Are Associations All You Need? Overcoming the Problem of One-to-Many Association and Nonlinear Relationships with Contiguity in a Bidirectional Associative Memory Framework

Damiem Rolon-Mérette

Thesis submitted to the University of Ottawa

in partial fulfillment of the requirements for the

PhD in Experimental Psychology

School of Psychology

Faculty of Social Sciences

University of Ottawa

Acknowledgments

This culmination of my research and my journey toward a doctorate in philosophy would not have been possible without the belief bestowed on me by my supervisor, Dr. Sylvain Chartier. His guidance, support, exceptional mentorship, and humanness throughout my thesis have helped me traverse the rollercoaster of emotions that is often associated with this endeavor. His expertise, dedication, and commitment to my personal and academic growth have been instrumental in shaping my mind as a researcher and my motivation for writing this body of work. He is genuinely an excellent supervisor and a dear friend.

I must also express my gratitude to my thesis committee members: Dr. Melanie Sekeres, Dr. Jean-Philippe Thivierge, Dr. Dennis Cousineau, and Randall K. Jamieson. Their time, expertise, and willingness to share their knowledge have contributed significantly to the refinement and completion of this thesis.

This journey would not have been associated with an extraordinary feeling of accomplishment without the support of my colleagues and fellow researchers. From the hallway think tanks and bubbling lab meetings to the unforgettable discussions and outings, Thank you, Nareg Berberian, Matthew Ross, Kinsey Church, Matias Calderini, André Cyr, Melissa Johnson, Albino Nikolla and Marc Dufour.

I wish to emphasize how much completing this thesis would not have been possible without the pillars of my life and their extraordinary support. Thank you to my parents, Marcel Mérette and Sueli Rolon dos Santos; without your insistency, belief and help, this dream of mine would not have been possible. Thank you to my daughter, Gisëlle d'Amours Mérette; you have been my source of inspiration and motivation; without you, all this effort would have been meaningless. Thank you to my siblings, Diego, Muriel, Thaddé and Rita Rolon-Mérette, as you

always made me feel valued and loved. Thank you to my friends Chris, Omar, Robbie, Dave, Martin, Audrey-Anne, and Tarique; without you guys, I would not have had the chance to practice and refine my communication skills, nor would I have enjoyed doing so.

A special thank you to my brother, the motivator, hard worker and great believer, Thaddé Rolon-Mérette. Without you, I would have never gotten back up when I fell those many times. Without you, I would not have had such an incredible and extraordinary journey as a researcher and would not have been able to dream of this life.

I would also like to thank my partner, Marwa Younes, for her love, understanding, unshaken support, and undeniable belief in me. I would also like to express my gratitude to several families and friends who have supported my journey since the beginning: Sarah, Jessica and all the Morins, Florence, Christian, Hélène and all the D'Amours, David and all the Tavares, Danielle, Terry and all the Godfreys, and, of course, the entirety of the Mérettes. I am so grateful and lucky to have you all by my side, and I look forward to celebrating this achievement and many more milestones together.

Finally, I would like to offer a thought to anyone who reads this body of work, including myself or my kin, in years to come. Throughout this adventure, I have stumbled many, many, many times. I have felt lost and hopeless in a sea of unfair and unjust retroaction. Dreaming and being creative is not often accompanied by open arms. Rather, skepticism and an insistence toward known and familiar dogmas are what is most appealable. That said, it is the dreams and the desire to create them that will allow you to overcome any obstacles consistently. I learned this the hard way. And so, to all those who read this thesis, I say: Never stop dreaming, protect those you believe in at all costs, and never doubt your ability to make them come true. After all, is it not, in part, what being human is all about, to dream?

Contents

Table of Acronyms

(Sorted alphabetically)

Acronyms	Meaning
α-β BAM	alpha-beta Bidirectional Associative Memory
AGI	Artificial general intelligence
AM	Associative Memory
ANN	Artificial Neural Network
ART	Adaptive Resonance Theory model
BAM	Bidirectional Associative Memory
BHM	Bidirectional Heteroassociative Memory
BSB	Brain-State-in-Box
CCBAM	Chaotic Complex-valued Bidirectional Associative Memory
DCCS	Dimension Change Card Sort
FEBAM	Feature Extraction Bidirectional Associative Memory
HL	Hebbian Learning
LAM	Linear Associative Memory
LieBAM	Lie algebra-valued BAM
MAM1	Multidirectional Associative Memory
MAM2	Morphological Associative Memories
MBAM	Morphological Bidirectional Associative Memory
MF	Multi-Feature extracting bidirectional associative memory
MF-BAM	Multi-Feature - Bidirectional Associative Memory
ML	Machine Learning

MLP	Multi-Layer Perceptron
MOA	Many-to-one association
MSE	Mean Squared Error
NAM	Neural Associative Memory
NDRAM	Nonlinear dynamic recurrent associative memory
NF	Novelty Filter
OMA	One-to-many association
OOA	One-to-one association
PCA	Principal Component Analysis
RAM	Recurrent Associative Memory
RAM	Recurrent Associative Memory
RGB	Red Green Blue
RNAM	Recurrent Neural Associative Memory
SL	Supervised network Layer
S-R	Stimulus-response
Std	Standard deviation
STDP	Spike-timing-dependent plasticity
SVM	Support vector machine
TAM	Temporal Associative Memory
UL	Unsupervised network Layer
WCST	Wisconsin Card sort test

List of Figures

List of Tables

Summary

This thesis investigated how mechanisms related to contiguity could help explain the link between biology and cognition through an information signal processing model. It focused on using recurrent neural associative memories, specifically, a biological and cognitively plausible Bidirectional Associative Memory framework. The objective was to overcome its current limitations in learning one-to-many associations and nonlinear relationships. It proposed two major changes to achieve this: the use of contextual information and a new architecture inspired by the laminar organization of the brain.

In Chapter 1, contextual units were added to the input layer to change its overall representation. Results showed that by adding context, it was possible to manipulate the inter- and intra-group correlation during an association task. Of importance, it enabled the ability to learn one-to-many associations within the framework. Chapter 2 introduced the multi-feature extraction bidirectional associative memory, a multilayered model designed to investigate if nonlinear tasks, such as the N-bit, double moon, and its variants, could be solved within the framework. Results showcased how this recurrent multi-structure could generate various representations from a single signal. These were used to solve the nonlinear tasks. Results also showed how a transition from linear to nonlinear answers could be achieved within the proposed architecture. Finally, Chapter 3 employed two of these models and used contextual information to have them interact to solve complex cognitive tasks, such as the N-bit and dimension change card sort. Results showed that by using interacting associations, the joint model could learn rule-like behaviors and solve both tasks more efficiently and generally.

This thesis showed how the simple mechanism of contiguity could account for complex cognitive behavior. It showcases how associative memory mechanisms can emulate rule-like

behavior and surpass the traditional rote association strategy. This avenue holds great promise for enhancing the generalization ability of artificial neural networks. Of interest is that it also showed abundant evidence for new accountability of the associative memory mechanism. This paves the way for a deeper understanding of cognition and how information signal processing, through mechanisms related to contiguity, can amount to a human's vast repertoire of cognitive abilities.

Preface

A passion of mine has always been to question how humans learn and store information. What I consider especially fascinating regarding this topic is how this is accomplished by billions of interconnected neurons in the brain. Due to my curiosity, I have been on a journey searching for answers related to how these networks of neurons, through information signal processing, are capable of accounting for learning and memory in humans.

Throughout this voyage, I sought answers in biochemistry, neuroscience, and psychology. In these rich bodies of literature, I found evidence of a link between Hebbian learning (**HL**) and the theory of associationism. For more context, let me begin by elaborating on HL. One can simplify this notion to neurons that fire together wire together. In other words, if neuron A produces an action potential and neuron B follows the same trend, their connection strengthens, especially if repeated over time. Consequently, this augmentation makes it easier for A to activate B in the future. The theory of associationism has a similar coactivation mechanism but at the cognitive level. When different ideas, thoughts, and concepts, i.e., mental states, are seen or exposed together, an invisible glue is formed between them, linking them together. This association is responsible for a mental state evoking its paired state when being recalled from memory. Like HL, the theory of associationism suggests that this association happens in part due to repeated exposure. In other words, mental states that are seen together (neurons that fire together) become associated together (wire together).

At the macro level, this questionable link has been a focus of mine. I have wondered if this mechanism of coactivation in the information signal processing system of the brain can be seen as Occam's razor of human cognition, especially for learning and memory. As some pure vindicators of associationism would suggest, we are association machines, moldable from the ground up, built

from this invisible glue. Through my research, I wished to get a little closer to answering this question: Are associations all we need?

I have opted for a different route from traditional empirical research to investigate this. As I discovered the field of computational modelling due to Dr. Claude Lamontagne and Dr. Sylvain Chartier, I fell in love with the general idea of this method: abstract the observable phenomenon into axioms, use them to form theorems, and from there, interpret and explain how empirical regularities emerge. Applied to the biological neuron results in a simple representation. This depiction, often referred to as a node, can account for the mechanisms related to receiving synaptic currents, tallying them up while respecting a threshold function and the synaptic sites' efficacy. By combining various nodes, networks of artificial neurons can be built and used to simulate information signal processing. This enables a particular understanding and explanation of how networks of neurons can amount to learning and memory in the human brain.

Of importance, artificial neural networks (**ANNs**) that simulate Hebbian learning and associative memory (**AM**) are scarce but do exist. Often referred to as recurrent neural associative memory (**RNAM**), this subclass of ANNs has many mechanisms auspicious for modelling human cognition through information signal processing. However, many limitations are present in these models. Specifically, their inability to simulate one-to-many associations (**OMAs**) and deal with nonlinear relationships between the associated mental states. Both, further described in this thesis, are essential hurdles to account for complex cognitive behaviors found in humans. Thus, at the micro level, the goal of my thesis has been to try to surpass the current drawbacks in hopes of getting closer to understanding how information signal processing, HL, AM and human cognition all fit together. Thus, the main contribution of this thesis is a method for dealing with OMAs and a new RNAM model that can use this method while being impervious to nonlinear relationships.

The thesis's final chapters try to show how these novelties can be used to solve complex tasks more efficiently and cognitively plausible. It ends with my thoughts on how these mechanisms can inform us about human cognition and the limits related to the means of coactivation.

It is important that I mention that throughout my thesis, I have collaborated with my brother, Thaddé Rolon-Mérette and my supervisor, Dr Sylvain Chartier. I owe both gratitude for their different takes and knowledge on the many topics of interest we investigated. In fact, this thesis is composed of the following articles we published together:

Rolon-Mérette, D., Rolon-Mérette, T., & Chartier, S. (2018, July). Distinguishing Highly Correlated Patterns using a Context Based Approach in Bidirectional Associative Memory. In *2018 International Joint Conference on Neural Networks (IJCNN)* (pp. 1-8). IEEE.

Rolon-Mérette, D., Rolon-Mérette, T., & Chartier, S. (2019). Learning and Recalling Arbitrary Lists of Overlapping Exemplars in a Recurrent Artificial Neural Network. In *Proceedings of the International Conference on Cognitive Modelling* (pp. 186-191).

Rolon-Mérette, D., Rolon-Mérette, T., & Chartier, S. (2023). A multilayered bidirectional associative memory model for learning nonlinear tasks. *Neural Networks, 167*, 244-265.

Rolon-Mérette, D., Rolon-Mérette, T., & Chartier, S. (2023, May). Using Bidirectional Associative Memory Neural Networks to Solve the N-bit Task. In *The International FLAIRS Conference Proceedings* (Vol. 36).

Rolon-Mérette, D., Rolon-Mérette, T., Chartier, S. (2024). Are Associations All You Need to Solve the Dimension Change Card Sort and N-bit Parity Task? *In: Samsonovich, A.V., Liu, T. (eds) Biologically Inspired Cognitive Architectures 2023. BICA 2023. Studies in Computational Intelligence, Springer, 1130* (730-740).

Each article has been reformatted to ensure consistency and compliance with the demands of the

thesis. However, the content has not been altered and remains faithful to its original published version. For each article, I must highlight that I was the first author. My contributions ranged from creating the conceptual design of the research itself, performing the literature review, capturing data, testing the models, writing the scientific report, addressing the comments prior to publication, and conducting follow-ups. My work can be seen as a continuation of Dr Chartier's previous research on the bidirectional hetero-associative memory and the feature extraction bidirectional associative memory. What I have created is a new model that can overcome its predecessors' limitations while also being more consistent. It is a natural evolution of the RNAM class of ANNs and a new opportunity to investigate to what extent the mechanisms related to AM can explain human cognition.

Finally, as a disclaimer, although my efforts can be linked to the field of artificial intelligence and machine learning, they distance themselves by trying to respect the biological and cognitive constraints found in the human brain. As such, it does not try to achieve higher performances at any task or be at metahuman levels. Instead, it searches to be closer to how humans function. As such, this body of work is an attempt to model the link between information signal processing in networks of artificial neurons and human cognition. It is not a means of achieving an omnipotent artificial intelligent entity.

What follows is the fruit of this effort, as well as my thoughts on information signal processing, HL, associationism, RNAM, and human cognition.

General Introduction

The human brain contains over 99 billion neurons, all interconnected in elaborate recurrent networks [Herculano-Houzel, 2009]. By communicating with each other, these systems of information signal processing amount to the vast range of behavior found in cognition. Cognitive properties such as learning from exposure and storing that information in memory, crucial abilities for the dominance of the species, are all rooted in these recurrent networks. Much effort has been made to link these biological systems to cognitive properties and vice-versa [Bechtel & Bich, 2021; Dukas, 2004; Fitch, 2014; Wallace, 2014]. However, this spectrum of biology-cognition is still not well documented. The challenge comes partly from joining biological and cognitive mechanisms and explaining their connection. As such, theories on both ends of the spectrum must correlate to arrive at more robust conclusions.

A Promising Avenue Between Biology and Cognition

An interesting observation can be found in how neurons change their connections' efficacy and the associative nature of human cognition. Biological neurons tend to follow Hebbian learning (HL), a local rule that states that neurons that fire together wire together. Proposed by Donald Hebb in 1949, this learning theory emphasizes how the synaptic efficacy of interconnected neurons changes based on subsequent coactivation [Hebb, 1949, 2005]. In other words, the strength of connections is favored when e.g. presynaptic neuron A fires an action potential and its connected postsynaptic neuron B subsequentially follows the same trend. Consequently, when neuron A fires in a future instance, the probability of neuron B firing increases, suggesting a stronger connection forming between them. This type of mechanism holds for populations of neurons and even larger networks that are interconnected to others. To this day, this learning theory maintains its dominance, as new discoveries, such as Spike-timing-dependent plasticity (**STDP**), emphasize

how synaptic efficacy changes with timed coactivation between pre- and post-synaptic neurons [Caporale & Dan, 2008]. As such, it is believed that this simple mechanism based on contiguity would be one of the causes for these systems of information signal processing being able to learn and store information in memory.

At the cognitive end, associative memory (AM) states that thoughts, ideas, feelings, etc., i.e. mental states, that are encountered together associate together. This association results in a type of social solidarity between mental states, as when one of the associated pairs is evoked, its partner accompanies it from memory. This allows an individual to jump from one mental state to another and accomplish a wide range of conscious and unconscious goals. This theory, emanating from the school of associationism (see the following papers for an in-depth explanation of the history and findings of associationism: [Aristotle, 2001; Bain, 1868,1887; Anderson & Bower, 2014; Dacey, 2016, 2020; Hume, 1739, 1978; Locke, 1700, 1974; Mandelbaum, 2015]), has made it clear that humans can be seen as association machines and that cognition may be built on this simple principle. Like HL, associations are formed when mental states are coactivated subsequently. For instance, encountering or repeating “parrot is blue” can lead to “parrot” and “blue” becoming associated together. Mental states can represent individual items or categories and be decomposed into subcategories or regrouped into supercategories. Each mental state can be associated through different types of associations: one-to-one (**OOA**; i.e., one mental state evoking another), many-to-one (**MOA**; i.e., different mental states linked to the same one, e.g. categorization), and one-to-many (OMA; i.e., a single mental state paired to various other ones, e.g., enumeration). Through these types of associations, humans would be able to use them in complex manners and display various cognitive properties [Anderson & Bower, 2014; Davachi & Wagner, 2002; Giovanello et al., 2006; Hockley & Consoli, 1999; Mayes et al., 2007; Puig et al.,

2014; Yonelinas et al., 1999]. For instance, theories involving spreading activation and hierarchical structuring of memory [Collins & Loftus, 1975; Collins & Quillian, 1969] are good examples of this associative mechanism being applied and how it can be seen as a fundamental process in cognition.

To summarize, at both ends of the spectrum, mechanisms based on contiguity are seen for HL and AM. It would be of interest to try to link both HL and AM through information signal processing and explore how these can help fill the gap between the spectrum of biology and cognition. As AM constitutes a major part of human cognition, and since its role is still misunderstood [Anderson & Bower, 2014; Dacey, 2020; Mandelbaum, 2015, 2020], a promising avenue would be to focus on it and determine how mental states can be learned and represented in networks of interconnected neurons. By restricting this process in following HL, interesting conclusions can be made at the neurophysiological and cognitive levels. Particularly, it would help better explain the extent of the mechanisms related to AM in human cognition. As such, links between network architectures, the flow of information, associative behaviors, cognitive properties, and their limits can be further documented.

A Modern Way of Portraying Information Signal Processing

A recently growing and popular domain to study information signal processing in the human brain is computational modelling [Honey et al., 2010], particularly with artificial neural networks (ANNs). These formal models allow researchers to abstract the mechanism of biological neurons into axioms and use these mathematical expressions to explain how information signal processing can amount to cognitive properties [Coombs,1983]. ANNs can be seen as formal models that try to describe sequences of cognitively relevant events by using activation that spreads through sequential layers of nodes [Elman et al., 1996; Smolensky, 1988; Yang & Wang,

2020]. These nodes can be treated as "sub-representational" units of information that can correspond to a neuron, set or assemblies of neurons, etc. [Somolensky, 1988]. These ANNs have shown great explanatory value in a wide range of fields, including artificial intelligence, neuroscience, and psychology [Gupta, 2013; Kumar & Thakur, 2012; Ritter et al., 2017]. Of interest, they have been used as a vehicle to draw parallels between neurophysiology and cognition [Elman et al., 1996; Smolensky, 1988; Yang & Wang, 2020]. These models permit researchers to infer how networks of neurons and information signal processing can lead to different cognitive properties, such as AM, in a distributed manner [Alamia et al., 2020; Barak, 2017; Hintzman, 1991; McClelland, 2000a, 200b; O'Reilly & Munakata, 2000; Thomas & McClelland, 2008].

Recurrent Neural Associative Memories

Among the large family of ANNs, a particular subclass, known as recurrent neural associative memories (RNAMs), has shown promise in studying the relationship between neurophysiology and cognition through AM [Anderson, 1983; Palm, 2013; Zhang et al., 2021]. RNAMs have an interesting value for cognition, as they can link the neurodynamic perspective to AM. They bring a unique point of view of how learned patterns can correspond to attractors in the brain's neuronal state space. Consequently, they find themselves in various brain theories [Knoblauch, 2005]. Contrary to the machine learning (**ML**) approach [Mahesh, 2020], which focuses on performance exclusively, RNAMs usually opt for a more cognitively or biologically plausible approach. This means they will restrict themselves to using mechanisms grounded in the animal brain rather than applying non-biologically/cognitively plausible engineering tools. This makes RNAMs an interesting class of ANNs that can be used to better understand how HL, AM, neurophysiology, and cognition all fit together.

What are Recurrent Neural Associative Memories?

Similarly to ANNs, most RNAM also consist of neuron-like and synapse-like elements based on the McCulloch-Pitts neuron [McCulloch & Pitts, 1943]. RNAMs distinguish themselves by using content addressable memory, a type of computer memory that searches for pattern similarity among its stored content through a process akin to AM. This allows them to retrieve a corresponding address pattern despite being given a noisy one, which is a feature incorporated in most memory models in cognition [Kohonen et al., 1977]. Figure 1 sums the properties attributable to what constitutes a RNAM. RNAMs can usually perform either auto-associations (i.e., associating a pattern to itself), hetero-associations (i.e., associating a pattern to another one), or both. This makes them attractive when explaining various cognitive behaviors through AM (see Anderson and Bower's [2014] human associate memory model for details on how associations can be used to build cognition). As such, one of their many benefits is that they permit to explain how information signal processing can lead to human cognition through AM.

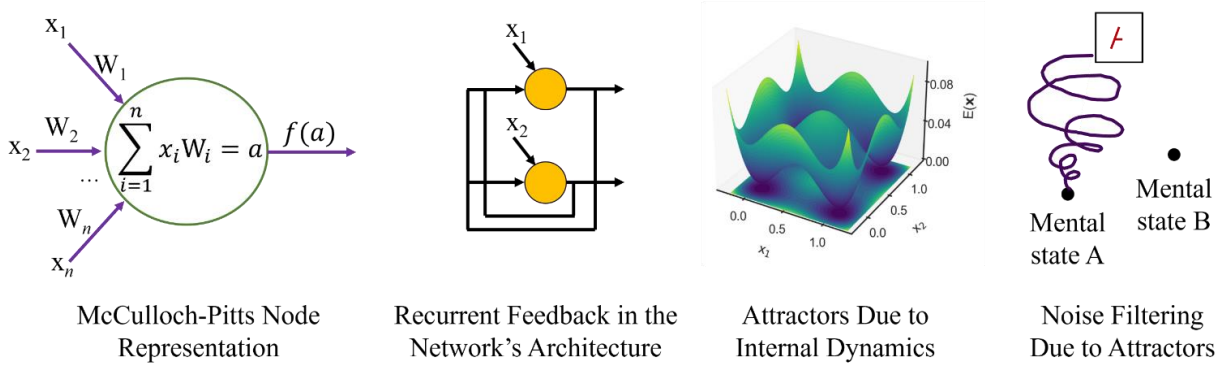

Figure 1 Attributable RNAM properties

History of Recurrent Neural Associative Memory

The history of RNAMs can be traced back to the 60s, following the arrival of the perceptron [Rosenblatt, 1958]. Early iterations were limited and far from displaying attractor and AM-like behaviors [Steinbuch, 1961; Willshaw et al., 1969]. These behaviors only began to emerge in the

70s, with networks being able to perform auto- and hetero-associations [Kohonen, 1972, 1974; Kohonen et al., 1977; Palm, 1980]. However, learned patterns had to be orthogonal (i.e., not correlated) for the model to function correctly. A further caveat of these early AM models was that they lacked the ability to form attractors due to their internal dynamics. This limited their explicability of human cognition. However, this internal dynamic behavior appeared with the brain-state-in-box (**BSB**) [Anderson et al., 1977]. The BSB could perform auto-association by storing patterns in attractors produced by nonlinear recurrent feedback within its network. This model was, in fact, what inspired the new subclasses of RNAM.

Interestingly, RNAMs only became popular a few years later due to the Hopfield network [Hopfield, 1982]. Of importance, this model showed how stimuli could be learned through a simple one-shot HL rule and proved its stabilization. However, the model could only display auto-association, limiting its generalizability to human cognition. A few years later, this limitation was overcome with the Bidirectional Associative Memory (**BAM**), which could perform both auto-association and hetero-association while retaining its neurodynamic perspective [Kosko, 1988]. That being said, the early iteration of the BAM was plagued with low memory capabilities, and its HL rule would display, at times, exploding weight values during learning. In a parallel effort, through the Eidos and nonlinear dynamic recurrent associative memory (**NDRAM**) model [Bégin, 1996; Chartier & Proulx, 2005], the HL rule was slightly modified to incorporate anti-Hebbian. This addition enabled the creation of the bidirectional heteroassociative memory (**BHM**) [Chartier & Boukadoum, 2006a, 2006b], a BAM model capable of learning correlated patterns while guaranteeing weight convergence. By combining Hebbian and anti-Hebbian, the model's learning behavior resembled more what was being observed in the human brain [Bégin & Proulx, 1996; Földiak, 1990; Lim, 2021; Murray & Escola, 2017; Storkey & Valabregue, 1999].

Other AM models using HL (see Brown et al., 1990, for a detailed review of HL rules) also appeared in the 80s and 90s. Models such as Temporal Associative Memory (**TAM**), Linear Associative Memory (**LAM**), Novelty Filter (**NF**), Multidirectional Associative Memory (**MAM$_1$**), Morphological Associative Memories (**MAM$_2$**) [Hagiwara, 1990; Hattori & Hagiwara, 1995, 1998; Margaritis, 1995; Ritter et al., 1998], and this just to name a few, all displayed some mechanism related to AM. However, to this day, the BAM remains one of the most popular RNAMs for studying AM while following the neurodynamic perspective. This is because it can perform auto- and hetero-association in a bidirectional fashion while using simple HL rules. These properties are essential aspects for generalizing an ANN to human cognition [O'Reilly, 1998]. As such, the BAM is an ideal framework to follow if one searches to understand better how biology and cognition can be expressed by an information signal processing system. However, current BAMs are still far from accounting for the wide range of cognitive and associative behaviors. One of the main goals of this body of work is to surpass current limitations and improve the BAM's framework. By doing so, a better understanding of how systems of information signal processing can lead to cognition will be possible.

Current Limitations of the Bidirectional Associative Memory

Over the years, numerous studies have enhanced the original BAM by allowing it to perform various complex tasks and be used for real-world applications. These include functions related to data compression, image recognition, and classification (see [Acevedo-Mosqueda et al., 2013] for a more comprehensive review of BAMs). As with ANNs in general, these advancements mainly came from two perspectives: those interested in performance exclusively (i.e., the ML approach) and those searching to build more plausible models of mechanisms found in the animal brain (i.e., the cognitive or biological approach). Although the ML perspective allowed the BAM

to be used in the aforementioned applications, it did so at the cost of losing some of its core properties related to cognitive plausibility: biological realism, distributed representations, error-driven learning, Hebbian learning, and bidirectional activation propagation [O'Reilly, 1998].

On the flip side, cognitively plausible models that try to follow these principles are often seen as either too complex [Labib, 1999], requiring highly nonlinear transmission functions [Shen et al., 2004], using subjective learning procedures [O'Reilly, 1996] or in the BAMs' case, limited when it comes to learning OMA and nonlinear relationships [Chartier et al., 2009]. These are quite cumbersome for the generalizability of the BAM, especially when considering that humans can associate patterns regardless of the type of association and the nature of their relationship being linear or nonlinear [Levering et al., 2020; Rosedahl & Ashby, 2022].

The Importance and Challenges of One-to-Many Associations

What to bring on a trip, the ingredients of a delicious recipe, the act of describing a topic, and recalling the steps to solve an important task, are all examples of day-to-day scenarios that illustrate the presence of OMA in human behavior. The ability to spring from one mental state to all its associated ones, as seen when enumerating the items of a category or list of actions to arrive at an answer, is a crucial building block for cognition. And yet, despite its importance, the BAM framework struggles to account for such an ability.

To be fair, ANNs are generally also poor at dealing with OMA. Although they can be considered universal function approximators [Csáji, 2001], they cannot inherently deal with relations. In other words, they cannot represent all the different observed human cognitive associated behaviors (see figure 2 for an illustrative example of the different types of associations). Because of this, different approaches were developed to overcome this problem. The two most dominant methods are the ones introduced by Elman (1990) and Jordan (1985, 1997). Both use

the notion of contextual information to transform the OMA into OOA (i.e., transform the relation into functions) and facilitate learning and recall. However, the form in which the context is portrayed differs between the two.

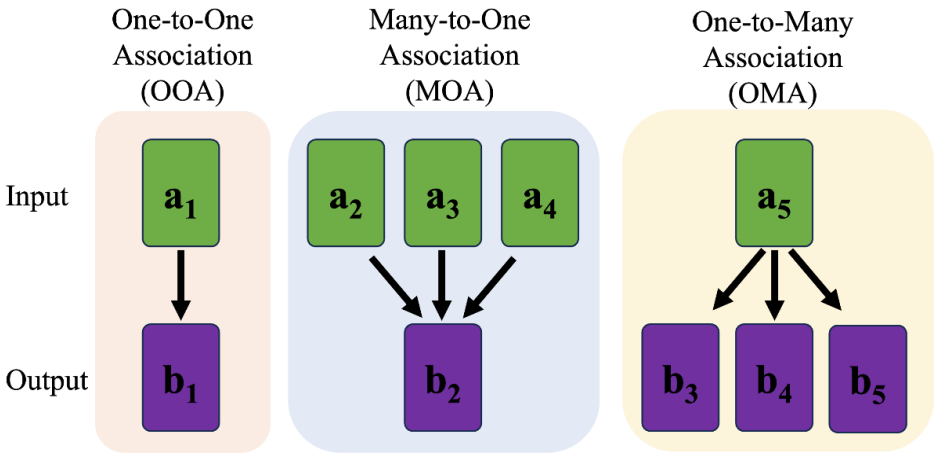

Figure 2 a) Example of one-to-one, many-to-one and one-to-many associations.

For Elman (1990), contextual information was viewed as a time window represented by time-delayed units. The content of these units would be integrated into the current input to accurately predict the following output (Collobert et al., 2011). The understanding of this method becomes intuitive when considering an example of a time series task containing an OMA. Take the following two sequences: African-Parrots-Gray and Brazilian-Parrots-Blue. Here, the OMA is described by the associations Parrots-Gray and Parrots-Blue. In this form, presenting only Parrots to an ANN will generally result in retrieving an average depiction of all associated stimuli. However, when presented with Parrots, Elman proposed to use the prior item of the sequence (or items, depending on the size of the contextual window) to take on the role of the context. This introduces a distinguishing element which transforms the OMA into a OOA. Thus, with Brazilian being in the time window, the answer in the presence of Parrots becomes Blue rather than an average or Red. Unfortunately, this solution of the time window (or surrounding context) requires the model to know the size of the window prior to hand. This means it needs access to global

information, which is not cognitively plausible. Furthermore, it becomes impossible to learn OMA that are present at the end of identical sequences (e.g., Brazilian - Parrots – Blue and Brazilian - Parrots – Red), as any window size fails to provide valuable information to distinguish the sequences.

For Jordan (1985, 1997), contextual information was seen as a label. For instance, items pertaining to a category would be "tagged" by a context representing their group. This label would become the distinguishing factor that would aid in dealing with the OMA. In the case of the previous example, "Brazilian - Parrots – Blue" and "Brazilian - Parrots – Red," a context representing sequence one and another for sequence two would be attached to each element of their respective sequence. This addition transforms the OMA into a OOA due to the contextual label being graphed to each stimulus and, in other words, modifying each item. Contrary to Elman's time window, this method permits dealing with identical sequences. Despite this transformation, the original representation can be kept. Furthermore, the solution to the OMA problem can be developed without any prior knowledge (i.e., no global information is required). Additionally, this method allows a model to retrieve every element of a specific sequence simply by presenting the initial label. This makes the method more akin to human behavior, as when faced with the initial examples (what to bring on a trip, the ingredients of a recipe, etc.), an individual can enumerate associated items simply by being presented with the question (context).

As it is well documented that context can influence learning and recall in humans by allowing for better differentiation between similar stimuli [Kushiro et al., 2013; Megill, 2014; Ren et al., 2013; Wurm & Schubotz, 2017], it would be of interest to incorporate this notion to the BAM framework. By following Jordan's methods, OMAs could be learned while avoiding the

need for global information and being closer to typical human associative behaviors. This addition would be a plausible and interesting solution to help cognitively driven BAMs deal with OMA.

The Importance and Challenges of Nonlinear Relationships

When considering the relationship between mental states in AM, one can observe their nature being either linear or nonlinear. This concept is highlighted when considering classification and categorization tasks [Jacob, 2004]. For example, when collecting wild mushrooms, each mushroom encountered will have characteristics that allow an individual to determine whether it is edible (e.g., the color, smell, and size) [Baroni, 2017]. In some instances, a single feature (foul smell) will be enough to determine that a mushroom is inedible, regardless of all other features (i.e., linear relationship). However, in most cases, many features must be considered simultaneously. Dealing with such situations often necessitates the ability to form nonlinear associations between features and their category. Humans can constantly perform these types of associations regardless of whether the nature of the relationship is linear or nonlinear. However, it remains an essential challenge for cognitively plausible BAMs. This is because a standard BAM's behavior will always be linear even when encountering nonlinear tasks (e.g., the XOR task). Thus, in the mushrooms example, such a model will always misidentify them due to its inability to learn nonlinear relationships between the features and the categories.

Different efforts have gone towards teaching the BAM to learn nonlinear tasks. These usually consist of modifying the BAM itself (e.g., changing its learning rules, architecture, transmission function, etc.) or pre-processing the inputs before being presented to the BAM (i.e., changing their representations). Of noticeable mention are the methods that use multilayered (i.e., deep) architectures with backpropagation and gradient descent prior to the BAM [Adigun & Kosko, 2019; Kosko, 2021; Liu et al., 2019; Rumelhart et al., 1986; Singh & Saraswat, 2017; Wu

et al., 2018]. However, these models are not necessarily concerned with following cognitive constraints and instead aim to solve tasks optimally [Marcus, 2018]. Furthermore, they often require different learning or transmission functions within the combined model while also having to freeze some of its modules during learning [Liu et al., 2019; Wu et al., 2018]. When modifying the BAM itself, certain conceptual shortcomings arise. For instance, in a recent effort to include multiple layers within its architecture and use a form of bidirectional backpropagation learning algorithm [Adigun & Kosko, 2019; Kosko, 2021; Singh & Saraswat, 2017], the method assumes that information can circulate through the same pre-post connection bidirectionally. Furthermore, its learning cycle uses an unrealistic flow of information to update its weights during learning, limiting its biological and cognitive plausibility.

Regardless of which approach above is used, they go against the current understanding of the human brain:

- Networks of neurons seem to behave similarly throughout the brain [Buckner & DiNicola, 2019], meaning that the same transmission function should be used for all nodes, and internal consistency must be sought out.
- No evidence of brain regions stopping their activity from accommodating learning [Raichle, 2010]. Thus, no weight-freezing schemes must be used.
- Information at the synapse level seems to circulate in one direction, so no transposition of weight matrices for backward passes can be utilized [Südhof, 2021].
- No strong evidence that the strength of synapses changes based on the error being backpropagated through them [Song et al., 2020]. Thus, the backpropagation learning algorithm cannot be used.

Despite being able to solve nonlinear tasks consistently, in their current form, these "deep networks" undermine the plausibility and weaken the legitimacy of these models to be used as a tool to mimic human cognition.

A different approach can be found in the morphological bidirectional associative memory [**MBAM**; Ritter et al., 1999], gray-scale morphological associative memories [Sussner & Valle, 2006], the alpha-beta bidirectional associative memory [**α-β BAM**; Acevedo-Mosqueda et al., 2006], chaotic complex-valued bidirectional associative memory [**CCBAM**; Yano & Osana, 2009], Lie Algebra-Valued BAM [**LieBAM**; Popa, 2017], and other variants (see [Acevedo-Mosqueda et al., 2013 for a chronological history of other relevant BAMs). These models have been built to deal with nonlinearity and have interesting properties, from the ability to increase storage capacity [Lee et al., 2015], display chaotic behaviors, to using drastically fewer resources (artificial nodes) in the process [Sussner & Campiotti, 2020]. However, these models do so by increasing the complexity, either by changing the nature of the artificial node or using different algebra types for its computations. As such, they move away from the traditional view of the McCulloch-Pitts' neuron and the simple neurodynamic perspective captured by the original BAM.

An important question to be answered is whether only changing the connectivity is sufficient to learn nonlinear tasks while keeping simple HL and the McCulloch-Pitts' neuron model. One approach that searched to investigate this can be found in [Chartier & Boukadoum, 2006a, 2006b]. This modified version removed the original notion in Koskos' BAM [1988], where the bidirectional connections between two units had to be symmetric. Instead, it used two distinct weight matrices to account for the independence of the forward and backward connections. In this modified BAM, a simple Hebbian and anti-Hebbian learning rule is used. From this model, an unsupervised version was proposed by replacing one set of connections with its generated

representation [Chartier et al., 2007; Chartier et al., 2012]. This variation called the Feature Extraction Bidirectional Associative Memory (**FEBAM**), has been shown to learn nonlinear associations if combined with another FEBAM in a head-to-toe fashion (it solved the 2-bit, 3-bit, 4-bit, and 5-bit parity problem). This was evidence that learning these types of associations was possible by only changing the flow of information. In other words, success is achieved without undermining the model's internal consistency while maintaining the modified BAM's core properties [Chartier et al., 2012]. However, this method was only applicable if bipolar values represented the task and, as such, did not generalize well to continuous values. Nevertheless, the FEBAM can be considered an interesting model, as it has the property to generate patterns associated with the original inputs in an unsupervised manner but of different dimensionality to them. Thus, the FEBAM can generate a version of the initial patterns that are either compressed, expanded, or the same size [Giguère et al., 2007a, 2007b]. In other words, this means that the FEBAM could be used as a type of homogenous pre-processing network.

In fact, this was explored in a study that combined the FEBAM and the BAM as a multinetwork model [Tremblay et al., 2013]. Here, the FEBAM was used to generate a higher dimensionality representation that was joined with the original input to make the task linear. This FEBAM-BAM implementation proved interesting as it could learn the N-bit parity problem and other real-valued (continuous) nonlinear tasks (e.g., the double Moons). However, the downside of this multinetwork model was that its learning was erratic and required an external stopping mechanism to halt its weight update. Additionally, even with this stopping mechanism, the model could only find the right solution a fraction of the time. As such, although the FEBAM-BAM approach had internal consistency, it lacked reliability and the capacity to self-stabilize.

Still, the combination of FEBAM and BAM may prove to be an interesting avenue in dealing with nonlinear relationships. By allowing the joint networks to work together and by only modifying the connectivity of one of them, it was possible to display new behaviors. This multi-model gets closer to being analogous to information circulating through different brain regions (neural circuitry) and being transformed along the way [Wang & Cui, 2018]. It could be possible to expand this transformation by using multiple FEBAMs in a head-to-toe fashion. This could combine the reliability and strength of deep networks, as well as the plausibility and homogeneity of the joint FEBAM-BAM model. As such, only changing the connectivity could be an inherent explanation of how these homogenous systems can display more complex behaviors. Suppose the BAM is to remain the sought-out framework to model human cognition through the neurodynamic perspective, then it must also be able to learn linear and nonlinear relationships.

Solving One-to-Many Associations and Nonlinear Relationships

To address the issue of OMA and nonlinear relationship, we propose to use contextual information and introduce a novel homogenous multi-model, inspired by the brain's laminar architecture, to the BAM framework. This effort aims to ensure a more biological and cognitive plausible approach in maintaining the aforementioned properties: biological realism, distributed representations, error-driven learning, Hebbian learning, and bidirectional activation propagation [O'Reilly, 1998]. By overcoming current limitations while respecting imposed constraints, the BAM framework can be used to document better how information signal processing in the brain, AM and cognition all fit together. By using context and the new model, a deeper understanding of learning and memory and the benefits and limitations related to the mechanisms of contiguity can be expanded upon. Ultimately, this aids in filling the gap between the polar end of the biology and cognition spectrum.

Overview of Chapters

The expansion of the BAM framework in regard to solving OMA and nonlinear relationships and the implications of such solutions are investigated across three chapters:

Chapter 1 introduces the addition of contextual information to overcome OMAs. This implementation followed Jordan's approach to modifying the learned patterns by concatenating contextual representations. The implication of such modification was tested by measuring the inter and intra-correlations between groups of correlated patterns after modification. Additionally, the ability to deal with OMA was evaluated through a time series task containing multiple overlapping OMA.

Chapter 2 introduces a novel multi-network BAM to deal with nonlinear tasks. To achieve this, multiple FEBAMs were joined in a head-to-toe fashion to act as a homogenous pre-processing unit prior to the BAM. The relationship between architecture (flow of information) and performances of various nonlinear tasks was investigated through decision zones, learning times and correct recall. Several additional properties pertaining to consistency, online continuous learning, and multiclass tasks were also evaluated.

Chapter 3 investigates the advantages of using context and the new multi-network model to deal with complex tasks. A combination of two multi-network models was used to learn the N-bit and the Dimension Change Card Sort (**DCCS**) task while focusing on using more human-like associative behaviors. The joint model was compared to the traditional ML approach using rote association through learning time, recall performances and overall behavior.

Chapter 1

Chapter 1 introduces the addition of contextual information to overcome OMAs in the BAM framework. This chapter divides itself into two parts. In the first part, the paper titled *Distinguishing Highly Correlated Patterns using a Context Based Approach in Bidirectional Associative Memory*, highlights the effects of modifying input patterns by adding contextual representations through the Jordan approach. Manipulation of the size and nature of the contextual representations was performed to understand better the consequences of these elements on learning and recall performances. In the second part, the publication titled *Learning and Recalling Arbitrary Lists of Overlapping Exemplars in a Recurrent Artificial Neural Network* investigated how the knowledge acquired from the previous paper could be used to learn different overlapping sequences containing multiple OMAs.

Distinguishing Highly Correlated Patterns using a Context Based Approach in Bidirectional Associative Memory

Publication reference:

Rolon-Mérette, D., Rolon-Mérette, T., & Chartier, S. (2018, July). Distinguishing Highly Correlated Patterns using a Context Based Approach in Bidirectional Associative Memory. In *2018 International Joint Conference on Neural Networks (IJCNN)* (pp. 1-8). IEEE.

Abstract

Artificial neural networks can store stimuli of different types. However, if these are highly correlated, this becomes problematic. Pairs of stimuli that are very similar will have longer learning times and lower noise tolerance. To remedy this problem, this study proposes to use context (C) juxtaposed to a pattern before being presented to the network. Thus, by modifying the correlation of the initial stimuli, a more efficient encoding will be achieved. A two-part experiment was performed to examine the benefit of incorporating cognitive contexts into different sets of strongly or weakly correlated patterns in a bidirectional associative memory. Results from the first experiment showed that in the presence of Cs who reduced the correlation of the initial patterns, the learning time decreased up to 96%, while the tolerance to noise during recall increased by 50%. Furthermore, by increasing the relative proportion of the Cs compared to the initial patterns, an additional 30% noise tolerance for medium noise and 80% noise tolerance for high levels of noise was achieved. Also, a reduction of 50% in learning time was observed. Therefore, the addition of contexts can greatly improve the learning of similar patterns in artificial neural networks in associative memories.

Keywords—Bidirectional associative memory, Learning, Context, Correlated patterns, Top-down processes.

I. Introduction

A lot of effort has been put into better understanding the mechanisms of learning and memory. There seem to be many factors that can influence these mechanisms, thus making it hard to fully comprehend. Factors such as expertise, attention and emotions have already shown that they may influence learning and memory [Kushiro et al., 2013; Megill, 2014; Ren et al., 2013]. Past studies have also shown evidence of a connection between these factors and the mechanisms behind learning and memory, but how they may influence these cognitive properties remains an open question. This paper investigates the possible properties that these factors may share in relation to learning and memory. One basic concept of memory is the one of association [Mahani et al., 2016]. AM consists of storing identical (auto-association) or different patterns (hetero-association). If the association is successful, then it can be recalled when using partial cues. Alternatively, in the presence of a novel pattern, the closest one will be given. This is a process akin to categorization. Categorization can be summarized as grouping similar patterns based on their proximity in space [Berger et al., 2006]. It is consistent throughout human lifespan and serves as an efficient way of storing and retrieving information [Gelman & Meyer, 2011].

It is well-documented that context can influence learning and recall [Wurm & Schubotz, 2017]. In environments where context allows for better differentiation between similar stimuli, the performance of learning and recalling should increase. Thus, it is a promising path to investigate the effects of context on learning. Furthermore, context is not exclusively external; it can also be internal. Emotions and other cognitive processes may act as context. When looking at emotions, there is already evidence linking them to cognitive mechanisms such as attention, decision-making and learning [Kushiro et al., 2013; Megill, 2014; Ren et al., 2013]. It is clear that there are various players involved in the mechanisms of learning and memory. Interestingly, it is possible that they

may all share similarities to the mechanisms found in context. This is plausible since it is known that the human brain has top-down functions that can modify perceived stimuli and aid in decision-making [Iacoboni, 2008; Kadel et al., 2017].

In cognition, artificial neural networks (ANNs) are the best candidates to model cognitive processes. Many models can mimic associative memories [Acevedo-Mosqueda et al., 2013]. Despite their ability to show associative learning, they are faced with a challenge when learning highly correlated patterns. The higher the correlation between the patterns, the longer it will take to store them. Moreover, in cognition as well as is ANNs, highly correlated patterns will also increase the likelihood to output the incorrect association under noise [Werker et al., 2002; Yoshida et al., 2009]. This is illustrated in Figure 3, where the results of a preliminary test looking at correlation value between patterns during an associative learning task shows a decrease in performances for highly correlated patterns.

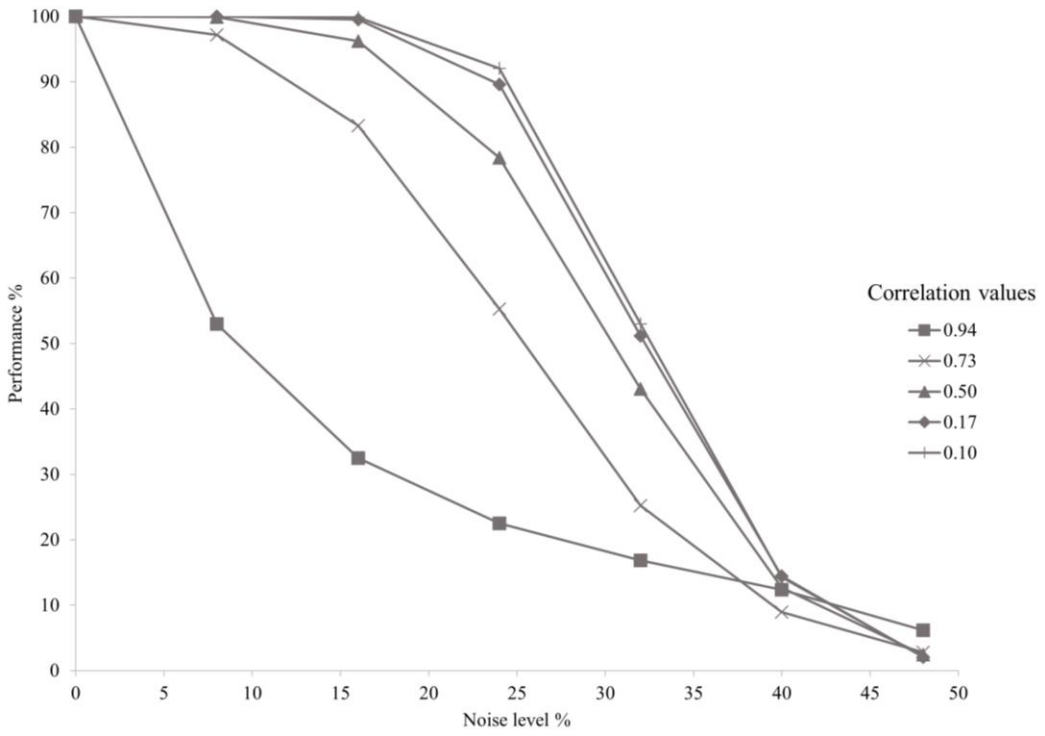

Figure 3. Performance across noise levels for different correlation values.

A solution to decorrelate input patterns without losing the original representation could reside in the addition of context. In a situation where irrelevant information towards categorizing patterns is predominant, learning could be facilitated by adding a distinguishing component and/or by increasing its relative impact. This could be seen as a general cognitive mechanism relying on contextual information to distinguish correlated stimuli. Therefore, we introduce arbitrary context (**C**) that can modify the relationship between the input patterns without losing the original representation. This should result in a better recall performance and a shorter learning time. Since the desire is to decorrelate input patterns without losing the original representation, principal component analysis and other feature extraction methods are excluded from this approach.

A two-part experiment was conducted using a modified Bidirectional Associative Memory (BAM) [Chartier & Boukadoum, 2006; Chartier & Boukadoum, 2011; Chartier et al., 2009] with the goal of decorrelating input patterns without losing the original representation. This paper is divided in the following fashion: In section II a brief background of the proposed model will be presented. In section III, a first set of simulations investigates the effects of various levels of correlation for C on highly and lowly correlated patterns. In Section IV, a second set of simulations investigates the effect of relative importance (dimensionality) of the context on highly correlated patterns. Finally, section V ends the paper with a short discussion and conclusion.

II. Model

A. Model Description

The model used is inspired by Kosko's BAM [Kosko, 1988]. It can associate inputs and recall them under various levels of noise. A modified BAM network is used [Chartier & Boukadoum, 2006; Chartier & Boukadoum, 2011] as it has the particularity of having a unique

matrix for each layer of units (Figure 4). As with any ANN, this model can be entirely described by its architecture, transmission and learning functions.

B. Architecture

The architecture is illustrated in Figure 4. The model has two layers of interconnected units in a bidirectional fashion, where $\mathbf{x}_{[0]}$ and $\mathbf{y}_{[0]}$ represent the initial patterns, **W** and **V**, the weights for the connection matrices, *m* and *n* the number of units in each layer and, $\mathbf{x}_{[c]}$ and $\mathbf{y}_{[c]}$ the output of the network after *c* iterations.

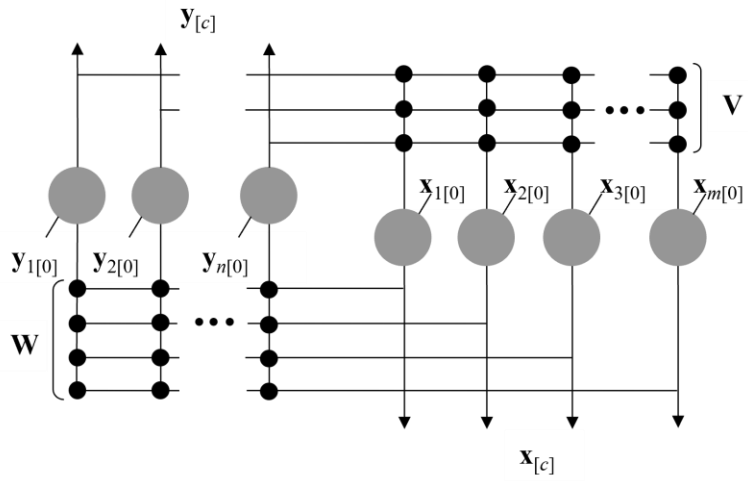

Figure 4. BAM's architecture.

C. Output Function

The transmission function is defined by equation 1a and equation 1b:

$$\text{Equation 1a} \quad \forall i, \dots, n, \mathbf{y}_{i[c+1]} = f\left(\mathbf{a}_{i[c]}\right) = \begin{cases} 1, if\ \mathbf{a}_{i[c]} > 1 \\ -1, if\ \mathbf{a}_{i[c]} < -1 \\ (\delta + 1)\mathbf{a}_{i[c]} - \delta \mathbf{a}_{i[c]}^{3}, Else \end{cases}$$

$$\text{Equation 1b} \quad \forall i, \dots, m, \mathbf{x}_{i[c+1]} = f\left(\mathbf{b}_{i[c]}\right) = \begin{cases} 1, if\ \mathbf{b}_{i[c]} > 1 \\ -1, if\ \mathbf{b}_{i[c]} < -1 \\ (\delta + 1)\mathbf{b}_{i[c]} - \delta \mathbf{b}_{i[c]}^{3}, Else \end{cases}$$

Where *n* and *m* are the total number of units in each layer, *i* is the index unit, δ is the general transmission parameter and **a** and **b** are the activations. These activations are obtained the usual way and are defined by equation 2a and equation 2b:

$$\text{Equation 2a} \quad \mathbf{a}_{[c]} = \mathbf{W}\mathbf{x}_{[c]}$$

$$\text{Equation 2b} \quad \mathbf{b}_{[c]} = \mathbf{V}\mathbf{y}_{[c]}$$

Figure 5 shows the transmission function for $\delta = 0.2$.

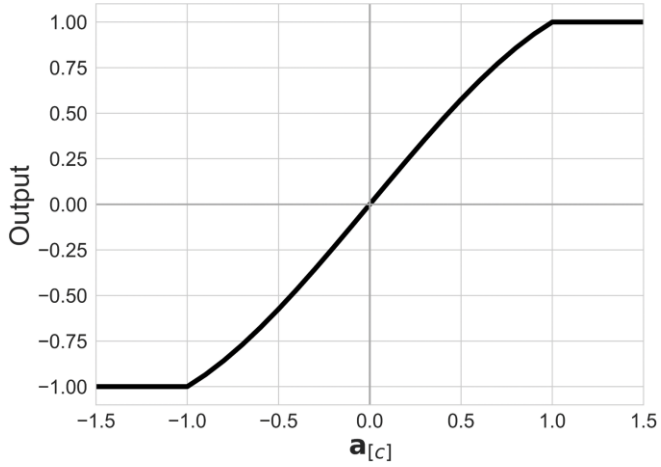

Figure 5. Transmission function for δ = 0.2.

D. Learning Rule

The connection weights are modified following a hebbian/anti-hebbian rule ([Bégin & Proulx, 1996; Storkey & Valabregue, 1999])

$$\text{Equation 2a} \quad \mathbf{W}_{[k+1]} = \mathbf{W}_{[k]} + \eta\left(\mathbf{y}_{[0]} - \mathbf{y}_{[c]}\right)\left(\mathbf{x}_{[0]} + \mathbf{x}_{[c]}\right)^{\mathrm{T}}$$

$$\text{Equation 3b} \quad \mathbf{V}_{[k+1]} = \mathbf{V}_{[k]} + \eta\left(\mathbf{x}_{[0]} - \mathbf{x}_{[c]}\right)\left(\mathbf{y}_{[0]} + \mathbf{y}_{[c]}\right)^{\mathrm{T}}$$

Where $\mathbf{x}_{[0]}$ and $\mathbf{y}_{[0]}$ are the initial inputs, η is the learning parameter and *k* is a given learning trial. Equations 3a and 3b show that the matrix weights will converge only when $\mathbf{x}_{[0]} = \mathbf{x}_{[c]}$ or $\mathbf{y}_{[0]} = \mathbf{y}_{[c]}$.

It is guaranteed that the learning will converge if the learning parameter (η) is smaller than the following value defined by equation 4 [Chartier & Boukadoum, 2006]:

$$\text{Equation 4} \quad \eta < \frac{1}{2(1-2\delta)\text{Max}[M,N]}\ , \delta \neq \frac{1}{2}$$

III. Simulation I

To understand how cognitive strategies influence storage of arbitrary patterns, C and patterns templates were generated with various levels of correlation. An instance from each template (pattern and C) were juxtaposed together to form a new input. This is repeated for all the patterns before heteroassocative learning. The network was then tested through recall under various levels of noise.

A. Methodology

In order to replicate the effects of learning similar and dissimilar patterns, two sets of 16 patterns with high ("**H**") and low ("**L**") levels of correlation were produced. Each pattern was an 8X8 image flattened into a vector of 64 dimensions. A black pixel was given a value of 1, and a white pixel a value of -1. In the "H" set, each individual pattern had a correlation value above 0.75 with every other pattern from the set (see Figure 6). In the "L" set, each individual pattern had a correlation value below 0.25 with every other pattern from the set.

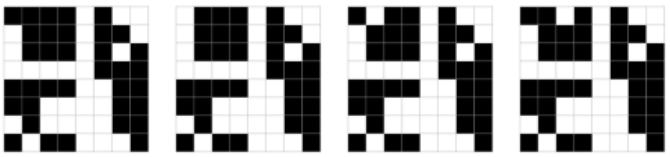

Figure 6. Example of 8X8 patterns for the H set.

To create a distinguishing feature and mimic the possible effects of context on learning, two sets of 16 Cs of dimension 64 were used. The first set contained 16 orthogonal patterns

(orthogonal C; Figure 7), while the second one contained correlated patterns. Figure 8 illustrates some examples of the new input composed of the original highly correlated patterns and the orthogonal contexts. From the manipulation of both the correlation within a set and between a set, four conditions were extracted. Figure 9 displays all correlation manipulations, and Table 1 describes each condition applied to the highly correlated patterns ("H") and lowly correlated patterns ("L"). These conditions were compared to a baseline, one for each set of patterns ("H" and "L"). These baseline conditions had their dimensionality equal to the new inputs (16X8) and presented the same correlation as patterns from the "H" set and "L" set. Figures 10, 11, 12 and 13 illustrate both the patterns and input correlation for each of the four conditions on the H set.

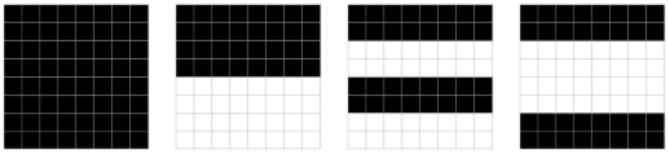

Figure 7. Example of 8X8 orthogonal C.

B. Associative Task

Learning was done according to a simple one-to-one associative task where one input (highly or lowly correlated patterns and C) from a category (category "a") was associated with another input from a different category (category "b"). The addition of C impacted on correlation on two levels: pattern wise and input wise. The patterns correlation ($\boldsymbol{r_{patterns}}$) refers to the correlation found between all the patterns of a category and themselves. The inputs correlation ($\boldsymbol{r_{inputs}}$) refers to the correlation between inputs from category "a" and the associated inputs from category "b".

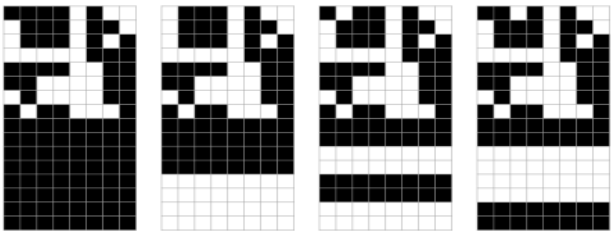

Figure 8. Example of newly formed input by juxtaposition of the patterns from the H set with orthogonal C.

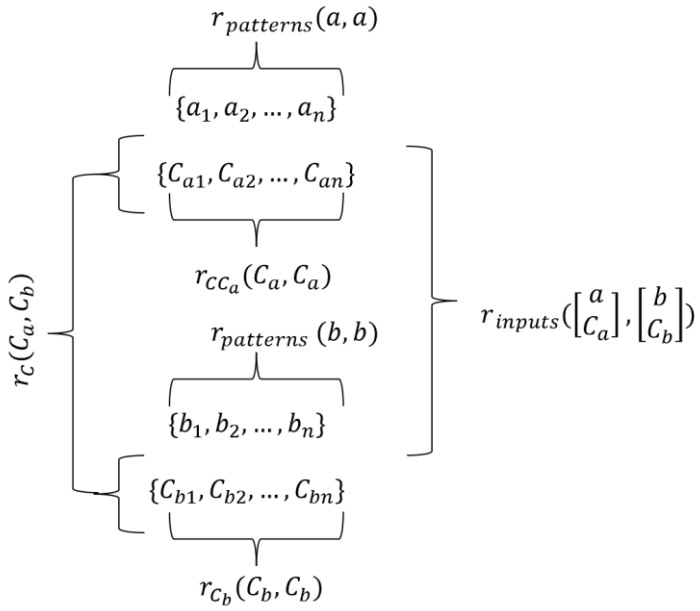

Figure 9. Correlation manipulation for every relationship between category "a" and category "b".

C. Pairing Condition

Four different pairing conditions were created (see Table 1) and were applied to both "H" and "L" set. Condition 1 has an orthogonal C juxtaposed to the patterns from category "a" and category "b" (Figure 10). In this condition, every C juxtaposed to the patterns are orthogonal, even the associated inputs have a different orthogonal C between them, allowing to lower both $r_{patterns}$ and r_{inputs}. Condition 2 controls for the relationship of juxtaposing orthogonal Cs to both categories, but associated inputs have the same orthogonal C (Figure 11). This lowers the $r_{patterns}$

while increases the r_{inputs}. Condition 3 controls for juxtaposing the same C to all the patterns from a category while adding a single orthogonal C to all patterns from the other category (Figure 12). This increases the $r_{patterns}$ while lowers the r_{inputs}. Finally, condition 4 controls for the juxtaposition of the same C to all the patterns from both categories (Figure 13). This increases both the $r_{patterns}$ and r_{inputs} . Table 2 summarizes the effects of C on the $r_{patterns}$ and the r_{inputs} for each condition.

Table 1

Correlation value, pairing conditions for the H, L sets and correlation value for each C relationship.

$r_{patterns}$	Pairing Conditions	r ($C_{a,b}$, $C_{a,b}$)	r (C_a,C_b,)	Labels
>0.75	1	0	0	H1
>0.75	2	0	1	H2
>0.75	3	1	0	H3
>0.75	4	1	1	H4
<0.25	1	0	0	L1
<0.25	2	0	1	L2
<0.25	3	1	0	L3
<0.25	4	1	1	L4

Table 2

Effect of C on correlation between inputs and associated inputs for pairing condition of H and L set.

Condition	$r_{patterns}$	r_{inputs}
1	Decreased	Decreased
2	Decreased	Increased
3	Increased	Decreased
4	Increased	Increased

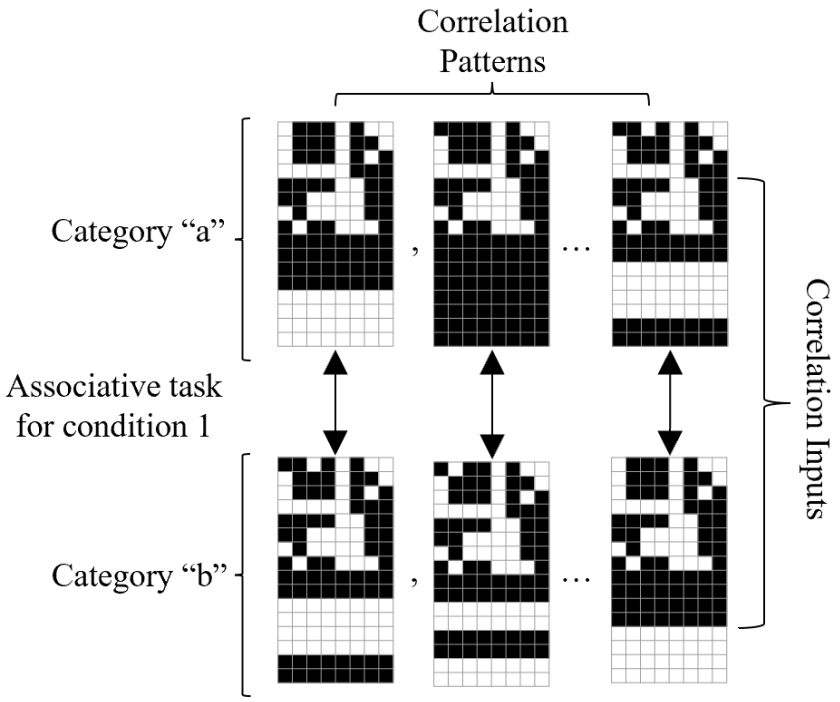

Figure 10. Example of association task for the H set juxtaposed to an C for condition 1.

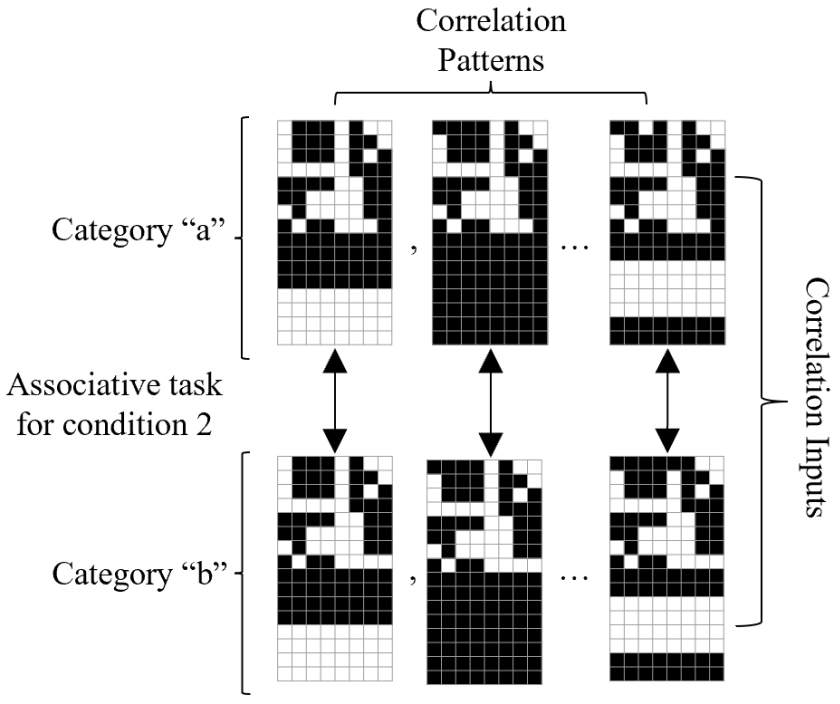

Figure 11. Example of association task for the H set juxtaposed to an C for condition 2.

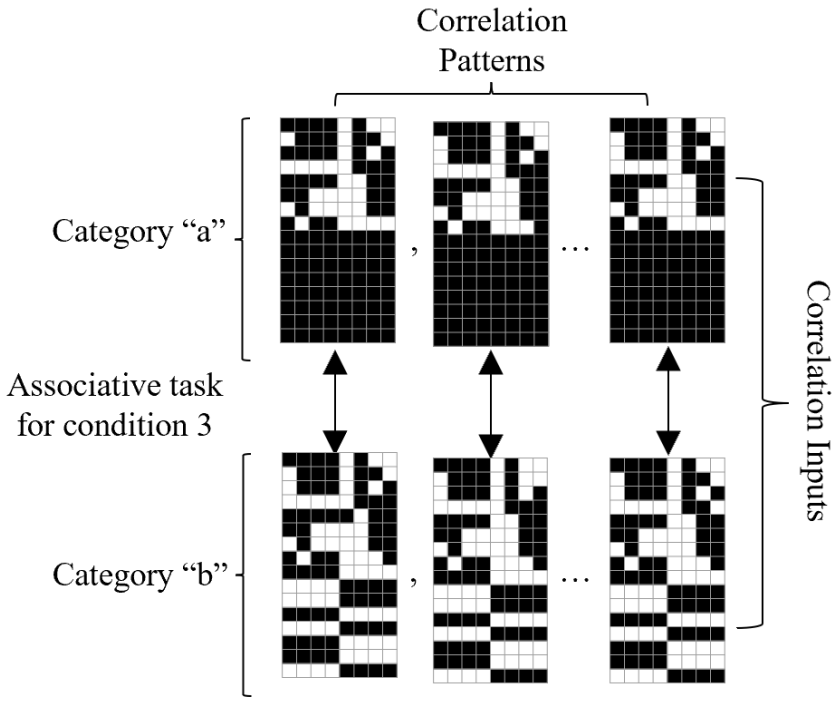

Figure 12. Example of association task for the H set juxtaposed to an C for condition 3.

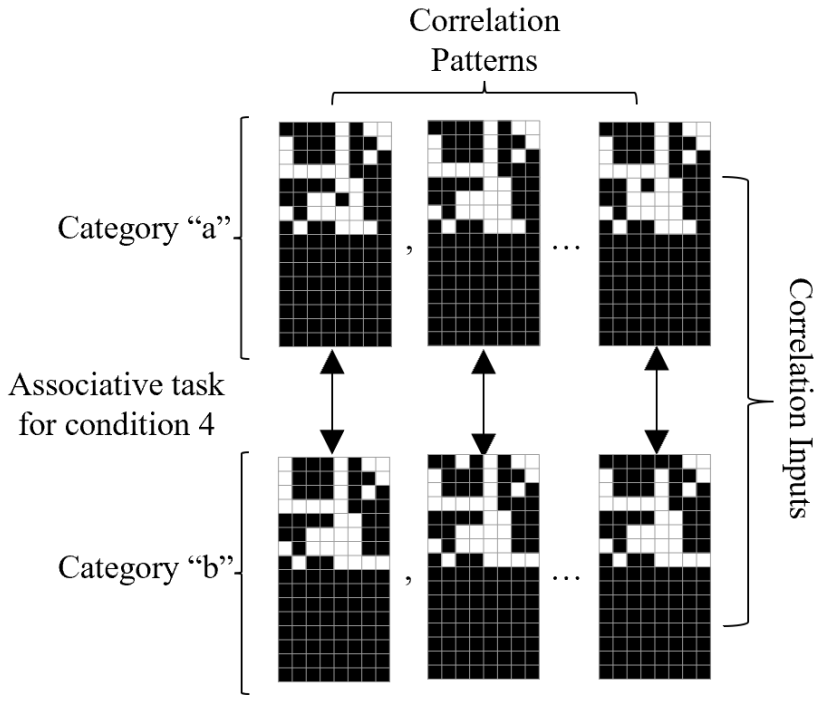

Figure 13. Example of association task for the H set juxtaposed to an C for condition 4.

D. Procedure

The transmission function's parameter (δ) was set to 0.2 and the value for the learning parameter (η) respected equation 4 for all the simulations. During the learning, the number of cycles before the weights update was set to c =1. The learning was stopped when the mean squared error (**MSE**) was lower than 10^{-15}. For the recall task, the pixel flip percentage varied from 0 to the theoretical limit of 50%.

The learning phase was done following this procedure:

1. Creating a random list containing patterns and Cs following the conditions priory mentioned.
2. Pairing inputs using the list created while following the conditions previously mentioned.
3. Randomly choosing a pair from the list $\mathbf{x}_{[0]}$ and $\mathbf{y}_{[0]}$.
4. Calculate $\mathbf{x}_{[1]}$ and $\mathbf{y}_{[1]}$ by using the transmission function from equation 1a and 1b.
5. Calculate the new connection weights according to equation 2a and 2b
6. Repeat step 3) and 5) until all the pairs are selected.
7. Repeat step 3) to 6) until the desired MSE is reached.

Recall task was done following this procedure:

12. Selection of an input from the learned list.
13. Random pixels flips are introduced to form a noisy input pattern $\mathbf{x}_{[0]}$
14. Compute $\mathbf{y}_{[c]}$ in accordance to the transmission function of equation 1a and 1b until convergence.
15. Comparison of the final output with correct one and calculate the performance
16. Repeat steps 2) to 4) for 200 trials.
17. Repeat 1) to 5) for each learned stimulus
18. Repeat 1) to 6) with increasing ratio of random pixel flips (from 0 to 50%).

E. Results

The learning times for "H" and "L" sets in all four conditions are compared to their respective baseline and are indicated in epochs (see Table 3). Values below 1 for the comparison between a condition and the baseline are indicative of faster learning time. In contrast, values above 1 are interpreted as longer learning time. Results can be regrouped into four Behaviors. The first is seen for the "H" set in conditions 3 and 4. Learning times were more than doubled. Both are found to have their $r_{patterns}$ increased after the juxtaposition of the C. The second behavior is seen for "H1" and "H2". Learning took 95% less time for both inputs, they had their $r_{patterns}$ lowered after the juxtaposition of the C. The third behavior observed regrouped "L3" and "L4". Learning times took roughly 90% longer because their $r_{patterns}$ increased after the juxtaposition of the C. Finally, the fourth behavior regrouped "L1" and "L2". Both took 50% less time when they had their $r_{patterns}$ lowered after the juxtaposition of the C. In other words, in conditions ("3" and "4") where the C increased $r_{patterns}$, learning times were longer compared to baseline regardless of the pattern set. In contrast, in conditions ("1" and "2") where the opposite correlation effect is seen ($r_{patterns}$ is lowered), learning times were much faster than the baseline. Thus, in the best scenario, it is possible to lower the learning time for highly correlated patterns by 94% and by 50% for lowly correlated patterns simply by juxtaposing an orthogonal C.

Table 3

Comparison of learning times in epoch between all conditions and baselines during Simulation I.

	L set		**H set**	
Condition	Learning time	Condition/ baseline	Learning time	Condition/ baseline
Baseline	40	1	585	1
1	20	.50	34	0.058
2	20	.50	34	0.058
3	75	1.88	1321	2.26
4	76	1.90	1338	2.28

Results for recall performances in function of different levels of noise for the “H” and “L” sets under each of the four conditions are illustrated in Figure 14. Noise level is defined as the percentage of pixels flipped. Results were compared at low (10%), medium (20%) and high (30%) noise levels. Four types of behaviors are observed. The first regroups “H3” and “H4. They had the worst recall performance. The second behavior observed regroups “H1” and “H2”. Both showed a significant increase in performance compared to baseline “H”. It is important to note that they both shared a decrease in $r_{patterns}$ after the juxtaposition of the C. Interestingly, “H1” outperformed “H2” in a considerable fashion. “H1” differed from “H2” by having its r_{inputs} lowered instead of increased by the addition of the C. The third behavior observed regrouped “L3” and “L4”. They had better performances compared to all H conditions but displayed lower performances compared to baseline “L”. For both, their $r_{patterns}$ was increased by the juxtaposition of the C. Finally, the fourth behavior regroups “L1” and “L2”. Both displayed the highest overall performance levels. As for “H1” and “H2”, they had their $r_{patterns}$ lowered. In short, adding an orthogonal C that lowers the $r_{patterns}$, as seen in conditions “1” and “2”, increases the tolerance to noise degradation. Whereas increasing this correlation with a juxtaposition of correlated C results in worse performances during recall regardless of the set of patterns.

IV. Simulation II

Adding an orthogonal C to a highly correlated pattern was shown to be effective in lowering learning time and increasing recall performances. However, the relative “importance” of C can also have an impact on learning. In other words, increasing the dimensionality of the C while keeping the dimensionality of the patterns constant will also reduce the correlation and the memory load; therefore, it should be beneficial to the network.

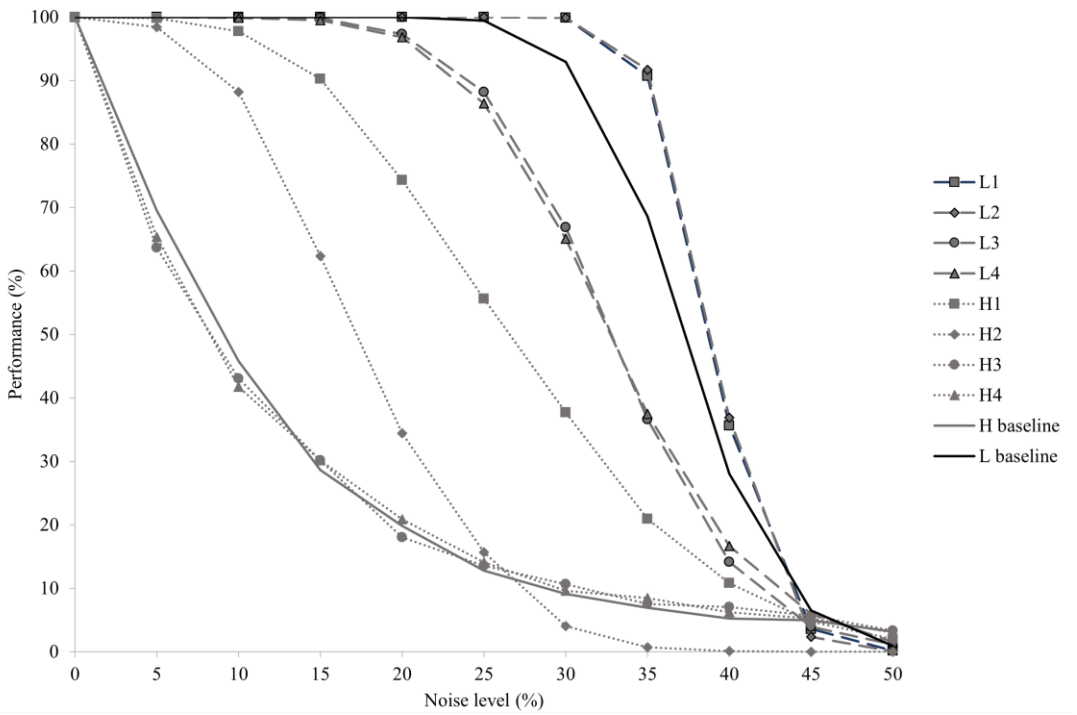

Figure 14. Results of all four conditions for the "H" and "L" with their respective baseline for the recall test. Performance decrease in function of the noise level.

A. Methodology

The same 16 patterns (8x8 dimension) for the "H" set, previously generated in simulation I, were used. Condition "1" and condition 4 were used but dimensionality of the C varied. As a reminder, condition 1 refers to juxtaposing different orthogonal C to the patterns to decrease both $r_{patterns}$ and r_{inputs} while in condition "4", all patterns were juxtaposed with the same C to increase both $r_{patterns}$ and r_{inputs}. With the pattern size kept constant (dimensionality of 64) and C size varying, relative importance could be studied by manipulating the ratios of patterns and C. Those ratios of pattern/C were: 4/1, 2/1, 1/2, 1/4 and 1/8. Figure 15 shows an example of for H1. Results were compared to a baseline ratio of 1/1 for "H1" and "H4".

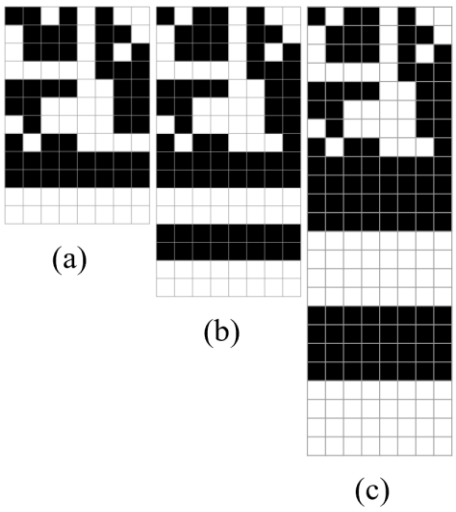

Figure 15. Examples for H1 with dimensionality kept constant (64) paired with: (a) C of 32, (ratio 2/1), (b) C of 64 (ratio 1/1) and (c) C of 128 (ratio 1/2).

B. Results

Table 4 shows the average learning times in epochs for the baseline and every ratio of patterns/C for “H1” and “H4”. Values below 1 for the comparison between ratios and baselines are indicative of faster learning time. In contrast, values above 1 are interpreted as longer learning time. Results show two types of behaviors. The first regroups all the ratios for H4, such as 4/1 to 1/8. Ratios 4/1 and 2/1 displayed faster learning times compared to baseline (44% faster and 30% faster, respectively). However, as the C increased in size, the longer it took the model to learn. Ratios of 1/2, 1/4 and 1/8 displayed longer learning times compared to baseline (ranging from 60% to 6 times longer). The second behavior is seen for all the ratios of “H1”. In presence of ratios of 4/1 and 2/1, learning times were up to two and a half times longer compared to baseline. Finally, when the dimensionality of C was higher than the pattern, as seen in ratios 1/2, 1/4 and 1/8, learning time was twice as fast compared to baseline.

In sum, results for "H4" showed that as the size of the C increased, the longer it took the model to complete the learning task. In contrast, for "H1", as the size of the C increased, the model was able to learn faster. Thus, in the best-case scenario, adding an orthogonal C in presence of highly correlated patterns can decrease learning times by 94% compared to the baseline H. In addition, by increasing its size, the learning can be accomplished 97% faster than the baseline H.

Table 4

Comparison of learning times in epochs between baseline 1/1 ratio and every other ratio for H1 and H4.

	H 1		**H4**	
Ratio Pattern/C	Learning time (epochs)	Ratio/ baseline	Learning time (epochs)	Ratio/ baseline
Baseline (1/1)	34	1	1338	1
4/1	86	2.53	745	0.56
2/1	51	1.59	934	0.70
1/2	24	0.71	2146	1.60
1/4	19	0.56	4022	3.01
1/8	17	0.50	8340	6.23

Figure 16 shows the performance in function of noise during recall. Without surprise, the performance decreases as the level of noise increases for all the ratios of "H1" and "H4". Moreover, from the results, three types of behaviors are observed. The first behavior regroups all "H4" inputs regardless of their dimensionality. Their performances were the lowest and highly similar to the baseline. It is important to remember that the C in "H4" increased both $r_{patterns}$ and r_{inputs}. The second behavior regroups "H1" and ratio of 4/1 and 2/1. Their performances were also similar to baseline. Finally, the last behavior is seen for condition H1 and ratios of 1/2, 1/4 and 1/8. All three significantly outperformed the baseline. As we increased the dimensionality of the C, the performance improved. Thus, regardless of the ratio between patterns and C, a highly correlated pattern juxtaposed with a C that lowers both $r_{patterns}$ and r_{inputs} will be more noise tolerant. Also, if the importance/size of the orthogonal C increases, this will result in even greater

noise tolerance. In counterpart, if the C increased both $r_{patterns}$ and r_{inputs}, noise tolerance will be degraded.

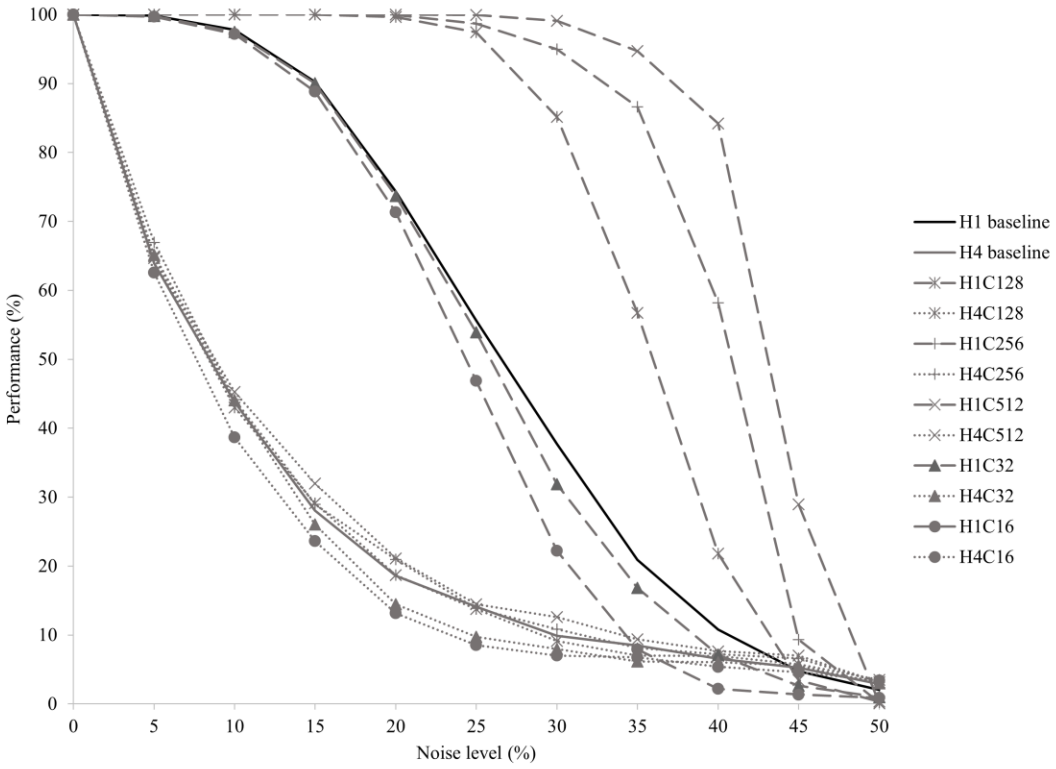

Figure 16. Results of condition H1, represented in dashed lines, and H4, represented in dotted lines, for ratio of pattern and C of 4/1, 2/1, 1/1, 1/2, 1/4 and 1/8. H1 baseline is represented as the black line and H4 baseline is represented as the gray line.

V. Discussion

The results showed an interesting effect of an arbitrary context in regard to learning times of similar and dissimilar pairs of patterns. The results have shown that the learning time can be reduced and noise tolerance increased if orthogonal Cs are added to the patterns. The results also showed that by adding a component to help differentiate the patterns, learning times can be reduced by 94%. Results for recall performances were also greatly improved in the case of highly correlated patterns. When comparing highly correlated patterns to the baseline "H", the addition of a

condition 1 C of equal proportion to the patterns increases the noise tolerance by 50% for low and medium noise and by 10% for high noise. Subsequently, by increasing the relative proportion of the Cs compared to the initial stimuli, learning times and recall performances can be further improved. With a ratio where Cs are eight times bigger in dimension than the stimuli, an additional reduction of 50% of the learning time can be achieved. Also, the noise tolerance increases by an additional 30% for medium noise and 80% for high noise compared to the previous results. It is possible to outperform lowly correlated patterns even if the learning set of patterns is highly correlated. Augmenting the ratio in favor of the orthogonal C in condition 1 allowed to lower both $r_{patterns}$ and r_{inputs} significantly, resulting in better recall performances. This opens the possibility of manipulating the distance between patterns by increasing the size of an orthogonal C prior to learning to aid the network in distinguishing correlated patterns. This can be analogous to the level of resources allocated to a given aspect of a stimulus. When trying to discriminate similar stimuli, if attention is oriented on the distinguishing features of the stimuli rather than the similar features, the capacity of distinguishing them is facilitated, thus making it easier to learn and categorize [Hammer et al., 2008]. In short, the most relevant information comes from the distinguishing trait that can separate the patterns. Therefore, prior knowledge towards finding the distinguishing feature in a stimulus could allow the modification of the correlation of that stimuli and facilitate learning.

Interestingly, as shown by the conditions where both $r_{patterns}$ and r_{inputs} are increased, dimensionality had no effect on performances. In addition, when increasing dimensionality, memory load is decreased, which further improves performance. However, bigger dimensions in both layers increase the number of nodes as well, thus increasing the real-time length of learning. In terms of correlation, when looking at the correlation between the newly formed inputs,

performances are optimal for lowly correlated inputs. As the correlation increases, performance diminishes. It would be interesting to investigate the relationship between correlation and performances more in-depth in order to extract an optimal value.

The method presented requires prebuilt orthogonal C. This arbitrary representation was necessary in order to establish the optimal conditions. However, in a more natural setting, the network should be able to generate the context by itself. Therefore, the network would be able to modify the presented stimuli in order to facilitate its own learning in various situations. This could lead to more general-purpose applications, where the membership of a given pattern could differ in function of the task. It would also reflect how contexts can be seen as an independent mechanism in the brain and how it can be influenced both externally and internally. That being said, if the goal is shifted from discriminability towards grouping for a classification problem, then it would be interesting to investigate the advantage of modifying the stimuli to increase the $r_{patterns}$. As previously mentioned, highly correlated patterns are closer in space compared to their counterparts. Therefore, in the presence of unsupervised feature extraction tasks, a higher correlation between the patterns that belong to the same category would be beneficial. This would allow the network to regroup completely different inputs together to form a new category. Investigating such approach is crucial towards the development of general-purpose intelligence while also helping better understand the mechanism behind distinguishing or grouping stimuli. Finally, the method presented was based on the assumption that the initial patterns must be kept intact throughout the simulations. If this constraint is not required, then decorrelation methods could be used such as feature extraction, principal component analysis or pseudo-orthogonalization [Chartier et al., 2009; Minemoto et al., 2017; Chartier et al., 2007].

The goal of this paper was to introduce a new mechanism capable of better understanding how context affects learning and memory when in presence of correlated stimuli while keeping the original patterns intact. Results from incorporating a context showed that learning times can be lowered, and noise tolerance can be increased without any modification to the model. This is a promising path for ANNs to deal with highly correlated stimuli.

Learning and Recalling Arbitrary Lists of Overlapping Exemplars in a Recurrent Artificial Neural Network

Publication reference:

Rolon-Mérette, D., Rolon-Mérette, T., & Chartier, S. (2019). Learning and Recalling Arbitrary Lists of Overlapping Exemplars in a Recurrent Artificial Neural Network. In *Proceedings of the International Conference on Cognitive Modelling* (pp. 186-191).

Abstract

The mechanisms behind the ability to retrieve all exemplars in a class when presented a specific contextual cue, have puzzled the world of cognition. Various approaches have been used to better understand this concept, especially in the field of artificial neural networks. That being said, very few models can enumerate all exemplars associated to multiple lists in a cognitively plausible way. This is mostly due to the problem of multiple One-to-Many Associations (OMA)s where various exemplars can belong to different lists. To resolve this issue, different approaches have been used; from deep learning and natural language processing to time delayed contextual units. However, none of them is satisfying for a biologically based computational model of cognition. A promising solution is using the class label as context and associates it with each exemplar of the corresponding class. This allows each input to be unique and the problem becomes a standard association one. This strategy has been implemented within the neurodynamic perspective using a bidirectional associative memory. The simulations consisted of learning three arbitrary sequences of various lengths containing multiple intertwined exemplars. Results showed that it was possible to enumerate all associated exemplars from a class simply by presenting the corresponding contextual label. These findings are an important step towards developing cognitively plausible neural implementation of multi-step patterns as well as semantic networks in order to develop generalized artificial intelligence.

Keywords— Cognition; Class enumeration; One-to-many associations; Learning; Memory; Bidirectional associative memory; Feature extraction bidirectional associative memory

Introduction

When presented with a class label, the brain has no difficulty in enumerating all associated exemplar. This cognitive ability is remarkable since learned exemplars are rarely exclusive to a single class; they can have multiple associations (also referred as One-to-Many Associations; OMAs). A simple example to illustrate this would be to enumerate all actions (exemplars) needed to score in a specific sport. In soccer a sequence may resemble, kick off, pass, dribble, run, and kick; while in hockey: face off, pass, dribble, skate and slap; and finally, in basketball: jump ball, pass, dribble, run and throw. Depending on the class of the sport (soccer, hockey or basketball), different exemplars (kick, slap and throw) and identical ones can be found (pass, dribble). Thus, in such a case, when enumerating exemplars from a class, it is easily seen that these exemplars can be associated to a single or multiple classes (OMA). Furthermore, when enumerating a list, it may always follow a specific sequence (e.g., opening a door) or, in the case of semantic memory, may not (e.g., free association task [Nelson et al., 2004]).

Many formal models in cognitive sciences have been proposed over the years that can accomplish this listing task. Models, such as the Semantic memory models, are known to predict human performances accurately [Jones et al., 2015] but remain limited for neural implementation. ANNs have also been used to perform this listing task with most using the multi-layer Perceptron approach [Collobert et al., 2011; Elman, 1990; Jordan, 1997; Neville, 2008]. Although these are all interesting models, they are limited in meeting the requirements to be considered biologically based computational models of cognition [O'Reilly, 1998]. One such class of models that fills these requirements are the Recurrent Associative Memories (**RAMs**) that belong to the neurodynamic approach [Haykin, 2009]. AM consists of learning and storing pairs of identical (auto-association) or different (hetero-association) exemplars. This has been popularized by

Hopfield (1982) for auto-association and generalized to Bidirectional Associative Memory (BAM) by Kosko (1988) for hetero-association. Since then, BAMs have seen many modifications allowing them to perform various tasks with better performances; see Acevedo-Mosqueda and colleagues (2013) for a review. Previous studies have shown that RAMs are able to enumerate simple independent lists of exemplars [Chartier & Boukadoum, 2006]. However, in the presence of overlapping list of exemplars, like the initial example, they are not able to accomplish the task. This is due to the fact that they must deal with several OMAs. In other words, they are dealing with a relationship instead of a function.

An early solution to solve OMA following findings in cognition [Clarke, 2017; Spillers & Unsworth, 2011; Stoet & Snyder, 2007] was the use of time delayed (context) units [Elman, 1990]. This method integrated previous output(s) with the current input in order to accurately predict the next exemplar in the list [Collobert et al. 2011]. Therefore, the OMA was transformed into a OOA. Unfortunately, this solution of delay units (or surrounding context) requires the global knowledge of the number of contextual units prior to learning. Furthermore, it does not really help towards the original task itself; time delayed units (context) are not representative of the class but only of the previous exemplars. A more interesting solution in ML was introduced by Jordan (1997), which used context as a label to modify each exemplar for the enumeration of a given class. Therefore, this contextual label also modifies each exemplar to make them unique without any prior global knowledge.

A second problem may also arise in RAMs if the OMAs are overlapping. In this case there is the possibility that the task becomes a nonlinearly separable one. Unfortunately, standard BAMs are not able to solve this unless the model is complex with a wide range of arbitrary parameters, thus losing its simplistic nature. However, recent studies have shown that an unsupervised version

of the BAM can be used to increase the dimensionality of the inputs, and therefore, a linear solution can be found when combined with the BAM [Tremblay et al., 2013]. Following recent progress in using contextual labels [Rolon-Mérette et al., 2018a], it is thus proposed that the class label be used to make each exemplar unique using a combination of supervised and unsupervised BAMs. This will increase the BAM's versatility and help in learning any number of overlapping OMAs of any length, where exemplars can have any level of correlation and where a nonlinear solution is required.

The remainder of the paper is divided as follows: The next section gives a brief background of the BAM used in the study. This is then followed by Simulation I, where context is used to show the feasibility of enumerating exemplars from a class and the limits when facing overlapping OMAs; Simulation II is then presented with a brief background of the unsupervised BAM and how its interaction with the BAM allows the network to perform the desired task; Finally, a short discussion ends this paper.

Bidirectional associative memory

Model Description

The model is a modified version of the BAM. Like any neural network, it is defined by an architecture, transmission and learning functions.

Architecture. The BAM's architecture is illustrated in Figure 17. The supervised model has two layers of interconnected units in a bidirectional fashion, where the **W** and **V** layers return information to each other (both acting as teachers to one another), where M and N represent the number of units in each layer. The initial patterns are represented by $\mathbf{x}_{[0]}$ and $\mathbf{y}_{[0],}$ while the outputs of the network are $\mathbf{x}_{[c]}$ and $\mathbf{y}_{[c]}$ after c cycles.

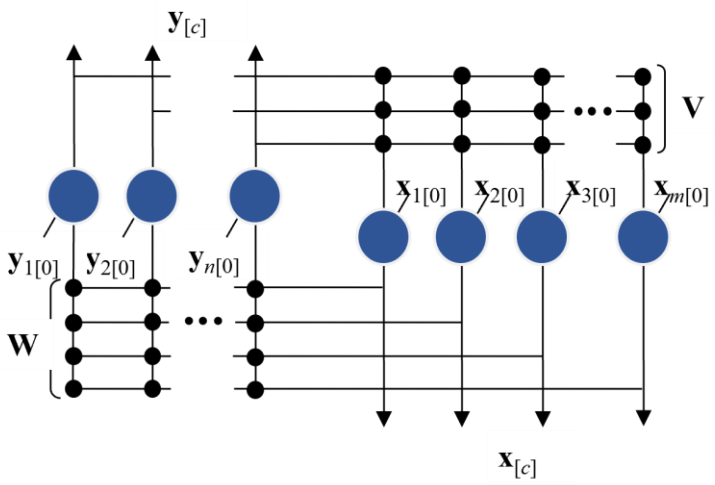

Figure 17. Architecture of the BAM

Output function. The transmission function is defined by equation 1a and 1b:

$$\text{Equation 1a} \quad \forall i, \dots, n, \mathbf{y}_{i[c+1]} = f\left(\mathbf{a}_{i[c]}\right) = \begin{cases} 1, if\ \mathbf{a}_{i[c]} > 1 \\ -1, if\ \mathbf{a}_{i[c]} < -1 \\ (\delta + 1)\mathbf{a}_{i[c]} - \delta \mathbf{a}_{i[c]}^3, Else \end{cases}$$

$$\text{Equation 1b} \quad \forall i, \dots, m, \mathbf{x}_{i[c+1]} = f\left(\mathbf{b}_{i[c]}\right) = \begin{cases} 1, if\ \mathbf{b}_{i[c]} > 1 \\ -1, if\ \mathbf{b}_{i[c]} < -1 \\ (\delta + 1)\mathbf{b}_{i[c]} - \delta \mathbf{b}_{i[c]}^3, Else \end{cases}$$

Where i is the index unit, δ the general transmission parameter, and a and b are the activations. These activations are obtained the usual way and are defined by equation 2a and 2b:

$$\text{Equation 2a} \quad \mathbf{a}_{[c]} = \mathbf{W}\mathbf{x}_{[c]}$$

$$\text{Equation 2b} \quad \mathbf{b}_{[c]} = \mathbf{V}\mathbf{y}_{[c]}$$

Learning rule. The connection weights for the model are modified following a hebbian/anti-hebbian rule [Chartier & Boukadoum, 2006].

$$\text{Equation 3a} \quad \mathbf{W}_{[k+1]} = \mathbf{W}_{[k]} + \eta\left(\mathbf{y}_{[0]} - \mathbf{y}_{[c]}\right)\left(\mathbf{x}_{[0]} + \mathbf{x}_{[c]}\right)^{\mathrm{T}}$$

$$\text{Equation 3b} \quad \mathbf{V}_{[k+1]} = \mathbf{V}_{[k]} + \eta\left(\mathbf{x}_{[0]} - \mathbf{x}_{[c]}\right)\left(\mathbf{y}_{[0]} + \mathbf{y}_{[c]}\right)^{\mathrm{T}}$$

Where $\mathbf{x}(0)$ and $\mathbf{y}(0)$ are the initial inputs, η is the learning parameter and k is a given learning trial. Equations 3a and 3b show that the matrix weights will converge when $\mathbf{x}_{[0]} = \mathbf{x}_{[c]}$ or $\mathbf{y}_{[0]} = \mathbf{y}_{[c]}$. To reduce the simulation time, the number of cycles performed according to equation 1 is usually set to $c = 1$. It is guaranteed that the learning will converge if the learning parameter (η) is smaller than the following value [Chartier & Boukadoum, 2006] defined by equation 4:

$$\text{Equation 4} \quad \eta < \frac{1}{2(1-2\delta)\text{Max}[M,N]} \, , \delta \neq \frac{1}{2}$$

Simulation I: BAM

The general task is illustrated in Figure 18. In order to recreate the task of enumerating exemplars from a class by only presenting its class label, three overlapping lists of arbitrary patterns were used. The general goal was to learn all overlapping lists. Two simulations (conditions) were created to better understand the complexity of the task and the feasibility of using a single BAM. The first condition was to establish if labels can be used to solve a simple OMA and enumerate all the exemplars of a class, while the second was to show its limitation with overlapping OMAs. Both conditions are illustrated in Figure 19.

Methodology

Arbitrary alphabetic patterns were used to test the network. Each pattern was a 49-dimensional pixel base pattern where black pixels represent the value of +1 and white pixels -1. Those patterns have the property of showing various levels of correlation (between 0.02 and 0.92). Moreover, they can be naturally partitioned into multiple overlapping classes and are easily recognized by experimenters. Of course, any other arbitrary patterns could have been used without any modification in the results. Two conditions were created.

In condition 1, two sequences of three exemplars (class label followed by two letters) were

used to generate an OMA scenario. For both sequences, the class label was concatenated to each exemplar of the list, allowing exemplar modification and transforming the OMA into a OOA (Figure 19a). In condition 2, multiple intertwined lists were used (Figure 19b). The number of lists was set to three for proof of concept while avoiding the simplicity of having a binomial solution. Furthermore, contrary to condition 1, each list was of different lengths and contained multiple OMAs. The first list contained the class label "L" followed by the 26 letters of the alphabet in lowercase. The second sequence was a subset of the first, containing the class label "V" followed by all the vowels in lowercase. Finally, the third sequence was a different subset of the first list containing the class label "C" followed by all the consonants in lowercase. For both conditions, each list was ended by an auto-association on the last exemplar (final attractor).

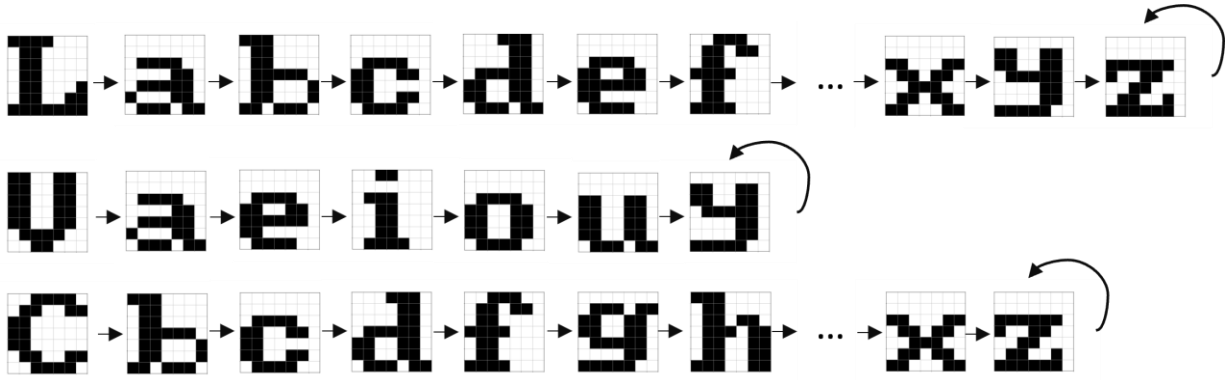

Figure 18. Sequences with class labels ("L", "V" and "C") at the beginning of each class for the overall task.

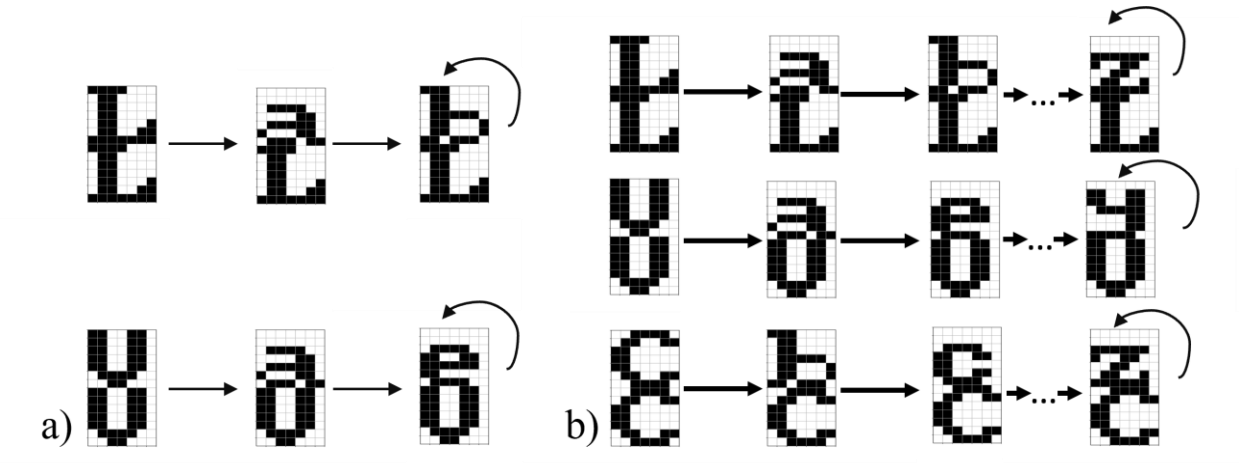

Figure 19. Class labels followed by modified exemplars for condition 1(a) and condition 2 (b).

Procedure

For all simulations, the transmission parameter (δ) was set to 0.2 and the learning parameter (η) respected equation 4. The *M* and *N* layers were set to 98 units each, which represents the dimensionality of the combined exemplar (49) and the class label (49). Learning was stopped when the mean squared error (MSE) was lower than 10^{-15} or when 5000 learning trials was reached.

Learning.

1. Selection of a list containing both the exemplars and the combined context (Figure 19b).
2. Random selection of a pair ($\mathbf{x}_{[0]}$ and $\mathbf{y}_{[0]}$).
3. Computation of $\mathbf{x}_{[1]}$ and $\mathbf{y}_{[1]}$ according to the transmission function (equation 1a, 1b, 2a and 2b).
4. Computation of the weights according to the learning function (equations 3a and 3b)
5. Repeat step 2) and 4) until all the pairs are selected.
6. Repeat step 2) to 5) until the desired MSE or the maximum learning trial is reached.

Recall.

1. Selection of an initial contextual label of a given list, $\mathbf{x}_{[0]}$.
2. Compute $\mathbf{y}_{[c]}$ in accordance to the transmission function (equation 1a,1b, and 2a and 2b) until convergence; end of the list.
3. Comparison of the outputted exemplars with the correct ones.
4. Repetition of steps 1) to 4) for each contextual label ("L," "V," and "C").

Results

Figure 20 shows the output for each of the conditions when presented with the class label. In condition 1, the output is a clear representation of the desired solution for both associated outputs. By using this approach, it was possible to solve a simple OMA. In condition 2, results showed that appropriate retrieval was unobtainable in the situation of many OMAs. This is not surprising because in such a scenario the task becomes nonlinear as well. Therefore, a single BAM will not be able to perform this task. However, by combining the BAM with its unsupervised version, it is possible to overcome this limit. Therefore, in the next simulation, we show such an implementation while keeping the contextual encoding strategy to allow the network to achieve the desired behavior shown in Figure 18.

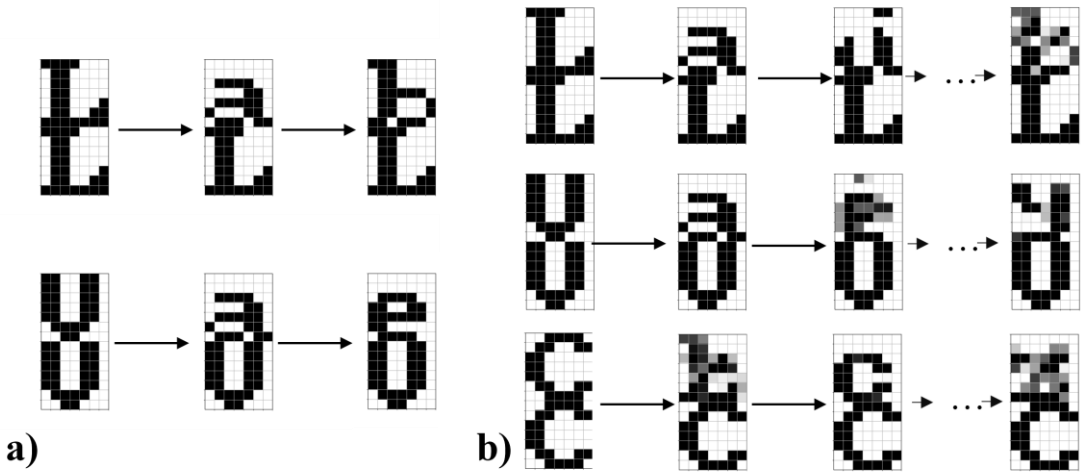

Figure 20. Recall outputs for condition 1(a) and 2 (b).

Simulation II: FEBAM-BAM

In order to use the context to discriminate identical exemplars and to solve non-nonlinear classification, the BAM network is modified to take into account information from its unsupervised version, the Feature Extracting Bidirectional Associative Memory (**FEBAM**). The FEBAM generates a representation that, when combined with the initial input, increases the dimensionality and makes the classification problem into a linear one [Tremblay et al., 2013]. By still having the same learning and transmission functions and the same general bidirectional architecture, this contributes towards increasing the internal consistency of the overall model.

FEBAM model description

The FEBAM is the unsupervised version of the BAM previously described. The only notable difference between the two is the absence of external ($\mathbf{y}_{[0]}$) connections. Consequently, there is no teacher, and the model must rely only on one set of inputs ($\mathbf{x}_{[0]}$). Therefore, the goal of this model is to generate the best representation that allows optimal reconstruction of the inputs. It is a process akin to feature extraction [Chartier et al., 2007].

Architecture. The FEBAM's architecture is illustrated in Figure 21. Like the BAM, this model has two layers of interconnected units in a bidirectional fashion, where the **W** and **V** layers return information to each other. As mentioned, there is only one explicit set of connections, $\mathbf{x}_{[0]}$, used to store information.

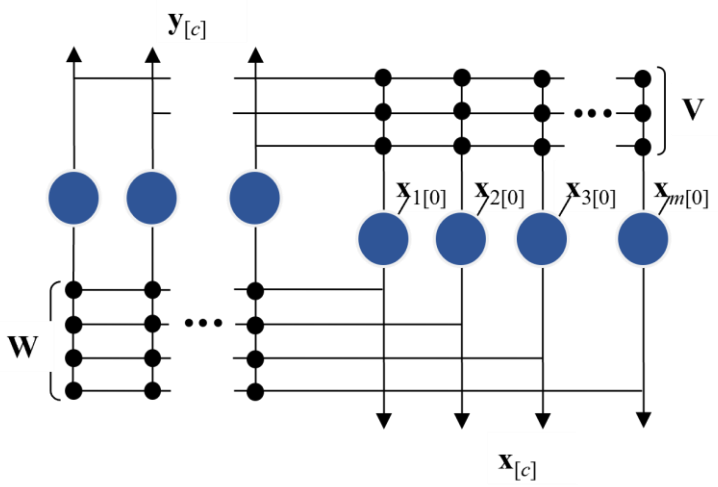

Figure 21. Architecture of the FEBAM.

Transmission and learning functions. Both transmission (equation 1a and 1b) and learning functions (equation 3a and 3b) remained the same. However, since $\mathbf{y}_{[0]}$ is not explicitly given, the information has to circulate a little longer in the network in order to get all needed inputs. As shown in Figure 22, $\mathbf{y}_{[0]}$ is obtained by iterating $\mathbf{x}_{[0]}$ through its corresponding weight connections **W** using the transmission function. Subsequently, $\mathbf{x}_{[1]}$ is obtained from $\mathbf{y}_{[0]}$ and finally, $\mathbf{y}_{[1]}$ from $\mathbf{x}_{[1]}$. Through weight updates, each $\mathbf{x}_{[1]}$ and $\mathbf{y}_{[1]}$ will converge to a solution that will try to best reconstruct its associated initial pattern $\mathbf{x}_{[0]}$ and/or its representation $\mathbf{y}_{[0]}$. Thus, in the case where it is impossible for $\mathbf{x}_{[1]}$ to equal $\mathbf{x}_{[0]}$ (e.g., information compression), weight convergence will only be guaranteed by $\mathbf{y}_{[1]}$ and $\mathbf{y}_{[0]}$. The number of units in the **y**–layer determines the dimensionality (level of compression) of the generated representations. The more units there are, the better the reconstruction will be [Giguère et al., 2007a, 2007b].

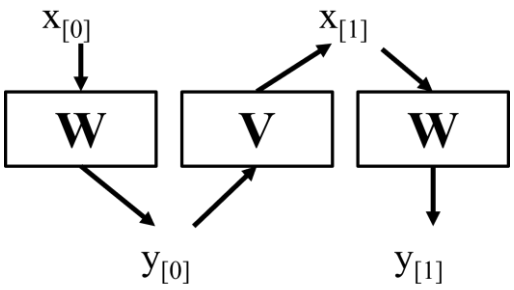

Figure 22. Iterative process used to gather inputs for learning.

FEBAM-BAM Model

Figure 23 illustrates the overall network to accomplish the task where the FEBAM is used to generate features (context).

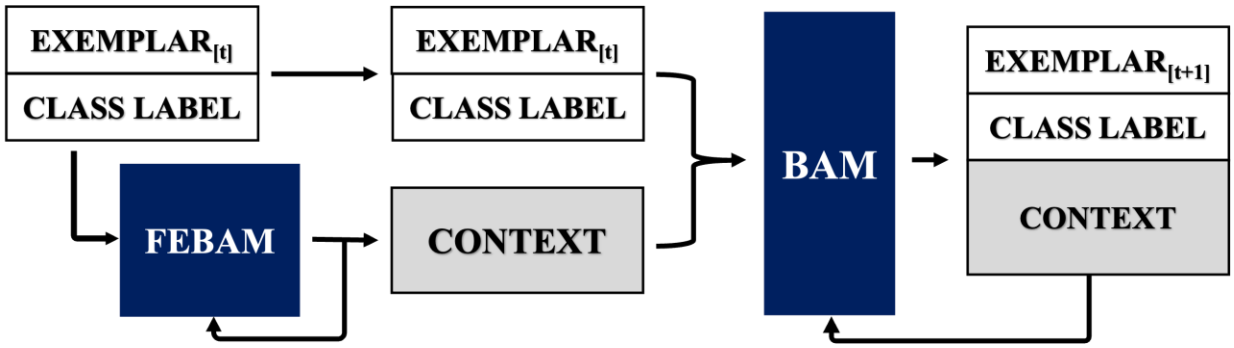

Figure 23. Overall architecture of the FEBAM-BAM.

Methodology

The task consisted of learning the same three sequences from simulation I's condition 2 (Figure 19b). This time, exemplars with their class label were fed to the FEBAM first. This allowed the FEBAM to generate features which acted as a unique "signature" for the current exemplar. This representation was then concatenated to the initial input and fed to the BAM for learning. The number of y units in the FEBAM was fixed at a dimension of 98. This was determined in order to increase the probability of success [Rolon-Mérette et al., 2018b]. The number of y-units can be lower than the number of x-units, but for the scope of this study, it was not investigated. Finally,

in addition to recalling condition 2's lists, a simple noisy (pixel flip) recall task was performed, where the class labels were distorted prior to being presented to the FEBAM-BAM. For the noisy recall task, pixel flip ranged from 0 to 50% of the original class label to show the FEBAM-BAM's ability to deal with noise.

FEBAM Learning. The inputs were presented to the FEBAM. To maintain internal consistency, the transmission parameter (δ) and the learning parameter (η) were not changed from simulation I. The weights were randomly initialized with values between -0.1 and 0.1 and were updated after one cycle ($c = 1$). Learning stopped when the network achieved a mean squared error (MSE) of less than 10^{-15} or when 5000 learning trials were reached. The learning procedure can be described as follows:

1. Selection of a list containing all three sequences of modified exemplars as seen in Figure 19b).
2. Random selection of a given exemplar from the list to obtain $\mathbf{x}_{[0]}$.
3. Iteration through the network (as illustrated in Figure 22) using the output function (equation 1a, 1b, 2a and 2b) to obtain $\mathbf{y}_{[0]}$, $\mathbf{x}_{[1]}$ and $\mathbf{y}_{[1]}$.
4. Computation of weight updates according to the learning rule (equations 3a and 3b).
5. Repetition of steps 2) to 4) until the minimum mean squared error between $\mathbf{y}_{[0]}$ and $\mathbf{y}_{[1]}$ or max trials is reached.

Each output was then concatenated to its associated input before being presented to the BAM for learning. The same learning and recall procedure from simulation I was used for the BAM except for the *M* and *N* layers; they were increased to 196 units due to the concatenation.

Results

All three sequences (Figure 18) were successfully learned by the combined FEBAM-BAM

model (Figure 23). Furthermore, contrary to condition 2 in simulation I, every exemplar for each sequence is retrieved correctly without any distortion. These results are similar to those obtained in ML [Collobert et al., 2011; Jordan, 1997; Neville, 2008].

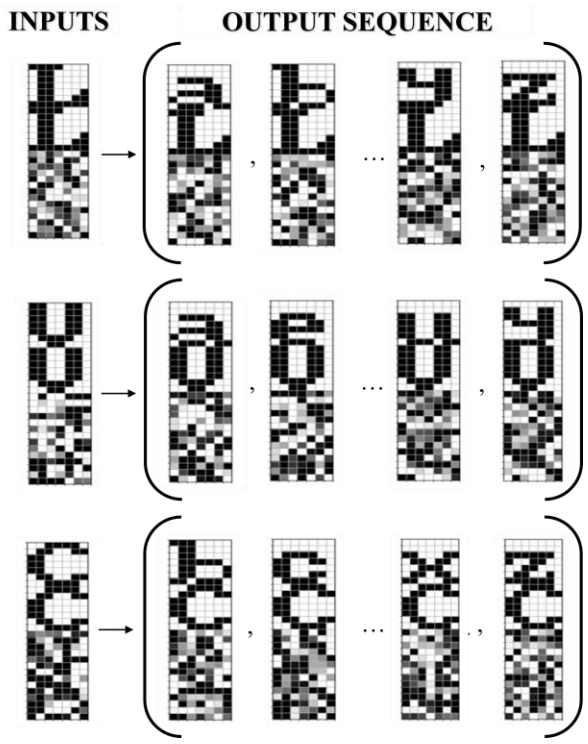

Figure 24. Recall of the learned three sequences.

Likewise, during the pixel flip recall task, correct retrieval was possible for distortion between 0 and 25 %. Figure 25 shows results for a pixel flip of 10% (10 pixels) of the original class label ‘L’. This “cleaning” by the FEBAM portion of the FEBAM-BAM allowed to obtain the same retrieval results (Figure 24) while dealing with noisy class labels (inputs).

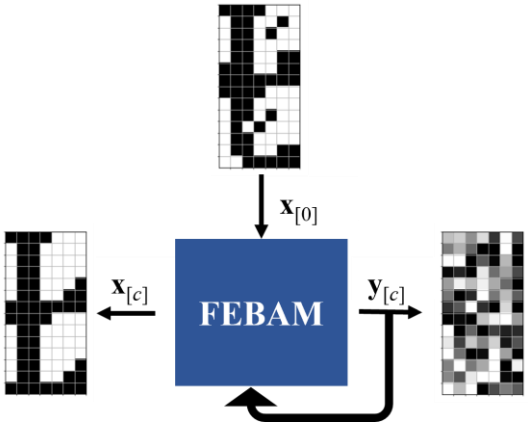

Figure 25. Noisy recall (10 % pixel flip) of class label 'L'.

Discussion

In Simulation I, the goal was to learn a simple OMA task (condition 1) by using the context to discriminate the exemplar in a BAM. Results showed that joining the input with fixed contextual information allows the network to solve a simple OMA task using a cognitively plausible neuronal implementation. However, condition 2 showed that this strategy alone is not sufficient if the task contains multiple OMAs. In this last case, a nonlinear classification is then required.

To remedy this problem, in simulation II, the BAM was employed in combination with the FEBAM. This addition allowed the network to create its own generated features and, when combined with the initial input, allowed it to solve the nonlinear task. The network was able to achieve perfect learning and recall while maintaining internal consistency; the same transmission, learning functions and the same general bidirectional architecture were used. Furthermore, when faced with distorted exemplars, the model was able to "clean" noisy inputs and reconstruct the appropriate class label while retrieving the associated generated feature. This allowed the FEBAM-BAM to solve the enumeration task despite being presented with noisy inputs. This is an important feature towards having a model deal with real world stimuli.

This combination (FEBAM-BAM) is an interesting solution because it avoids the current task

specific problem [Marcus, 2018]. In approaches where context is given through time delay units [Chartier & Boukadoum, 2006; Collobert et al., 2011; Elman, 1990], the network must know beforehand how many of those units will be necessary for the task, limiting its versatility and plausibility.

That being said, in this model, the proposed mechanism is sequence specific. In other words, although the sequences themselves were arbitrary and could be replaced by any sequences of exemplars, the network outputs will always be in the same order. This is accurate in the case of learning multi-step patterns like motor outputs. However, a future desired property would be the inclusion of more flexibility where the order of outputs is determined from the frequency of occurrence or the success rate of past experience using reinforcement learning. Furthermore, it would be interesting to follow up on the inherent characteristic task [Hattori & Hagiwara, 1998] while using the FEBAM-BAM's ability to modify the exemplars with pseudo-contextual compartments [Clarke, 2017; Spillers & Unsworth, 2011; Stoet & Snyder, 2007]. This would open the door towards a cognitively plausible artificial neural-network capable of combining knowledge acquisition and knowledge transfer, increasing even further the model's versatility. Additionally, it is known that the number of y-units must be greater or equal to the number of unique exemplars for feature extraction in the FEBAM [Tremblay et al., 2013]. That being said, it would be advantageous to investigate the probability of success for this multi-OMA task while controlling for the dimensionality of the generated context (FEBAM y-unit). This could determine if the number of exemplars in a list or the number of intertwined exemplars have an impact on the number of y-units needed for the nonlinearly separable OMA task. Finally, it would be interesting to account for exemplars in a list representing a single exemplar or a whole category in itself. This would allow the model to perform an important semantic memory task while being a simple

neuronal model (free association task [Nelson et al., 2004]).

In sum, it was shown that a simple recurrent BAM with a Hebbian/anti-Hebbian learning algorithm is sufficient to solve a complex task requiring the enumeration of all associated arbitrary exemplars from a class by the sole presentation of a class label. These findings are an important step toward developing a neural implementation of semantic networks in order to shift from narrow intelligence to artificial general intelligence (AGI) [Bengio et al., 2015; Marcus, 2018].

Chapter 1 Conclusion

The ability to deal with OMA in the BAM framework was investigated in this chapter. By following Jordan's approach and juxtaposing contextual representations to sets of patterns, it was possible to change the correlation of these items. These changes facilitated the overall performance of the model. Additionally, using these representations allowed the model to learn a OMA. In other words, simple associative mechanisms can account for this behavior. However, an additional BAM-like model, the FEBAM, was needed to ensure success when sequences contained multiple overlapping OMAs. This requirement hinted towards the relationship between patterns being nonlinear and deserved further investigation to ensure reliability, consistency, and simplicity in the approach.

Chapter 2

Chapter 2 introduces a novel multi-network BAM to deal with nonlinear tasks. In the paper titled *A Multilayered Bidirectional Associative Memory Model for Learning Nonlinear Tasks*, the consequence of using multiple FEBAM architectures in conjunction with the BAM was evaluated. Various nonlinear tasks were used to ensure the model could perform consistently and in different settings (discreet and continuous-valued tasks, segregated and overlapped decision zones, online continuous learning, and multiclass setting). The proficiency of the new model was evaluated through decision zones, learning times and correct recall.

A Multilayered Bidirectional Associative Memory Model for Learning Nonlinear Tasks

Publication reference:

Rolon-Mérette, D., Rolon-Mérette, T., & Chartier, S. (2023). A multilayered bidirectional associative memory model for learning nonlinear tasks. *Neural Networks*, *167*, 244-265.

Abstract

A multilayered bidirectional associative memory neural network is proposed to account for learning nonlinear types of association. The model (denoted as the MF-BAM) is composed of two modules, the Multi-Feature extracting bidirectional associative memory (MF), which contains various unsupervised network layers, and a modified Bidirectional Associative Memory (BAM), which consists of a single supervised network layer. The MF generates successive feature patterns from the original inputs. These patterns change the relationship between the inputs and targets in a way that the BAM can learn. The model was tested on different nonlinear tasks, such as the *N*-bit, Double Moon and its variants, and the 3-class spiral task. Behaviors were reported through learning errors, decision zones, and recall performances. Results showed that it was possible to learn all tasks consistently. By manipulating the number of units per layer and the number of unsupervised network layers in the MF, it was possible to change the level of nonlinearity observed in the decision boundaries. Furthermore, results indicated that different behaviors were achieved from the same set of inputs by using the different generated patterns. These findings are significant as they showed how a BAM-inspired model could solve nonlinear tasks in a more cognitively plausible fashion.

Keywords—ANNs, Associative Memory, BAM Neural Networks, Cognition, Multilayer, Nonlinear Tasks

1. Introduction

According to the associative memory theory, the human brain is an association machine [Anderson & Bower, 2014; Buckner & DiNicola, 2019]. Every day, it is learning, storing, and retrieving associations between objects, concepts, and other mental items that can be vastly different [Caudill & Butler, 1990]. This process, referred to as Associative Memory (AM), is said to be imprinted in various cognitive behaviors and has been studied extensively in a wide array of fields [Kumar et al., 2015; Lansner, 2009; Letzkus et al., 2015; Leuner et al., 2003; Mayeset al., 2007; Puig et al., 2014; Stalnaker et al., 2010; Suzuki, 2008]. Of particular interest is the computational modeling domain, which has grown in popularity over the last decades. Specifically, Artificial Neural Networks (ANNs) have been widely used to draw parallels between neurophysiology and cognition in many different settings [Elman et al., 1996; Smolensky, 1988; Yang & Wang, 2020]. One of their many benefits is that they permit one to infer how neural information processing can lead to certain cognitive properties, like AM [Alamia et al., 2020; Barak, 2017; Hintzman, 1991; McClelland, 2000a, 2000b; O'Reilly & Munakata, 2000; Thomas & McClelland, 2008].

1. 1. Background and Problematic

ANNs built for studying AM are often referred to as Neural Associative Memory (**NAM**) [Palm, 2013; Zhang et al., 2021]. Like ANNs, most NAM also consist of neuronlike and synapse-like elements based on the McCulloch-Pitts neuron [McCulloch & Pitts, 1943]. NAMs play an important role in cognition. They provide an interesting explanation for how learned patterns may correspond to attractors in the brain's neuronal state space. As such, they permit linking the neurodynamic perspective to AM and consequently find themselves in various brain theories [Knoblauch, 2005]. Most NAMs use content addressable memory, a type of computer memory

that searches for pattern similarity among its stored content through a process akin to AM. This allows them to retrieve a corresponding address pattern despite being given a noisy one, which is a feature incorporated in most memory models in cognition [Kohonen et al., 1977]. NAMs can usually perform either auto-associations, hetero-associations, or both, which makes them interesting when trying to explain various cognitive behaviors through AM (see Anderson and Bower's [2014] HAM model for details on how associations can be used to build cognition). As such, one of their many benefits is that they permit to explain how information signal processing can lead to human cognition.

Remarkably, NAMs are nothing new and can be traced back to the 60s, albeit limited and far from displaying proper AM-like behaviors [Steinbuch, 1961; Willshaw et al., 1969]. These behaviors only began to emerge in the 70s, with networks being able to perform auto- and hetero-associations [Kohonen, 1972, 1974; Kohonen et al., 1977; Palm, 1980]. However, these models lacked the ability to form attractors due to their internal dynamics. This came with the brain-state-in-box (BSB) [Anderson et al., 1977]. The BSB could perform auto-association by storing patterns in attractors that were produced by nonlinear recurrent feedback within its network. Consequently, the BSB inspired a new subclass of NAMs, referred to as recurrent neural associative memory (RNAM), mainly interested in the neurodynamic perspective and cognition [Anderson, 1983].

RNAMs became popularized a few years later due to the Hopfield network [Hopfield, 1982]. Although this model showed how correlated stimuli could be learned through a simple one-shot learning rule and proved this stabilization, the model could only display auto-association. A few years later, this limitation was overcome with the Bidirectional Associative Memory (BAM), which could perform both auto-association and hetero-association while retaining its neurodynamic perspective [Kosko, 1988].

To this day, the BAM remains one of the most popular RNAM for studying AM through the neurodynamic perspective. This is because it can perform auto- and hetero-association in a bidirectional fashion while using a simple HL rule, essential aspects for generalizing an ANN to human cognition [O'Reilly, 1998]. Over the years, numerous studies have enhanced the original BAM by allowing it to perform various complex tasks and be used for real-world applications, such as data compression, image recognition, and classification (see [Acevedo-Mosqueda et al., 2013] for a more comprehensive review on BAMs). As with ANNs in general, these advancements mainly came from two perspectives: those interested in performance exclusively, which can be referred to as the Machine Learning (ML) approach, and those searching to build more plausible models of mechanisms found in the animal brain, occasionally denoted as the cognitive or biological approach. Although the ML perspective allowed the BAM to be used in the aforementioned applications, it did so at the cost of losing some of its core properties related to cognitive plausibility: biological realism, distributed representations, error-driven learning, Hebbian learning, and bidirectional activation propagation [O'Reilly, 1998].

On the flip side, cognitively plausible models that try to follow these principles are often seen as either too complex [Labib, 1999], requiring highly nonlinear transmission functions [Shen et al., 2004], using subjective learning procedures [O'Reilly, 1996] or in the BAMs' case, limited when it comes to learning nonlinear associations [Chartier et al., 2009]. This is quite cumbersome for the generalizability of the BAM, especially when considering that humans can associate patterns regardless of the nature of the association being linear or nonlinear [Levering et al., 2020; Rosedahl & Ashby, 2022]. For example, when collecting wild mushrooms, each mushroom encountered will have characteristics that allow an individual to determine whether it is edible (e.g., the color, smell, and size) [Baroni, 2017]. In some instances, a single feature (foul smell)

will be enough to determine that a mushroom is inedible, regardless of all other features. As shown in Figure 26(a), this represents a linear association between the mushroom's feature and its edibility category. However, in most cases, many features must be considered simultaneously. Dealing with such situations often necessitates the ability to form nonlinear associations between features and their category, as shown in Figure 26(b). Although humans constantly do this, it remains an important challenge for cognitively plausible BAMs. Indeed, when these models try to learn a nonlinear task, such as the XOR problem, they remain incapable of finding the appropriate solution. A standard BAM's behavior, illustrated in Figure 26(c), will be linear even when encountering a nonlinear task, meaning in the mushrooms example, the model will miss identify them. Suppose the BAM is to remain a model of human cognition through the neurodynamic perspective. It must be able to learn linear and nonlinear association types.

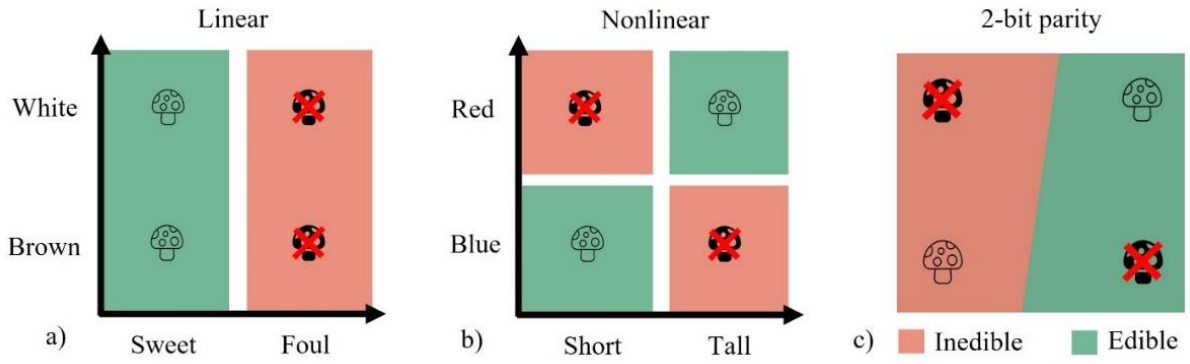

Figure 26. (a) Example of linear associations. (b) Example of nonlinear associations. (c) A standard BAM's classification behavior on the XOR task.

1. 2. Related Work

Different efforts have gone towards teaching the BAM to learn nonlinear tasks. These usually consist of either modifying the BAM itself (e.g., changing its learning rules, architecture, transmission function, etc.) or pre-processing the inputs before being presented to the BAM (i.e., changing their representations). The most popular methods for dealing with nonlinear tasks consist

of using multilayered architectures while adjusting learning through backpropagation and gradient descent [Adigun & Kosko, 2019; Kosko, 2021; Liu et al., 2019; Liwanag & Becker, 1997; Rumelhart et al., 1986; Singh & Saraswat, 2017; Wu et al., 2018;]. The popularity of these methods can be attributed to the much-documented accomplishment of "deep" networks in numerous successful applications [Ferrucci et al., 2010; Silver et al., 2016; Vinyals et al., 2019]. Deep networks are built upon the abstraction of the brain's multilayered architecture, which, in turn, could be an underlying reason for their immense power. However, these models are not necessarily concerned with following cognitive constraints and instead aim to solve tasks optimally [Marcus, 2018]. The main problem with this approach is that it often requires different learning or transmission functions within the combined model while also having to freeze some of its modules during learning [Liu et al., 2019; Wu et al., 2018]. On the other hand, when the BAM is expanded upon to include multiple layers within its architecture and uses a form of bidirectional backpropagation learning algorithm [Adigun & Kosko, 2019; Kosko, 2021; Singh & Saraswat, 2017], it assumes that information can circulate through the same connection bidirectionally. Moreover, its weight update usually requires an unrealistic flow of information, having to circulate the incoming signals many times between inputs and outputs layers before learning can occur.

Regardless of which approach above is used, they go against the current understanding of the human brain. For instance, at both the micro and macro level, networks of neurons seem to behave similarly throughout the brain [Buckner & DiNicola, 2019], meaning that the same transmission function should be used for all nodes, and internal consistency must be sought out. Moreover, there appears to be no evidence of brain regions stopping their activity from accommodating learning [Raichle, 2010]. Thus, no weight-freezing schemes must be used. Additionally, information at the synapse level seems to only circulate in one direction, so no

transposition of weight matrices for backward passes can be utilized [Südhof, 2021]. Lastly, there still needs more evidence that the strength of synapses changes based on the error being backpropagated through them [Song et al., 2020]. Thus, although these "deep networks" approaches permit the BAM to learn nonlinear associations, they often do so in ways that undermine its plausibility and weaken its legitimacy to be used as a tool to model human cognition.

A different approach can be found in the morphological bidirectional associative memory [MBAM; Ritter et al., 1999], gray-scale morphological associative memories [Sussner & Valle, 2006], the alpha-beta bidirectional associative memory [α-β BAM; Acevedo-Mosqueda et al., 2006], chaotic complex-valued bidirectional associative memory [CCBAM; Yano & Osana, 2009], Lie Algebra-Valued BAM [LieBAM; Popa, 2017], and other variants (see [Acevedo-Mosqueda et al., 2013 for a chronological history of other relevant BAMs). These models have been built to deal with nonlinearity and have interesting properties, from the ability to increase storage capacity [Lee et al., 2015], display chaotic behaviors, to using drastically fewer resources (artificial nodes) in the process [Sussner & Campiotti, 2020]. However, these models do so by changing the nature of the artificial node and using different algebra types for its computations. As such, they move away from the traditional view of the McCulloch-Pitts' neuron and the simple neurodynamic perspective captured by the original BAM.

An important question to be answered is whether changing the connectivity is sufficient to learn nonlinear tasks while keeping simple HL and the McCulloch-Pitts' neuron model. One approach that searched to investigate this can be found in [Chartier & Boukadoum, 2006]. This modified version removed the original notion in Koskos' BAM [1988], where the bidirectional connections between two units had to be symmetric. Instead, it used two distinct weight matrices to account for the independence of the forward and backward connections. In this modified BAM,

a simple Hebbian and anti-Hebbian learning rule is used. From this model, an unsupervised version was proposed by replacing one set of connections with its generated representation [Chartier et al., 2007; Chartier et al., 2012]. This variation, called the Feature Extraction Bidirectional Associative Memory (FEBAM), has been shown to learn nonlinear associations if combined with another FEBAM in a head-to-toe fashion (it solved the 2-bit, 3-bit, 4-bit, and 5-bit parity problem). This was evidence that learning these types of associations was possible by only changing the flow of information. In other words, success without undermining the model's internal consistency and while maintaining the modified BAM's core properties [Chartier et al., 2012]. However, this method was only applicable if bipolar values represented the task and, as such, did not generalize well to continuous values. Nevertheless, the FEBAM can be considered an interesting model, as it has the property to generate patterns associated with the original inputs in an unsupervised manner but of different dimensionality to them. Thus, the FEBAM can generate a version of the initial patterns that are either compressed, expanded, or the same size [Giguère et al., 2007a, 2007b]. This is similar to what can be observed in other ML models, such as autoencoders [Bank et al., 2020]. In other words, this means that the FEBAM could be used as a type of pre-processing network.

This was explored in a study that combined the FEBAM and the BAM as a multinetwork model [Tremblay et al., 2013]. Here, the FEBAM was used to generate a higher dimensionality representation that was joined with the original input to make the task linear. This FEBAM-BAM implementation proved interesting as it could learn the *N*-bit parity problem and other real-valued (continuous) nonlinear tasks (e.g., the double Moons). However, the downside of this multinetwork model was that its learning was erratic and required an external stopping mechanism to halt its weight update. Additionally, even with this stopping mechanism, the model could only

find the right solution a fraction of the time. As such, although the FEBAM-BAM approach had internal consistency, it lacked reliability and the capacity to self-stabilize.

Still, this combination of FEBAM and BAM showed promise, as it pointed towards an advantage of using networks of the same nature. By allowing the multiple networks to work together and by only modifying the connectivity of one of them, it was possible to display new behaviors. Using such restrictions, the multimodel gets closer to being analogous to information circulating through different brain regions (neural circuitry) and being transformed along the way [Wang & Cui, 2018]. As such, it may be an inherent explanation of how these homogenous systems can display more complex behaviors.

1. 3. Proposed Solution

In a previous study looking at how to decorrelate highly similar stimuli, a model composed of several FEBAMs was used. Results showed that as the number of FEBAMs increased, the correlation between the generated representations decreased [Rolon-Mérette et al., 2018a]. The human brain's laminar architecture inspired this multi-FEBAM model; as such, it was structured by using various versions of itself stacked one on top of each other. Its decorrelation properties functioned by generating an initial expanded representation of the inputs from the first FEBAM. This generated pattern was then given to the second FEBAM to generate a representation from it. This process was repeated until the last generated representation from the final FEBAM was obtained. Each subnetwork was trained sequentially in isolation, meaning the first FEBAM had to finish learning before the second one began. Although this multiple FEBAM was highly successful in decorrelating inputs, there may be an additional property that was not foreseen. More precisely, it may also be progressively transforming nonlinear relationships into linearly separable ones in a reliable fashion. This could be possible due to the FEBAM's ability to generate patterns through

information expansion. Generating representation from projecting inputs onto a higher dimension could result in finding a linearly separable solution to a nonlinear problem, like what can be observed in other ML models, such as the support vector machine (**SVM**) [Chauhan et al., 2019]. By repeating this process over multiple FEBAMs, this generated representation can be "shuffled" until a set of patterns can share a linear association with its corresponding targets. Therefore, using this multimodel before the BAM as a pre-processing network could allow the latter to learn a task regardless of the level of nonlinearity. In other words, it would be possible to maintain the internal consistency displayed by the FEBAM-BAM approach while also showing reliability and self-stabilize.

Because the learning stabilizes from any starting condition, multiple FEBAMs could be linked together, such as each FEBAM would send its generated representations to the following one. By default, this sequential learning will require additional time because each FEBAM layer must receive constant information for learning to stabilize. At first, only the first network layer would receive constant information (the inputs) during the first few learning epochs. As this initial network layer generates the same representations, the following layer can then begin to produce a stable solution. This process would trickle down until the last FEBAM layer stabilizes. Therefore, as a given layer begins to provide a steady solution, learning (weight update) can continue without hindering the model. In other words, by using a head-to-toe architecture and giving more time for each layer to provide a stable solution, it should be possible to exploit the feature-extracting abilities of the multiple FEBAMs while avoiding the need to train each layer in isolation, solving the design flaw.

As such, this research proposes a new multimodel RNAM capable of consistently learning nonlinear tasks in a cognitively plausible fashion. This model is composed of an unsupervised

module, referred to as the Multi-Feature extraction bidirectional associative memory (**MF**), which contains multiple layers of interconnected FEBAMs, and a supervised module, the Bidirectional Associative Memory (which contains a single modified BAM). It is hypothesized that this design would allow the BAM module to receive linearly separable representations generated by the MF. Consequently, this combined architecture, MF-BAM, would enable the model to find a solution through self-governed dynamics and in a consistent fashion. Most importantly, the MF-BAM would be capable of learning nonlinear tasks while keeping its internal structure consistent throughout its composing networks (e.g., same learning rule, transmission function, etc.), making it a more cognitively accurate model. Furthermore, like the layered organization of biological neurons (e.g., hypercolumns and macrocolumns in the cortex; [Wang & Cui, 2018]), only the number of neurons in each layer (number of units in each network) and the number of layers (number of networks) of the unsupervised component should have an impact on the model's behavior. Changing the MF's structural parameters should substantially affect the generated output, as reported in the decorrelation study [Rolon-Mérette et al., 2018a]. Suppose the MF can change nonlinear relationships between input and targets of a task into a linearly separable one. Then, the final BAM layer should easily be able to learn the task. Otherwise, the outputted behavior of the MF-BAM will be linear, and the model will fail.

To evaluate if the MF-BAM can consistently solve nonlinear tasks, various nonlinear problems were used [Goodwin & Johnson-Laird, 2013; Nadler et al., 2008; Wattenmaker, 1995; Wattenmaker et al., 1986]. These were the *N*-bit parity (bipolar value) and the Moons, Eclipse, and Suns tasks (continuous value). Other abilities, such as continuous online learning and multiclass (3-class Spiral task), were also investigated. Each problem was presented as a hetero-association task, where the model had to learn the appropriate input-target pairs. Additionally, the

structural parameters, *number of units per layer*, and *number of unsupervised network layers* were manipulated to understand the effects of their interaction on performance. Lastly, learning error, decision zones, and recall performance measurements were all used to assess these effects.

As such, to adequately present the MF-BAM, this paper is divided into the following sections: Model, General Methodology, Impact of the Number of Units per Layer (*u*), Impact of the Number of Unsupervised Network Layers (*l*), Impact of Manipulating the Number of Units (*u*) and Unsupervised Network Layers *(l*), Continuous Online Learning, Multiclass Nonlinear Task, General Discussion, and Conclusion.

2. Model

We introduce the MF-BAM, a Bidirectional Associative Memory neural network consisting of multiple unsupervised network layers (**UL**s) and a single supervised network layer (**SL**). The ULs are based on the FEBAM architecture [Chartier et al., 2007], while the SL consists of a modified BAM architecture [Chartier & Boukadoum, 2006]. The MF-BAM functions by associating inputs to their corresponding targets in a bidirectional fashion via multiple transformations through its ULs. For a given input pattern, **p,** learning begins in the MF, where an initial unsupervised network layer (UL_1) learns to generate a first hidden representation, $\mathbf{h_1}$. This representation is then passed to the second unsupervised network layer (UL_2), which will learn to generate a second hidden representation, $\mathbf{h_2}$. This process is repeated for *l* number of unsupervised network layers until the deepest hidden representation, $\mathbf{h_l}$, is generated. Once the entirety of the MF has learned to produce a consistent $\mathbf{h_l}$ representation, it is then sent to the BAM component (SL), which will learn to associate $\mathbf{h_l}$ with its corresponding output target, **o**. Figure 27 displays the overall architecture. The architecture, transmission function, and learning rule are presented in the following sections to understand further how the MF-BAM operates.

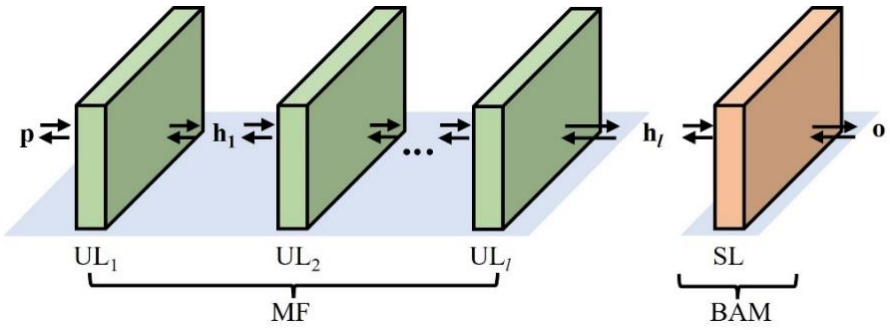

Figure 27. MF-BAM architecture box model.

2. 1. Architecture

The MF-BAM's architecture consists of multiple networks connected in a bidirectional fashion. The model is divided into two components: the Multi-Feature extraction bidirectional associative memory (MF), which contains various ULs, and the Bidirectional Associative Memory (BAM), which encompasses a single SL. Both the ULs and SL have two sublayers (input layer **x** and output layer **y**) of interconnected units in a bidirectional fashion. They are based on a modified version of the BAM [Chartier & Boukadoum, 2006]. Contrary to Kosko's original BAM [1988], the UL and SL are connected via the independent weight matrices **W** and **V**. Due to the recurrent nature of the UL and SL, these sublayers return information to each other. As such, **W**'s shape is given by **y**'s and **x**'s dimensionality and **V** by **x**'s and **y**'s (i.e., if **x** has a dimensionality of *m* and **y** of *n*, **W** becomes a *n* x *m* matrix and **V** a *m* x *n*). As shown in Figure 28, the only difference between the UL and SL is their number of external input connections. The SL has two sets of external connections and receives two inputs, $\mathbf{x}_{[0]}$ and $\mathbf{y}_{[0]}$. On the other hand, the UL only has one set and receives one input, $\mathbf{x}_{[0]}$, and generates its outputs, $\mathbf{y}_{[c]}$. This means that the number of y-units (i.e., the dimensionality of the generated hidden representations) can be manipulated for the UL exclusively.

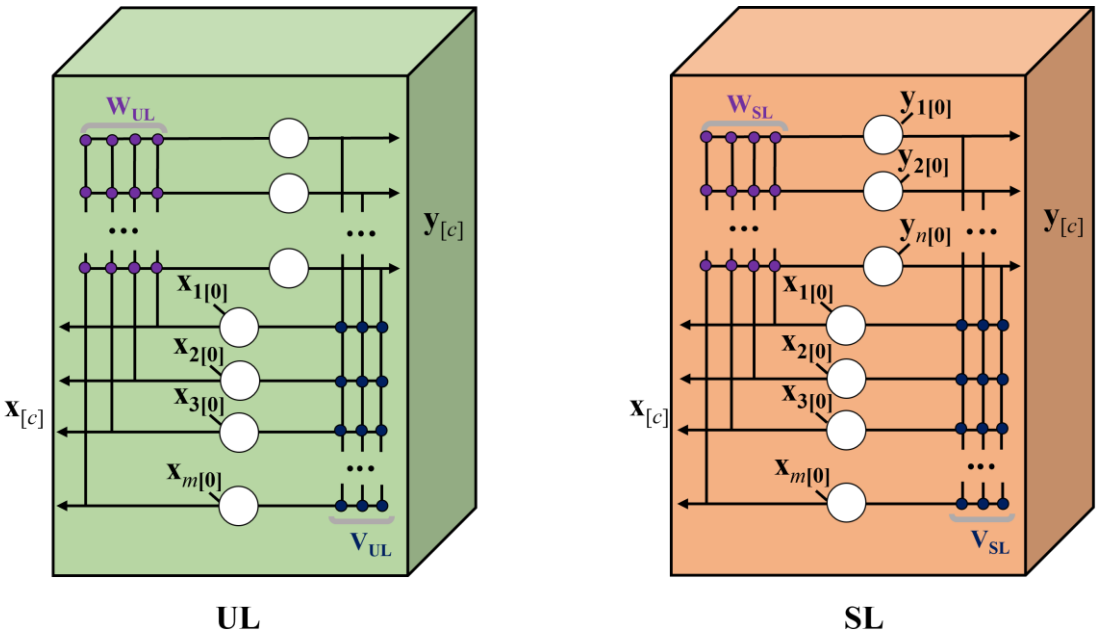

Figure 28. The unsupervised (UL) and supervised (SL) network's architectures.

Since the MF-BAM is composed of two networks of the same nature, it uses the same transmission function and learning rules for all its network layers. Thus, any change in the behavior of the MF-BAM comes only with modifications to its architecture through two parameters: the number of y-units in each UL (i.e., number of units per layer, denoted by u) and the number of ULs used (i.e., number of unsupervised network layers, represented by l). Figure 29 shows these two parameters and how they shape the overall architecture of the MF.

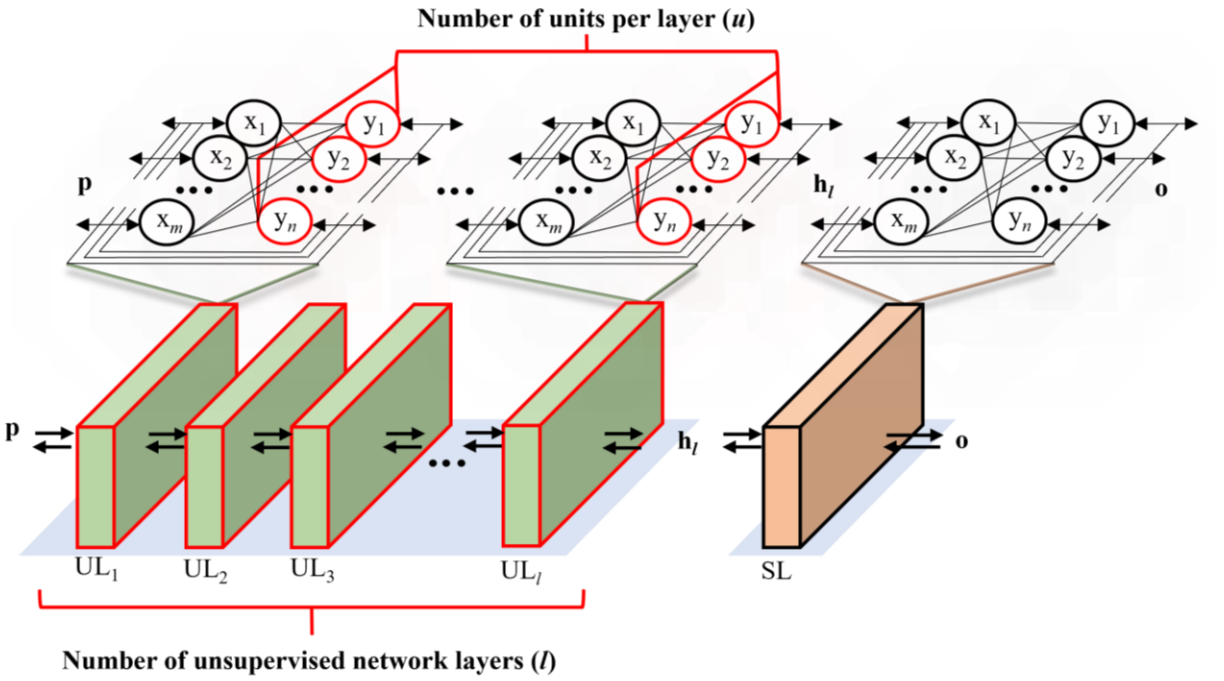

Figure 29. The *u* (number of units per layer) and *l* (number of unsupervised network layers) parameters of the MF component in the MF-BAM.

2. 2. Transmission Function

The MF-BAM uses a cubic transmission function (see [Chartier & Boukadoum, 2006; Chartier et al., 2009] for more details) with saturating limits at ±1, defined by equations 1a and 1b:

$$\text{Equation 1a} \quad \forall i, \ldots, n, \mathbf{y}_{i[c+1]} = f(\mathbf{a}_{i[c]}) = \begin{cases} 1, if\ \mathbf{a}_{i[c]} > 1 \\ -1, if\ \mathbf{a}_{i[c]} < -1 \\ (\delta + 1)\mathbf{a}_{i[c]} - \delta \mathbf{a}_{i[c]}^3, Else \end{cases}$$

$$\text{Equation 1b} \quad \forall i, \ldots, m, \mathbf{x}_{i[c+1]} = f(\mathbf{b}_{i[c]}) = \begin{cases} 1, if\ \mathbf{b}_{i[c]} > 1 \\ -1, if\ \mathbf{b}_{i[c]} < -1 \\ (\delta + 1)\mathbf{b}_{i[c]} - \delta \mathbf{b}_{i[c]}^3, Else \end{cases}$$

where *i* is the index unit, *c* the iteration cycles index, δ a general transmission parameter, *m* and *n* the number of units in each sublayer **x** and **y**, respectively, and **a** and **b** their activations. The activations are obtained following equations (2a) and (2b):

$$\text{Equation } 2a \quad \mathbf{a}_{[c]} = \mathbf{W}\mathbf{x}_{[c]}$$

$$\text{Equation } 2b \quad \mathbf{b}_{[c]} = \mathbf{V}\mathbf{y}_{[c]}$$

Figure 30 displays the transmission function for $\delta = 0.2$.

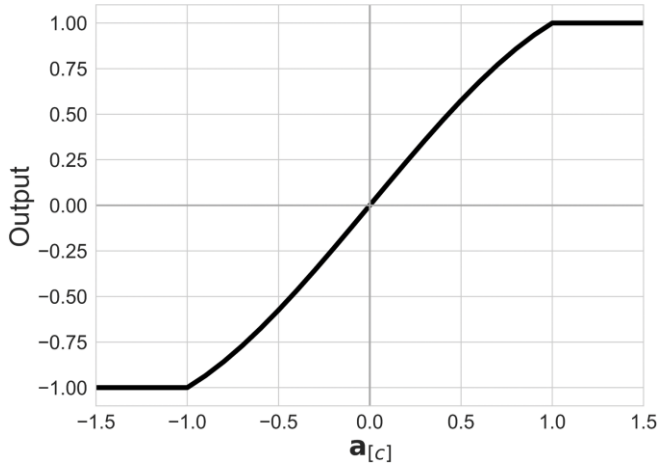

Figure 30. The MF-BAM's transmission function for δ = 0.2.

2. 3. Learning rule

The MF-BAM uses a Hebbian/anti-Hebbian learning rule to modify its weight connections according to equation (3a) and (3b):

$$\text{Equation } 3a \quad \mathbf{W}_{[k+1]} = \mathbf{W}_{[k]} + \eta\left(\mathbf{y}_{[0]} - \mathbf{y}_{[c]}\right)\left(\mathbf{x}_{[0]} + \mathbf{x}_{[c]}\right)^{\mathrm{T}}$$

$$\text{Equation } 3b \quad \mathbf{V}_{[k+1]} = \mathbf{V}_{[k]} + \eta\left(\mathbf{x}_{[0]} - \mathbf{x}_{[c]}\right)\left(\mathbf{y}_{[0]} + \mathbf{y}_{[c]}\right)^{\mathrm{T}}$$

where η is the learning parameter, k is a given learning trial, $\mathbf{x}_{[0]}$ and $\mathbf{y}_{[0]}$ are the initial incoming patterns presented to the **W** and **V** matrices and $\mathbf{x}_{[c]}$ and $\mathbf{y}_{[c]}$ are their respective reconstructed patterns at the iteration cycle c. Figure 31a) shows an example for a UL of how an incoming signal will go through **W**, **V,** and back to **W** to obtain the reconstructed generated pattern $\mathbf{y}_{[1]}$ and mark the end of an iteration cycle. Furthermore, it also shows how this process can continue for c number of iteration cycles. For the SL, the iteration cycle is shortened as the incoming pattern $\mathbf{x}_{[0]}$ traverses

W, and the incoming "teacher" pattern $\mathbf{y}_{[0]}$ passes **V** simultaneously to produce the reconstructed "teacher" pattern $\mathbf{y}_{[1]}$ and the reconstructed incoming patterns $\mathbf{x}_{[1]}$, respectfully. Likewise, $\mathbf{x}_{[1]}$ and $\mathbf{y}_{[1]}$ can be sent back for a *c* number of iteration cycles to give $\mathbf{x}_{[c]}$ and $\mathbf{y}_{[c]}$. Figure 31b) shows this process for the SL, where the black lines represent the process for an iteration cycle of 1, and the dashed lines show how information can circulate back in **W** and **V** for *c* iteration cycles.

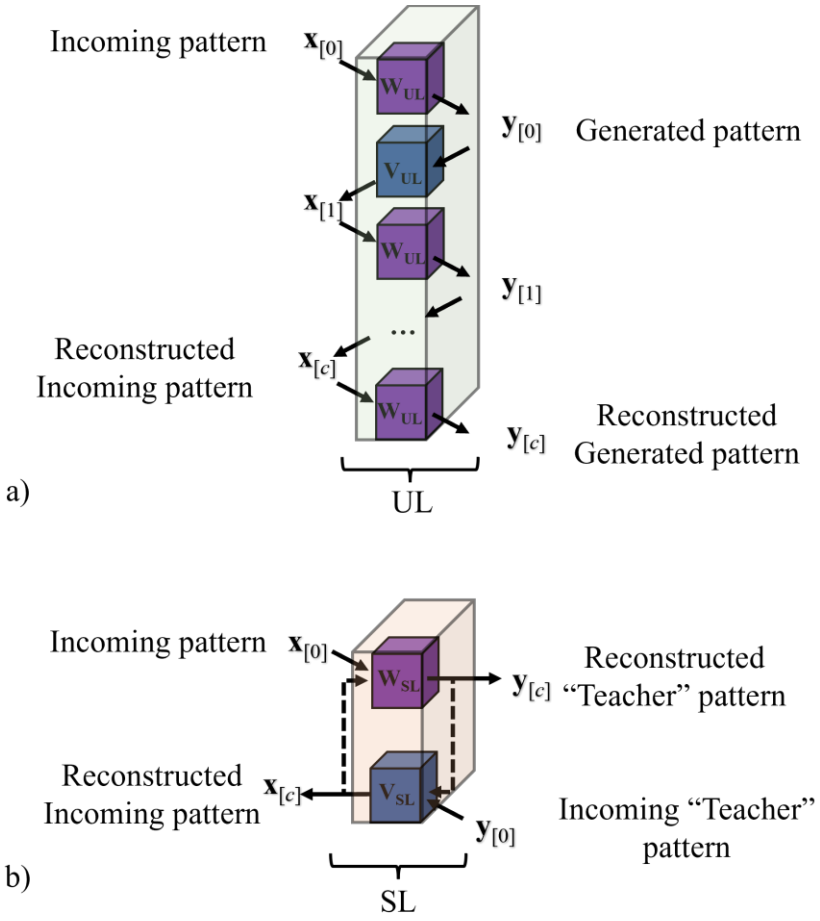

Figure 31. Box diagram of a) UL's iterative cycle and b) SL's iterative cycle.

It is important to note that the Hebbian/anti-Hebbian learning rule (equation (3a) and (3b)) can be driven by the incoming pattern, $\mathbf{x}_{[0]}$ and/or $\mathbf{y}_{[0]}$, and their reconstructed version at any *c* iteration cycle. As such, once all desired terms are acquired ($\mathbf{x}_{[0]}$, $\mathbf{x}_{[c]}$, $\mathbf{y}_{[0]}$, and $\mathbf{y}_{[c]}$) in a network layer, a weight update for both **W** and **V** can occur, resulting in the learning trial, *k*, incrementing

by one and the *c* variable resetting. Furthermore, during learning, weight update is said to have stabilized when $\mathbf{x}_{[0]} = \mathbf{x}_{[c]}$ or $\mathbf{y}_{[0]} = \mathbf{y}_{[c]}$ (i.e., when the input layer and output layer have become constant in their activation). This means that despite continuing the weight update, both **W** and **V** remain centered at a particular set of values which represents minima in the solution space. In other words, a network layer's stabilization means that its internal dynamics have produced stable attractors representing the learned patterns. This stabilization can only be guaranteed if the learning parameter (η) is small and positive (see [Chartier & Boukadoum, 2006] for more details). Figure 32a) shows an example of a UL and 30b) a SL, stabilizing during learning to minima when $\mathbf{x}_{[0]} = \mathbf{x}_{[c]}$ (Red) and $\mathbf{y}_{[0]} = \mathbf{y}_{[c]}$ (Blue) for $c = 1$.

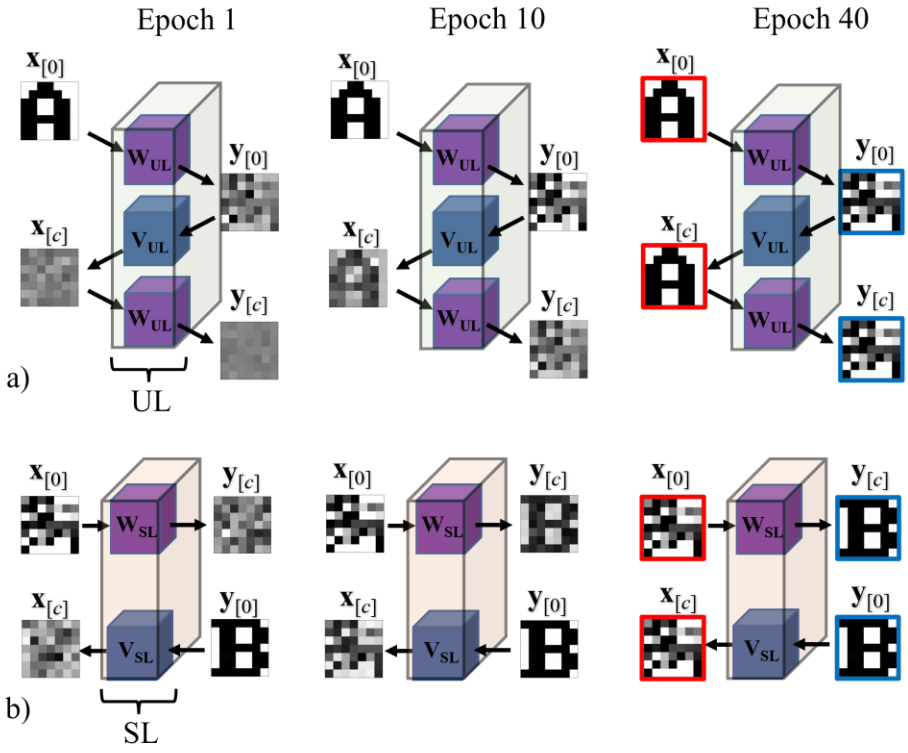

Figure 32. Example of a) UL's and b) SL's learning for an iterative cycle of c = 1 and for 40 epochs on a simple association task.

2. 4. MF-BAM's learning procedure

To better understand how the MF's architecture impacted the BAM's behavior, each module was trained individually to highlight this effect. As such, in the MF-BAM, information cycling begins with its MF component. Each UL of the MF functions the same as described in section 2.3., where an input pattern arrives at the input layer, denoted as $\mathbf{x}_{UL[0]}$, and traverses through the weight matrix $\mathbf{W}_{UL}$ to generate $\mathbf{y}_{UL[0]}$ at the output layer. This generated pattern $\mathbf{y}_{UL[0]}$ is then sent back to the input layer via the $\mathbf{V}_{UL}$ weight matrix to reconstruct the incoming pattern, denoted as $\mathbf{x}_{UL[c]}$. The first iterative cycle is complete once $\mathbf{x}_{UL[c]}$ is sent back to the $\mathbf{W}_{UL}$ matrix to generate $\mathbf{y}_{UL[c]}$. Once all four terms, $\mathbf{x}_{UL[0]}$, $\mathbf{y}_{UL[0]}$, $\mathbf{x}_{UL[c]}$, and $\mathbf{y}_{UL[c]}$, are obtained for a given UL, a weight update can occur as per equations (3a) and (3b). The $\mathbf{y}_{UL[c]}$ can then be sent to the following UL as input, and the c can be reset for the following incoming signal. This initiates the same process for the subsequent UL. In other words, when an initial pattern $\mathbf{p}$ arrives in the MF component, it triggers a cascade of this cyclic process until the final network layer, UL_l, produces a stable $\mathbf{y}_{UL_l[c]}$. For simplicity, each UL's $\mathbf{y}_{UL[c]}$ will be referred to as a hidden generated representation of that network layer, denoted by $\mathbf{h}$. Thus, UL_1 will generate a $\mathbf{h}_1$, UL_2, a $\mathbf{h}_2$, …, and UL_l, a $\mathbf{h}_l$. Figure 33 shows this process for the MF and its bidirectional nature.

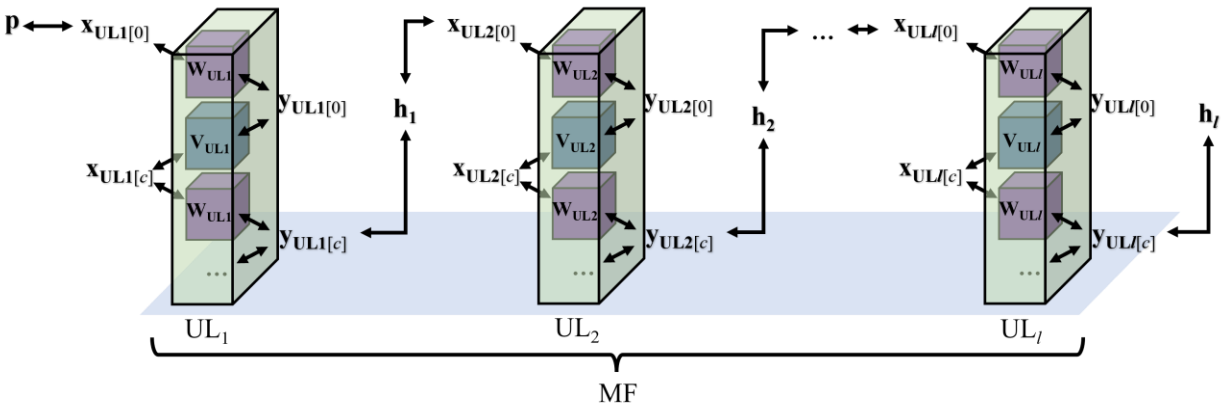

Figure 33. Box diagram of the MF's learning process.

In sum, in the MF, when an input pattern arrives at a UL, the signal can circulate between the $\mathbf{W}_{UL}$ and $\mathbf{V}_{UL}$ matrices for a c number of iteration cycles. Once all four desired terms are obtained ($\mathbf{x}_{UL[0]}$, $\mathbf{y}_{UL[0]}$, $\mathbf{x}_{UL[c]}$, $\mathbf{y}_{UL[c]}$), a weight update occurs, c is reset, and a learning trial (k) has passed for that UL. Once all input patterns have been individually presented, it is said that one epoch (e) has passed. This learning process can continue until a maximum number of epochs has been reached or if the entirety of the MF has stabilized. In other words, until the maximum number of repetitions has passed or when weight updates in every UL have stabilized.

Because the MF doesn't require error to be backpropagated, weight updates can occur independently for each UL. Furthermore, since all ULs are sequentially linked, a cascade of weight update occurs, and a stable final behavior $\mathbf{h}_l$ is obtainable from UL_l once all previous network layers have stabilized. This independence also means that each UL can continue its weight updates even if it has stabilized. In other words, if the input signal is constant and a UL has stabilized, no significant change should occur while maintaining the updates. Figure 34 shows the MF's learning process and how its deeper networks require more time to stabilize and produce a consistent hidden representation of a solution on a simple association task. This is highlighted in Figure 34, where, as an individual UL stabilizes, their generated representations even outs, as shown by the contour color shifting from blue (initial representation) to red (stable solution).

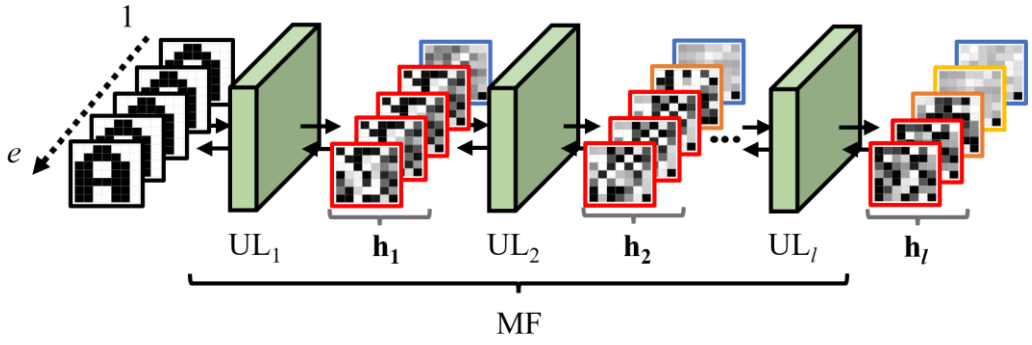

Figure 34. Visual representation of MF's learning process on a simple paired association task.

Once the MF has learned to produce a stable set of generated patterns $\mathbf{h}_l$ from its final UL, these are then given to the BAM component for learning. From its SL, this component can start associating the set of $\mathbf{h}_l$ with their corresponding set of output targets, **o**. As previously described in section 2.3., this learning process is shortened compared to the MF. As $\mathbf{h}_l$ arrives as the incoming pattern in the input layer $\mathbf{x}_{SL[0]}$ and its corresponding target **o** as the teacher pattern in the output layer $\mathbf{y}_{SL[0]}$, they traverse $\mathbf{W}_{SL}$ and $\mathbf{V}_{SL}$ simultaneously to give $\mathbf{y}_{SL[c]}$ and $\mathbf{x}_{SL[c]}$, respectively. Once all four terms are obtained ($\mathbf{x}_{SL[0]}$, $\mathbf{y}_{SL[0]}$, $\mathbf{x}_{SL[c]}$, and $\mathbf{y}_{SL[c]}$), a weight update can occur, and the learning trial (*k*) is incremented by one, and *c* is reset. Once all $\mathbf{h}_l$ and their corresponding **o** have been individually presented, it is said that one epoch (*e*) has passed. This learning process can continue until a maximum number of epochs has been reached or if the SL has stabilized. Figure 35 shows this process for the BAM component where the solid lines represent an iteration cycle *c* = 1, and the dashed lines show how information can circulate back for several *c* iteration cycles.

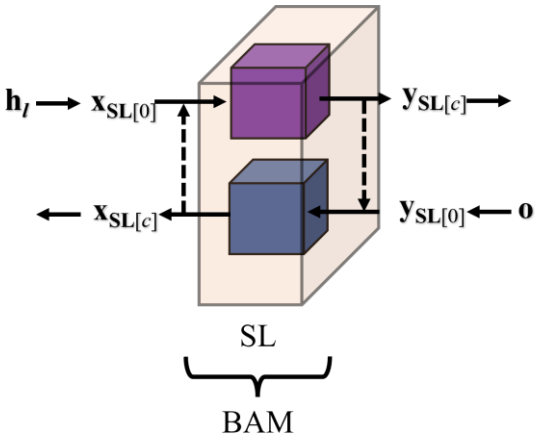

Figure 35. Box diagram of the BAM component's learning process.

Conjunctively, the MF and BAM component function by having the prior learn to produce a set of generated patterns that the latter can associate with a given target. This entire rationale is shown in Figure 36 for a simple association task (pair patterns A and B). As the MF receives constant information, its individual UL stabilizes to a set of weight values, producing a stable

generated pattern $\mathbf{h}_l$ (red) that can then be given to the BAM to learn to pair it with B. Once learning is complete, presenting A to the MF-BAM will generate all hidden representations and the desired output, pattern B. Furthermore, due to its bidirectional nature, giving B to the output layer of the SL can also retrieve all hidden representations and reconstruct pattern A at the input layer of UL_1.

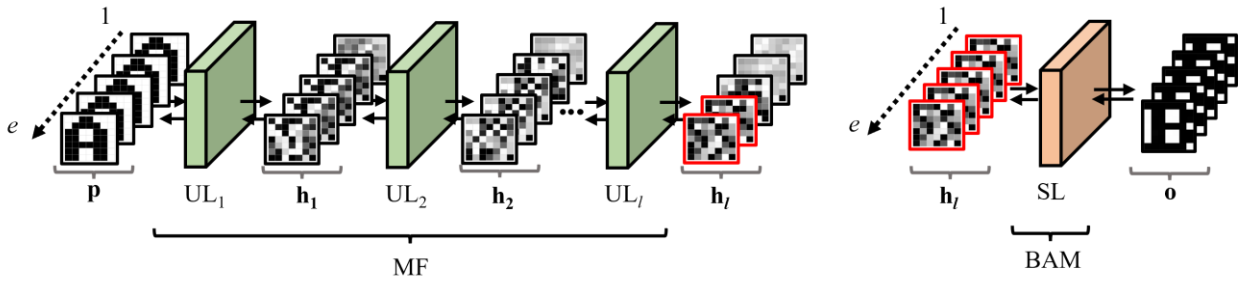

Figure 36. Box diagram of the MF-BAM's learning process for a simple association task.

3. General Methodology

Multiple tasks were used in different simulations to evaluate the MF-BAM's learning capabilities. For all simulations, the learning and recall procedure remained identical, with the transmission function parameter (δ) set to 0.2 and the learning parameter (η) set to respect $0 < \eta < 1/max(m, n)$. The use of input bias neurons initialized to a value of 1 was added to every input layer (ULs and SL) of the MF-BAM **[**Aggarwal, 2018, pp. 4-21**]**. For all network layers, weights were randomly initialized following a uniform distribution with values between -0.1 and 0.1. The number of units per layer (u) and the number of unsupervised network layers (l) were set according to the simulation condition. During the learning process, the number of cycles before the weight updates was set to $c = 1$. The MF and the BAM component stopped learning when the maximum number of epochs (e) for a given simulation was achieved. The entire model and testing protocols were created and implemented with the Python programming language installed from Anaconda [Anaconda Software Distribution, 2020] and run on a RTX3090 graphics card.

3.1. Measurements

Three measurements were used to ensure that the MF-BAM's behavior would be adequately evaluated: learning error, decision zones, and recall performance.

3. 1. 1. Learning Error. Learning error was measured with mean squared error (MSE). For the MF component, the MSE value represented the difference between the generated patterns at interaction cycle *c* and those at *c*-1. For the BAM component, MSE described differences between the predicted output and the actual target. Thus, low MSE values (< 0.001) were sought out for both parts, as they indicated a stable generated representation (MF) and successful input-target associations (BAM).

3. 1. 2. Decision Zones. For all tasks containing 2-dimensional input vectors, decision zones were generated by presenting test inputs. These test inputs were created by mapping out the 2D space between the [-1, -1] and [1, 1] coordinates in steps of 0.01. Thus, 40 401 test inputs of 2 dimensions were created. Using decision zones as a measurement can be equivalent to training the network on a subset of data and testing the model on novel instances, like what is traditionally done in train-test split protocols [Raschka, 2018]. As such, this measurement allows seeing the overall classification behavior of the model.

3. 1. 3. Recall Performance. To ensure the model could learn the appropriate associations to provide a steady solution, recall performances were measured in two ways. The first was simply by recording the average recall performance (how many patterns were successfully classified) and the standard deviation. The second was by only counting perfect scores, i.e., successes vs. failures. A step-function was applied to the SL's output to facilitate recall performance calculations, where any node values ≥ 0 would be rounded up to 1, and anything < 0 would be given a -1 value. This step function was strictly applied for recall performance calculations only.

3. 2. General Learning Procedure

The learning procedure was conducted in two phases: UL learning (3.2.1.) and SL learning (3.2.2.).

3. 2. 1. - UL learning. For the MF, each epoch (*e*) consisted of the following steps:

1. Creation of a list, **P**, containing all input patterns **p**.
2. Selection of a random input pattern **p** to be given to the first UL.
3. Incoming pattern for UL_l, denoted by $\mathbf{x}_{\mathbf{UL}l[0]}$.
4. Computation of the generated pattern $\mathbf{h}_l$, denoted by $\mathbf{y}_{\mathbf{UL}l\,[c]}$, according to equations (1) and (2), and as described in sections 2.3 and 2.4.
5. Computation of weight update for $\mathbf{W}_l$ and $\mathbf{V}_l$ according to equation (3).
6. Relaying of $\mathbf{h}_l$ to the following UL.
7. Repetition of 3-6 for the following UL until all ULs have been updated.
8. Repetition of 2-7 until all input patterns in **P** have been exhausted.

Learning continued until the *e* condition was reached for a given simulation.

3. 2. 2. - SL learning. For the BAM, it was deemed that an epoch had passed after the following steps were completed:

1. Creation of a list, **HO**, containing all generated patterns from the final UL of the MF ($\mathbf{h}_l$) and their corresponding targets (**o)**.
2. Selection of a random pair of generated patterns and their corresponding targets to be given as $\mathbf{x}_{\mathbf{SL}[0]}$ and $\mathbf{y}_{\mathbf{SL}[0]}$, respectfully.
3. Computation of $\mathbf{x}_{\mathbf{SL}[c]}$ and $\mathbf{y}_{\mathbf{SL}[c]}$ according to equations (1) and (2) and as described in sections 2.3 and 2.4.
4. Weight update according to equation (3).

5. Repetition of 2-4 until all pairs of patterns in **HO** have been exhausted.

Learning continued until the *e* condition was reached for a given simulation.

3. 3. General Recall Procedure

The recall procedure for the MF-BAM was performed in the following fashion:

1. Selection of a desired input pattern to be given to the first UL as $\mathbf{x}_{\mathbf{UL}[0]}$.
2. Computation of $\mathbf{y}_{\mathbf{UL}[c]}$ according to equations (1) and (2).
3. Relaying $\mathbf{y}_{\mathbf{UL}[c]}$ to the following UL to be used as $\mathbf{x}_{\mathbf{UL}\boldsymbol{l}[0]}$.
4. Repetition of 2-3 for each UL until the final generated representation is obtained, $\mathbf{y}_{\mathbf{UL}\boldsymbol{l}[c]}$.
5. Relaying $\mathbf{y}_{\mathbf{UL}\boldsymbol{l}[c]}$ as $\mathbf{h}_{\boldsymbol{l}}$ to the SL to be used as $\mathbf{x}_{\mathbf{SL}[0]}$.
6. Computation of $\mathbf{y}_{\mathbf{SL}[c]}$ according to equations (1) and (2).

A comparison between $\mathbf{y}_{\mathbf{SL}[c]}$ and its original corresponding target $\mathbf{y}_{\mathbf{SL}[0]}$ was performed by following section 3.1.3 to determine if the model made a mistake. Refer to Appendix A for a summary of the acronyms used (Table A.1.) and a comprehensive guide to the notations (Table A.2.).

4. Impact of the Number of Units per Layer (*u*)

In this section, the main two goals were to evaluate whether the MF-BAM could solve different nonlinear tasks and to isolate the effect of *u* (number of units per layer) on performance. As such, several nonlinear tasks separated into two types, bipolar (*N*-bit) and continuous (Moons, Eclipse, and Suns), were used. Furthermore, the number of units per layer (*u*) was systemically varied while the number of unsupervised network layers (*l*) and epochs (*e*) were kept constant.

4. 1. The *N*-bit Parity Problem

The MF-BAM was initially tested on the *N*-bit parity problem, a classical and popular benchmark for ANNs. The task involves determining if a sequence of binary digits (consisting of 0s and 1s) contains an odd or even number of 1s. The complexity of the challenge increases as the sequence length, represented by *N*, increases. Notably, in the bipolar version, the solution involves counting the number of -1s instead of 1s and determining if it's odd or even.

4. 1. 1. Methodology. The *N*-bit parity problem was implemented by creating bipolar input pattern vectors, **p**, containing values of -1 or 1, of *N* dimensionality (i.e., $\mathbf{p} = [p_1, p_2, \ldots, p_N]$). The total number of inputs, **P**, determined by 2^N, was created by mapping the bipolar input space. The target for each input was determined by equation 5 and was represented as a 1-dimensional vector containing either the values of -1 or 1, signifying odd or even counts of -1s, respectively.

Equation 5

$$For\ \mathbf{p}\ in\ \mathbf{P}, \varphi(\mathbf{p}) = \begin{cases} 1, if \prod_{k=1}^{N} \mathbf{p}_k \ \ is\ positive \\ -1, if \prod_{k=1}^{N} \mathbf{p}_k \ \ is\ negative \end{cases}$$

To evaluate different levels of difficulties, *N* was varied from 2 to 6. Higher *N* values were also studied but were omitted from this section since the results followed the same trends. An example of the 2-bit and 3-bit can be seen in Table 5, where the inputs and their corresponding targets are shown for both tasks.

Table 5

Inputs and corresponding targets for the 2- and 3-bit tasks.

2-bit			**3-bit**							
Inputs		Targets	Inputs			Targets	Inputs			Targets
1	1	1	1	1	1	1	-1	-1	-1	-1
-1	-1	1	1	-1	1	-1	-1	1	-1	1
1	-1	-1	1	1	-1	-1	-1	-1	1	1
-1	1	-1	1	-1	-1	1	-1	1	1	-1

For these simulations, the model's architecture was set to $l = 4$, and u varied from 1 to 150. The learning and recall followed the procedure in section 3. Learning for the MF and BAM was set to 2500 epochs (e) each. The task was repeated 100 times for every bit size for reliability assessment. Average learning times, decision boundaries, and total recall performances were also reported.

4. 1. 2. Results. The average number of epochs and mean squared error (MSE) for the 2-bit task are shown in Figure 37a). In general, MF learning was characterized by a decrease in error (MSE) over time (number of epochs), stabilizing at low MSE values (< 0.001). However, when looking at individual ULs' MSE, spikes in error are observed when previous ULs stabilize. Nevertheless, after these spikes, the MSE drops to low values. For example, when $u = 100$, the MSE of the second UL (orange) spiked once the first UL (blue) stabilized but decreased again to low values.

For the BAM component, learning was also characterized by a decrease in MSE as the number of epochs increased. However, the BAM did not always stabilize at low MSE values. For the 3 and 5 units per layer conditions, the BAM stabilized on high MSE values (≈ 1.75 and ≈ 0.75, respectively), which hinted that it did not successfully learn the task. However, for other conditions ($u \geq 10$), it stabilized at an MSE value lower than 0.001.

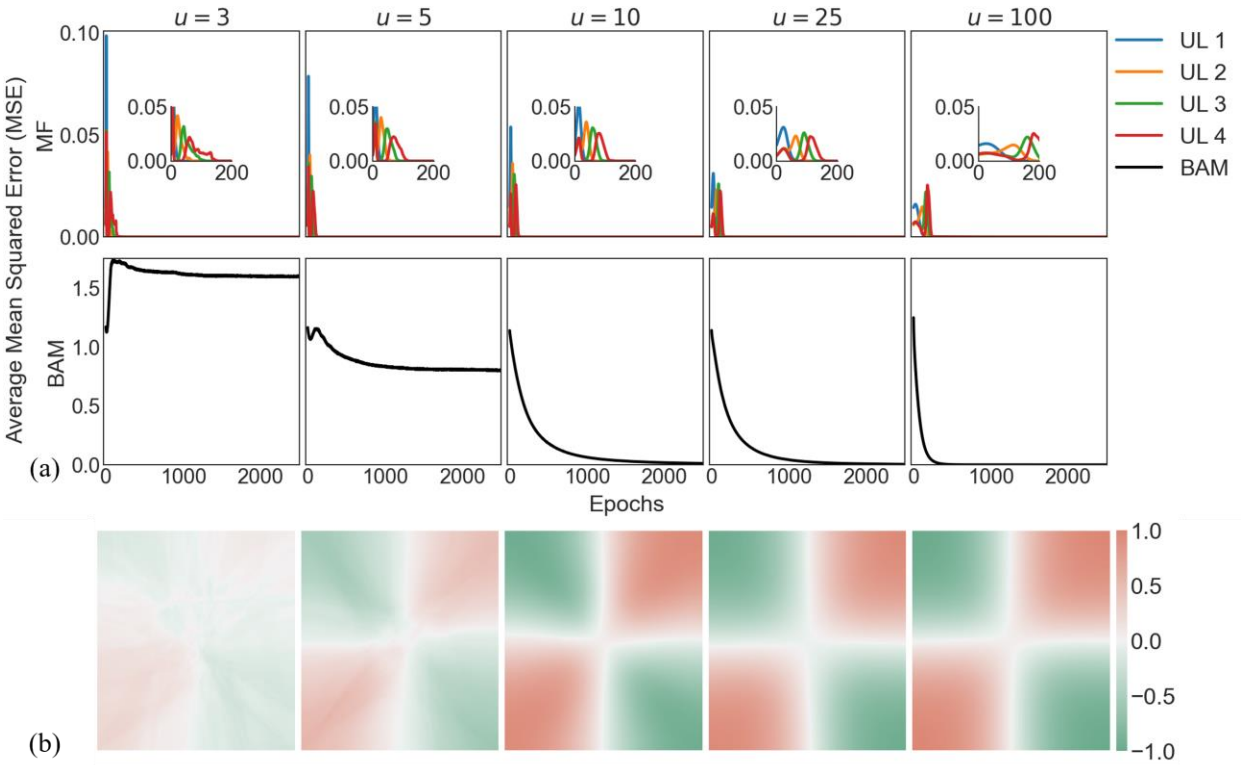

Figure 37. a) Average mean squared error during learning for both the MF and BAM component for the 2-bit task under different units per layer (*u*) conditions. b) Average decision boundaries for the 2-bit task under the same conditions.

Results of average decision zones after 100 trials for the 2-bit task are shown in Figure 37b). For $u = 3$, there was no clear observable decision boundary, confirming the failure of the MF-BAM to learn the 2-bit task as suggested by the high values of MSE. However, at $u = 5$, the decision boundary containing the shape of the appropriate solution began to emerge, albeit with prediction values very close to 0, which coincided with the BAM's high error. Increasing *u* resulted in these values becoming increasingly closer or equal to –1 and 1, ultimately giving rise to the decision zones that best encapsulate the solution to the task.

Performances for all *N*-bit tasks across the 100 trials for each *u* condition are shown in Figure 38. For each version of the *N*-bit, 100% recall performance was achieved once *u* was large enough. For instance, perfect performance was observed at $u \geq 10$ for the 2-bit task but at $u \geq 30$

for the 3-bit. Generally, the MF-BAM would begin to correctly learn the task at a given number of units per layer and quickly reach perfect performance with only a few additional units per layer.

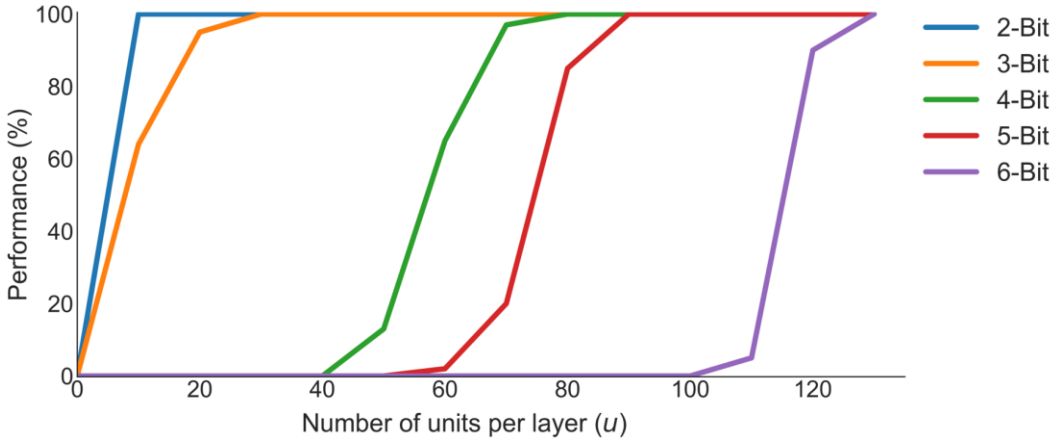

Figure 38. Recall performance of the MF-BAM for the N-bit task in function to the number of units per layer (*u*).

4. 2. Continuous Value Tasks

The MF-BAM was also tested on other sets of popular nonlinear tasks. However, the inputs now consisted of continuous values rather than bipolar ones. More specifically, the Moons, the Eclipse, and the Suns tasks were used. Like section 4.1, the number of units per layer was manipulated, while *l* and *e* were kept constant.

4. 2. 1. Methodology. To create the Moons, Eclipse, and Suns tasks, 40 two-dimensional input vectors were generated to represent a Cartesian plane's x and y coordinates. These inputs were divided into two classes and assigned one-dimensional target vectors of value -1 or 1. The Moons and Eclipse were generated using the Scikit Learn library in Python (https://scikit-learn.org/stable/; [Buitinck et al., 2013; Pedregosa et al., 2011]) with the *make_moons* and *make_circles* functions, respectively. The parameters used for both functions were *n_sample* = 40 and *noise* = 0, and the additional *factor* parameter was set to 0.7 for the Eclipse. The Suns task was

generated through a custom function, which spawned two circles, one of which was 0.5 times smaller than the other, and its center offset to the left by a factor of 1.2. The input vector values for each task were scaled between -1 and 1, and the goal was to associate them with their respective target vectors. All three tasks are displayed in Figure 42. Although larger data sets were also examined, their results were not included as they mirrored the trend in the presented data.

To evaluate the effect of *u*, the value of *l* was fixed at 4, and five conditions were created where *u* ranged from 100 to 500 in increments of 100. The learning and recall process followed the procedure described in section 3. For every condition, *e* was set to 2500 for both the MF and BAM components. To assess reliability, the learning and recall process was conducted for 100 trials. The average learning and recall performance and decision zones for all trials were reported.

4. 2. 2. Results. The MF and BAM's average number of epochs and MSE are shown in Figures 39, 40, and 41 for the Moons, Eclipse, and Suns task, respectively. In each task, the learning error of the MF component followed the same sequence found in the N-bit task. As each previous UL stabilized, a minor blip in MSE was observed for the following ULs and afterward quickly decreased and stabilized at low values (<0.001). In the BAM component, for the Moons task, the MSE error was higher than 0.0001 when the number of units per layer was 300 or less, suggesting unsuccessful learning of the input-target pairs.

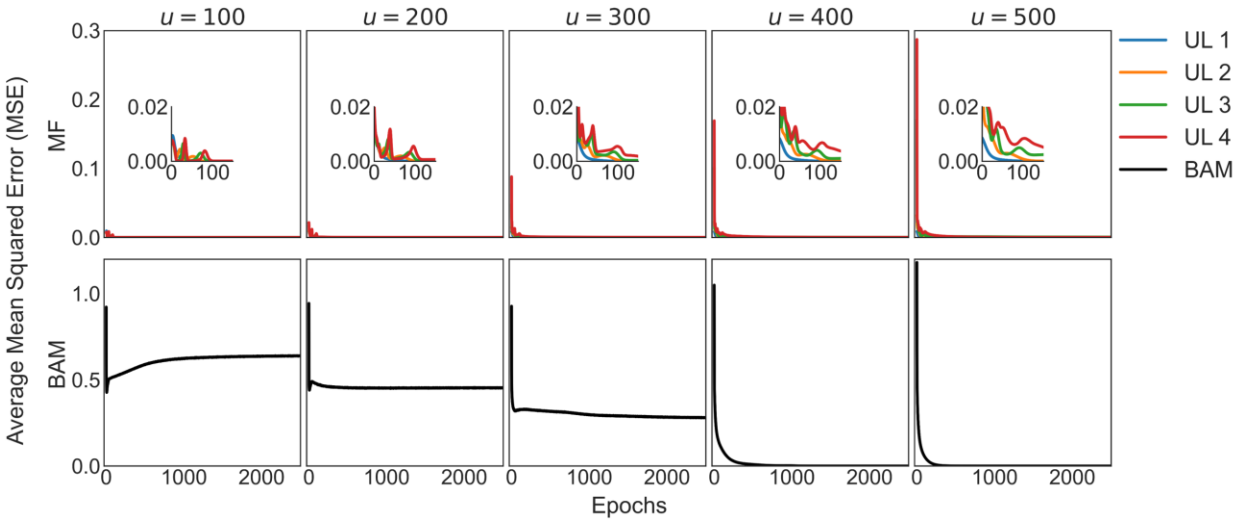

Figure 39. Average mean squared error during learning for both the MF and BAM components for the Moons task in function of the number of units (u) per layer.

For the Eclipse task, the BAM's MSE values were below 0.001 when $u \geq 200$. However, in the 100 units per layer condition, the observed trend indicates that with more learning time (more epochs), the BAM would reach lower MSE values.

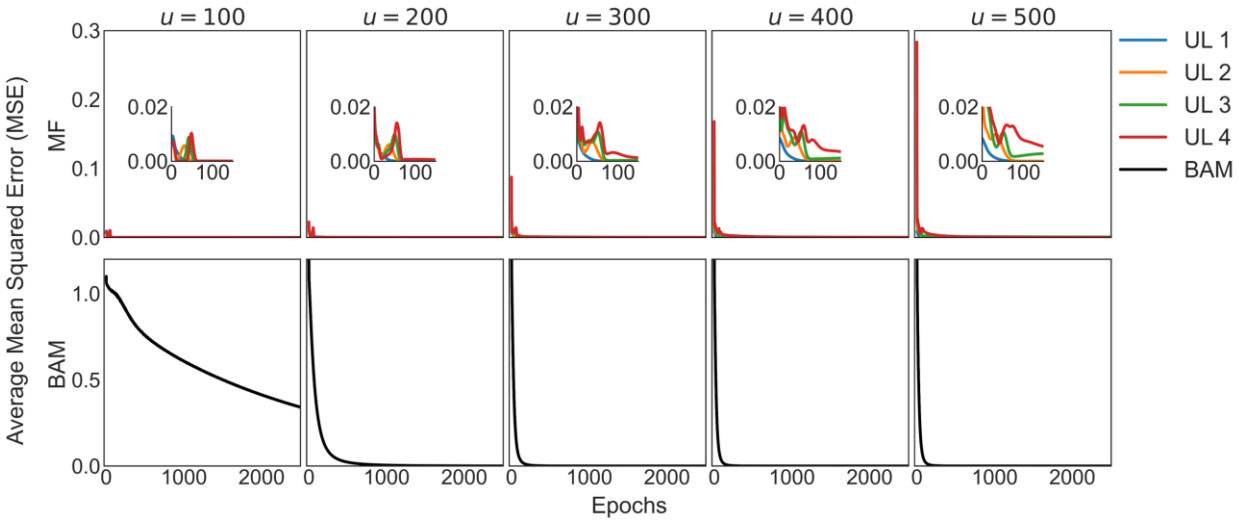

Figure 40. Average mean squared error during learning for both the MF and BAM components for the Eclipse task in function of the number of units (u) per layer.

For the Suns task, the BAM's MSE values were higher than 0.001 regardless of the units per layer condition. This is shown in Figure 41.

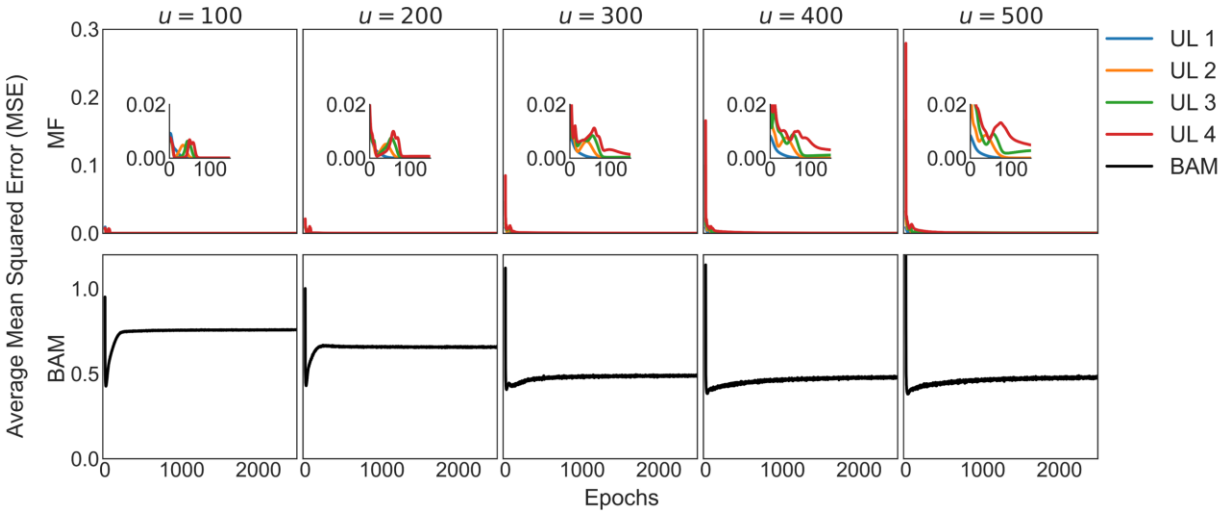

Figure 41. Average mean squared error during learning for both the MF and BAM components for the Suns task in function of the number of units (u) per layer.

Figure 42 shows the MF-BAM's decision zones for all three tasks. Regarding the Moons, with $u < 200$, the MF-BAM could only generate a quasilinear decision boundary. As u increased, the more nonlinear the separation became. With $u \geq 400$ units, the decision boundary generated by the MF-BAM allowed perfect classification. Regarding the Eclipse task, it was much easier for the model to achieve an ideal classification ($u \geq 200$ units). Finally, the Suns task showed that as u increased, the decision boundary was more and more curved. Still, the model couldn't learn to associate the three patterns inside the inner circle with the appropriate target, as shown by the misclassification zone in this area.

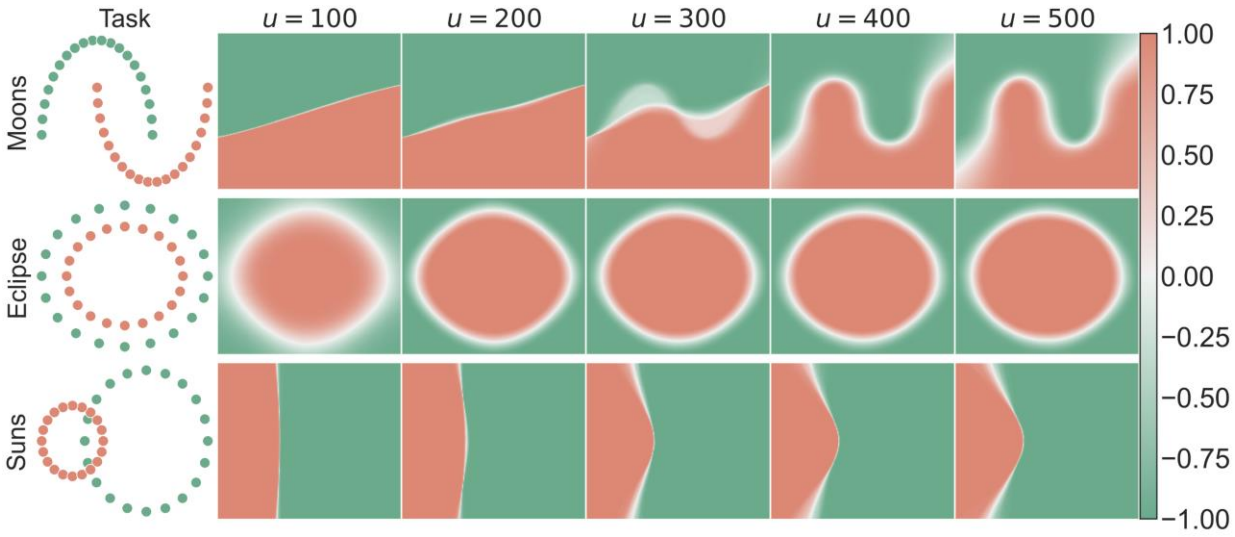

Figure 42. Average decision boundaries of the MF-BAM for the Moons, Eclipse, and Suns task in function of units (*u*) per layer.

Average recall performances and standard deviations for all three tasks are presented in Table 6. For the Moons and Eclipse, average recall performance improved as *u* increased. A perfect recall was achieved at $u \geq 400$ for the Moons and $u \geq 200$ for the Eclipse. Further increasing of *u* had no effect on performance or consistency (confirmed by standard deviation = 0).

Concerning the Suns task, no perfect recall was observed under any *u* value, with the highest average performance being 87.45% at the 300 units per layer condition. Standard deviation increased as *u* increased, ranging from 1.82 for $u = 100$ to 3.76 for $u = 500$.

Table 6

Average recall performance and standard deviation for all three nonlinear tasks.

		Number of units per layer (*u*)				
		100	200	300	400	500
Moons	Mean	83.05	87.85	92.20	100.00	100.00
	Std.	2.11	2.03	5.37	0.00	0.00
Eclipse	Mean	92.08	100.00	100.00	100.00	100.00
	Std.	9.70	0.00	0.00	0.00	0.00

Suns	Mean	80.12	82.42	87.45	86.40	86.80
	Std.	1.82	1.89	2.78	3.51	3.76

4. 3. Impact of the Number of Units per Layer Discussion

In section 4, the *u* parameter was manipulated while keeping *l* constant at 4, and it was first revealed that the MF-BAM could consistently learn nonlinear tasks. As shown in sections 4.1 and 4.2, 100% performance was always achieved in the *N*-bit, eclipse, and moons tasks. Furthermore, increasing *u* produced lower learning errors (Figures 37a), 39, 40, and 41), more explicit bipolar decision boundaries meaning improved classification behavior (Figures. 37b) and 41), and better recall performance (Figure 37 and Table 6). If *u* was too low, the SL would struggle to learn a task consistently or at all, leading to high learning errors, poor performance, and unclear decision boundaries. For example, in a condition with $u = 5$ in the 2-bit task, the SL presented high MSE values (> 0.7) and only learned the task half the time (performance $\approx 50\%$). On the flip side, when $u = 100$, the supervised network layer stabilized to low MSE values > 0.001 and achieved consistent 100% performance. When looking at the model's classification behavior in Figure 37b), the appropriate decision boundaries showing four regions in the corners of the input space were not present when $u = 3$ and were still unclear when $u = 5$. However, these increased in clarity as the number of units rose. In these cases, predictions for unseen inputs were higher nearer the corner than the center. Inputs very close to the Cartesian axes predict a specific class membership around 0.

Moreover, the minimum value of *u* required to solve a task varied from task to task. In the *N*-bit tasks, perfect performance was achieved all the time when *u* was > 10 for 2-bit, > 30 for 3-bit, > 65 for 4-bit, > 85 for 5-bit, and > 125 for 6-bit. The higher the number of bits, the higher the number and size of inputs required for the model to learn the task. Interestingly, 200 and 400 units

were needed to achieve steady, perfect performance for the Eclipse and Moons tasks, respectively. The inputs' dimensionality and quantity remained the same in both tasks, but the complexity of the decision boundaries differed. The decision boundary for the Eclipse task was circular, while a more complex boundary was required for the Moons task. Hence, increasing *u* also assisted in learning tasks with more complicated arrangements of inputs.

However, increasing *u* alone was not enough to achieve perfect learning of the Suns task. This was indicated by the SL's MSE values never reaching zero (as seen in Figure 41) and the performance plateauing at around 85% (as shown in Table 6). This can be explained by looking at the decision boundaries illustrated in Figure 42. While the general shape of the task was achieved, the three exceptions (green points inside the red circle) were not accounted for, even with 500 units. These results become interesting when contrasted with what is traditionally sought with multilayered ANNs. With such models, a train-test split protocol is usually used to avoid overfitting a task where a subset of the data is withheld during learning to ensure that the network's solution remains general. However, the results of the MF-BAM showed that despite presenting all the training data, the model did not overfit the task, and the general solution was obtained.

To sum up, results from section 4 have shown that by manipulating the number of units per layer in an architecture composed of 4 ULs, it was possible to teach the BAM module to classify the different nonlinear tasks regardless of them containing inputs with bipolar or continuous values. However, the model didn't thoroughly learn the Suns task and could only produce a generalized solution, not an overlearning (overfitting) one. To successfully solve the Suns task, it is required that the MF-BAM displays some sort of overlearning. This was investigated in the next section by manipulating the *l* parameter rather than *u*.

5. Impact of the Number of Unsupervised Network Layers (*l*)

In the previous section, a fixed number of unsupervised network layers (*l*) was used while the number of units per layer (*u*) was manipulated. This section investigated the effect of varying *l* while controlling for *u* and *e*. The Suns task was revisited to show if the MF-BAM could solve it by displaying some overlearning. Furthermore, longer simulation times for the MF were used to investigate if each UL would maintain its stability while continuing weight updates.

5. 1. Methodology

The Suns task previously described in 4.2. was used. Here, *l* varied from 1 to 13, whereas *u* was fixed to 500 units. Learning and recall performance and decision zones were collected as described in section 3, and *e* was set to 25000 for the MF and 2500 for the BAM. Learning was repeated 100 times per condition for reliability purposes, and average results were reported.

5. 2. Results

Figure 43a) shows the average MSE for both the MF and BAM. For the MF component, as shown in the previous section, while the number of learning epochs increased, small increases in error were seen for all ULs once a previous layer stabilized. This stability was maintained while the learning continued. For the BAM component, average MSE decreased until it stabilized. Results showed that with an MF of 1 layer, the BAM's error increased with the number of epochs. With an MF of 4 or 7 unsupervised network layers, the BAM's error decreased rapidly but then fluctuated around a high MSE value, reflecting results from the previous section. However, with 10 or 13 unsupervised network layers, the BAM's error lowered and stabilized at MSE values of ≈ 0.02 and < 0.001, respectively.

Figure 43b) shows average decision zones for all conditions. For $l = 1$, the model produced a linear decision boundary, misclassifying the task. For $l = 4$, the general solution from the previous

section was observed again. For l = 7, the overlapping region (peak of the parabola) was seen to lose its smoothness and showed a shift in the classification behavior. This change was accentuated for $l \geq 10$, where "pockets" appeared around the three exceptions, showing the appropriate solution for the task.

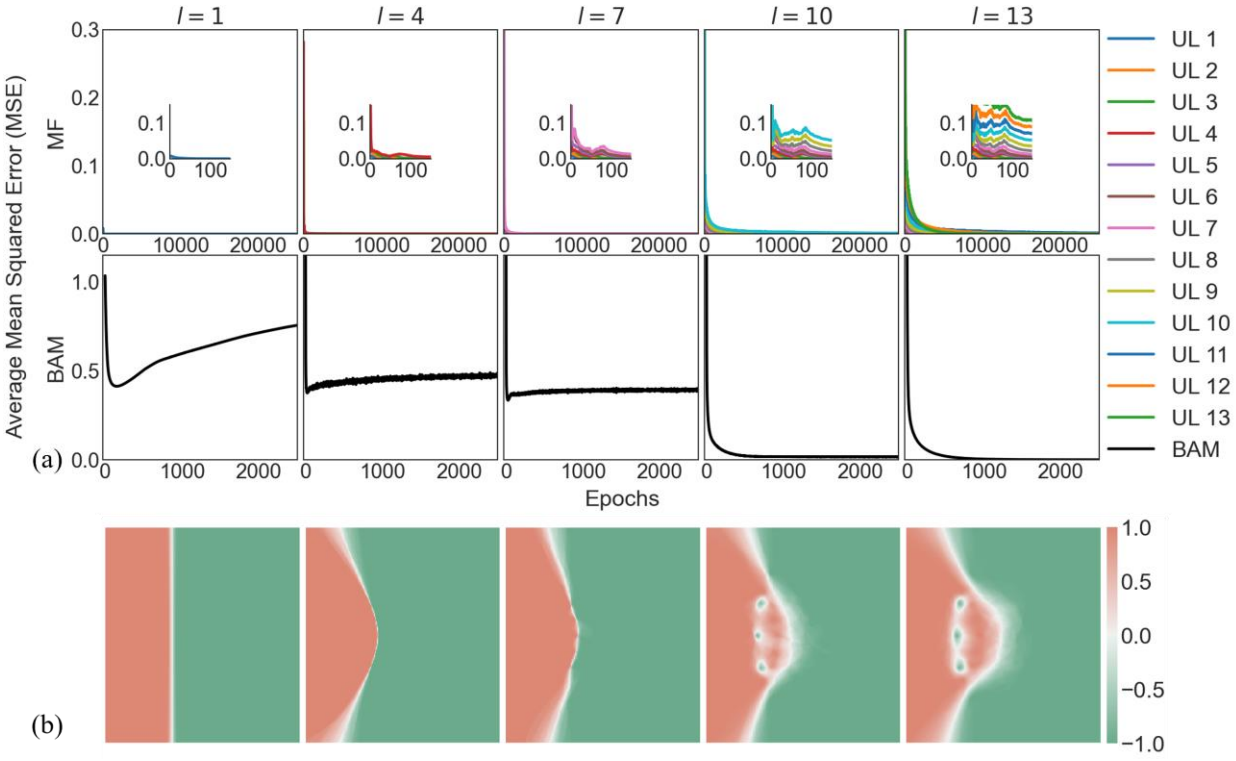

Figure 43. a) Average mean squared error for the Suns task under different number of unsupervised network layer (l) conditions. b) Decision boundaries for the Suns task under the same conditions.

Table 7 shows the average recall performance and the standard deviation for each condition of the Suns task. In general, as l increased, the recall performance increased. Notably, a perfect performance (100% input-target association with a standard deviation of 0) was achieved with l = 13.

Table 7

Average recall performance and standard deviation for the Suns task.

		Number of unsupervised network layers (*l*)				
		1	4	7	10	13
Suns	Mean	80.00	87.45	87.85	98.70	100.00
	Std.	0.00	3.10	3.32	3.05	0.00

5. 3. Impact of the Number of Unsupervised Network Layers Discussion

In section 5, results showed that by varying *l* while fixing *u* at 500, the model could get closer to providing 100% performance consistently. This perfect input-target associative behavior came with $l = 13$ (Table 7), where the SL achieved an MSE value near zero (Figure 43a)), and the three previously problematic exceptions were separated adequately in the decision boundaries (Figure 43b)). This indicates that increasing the depth of the MF impacts the final generated representations making it possible to learn to classify the Suns task. As shown in Figure 43b) for conditions $l \geq 10$, decision boundaries exhibited pocket-like zones around the three conflicting points. In other words, the MF-BAM learned exceptions to the general rule. However, it is hypothesized that the more inputs learned in those contradictory positions, the more these pockets would become pronounced, and the fewer units and layers would be needed to learn the task. Further testing should study this frequency effect. If true, this could give important insights into how learning exceptions to a general rule may depend on the architectural structure and not just exposure, as seen in young children learning to speak and call all animals, either cats or dogs.

Interestingly, in all the simulations, a common trend emerged in the MF during learning. As the first layer learned and its MSE tended toward zero, the second layer's error increased for a short time before decreasing back toward zero (Figure 43a)). This was repeated for all the subsequent layers until the last layer's error stabilized near zero. In other words, as a layer learned, its generated representations kept fluctuating, and once stabilized, low errors would be reached. This allowed the subsequent layer to receive consistent information and eventually stabilize.

Moreover, the patterns of each layer can be extracted and used separately, meaning different behaviors could be observed even if the inputs do not change. For example, patterns from the 4th layer could be used by an SL to capture the general form. At the same time, a different SL could simultaneously learn patterns from the 13th layer to capture the exceptions to the general rule.

In summary, results from section 5 showed how the MF-BAM could be tuned to either overlearn or produce a more generalized solution. This behavior change came from manipulating the *l* parameter of the MF component. Interestingly, depending on the depth of the MF, the generated patterns could be used to train the BAM module to produce a linear (generated patterns from the shallow layers), nonlinear (generated patterns from the deeper layers), or an overfitted response (generated patterns from the deepest layers). In other words, with a MF of various layers, it is possible to have all three behaviors simply by extracting the appropriate generated pattern from either the shallower, deeper, or deepest layers.

So far, sections 4 and 5 have shown that the MF-BAM can learn both a general and specific solution to a nonlinear task. These behaviors were obtainable by manipulating the *u* or *l* architectural parameter in isolation. As results have suggested so far, these parameters influence the nature of the generated patterns produced by the MF. Depending on this nature, the BAM module had poor or good performances. Thus, the goal of the next section was to understand better what type of architecture (e.g., shallower with more units vs. deeper with fewer) was more favorable. As such, both parameters *u* and *l* were systematically manipulated to study their impact on the model's behavior. Furthermore, learning time in the MF module was varied to evaluate if any architecture brought better performances in a time restraint setting.

6. Impact of Manipulating the Number of Units (*u*) and Unsupervised Network Layers (*l*)

In this section, both the number of units per layer (*u*) and number of unsupervised network layers (*l*) were manipulated. The goal was to measure the effect of varying these architectural parameters on the 2-bit parity problem and determine how it affected the BAM module's learning and associative performances. Furthermore, the MF's max learning time (*e*) was constrained and varied to further determine its effect on the BAM module. Finally, to better understand how *u* and *l* could impact the learning speed of the supervised module, a follow-up analysis was performed on the generated outputs from the different architectures.

6. 1. Methodology

The 2-bit task was used again in this simulation. As such, the inputs and targets were generated in the same fashion as described in section 4.1, i.e., inputs contained 2^2 bipolar vectors of length 2, associated with targets of either 1 or -1. To evaluate the effects of *u*, *l*, and *e* in the MF component, different architecture conditions were created. Thus, *u* varied from 5 to 100, *l* from 1 to 5 (for comparison, $l = 10$ was also investigated), and *e* was constrained from 10 and 2500 for the MF only. Furthermore, the BAM module was allowed to learn for 2500 epochs for all conditions. To help determine if the BAM had learned the 2-bit task correctly, the number of epochs required to obtain a MSE < 0.0001 in the BAM module was recorded for each trial. Additionally, average performance was computed over 100 repetitions per condition to establish reliability. Both learning and recall procedures followed what was described in section 3. Finally, density plots were generated to better illustrate the impact of *u* and *l* combinations on the BAM. These plots examined the values of each node of the generated patterns in each condition analyzed.

6. 2. Results

Figure 44a) shows the MF-BAM's average recall performance on the 2-bit task according to each architecture condition and time constraints in the MF. In general, results show that the outcome on performance when changing u and l varied for different time constraints. With little learning times (e = 100), performance was higher for shallower (smaller l) and smaller networks ($u \leq 30$) rather than deeper ($l > 1$) or larger networks ($u \geq 70$). For longer learning times ($e \geq 500$), the opposite effect could be seen, i.e., higher l and u led to better performances. Alternatively, when e = 250, a mixed effect was observed, as 2 to 5 ULs yielded better performances than 1 or 10.

More specifically, results showed that if learning time was too low (e = 10), no configuration of the MF's parameters could lead to a consistent BAM performance increase. However, when e = 100, the performance achieved from ULs 1 to 4 resembled an inverted "U" shape. That is, it initially increased at low levels of u, reached a maximum performance before 30 units, and subsequently plummeted towards zero as u further increased. In general, in this time constraint, the lower l was, the better performance was; e.g., the highest "peak" performance achieved was 98% when l = 1 (dark green) and u = 23. In fact, for l = 5 (light green) or 10 (yellow), performance did not increase.

When e = 250, performance when l = 1 still exhibited the same inverted "U" shape but with slightly higher scores, where 100% was achieved once with u = 25. For l = 2 (orange), performances improved to 91% at u = 10, fluctuated between 80% to 90% until u = 80, and slowly decreased to 70% at u = 100. For l = 3 (purple), performances fluctuated between 80% and 90% from 10 to 100 units, while for l = 4 (red) or l = 5, performances similarly fluctuated between 90% and 99% from 10 to 100 units. Interestingly, performances when l = 10 (or 1) also resembled an inverted-U curve. Under the e = 500 constraint, performances fluctuated between 97% and 100%

for the l = 10 and $u \geq 20$, l = 5 and $u \geq 35$, or l = 3 and $u \geq 45$ architectural conditions. For l = 3, performances fluctuated between 90% and 99% when $u \geq 10$.

Steady 100% performance was achieved when e = 1000 for the l = 10 and $u \geq 25$, l = 5 and $u \geq 50$, l = 4 and $u \geq 65$, and l = 4 and $u \geq 80$ conditions. Like before, no significant performance changes were observable with l = 1 or l = 2. Finally, when learning was completed at e = 2500, consistent 100% performance was achieved when $l \geq 3$ and $u \geq 15$. Additionally, their performance curves for the l = 1 and l = 2 conditions remained similar to those found in the e = 500 and e = 1000 constraints. Thus, showing marginal changes as learning time increased passed 500 epochs.

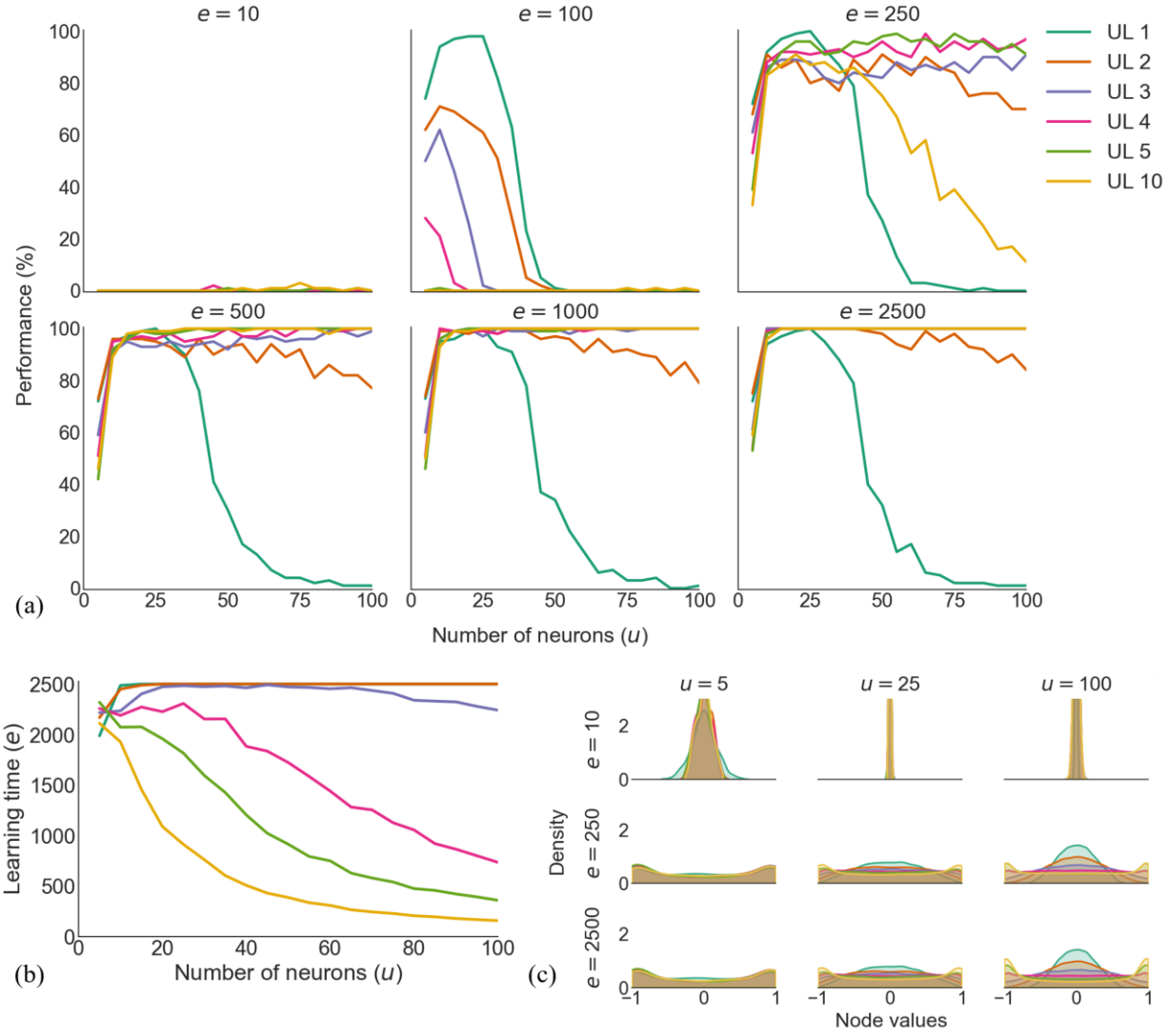

Figure 44. a) 2-bit average performance for different architectures and learning time conditions. b) BAM module's number of epochs (e) to reach MSE < 0.0001 in function of the number of units and unsupervised network layers in the MF. c) Density plots of the generated patterns from the MF in function of the number of units and unsupervised network layers conditions and time constraints.

Figure 44b) shows the average learning time for the BAM to stabilize to an MSE < 0.0001 according to the number of units and layers in the MF. If the number of epochs reached the maximum of 2500, it indicated that the network was unable to fully learn the task. As results showed, for $l \geq 3$, an increase in u and l lowered the number of epochs required for the BAM to achieve an MSE of less than 0.0001. This was evident when the comparison was made between $l = 4$ under the $u = 5$, $u = 50$, and $u = 100$ conditions. A decrease in the number of epochs was observed (2257, 1722, and 357, respectively). Increasing l to 10 led to a further reduction of average learning time, where for $u = 5$, $u = 50$, and $u = 100$, the epochs went from 2111 to 385, then to 155, respectively. On the flip side, for $l \leq 2$, the average learning time was faster for lower u values and slower for higher ones. This was seen when $l = 1$, as it took on average 1987 epochs for $u = 5$ and 2486 epochs for $u = 10$, while an MSE error smaller than 0.0001 was not achievable within 2500 epochs for $u \geq 15$.

Figure 44c) shows the density plots of the distribution of values in each unit of the generated representations at different e, u, and l constraints. Generally, two distributions could be seen: normal (bell-shaped) or bimodal. On average, representations created from networks with low l resulted in a more normal distribution with a density peak centered at 0. In contrast, those from networks with higher l had a bimodal distribution with density peaks at the bipolar extremes -1 and 1. Exceptionally, if $e = 10$, the distribution was normal with a very narrow and high density

peak around zero regardless of *u* and *l*, showing that 10 epochs were insufficient for the MF to generate patterns that were not linked to the values of the weight initialization. In general, results showed that an incremental increase of *l* produced a gradual transition from a normal distribution to a bimodal distribution. This can be seen for $e = 2500$ and $u = 100$, as the normal distribution (for $l = 1$) gradually flattened and became bimodal at -1 and 1 as *l* increased. On the other hand, increasing *u* values did not change the distribution under a given *l* but led to a stronger density peak whether normal or bimodal. Similar results were observed when *e* was increased.

6. 3. Impact of Manipulating the Number of Units (*u*) and Unsupervised Network Layers (*l*) Discussion

Results in section 6 have shown the impact of manipulating *e*, *u*, and *l* on the BAM module. Generally, results showed that when the MF's learning was not too restricted ($e \geq 500$), increasing *u* and *l* resulted in better performances on the 2-bit task. This also allowed the BAM module to exhibit faster and steadier learning, i.e., shorter learning times and perfect performances throughout the 100 trials.

Of course, if insufficient learning times ($e = 10$) were given, the network was not able to perform the task regardless of the architecture. If more time was allowed ($e \geq 100$), then better performances were observed. In those cases, it seems that the initial layer (UL1) was finally stabilizing, as shown in Figure 44a), which was independent of the *u* constraint. Additionally, results showed that in short learning times ($e = 100$), having a single layer was more advantageous than having multiple ones. Again, stabilization of the UL1 was essential if any subsequent layers were to provide an adequate representation to the BAM module. Suppose enough time is allowed ($e = 250$), then deeper layers ($2 \leq l \leq 5$) can fully deploy their power, as shown by the better performances of the BAM module. This suggests that if the weights of the deeper layers have

enough time to distance themselves from their initial value, performance in the BAM module will increase. This also means that generating a viable solution does not require the MF layers to stabilize fully. Still, to achieve perfect learning of the input-target associations for every trial, enough time should be given and enough layers, as increasing *l* seemed to raise the likelihood of generating sets of linearly separable patterns. However, increasing only the depth is not always enough. This is especially true for fewer units per layer ($u \leq 15$). Therefore, the role of *u* may be to create a big enough feature space to generate a steady solution if the number of layers is high enough. In other words, both *u* and *l* must be considered together to guarantee perfect performance all the time. An interesting example of this is highlighted when $u = 25$ and $l = 1$, as it resulted in 100% performance.

A parallel trend was echoed in the density plots (Figure 44c)), as deeper and wider architectures seemed to be linked with better performances and faster learning times. These results emphasized that the architectural parameters of the MF were indeed changing the "nature" of the generated patterns. Bimodal distribution resulted in faster learning times for the BAM module as well. However, this type of distribution did not fully guarantee success in solving the task. This was seen in the $u = 5$ and $e = 2500$ constraints, as despite being bimodal for all $l > 1$, no consistent perfect performances were observed. Further investigations are needed to understand better how these patterns change over time and why these shapes impact the transformation from nonlinearity to linearity required for the BAM.

If learning a task by the BAM must be done quickly, then more time should be allocated to a MF with more layers and units. Although learning in the MF for these conditions was longer, this was compensated by fast learning times in the BAM module (as seen in Figure 44b)). Thus, increasing both *u* and *l* parameters was very beneficial. However, suppose the total number of units

must be considered. In that case, results from section 6 suggest that 100 units should be organized throughout more layers with fewer units than having all of them in one layer.

To sum up, both *u* and *l* in the MF contribute towards generating a linearly separable representation. Which parameter is the most important? It is hard to give a definitive answer. It will depend on the restriction applied; fewer layer with fewer units outperforms its counterpart (when the MF is restricted to 100 epochs and BAM's learning time is disregarded). Still, in other paradigms, the opposite becomes true (when total learning time is to be minimized and performances maximized). However, results have shown that increasing *u* and *l* do not impair performance. Thus, if resources are limitless and the best performances are to be sought, having a deeper and larger model will only require a bit more learning time for the MF component as a counterweight. This situation may be closer to cognition, as in the brain, billions of neurons organized in parallel systems are used, and in most cases, learning continues throughout its life span.

So far, it has been shown in the previous sections that different nonlinear behaviors can be obtained by modifying the architecture of the MF-BAM. However, it remains to be investigated if the MF can enable continuous online learning without needing a reset in weights values between tasks. Moreover, will the generated representation be self-adaptable if novel inputs for a given task are used? Thus, in the next section, the MF-BAM performance was studied under the condition of learning multiple tasks one after the other.

7. Continuous Online Learning

Humans are often said to be flexible, able to shift from learning one task to another seemingly effortlessly. However, this is not always the case for ANNs, as they can stay stuck to a previous solution when learning a different task. This refers to sequentially learning various tasks

by presenting the input-target pairs incrementally and without resetting the model's weight values between tasks. Thus, it is a means to show some plasticity during learning. To evaluate this, a sequence of tasks, namely the 2-bit, Moons, and Eclipse, were presented in succession to the MF-BAM, and its behaviors (decision boundaries) were recorded over time.

7. 1. Methodology

The 2-bit, Moons, and Eclipse tasks were created as described in section 4. Each task was sequentially presented to the MF-BAM in the following order: 2-bit, Moons, and Eclipse. The architecture parameters were set to $u = 500$ and $l = 4$, following results in section 4, where this architecture was shown to solve all three tasks successfully. The model had 500 epochs to learn a task before introducing the next one. Likewise, this number was based on results from section 6, which showed that this number of epochs was enough. Decision boundaries were produced after each epoch to assess the model's behavior across time. The learning and recall procedure from section 3 was followed. For consistency, 100 trials were conducted, and the average learning error and decision zones were reported.

7. 2. Results

Average learning errors (MSE) across time (e) after the introduction of all three tasks for both the MF and BAM are shown in Figure 45a). Results indicated that in the BAM, there was an increase in error (MSE values ≥ 1) followed by a gradual decrease towards 0 after a new task was introduced during learning. In each task, error achieved low values (MSE ≤ 0.0001) within the 500 epochs the task was presented. Likewise, an increase in error was observed for all ULs of the MF once each task was presented. However, the extent to which error increased was smaller with the introduction of each successive task.

In Figure 45b), decision boundaries in function of the BAM's learning (*e*) are shown. Notably, decision boundaries were seen to transition from one task to another successfully, and 100% recall performance was achieved. This was gradually done within the time window of each task being presented. As the 2-bit task was introduced *(e* = 1), the decision boundary first exhibited random values until forming the correct zones. Once the second task was introduced *(e* = 501), this shape moved toward the appropriate general structure for the Moons. Similarly, after the Eclipse task was introduced (*e* = 1001), the decision boundary slowly transitioned toward the correct general solution. In each case, the final decision boundaries were like the previous results in section 4 (Figures 37b) and 42). Furthermore, a proper decision boundary was usually achieved within the first 400 epochs of a presented task.

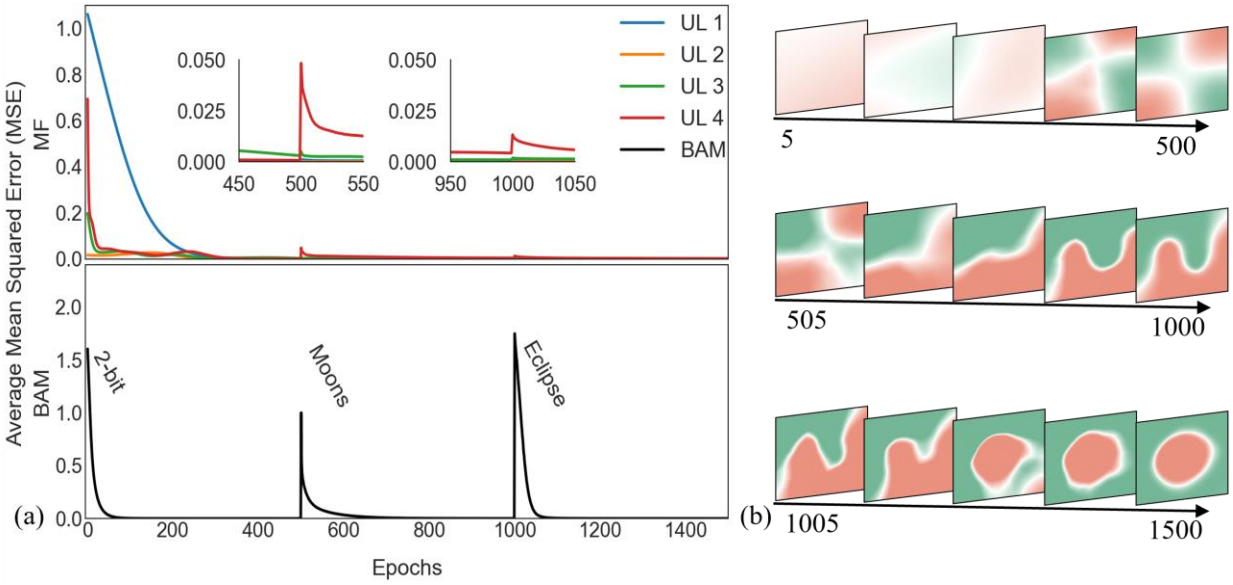

Figure 45. a) Average mean squared error for the continuous learning task. b) Decision boundaries of the continuous learning task.

7. 3. Continuous Online Learning Discussion

In summary, results have shown that the MF-BAM can sequentially learn multiple nonlinear tasks. This was expected since an essential feature of associative memories is that they

are plastic (i.e., updating to new data; [Zilly et al., 2021]). This usually results in the model "forgetting" old tasks since weights stabilize to the new solution, to the detriment of performance on the previous task. This was seen in the BAM module, as learning errors always spiked over 1 MSE when learning a new task. As such, the BAM would steadily adapt to the new task and forget old ones. This behavior was also shown in the decision boundaries, seen in Figure 45b).

However, this was different for the MF module. After introducing a new task to the MF, learning errors across unsupervised network layers would increase to a lesser extent. This might indicate that generated hidden representations were not drastically modified when learning a new task. In other words, some features generated from task A could be maintained when learning task B. In turn, the MF could be interpreted as accumulating knowledge from new tasks. In other words, random stimuli would be sufficient for the MF to learn to organize itself and help distinguish the totality of the input space. This may be akin to how the early stages of the sensory system mature as the brain is exposed to new stimuli [Ackman & Crair, 2014; McMahon et al., 2012]. As such, continuously training the MF could eventually lead it to generate patterns that can be reused in the BAM to relearn previous tasks or even future ones successfully.

Although the model has been shown as a promising approach to dealing with nonlinear associations, all previous tasks contained only two classes. In many instances, a learner will be faced with a multiclass problem. Therefore, the following section investigated whether the MF-BAM could learn nonlinear tasks while maintaining previously observed behaviors in a multiclass setting.

8. Multiclass Nonlinear Task

This section assessed the MF-BAM using the Spiral task, a 2D continuous valued nonlinear task consisting of 3 classes. When faced with an *n*-class problem, multiple methods can reduce a

multiclass task into different binary-class problems to facilitate learning [Chaitra & Kumar, 2018]. However, the goal for this section was to avoid these methods and see if the network could learn a nonlinear multiclass task without any preprocessing and through simple input-target associations. Using the Spiral task was the next logical step, as results from a 3-class setting can be extended to tasks with even more classes. Lastly, the task was kept in a 2D setting to allow the visualization of the network's behavior through its decision boundaries.

8. 1. Methodology

The 3-class Spiral task was created similarly to previous 2D tasks, where 2-dimensional input vectors ranging with values from -1 to 1 were generated. The target vectors were changed from 1-dimensional to 3-dimensional ones to ensure that behavior would also scale to higher-dimensional target patterns. As such, the first class was represented by the target vector [1, -1, -1], the second by [-1, 1, -1], and the third by [-1, -1, 1]. Each class contained 40 inputs, giving 120 data points for the entire dataset. Figure 46a) shows the visual representation of the 3-class Spiral task. Learning and recall followed the procedure in section 3. However, predictions for decision boundaries were determined as follows: if the output was the correct class, it was considered a success. Otherwise, it was considered a failure. The simulation was repeated 100 times, and average decision zones were reported.

For this simulation, the architecture was set to u = 500 and l =10, following results from sections 4 and 5. The number of epochs (e) was set to 25000 for the MF for investigating stability purposes and at e = 2500 for the BAM.

8. 2. Results

Figure 46b) shows the average learning error for both the MF and BAM components. In both cases, learning followed the same trend from previous results: Higher MSE that decreased

over time. In addition, it also showed how from 0 to 50 epochs in the MF module, slight increases of MSE were gradually seen in each UL after previous layers stabilized. Moreover, as previously observed, the deeper the UL, the higher the learning error spiked for that layer. Despite these peaks, the MSE from all layers steadily decreased toward 0. For the BAM component, MSE values decreased rapidly within 247 epochs.

The model's average classification behavior can be seen in Figure 46c). In all 100 trials, the MF-BAM could learn the Spiral task perfectly. Decision zones confirm that the classification behaviors of the MF-BAM were stable. Additionally, decision zones showed how each class was well separated by an uncertain area where no correct classification could be observed.

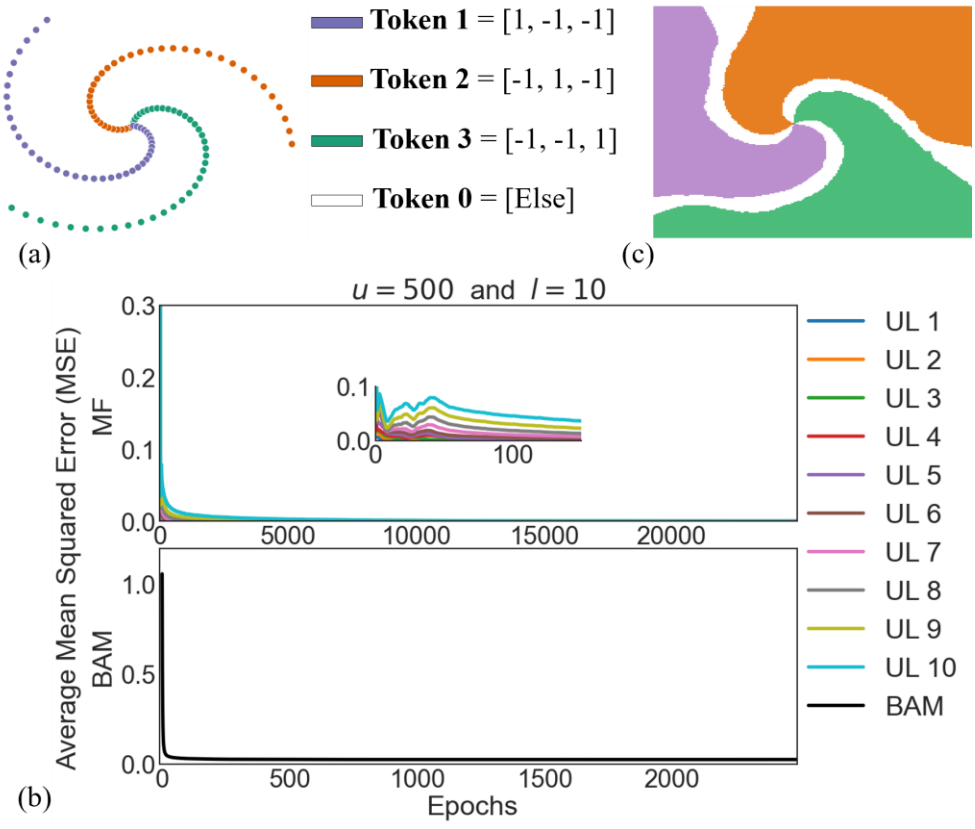

Figure 46. a) 3-class Spiral task. b) Average mean squared error for the MF and BAM components on the 3-class Spiral task. c) Decision boundary for the Spiral task.

8. 3. Multiclass Nonlinear Task Discussion

This simulation evaluated whether the MF-BAM could deal with more than two classes in a nonlinear task. Results showed that the model could classify the three classes perfectly. Going from 2-class to 3-class did not have an impact on the general behaviors of the model. Furthermore, learning in the MF and BAM stabilized to a consistent solution. These findings suggest that the general trends found from previous sections also apply to an *n*-class setting.

9. General Discussion

We proposed the Multi-Feature - Bidirectional Associative Memory (MF-BAM), a RNAM composed of multiple unsupervised network layers and a single supervised network layer. This model could solve various bipolar and continuous values nonlinear classification tasks within the neurodynamic perspective.

Of importance, each unsupervised network layer interacts with only local information without backpropagation of the error. In addition, the same transmission and HL were used throughout the network. Thus, it maintained the same internal dynamics for all its layers. Changes in behavior only came from manipulating the architectural parameters of the MF. This significantly increases biological and cognitive plausibility [O'Reilly, 1998]. Furthermore, the MF-BAM did not rely on preprocessing nor required external intervention (e.g., early stopping). Table 8 shows a summary of comparisons between the modified BAM [Chartier & Boukadoum, 2006], the bidirectional backpropagation BAM [Adigun & Kosko, 2019; Kosko, 2021], the combined Multi-Layered Perceptron-BAM [Wu et al., 2018], and the MF-BAM.

Table 8

Different properties from cognitively inspired models.

	BAM	Backprop-BAM	MLP-BAM	MF-BAM

Biological realism	Yes	No	No	Yes
Distributed representation	Yes	Yes	Yes	Yes
Error-driven	Yes	Yes	Yes	Yes
Hebbian learning	Yes	No	No	Yes
Activation Spreading	Yes	Yes	Yes	Yes
Bidirectionality	Yes	Yes	No	Yes
Learns Nonlinear tasks	No	Yes	Yes	Yes

This paper's main takeaway is that the MF-BAM can effectively learn a wide range of nonlinear associations. This can be explained by revisiting the MF-BAM's two modules: the Multi-Feature (MF), composed of unsupervised network layers (ULs), and the BAM, which is a supervised network layer (SL). Given that the latter remained untouched from previous research, success on nonlinear tasks depended solely on the MF. Consequently, for the BAM to learn a given nonlinear task, the MF had to generate representations from the inputs that shared a linear relationship with the corresponding targets. Results showed that this was achieved steadily, as the model showed 100% performance across all nonlinear tasks if enough units per layer (u) or unsupervised network layers (l) were used.

In addition to these findings, the study revealed several other interesting properties of the MF-BAM model. Most notably, the shift from a linear to a nonlinear decision boundary was not an all-or-nothing process. Instead, it was a smooth transition controlled by changes in architecture. Given enough layers, increasing the number of units would gradually develop a stable solution (Section 4). Adding layers can make the model highly specific (overfitted) to a task. In other words,

it can learn exceptions to rules. Given the dynamic taking place during learning, where layers incrementally stabilize one at a time, the generated patterns from the first few layers can be used quickly. This means it would be possible to sacrifice performance to tackle problems needing faster response times by using shallower layers. For better performances, increasing learning times and using deeper layers would suffice. This tailored answer goes by the speed-accuracy trade-off literature in ANNs [Huang et al., 2017; Li, Zhou, Pan & Feng, 2019]. There is also a possibility of using the output from each hidden layer rather than only the final one. Each layer could connect to a single BAM to obtain diverse behaviors (linear, nonlinear, specific) from the same inputs. These structural and dynamic properties found in the MF-BAM align with what is observed in humans. For example, individuals can give different answers when encountering the same stimulus, which can change as more exposure time is provided or context [Hesselmann et al., 2008; Standage et al., 2014].

Section 6 presented a comprehensive analysis of the impact of both u and l parameters which yielded significant findings. Specifically, deeper configurations were found to have a clear advantage over having more units onto a single layer. The distribution of values in the generated patterns was also identified as essential, with bimodal distributions proving more advantageous than normal ones. Furthermore, the generated patterns from the deeper and larger MF architectures not only shared a linear relationship with their corresponding targets but were also much more distinct. Each UL (FEBAM) may be doing a feature extraction process akin to something like principal component analysis (**PCA**) by generating a pattern that has low-level statistical features that represent the input pattern's intrinsic information [Chartier et al., 2007]. Repeating this process over multiple layers would mean that each deeper generated pattern becomes a more salient representation of the distinguishing features of the input pattern. In other words, it

decorrelates the input patterns [Rolon-Mérette et al., 2018a]. This could explain why the BAM could learn quickly after the stabilization of the MF. These findings may provide valuable insights into why the brain's neural networks are recursively organized and why they are thought to function in an associative manner [Buckner & DiNicola, 2019; Wang & Cui, 2018].

Furthermore, it was shown that the generated patterns in the MF would remain relatively consistent even with the addition of new inputs. This was indicated by results from section 7, where the increase in MSE in the MF was smaller and smaller as new tasks and more inputs were introduced. This suggests that some information was being conserved and that the model was not entirely subjected to catastrophic forgetting [Kemker et al., 2018]. This hints towards an advantage of presenting more inputs to the MF. Consequently, it is imaginable to train an MF on the entire input space through random exposure. Although this process would require more time, once trained, the MF could be reused in an online setting without catastrophic forgetting if the dimensionality of the input does not increase. Furthermore, results from section 7 suggested that as the MF learns, the computational cost (number of epochs) of learning a new task diminishes. It would be interesting to verify this property's extent, as it is crucial in modeling how humans learn online, as information is constantly being captured, processed, and stored throughout their lifespan.

Despite these interesting properties, some limitations must be addressed in future studies. This research maintained an equal number of units in each layer of the MF. This was done to isolate the effects of having more layers or units. Since each layer can have an independent number of units, further research could explore utilizing various layer sizes and how this impacts behavior. Previous studies have shown that in a single FEBAM, it is possible to adapt the number of units to generate diverse behaviors, such as creating prototypes (compression) or exemplars (expansion) [Giguère et al., 2007a, 2007b; Rolon-Mérette et al., 2019). As such, it would be possible to use the

bidirectional nature of the model to generate prototypical categories based on small u values and which layer depth information is compressed on. It may also be possible to incrementally add units and layers to a model without retraining the entire model, similar to other models such as the adaptive resonance theory (**ART**) model [Carpenter & Grossberg, 1988]. This would be highly advantageous from an engineering perspective since it could automate finding an optimal configuration for a given task. However, this may be less crucial when modeling human cognition, as the architecture is typically considered fully developed, much like an adult brain, and does not change significantly depending on the task. Similarly, it was shown throughout the paper that even if the number of units or layers is greater than necessary, it does not hinder performance. This suggests that setting the architecture is not a trial-and-error process, and using more units and layers can be the default approach.

Finally, the bulk of this paper focused on classical nonlinear tasks using two-dimensional input vectors. This was necessary to examine and visualize the model's task performance. However, the BAM is not optimal for two-dimensional input and may have slower learning times. In addition, higher dimensional input is more relevant to human cognition. Results from the N-bit, where $N \geq 3$, have shown that the MF-BAM can scale up to learn tasks in higher dimensions. Therefore, future studies could investigate if changing the representation of a nonlinear task to a higher dimension can impact behavior. Similarly, it was shown that the MF-BAM could deal with a nonlinear task containing three classes. Still, future research should examine if there is a relation between the dimensionality of the input and the generated patterns on the number of learned classes.

10. Conclusion

This research has shown that the MF-BAM can successfully learn nonlinear tasks when given sufficient units and layers. The representations generated from the last unsupervised network layer of the MF component are linearly separable, ensuring the BAM's success. The model achieves this without adding external factors and relying on architectural changes alone. The internal mechanisms remain homogeneous, utilizing the same local learning and transmission function while maintaining the general bidirectional structure. This work highlights that a RNAM is all you need to account for complex behaviors like nonlinear tasks.

11. Acknowledgement

This research was supported by the Fond de Recherche du Quebec Nature et Technologies (FRQNT) and the Ontario graduate scholarship (OGS) and was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) (210663).

Appendix A

Table A.1.

Summary of the acronyms used.

Acronyms	Descriptions
α-β BAM	alpha-beta Bidirectional Associative Memory
AM	Associative Memory
ANN	Artificial Neural Network
ART	Adaptive Resonance Theory model
BAM	Bidirectional Associative Memory
BSB	Brain-State-in-Box
CCBAM	Chaotic Complex-valued Bidirectional Associative Memory
FEBAM	Feature Extraction Bidirectional Associative Memory
LieBAM	Lie algebra-valued BAM
MBAM	Morphological Bidirectional Associative Memory
MF	Multi-Feature extracting bidirectional associative memory
MF-BAM	Multi-Feature - Bidirectional Associative Memory
ML	Machine Learning
MSE	Mean Squared Error
NAM	Neural Associative Memory
PCA	Principal Component Analysis
RNAM	Recurrent Neural Associative Memory
SL	Supervised network Layer

Std	Standard deviation
UL	Unsupervised network Layer

Table A.2.

Summary of the notations used.

Notations	**Descriptions**
p	Input pattern
$\mathbf{h}_1$, $\mathbf{h}_2$, ..., $\mathbf{h}_l$	Hidden representations generated by each UL
o	Output target
x	Input sublayer
y	Output sublayer
W	Weight matrix connecting **x** to **y**
V	Weight matrix connecting **y** to **x**
m, *n*	Number of units in **x** and **y**, respectfully
u	Architectural parameter: Number of units per UL in the MF
l	Architectural parameter: Number of unsupervised network layers in the MF
i	Index unit
c	Iteration cycle index
δ	General transmission parameter
a, **b**	Activations of **x** and **y**, respectfully
k	Learning trial index

η	Learning parameter
$\mathbf{x}_{UL_l}$	Input sublayer of a given UL_l
$\mathbf{y}_{UL_l}$	Output sublayer of a given UL_l
$\mathbf{W}_{UL_l}$	Weight matrix **W** of a given UL_l
$\mathbf{V}_{UL_l}$	Weight matrix **V** of a given UL_l
e	Epoch
$\mathbf{x}_{SL}$	Input sublayer of the SL
$\mathbf{y}_{SL}$	Output sublayer of the SL
$\mathbf{W}_{SL}$	Weight matrix **W** of the SL
$\mathbf{V}_{SL}$	Weight matrix **V** of the SL
P	A list containing all input patterns **p**
HO	A list containing all pairs of generated patterns from the final UL of the MF ($\mathbf{h}_l$) and their corresponding targets (**o)**.

Chapter 2 Conclusion

The ability to learn nonlinear relationships in the BAM framework was investigated in this chapter. By following a multilayer and multi-network architecture, it was possible to solve these complex tasks in a consistent fashion. By only manipulating the architecture of the MF, it was possible to change the degree of linearity of the decision zones of the task. Furthermore, the ability to solve these nonlinear tasks was not detrimental to the overall behavior of the model. In fact, by joining highly similar BAMs, additional properties emerged, opening the door to investigate further the added cognitive properties emanating from this model. Interestingly, contrary to previous efforts requiring specific adjustment (e.g., different transmission functions per layer, algebra types, etc.), access to global information or complex learning algorithms, success here came only from simple repetition. These findings provide more evidence for the strength of the recursive structure and may also contribute towards understanding their functionality in the human brain. Chapter 1 showed that tasks may become nonlinear when complexity is increased. Chapter 2 introduced a more efficient and consistent model for such instances. With the MF-BAM and the contextual information approach from Chapter 1, it is now possible to investigate how associative mechanisms can aid in learning and solving complex cognitive tasks. Additionally, the joint approach will permit the speculation of the extent of AM mechanisms in relation to human cognition.

Chapter 3

Chapter 3 introduces the joint benefits of using the contextual information approach and the MF-BAM. The chapter divides itself in two. In the first part, the paper titled *Using Bidirectional Associative Memory Neural Networks to Solve the N-bit Task* lays the groundwork for using the previously shown associative mechanism to learn behaviors akin to employing rules. The task used to illustrate these additional properties was the N-bit. The goal was to show the advantages of departing from the traditional rote associations learning strategy and get closer to what can be observed in humans during similar problems. Thus, different associations related to counting and parity were taught. In the second part, the publication titled *Are Associations All You Need to Solve the Dimension Change Card Sort and N-bit Parity Task* further investigated this by incorporating more rule-like behavior. The tasks used were the N-bit and the dimension change card sort (DCCS) to illustrate how the model could use these rule-like behaviors and switch between them based on the task at hand.

Using Bidirectional Associative Memory Neural Networks to Solve the N-bit Task

Publication reference:

Rolon-Mérette, D., Rolon-Mérette, T., & Chartier, S. (2023, May). Using Bidirectional Associative Memory Neural Networks to Solve the N-bit Task. In *The International FLAIRS Conference Proceedings* (Vol. 36).

Abstract

Nowadays, artificial neural networks can easily solve the *N*-bit parity problem. However, each time a different level must be learned, the network must be retrained. This, combined with the exponential increase of learning trials required as *N* grows, makes these models too different from how their biological counterpart solves them. This is because humans learn to recognize patterns, count, and determine if numbers are odd or even. Once they have learned these tasks, they can have them interact to solve any level without further training. This behavior is akin to performing multiple associations of different tasks. Therefore, it is proposed that by using bidirectional associative memory neural networks, it would be possible to solve the *N*-bit parity problem in a similar fashion to humans. To achieve this, two networks interacted; one served as a task Identifier and the other as a memory Extractor, giving the desired behavior influenced by the Identifier. Results showed that the model could solve the 2- to 9-bit in linear time once the associations were learned. Moreover, this was possible with 97% fewer inputs and no retraining. In addition, because of the recurrent nature of the model, it could also solve the tasks even under high noise levels.

Keywords— Associative Memory, Interacting associations, ANN, BAM, N-bit.

Introduction

In the last decade, artificial neural networks (ANNs) have grown in popularity, largely due to their capability to tackle complex tasks [Cichy & Kaiser, 2019; Silver et al., 2016]. That being said, when drawing the parallel between ANNs and human cognition [Anderson et al., 1977; Martindale, 1991; Wang & Cui, 2018], there are concerns about how they achieve this [Marcus, 2018]. This becomes evident when revisiting the classic benchmark, the XOR [Marvin & Seymour, 1969], and its generalization, the *N*-bit parity task.

For an ANN, these input-target pairing tasks require learning to associate different input vectors of binary values (1 or 0) of length *N*, to their corresponding targets (1 or 0), through rote associations. Nowadays, solving the *N*-bit parity task has become trivial, as many feedforward neural networks can do it with ease [Dhar et al., 2010; Yanling, Bimin & Zhanrong, 2002; Yang, Yang & Wu, 2011]. However, most models must be retrained from scratch each time they have to solve a different *N*-bit, which hinders their generalizability. Also, the number of inputs a network must learn increases exponentially (2^N) with the task's level, negatively impacting learning times [Tesauro & Janssens, 1988]. Nevertheless, how ANNs are trained differs from how humans learn to solve these problems [Zador, 2019].

Evidence suggests that once a human has learned to recognize numbers, count, and determine what values are odd and even, it is sufficient to solve any *N*-bit task [Berch et al., 1999; Graham, 1999; Soto-Calvo et al., 2015]. In other words, they do not memorize each input-target pair but rather have distinct modules with different learned tasks interacting to solve the problem [Barrett & Kurzban, 2006; Chandra et al., 2018; Fang et al., 2004]. This may be one of the reasons why humans are much faster and better at generalizing than ANNs.

Interestingly, this process in humans can be seen as similar to performing multiple

associations of different tasks. Thus, associative mechanisms could play a key role in solving these types of problems [Anderson & Bower, 2014; Buckner & DiNicola, 2019; Dacey, 2016, 2020; Lansner, 2009; Mandelbaum, 2015]. As such, an interesting avenue would be to use recurrent neural associative memories (RNAM) [Anderson, 1983], a type of ANN that explains how learned patterns may correspond to attractors in the brain's neuronal state space. Consequently, RNAMs find themselves in various cognitive theories [Knoblauch, 2005]. Of interest, due to its ability to learn auto- and hetero-association, the bidirectional associative memory (BAM) subtype would be used to investigate if the *N*-bit can be solved using human strategies instead of pure rote associations.

Remarkably, BAMs were introduced in the 80s [Kosko, 1988] but have remained one of the most widely used types of RNAM to study associative memory. Over the years, various versions have been proposed to overcome its original limitations (see [Acevedo-Mosqueda, Yanez-Marquez & Acevedo-Mosqueda, 2013] for a review). But recently, the Multi-Feature - Bidirectional Associative Memory (MF-BAM) has been proposed as a cognitively inspired model with interesting properties [Rolon-Mérette, Rolon-Mérette & Chartier, 2023a]. This multi-network can learn any type of association while remaining plausible [O'Reilly, 1998]. It combines unsupervised and supervised learning and is based on simple local Hebbian and anti-Hebbian learning. Thus, it is an interesting candidate for studying human cognition.

However, when learning different tasks, inputs may belong to many different sets, leading to OMA [Elman, 1990; Jordan, 1997]. One remedy is using contextual information [Rolon-Mérette, Rolon-Mérette & Chartier, 2019]. Context permits a unified representation of the inputs and makes learning multiple tasks possible [Church, Ross & Chartier, 2020].

Thus, two MF-BAMs are needed. An initial MF-BAM module will recognize incoming

stimuli and generate appropriate instructions (contextual information). These would then be given to a second MF-BAM module to generate the corresponding desired behavior (tasks). In other words, by combining two MF-BAMs, it could be possible to learn all prior associations necessary to solve any *N*-bit parity problem faster and in a more general way.

The remainder of the paper divides itself as follows: Section I: Model, Section II: Methodology, Section III: Results, and Section IV: Discussion.

Section I. Model

Learning to solve the *N*-bit parity requires the interaction between two modules (MF-BAMs). The first module, the Identifier, will generate the appropriate instruction set (contextual information) from the incoming stimuli (inputs). The second module, called the Extractor, will retrieve the corresponding stored behaviors from the concatenation of the inputs and the contexts. Figure 47a) shows the overall implementation of the proposed model. In short, an input pattern, $\mathbf{p}_{[c]}$ at cycle c, is sent to the Identifier, which will recall the according context $\mathbf{c}_{[c]}$ (instruction). This pattern will then be concatenated with a portion ($\mathbf{e}_{[c\text{-}1]}$) of the outputted behavior $\mathbf{b}_{[c\text{-}1]}$ from the Extractor from the previous time step (c - 1). This concatenated pattern $\mathbf{o}_{[c]} = (\mathbf{e}_{[c\text{-}1]} \circ \mathbf{c}_{[c]})$ is sent to the second module to generate the desired behavior. The Extractor's output is composed of two concatenate parts $\mathbf{b}_{[c]} = (\mathbf{z}_{[c]} \circ \mathbf{e}_{[c]})$. The output $\mathbf{e}_{[t]}$ will keep track of what number the model is currently counting, while $\mathbf{z}_{[c]}$ will give the final behavior (odd or even).

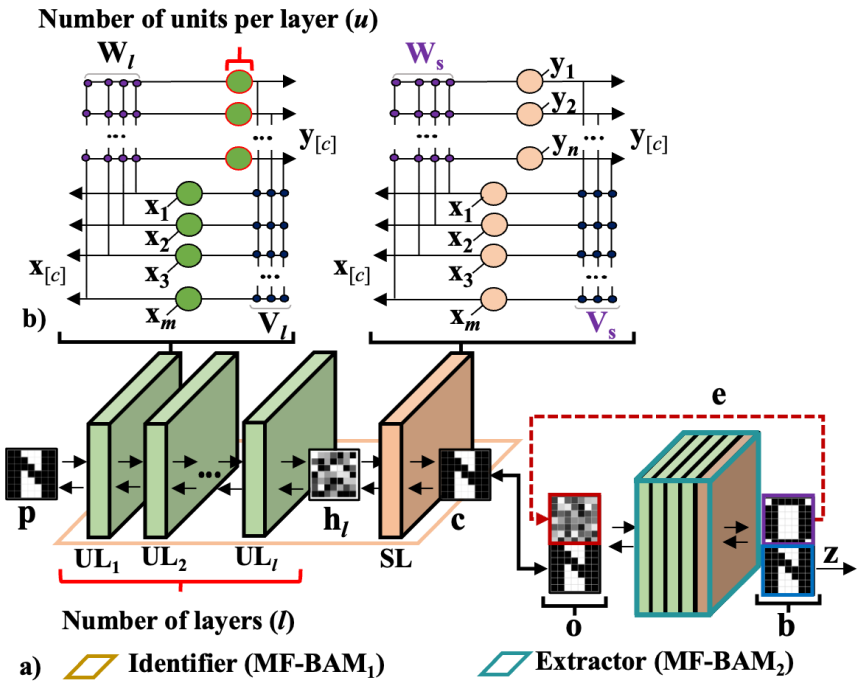

Figure 47. a) The two MF-BAM composed of Unsupervised Layers (UL, green) and a single Supervised Layer (SL, orange), where l represents the number of UL and u the number of hidden units. b) The detailed underlying network that composes each layer.

Architecture

Each MF-BAM has two components, the MF, which includes several unsupervised networks (ULs), and the BAM, which contains a single supervised network (SL). These are all based on the unsupervised and supervised version of a modified BAM [Chartier & Boukadoum, 2006; Chartier et al., 2007]. Figure 47b) shows the architecture of the UL (green) and SL (orange). The MF-BAM uses the same transmission function and learning rules for all its layers. The model's behavior changes only by modifying the MF's number of y-units in each hidden layer (denoted by *u*) and the number of hidden layers (represented by *l*) used. For all units, the same cubic transmission function was used, and for each connection, the Hebbian/anti-Hebbian learning rule was used. For details see [Rolon-Mérette, Rolon-Mérette & Chartier, 2023a].

Section II: Methodology

Instead of learning all input-target pairs (2^N), the model will first learn to perform four different tasks selected to be sufficient to solve any parity. These tasks are:

- To recognize 1 and 0.
- To count from 0 to 9.
- To know what numbers are odd and even.
- To know when the task starts and ends.

Multiple associations will represent each task. All the simulations were performed using Python programming language installed from Anaconda [Anaconda Software Distribution, 2020; Rolon-Mérette et al., 2020] and were run on a RTX3090 graphics card.

Stimuli

For all tasks, 7x7 bipolar (values of -1 or 1) pixelated images representing alphanumerical patterns were used for the *N*-bit task and the contextual tags. These patterns represented the numbers from 0 to 9, the letters N, C, D, P, E, and O for the context *N*-bit task, Count, Fixed, and Parity, and for the targets Even and Odd, respectively. A given *N*-bit task was portrayed as a time series starting with N then, followed by the desired numbers of 0 and 1, and ending with the pattern P. Figure 48 shows some examples of different *N*-bit.

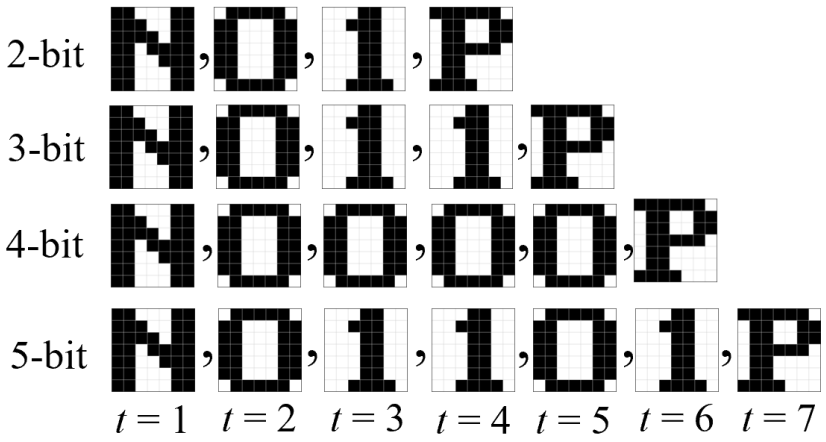

Figure 48. Example of time series representing the input sequence during recall for 2- to 5-bit.

Input-Target Pairing

Before testing the model, each MF-BAM had to learn specific input-target pairs depicted in Figure 49. The Identifier module had to learn a total of 4 associations directly linked to either recognizing 1 or 0 and generating the corresponding instruction (count or fixed) or recognizing to start the *N*-bit task or end it by giving the parity of the value in memory. The Extractor module had to learn 29 associations, which represented the possible actions for the numbers between 0 and 9 in the different contexts of either initializing the task, staying fixed, counting, and giving parity. Thus, a total of 33 associations were needed to solve any *N*-bit task up to 9. Also, Figure 49 shows that for task initialization, the upper portion of the input pattern is randomly generated, ~*U*(-1,1). This was to simulate chatter noise (i.e., rest behavior).

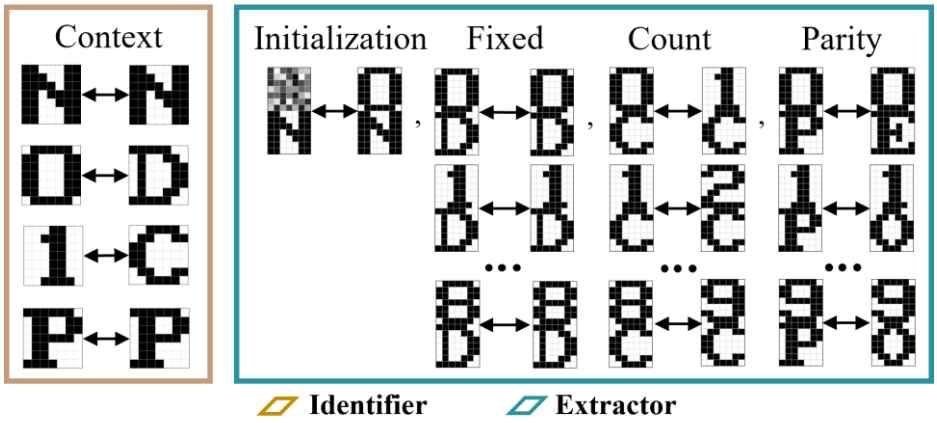

Figure 49. Input-target pairs for each module.

Model's Flow of Information

Once the learning is accomplished, the model can be used to solve the *N*-bit parity task. The first MF-BAM receives the input stream and determines the instructions that should be sent to the second MF-BAM. As the inputs are processed, the network should be able to output the correct behavior. Figure 50 shows this process for the 2-bit.

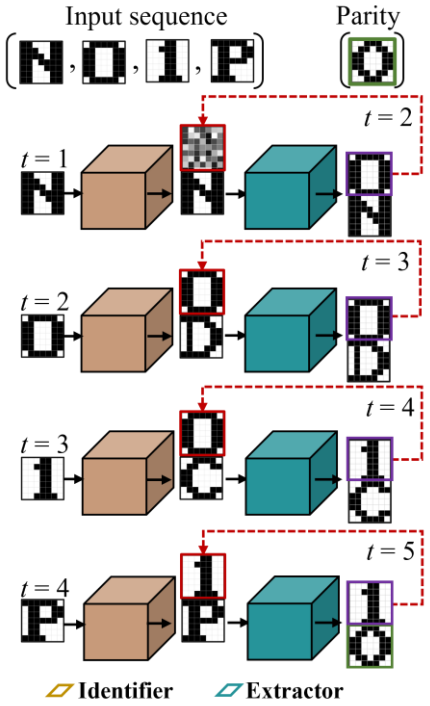

Figure 50. Flow chart of the model when presented with the 2-bit.

Simulations Protocols

The model was tested under different conditions to better understand its capabilities.

Simulation I: The N-bit and Interacting Association. The model had to learn all the input-target pairings shown in Figure 49 within 2500 epochs. This value was based on the past implementation of the MF-BAM when learning the *N*-bit [Rolon-Mérette, Rolon-Mérette & Chartier, 2023a]. During recall, the entire 2- to 9-bit parity inputs were presented. Therefore, a total of 1020 ($= \sum_{i=2}^{9} 2^i$) time series were generated and presented one at a time. For reliability purposes, learning and recall were repeated 50 times. Thus, average mean squared error (MSE), performances, and 95% confidence intervals were reported.

Simulation II: Rote vs. Interacting Association. To better contrast the differences with traditional ANNs and their use of rote association, the current model was evaluated in parallel to the vanilla Multi-Layer Perceptron (**MLP**). Each model was presented the 2- to 9-bit in their respectful format and under fair conditions.

This meant that for the MLP, the task required learning and recalling a total of 2^N inputs. As usual, to accommodate the level increase, the number of units in the input layer of the MLP varied in function of *N* [Chen, Tseng & Wu, 2014]. Thus, for each level increase, the architecture had to be adjusted. The MLP was implemented in Scikit-learn [Pedregosa et al., 2011] with the parameters kept at their default values with the exception of the *batch_size*, which was set to 2^N to mimic being presented one stimulus at a time, the *solver*, which was set to "sgd" (stochastic gradient descent) to follow the error rate per epoch, and *learning_rate_init*, which was set to $1/(2^N)$ to minimize the number of learning epochs.

For the proposed model, the task format required learning only the interacting associations needed for a specific bit value and was evaluated during recall on that size. This restriction ensured

a high-level comparison, as the MLP can only learn one level at a time. Thus, for each increase, the model was retrained on the selected associations. For example, for the 2-bit, the MF-BAMs were only taught the context associations and the ones needed to count to 2 in Figure 49. This meant that the model only had 12 associations in total to learn. However, during recall, it was presented all four times series representing the 2-bit parity task. Thus, for each bit size increase, three additional associations were needed during learning. In other words, for the 3-, 4-, 5-bit, and so on, 15, 18, 21, etc., associations were used, respectively.

To show the differences between approaches, learning was stopped after each MF-BAM or MLP reached a MSE < 0.005 for the *N*-bit task in question (indicating that the task was correctly learned). As such, the total number of epochs for each network was reported. To assess reliability, learning and recall were repeated 50 times for both models, and 95% confidence intervals were reported.

Simulation III: Noise Filtering. The model's capability to still filter noise like any other BAM was evaluated on the 2-bit time series through pixel flip noise, which was introduced to all patterns of the time series. This bit level was chosen for simplicity, as increasing *N* only lengthens the time series and does not change the nature of the task. An example of different degrees of noise can be seen in Figure 51. The number of pixel flips varied from 0 to 35, giving a proportion of noise from 0 to 71 % (35/49). To ensure consistency, the model was trained on all input-target pairs shown in Figure 49. For reliability purposes, 50 learning trials were conducted of 2500 epochs. However, 100 testing trials were performed for each learning run due to the random selection of the pixel to be flipped. As such, average performances and 95% confidence intervals were reported.

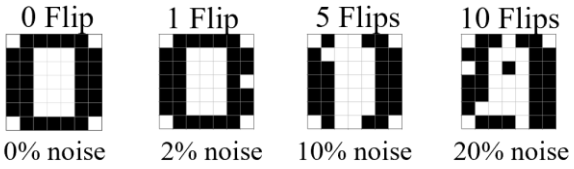

Figure 51. Example of different levels of pixel flip.

Learning Parameters

For all simulations, each MF-BAM's learning was conducted in the same fashion as in [Rolon-Mérette, Rolon-Mérette & Chartier, 2023a]. The transmission function parameter (δ) was set to 0.2, weights were randomly initialized $\sim U(-0.1,0.1)$, and the number of units (u) per layer was set to 100, which is about the size of the input patterns of the Extractor. To assess the complexity of the associations, the number of layers (l) ranged from 1 to 4 for simulation I and III and was kept constant at $l = 4$ for simulation II.

Recall Process

Once learning was completed, the network was then tested by presenting the time series given by the simulation condition (for an example of the *N*-bit as a time series, see Figure 48). As previously mentioned, the model's final output (parity) is given once the times series ends with the P pattern. Performance was calculated in the same fashion for each task, where the generated parity was compared to the original target on a pixel-per-pix basis. If the generated and the target patterns were identical, then it was recorded as a success.

Section III: Results

Simulation I: The *N*-bit and Interacting Association

The model learned all interacting associations and was tested on the entirety of the times series representing the 2- to 9-bit. Figure 52 shows the SL layer's MSE for both the Identifier and Extractor when learning its 4 and 29 input-target associations, respectively. Results indicated that

as *l* increased, error decreased. This was more evident for the Extractor, where at the 500 epochs mark, *l* = 1 had an MSE of 0.02596, while for *l* = 4, it was 0.00079. For the Identifier, this was also observed. However, a major difference could only be seen for very small numbers of epochs (< 100). For example, at 25 epochs, *l* = 1 had an MSE of 0.025507 and 0.00617 for *l* = 4, while at 500 epochs, both were lower (MSE < 0.0001). Figure 52 also shows that although learning continued for 2500 epochs, no spikes in error were observed for either network. Results also showed that for *l* = 1, the MSE values remained unchanged (≈ 0.0027) for the last 200 epochs, indicating that the model was stuck. All other layers converged at lower values (MSE < 0.0001).

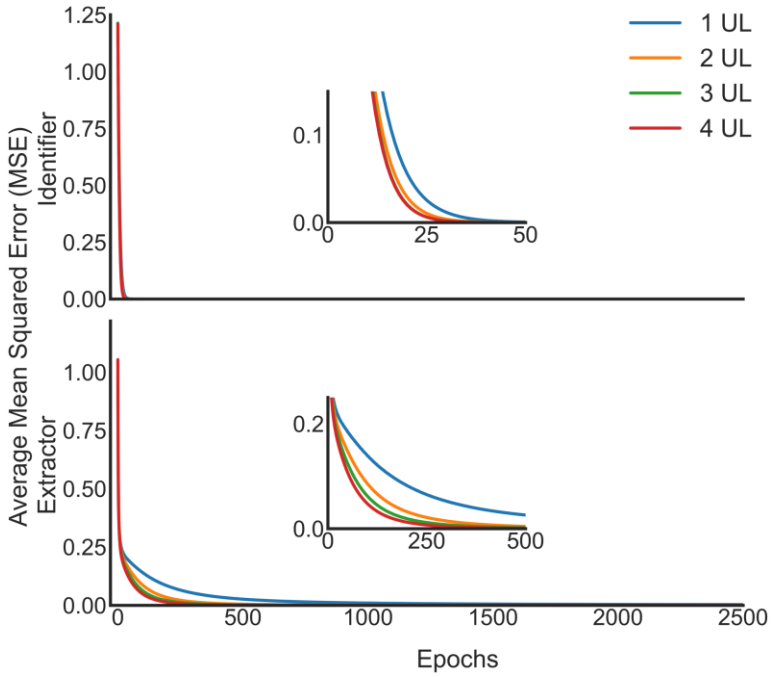

Figure 52. Average mean squared error (MSE) for the Identifier and Extractor under different l conditions.

Figure 53 shows the average recall performance and 95% confidence intervals in function of the 2- to 9-bit parity task. Results showed that a single layer could only solve the 2-bit consistently. However, for deeper architectures (*l* > 1), all parity tasks could be learned systematically despite changes in the time series length.

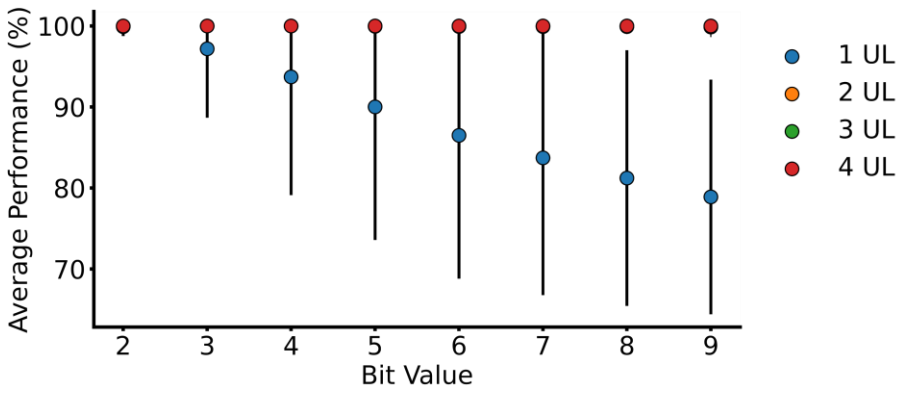

Figure 53. Average recall performance and 95% confidence intervals for each bit size and l condition.

Simulation II: Rote vs. Interacting Association

The model was compared to a MLP, and learning was stopped when each network's MSE was < 0.005. Results showed that both had perfect recalls and that as the number of bits increased, the minimum learning time increased exponentially for the MLP (Figure 54a)). However, this was not the case for the MF-BAMs as the total learning times only slightly increased, going on average from around 700 epochs (2-bit) to almost 1400 (9-bit), thus showing a much more linear trend. 95% confidence intervals also showed that the later model fluctuated less. Figure 54b) showed that as the number of bits increased, the Extractor module learning time also increased. However, this was not the case for the Identifier, as no changes in learning times were observed. This was expected since the number of learned patterns did not change (i.e., the same four context associations).

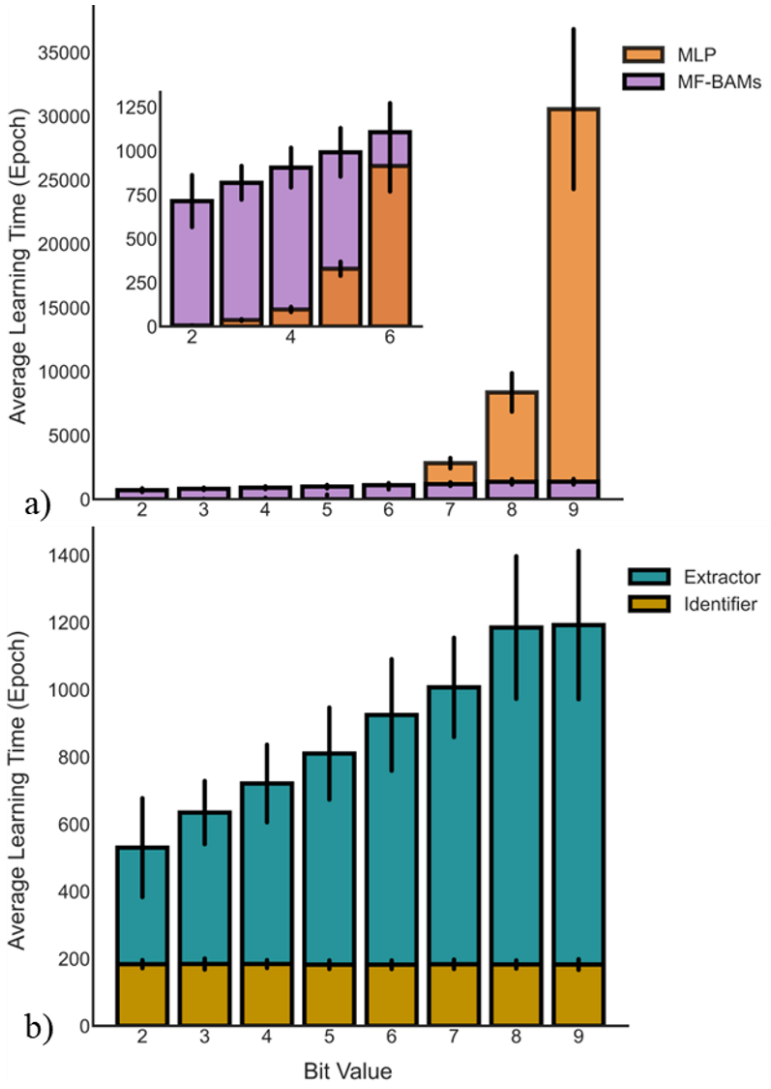

Figure 54. Average learning times and 95% confidence intervals for each bit value a) MF-BAMs (Identifier + Extractor; purple) and MLP (orange). b) Identifier (sand) and Extractor (turquoise).

Simulation III: Noise Filtering

The effect of noisy inputs was studied using pixel flips for the 2-bit task only. Figure 55 shows that as the number of flips increased, performances decreased. Interestingly, results show that even if a single layer (l = 1) was sufficient to solve it (Figure 53), there is an increase in noise tolerance proportional to the number of layers.

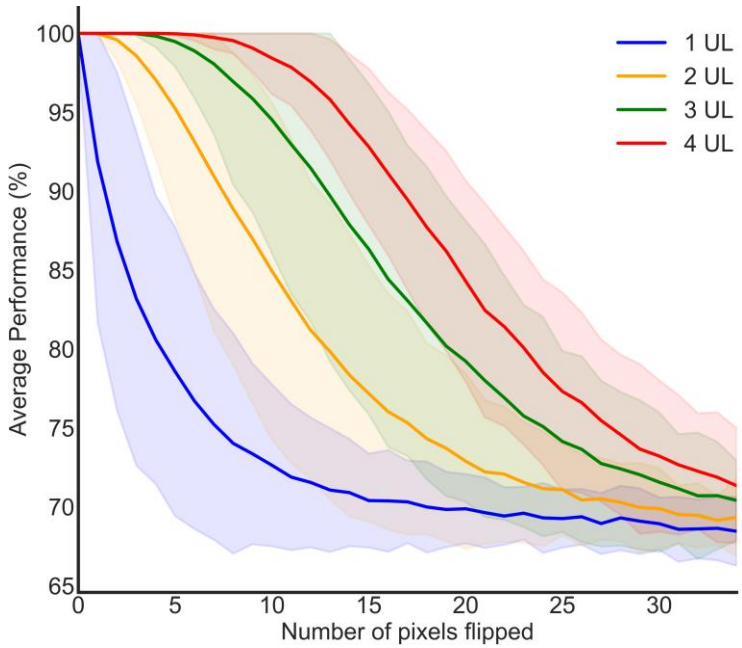

Figure 55. Average pixel flip performances for all l conditions (full lines) and 95% confidence intervals (shaded area).

Section IV: Discussion

A RNAM composed of two modules showed that it was possible to solve any *N*-bit parity tasks (2-9) solely by associative mechanism. This was achievable by the model's capability to learn multiple interacting associations.

Because of some nonlinear nature between the 33 associations, it was expected that the architecture of each module had to contain multiple hidden layers to learn correctly. Results in Figure 52 showed that to learn its 29 associations and ensure systematic performances (Figure 53), an architecture of $l > 1$ was necessary for the Extractor. This was not the case for the Identifier. Learning times in Figure 52 suggested that a shallower architecture would have sufficed, as there were no major differences between layer sizes. This can be attributed to the fact that the four associations were one-to-one, making them simpler and quicker to learn. Interestingly, this difference in learning speed between modules could be used to mimic a sort of short-term scenario

where the Identifier would constantly be retrained with new directives. This would be particularly handy in situations where the nature of the tasks does not drastically change, for example, counting the number of 0s instead of 1s and determining the parity. In such an instance, just the directive of when to count or stay fixed needs to change. In other words, the Extractor does not need to be retrained. Thus, adapting to such a shift becomes only a question of a few epochs, as only the Identifier, the small and fast portion of the model, would have to be adjusted. Regardless, it is worth further investigating if different behaviors could be obtained by changing the architectural parameter of each module independently.

By using pattern representations for the *N*-bit task, the dimensionality of the inputs remained the same. This had three important consequences. First, no changes in architecture are required to accommodate a shift in level, as shown by Simulation I. Second, no bias units are needed anymore due to the nature of the task becoming associative. Third, the representation of the *N*-bit became much more realistic, as presenting time series is akin to having a human gaze upon a series of values one at a time. As such, the consequences of using pattern representations are of interest and would be worth maintaining for future tasks, especially from a cognitive perspective.

In addition, by using the model's recursive properties, varying *N* only impacted the number of inputs presented without affecting learning times. Results in simulation II showed that the learning times did not increase exponentially but rather linearly. Although requiring ≈1400 epochs is still quite lengthy for solving the 2- to 9-bit from a plausible perspective, past research has shown that increasing layer depth and width can lower learning times and increase performances [Rolon-Mérette, Rolon-Mérette & Chartier, 2023a]. As such, it would be worth investigating which specific architectures results in the fastest and most accurate performances.

Of importance, when learning the correct associations and interactions, the objective no longer becomes to map the entire solution space but rather learn the necessary associations to solve the task. This is where this research distinguished itself, as it became more akin to how humans learn to perform these types of problems [Soto-Calvo et al., 2015] while remaining in a distributed bidirectional information signal processing approach. It showed that compared to rote learning, only 33 associations are needed rather than 1020. This becomes much more interesting from a cognitive perspective.

Results from simulation III showed that increasing the number of layers had a beneficial effect on noise tolerance. This is expected as MF-BAMs were not allowed to cycle their individual networks to clean the signal further. Therefore, the noise was filtered as the pattern went through each layer. Future results should look if there is a tradeoff between the number of cycles per layer and the number of layers. Accordingly, each module should be tweaked independently, as their roles seem complementary. Evidence in humans also points to this modular distinction [Barrett & Kurzban, 2006; Chandra et al., 2018; Fang et al., 2004].

Although it was shown that the *N*-bit could be solved regardless of its length, some improvements could be made. First, the contextual tags were explicitly given. A more interesting solution would be to have the model generate these tags by itself and then use them for learning [Rolon-Mérette, Rolon-Mérette & Chartier, 2018]. This would increase the plausibility of the approach. Furthermore, the information had to be carefully divided between the Identifier and Extractor during learning. An interesting solution would be that such division would be automatic and that the model could learn these online. Thus, a sort of self-specialization should occur without external intervention [Kohonen, 1990; Miljković, 2017].

It is worth highlighting that if solving a task greater than the 9-bit was desired, then by

doubling the dimension of the patterns (2x49), the model could also solve up to 99-bit with the same approach and advantages. Of further importance, this implementation is not specific to *N*-bit. The same approach could be applied to more cognitive relative tasks, such as the Dimensional Card Change Sort (DCCS) [Zelazo, 2006] and the Wisconsin Card Sorting test (**WCST**) [Miles et al., 2021].

In summary, by using associative mechanisms, it was possible to have different modules interact, and by simply teaching the appropriate associations, the desired behaviors could be obtained in a more generalized manner. As such, by following approaches that try to respect plausibility, new important steps toward better understanding human cognition through ANNs can be made.

Acknowledgement

This research was supported by the Fond de Recherche du Quebec Nature et Technologies (FRQNT) and the Ontario Graduate Scholarship (OGS) and was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) (210663).

Are Associations All You Need to Solve the Dimension Change Card Sort and N-bit Parity Task

Publication reference:

Rolon-Mérette, D., Rolon-Mérette, T., Chartier, S. (2024). Are Associations All You Need to Solve the Dimension Change Card Sort and N-bit Parity Task. *In: Samsonovich, A.V., Liu, T. (eds) Biologically Inspired Cognitive Architectures 2023. BICA 2023. Studies in Computational Intelligence, Springer, 1130* (730-740).

Abstract

When problem-solving, humans can cycle between learned rules to solve tasks. Yet, in artificial neural networks, this cognitive strategy is replaced by learning the entire solution space, making it far less effective. This work aimed to emulate the basis of this human strategy by using a recurrent neural associative memory model. To achieve this, two networks interacted; one served as a task Identifier and the other as a memory Extractor, giving the desired behavior influenced by the Identifier. Each network was trained on sets of interacting associations to represent behavior, such as recognizing shape, color, parity, and when a task started and ended. Once learned, the proposed model was subject to the dimension change card sort (DCCS) and the *N*-bit parity task. Results showed that the model could switch between behaviors to solve both tasks in linear time once the associations were learned. Moreover, this was possible with 93.3% fewer inputs and no retraining.

Keywords— Set-shifting, Associative Memory, ANN, BAM, DCCS, *N*-bit.

Introduction

Human cognition is very intricate to study. Different avenues have been used to comprehend how mental processes can emanate from networks of neurons [Arsalidou & Pascual-Leone, 2016; Goldwag & Wang, 2023]. Nowadays, a popular approach has been to use cognitive modeling to emulate behaviors found in cognition [Anderson, Matessa & Lebiere, 1997; Galotti, 2017; Lamb et al., 2014; Stella & Kenett, 2022; Sun, 2016]. In particular, formal models, known as artificial neural networks (ANNs), have seen a boom in their use due to their great success in learning complex tasks [Vinyals et al., 2019]. However, a significant question relates to how they accomplish these feats, especially from a biological and cognitive perspective.

For instance, ANNs usually require information prior hand for learning to be successful. Error signal at the output layer (i.e., global information) containing the solution typically needs to be backpropagated to direct the model's learning. From a biological perspective, evidence points to local changes, not global ones, steering this process [Hebb, 2005; Manneschi & Vasilaki, 2020]. Furthermore, ANNs usually accomplish significant feats, like multi-task or transfer learning, by combining various types of sophisticated mechanisms [Vinyals et al., 2019; Zhang & Yang, 2018], hence forming a heterogeneous model. Although it is undebatable that the brain has diverse functioning, findings suggest that it comes from recursive structures (i.e., homogenous networks) [Buckner & DiNicola, 2019]. Additionally, ANNs need training data to span most of the solution space [Marcus, 2018]. From a cognitive perspective, this can be akin to humans learning everything by heart. Although this is common, most tasks don't require individuals to absorb and retain an astronomical amount of data. Instead, signs point to humans using strategies they can build upon, forming heuristics rather than learning everything by heart [Dale, 2015]. This can be seen in individuals facing the Wisconsin Card Sorting Test (WCST) [Nyhus & Barceló, 2009].

Success in this task usually depends on working memory, attention, and set-shifting, to name a few [Coulacoglou & Saklofske, 2017]. The latter is significant, as it suggests that different abilities are learned and reused as "rules". For instance, the ability to recognize shapes and colors and how to count is necessary for solving the WCST. Rather than learning all possible combinations by heart (rote association), individuals apply these "rules" (e.g., shape, color, count, etc.) in different permutations based on the appropriate context to solve tasks. Integrating this strategy to ANNs could be fruitful, as it would lower the number of training data and aid in its multi-task and transfer learning capabilities.

Interestingly, this "rule" learning strategy can be represented by associative mechanisms. In fact, the associative memory (AM) framework states that these mechanisms enable various cognitive abilities and would function by interacting with each other [Anderson & Bower, 2014; Dacey, 2016, 2020; Lansner, 2009; Mandelbaum, 2015; Zhang & Yang, 2018]. The relevance of this information comes from the existence of an interesting subgroup of ANNs that are concerned with cognitive plausibility and that try to model AM. These are the recurrent neural associative memories (RNAMs) [Anderson, 1983]. Like ANNs, they are composed of artificial nodes and weighted connections. However, these networks typically create attractors due to their internal dynamics, which are often said to correspond to those found in the brain's neuronal state space. Consequently, RNAMs find themselves in various cognitive theories [Knoblauch, 2005]. An example of a popular RNAM that can learn auto- and hetero-association is the bidirectional associative memory (BAM). BAMs are nothing new and date back to the 80s [Kosko, 1988]. Yet, they remain one of the most widely used types of RNAM to study AM. Over the years, various versions have been proposed to overcome its original limitations [Acevedo-Mosqueda, Yanez-Marquez & Acevedo-Mosqueda, 2013; Rolon-Mérette, Rolon-Mérette & Chartier, 2018a, 2018b;

Rolon-Mérette et al., 2019; Rolon-Mérette, Rolon-Mérette & Chartier, 2019; Rolon-Mérette, Rolon-Mérette & Chartier, 2023a, 2023b, 2023c]. Of interest, the use of the Multi-Feature extracting bidirectional associative memory (MF) prior to a BAM has been proposed as a cognitively inspired model with interesting properties [Rolon-Mérette, Rolon-Mérette & Chartier, 2023b]. This homogenous multi-network can learn any association while remaining plausible [O'Reill, 1998]. It combines unsupervised and supervised learning and is based on simple local Hebbian and anti-Hebbian learning. Despite this combination, the MF-BAM alone cannot learn multiple interacting associations needed to emulate the "rule" learning strategy.

However, in a recent study, two MF-BAMs solved the *N*-bit parity task with 97% fewer training data and in a more general manner [Rolon-Mérette, Rolon-Mérette & Chartier, 2023b]. This was possible by teaching the joint model the interacting associations necessary to display a counting behavior. However, the proposed approach was only shown to learn "rules" for recognizing grayscale patterns, count, and parity. Suppose the goal is to eventually solve more complex tasks, like WCST, in a more cognitively plausible manner while using less training data and displaying some form of multi-task and transfer learning. In that case, models must also be able to scale up and integrate more "rules".

A proposed solution to remedy this is to use a modified version of the joint MF-BAMs model. By expanding the input and output layers to contain a distributed representation of color, the model could integrate this novel dimension into its learning. Furthermore, by adjusting the learned associations to profit from this extra dimensionality, it is believed that more "rules" could be learned. This would differ from previous works where separate channels were required to solve these complex problems [Kaplan, 2006]. To investigate scalability from past efforts [Rolon-Mérette, Rolon-Mérette & Chartier, 2023b], the *N*-bit learning strategy will be reused in

combination with those required to solve the Dimension Change Card Sort [Zelazo, 2006]. Both tasks are interesting as they can be seen as precursors to solving more complex ones like the WCST. Thus, this body of work aimed to teach the model rules related to recognizing shape, color, parity, counting, and when a task started and finished, to enable it to cycle through them to solve the *N*-bit parity and DCCS tasks in different succession.

To better illustrate this, the remainder of the paper divides itself as follows: Model, Methodology, Learning the DCCS and *N*-bit parity, and Discussion.

Model

Learning to solve the N-bit parity and DCCS requires interaction between two modules (MF-BAMs). The first module, the Identifier, functions by generating the appropriate instruction set (contextual information) from the incoming stimuli (input patterns). The second module, called the Extractor, retrieves the corresponding stored behaviors from the concatenation of its own internal state and the contexts. Figure 56a) shows the overall implementation of the proposed model. In short, an input pattern, **p**, is sent to the Identifier, which will recall the according context, **c** (instruction). This pattern will then be concatenated with a portion of the outputted behavior from the Extractor from the previous step. This concatenated pattern $\mathbf{o} = (\mathbf{s} \circ \mathbf{c})$ is sent to the second module to generate the desired behavior **b**. The Extractor's output comprises two concatenate parts $\mathbf{b} = (\mathbf{s} \circ \mathbf{z})$. The output **s** keeps track of which state the model is currently in and will be sent back to be concatenated to form **o** of the next time step, while **z** gives the final behavior.

Each MF-BAM has two components, the MF, which includes several unsupervised network layers (ULs), and the BAM, which contains a single supervised network layer (SL). These are all based on the unsupervised and supervised version of a modified BAM [Chartier & Boukadoum, 2006; Chartier et al., 2007]. Figure 56b) shows the architecture of the UL (white) and SL (grey).

The MF-BAM uses the same transmission function and learning rules for all its layers. The model's behavior changes only by modifying the MF's number of y-units in each hidden layer (denoted by *u*) and the number of hidden UL (represented by *l*) used. For all units, the same cubic transmission function, and for each connection, Hebbian/anti-Hebbian learning rules were used. For details, see [Rolon-Mérette, Rolon-Mérette & Chartier, 2023a].

Methodology

Instead of learning all input-target pairs required in rote associations, the model will first learn to perform five different "rules" selected to be sufficient to solve the *N*-bit and the DCCS. These are: To recognize the shape of 1 and 0, to recognize red and blue, to count from 0 to 9, to know parity, and to know when a task starts and ends. Multiple associations will represent each "rule". All the simulations were performed using Python installed from Anaconda [Anaconda Software Distribution, 2020; Rolon-Mérette et al., 2020] and were run on an RTX3090 graphics card.

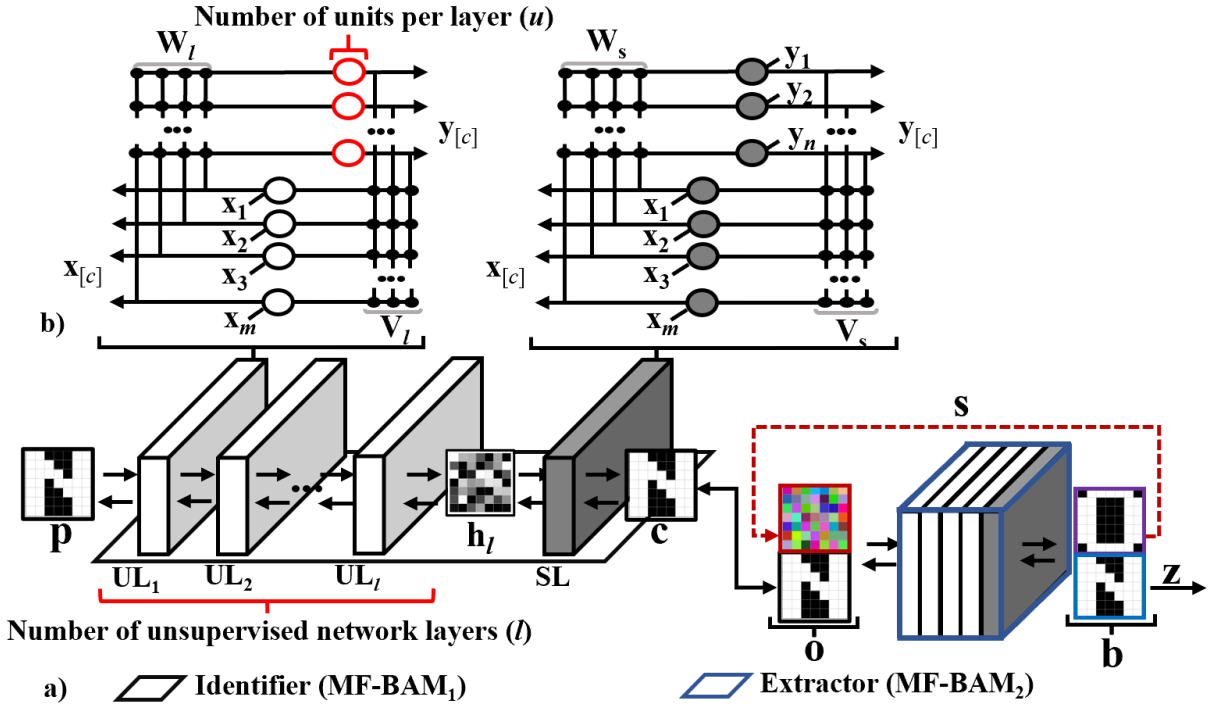

Figure 56. a) The two MF-BAM composed of Unsupervised network Layers (UL, white) and a single Supervised network Layer (SL, grey), where l represents the number of ULs and u the number of hidden units. b) The detailed underlying network that composes each layer.

For each pattern, the 3rd dimension represented an RGB format, where values of 1 exclusively found in the first dimension produced a color of red, in the second, green, and in the third, blue. If 1 was present in all 3 dimensions, white was obtained, whereas if -1 populated this space, the output was black. Figure 57 shows an example of this RGB format compared to a simple grayscale one.

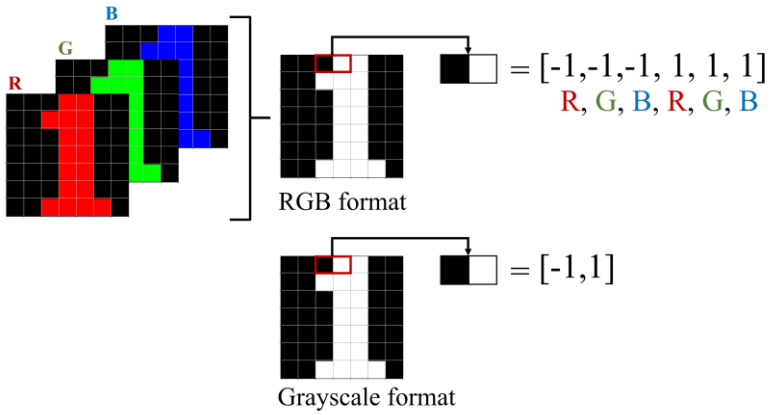

Figure 57. Example of RGB format compared to grayscale.

Different input-target pairs were created to represent each of the five "rules" (to recognize shape, color, parity, count, and when a task starts and finishes). Thus, a total of 9 pairs for the Identifier and 60 for the Extractor were created and taught to the modules. Figure 58 shows an example of these associations (see [Rolon-Mérette, Rolon-Mérette & Chartier, 2023b] for more details).

Once learning is accomplished, the model can be used to solve the *N*-bit parity task and DCCS in different orders. Each task was represented by different time series. For the DCCS, a similar protocol to [Zelazo, 2006] was used. However, an input sequence started with the pattern

D, followed by 14 patterns depicting either a blue 1 or red 0 and a corresponding target sequence depicting either the class blue 0 or red 1. For the first half, the task required classifying by color, and for the remainder, a pattern S was introduced to signal the condition changed (classifying by shape). For the *N*-bit parity task, a given bit size was portrayed as a time series starting with the pattern N, followed by the desired numbers of blue 1s and red 0s, and ending with the pattern P [Rolon-Mérette, Rolon-Mérette & Chartier, 2023c]. This recall process functioned by having the Identifier receive the input stream and determine the instructions that should be sent to the Extractor to determine the appropriate response. As the inputs are processed, the network should be able to output the correct behavior. Figure 59a) shows this for the 2-bit and 57b) for a short version of the DCCS.

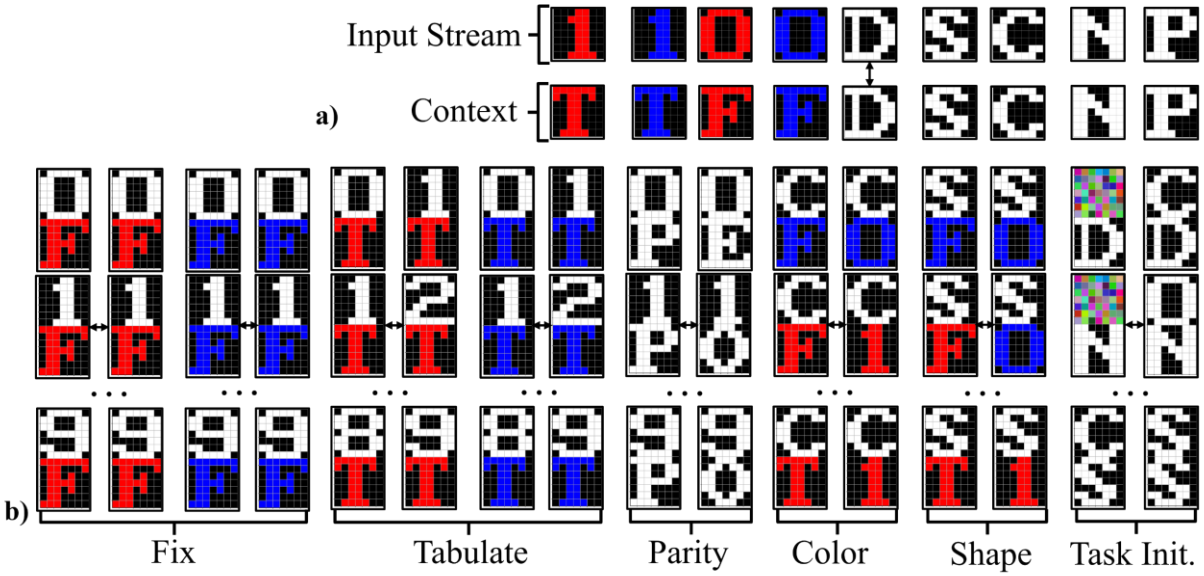

Figure 58. Example of interacting association "rules" for a) Identifier and b) extractor.

Learning the DCCS and *N*-bit

The model was tested in two paradigms. First, it had to learn only the interacting associations required for a specific task. As such, it was trained with a specific set for the DCCS (color and shape) and a different set for the *N*-bit (Fix, Tabulate, and Parity). This was performed

to assess if the model could solve both tasks independently and establish a type of baseline. Perfect performance for the DCCS was only possible if the model matched the 14 input patterns to their corresponding targets. For the *N*-bit, the entirety of the 2- to 9-bit ($1020 = \sum_{N=2}^{9} 2^N$) time series were presented during recall, and success was only possible if the correct parity was given for each.

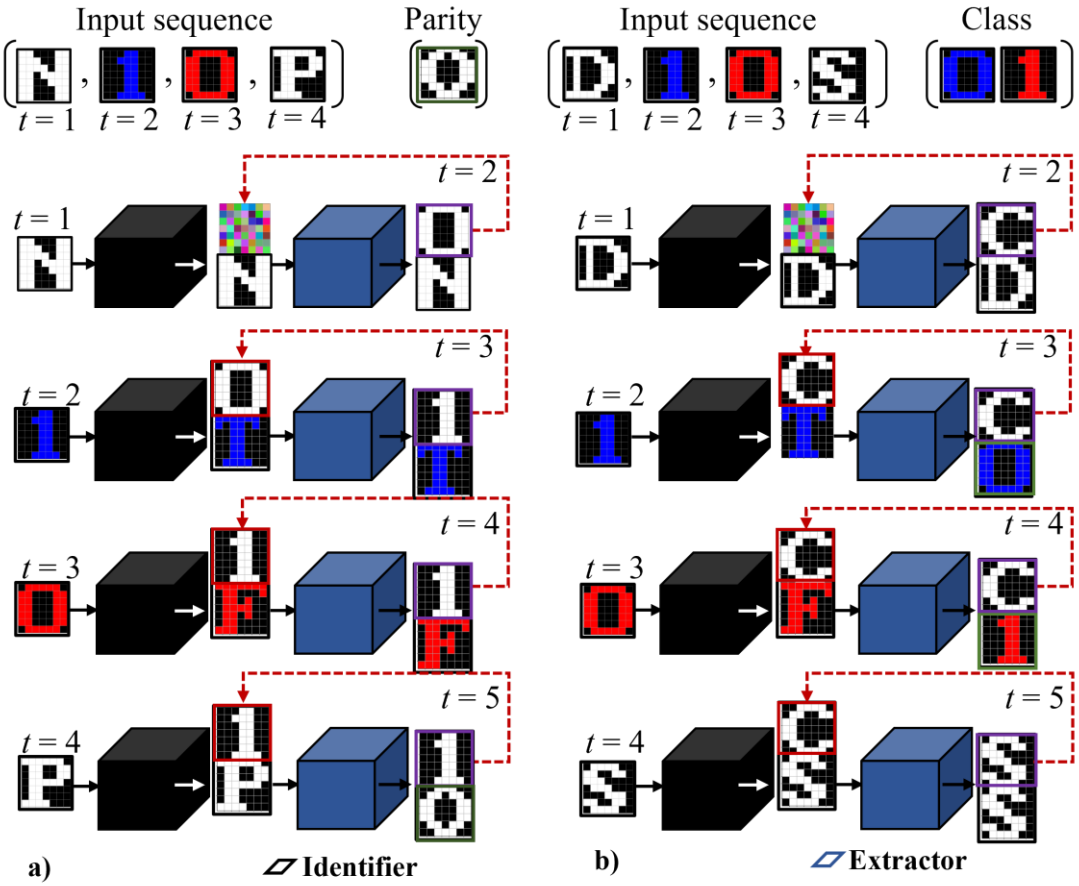

Figure 59. Example of the model on a) the 2-bit and b) a short version of the DCCS.

Furthermore, the model was compared to an out-of-the-box multilayer perceptron (MLP) from scikit-learn. Only the batch size was specified to ensure each pattern was presented once during an epoch. For the MLP, the rote association learning strategy was used on both the *N*-bit parity and DCCS task. Second, it had to learn all interacting associations and perform both tasks in random order. This was sought to investigate whether scalability had any drawbacks during

learning and whether the model could perform both tasks in harmony. After each training period, performances were measured by either presenting the DCCS followed by a random sequence of the 9-bit or vice versa and calculating their respective score. For each run, this was calculated in the same fashion for each task, where the generated behavior was compared to the original target on a pixel-per-pixel basis. If the generated and target patterns were identical, it was recorded as a success. As such, average mean squared error (MSE), performances, and 95% confidence intervals were reported. For all simulations, each MF-BAM's learning was conducted in the same fashion as in [Rolon-Mérette, Rolon-Mérette & Chartier, 2023b].

The transmission function parameter (δ) was set to 0.2, weights were randomly initialized ~U(-0.1,0.1), and the number of units (u) per layer was set to 100, which is about the size of the input patterns of the Extractor. To assess the complexity of the associations, the number of unsupervised network layers (l) ranged from 0 to 5, where 0 signified removing the MF component of the Identifier and Extractor (i.e., SL/BAM only for each). The model was trained for each l condition and set of interacting associations 50 times and for 3000 epochs.

Results

The model was tested on two paradigms: learning a subset of interacting associations for a specific task and learning all of them. Figure 60a) shows the SL layer's MSE for both the Identifier and Extractor when learning the associations related to the DCCS task only. Typically, very low MSE suggests successful learning. For the Identifier, all l conditions indicated this at the end of the 3000 epochs. However, for the extractor, $l = 0$ was unable to arrive at low values and was stuck at ≈0.105. On average, for $l>0$, MSE<0.001 after 436 epochs (std = 22.193). Figure 60b) shows the same measurements but for learning all interacting associations. The same previous general trend can be observed. However, for the Extractor, the $l = 0$ condition was stuck at a higher MSE

value (≈ 0.254), and to achieve MSE < 0.001 for $l > 0$, it took, on average, 1313 epochs (std = 43.046). Figure 60c) shows the average performance and 95% confidence interval for the 2- to 9-bit parity task when learning only the corresponding associations. In general, as l increased, the performance also increased consistently for the model. For $l = 0$, the model was unable to solve any bit size. Interestingly, as the number of bits increased, more ls were needed to achieve consistent perfect performances.

For the MLP, as N increased, performance plummeted. Figure 60d) shows the average performances for solving the DCCS (Purple line) and 9-bit (black line) when learning all interacting associations. The results from learning with only the specific interacting association for the DCCS (Purple dash) and N-bit parity (black dash) are also shown for comparison purposes. Results indicated that as l increased, performance increased as well. Furthermore, it showed that performances did not deteriorate when more interacting associations were learned for conditions where $l \geq 3$.

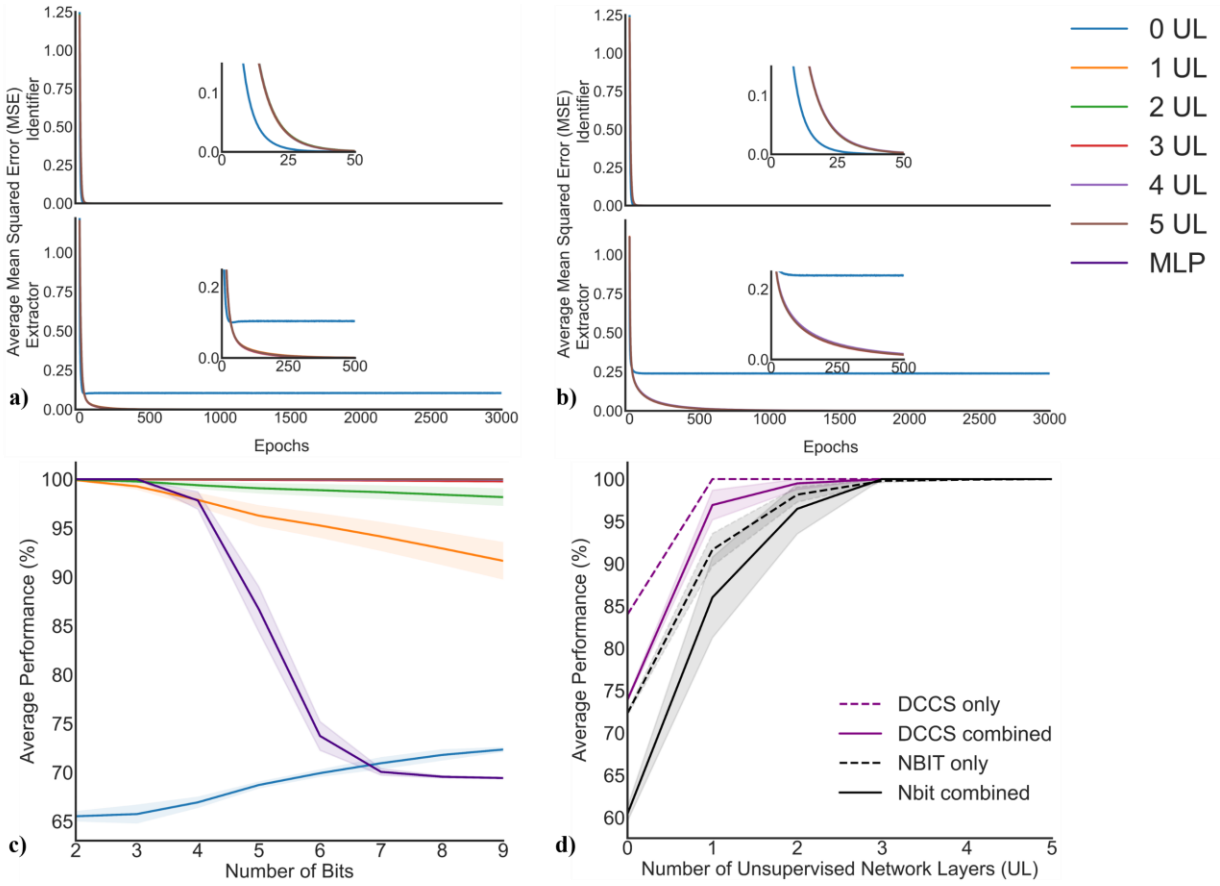

Figure 60. Average MSE for the Identifier and Extractor during learning of the interacting associations of the a) DCCS only, b) DCCS and N-bit. c) Average performance and 95% confidence interval on the 2- to 9-bit parity task for learning the interacting association for the *N*-bit only. d) Average performance and 95% confidence interval after learning the interacting associations of the DCCS (purple dash) and *N*-bit (black dash) only, and when combined for the DCCS (purple line) and *N*-bit (black line).

Discussion

In this work, it was shown that a RNAM composed of two modules (MF-BAM) could solve the DCCS and the 2-to 9-bit parity task in different order solely by associative mechanism. This was achievable by the model's capability to learn multiple interacting associations resembling "rules" related to recognizing shape, color, parity, count, and when to start and finish a task.

Furthermore, this was done by learning these "rules" outside the solution space of each task. Consequently, 93.3% fewer training data were needed to solve both tasks (69 inputs rather than $1028 = 8 + \sum_{N=2}^{9} 2^N$).

Contrary to the MLP and the rote association strategies that require constant adjustments to the architecture and retraining, the proposed model solved these tasks without these drawbacks and in a more general manner. Once the model learned the 69 patterns, both the 2- to 9-bit and DCCS could be solved regardless of the order of presentation. Of particular interest, it was shown in Figure 60d) that increasing the number of interacting associations had no significant drawbacks in terms of performance once $l \geq 3$. This is significant, as it indicates that the model can indeed scale up towards learning more "rules" and that these don't interfere with one another, a trait needed for lifelong learning [Parisi et al., 2019]. However, results shown in Figures 60a) and 60b) indicated that longer learning times should be expected if more pairs of patterns are to be learned. Furthermore, results in Figures 60c) and 60d) showed that the model had to possess a certain number of ULs for consistent perfect performances. Interestingly, for both tasks to be learned in random order, the model required $l \geq 3$. This can be explained by the ability of deeper MFs to consistently create representations of the original pairs (input-targets) that are linearly separable [Rolon-Mérette, Rolon-Mérette & Chartier, 2023a]. If this process is poorly executed, different patterns eventually fall in the same attractor, producing a linear solution when a nonlinear one is needed. It would be interesting to evaluate if other similar cognitive tasks require the model to have a minimum shape. This could help explain why younger children struggle in tasks such as the DCCS and WCST [Kaplan et al., 2006; Moriguchi & Hiraki, 2011; Rosselli & Ardila, 1993]. Results seem to hint towards requiring structural maturation to perform the appropriate computations to deal with nonlinearity. The performance of Figure 60c) also showed that despite

integrating color into the *N*-bit parity task, the model could also solve it similarly to previous findings [Rolon-Mérette, Rolon-Mérette & Chartier, 2023c]. This is important, as it gives substance to believe that the proposed approach can scale up and that RGBA and other types of color representation could be used. Humans can represent objects as multidimensional [Kloo, 2010]. Thus, a model must also be able to accommodate this.

It is essential to highlight some future directions that need to be addressed. First, all "rules" had to be presented at once. Although this method did show how transfer learning could be represented by the interacting associations, it did not address the problem of catastrophic forgetting. Individuals can learn new "rules" without eliminating those already in memory. A possible avenue could be having multiple MF-BAMs functioning as short- and long-term modules. Interacting associations could dictate important behaviors, such as when to extract novel 'rules' from the short-term module and when to perform replays to avoid catastrophic forgetting in the long-term component [Hayes, 2020]. Second, using a bipolar representation was problematic, as the absence of signal was represented by -1s. This meant an increase in correlation due to the higher dimensional representations. High correlation typically deteriorates performances. An interesting solution would be to modify the model to function only in a binary format, as the properties of 0 would help eliminate this correlation. Lastly, reinforcement behaviors were omitted. Tasks like the WCST depend on individuals to receive feedback and determine if a rule selection change should occur. It would be interesting to see if interacting associations and the attractor properties of the model could be used as a reinforcement mechanism to dictate this process of exploitation.

In summary, by using associative mechanisms, it was possible to have different modules interact, and by teaching the appropriate associations, the desired behaviors could be obtained.

Although the method was geared towards the DCCS and *N*-bit, results showed that its learned "rules" could be used harmoniously and potentially towards other similar tasks. This opens the door to solving complex tasks using a distributed AM model more efficiently than traditional rote learning. Learning rules or sets of actions that can be reused are interesting avenues, especially to diminish the amount of data required to train models. As such, by following approaches that try to respect biological and cognitive plausibility, significant improvements can be made to ANNs while also taking new steps toward better understanding human cognition.

Chapter 3 Conclusion

The advantages of jointly using contextual information and the MF-BAM were investigated in this chapter. By only using associative mechanisms, it was possible to learn sequences of associations that could be retrieved based on the context at hand. Furthermore, it was shown that the model could be directed to cycle through these sequences depending on the task at hand. This implementation allowed the model to respond in a more human-like manner by having it behave in a way that is akin to applying rules to solve a problem. These results show how AM mechanisms, while restrained to plausible mechanisms and centred around the notion of contiguity, can explain a variety of human cognitive behaviors through information signal processing and deserve further investigation.

General Discussion

Information signal processing models of AM are essential to understanding how human biology and cognition function and correlate. Current RNAM models offer an interesting avenue for studying this link. However, biologically and cognitively constrained RNAMs, such as the ones found in the BAM framework, lack the ability to deal with OMA and nonlinear tasks. This limits their interpretability of human behavior and their usefulness in explaining biology, cognition, and their connection. This thesis presented a solution to these problems by inspiring itself on human associative behavior, contextual information, and the laminar architecture of the brain.

Chapter 1 aimed to verify if the BAM framework could be tweaked to deal with OMA without losing its core properties. The proposed solution was based on Jordan's contextual information approach [Jordan, 1985, 1977], where contextual nodes were added to the input layer. Initial testing showed that context enabled a change in the pattern's overall representation without losing its original signal. Follow-up results showed that this could indeed be used to transform the OMA into multiple OOAs. However, the problem of linearity arose in more real-life scenarios where overlaps were present. This sparked the desire to deal with nonlinear relationships.

Thus, in Chapter 2, the goal was to verify if the BAM framework could be tweaked to deal with nonlinear relationships while restraining itself to biological and cognitive plausibility. This search resulted in creating a new model named the Multi-Feature - Bidirectional Associative Memory (MF-BAM). This novelty, inspired by the multilayer architecture found in columns and hypercolumns of neurons in the brain, used recursive homogeneous structures to self-generate different representations of patterns. Results showed that the MF-BAM could solve a wide array of nonlinear tasks in a consistent fashion. This encouraged the idea of using the novel architecture

as a module and investigating how a couple could interact through contextual information to achieve more complex behaviors.

As such, in Chapter 3, two MF-BAMs interacting with the aid of contextual information were used to emulate rules in human cognition. This enabled a type of dynamic where the Identifier would recognize the situation at hand and direct the Extractor to retrieve the corresponding behavior from memory. Results showed that the configuration of the Identifier and Extractor could be taught simple interacting associations that would allow it to learn rule-like behavior to solve tasks. These rules were akin to recognizing when to start and finish a task, when to count, counting from zero to nine, providing the parity of the value in memory, recognizing shapes and colors, and when to switch tasks. This permitted the model to depart from traditional rote association and focus more on a general and invariant way of solving tasks like the *N*-bit and DCCS. Results also showed an advantage in lowering the number of patterns to be learned and the total amount of learning time compared to traditional ML strategies.

These findings are interesting, as they showed how the BAM framework can be tweaked to overcome its limitations regarding OMA and nonlinearity without disregarding its core properties. Furthermore, they showed how additional abilities can be gained in the process. Of importance, it alluded to how simple mechanisms related to contiguity at both the biological (i.e., information signal processing) and cognitive (i.e., associative memory) levels can arrive at complex behavior. Each chapter provided interesting insights into their corresponding findings, and the reader is invited to dive back into them for a more detailed understanding. However, certain topics deserve to be expanded upon. What follows is a synthesis of these subjects and their possible inferences.

Implications for ANNs and RNAMs

The body of work presented here has important implications for RNAM and ANNs. Despite their popularity in the 80s, RNAM have been in the shadow of the so-called "deep networks" and their ability to solve complex problems through rote association. This is not surprising as deep networks have shown extraordinary feats in terms of engineering properties and application functionality [Floridi & Chiriatti, 2020; Vinyals et al., 2019]. Nowadays, it is hard to find a corner where these types of networks and their learning strategies are not employed. However, as many have alluded to [Johns et al., 2023; Thivierge et al., 2023; Marcus, 2018; Zhao et al., 2023], this popularity may be dwindling, and an imminent shift will be needed if topics like AGI are to be sought out for. This is in part due to the conceptual implication of these deep network strategies. Having access to answers prior to learning (global information), needing vast amounts of data spanning the solution space and being limited to learning everything by heart (rote association) are some of the limitations associated with these networks. Contrasted to human cognition, these networks seem foreign in the way they approach a problem [Bunge & Wallis, 2008]. Individuals, through exposure, learn simple rules that can be expanded upon, combined, and cycled through to produce more sophisticated actions. In fact, it could be said that a good part of human knowledge is built on these rules [Bunge & Wallis, 2008]. Furthermore, despite their ability to learn a solution by heart, employing more general tactics to solve a task seems to be the default setting. Thus, humans and ANNs generally differ in their conceptual approach. Mitigating this difference may help to progress the field of artificial intelligence, get closer to AGI, and understand its natural counterpart.

Results from this thesis showed a few glimpses towards such an initiative. For instance, by using a different approach, one inspired by biological and cognitive plausibility, it was shown that

overcoming OMA and nonlinear tasks is possible. Interestingly, this was achieved through simple Hebbian and anti-Hebbian learning rules (i.e., presentation by contiguity and use of local information) rather than backpropagation. Additionally, the opportunity to alter the model's behavior came from simply changing its architecture (i.e., the flow of information) rather than a functional parameter. Of importance, it was displayed that a shift in learning strategy could be achieved by representing actions (i.e., rules) required to solve a task through interacting associations. This interaction enabled a type of "IF" operator behavior often seen in symbolic networks [Fodor & Pylyshyn, 1988]. However, contrary to them, this was not explicitly encoded in the model. Rather, it emerged from learning pairs of stimuli. In fact, results from this thesis hint toward the possibility of teaching other types of operators, building upon them, while also being tolerant to noise, a challenge that remains important for certain cognitive models (see [Weng, 2011] for a review of symbolic and emergent models of human cognition). The totality of these properties, which are akin to those found in humans, are essential steps towards getting closer to AGI and merit further investigation.

At the minimum, these results highlight how other viable strategies can be pursued, even if the focus is on engineering properties and application functionalities. For instance, departing from rote association is not exclusive to the BAM framework. Other ANNs can be used to replicate the "rule learning" strategy and reap its advantages. This alone should be further investigated as it would aid in lowering the amount of training data and help with network resetting. At the maximum, results should be encouraging enough to rekindle great interest in the BAM framework and expand upon it. The thesis showed how additional properties were collected with its modifications. However, it did not show all of those that were conserved. It is important to mention that despite the alteration to the BAM framework, it maintained its ability to perform data

compression, noise filtering, prototype learning, image recognition, and just to name a few [Chartier & Boukadoum, 2006a, 2006b, 2011; Chartier et al., 2007; Rolon-Mérette et al., 2019]. These abilities become important when exploring more general-purpose paradigms, as having a single biological and cognitively plausible framework capable of tackling various tasks aids in studying human cognition as well as AGI.

What Are the Possible Roles of Context?

A recurrent theme of this thesis is the use of a representation for contextual information. In the technical mechanistic sense, context englobed external and internal information. In other words, it is an additional signal cooccurring with the mental state of interest. This simplistic view of contextual information may pose a problem by overreaching into various mechanisms and mixing them. However, it permits the distinction of information by following reasoning similar to the ones found in attentional processes [Chun, 2000; Itti & Koch, 2001]. Context could be viewed as the surrounding information that is not part of the primary focal point of attention. In such instances, it becomes more intuitive to imagine feelings, subsequent thoughts, surrounding backgrounds, etc., being part of the contextual information. Regardless of the definition, the research in this thesis alluded to important properties that involved additional information.

For instance, results showed how context can transform the overall interpretability of the mental state in question. This operated by having the additional information aid in distinguishing mental states and lowering their intra and inter-group correlation (see results in chapter one). Although it was not shown, the opposite effect of regrouping should be possible through this same principle. Thus, context could play an important role in classification and categorization behavior. By compressing information, the FEBAM model can regroup mental states and generate prototypes from them [Giguère et al., 2007a, 2007b]. This would also be true for the MF-BAM, as

it is composed of the same model. Thus, altering the mental state with context prior to compression can alter the result of this regrouping. In instances where information should be regrouped, context could guide the encoding process to ensure that learned mental states are closer in space. This would mean that the attractors representing these mental states are also being displaced towards a more proximal configuration. In the opposite scenario, it would perform like the decorrelation study by distancing and distinguishing them. Additionally, Chapters 1 and 3 showed how a single mental state can be learned in different contexts. This permits different interpretations, where, in certain instances, a mental state can be seen closer or further to other states. In other words, context would aid in creating this kind of push-and-pull effect seen in behaviors like classification and categorization.

Additionally, this thesis showed how there is an advantage of learning mental states in conjunction with contextual information, either for permitting the ability to perform OMA or to have IF-like behaviors in interacting modules. However, the question can be asked if context is also being learned and if this process is also governed by contiguity in humans. Evidence in the literature of contextual cueing in attention seems to support this idea [Chun, 2000]. As individuals learn, context is also registered, and during recall, this additional information is retrieved. Thus, it is logical to assume that this process could also be governed by contiguity. This could explain why manipulating the frequency of contextual exposure can influence classification and categorization behaviors, as seen in priming experiments (i.e., when a context is being prompt), where changes in how individuals' group mental states are observed [De Luca & Botelho, 2020]. An interesting question is whether there is a discrepancy between the frequency of presentation needed for information to be learned between mental states and context. Currently, the novel BAM framework does not consider both as separate entities. However, it would be of interest to follow up on having

additional modules responsible for treating exclusively mental states or contextual information. This would enable researchers to simulate a possible discrepancy between the two signals and evaluate if there would be an advantage of learning one faster than the other.

Of importance, in this body of work, contextual information was represented as a concatenated pattern. This means that an increase in dimensionality must occur to accommodate this additional signal. It also means that a specific position is key in determining what can be considered as a mental state in question or contextual information. However, this is but one way in which context can be depicted. An alternative form would be to merge the context into the mental state and create a novel representation that does not change the dimensionality of the inputs and is not sensitive to positioning. Through statistical redundancy, features emanating from the mental state and the context could be extracted separately. Meaning that the same opportunities (i.e., decorrelation, OMA, and IF-like behavior) related to contextual information shown in this thesis could be found.

Regardless of the form of presentation, this thesis showed how contextual information is essential to the BAM framework to produce complex behavior. One could extrapolate these findings and speculate how this may also be true in human cognition. Context could participate in compartmentalizing and organizing learned mental states. It could help explain why the outcome of a response can be influenced by context [Bugmann et al., 2007; McRobert et al., 2011; Tiberghien, 1986; Winocur et al., 2007]. In other words, context may have a role closer to that of a gating mechanism for the flow of information. These topics are of interest and deserve further consideration and investigation.

What Are the Possible Roles of Multilayer Architecture?

In chapter two, the creation of the MF-BAM was instrumental in dealing with nonlinear tasks. Results showed that success in transforming the nonlinear representation came mainly from the number of unsupervised network layers in the MF. By only changing the MF's architecture (flow of information), it was possible to obtain different answers. Thus, it was the employment of the same recursive architectures in a multilayered configuration that enabled a BAM to learn the nonlinear tasks. Interestingly, these network layers were able to learn from each other in a continuous fashion while using Hebbian and anti-Hebbian learning rules. In other words, it was based on mechanisms related to contiguity. Of importance, the head-to-toe organization did not hinder the process, as deeper network layers would eventually stabilize once its predecessor produced a consistent representation through associative learning. Thus, local information guided each network layer independently, and the entirety of the MF required a trickling-down effect from shallower layers. That being said, it would be interesting to investigate if information from deeper layers could influence shallower ones. Information, such as the variability in pattern generation or weight update fluctuation, for example, could be used. This could help lower learning times and maybe even address one conceptual limitation of ANNs: the decoupling of the learning and recall procedures [Basheer & Hajmeer, 2000; Jain et al., 1996]), where the latter permits weight update when information flows through the network and the former presume that no changes in synaptic efficacy can occur during this process. Regardless of the approach, it would be interesting to evaluate if a type of local and global-like signal emanating from these multilayers could be used to orient the model’s behavior. Results from Chapter 2 open to such possibility and deserve further investigation.

It is well documented that the human brain contains repeating structures within its organization [Buckner & DiNicola, 2019; Molnár & Rockland, 2020; Roe, 2019]. The whys of such configurations from an information signal processing standpoint based on HL remain to be fully understood. Despite focusing on the BAM framework and the limitations related to nonlinear tasks, results from this thesis have raised an important perspective regarding this topic. Chapter 2 showed that by using a multilayered architecture, incoming signals could be represented in various altered forms. Each form was generated based on a previous layer's depiction by extracting statistical occurrences. These changed representations had a different relationship with a common target and could be used to produce different behaviors. It was also shown that as the signal came from shallower to deeper layers, the overall behavior of the MF-BAM would transition from linear to nonlinear. This phenomenon permitted the expression of a general and a specific solution within the same model. Thus, information was not only altered, but the transition from layer to layer followed a smooth evolution, permitting such a shift in response. In other words, it was not a random nor an all-or-nothing process. It could be speculated that similar structures, like columns and hypercolumns in the brain, may also be altering an incoming signal gradually, like the MF. If true, this would imply that shallower layers provide faster responses compared to their distant twins, as the travelling time would be shorter. If shallower layers provide means to generate general answers, then these should naturally be faster to recall than specific ones (overfitted). Thus, distance of travel in a recursive architecture could be one of the reasons why there is a speed-accuracy trade-off [Huang et al., 2017; Li et al., 2019]. From an information signal processing point of view, this would give much more importance to the configuration of these repeated structures and would help suggest why they contain a certain number of layers. Too few would

result in the inability to perform complex operations, and too many could demand an unrealistic waiting time.

Additionally, in a parallel study outside the scope of this thesis, it was shown that the generated patterns from the deeper layers became more decorrelated [Rolon-Mérette et al., 2018a]. For the BAM framework, learning very lowly correlated or orthogonal patterns accelerates learning times. This is also true for more sparse encoding [Berberian et al., 2016]. It would be interesting to investigate if this also applies to biological networks. It is known that signals in the brain can be encoded into sparse representations [Quian Quiroga & Kreiman, 2010; Rolls & Treves, 2011]. Research from this body of work suggests that deeper structure may contain more sparse representations or, at the very least, differences in terms of correlation. This may also affect learning times and performances. Further investigation on this front would be of interest, as it would provide more evidence on the potential role of multilayer architectures.

In sum, this thesis touched on the potential roles of multilayer architectures from an information signal processing point of view. Topics such as self-learning governance, speed-accuracy tradeoff, and sparse encoding, to name a few, can be further investigated with this novel BAM framework. As layout in this section, the role of multilayer architectures could be related to transforming the incoming signal into different associated forms. These diverse forms could then be used by other structures to have a more assorted depiction of the incoming signal and thus produce more complex behavior. These topics are important and deserve to be further documented.

Is There a Need for Interaction?

In the final chapter of this thesis, a slight extension was introduced to the research question being pursued. By enabling the BAM framework to overcome its limitations, the complexity related to linearity and OMA could now be dealt with. This meant that it was possible to investigate

how different modules of the same model could interact through associative mechanisms to display new properties. Thus, the idea was to document how these could be used to emulate more complex behavior and speculate on the possible scope of association in cognition.

In this process, two MF-BAMs were employed with the initial objective of representing a learning strategy different from that of rote association. As previously stated, humans seem to employ rules when solving tasks [Bunge & Wallis, 2008]. This strategy enables a faster and more general way to navigate the day-to-day challenges. Thus, in Chapter 3, the goal was to represent a form of this strategy through associative mechanisms. In the process of representing rules, an interesting observation was made. By using only associations, it was possible to teach a module, the Extractor, all the necessary associative sequences required to represent different responses in solving the tasks. However, recognizing when to use them needed an additional signal. As such, the Identifier was employed to interact with the Extractor and dictate its retrieval behavior. This configuration allowed a type of working/short- and long-term memory dynamic, where the former (i.e., Identifier) would identify the situation at hand and direct the ladder (i.e., Extractor) to retrieve the corresponding behavior from memory. Results in Chapter 3 showed how this configuration could learn rule-like behavior to solve the task. Of interest, it also showed that scaling up to incorporate more rules had no significant drawbacks. This encourages the pursuit of this avenue, as the possibility of solving tasks, such as the Wisconsin Card sort Task [Kaplan et al., 2006; Miles et al., 2021] and other variants, while using these associative mechanisms, seem to be promising.

Despite this approach being a proof of concept of how the interaction between identical modules can arrive at interesting behavior, it hinted towards the need for a dynamic related to guided retrieval and storage. This possible requirement could perhaps only be achievable through the interaction of different areas. This thesis's results point towards how networks can learn from

contiguity. Regardless of the nature of the signal, it was shown that with local learning rules, it was possible to learn to associate information together based on pair presentation. This information was represented by attractors formed through the internal dynamics of the network. Chapter 1 showed how the association between one attractor and multiple other ones (i.e., OMA) could be selected with contextual information. This mechanism can result in multiple actions but remains limited to basic stimulus-response (**S-R**) behaviors [Bouton & Moody, 2004]. Thus, even if a network contains vast information, its outcome may remain limited to simple S-R. However, by having an additional network interacting with it, indicating what information to retrieve, when to retrieve it, when to change, etc., more complex behavior can emerge. Thus, a dynamic between guided retrieval and storage may be required. Take, for example, task-switching paradigms in humans [Siqi-Liu & Egner, 2020], where evidence of AM mechanisms is found. In such a scenario, task switching is cued, and a series of alternative actions are taken. This cue can often come from a range of information (a mental state, context, reinforcement, etc.). Regardless, once a cue successfully triggers a switch, the retrieval of information changes despite being in the presence of the same input. It could be argued that this process follows a similar dynamic to what was shown in Chapter 3. This dynamic may find itself at different levels of networks, be it in small populations of neurons or entire brain regions. It would be interesting to investigate this in biological systems, as it would help to better understand the role of interaction between networks of neurons and the ones between different types of memory (e.g., Working memory, short-term memory, long-term memory, etc.).

In sum, Chapter 3 showed how departing from rote association and using strategies that are akin to human cognition enable a more efficient and general approach toward learning and solving tasks. This outcome came from the interaction between identical models, whose roles

differed in terms of guided retrieval and storage. Thus, it may be possible that interaction is necessary to achieve complex cognitive behavior. Results in this thesis have shown how associations can be used to enable this interaction. There is promise in scaling up this approach to incorporate more rules and task-switching properties to emulate more complex behavior. Consequently, this avenue deserves more attention and further investigation.

Are Associations All We Need?

Throughout this body of work, different topics were touched upon while adhering to mechanisms related to associative memory. By investigating how networks of artificial nodes could communicate with each other and modify their connectivity through Hebbian and anti-Hebbian learning, the concept of contiguity was put at the forefront. This allowed this thesis to investigate how biology and cognition could be linked. It was shown that through internal dynamics, a BAM framework could generate attractors and have them associate with each other. Surpassing previous limitations, the findings of this study have shown how these attractors can be associated with situations of OMA and nonlinear relationships. From these mechanisms, elaborate interacting associations were used to emulate different mental states required to simulate rule-like behaviors. These were akin to recognizing when to start and finish a task, when to count, counting from zero to nine, providing the parity of the value in memory, recognizing shapes and colors, and when to switch tasks. The possibility of using contiguity to represent such complex behavior was a surprise. As it was important for this thesis to ground these findings in psychology, the results permitted a reimagination of what type of behavior could and couldn't be represented by associative mechanisms. For instance, despite not incorporating reinforcement into the BAM framework, it would be conceivable to represent it as an additional context. This would permit the orientation of the model's behavior based on reinforcement, like humans [Niv, 2009]. Through

contiguity, the associations of a series of actions and contexts that have, for example, the lowest or highest rewards can be imagined. Furthermore, the ability to construct and deconstruct mental states [Confalonieri & Kutz, 2020; Marcinowski et al., 2019] could also be investigated through AM mechanisms. As Chapter 2 showed, an incoming signal can be denoted in a series of altered representations through paired presentation in a multilayered network. These could be used to regroup or differentiate mental states based on these representations rather than their original depiction. The contiguity of these representations could be used to build novel categories or decompose them along a spectrum. In essence, with the findings from this thesis, various topics in cognition can be looked upon through mechanisms related to contiguity in an information signal processing setting.

That being said, some key features in human cognition remained a head-scratcher while questioning them through associative lenses. For instance, what can be considered a context and a mental state of interest raises the need for attentional mechanisms. Attention poses a problem for believing that associations are all you need. In fact, it could be speculated that attention is cut from a different cloth and should be considered closer to high-order processes [Lindsay, 2020]. The logic behind this thought would be that associations, formed through contiguity, would enable the creation of a mental state landscape, and attention would use these routes to navigate it. This would imply that associations have a more structural role in cognition and should be considered closer to lower-level processes. This could correlate with mental searches in tip-of-the-tongue phenomenon [Brown, 1991; Maril et al., 2001; Polyn et al., 2009], as attention may be navigating from one mental state to another by only traversing formed associated path between them to find the correct answer. Context could change the layout of the landscape and facilitate or hinder the navigational route of attention. Thus, attention may be influenced by associations, as also documented in the

attentional bias literature [Cisler & Koster, 2010; LeMoult & Gotlib, 2019], but it may not necessarily emanate from it. Regardless, attention seems to provide substance that it is not fully related to contiguity. Consequently, it provides evidence that associations may not be all you need.

Another concept that is harder to explain through contiguity is seen when manipulating information. For example, take the sequence of the English alphabet. The ease of going from "A" to "B" and then to "C," all the way to "Z," can be the consequence of learning and strengthening the associations between these mental states. However, when having to retrieve this sequence in reverse, the task becomes much more challenging and far from automatic [Hélie & Cousineau, 2011]. This could be because information is being retrieved differently, as it is kept in suspense and reworked to obtain the correct answer. So far, results from this thesis have shown that the novel BAM framework has the properties necessary to emulate retrieving sequences, stopping, and performing bidirectional associations. However, this inverse recall also requires a space where information is held in suspense and compared to a target from long-term storage without affecting it in the process [Cowan, 2008; Guérard & Saint-Aubin, 2012]. Likewise, attention is needed to shift from this working space and long-term memory, as there is a limit in storage capacity in this operational space [Cowan, 2014; Woodman & Chun, 2006]. In other words, more evidence that association and contiguity may not be enough to explain human cognition. However, automatic retrieval can be possible once an inverse sequence is repeated over time [Hélie & Cousineau, 2011]. Suggesting that associative mechanisms are not completely disregarded in such scenarios.

In essence, results from this body of work alluded to important behaviors, such as OMA, nonlinear associations, and rule-like actions, being represented by a mechanism related to contiguity. This opens the door to explaining how systems of information signal processing can amount to fundamental cognitive properties. It could be said that automatic and S-R behaviors are

governed by these mechanisms. Thus, at the lower levels, it could be imaginable that associations are all you need. Results from this thesis have shown how the flow of information can be configured in such a way that elaborate associative systems can emerge. From them, important properties related to storing information from paired exposure, cycling through responses and combining them through interactions can be emulated by associative mechanisms. These fundamental properties are arguably strong pillars of human cognition. However, human cognition may not be encapsulated with only associations and would require other processes to achieve the wide range of behavior portrayed by individuals. This means that the current BAM framework remains ill-equipped to explain human cognition, and other constrained variations will need to be proposed.

Lastly, it is important to remind the reader that the goal of this body of work was never designed to be overambitious and fit the model to human data. Rather, it was a continuation of past efforts to generate a framework capable of representing important mechanisms of associative memory. It is believed that by overcoming the limitations related to OMA and nonlinear relationship, the proposed model could be used to investigate more AM concepts such as extinction, rapid reacquisition, and latent inhibition, to name a few [Anderson & Bower, 2014; Bouton, 2013; Jamieson et al., 2012; Le Pelley et al., 2016]. Doing so could draw important parallels between behavioral data and computational experiments. Regardless, the present study permitted the additional documentation of the possible scope of associative mechanisms in human cognition [Buckner, 2011, 2017; Mitchell et al., 2009]. It also allowed to understand better how HL, AM, information signal processing, and cognition can all fit together.

Concluding Statement

Hebbian learning and associative memory share a simple mechanism that is based on contiguity. This thesis investigated how this simple process can help explain the link between biology and cognition through an information signal processing model. It focused on using RNAMs, most specifically expanding the BAM framework to overcome its limitations related to learning OMAs and nonlinear relationships. It proposed two major changes to achieve this. The first was to use contextual information to orient the model's behavior in situations of OMA. The second was to create a novel model inspired by the laminar architecture of the brain to tackle nonlinearities. It follows up by combining both approaches and showing how such modification can emulate complex cognitive behavior.

The experimental results showed the effectiveness of such modification and the possible role associative memory has in cognition. Chapter One showed that by using contextual information, it was possible to transform the one-to-many problem into a simple OOA. Thus overcoming the first limitation. Chapter Two introduced the multi-feature extraction bidirectional associative memory, a multilayered model capable of changing the nonlinear representations into a more manageable form. It showcased how a recurrent structure could transform a signal to follow a smooth transition between linear and nonlinear answers. Additionally, in Chapter Three, the effects of combining contextual information with the novel model revealed that more complex cognitive tasks could be emulated with only associative mechanisms.

Contextual information, in conjunction with the novel model, can now be seen as the associative brick that can be used to represent important pillars of human cognition. This biological and cognitively plausible approach shows how the simple mechanism of contiguity can account for complex behavior. It showcases how associative memory mechanisms can

emulate rules and how they can surpass the traditional rote association strategy found in ANNs. This avenue holds great promise for enhancing the generalization ability of ANNs. It also points to abundant evidence for a new accountability of associative memory mechanism. This paves the way for a deeper understanding of human cognition and how biologically plausible systems of information signal processing can arrive at such complex behaviors.

References

Acevedo-Mosqueda, M. E., Yanez-Marquez, C., & Acevedo-Mosqueda, M. A. (2013). Bidirectional associative memories: Different approaches. *ACM Computing Surveys (CSUR), 45*(2), 1-30.

Acevedo-Mosqueda, M. E., Yáñez-Márquez, C., & López-Yáñez, I. (2006). A new model of BAM: Alpha-beta bidirectional associative memories. *Computer and Information Sciences*, *21*, 286-295.

Ackman, J. B., & Crair, M. C. (2014). Role of emergent neural activity in visual map development. *Current opinion in neurobiology*, *24*, 166-175.

Adigun, O., & Kosko, B. (2019). Bidirectional backpropagation. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *50*(5), 1982-1994.

Aggarwal, C. C. (2018). Neural networks and deep learning. *Springer*, *10*, 978-3.

Alamia, A., Gauducheau, V., Paisios, D., & VanRullen, R. (2020). Comparing feedforward and recurrent neural network architectures with human behavior in artificial grammar learning. *Scientific reports*, *10*(1), 1-15.

Anaconda Software Distribution. (2020). *Anaconda Documentation*. Anaconda Inc. Retrieved from https://docs.anaconda.com/

Anderson, J. A. (1983). Cognitive and psychological computation with neural models. *IEEE transactions on systems, man, and cybernetics*, (5), 799-815.

Anderson, J. A., Silverstein, J. W., Ritz, S. A., & Jones, R. S. (1977). Distinctive features, categorical perception, and probability learning: Some applications of a neural model. *Psychological review*, *84*(5), 413.

Anderson, J. R., & Bower, G. H. (2014). *Human associative memory*. Psychology press. (Chapters 6-14)

Anderson, J. R., Matessa, M., & Lebiere, C. (1997). ACT-R: A theory of higher level cognition and its relation to visual attention. *Human–Computer Interaction*, *12*(4), 439-462.

Aristotle (2001). Aristotle's On the Soul and On Memory and Recollection. *J. Sachs (Trans.). Santa Fe: Green Lion Press*.

Arsalidou, M., & Pascual-Leone, J. (2016). Constructivist developmental theory is needed in developmental neuroscience. *npj Science of Learning*, *1*(1), 1-9.

Bain, A. (1868). The Senses and the Intellect. 3rd ed. *London: Longman's, Green, and Co*.

Bain, A. (1887). On 'Association'-Controversies. *Mind, 12*(46), 161-182.

Bank, D., Koenigstein, N., & Giryes, R. (2020). Autoencoders. *arXiv preprint arXiv:2003.05991*.

Barak, O. (2017). Recurrent neural networks as versatile tools of neuroscience research. *Current opinion in neurobiology*, *46*, 1-6.

Baroni, T. J. (2017). Mushrooms of the northeastern United States and eastern Canada. *Timber Press*.

Barrett, H. C., & Kurzban, R. (2006). Modularity in cognition: framing the debate. *Psychological review*, *113*(3), 628.

Basheer, I. A., & Hajmeer, M. (2000). Artificial neural networks: fundamentals, computing, design, and application. *Journal of microbiological methods*, *43*(1), 3-31.

Bechtel, W., & Bich, L. (2021). Grounding cognition: heterarchical control mechanisms in biology. *Philosophical Transactions of the Royal Society B, 376*(1820), 20190751.

Bégin, J., & Proulx, R. (1996). Categorization in unsupervised neural networks: The Eidos model. *IEEE Transactions on neural networks*, *7*(1), 147-154.

Bengio, Y., Lee, D. H., Bornschein, J., Mesnard, T., & Lin, Z. (2015). Towards biologically plausible deep learning. *arXiv preprint arXiv:1502.04156*.

Berberian, N., Aamir, Z., Hélie, S., & Chartier, S. (2016, July). Encoding sparse features in a bidirectional associative memory. In *2016 International Joint Conference on Neural Networks (IJCNN) IEEE*, (pp. 5119-5126) IEEE.

Berch, D. B., Foley, E. J., Hill, R. J., & Ryan, P. M. (1999). Extracting parity and magnitude from Arabic numerals: Developmental changes in number processing and mental representation. *Journal of experimental child psychology*, *74*(4), 286-308.

Berger, C., & Donnadieu, S. (2006). Categorization by schema relations and perceptual similarity in 5-year-olds and adults: A study in vision and in audition. *Journal of Experimental Child Psychology*, *93*(4), 304-321.

Bouton, M. E., & Moody, E. W. (2004). Memory processes in classical conditioning. *Neuroscience & Biobehavioral Reviews*, *28*(7), 663-674.

Bouton, M. E. (2013). Context and retrieval in extinction and in other examples of interference in simple associative learning. In *Current topics in animal learning* (pp. 25-54). Psychology Press.

Brown, T. H., Kairiss, E. W., & Keenan, C. L. (1990). Hebbian synapses: biophysical mechanisms and algorithms. *Annual review of neuroscience*, *13*(1), 475-511.

Brown, A. S. (1991). A review of the tip-of-the-tongue experience. *Psychological bulletin*, *109*(2), 204.

Bunge, S. A., & Wallis, J. D. (2008). *Neuroscience of rule-guided behavior*. OUP USA, 419-456

Buckner, C. (2011). Two approaches to the distinction between cognition and ‘mere association’. *International Journal of Comparative Psychology*, *24*(4).

Buckner, C. (2017). Understanding associative and cognitive explanations in comparative psychology. *The Routledge handbook of philosophy of animal minds*, 409-419.

Buckner, R. L., & DiNicola, L. M. (2019). The brain's default network: updated anatomy, physiology and evolving insights. *Nature Reviews Neuroscience, 20*(10), 593-608.

Bugmann, D., Coventry, K. R., & Newstead, S. E. (2007). Contextual cues and the retrieval of information from cognitive maps. *Memory & cognition, 35*, 381-392.

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F, Mueller, A., … & Varoquaux, G. (2013) API design for machine learning software: experiences from the scikit-learn project. *European Conference on Machine Learning and Principles and Practices of Knowledge Discovery in Databases, Workshop: Languages for Data Mining and Machine Learning,* 108-122.

Caporale, N., & Dan, Y. (2008). Spike timing–dependent plasticity: a Hebbian learning rule. *Annu. Rev. Neurosci., 31*, 25-46.

Carpenter, G. A., & Grossberg, S. (1988). The ART of adaptive pattern recognition by a self-organizing neural network. *Computer, 21*(3), 77-88.

Caudill, M., & Butler, C. (1990). Naturally intelligent systems. *MIT press*. (pp. 3-163).

Chaitra, P., & Kumar, D. R. S. (2018). A review of multi-class classification algorithms. *Int. J. Pure Appl. Math, 118*(14), 17-26.

Chandra, R., Gupta, A., Ong, Y. S., & Goh, C. K. (2018). Evolutionary multi-task learning for modular knowledge representation in neural networks. *Neural Processing Letters*, *47*, 993-1009.

Chartier, S., & Proulx, R. (2005). NDRAM: Nonlinear dynamic recurrent associative memory for learning bipolar and nonbipolar correlated patterns. *IEEE Transactions on Neural Networks*, *16*(6), 1393-1400.

Chartier, S., & Boukadoum, M. (2006a). A bidirectional heteroassociative memory for binary and grey-level patterns. *IEEE Transactions on Neural Networks, 17*(2), 385-396.

Chartier, S., & Boukadoum, M. (2006b). A sequential dynamic heteroassociative memory for multistep pattern recognition and one-to-many association. *IEEE Transactions on Neural Networks, 17*(1), 59-68.

Chartier, S., & Boukadoum, M. (2011). Encoding static and temporal patterns with a bidirectional heteroassociative memory. *Journal of applied mathematics, 2011*.

Chartier, S., Boukadoum, M., & Amiri, M. (2009). BAM learning of nonlinearly separable tasks by using an asymmetrical output function and reinforcement learning. *IEEE transactions on neural networks, 20*(8), 1281-1292.

Chartier, S., Giguère, G., Renaud, P., Lina, J. M., & Proulx, R. (2007, August). FEBAM: A feature-extracting bidirectional associative memory. *Proceedings of the 2007 International Joint Conference on Neural Networks,*1679-1684.

Chartier, S., Leth-Steensen, C., & Hébert, M. F. (2012). Performing complex associations using a generalised bidirectional associative memory. *Journal of Experimental & Theoretical Artificial Intelligence, 24*(1), 23-42.

Chauhan, V. K., Dahiya, K., & Sharma, A. (2019). Problem formulations and solvers in linear SVM: a review. *Artificial Intelligence Review*, *52*(2), 803-855.

Chun, M. M. (2000). Contextual cueing of visual attention. *Trends in cognitive sciences, 4*(5), 170-178.

Church, K., Ross, M., & Chartier, S. (2020). Using a Bidirectional Associative Memory and Feature Extraction to model Nonlinear Exploitation Problems. *In Proceedings of the 18th International Conference on Cognitive Modelling* (pp. 37-43).

Cichy, R. M., & Kaiser, D. (2019). Deep neural networks as scientific models. *Trends in cognitive sciences*, *23*(4), 305-317.

Cisler, J. M., & Koster, E. H. (2010). Mechanisms of attentional biases towards threat in anxiety disorders: An integrative review. *Clinical psychology review*, *30*(2), 203-216.

Clarke, A. (2017). Top-down cognitive influences on object recognition: a commentary on Perry and Lupyan (2016). *Language, Cognition and Neuroscience*, *32*(8), 947-949.

Collins, A. M., & Loftus, E. F. (1975). A spreading-activation theory of semantic processing. *Psychological review*, *82*(6), 407.

Collins, A. M., & Quillian, M. R. (1969). Retrieval time from semantic memory. *Journal of verbal learning and verbal behavior*, *8*(2), 240-247.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of machine learning research*, *12*(aug), 2493-2537.

Confalonieri, R., & Kutz, O. (2020). Blending under Deconstruction: The roles of logic, ontology, and cognition in computational concept invention. Annals of Mathematics and Artificial Intelligence, 88, 479-516.

Coombs, C. H. (1983). Psychology and mathematics. *University of Michigan Press*.

Coulacoglou, C., & Saklofske, D. H. (2017). Executive function, theory of mind, and adaptive behavior. *Psychometrics and psychological assessment*, 91-130.

Cowan, N. (2008). What are the differences between long-term, short-term, and working memory?. *Progress in brain research*, *169*, 323-338.

Cowan, N. (2014). Working memory underpins cognitive development, learning, and education. *Educational psychology review*, *26*, 197-223.

Csáji, B. C. (2001). Approximation with artificial neural networks. *Faculty of Sciences, Etvs Lornd University, Hungary*, *24*(48), 7.

Dacey, M. (2016) Rethinking associations in psychology. *Synthese*, *193*(12), 3763-3786

Dacey, M. (2020), Associationism in the Philosophy of Mind. *The Internet Encyclopedia of Philosophy*, https://iep.utm.edu/associat/#H7.

Dale, S. (2015). Heuristics and biases: The science of decision-making. *Business Information Review*, *32*(2), 93-99.

Davachi, L., & Wagner, A. D. (2002). Hippocampal contributions to episodic encoding: insights from relational and item-based learning. *Journal of neurophysiology*, *88*(2), 982-990.

De Luca, R., & Botelho, D. (2020). Olfactory priming on consumer categorization, recall, and choice. *Psychology & Marketing*, *37*(8), 1101-1117.

Dhar, V. K., Tickoo, A. K., Koul, R., & Dubey, B. P. (2010). Comparative performance of some popular artificial neural network algorithms on benchmark and function approximation problems. *Pramana*, *74*, 307-324.

Dukas, R. (2004). Evolutionary biology of animal cognition. *Annu. Rev. Ecol. Evol. Syst.*, *35*, 347-374.

Elman, J. L. (1990). Finding structure in time. *Cognitive science*, *14*(2), 179-211.

Elman, J. L., Bates, E. A., & Johnson, M. H. (1996). *Rethinking innateness: A connectionist perspective on development*, (10). MIT press. (Chapter 2).

Fang, C. Y., Fuh, C. S., Yen, P. S., Cherng, S., & Chen, S. W. (2004). An automatic road sign recognition system based on a computational model of human recognition processing. *Computer vision and Image understanding*, *96*(2), 237-268.

Ferrucci, D., Brown, E., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A. A., ... & Welty, C. (2010). Building Watson: An overview of the DeepQA project. *AI magazine*, *31*(3), 59-79.

Fitch, W. T. (2014). Toward a computational framework for cognitive biology: Unifying approaches from cognitive neuroscience and comparative cognition. *Physics of life reviews*, *11*(3), 329-364.

Floridi, L., & Chiriatti, M. (2020). GPT-3: Its nature, scope, limits, and consequences. *Minds and Machines*, *30*, 681-694.

Fodor, J. A., & Pylyshyn, Z. W. (1988). Connectionism and cognitive architecture: A critical analysis. *Cognition*, *28*(1-2), 3-71.

Földiak, P. (1990). Forming sparse representations by local anti-Hebbian learning. *Biological cybernetics*, *64*(2), 165-170.

Galotti, K. M. (2017). *Cognitive psychology in and out of the laboratory*. Sage Publications.

Galton, F. (1879). Psychometric Experiments. *Brain*, *2*(2), 149-162.

Galton, F. (1883). Inquiries into human faculty and its development. *Macmillan*.

Gelman, S. A., & Meyer, M. (2011). Child categorization. *Wiley Interdisciplinary Reviews: Cognitive Science*, *2*(1), 95-105.

Giguère, G., Chartier, S., Proulx, R., & Lina, J. M. (2007a). Category development and reorganization using a bidirectional associative memory-inspired architecture. *Proceedings of the 8th international conference on cognitive modeling*, 97-102.

Giguère, G., Chartier, S., Proulx, R., & Lina, J. M. (2007b, January). Creating perceptual features using a BAM-inspired architecture. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, 29 (29).

Giovanello, K. S., Keane, M. M., & Verfaellie, M. (2006). The contribution of familiarity to

associative memory in amnesia. *Neuropsychologia*, *44*(10), 1859-1865.

Goldwag, J., & Wang, G. (2023). DishBrain plays Pong and promises more. *Nature Machine Intelligence*, 1-2.

Goodwin, G. P., & Johnson-Laird, P. N. (2013). The acquisition of Boolean concepts. *Trends in cognitive sciences*, *17*(3), 128-133.

Graham, T. A. (1999). The role of gesture in children's learning to count. Journal of Experimental Child Psychology, 74(4), 333-355.

Guérard, K., & Saint-Aubin, J. (2012). Assessing the effect of lexical variables in backward recall. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *38*(2), 312.

Gupta, N. (2013). Artificial neural network. *Network and Complex Systems*, *3*(1), 24-28.

Hagiwara M. (1990) Multidirectional associative memory. In Proceedings of INNS-International *Joint Conference on Neural Networks*, *1*:3–6. IEEE

Hammer, R., Bar-Hillel, A., Hertz, T., Weinshall, D., & Hochstein, S. (2008). Comparison processes in category learning: From theory to behavior. *Brain Research*, *1225*, 102-118.

Hattori, M., & Hagiwara, M. (1995). Knowledge processing system using multidirectional associative memory. *In Proceedings of ICNN'95-International Conference on Neural Networks* (Vol. 3, pp. 1304-1309). IEEE.

Hattori, M., & Hagiwara, M. (1998). Multimodule associative memory for many-to-many associations. *Neurocomputing*, *19*(1-3), 99-119.

Hayes, T. L., Kafle, K., Shrestha, R., Acharya, M., & Kanan, C. (2020, August). Remind your neural network to prevent catastrophic forgetting. In *European Conference on Computer Vision* (pp. 466-483). Cham: Springer International Publishing.

Haykin, S. (2009). Neural networks and learning machines, 3rd ed., Copyright by Pearson

Education. *Inc., Upper Saddle River, New Jersey*, *7458*.

Hebb, D. O. (1949). The organisation of behaviour: a neuropsychological theory. *New York: Science Editions.*

Hebb, D. O. (2005). The organization of behavior: A neuropsychological theory. Psychology press.

Hélie, S., & Cousineau, D. (2011). The cognitive neuroscience of automaticity: Behavioral and brain signatures. *Cognitive sciences*, *6*(1), 25-43.

Herculano-Houzel, S. (2009). The human brain in numbers: a linearly scaled-up primate brain. Frontiers in human neuroscience, 31.

Hesselmann, G., Kell, C. A., Eger, E., & Kleinschmidt, A. (2008). Spontaneous local variations in ongoing neural activity bias perceptual decisions. *Proceedings of the National Academy of Sciences*, *105*(31), 10984-10989.

Hinton, G., & Sejnowski, T. J. (Eds.). (1999). Unsupervised learning: foundations of neural computation. *MIT press*.

Hintzman, D. L. (1991). Why are formal models useful in psychology. *Relating theory and data: Essays on human memory in honor of Bennet B. Murdock*, 39-56.

Hockley, W. E., & Consoli, A. (1999). Familiarity and recollection in item and associative recognition. *Memory & Cognition, 27*(4), 657-664.

Honey, C. J., Thivierge, J. P., & Sporns, O. (2010). Can structure predict function in the human brain?. *Neuroimage*, *52*(3), 766-776.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences, 79*(8), 2554-2558.

Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., ... & Murphy, K. (2017). Speed/accuracy trade-offs for modern convolutional object detectors. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 7310-7311.

Hume, D. (1739/1978). A Treatise of Human Nature. L. A. Selby-Bigge, and P. H. Niddich (eds.), *Oxford: Clarendon Press*.

Iacoboni, M. (2009). The problem of other minds is not a problem: Mirror neurons and intersubjectivity. *Mirror neuron systems: The role of mirroring processes in social cognition*, 121-133.

Itti, L., & Koch, C. (2001). Computational modelling of visual attention. *Nature reviews neuroscience*, *2*(3), 194-203.

Jacob, E. K. (2004). Classification and categorization: a difference that makes a difference. *Library Trends*, *52*(3), 515–540.

Jain, A. K., Mao, J., & Mohiuddin, K. M. (1996). Artificial neural networks: A tutorial. *Computer*, *29*(3), 31-44.

Jamieson, R. K., Crump, M. J., & Hannah, S. D. (2012). An instance theory of associative learning. *Learning & Behavior*, *40*, 61-82.

Johns, B. T., Jamieson, R. K., & Jones, M. N. (2023). Scalable cognitive modelling: Putting Simon's (1969) ant back on the beach. *Canadian Journal of Experimental Psychology/Revue canadienne de psychologie expérimentale*.

Jones, M. N., Willits, J., Dennis, S., & Jones, M. (2015). Models of semantic memory. *Oxford handbook of mathematical and computational psychology*, 232-254.

Jordan, M. I. (1985). The learning of representations for sequential performance (connectionism, parallel models, serial order). *Doctoral dissertation, University of California, San Diego*.

Jordan, M. I. (1997). Serial order: A parallel distributed processing approach. *In Advances in psychology* (Vol. 121, pp. 471-495). North-Holland.

Kadel, H., Feldmann-Wüstefeld, T., & Schubö, A. (2017). Selection history alters attentional filter settings persistently and beyond top-down control. *Psychophysiology, 54*(5), 736-754.

Kaplan, G. B., Şengör, N. S., Gürvit, H., Genç, I., & Güzeliş, C. (2006). A composite neural network model for perseveration and distractibility in the Wisconsin card sorting test. *Neural Networks, 19*(4), 375-387.

Kemker, R., McClure, M., Abitino, A., Hayes, T., & Kanan, C. (2018, April). Measuring catastrophic forgetting in neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence, 32*(1).

Kloo, D., Perner, J., Aichhorn, M., & Schmidhuber, N. (2010). Perspective taking and cognitive flexibility in the Dimensional Change Card Sorting (DCCS) task. *Cognitive Development, 25*(3), 208-217.

Knoblauch, A. (2005). Neural associative memory for brain modeling and information retrieval. *Information Processing Letters, 95*(6), 537-544.

Kohonen, T. (1972). Correlation matrix memories. *IEEE transactions on computers, 100*(4), 353-359.

Kohonen, T. (1974). An adaptive associative memory principle. *IEEE Transactions on Computers, 100*(4), 444-445.

Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE, 78*(9), 1464-1480.

Kohonen, T., Lehtio, P., Oja, E., Kortekangas, A., & Makisara, K. (1977). Demonstration of pattern processing properties of the optimal associative mappings. *In Proc Intl. Conf. on Cybernetics and Society.*

Kohonen, T., Lehtiö, P., Rovamo, J., Hyvärinen, J., Bry, K., & Vainio, L. (1977). A principle of neural associative memory. *Neuroscience, 2*(6), 1065-1076.

Kosko, B. (1988). Bidirectional associative memories. IEEE Transactions on Systems, man, and *Cybernetics, 18*(1), 49-60.

Kosko, B. (2021). Bidirectional associative memories: unsupervised Hebbian learning to bidirectional backpropagation. *IEEE Transactions on Systems, Man, and Cybernetics: Systems, 51*(1), 103-115.

Kumar, C. A., Ishwarya, M. S., & Loo, C. K. (2015). Formal concept analysis approach to cognitive functionalities of bidirectional associative memory. *Biologically Inspired Cognitive Architectures*, *12*, 20-33.

Kumar, K., & Thakur, G. S. M. (2012). Advanced applications of neural networks and artificial intelligence: A review. *International journal of information technology and computer science*, *4*(6), 57.

Kushiro, K., Harada, Y., & Takeno, J. (2013). Robot uses emotions to detect and learn the unknown. *Biologically inspired cognitive architectures*, *4*, 69-78.

Labib, R. (1999, July). New single neuron structure for solving nonlinear problems. *Proceedings of the International Joint Conference on Neural Networks*, *1*, 617-620.

Lamb, R. L., Vallett, D. B., Akmal, T., & Baldwin, K. (2014). A computational modeling of student cognitive processes in science education. *Computers & education, 79*, 116-125.

Lansner, A. (2009). Associative memory models: from the cell-assembly theory to biophysically detailed cortex simulations. *Trends in neurosciences*, *32*(3), 178-186.

Lee, S. W., O'Doherty, J. P., & Shimojo, S. (2015). Neural computations mediating one-shot learning in the human brain. *PLoS biology*, *13*(4).

LeMoult, J., & Gotlib, I. H. (2019). Depression: A cognitive perspective. *Clinical psychology review*, *69*, 51-66.

Le Pelley, M. E., Mitchell, C. J., Beesley, T., George, D. N., & Wills, A. J. (2016). Attention and associative learning in humans: An integrative review. *Psychological bulletin*, *142*(10), 1111.

Letzkus, J. J., Wolff, S. B., & Lüthi, A. (2015). Disinhibition, a circuit mechanism for associative learning and memory. *Neuron*, *88*(2), 264-276.

Leuner, B., Falduto, J., & Shors, T. J. (2003). Associative memory formation increases the observation of dendritic spines in the hippocampus. *Journal of Neuroscience*, *23*(2), 659-665.

Levering, K. R., Conaway, N., & Kurtz, K. J. (2020). Revisiting the linear separability constraint: New implications for theories of human category learning. *Memory & cognition*, *48*(3), 335-347.

Li, X., Zhou, Y., Pan, Z., & Feng, J. (2019). Partial order pruning: for best speed/accuracy trade-off in neural architecture search. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9145-9153.

Lim, S. (2021). Hebbian learning revisited and its inference underlying cognitive function. *Current Opinion in Behavioral Sciences*, *38*, 96-102.

Lindsay, G. W. (2020). Attention in psychology, neuroscience, and machine learning. *Frontiers in computational neuroscience*, *14*, 516985.

Liu, J., Gong, M., & He, H. (2019). Deep associative neural network for associative memory based on unsupervised representation learning. *Neural Networks*, *113*, 41-53.

Liwanag, A., & Becker, S. (1997). Improving associative memory capacity: One-Shot learning in

multilayer hopfield networks. *In Proceedings of the 19th Annual Conference of the Cognitive Science Society*. 442–447.

Locke, J. (1700/1974). An Essay concerning Human Understanding. Peter H. Nidditch (ed.). *Oxford: Clarendon Press*.

Mahesh, B. (2020). Machine learning algorithms-a review. *International Journal of Science and Research* (IJSR). 9, 381-386.

Mandelbaum, E. (2015). Associationist theories of thought. *In Zalta, E. N. (Ed.) The Stanford encyclopedia of philosophy*. http://plato.stanford.edu/archives/spr2015/entries/associationist-thought/.

Manneschi, L., & Vasilaki, E. (2020). An alternative to backpropagation through time. *Nature Machine Intelligence*, *2*(3), 155-156.

Marcinowski, E. C., Nelson, E., Campbell, J. M., & Michel, G. F. (2019). The development of object construction from infancy through toddlerhood. Infancy, 24(3), 368-391.

Marcus, G. (2018). Deep learning: A critical appraisal. arXiv preprint arXiv:1801.00631

Margaritis, K. G. (1995). On the systolic implementation of associative memory artificial neural networks. *Parallel Computing*, *21*(5), 825-840.

Maril, A., Wagner, A. D., & Schacter, D. L. (2001). On the tip of the tongue: An event-related fMRI study of semantic retrieval failure and cognitive conflict. *Neuron*, *31*(4), 653-660.

Martindale, C. (1991). Cognitive psychology: A neural-network approach. Thomson Brooks/Cole Publishing Co.

Marvin, M., & Seymour, A. P. (1969). Perceptrons. Cambridge, MA: MIT Press, 6, 318-362.

Mayes, A., Montaldi, D., & Migo, E. (2007). Associative memory and the medial temporal lobes. *Trends in cognitive sciences*, *11*(3), 126-135.

McClelland, J. L. (2000a). Connectionist models of memory. *The Oxford handbook of memory*, 583-596.

McClelland, J. L. (2000b). The basis of hyperspecificity in autism: A preliminary suggestion based on properties of neural nets. *Journal of Autism and Developmental Disorders, 30*(5), 497-502.

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics, 5*(4), 115-133.

McMahon, E., Wintermark, P., & Lahav, A. (2012). Auditory brain development in premature infants: the importance of early experience. *Annals of the New York Academy of Sciences, 1252*(1), 17-24.

McRobert, A. P., Ward, P., Eccles, D. W., & Williams, A. M. (2011). The effect of manipulating context-specific information on perceptual–cognitive processes during a simulated anticipation task. *British Journal of Psychology, 102*(3), 519-534.

Megill, J. (2014). Emotion, cognition and artificial intelligence. *Minds and Machines, 24*, 189-199.

Miles, S., Howlett, C. A., Berryman, C., Nedeljkovic, M., Moseley, G. L., & Phillipou, A. (2021). Considerations for using the Wisconsin Card Sorting Test to assess cognitive flexibility. Behavior research methods, 53(5), 2083-2091.

Miljković, D. (2017, May). Brief review of self-organizing maps. *In 2017 40th international convention on information and communication technology, electronics and microelectronics (MIPRO)* (pp. 1061-1066). IEEE.

Minemoto, T., Isokawa, T., Nishimura, H., & Matsui, N. (2017). Pseudo-orthogonalization of memory patterns for complex-valued and quaternionic associative memories. *Journal of*

Artificial Intelligence and Soft Computing Research, *7*(4), 257-264.

Mitchell, C. J., De Houwer, J., & Lovibond, P. F. (2009). The propositional nature of human associative learning. *Behavioral and Brain Sciences, 32*(2), 183-198.

Molnár, Z., & Rockland, K. S. (2020). Cortical columns. In Neural Circuit and Cognitive Development (pp. 103-126). Academic Press.

Moriguchi, Y., & Hiraki, K. (2011). Longitudinal development of prefrontal function during early childhood. *Developmental Cognitive Neuroscience*, *1*(2), 153-162.

Murray, J. M., & Escola, G. S. (2017). Learning multiple variable-speed sequences in striatum via cortical tutoring. *Elife*, *6*, e26084.

Nadler, R., Lin, L., & Minda, J. P. (2008, December). The effect of regulatory fit on the learning of complex rule-based categories. In *Canadian Journal of Experimental Psychology, 62*(4), 285-285.

Nelson, D. L., McEvoy, C. L., & Schreiber, T. A. (2004). The University of South Florida free association, rhyme, and word fragment norms. *Behavior Research Methods, Instruments, & Computers*, *36*(3), 402-407.

Neville, R. (2008). Third-order generalization: A new approach to categorizing higher-order generalization. *Neurocomputing*, *71*(7-9), 1477-1499.

Niv, Y. (2009). Reinforcement learning in the brain. *Journal of Mathematical Psychology*, *53*(3), 139-154.

Nyhus, E., & Barceló, F. (2009). The Wisconsin Card Sorting Test and the cognitive assessment of prefrontal executive functions: a critical update. *Brain and cognition, 71*(3), 437-451.

O'Reilly, R. C., & Munakata, Y. (2000). *Computational explorations in cognitive neuroscience: Understanding the mind by simulating the brain*. MIT press. (Chapters 6-8).

O'Reilly, R. C. (1996). Biologically plausible error-driven learning using local activation differences: The generalized recirculation algorithm. *Neural computation, 8*(5), 895-938.

O'Reilly, R. C. (1998). Six principles for biologically based computational models of cortical cognition. *Trends in cognitive sciences, 2*(11), 455-462.

Palm, G. (1980). On associative memory. *Biological cybernetics, 36*(1), 19-31.

Palm, G. (2013). Neural associative memories and sparse coding. *Neural Networks, 37*, 165-171.

Papineau, D., & Heyes, C. (2006). Rational or associative? Imitation in Japanese quail. In S. Hurley & M.Nudds (Eds.), Rational animals (pp. 187–195). *Oxford: Oxford University Press*

Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., & Wermter, S. (2019). Continual lifelong learning with neural networks: A review. *Neural networks, 113*, 54-71.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B. … & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research, 12*, 2825-2830.

Polyn, S. M., Norman, K. A., & Kahana, M. J. (2009). Task context and organization in free recall. *Neuropsychologia, 47*(11), 2158-2163.

Popa, C. A. (2017). Lie algebra-valued bidirectional associative memories. *Recent Advances in Soft Computing, 22*, 127-135.

Puig, M. V., Antzoulatos, E. G., & Miller, E. K. (2014). Prefrontal dopamine in associative learning and memory. *Neuroscience, 282*, 217-229.

Quian Quiroga, R., & Kreiman, G. (2010). Measuring sparseness in the brain: comment on Bowers (2009).

Raichle, M. E. (2010). Two views of brain function. *Trends in cognitive sciences, 14*(4), 180-190.

Raschka, S. (2018). Model evaluation, model selection, and algorithm selection in machine learning. *arXiv preprint arXiv:1811.12808*.

Ren, D., Wang, P., Qiao, H., & Zheng, S. (2013). A biologically inspired model of emotion eliciting from visual stimuli. *Neurocomputing*, *121*, 328-336.

Ritter, G. X., Sussner, P., & Diza-de-Leon, J. L. (1998). Morphological associative memories. *IEEE Transactions on neural networks*, *9*(2), 281-293.

Ritter, G.X., Diaz de Leon, J.L., & Sussner, P. (1999) Morphological bidirectional associative memories. *Neural Networks*, *12*(6), 851–867.

Ritter, S., Barrett, D. G., Santoro, A., & Botvinick, M. M. (2017, July). Cognitive psychology for deep neural networks: A shape bias case study. In *International conference on machine learning* (2940-2949).

Roe, A. W. (2019). Columnar connectome: toward a mathematics of brain function. Network Neuroscience, 3(3), 779-791.

Rolls, E. T., & Treves, A. (2011). The neuronal encoding of information in the brain. *Progress in neurobiology*, *95*(3), 448-490.

Rolon-Mérette, D., Rolon-Mérette, T., & Chartier, S. (2018a, July). Distinguishing Highly Correlated Patterns using a Context Based Aproach in Bidirectional Associative Memory. *In 2018 International Joint Conference on Neural Networks (IJCNN)* (1-8). IEEE.

Rolon-Mérette, T., Rolon-Mérette, D., & Chartier, S. (2018b). Generating Cognitive Context with Feature-Extracting Bidirectional Associative Memory. *Procedia computer science, 145*, 428-436.

Rolon-Mérette, D., Rolon-Mérette, T., & Chartier, S. (2019). Learning and Recalling Arbitrary Lists of Overlapping Exemplars in a Recurrent Artificial Neural Network. *In Proceedings of*

the International Conference on Cognitive Modelling (pp. 186-191).

Rolon-Mérette, T., Rolon-Mérette, D., Calderini, M., & Chartier, S. (2019) Different Brain, Same Prototype? Cognitive Variability within a Recurrent Associative Memory. *Proceedings of the International Conference on Cognitive Modelling*, 192-197.

Rolon-Mérette, D., Ross, M., Rolon-Mérette, T., & Church, K. (2020). Introduction to Anaconda and Python: Installation and setup. Quant. Methods Psychology, 16(5), S3-S11.

Rolon-Mérette, D., Rolon-Mérette, T., & Chartier, S. (2023a). A multilayered bidirectional associative memory model for learning nonlinear tasks. *Neural Networks*, *167*, 244-265.

Rolon-Mérette, D., Rolon-Mérette, T., & Chartier, S. (2023b, May). Using Bidirectional Associative Memory Neural Networks to Solve the N-bit Task. In *The International FLAIRS Conference Proceedings* (Vol. 36).

Rolon-Mérette, T., Rolon-Mérette, D., & Chartier, S. (2023c, May). Towards binary encoding in Bidirectional Associative Memories. In *The International FLAIRS Conference Proceedings* (Vol. 36).

Rolon-Mérette, D., Rolon-Mérette, T., Chartier, S. (2024). Are Associations All You Need to Solve the Dimension Change Card Sort and N-bit Parity Task. *In: Samsonovich, A.V., Liu, T. (eds) Biologically Inspired Cognitive Architectures 2023. BICA 2023. Studies in Computational Intelligence, Springer, 1130* (730-740). https://doi.org/10.1007/978-3-031-50381-8_79.

Rosedahl, L. A., & Ashby, F. G. (2022). Linear separability, irrelevant variability, and categorization difficulty. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *48*(2), 159.

Rosselli, M., & Ardila, A. (1993). Developmental norms for the Wisconsin Card Sorting Test in 5-to 12-year-old children. *Clinical Neuropsychologist, 7*(2), 145-154.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, *65*(6), 386.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, *323*(6088), 533-536.

Shen, Y., Wang, B., Chen, F., & Cheng, L. (2004). A new multi-output neural model with tunable activation function and its applications. *Neural processing letters*, *20*(2), 85-104.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., ... & Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. *nature*, *529*(7587), 484-489.

Singh, M. P., & Saraswat, V. K. (2017). Multilayer feed forward neural networks for non-linear continuous bidirectional associative memory. *Applied Soft Computing*, *61*, 700-713.

Siqi-Liu, A., & Egner, T. (2020). Contextual adaptation of cognitive flexibility is driven by task- and item-level learning. Cognitive, Affective, & Behavioral Neuroscience, 20, 757-782.

Smolensky, P. (1988). On the proper treatment of connectionism. *Behavioral and brain sciences, 11*(1), 1-23.

Song, Y., Lukasiewicz, T., Xu, Z., & Bogacz, R. (2020). Can the Brain Do Backpropagation? Exact Implementation of Backpropagation in Predictive Coding Networks. *Advances in neural information processing systems, 33*, 22566-22579.

Soto-Calvo, E., Simmons, F. R., Willis, C., & Adams, A. M. (2015). Identifying the cognitive predictors of early counting and calculation skills: Evidence from a longitudinal study. *Journal of Experimental Child Psychology*, *140*, 16-37.

Spillers, G. J., & Unsworth, N. (2011). Variation in working memory capacity and temporal–contextual retrieval from episodic memory. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 37(6), 1532.

Stalnaker, T. A., Calhoon, G. G., Ogawa, M., Roesch, M. R., & Schoenbaum, G. (2010). Neural correlates of stimulus-response and response-outcome associations in dorsolateral versus dorsomedial striatum. *Frontiers in integrative neuroscience*, *4*, 12.

Standage, D., Blohm, G., & Dorris, M. C. (2014). On the neural implementation of the speed-accuracy trade-off. *Frontiers in Neuroscience, 8*, 236.

Steinbuch, K. (1961). Die lernmatrix. *Kybernetik, 1*(1), 36-45.

Stella, M., & Kenett, Y. N. (2022). Knowledge Modelling and Learning through Cognitive Networks. *Big Data and Cognitive Computing*, *6*(2), 53.

Stoet, G., & Snyder, L. H. (2007). Task-switching in human and non-human primates: Understanding rule encoding and control from behavior to single neurons. *The neuroscience of rule-guided behavior*, 227-254.

Storkey, A. J., & Valabregue, R. (1999). The basins of attraction of a new Hopfield learning rule. *Neural Networks*, *12*(6), 869-876.

Südhof, T. C. (2021). The cell biology of synapse formation. *Journal of Cell Biology*, *220*(7), e202103052. https://doi.org/10.1083/jcb.202103052

Sun, R. (2006). The CLARION cognitive architecture: Extending cognitive modeling to social simulation. *Cognition and multi-agent interaction*, 79-99.

Sussner, P., & Campiotti, I. (2020). Extreme learning machine for a new hybrid morphological/linear perceptron. *Neural Networks*, *123*, 288-298.

Sussner, P., & Valle, M. E. (2006). Gray-scale morphological associative memories. *IEEE transactions on neural networks*, *17*(3), 559-570.

Suzuki, W. A. (2008). Associative learning signals in the brain. *Progress in brain research*, *169*, 305-320.

Tesauro, G., & Janssens, B. (1988). Scaling relationships in back-propagation learning. Complex Systems, 2(1), 39-44.

Thivierge, J. P., Giraud, É., & Lynn, M. (2023). Toward a Brain-Inspired Theory of Artificial Learning. *Cognitive Computation*, 1-8.

Thomas, M. S., & McClelland, J. L. (2008). Connectionist models of cognition. *Cambridge handbook on computational cognitive modelling. Cambridge University Press*

Tiberghien, G. (1986). Il context and cognition: Introduction. *Revue bimestrielle publiée avec le concours des Universités de Provenœ et Aix-Marseille II*, *6*(2).

Tremblay, C., Myers-Stewart, K., Morissette, L., & Chartier, S. (2013, July). Bidirectional Associative Memory and Learning of Nonlinearly Separable Tasks. In R. West & T. Stewart (Eds.), Proceedings of the 12th International *Conference on Cognitive Modeling*, Ottawa, Canada, 420-425

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., ... & Silver, D. (2019). Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, *575*(7782), 350-354.

Wallace, R. (2014). Cognition and biology: perspectives from information theory. *Cognitive Processing*, *15*, 1-12.

Wang, J. H., & Cui, S. (2018). Associative memory cells and their working principle in the brain. *F1000Research*, *7*, 108. https://doi.org/10.12688/f1000research.13665.1

Wattenmaker, W. D. (1995). Knowledge structures and linear separability: Integrating information in object and social categorization. *Cognitive Psychology, 28*(3), 274-328.

Wattenmaker, W. D., Dewey, G. I., Murphy, T. D., & Medin, D. L. (1986). Linear separability and concept learning: Context, relational properties, and concept naturalness. *Cognitive Psychology*, *18*(2), 158-194.

Weng, J. (2011). Symbolic models and emergent models: A review. *IEEE Transactions on Autonomous Mental Development*, *4*(1), 29-53.

Werker, J. F., Fennell, C. T., Corcoran, K. M., & Stager, C. L. (2002). Infants' ability to learn phonetically similar words: Effects of age and vocabulary size. *Infancy*, *3*(1), 1-30.

Willshaw, D. J., Buneman, O. P., & Longuet-Higgins, H. C. (1969). Non-holographic associative memory. *Nature, 222*(5197), 960-962.

Winocur, G., Moscovitch, M., & Sekeres, M. (2007). Memory consolidation or transformation: context manipulation and hippocampal representations of memory. *Nature neuroscience*, *10*(5), 555-557.

Woodman, G. F., & Chun, M. M. (2006). The role of working memory and long-term memory in visual search. *Visual Cognition*, *14*(4-8), 808-830.

Wu, J. X., Huang, P. T., Li, C. M., & Lin, C. H. (2018). Bidirectional hetero-associative memory network with flexible sensors and cloud computing for blood leakage detection in intravenous and dialysis therapy. *IEEE Transactions on Emerging Topics in Computational Intelligence*, *2*(4), 298-307.

Wurm, M. F., & Schubotz, R. I. (2017). What's she doing in the kitchen? Context helps when actions are hard to recognize. *Psychonomic bulletin & review*, *24*, 503-509.

Yang, G. R., & Wang, X. J. (2020). Artificial neural networks for neuroscientists: A primer.

Neuron, 107(6), 1048-1070.

Yang, J., Yang, W., & Wu, W. (2011). A novel spiking perceptron that can solve XOR problem. *Neural Network World*, *21*(1), 45.

Yanling, Z., Bimin, D., & Zhanrong, W. (2002, November). Analysis and study of perceptron to solve XOR problem. *In The 2nd International Workshop on Autonomous Decentralized System, 2002*. (pp. 168-173). IEEE.

Yano, Y., & Osana, Y. (2009, June). Chaotic complex-valued bidirectional associative memory. *Proceeding of the 2009 International Joint Conference on Neural Networks*, 3444-3449.

Yonelinas, A. P., Kroll, N. E. A., Dobbins, I. G., & Soltani, M. (1999). Recognition memory for faces: When familiarity supports associative recognition judgments. *Psychonomic bulletin & review, 6*(4), 654-661.

Yoshida, K. A., Fennell, C. T., Swingley, D., & Werker, J. F. (2009). Fourteen-month-old infants learn similar-sounding words. *Developmental science, 12*(3), 412-418.

Zador, A. M. (2019). A critique of pure learning and what artificial neural networks can learn from animal brains. *Nat Commun 10* (1): 1–7.

Zelazo, P. D. (2006). The Dimensional Change Card Sort (DCCS): A method of assessing executive function in children. *Nature protocols, 1*(1), 297-301.

Zhang, J., Zhu, S., Bao, G., Liu, X., & Wen, S. (2021). Analysis and design of multivalued high-capacity associative memories based on delayed recurrent neural networks. *IEEE Transactions on Cybernetics, 52*(12), 12989-13000.

Zhang, Y., & Yang, Q. (2018). An overview of multi-task learning. *National Science Review, 5*(1), 30-43.

Zhao, L., Zhang, L., Wu, Z., Chen, Y., Dai, H., Yu, X., ... & Liu, T. (2023). When brain-inspired AI meets AGI. *Meta-Radiology*, 100005.

Zilly, J., Achille, A., Censi, A., & Frazzoli, E. (2021). On Plasticity, Invariance, and Mutually Frozen Weights in Sequential Task Learning. *Advances in Neural Information Processing Systems, 34*, 12386-12399.