward and backward transfer, and they have been extensively studied in the field of continual learning in machine learning.

To address this problem, we have dived into the machine-learning community and approached our problem from a different angle. This approach builds on the long tradition of back and forth between biology and machine learning. We built our intuition around the GateON method from experimental results, that claim that dendritic input might act as a multiplicative gating on the axonal input [36, 37]. Indeed, one of the two main properties of the GateON method is the gating of the feed-forward input from the contextual neurons. The gating allows us to bypass our previous modular approach to create task-specific subnetworks that are not necessarily independent. The second property of GateON is neuron availability. The neuron availability depends on the activity of the neuron over the course of training. The more a neuron has been active, the less it will be available for integrating new information. Neuron availability can also be understood as the neuron 'age'. In other words, a neuron that has been active a lot is considered as old and can not change its representation anymore. We hope that by sharing this perspective, we can raise the reader's interest and curiosity about the actual bio-compatibility of our GateON method.

For the GateON method, the only non-bio-implementable aspect is the back-propagation used for the weight update of the context weight. We predict that in the future, a bio-implementation of such a model can be done using local learning rules. We have good reasons to believe that a network based on local error computation such as CLAPP[105] would allow for a bio implementation of GateON.

We anticipate that a model of this nature, particularly when adapted to a biologically plausible algorithm, could offer insights into key aspects of brain mechanisms, such as the interplay between dendritic and axonal inputs or the phenomenon of representational drift over time.

We trust that a combination of these approaches, informed by a deep understanding of the underlying mechanisms, can lead to significant advances in the field of continual learning and by extension in the understanding of continual learning in the neuroscience community.

Conclusion

This thesis has explored the integration of information at different timescales in the brain and the resulting properties that emerge from it.
Our models' smallest time resolution is the timescale of single action potentials, in the order of milliseconds, consistent with the literature[24] that these spikes (and their related activity) are the first responsible for the synaptic weight evolution.

From there, we explored how these short impulses can be integrated over time to detect wrongly predicted outputs and estimate surprise on the order of hundreds of milliseconds. We use this surprise for unsupervised updates of our world model (synaptic weights) and fast adaptation to novel information.

Our work has also shed light on the importance of context-based prediction and the role it plays in identifying dynamics that occur on timescales ranging from minutes to several hours.

Our latest method, GateON, which involves a context-gated and representation-fixing algorithm, has allowed us to develop a lifelong learning network that can integrate new knowledge continually over the putative course of years, without suffering from catastrophic forgetting.

From a theoretical perspective, identify two promising directions for further exploration in the realm of surprise-based learning. Firstly, developing a model capable of more intricate top-down predictions, instead of relying on our fixed buffer population, would represent a substantial advancement. While our classical Hebbian approach necessitated the use of shallow networks without recurrences, our learning modulation could be adapted to incorporate deep spiking neural network methods like e-prop[91] and others. Secondly, our current approach does not encompass complex multisensory processing or the crucial state-action pair. Consequently, it would be intriguing to delve deeper into surprise-based learning by exploring action-based tasks and model-based algorithms, which account for the interdependency that arises from model-building and action-taking. Regarding the continual learning framework, we did not study the impact of learning tasks with different input/output spaces or look at how we could enforce more hierarchical generalization such as understanding rotation vs. permutation that would make the method even more powerful.

In terms of experimental opportunities, there are various task designs that hold significant promise. In Chapter One, we proposed an experiment utilizing SpikeSum, which incorporates different levels of stochasticity to study the hypothesized variations in brain activity. The SpikeSum-C model also presents a range of hypotheses. Of particular interest would be investigating the activity patterns of L5 neurons at switch points and for reappearing contexts. Additionally, exploring the dis-inhibitory networks driving reversed winner-take-all mechanisms in the brain, as introduced for context detection, would be both intriguing and challenging. Regarding GateON, there are already opportunities for designing compelling studies. Our main assumption for continual learning revolves around the concept of soft neuronal fixing. Experimentally examining the existence of such fixing in the brain and determining whether it occurs at the parameter or neuronal level, as described in Chapter 2, would be highly valuable.

Overall, we are optimistic about the potential impact of our work and believe that it will continue to advance our understanding of the brain and its remarkable ability to integrate information at different timescales. We look forward to seeing how our findings will be applied in future research and practical applications, and we are excited to continue exploring this fascinating area of study.

Bibliography

[1] Stephen I Gallant et al. Perceptron-based learning algorithms. *IEEE Transactions on neural networks*, 1(2):179–191, 1990.

[2] Ashish Vaswani, 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.

[3] George N Reeke and Gerald M Edelman. Real brains and artificial intelligence. *Daedalus*, pages 143–173, 1988.

[4] Jingtao Fan, Lu Fang, Jiamin Wu, Yuchen Guo, and Qionghai Dai. From brain science to artificial intelligence. *Engineering*, 6(3):248–252, 2020.

[5] Anthony Zador, Sean Escola, Blake Richards, Bence Ölveczky, Yoshua Bengio, Kwabena Boahen, Matthew Botvinick, Dmitri Chklovskii, Anne Churchland, Claudia Clopath, et al. Catalyzing next-generation artificial intelligence through neuroai. *Nature Communications*, 14(1):1597, 2023.

[6] Ricardo Augusto de Melo Reis, Hércules Rezende Freitas, and Fernando Garcia De Mello. Cell calcium imaging as a reliable method to study neuron–glial circuits. *Frontiers in Neuroscience*, 14:569361, 2020.

[7] Peter A Rinck. *Magnetic resonance in medicine: a critical introduction*. BoD–Books on Demand, 2019.

[8] Winifried Denk, James H Strickler, and Watt W Webb. Two-photon laser scanning fluorescence microscopy. *Science*, 248(4951):73–76, 1990.

[9] Alan L Hodgkin and Andrew F Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117 (4):500, 1952.

[10] Santiago Ramón y Cajal. The croonian lecture.—la fine structure des centres nerveux. *Proceedings of the Royal Society of London*, 55(331-335):444–468, 1894.

[11] Vernon B Mountcastle. Modality and topographic properties of single neurons of cat's somatic sensory cortex. *Journal of neurophysiology*, 20(4):408–434, 1957.

[12] W. Schultz, P. Dayan, and R.R. Montague. A neural substrate for prediction and reward. *Science*, 275:1593–1599, 1997.

[13] Nicholas M Barnes and Trevor Sharp. A review of central 5-ht receptors and their function. *Neuropharmacology*, 38(8):1083–1152, 1999.

[14] Craig W Berridge and Barry D Waterhouse. The locus coeruleus–noradrenergic system: modulation of behavioral state and state-dependent cognitive processes. *Brain Research Reviews*, 42(1):33–84, 2003. ISSN 0165-0173. doi: https://doi.org/10.1016/S0165-0173(03)00143-7. URL https://www.sciencedirect.com/science/article/pii/S0165017303001437.

[15] Michael E Hasselmo. Neuromodulation: acetylcholine and memory consolidation. *Trends in cognitive sciences*, 3(9):351–359, 1999.

[16] Michael C Avery and Jeffrey L Krichmar. Neuromodulatory systems and their interactions: a review of models, theories, and experiments. *Frontiers in neural circuits*, page 108, 2017.

[17] A. Yu and P. Dayan. Uncertainty, neuromodulation, and attention. *Neuron*, 46:681–692, 2005.

[18] Matthew R Nassar, Robert C Wilson, Benjamin Heasly, and Joshua I Gold. An approximately bayesian delta-rule model explains the dynamics of belief updating in a changing environment. *Journal of Neuroscience*, 30(37):12366–12378, 2010.

[19] Terence W Picton. The p300 wave of the human event-related potential. *Journal of clinical neurophysiology*, 9(4):456–479, 1992.

[20] Santiago Ramón y Cajal. *Sobre las fibras nerviosas de la capa molecular del cerebelo*. 1888.

[21] Overview of neuron structure and function (article). URL https://www.khanacademy.org/science/biology/human-biology/neuron-nervous-system/a/overview-of-neuron-structure-and-function.

[22] Brian S Meldrum. Glutamate as a neurotransmitter in the brain: review of physiology and pathology. *The Journal of nutrition*, 130(4):1007S–1015S, 2000.

[23] William Van der Kloot and J Robbins. The effects of γ-aminobutyric acid and picrotoxin on the junctional potential and the contraction of crayfish muscle. *Experientia*, 15:35–36, 1959.

[24] Donald Olding Hebb. *The organization of behavior; a neuropsychological theory*. Wiley, 1949.

[25] John NJ Reynolds and Jeffery R Wickens. Dopamine-dependent plasticity of corticostriatal synapses. *Neural networks*, 15(4-6):507–521, 2002.

[26] Łukasz Kuśmierz, Takuya Isomura, and Taro Toyoizumi. Learning with three factors: modulating hebbian plasticity with errors. *Current opinion in neurobiology*, 46:170–177, 2017.

[27] A. Modirshanechi, J. Brea, and W. Gerstner. A taxonomy of surprise definitions. *J. Mathem. Psychol.*, 110:102712, 2022.

[28] Laurent Itti and Pierre Baldi. Bayesian surprise attracts human attention. *Vision research*, 49(10):1295–1306, 2009.

[29] Claude E Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[30] Timothy EJ Behrens, Mark W Woolrich, Mark E Walton, and Matthew FS Rushworth. Learning the value of information in an uncertain world. *Nature neuroscience*, 10(9): 1214, 2007.

[31] Matthew R Nassar, Katherine M Rumsey, Robert C Wilson, Kinjan Parikh, Benjamin Heasly, and Joshua I Gold. Rational regulation of learning dynamics by pupil-linked arousal systems. *Nature neuroscience*, 15(7):1040, 2012.

[32] Vasiliki Liakoni, Alireza Modirshanechi, Wulfram Gerstner, and Johanni Brea. Learning in volatile environments with the bayes factor surprise. *Neural Computation*, 33(2): 269–340, 2021.

[33] Timo Flesch, Keno Juechems, Tsvetomira Dumbalska, Andrew Saxe, and Christopher Summerfield. Orthogonal representations for robust context-dependent task performance in brains and neural networks. *Neuron*, 110(7):1258–1270, 2022.

[34] OECD. Measuring the environmental impacts of artificial intelligence compute and applications. (341), 2022. doi: https://doi.org/https://doi.org/10.1787/7babf571-en. URL https://www.oecd-ilibrary.org/content/paper/7babf571-en.

[35] Bernard Marr. Green intelligence: Why data and ai must become more sustainable, Mar 2023. URL https://www.forbes.com/sites/bernardmarr/2023/03/22/green-intelligence-why-data-and-ai-must-become-more-sustainable/?sh=685e2d8e7658.

[36] Guang Yang, Cora Sau Wan Lai, Joseph Cichon, Lei Ma, Wei Li, and Wen-Biao Gan. Sleep promotes branch-specific formation of dendritic spines after learning. *Science*, 344 (6188):1173–1178, 2014.

[37] Danke Zhang, Yuanqing Li, Malte J Rasch, and Si Wu. Nonlinear multiplicative dendritic integration in neuron and network models. *Frontiers in computational neuroscience*, 7: 56, 2013.

[38] K.C. Squires, C. Wickens, N.K. Squires, and E. Donchin. The effect of stimulus sequence on the waveform of the cortical event-related potential. *Science*, 193:1141–1146, 1976.

[39] Wulf-Uwe Meyer, Michael Niepel, Udo Rudolph, and Achim Schützwohl. An experimental analysis of surprise. *Cognition & Emotion*, 5(4):295–311, 1991.

[40] M.M. Hurley, D.C. Dennett, and R.B. Adams. *Inside jokes: Using humor to reverse-engineer the mind.* MIT Press, Cambridge, 2011.

[41] J. Schnupp, I. Nelken, and A.J. King. *Auditory Neuroscience: Making Sense of Sound.* MIT Press, Cambridge, Mass. (USA), 2011. ISBN: 9780262518024.

[42] D. Huron. *Sweet anticipation: music and the psychology of expectation.* MIT Press, Cambridge (Mass), USA, 2008. SBN-13: 978-0262582780.

[43] S. Koelsch. Brain correlates of music-evoked emotions. *Nat. Rev. Neurosci.*, 15:170–180, 2014.

[44] B.A. Ardekani, S.J. Choi, G.-A. Hossein-Zadeh, B. Projesz, J.L. Tanabe, K.O. Lim, R. Bilder, J. A. Helpern, and H. Begleiter. Functional magnetic resonance imaging of brain activity in the visual oddball task. *Cognitive Brain Research*, 14:347–356, 2002.

[45] Tobias Egner, Jim M Monti, and Christopher Summerfield. Expectation and surprise determine neural population responses in the ventral visual stream. *Journal of Neuroscience*, 30(49):16601–16608, 2010.

[46] J. Homann, S.A. Koay, K.S. Chen, D.W. Tank, and M.J. Berry II. Novel stimuli evoke excess activity in the mouse primary visual cortex. *Proc. Natl. Acad. Sci. (U.S.A.)*, 119: e2108882119, 2022.

[47] David Foster, Richard Morris, and Peter Dayan. Models of hippocampally dependent navigation using the temporal difference learning rule. *Hippocampus*, 10:1–16, 2000.

[48] F.A. Mansouri, K. Matsumoto, and K. Tanaka. Prefrontal cell activities related to monkeys' success and failure in adapting to rule changes in a wisconsin card sorting test analog. *J. Neurosci.*, 26:2745–2756, 2002.

[49] S. Fusi, W.F. Asaad, E.K. Miller, and X.-J. Wang. A neural circuit model of flexible sensorimotor mapping: Learning and forgetting on multiple timescales. *Neuron*, 54:319–333, 2007.

[50] S. Matias, E. Lottem, G.P. Dugue, and Z.F. Mainen. Activity patterns of serotonin neurons underlying cognitive flexibility. *eLife*, 6:e20552, 2017.

[51] Rajeev V Rikhye, Aditya Gilra, and Michael M Halassa. Thalamic regulation of switching between cortical representations enables cognitive flexibility. *Nature neuroscience*, 21 (12):1753–1763, 2018.

[52] He A Xu, Alireza Modirshanechi, Marco P Lehmann, Wulfram Gerstner, and Michael H Herzog. Novelty is not surprise: Human exploratory and adaptive behavior in sequential decision-making. *PLOS Computational Biology*, 17(6):e1009070, 2021.

[53] Andrew Barto, Marco Mirolli, and Gianluca Baldassarre. Novelty or surprise? *Frontiers in psychology*, 4:907, 2013.

[54] Michael Davis. The mammalian startle response. In *Neural mechanisms of startle behavior*, pages 287–351. Springer, 1984.

[55] Claudio Lavin, René San Martin, and Eduardo Rosales Jubal. Pupil dilation signals uncertainty and surprise in a learning gambling task. *Frontiers in behavioral neuroscience*, 7:218, 2014.

[56] Kerstin Preuschoff, Bernard Marius ’t Hart, and Wolfgang Einhäuser. Pupil dilation signals surprise: Evidence for noradrenaline’s role in decision making. *Frontiers in neuroscience*, 5:115, 2011. doi: 10.3389/fnins.2011.00115.

[57] Catarina Amado, Petra Hermann, Petra Kovács, Mareike Grotheer, Zoltán Vidnyánszky, and Gyula Kovács. The contribution of surprise to the prediction based modulation of fmri responses. *Neuropsychologia*, 84:105–112, 2016.

[58] Maxime Maheu, Stanislas Dehaene, and Florent Meyniel. Brain signatures of a multiscale process of sequence learning in humans. *elife*, 8:e41541, 2019.

[59] Gary Aston-Jones and Jonathan D Cohen. An integrative theory of locus coeruleus-norepinephrine function: adaptive gain and optimal performance. *Annu. Rev. Neurosci.*, 28:403–450, 2005.

[60] Georg B Keller and Thomas D Mrsic-Flogel. Predictive processing: a canonical cortical computation. *Neuron*, 100(2):424–435, 2018.

[61] James W. Antony, Thomas H. Hartshorne, Ken Pomeroy, Todd M. Gureckis, Uri Hasson, Samuel D. McDougle, and Kenneth A. Norman. Behavioral, physiological, and neural signatures of surprise during naturalistic sports viewing. *Neuron*, 109(2):377–390.e7, 2021. ISSN 0896-6273. doi: 10.1016/j.neuron.2020.10.029.

[62] Nuttapong Chentanez, Andrew Barto, and Satinder Singh. Intrinsically motivated reinforcement learning. In L. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 17. MIT Press, 2004.

[63] Charan Ranganath and Gregor Rainer. Neural mechanisms for detecting and remembering novel events. *Nature Reviews Neuroscience*, 4(3):193–202, 2003.

[64] ME Hasselmo and LM Giocomo. Cholinergic modulation of cortical function. *Journal of Molecular Neuroscience*, 30(1-2):133, 2006.

[65] Tal Neiman and Yonatan Loewenstein. Covariance-based synaptic plasticity in an attractor network model accounts for fast adaptation in free operant learning. *Journal of Neuroscience*, 33(4):1521–1534, 2013.

[66] Alireza Soltani, Daeyeol Lee, and Xiao-Jing Wang. Neural mechanism for stochastic behaviour during a competitive game. *Neural Networks*, 19(8):1075–1090, 2006.

[67] David Clewett, Andrej Schoeke, and Mara Mather. Locus coeruleus neuromodulation of memories encoded during negative or unexpected action outcomes. *Neurobiology of Learning and Memory*, 111:65–70, 2014.

[68] Samuel J Gershman, Marie-H Monfils, Kenneth A Norman, and Yael Niv. The computational nature of memory modification. *Elife*, 6:e23763, 2017.

[69] John Duncan and Glyn W Humphreys. Visual search and stimulus similarity. *Psychological review*, 96(3):433, 1989.

[70] J. Schmidhuber. Curious model-building control systems. In *Proceedings of the International Joint Conference on Neural Networks, Singapore*, 1991.

[71] Laurent Itti and Pierre Baldi. A principled approach to detecting surprising events in video. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 631–637. IEEE, 2005.

[72] Mohammadjavad Faraji, Kerstin Preuschoff, and Wulfram Gerstner. Balancing new against old information: the role of puzzlement surprise in learning. *Neural computation*, 30(1):34–83, 2018.

[73] Karl Friston. The free-energy principle: a unified brain theory? *Nature reviews neuroscience*, 11(2):127, 2010.

[74] Jürgen Schmidhuber. Formal theory of creativity, fun, and intrinsic motivation (1990–2010). *IEEE Transactions on Autonomous Mental Development*, 2(3):230–247, 2010.

[75] Ryan Prescott Adams and David JC MacKay. Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*, 2007.

[76] Gianluigi Mongillo and Sophie Deneve. Online learning with hidden markov models. *Neural computation*, 20(7):1706–1716, 2008.

[77] Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, et al. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538(7626):471–476, 2016.

[78] Tiago Ramalho and Marta Garnelo. Adaptive posterior learning: few-shot learning with a surprise-based memory module. *arXiv preprint arXiv:1902.02527*, 2019.

[79] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286. PMLR, 2014.

[80] Johanni Brea, Walter Senn, and Jean-Pascal Pfister. Matching recall and storage in sequence learning with spiking neural networks. *Journal of neuroscience*, 33(23):9565–9575, 2013.

[81] João Sacramento, Rui Ponte Costa, Yoshua Bengio, and Walter Senn. Dendritic cortical microcircuits approximate the backpropagation algorithm. In *Advances in neural information processing systems*, pages 8721–8732, 2018.

[82] V. Pawlak, J.R. Wickens, A. Kirkwood, and J.N.D. Kerr. Timing is not everything: neuromodulation opens the STDP gate. *Front. Synaptic Neurosci.*, 2:146, 2010.

[83] J. Lisman, A.A. Grace, and E. Duzel. A neoHebbian framework for episodic memory; role of dopamine-dependent late ltp. *Trends Neurosci.*, 34:536–547, 2011.

[84] Nicolas Frémaux and Wulfram Gerstner. Neuromodulated spike-timing-dependent plasticity, and theory of three-factor learning rules. *Frontiers in neural circuits*, 9:85, 2016.

[85] L. Kusmierz, T. Isomura, and T. Toyoizumi. Learning with three factors: modulating hebbian plasticity with errors. *Curr. Opin. Neurobiol.*, 46:170–177, 2017.

[86] Wulfram Gerstner, Marco Lehmann, Vasiliki Liakoni, Dane Corneil, and Johanni Brea. Eligibility traces and plasticity on behavioral time scales: experimental support of neohebbian three-factor learning rules. *Frontiers in neural circuits*, 12:53, 2018.

[87] Pieter R. Roelfsema and Anthony Holtmaat. Control of synaptic plasticity in deep cortical networks. *Nature Reviews Neuroscience*, 19(3):166–180, 2018. URL http://dx.doi.org/10.1038/nrn.2018.6.

[88] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[89] Timothy P Lillicrap, Daniel Cownden, Douglas B Tweed, and Colin J Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nature communications*, 7:13276, 2016.

[90] Emre O Neftci, Hesham Mostafa, and Friedemann Zenke. Surrogate gradient learning in spiking neural networks. *IEEE Signal Processing Magazine*, 36:61–63, 2019.

[91] Guillaume Bellec, Franz Scherr, Anand Subramoney, Elias Hajek, Darjan Salaj, Robert Legenstein, and Wolfgang Maass. A solution to the learning dilemma for recurrent networks of spiking neurons. *Nature communications*, 11(1):1–15, 2020.

[92] I. Pozzi, S.M. Bothe, and P.R. Roelfsema. Attention-gated brain propagation: How the brain can implement reward-based error backpropagation. In *Adv. Neur. Inform. Proc. Systems 33 (NeurIPS 2020)*, volume 33, pages 2516–2526. Curran Associates, Inc., 2020.

[93] Michael Okun and Ilan Lampl. Balance of excitation and inhibition. *Scholarpedia*, 4(8): 7467, 2009.

[94] Tim P Vogels, Kanaka Rajan, and Larry F Abbott. Neural network dynamics. *Annu. Rev. Neurosci.*, 28:357–376, 2005.

[95] Tim P Vogels, Henning Sprekeler, Friedemann Zenke, Claudia Clopath, and Wulfram Gerstner. Inhibitory plasticity balances excitation and inhibition in sensory pathways and memory networks. *Science*, 334(6062):1569–1573, 2011.

[96] Rajesh PN Rao and Dana H Ballard. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature neuroscience*, 2(1): 79, 1999.

[97] Karl Friston. A theory of cortical responses. *Philosophical transactions of the Royal Society B: Biological sciences*, 360(1456):815–836, 2005.

[98] Michael W Spratling. Predictive coding as a model of biased competition in visual attention. *Vision research*, 48(12):1391–1408, 2008.

[99] Laurence Aitchison and Máté Lengyel. With or without you: predictive coding and bayesian inference in the brain. *Current opinion in neurobiology*, 46:219–227, 2017.

[100] C.M.A. Pennartz, S. Dora, L. Muckli, and J.A.M. Loteije. Towards a unified view on pathways and functions of neural recurrent processing. *Trends Neursci.*, 42:589–603, 2019.

[101] L. Hertag and H. Sprekeler. Learning prediction error neurons in a canonical inteneuron circuit. *eLife*, 9:e57541, 2020.

[102] Tommaso Salvatori, Yuhang Song, Yujian Hong, Lei Sha, Simon Frieder, Zhenghua Xu, Rafal Bogacz, and Thomas Lukasiewicz. Associative memories via predictive coding. *Advances in Neural Information Processing Systems*, 34, 2021.

[103] William Lotter, Gabriel Kreiman, and David Cox. Deep predictive coding networks for video prediction and unsupervised learning. *arXiv preprint arXiv:1605.08104*, 2016.

[104] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[105] Bernd Illing, Jean Ventura, Guillaume Bellec, and Wulfram Gerstner. Local plasticity rules can learn deep representations using self-supervised contrastive predictions. *Advances in Neural Information Processing Systems*, 34, 2021.

[106] Tommaso Salvatori, Yuhang Song, Thomas Lukasiewicz, Rafal Bogacz, and Zhenghua Xu. Predictive coding can do exact backpropagation on convolutional and recurrent neural networks. *arXiv preprint arXiv:2103.03725*, 2021.

[107] H. B. Barlow. Possible principles underlying the transformation of sensory messages. In W. A. Rosenbluth, editor, *Sensory Communication*, pages 217–234. MIT Press, 1961.

[108] G. van de Ven and A.S. Tolias. Three scenarios for continual learning. *arXiv*, (arxiv): 1904.07734, 2019.

[109] G. Carpenter and S. Grossberg. The art of adpative pattern recognition by a self-organizing neural network. *Computer*, 21:77–88, 1988.

[110] S. Fusi, P.J. Drew, and L.F. Abbott. Cascade models of synaptically stored memories. *Neuron*, 45:599–611, 2005.

[111] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pages 3987–3995. PMLR, 2017.

[112] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[113] James B Heald, Máté Lengyel, and Daniel M Wolpert. Contextual inference underlies the learning of sensorimotor repertoires. *Nature*, 600(7889):489–493, 2021.

[114] Daniel M Wolpert and Mitsuo Kawato. Multiple paired forward and inverse models for motor control. *Neural networks*, 11(7-8):1317–1329, 1998.

[115] Youngmin Oh and Nicolas Schweighofer. Minimizing precision-weighted sensory prediction errors via memory formation and switching in motor adaptation. *Journal of Neuroscience*, 39(46):9237–9250, 2019.

[116] Florent Meyniel. Brain dynamics for confidence-weighted learning. *PLOS Computational Biology*, 16(6):1–27, 06 2020. doi: 10.1371/journal.pcbi.1007935.

[117] Sam Gijsen, Miro Grundei, Robert T Lange, Dirk Ostwald, and Felix Blankenburg. Neural surprise in somatosensory bayesian learning. *PLoS computational biology*, 17(2): e1008068, 2021.

[118] Florent Meyniel, Maxime Maheu, and Stanislas Dehaene. Human inferences about sequences: A minimal transition probability model. *PLoS computational biology*, 12 (12):e1005260, 2016.

[119] Roy A Wise and P-P Rompre. Brain dopamine and reward. *Annual review of psychology*, 40(1):191–225, 1989.

[120] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction.* MIT Press, Cambridge, MA, (in progress) second edition, 2018.

[121] Jacqueline Gottlieb and Pierre-Yves Oudeyer. Towards a neuroscience of active sampling and curiosity. *Nature Reviews Neuroscience*, 19(12):758–770, 2018. doi: 10.1038/s41583-018-0078-0.

[122] Aris Fiser, David Mahringer, Hassana K Oyibo, Anders V Petersen, Marcus Leinweber, and Georg B Keller. Experience-dependent spatial expectations in mouse visual cortex. *Nature neuroscience*, 19(12):1658–1664, 2016.

[123] Zahid Padamsey and Nathalie L Rochefort. Defying expectations: How neurons compute prediction errors in visual cortex. *Neuron*, 108(6):1016–1019, 2020.

[124] E.J. Nestler, Paul J. Kenny, S. J. Russo, and A. Schaefer. Widely projecting systems: Monoamines, acetylcholine, and orexin. In *Molecular Neuropharmacology: A Foundation for Clinical Neuroscience (4th ed.), Chapter 6*, chapter 6, pages 158–194. McGraw-Hill Medical, New York, 2020.

[125] M. Morales and E.B. Margolis. Ventral tegmental area: cellular heterogeneity, connectivity and behaviour. *Nat. Rev. Neurosci.*, 18:73–85, 2017.

[126] J. Aru, M. Suzuki, and M.E. Larkum. Cellular mechanisms of conscious processing. *Trends Cogn. Sci.*, 24:814–825, 2020. doi:10.1016/j.tics.2020.07.006.

[127] E.G. Jones. The thalamic matrix and thalamocortical synchrony. *Trends Neurosci.*, 24: 595–601, 2001.

[128] K.D. Harris and G.M.G. Shepherd. The neocortical circuit: themes and variations. *Nat. Rev. Neurosci.*, 18:170–181, 2015.

[129] Travis Meyer and Carl R Olson. Statistical learning of visual transitions in monkey inferotemporal cortex. *Proceedings of the National Academy of Sciences*, 108(48):19401–19406, 2011.

[130] Hanneke EM Den Ouden, Peter Kok, and Floris P De Lange. How prediction errors shape perception, attention, and motivation. *Frontiers in psychology*, 3:548, 2012.

[131] Richard Kempter, Wulfram Gerstner, and J Leo van Hemmen. Intrinsic stabilization of output rates by spike-based hebbian learning. *Neural computation*, 13(12):2709–2741, 2001.

[132] Wulfram Gerstner, Werner M Kistler, Richard Naud, and Liam Paninski. *Neuronal dynamics: From single neurons to networks and models of cognition*, chapter 6.4. Cambridge University Press, 2014.

[133] Jasper Poort, Adil G Khan, Marius Pachitariu, Abdellatif Nemri, Ivana Orsolic, Julija Krupic, Marius Bauza, Maneesh Sahani, Georg B Keller, Thomas D Mrsic-Flogel, et al. Learning enhances sensory and multiple non-sensory representations in primary visual cortex. *Neuron*, 86(6):1478–1490, 2015.

[134] Jan Gläscher, Nathaniel Daw, Peter Dayan, and John P O'Doherty. States versus rewards: dissociable neural prediction error signals underlying model-based and model-free reinforcement learning. *Neuron*, 66(4):585–595, 2010.

[135] Alireza Modirshanechi, Mohammad Mahdi Kiani, and Hamid Aghajan. Trial-by-trial surprise-decoding model for visual and auditory binary oddball tasks. *NeuroImage*, 196: 302–317, 2019.

[136] Dirk Ostwald, Bernhard Spitzer, Matthias Guggenmos, Timo T. Schmidt, Stefan J. Kiebel, and Felix Blankenburg. Evidence for neural encoding of bayesian surprise in human somatosensation. *NeuroImage*, 62(1):177–188, 2012. ISSN 1053-8119. doi: 10.1016/j.neuroimage.2012.04.050.

[137] V. Yakovlev, S. Fusi, E. Berman, and E. Zohary. Inter-trial neuronal activity in inferior temporal cortex: a putative vehicle to generate long-term visual associations. *Nat. Neurosci.*, 1:310–317, 1998.

[138] M.R. Warden and E.K. Miller. Task-dependent changes in short-term memory in the prefrontal cortex. *J. Neurosci.*, 30:15801–16810, 2010.

[139] Rebecca Jordan and Georg B Keller. Opposing influence of top-down and bottom-up input on excitatory layer 2/3 neurons in mouse primary visual cortex. *Neuron*, 108(6): 1194–1206, 2020.

[140] J.I. Kang, F. Huppe-Gourgues, and E. Vaucher. Boosting visual cortex function and plasticity with acetylcholine to enhance visual perception. *Front. Syst. Neurosci.*, 8:172, 2014.

[141] S.J. Sara. The locus coeruleus and noradrenergic modulation of cognition. *Nat. Rev. Neurosci.*, 10:211–223, 2009.

[142] W. Schultz. Predictive reward signal of dopamine neurons. *J. Neurophysiology*, 80:1–27, 1998.

[143] P. Redgrave and K. Gurney. The short-latency dopamine signal: a role in discovering novel actions? *Nat. Rev. Neurosci.*, 7:967–975, 2006.

[144] P. Devoto and G. Flore. On the origin of cortical dopamine: Is it a co-transmitter in noradrenergic neurons? *Current Neuropharm.*, 4:115–125, 2006.

[145] T. Takeuchi, A.J. Duszkiewicz, A. Sonneborn, P.A. Spooner, M. Yamasaki, M. Watanabe, C.C. Smith, G. Fernandez, K. Deisseroth, R.W. Greene, and R.G.M. Morris. Locus coeruleus and dopaminergic consolidation of everyday memory. *Nature*, 537:357–362, 2016.

[146] M. Rigotti, O. Barak, M.R. Warden, X.-J. Wang, N.D. Daw, E.K. Miller, and S. Fusi. The importance of mixed selectivity in complex cognitive tasks. *Nature*, 497:585–590, 2013.

[147] Alireza Soltani and Alicia Izquierdo. Adaptive learning under expected and unexpected uncertainty. *Nature Reviews Neuroscience*, 20:635–644, 2019. doi: 10.1038/s41583-019-0180-y.

[148] H.F. Clarke, J.W. Dalley, H.S. Crofts, T.W Robbins, and A.C. Roberts. Cognitive inflexibility after prefrontal serotonin depletion. *Science*, 304:878–880, 2004.

[149] Juanita Todd, Patricia T Michie, Ulrich Schall, Philip B Ward, and Stanley V Catts. Mismatch negativity (mmn) reduction in schizophrenia—impaired prediction-error generation, estimation or salience? *International Journal of Psychophysiology*, 83(2):222–231, 2012.

[150] Yuichi Yamashita and Jun Tani. Spontaneous prediction error generation in schizophrenia. *PLoS One*, 7(5):e37843, 2012.

[151] Judith M Ford and Daniel H Mathalon. Anticipating the future: automatic prediction failures in schizophrenia. *International Journal of Psychophysiology*, 83(2):232–239, 2012.

[152] Victoria B Gradin, Poornima Kumar, Gordon Waiter, Trevor Ahearn, Catriona Stickle, Marteen Milders, Ian Reid, Jeremy Hall, and J Douglas Steele. Expected value and prediction error abnormalities in depression and schizophrenia. *Brain*, 134(6):1751–1764, 2011.

[153] Buranee Kanchanatawan, Solaphat Hemrungrojn, Supaksorn Thika, Sunee Sirivichayakul, Kiat Ruxrungtham, André F Carvalho, Michel Geffard, George Anderson, and Michael Maes. Changes in tryptophan catabolite (trycat) pathway patterning are associated with mild impairments in declarative memory in schizophrenia and deficits in semantic and episodic memory coupled with increased false-memory creation in deficit schizophrenia. *Molecular neurobiology*, 55(6):5184–5201, 2018.

[154] Buranee Kanchanatawan, Supaksorn Thika, Sunee Sirivichayakul, André F Carvalho, Michel Geffard, and Michael Maes. In schizophrenia, depression, anxiety, and physiosomatic symptoms are strongly related to psychotic symptoms and excitation, impairments in episodic memory, and increased production of neurotoxic tryptophan catabolites: a multivariate and machine learning study. *Neurotoxicity research*, 33(3):641–655, 2018.

[155] David C Glahn, Jennifer Barrett, Carrie E Bearden, Jim Mintz, Michael F Green, E Serap Monkul, Pablo Najt, Jair C Soares, and Dawn I Velligan. Dissociable mechanisms for memory impairment in bipolar disorder and schizophrenia. *Psychological Medicine*, 36(8):1085, 2006.

[156] NF Forbes, LA Carrick, AM McIntosh, and SM Lawrie. Working memory in schizophrenia: a meta-analysis. *Psychological medicine*, 39(6):889, 2009.

[157] Peter U Diehl, Daniel Neil, Jonathan Binas, Matthew Cook, Shih-Chii Liu, and Michael Pfeiffer. Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. ieee, 2015.

[158] Katarzyna Kozdon and Peter Bentley. Normalisation of weights and firing rates in spiking neural networks with spike-timing-dependent plasticity. *arXiv preprint arXiv:1910.00122*, 2019.

[159] Tim Van Erven and Peter Harremos. Rényi divergence and kullback-leibler divergence. *IEEE Transactions on Information Theory*, 60(7):3797–3820, 2014.

[160] Kiyohito Iigaya. Adaptive learning and decision-making under uncertainty by metaplastic synapses guided by a surprise detection system. *Elife*, 5:e18073, 2016.

[161] Anna Kutschireiter, Simone Carlo Surace, Henning Sprekeler, and Jean-Pascal Pfister. Nonlinear bayesian filtering and learning: a neuronal dynamics for perception. *Scientific reports*, 7(1):8722, 2017.

[162] W. Gerstner. Time structure of the activity in neural network models. *Phys. Rev. E*, 51(1): 738–758, 1995.

[163] Simone Carlo Surace, Jean-Pascal Pfister, Wulfram Gerstner, and Johanni Brea. On the choice of metric in gradient-based theories of brain function. *PLoS computational biology*, 16(4):e1007640, 2020.

[164] W. Gerstner and W. K. Kistler. *Spiking Neuron Models: single neurons, populations, plasticity*. Cambridge University Press, Cambridge UK, 2002.

[165] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

[166] Tim Head, MechCoder, Gilles Louppe, Iaroslav Shcherbatyi, fcharras, Zé Vinícius, cmmalone, Christopher Schröder, nel215, Nuno Campos, Todd Young, Stefano Cereda, Thomas Fan, rene rex, Kejia (KJ) Shi, Justus Schwabedal, carlosdanielcsantos, Hvass-Labs, Mikhail Pak, SoManyUsernamesTaken, Fred Callaway, Loïc Estève, Lilian Besson, Mehdi Cherti, Karlson Pfannschmidt, Fabian Linzberger, Christophe Cauet, Anna Gut, Andreas Mueller, and Alexander Fabisch. scikit-optimize/scikit-optimize: v0.5.2, March 2018. URL https://doi.org/10.5281/zenodo.1207017.

[167] S. Recanatesi, M. Farrell, G. Lajoie, S. Deneve, M. Rigotti, and E. Shea-Brown. Predictive learning as a network mechanism for extracting low-dimensional latent space representations. *Nat. Comm.*, 12:1417, 2021.

[168] Y. Liu, M.G. Mattar, T.E.J. Behrens, N.D. Daw, and R.J. Dolan. Experience replay is associated with efficient non-local learning. *Science*, 372:abf1357, 2021.

[169] M.G. Mattar and M. Lengyel. Planning in the brain. *Neuron*, 110:xx, 2022.

[170] D.L.K. Yamins and J.J. DiCarlo. Using goal-driven deep learning models to understand sensory cortex. *Nature neuroscience*, 19:356–365, 2016.

[171] D. E. Rumelhart, J.L. McClelland, and the PDP research group. *Parallel distributed processing: Explorations in the microstructure of cognition. Vol. 1: Foundations.* MIT Press, Cambridge Mass., 1986.

[172] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997. doi: 10.1162/neco.1997.9.8.1735.

[173] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[174] Timo Flesch, Jan Balaguer, Ronald Dekker, Hamed Nili, and Christopher Summerfield. Comparing continual task learning in minds and machines. *Proceedings of the National Academy of Sciences*, 115(44):E10313–E10322, 2018.

[175] Benjamin Frederick Goodrich. *Neuron clustering for mitigating catastrophic forgetting in supervised and reinforcement learning.* PhD thesis, Knoxville, 2015.

[176] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

[177] Roger Ratcliff. Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. *Psychological review*, 97(2):285, 1990.

[178] Stephen Grossberg. Competitive learning: From interactive activation to adaptive resonance. *Cognitive science*, 11(1):23–63, 1987.

[179] Zoltán Dienes and Gerry Altmann. Transfer of implicit knowledge across domains: How implicit and how abstract. *How implicit is implicit learning*, 5:107–123, 1997.

[180] Vivian Cook, editor. *Effects of the Second Language on the First.* Multilingual Matters, Bristol, Blue Ridge Summit, 2003. ISBN 9781853596346. doi: doi:10.21832/9781853596346. URL https://doi.org/10.21832/9781853596346.

[181] Christos Kaplanis, Murray Shanahan, and Claudia Clopath. Continual reinforcement learning with complex synapses. In *International Conference on Machine Learning*, pages 2497–2506. PMLR, 2018.

[182] Abhiram Iyer, Karan Grewal, Akash Velu, Lucas Oliveira Souza, Jeremy Forest, and Subutai Ahmad. Avoiding catastrophe: Active dendrites enable multi-task learning in dynamic environments. *Frontiers in neurorobotics*, 16, 2022.

[183] Gail A. Carpenter and Stephen Grossberg. Adaptive resonance theory. In Claude Sammut and Geoffrey I. Webb, editors, *Encyclopedia of Machine Learning*, pages 22–35. Springer US, Boston, MA, 2010. ISBN 978-0-387-30164-8. doi: 10.1007/978-0-387-30164-8_11. URL https://doi.org/10.1007/978-0-387-30164-8_11.

[184] Rahul Ramesh and Pratik Chaudhari. Model zoo: A growing brain that learns continually. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=WfvgGBcgbE7.

[185] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, pages 4548–4557. PMLR, 2018.

[186] Fan-Keng Sun, Cheng-Hao Ho, and Hung-Yi Lee. {LAMOL}: {LA}nguage modeling is all you need for lifelong language learning. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=Skgxcn4YDS.

[187] Zixuan Ke, Hu Xu, and Bing Liu. Adapting BERT for continual learning of a sequence of aspect sentiment classification tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4746–4755, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.378. URL https://aclanthology.org/2021.naacl-main.378.

[188] K. Weiss, T. M. Khoshgoftaar, and D. Wang. A survey of transfer learning. *J Big Data 3*, 9, 2016.

[189] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423.

[190] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[191] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021.

[192] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017.

[193] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/fa7cdfad1a5aaf8370ebeda47a1ff1c3-Paper.pdf.

[194] Matthias De Lange and Tinne Tuytelaars. Continual prototype evolution: Learning online from non-stationary data streams. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8230–8239, 2021. doi: 10.1109/ICCV48922.2021.00814.

[195] Daniel L. Silver and Robert E. Mercer. The task rehearsal method of life-long learning: Overcoming impoverished data. In Robin Cohen and Bruce Spencer, editors, *Advances in Artificial Intelligence*, pages 90–101, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg. ISBN 978-3-540-47922-2.

[196] Cuong V. Nguyen, Yingzhen Li, Thang D. Bui, and Richard E. Turner. Variational continual learning. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=BkQqq0gRb.

[197] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[198] Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. Expert gate: Lifelong learning with a network of experts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[199] Prakhar Kaushik, Adam Kortylewski, Alex Gain, and Alan Yuille. Understanding catastrophic forgetting and remembering in continual learning with optimal relevance mapping. In *Fifth Workshop on Meta-Learning at the Conference on Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=Pvqe_hQEXTJ.

[200] Zixuan Ke, Bing Liu, Nianzu Ma, Hu Xu, and Lei Shu. Achieving forgetting prevention and knowledge transfer in continual learning. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=RJ7XFI15Q8f.

[201] R.P. Adams and D.J. MacKay. Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*, arXiv:0710.3742, 2007.

[202] P. Fearnhead and Z. Liu. On-line inference for multiple changepoint problems. *J. Roy. Statistical Soc.: Series B*, 69:589–605, 2007.

[203] M.R. Nassar, R.C. Wilson, B. Heasly, and J.I. Gold. An approximately bayesian delta-rule model explains the dynamics of belief updating in a changing environment. *J. Neurosci.*, 30:12366–12378, 2010.

[204] Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, and Balaji Lakshminarayanan. Detecting out-of-distribution inputs to deep generative models using typicality. *arXiv preprint arXiv:1906.02994*, 2019.

[205] V. Liakoni, A. Modirshanechi, W. Gerstner, and J. Brea. Learning in volatile environments with the bayes factor surprise. *Neur. Comput.*, 33:269–340, 2021.

[206] Ju Xu and Zhanxing Zhu. Reinforced continual learning. *Advances in Neural Information Processing Systems*, 31, 2018.

[207] Rupesh K Srivastava, Jonathan Masci, Sohrob Kazerounian, Faustino Gomez, and Jürgen Schmidhuber. Compete to compute. In C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013. URL https://proceedings.neurips.cc/paper_files/paper/2013/file/8f1d43620bc6bb580df6e80b0dc05c48-Paper.pdf.

[208] David Lopez-Paz and Marc' Aurelio Ranzato. Gradient episodic memory for continual learning. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/f87522788a2be2d171666752f97ddebb-Paper.pdf.

[209] Ian J Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*, 2013.

[210] Seyed Iman Mirzadeh, Mehrdad Farajtabar, Razvan Pascanu, and Hassan Ghasemzadeh. Understanding the role of training regimes in continual learning. *Advances in Neural Information Processing Systems*, 33:7308–7320, 2020.

[211] Seyed-Iman Mirzadeh, Mehrdad Farajtabar, Dilan Görür, Razvan Pascanu, and Hassan Ghasemzadeh. Linear mode connectivity in multitask and continual learning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=Fmg_fQYUejf.

[212] Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In *International Conference on Artificial Intelligence and Statistics*, pages 3762–3773. PMLR, 2020.

[213] Abraham C. Wickliffe and Anthony Robins. Memory retention – the synaptic stability versus plasticity dilemma. *Trends in Neurosciences*, 28(2):73–78, 2005. ISSN 0166-2236. doi: https://doi.org/10.1016/j.tins.2004.12.003. URL https://www.sciencedirect.com/science/article/pii/S0166223604003704.

[214] Hu Xu, Bing Liu, Lei Shu, and Philip Yu. BERT post-training for review reading comprehension and aspect-based sentiment analysis. 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)*, pages 2324–2335, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1242. URL https://aclanthology.org/N19-1242.

[215] Zixuan Ke, Bing Liu, Hao Wang, and Lei Shu. Continual learning with knowledge transfer for sentiment classification. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2020, Ghent, Belgium, September 14–18, 2020, Proceedings, Part III*, page 683–698, Berlin, Heidelberg, 2020. Springer-Verlag. ISBN 978-3-030-67663-6. doi: 10.1007/978-3-030-67664-3_41. URL https://doi.org/10.1007/978-3-030-67664-3_41.

[216] Ken Lang. Newsweeder: Learning to filter netnews. In *Machine learning proceedings 1995*, pages 331–339. Elsevier, 1995.

[217] Timothée Lesort, Vincenzo Lomonaco, Andrei Stoian, Davide Maltoni, David Filliat, and Natalia Díaz-Rodríguez. Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges. *Information Fusion*, 58:52–68, 2020. ISSN 1566-2535. doi: https://doi.org/10.1016/j.inffus.2019.12.004. URL https://www.sciencedirect.com/science/article/pii/S1566253519307377.

[218] A. Robins. Catastrophic forgetting in neural networks: the role of rehearsal mechanisms. In *Proceedings 1993 The First New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems*, pages 65–68, 1993. doi: 10.1109/ANNES.1993.323080.

[219] David B. T. McMahon, Adam P. Jones, Igor V. Bondar, and David A. Leopold. Face-selective neurons maintain consistent visual responses across months. *Proceedings of the National Academy of Sciences*, 111(22):8251–8256, 2014. doi: 10.1073/pnas.1318331111. URL https://www.pnas.org/doi/abs/10.1073/pnas.1318331111.

[220] K. A. Zhivago and S. P. Arun. Selective IT neurons are selective along many dimensions. *J Neurophysiol*, 115(3):1512–1520, March 2016.

[221] Laura N. Driscoll, Lea Duncker, and Christopher D. Harvey. Representational drift: Emerging theories for continual learning and experimental future directions. *Current Opinion in Neurobiology*, 76:102609, 2022. ISSN 0959-4388. doi: https://doi.org/10.1016/j.conb.2022.102609. URL https://www.sciencedirect.com/science/article/pii/S0959438822001039.

[222] Karl Pearson. Vii. note on regression and inheritance in the case of two parents. *Proceedings of the Royal Society of London*, 58(347-352):240–242, 1895.

[223] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[224] Gerald N Pho, Michael J Goard, Jonathan Woodson, Benjamin Crawford, and Mriganka Sur. Task-dependent representations of stimulus and choice in mouse parietal cortex. *Nature communications*, 9(1):2596, 2018.

[225] Beverly A Wright and Yuxuan Zhang. A review of the generalization of auditory learning. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 364(1515):301–311, 2009.

Martin Barry

3 rue du nord, 1005 Lausanne
✉ martin.barry@epfl.ch

Employment History

2018 – 2018	**Internship,** 6 months research in laboratory of computational neuroscience. Studying Surprise models in neuroscience, École Polytechnique Fédérale de Lausanne, EPFL
2017 – 2017	**Internship,** 6 months research in the image processing department of Sony. Specialized on skin detection with deep networks using multichannel images.Sony Deutschland GmbH, Stuttgart
2013 – 2017	**Teaching assistant,** physics, and analysis teaching assistant for many semesters, École Polytechnique Fédérale de Lausanne, EPFL
2013 – 2013	**Janitor.** Centre Hospitalier Universitaire de Lille.

Education

2018 – …	**Ph.D. Candidate, in theoretical neuroscience** Laboratory of computational neuroscience, EPFL. Thesis title: *From event-based surprise to lifelong learning. A journey in the timescales of adaptation*
2015 – 2018	**M.Sc. Computational science and engineering** École Polytechnique Fédérale de Lausanne, EPFL. Thesis title: *Neural implementation of a three factor learning rule based on surprise minimization.*
2012 – 2015	**B.Sc. Physics,** École Polytechnique Fédérale de Lausanne, EPFL.

Research Publications

Journal Articles

1. m. L. L. R. Barry and W. Gerstner, “Fast adaptation to rule switching using neuronal surprise,” *bioRxiv*, pp. 2022–09, 2022.

Conference Posters

1. Barry and Gerstner, “Surprise gated learning provides a solution to plasticity-stability dilemma,” 2022.
2. Barry, Stein, and Gerstner, “Surprise learning: Fast adaptation in volatile environments with spiking neural networks,” 2020.

Skills

Languages	Native French. Strong reading, writing, and speaking competencies in English. Basics in German and Italian.
Coding	Python, C++, Java, Julia,SQL, LaTeX, Git …
Databases	MySQL, SQLite.
Web Dev	HTML, CSS, JavaScript.
Sport	Climbing, judo, beach volley.
Misc.	Academic research, teaching.

Miscellaneous Experience

Awards and Achievements

2017	**Best talk**, SIAM Graduate Research day

Associations

2021-2022	Treasure of Effective Altruism, Lausanne. Organizations of conferences and regular meet-ups.
	Organization of conferences in collaboration with many political parties and ecological associations.