Investigating Building Blocks of Cognition in Realistic Reinforcement Learning Environments

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science
in
Computer Science and Engineering
by Research

by

Dolton Fernandes
2018111007
dolton.fernandes@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
April, 2024

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "**Investigating Building Blocks of Cognition in Realistic Reinforcement Learning Environments**" by **Dolton Fernandes**, has been carried out under my supervision and is not submitted elsewhere for a degree.

Date	Adviser: Prof. Bapi Raju S.

To Guiding Hands

Acknowledgments

The completion of this thesis would not have been possible without the guidance and support of numerous individuals. Firstly, I would like to express my deepest gratitude to my guide, Prof. Bapi Raju, for their invaluable mentorship and encouragement throughout the process. I am thankful to Pramod Kaushik who introduced and guided me heavily in the field of Computational Neuroscience & Reinforcement Learning and has been a very big part of my research journey. I also thank Prof. Frédéric Alexandre for insightful discussions, comments, and feedback during the course of this work.

I extend my thanks to Hugo Chateau-Laurent for his mentorship in some parts of this work. I also thank (in no particular order) Sambhav Solanki, George Tom, Ashish Singh, Tushar Chandra, Astitva Srivastava, Harsh Shukla, Sridhar M, Mohee Datta Gupta, Srinath Nair, Arohi Srivastav, Bhuvanesh Sridharan, Naren Akash, Ashwin Gopinath, Gautham Venugopal & Aishwarya Deogade for helping me out during the course of this work. Finally I would like to thank my parents for believing in me and making this possible.

This study was supported and part of the research project titled “Sequential Motor Skills: A Dual System View” funded by DST-INRIA Targeted Programme (CEFIPRA), Government of India.

Abstract

The convergence of Deep Reinforcement Learning (DRL) and cognitive neuroscience has opened avenues for probing neural representations within the brain. This thesis delves into the synergy between decision-making and representation by harnessing DRL's capabilities. Through a focus on dot motion perceptual decision-making task within a high-dimensional framework akin to psychological experiments, we acquire novel insights into how intricate neural networks tackle complex tasks. The resultant end-to-end model not only elucidates network strategies, but also offers a backdrop for understanding analogous brain processes.

Investigation reveals intriguing resemblances between the network's behavior and neural mechanisms observed in the middle temporal visual area (MT) of the brain, known for encoding direction and motion strength. Remarkably, direction selectivity emerges solely through reward-driven training, while graded firing patterns encode motion strength. A testable hypothesis arises, suggesting coherence-selective neurons within the MT population.

While conventional Reinforcement Learning (RL) presupposes uniform data distributions, real-world scenarios like autonomous driving and natural environments display Zipfian distributions, characterized by frequent occurrences amidst rare events. Inspired by complementary learning systems theory, this work introduces an architecture tailored for learning from Zipfian distributions. Unsupervised discovery of long tail states and their retention in episodic memory, coupled with recurrent activations, forms the core of the proposed architecture. Retrieval from episodic memory via similarity-based search, reinforced with weighted importance, amplifies performance across diverse Zipfian tasks. Our proposed architecture achieves higher accuracy than the IMPALA algorithm across all three tasks and evaluation metrics (Zipfian, Uniform, and Rare Accuracy).

This thesis advances the nexus of decision-making and neural representation through DRL, while proposing an innovative architecture capable of navigating the intricacies of Zipfian data distributions. It not only augments our grasp of cognitive building blocks but also bears implications for resolving challenges across real-world domains. This research in one small step that bridges the gap between computational models and cognitive processes, opening new vistas for understanding and application.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Reinforcement learning (RL) has emerged as a powerful framework for training intelligent agents to make sequential decisions in complex environments. RL algorithms have achieved remarkable success in various domains, such as game playing and robotic control. However, despite these achievements, there are still significant challenges that need to be addressed to bring RL closer to human-level cognitive capabilities.

One crucial aspect that distinguishes human cognition from current RL algorithms is their ability to understand and utilize the underlying building blocks of cognition. Humans possess a rich set of cognitive abilities, including memory, attention, and reasoning, which enable them to learn effectively and adapt to diverse environments. Integrating these cognitive building blocks into RL is a promising direction for advancing the field and bridging the gap between artificial and human intelligence.

The first building block that warrants attention is the emergence of direction selectivity and motion strength. Humans possess the ability to perceive and understand the direction and strength of motion in their environment. However, current RL algorithms often struggle to learn such perceptual abilities in complex tasks. By investigating the emergence of direction selectivity and motion strength through deep RL networks, we can gain insights into how these cognitive abilities can be integrated into RL algorithms, enabling agents to make more informed decisions in dynamic and visually rich environments.

Another vital building block to consider is episodic memory, particularly in long-tailed RL environments. While traditional RL algorithms focus on learning from immediate rewards, they often struggle to handle long-term dependencies and rare events. Episodic memory, inspired by human memory systems, can help RL agents store and recall past experiences, enabling them to learn from previous encounters and improve their decision-making capabilities. By investigating the application of momentum-boosted episodic memory in long-tailed RL environments, we can enhance the learning performance of RL agents and enable them to handle complex, rare, and non-uniformly distributed tasks.

Furthermore, reasoning abilities play a crucial role in human cognition. Humans excel at abstract reasoning, causal inference, and logical thinking, which enable them to perform complex tasks and

solve novel problems. Integrating reasoning mechanisms into RL algorithms can empower agents to make more informed decisions, plan ahead, and handle unseen scenarios with greater adaptability. By investigating the integration of reasoning abilities in realistic RL environments, we can develop algorithms that enhance the learning and decision-making capabilities of RL agents, enabling them to tackle complex and dynamic tasks effectively.

The goal of this thesis is to contribute to the advancement of RL towards more cognitively-inspired and human-like intelligent systems. By understanding and harnessing the fundamental building blocks of cognition, we can unlock new possibilities for RL algorithms, enabling them to tackle complex, dynamic, and visually rich scenarios with improved efficiency, adaptability, and generalization capabilities.

Overall, this thesis aims to shed light on the crucial role of cognitive building blocks in RL and provide valuable insights into the development of more capable and intelligent learning agents for real-world applications.

1.2 What are building blocks of cognition?

Cognition is the process of acquiring, storing, and using information. It is a complex process that involves many cognitive abilities, such as attention, perception, memory, language, problem-solving, and reasoning.

The building blocks of cognition are the fundamental mental representations and processes that underlie these abilities. One way to think about the building blocks of cognition is to consider the different types of information that we can represent mentally. For example, we can represent objects, events, relationships, and concepts. We can also represent our own thoughts and feelings.

Once we have represented information mentally, we can use various cognitive processes to manipulate it. For example, we can compare information, combine it in new ways, and reason about it. We can also use cognitive processes to control our behavior and emotions.

Here are some of the key building blocks of cognition that are relevant to the dot motion task:

- Visual perception: The ability to process and interpret visual information from the environment.

- Motion processing: The ability to detect and track the movement of objects in the environment.

- Spatial attention: The ability to focus attention on specific regions of space.

- Decision-making: The ability to choose between different options based on available information.

The dot motion task is one such perceptual decision-making task that requires participants to use their visual perception, motion processing, and spatial attention abilities to make accurate decisions about the direction of motion of a cloud of moving dots.

1.3 Experimental eco-systems for testing computational models of cognition

The human mind is an extraordinary complex system that has captivated scientists and philosophers for centuries. In recent decades, the field of cognitive science has made significant strides in understanding the mechanisms underlying human cognition. This progress has been fueled by the development of sophisticated computational models, which provide new insights into the neural basis of thought, perception, and behavior.

To effectively test and refine these computational models, researchers rely on experimental ecosystems. Experimental ecosystems are carefully designed environments that allow researchers to manipulate and observe the factors that influence cognition and decision making. These ecosystems provide a controlled setting where researchers can isolate specific variables, examine causal relationships, and draw meaningful conclusions about the cognitive processes involved. Below we discuss the test beds, that are used for such purposes.

1.3.1 DeepMind's PsychLab

In the realm of cognitive science, DeepMind's PsychLab [4] emerges as a powerful software platform meticulously designed to facilitate the creation and execution of cognitive experiments. Its comprehensive suite of tools empowers researchers to craft engaging and immersive experimental paradigms, encompassing a wide array of stimuli, including images, sounds, and videos. Moreover, PsychLab provides a rich collection of tasks, spanning visual search, decision-making, memory, and other cognitive domains. This versatility allows researchers to investigate a broad spectrum of cognitive processes within a single, unified framework.

A key strength of PsychLab lies in its ability to precisely control and manipulate experimental stimuli. This level of control is crucial for establishing causal relationships between specific factors and cognitive outcomes. Beyond its experimental flexibility, PsychLab offers a user-friendly interface and a comprehensive API, making it accessible to researchers with varying levels of programming expertise. This accessibility has fostered a growing community of PsychLab users, leading to the development of a rich ecosystem of plugins, extensions, and shared experimental paradigms.

The impact of PsychLab on cognitive science research is evident in its widespread adoption across diverse research domains. From studying visual perception and attention to investigating decision-making and memory processes, PsychLab has empowered researchers to make significant advancements in our understanding of the human mind.

Here are some specific examples of how PsychLab has been used to study various cognitive processes:

Visual Perception: Researchers have used PsychLab to study a wide range of visual perception phenomena, including motion perception, object recognition, and visual attention.

Decision-Making: PsychLab has been instrumental in studying decision-making processes under uncertainty, risk, and time pressure.

Memory: Researchers have used PsychLab to investigate various aspects of human memory, including working memory, long-term memory, and episodic memory.

Cognitive Development: PsychLab has been used to study cognitive development across the lifespan, from childhood to adulthood.

Neuropsychology: PsychLab has been used to study cognitive impairments in various neurological and psychiatric disorders.

1.3.2 Zipfian Environments

Imagine navigating a maze where the most lucrative treasure chests are hidden in the deepest, least-visited corners. Traditional RL algorithms, relying on frequent encounters for learning, might struggle to find these rare rewards. The paper [6] proposes Zipf's Playroom, a grid-world environment where rewards follow a Zipfian distribution [13], mimicking this real-world scenario. Agents must learn to explore efficiently, balancing the immediate gains of common rewards with the potential for larger, but less frequent, payoffs.

To further probe the limitations of RL in skewed environments, the authors introduce Zipf's Labyrinth and Zipf's Gridworld. These tasks involve navigating complex environments with dynamic transitions and delayed rewards, respectively. The labyrinth tests the agent's ability to adapt to unpredictable changes, while the gridworld challenges its planning capabilities in the face of delayed consequences. By varying the Zipfian exponent and environment parameters, the authors create a spectrum of difficulty, allowing researchers to benchmark and compare different RL algorithms under diverse levels of experience skew.

The paper's contribution lies not only in its novel tasks but also in its insightful analysis of existing RL approaches. The authors demonstrate how standard Deep Q-Learning [1] struggles to learn effectively in Zipfian environments, often neglecting rare experiences and focusing on easily obtainable rewards. This highlights the need for new RL algorithms that can prioritize exploration, handle delayed rewards, and adapt to dynamic environments with skewed experience distributions.

1.4 Contribution

- Proposing a computational model that works on input stimuli akin to those used in traditional monkey experiments.

- Model is able to show properties similar to the MT region of the brain purely through reward based training.

- Model can be used to further study/experiment and understand more about the dynamics of learning and motion perception.
- In Chapter 4, we propose an unsupervised long-tail discovery method using self-supervised momentum loss that is used to prioritize long-tail data.
- Further we propose an episodic storage of hidden activations, that is later reinstated in the recurrent layers so that rare trajectories are executed.
- Both of these proposed features are crucial in enabling the network to perform better than conventional architectures on a long-tail dataset.

1.5 Thesis Layout

This thesis is organized into five chapters, designed for flexible reading and information access. Following Chapter 2, which provides a comprehensive literature review and background, each subsequent chapter delves into a specific task discussed earlier. This modular structure allows readers to access directly the chapter relevant to their interest without requiring sequential reading. A brief description of all five chapters is given below.

- Chapter 1 provides with basic details on what cognition is and how it's relevant to this thesis. We also talk about the experimental ecosystems used for testing and give some details about the two majors task that we will look at.

- Chapter 2 describes the existing literature that is relevant to this thesis.

- Chapter 3 investigates the dot motion discrimination task from a RL perspective and provides with insights.

- Chapter 4 seeks to solve the task of applying reinforcement learning to tasks that follow a more real world distribution (Zipfian Distribution). We propose Momentum Boosted Episodic Memory (MBEM), inspired by the Complementary Learning System (CLS) theory [14].

- Chapter 5, the concluding chapter, serves as a culmination of our journey, where we unveil the key findings of this thesis and unlock promising doors for future investigation, inviting further inquiry and refinement.

Chapter 2

Literature Review and Background

2.1 Introduction to computational problems of perception and memory

Understanding visual motion plays a critical role in navigating our environment, interacting with objects, and perceiving the actions of others. However, accurately encoding and interpreting motion presents several computational challenges within the perceptual and memory systems. This subsection delves into these challenges, highlighting the specific difficulties faced in the context of the dot motion task and how they pose hurdles for both biological and artificial systems.

2.1.1 Feature Extraction and Integration

Visual motion information is inherent in the spatiotemporal changes of luminance across the retinas. Accurately extracting these changes and integrating them into a coherent representation of motion direction and strength requires complex processing. Early visual pathways in the brain (e.g., V1) encode spatial features like edges and orientations, while subsequent areas (e.g., V2, MT) specialize in integrating these features across time and space to extract motion information. This multi-stage hierarchical processing poses computational challenges in terms of feature selection, noise reduction, and efficient integration of spatiotemporal information.

2.1.2 Long-Range Dependencies and Context

Perceiving motion often involves integrating information across extended spatial and temporal scales. For example, judging the direction of a flock of birds requires tracking their collective movement over time, while recognizing a running person necessitates integrating body part movements into a coherent whole. This ability to capture long-range dependencies and interpret motion within context further complicates the computational demands of visual motion processing.

2.1.3 Ambiguity and Noise

Real-world visual input is inherently noisy and often ambiguous. Multiple interpretations of motion can arise from overlapping objects, shadows, and illumination changes. The brain must actively resolve these ambiguities, drawing on prior knowledge and context to infer the most likely motion percept. This requires sophisticated probabilistic reasoning and decision-making mechanisms within the visual system.

2.1.4 Memory and Expectation

Prior knowledge and expectations significantly influence our perception of motion. For instance, seeing a ball bounce after hitting the ground is less surprising than seeing it levitate. The brain dynamically incorporates stored memory representations and current expectations into the processing of motion information, further refining and biasing our perceptual interpretations. Integrating memory and expectation into real-time motion processing poses additional computational demands on the neural system.

2.1.5 Challenges for Artificial Systems

Emulating the human brain's remarkable ability to process motion remains a significant challenge for artificial intelligence. Traditional computer vision algorithms often struggle with feature extraction, noise reduction, and handling ambiguities inherent in real-world scenarios. Building artificial systems after investigating things can accurately perceive and interpret motion across diverse contexts and incorporate memory and expectation effectively.

2.2 Reinforcement Learning

Reinforcement learning (RL) is a powerful learning paradigm rapidly reshaping the landscape of artificial intelligence. Unlike traditional supervised learning, where labeled data dictates correct answers, RL takes inspiration from biological learning processes. Imagine a curious child exploring a playground, trying different actions and learning from the resulting rewards (joy of the swing) and punishments (bump from the slide). Similarly, RL agents interact with their environment, taking actions, and gradually learn what works best through a "trial-and-error" approach.

2.2.1 Markov Decision Process (MDP)

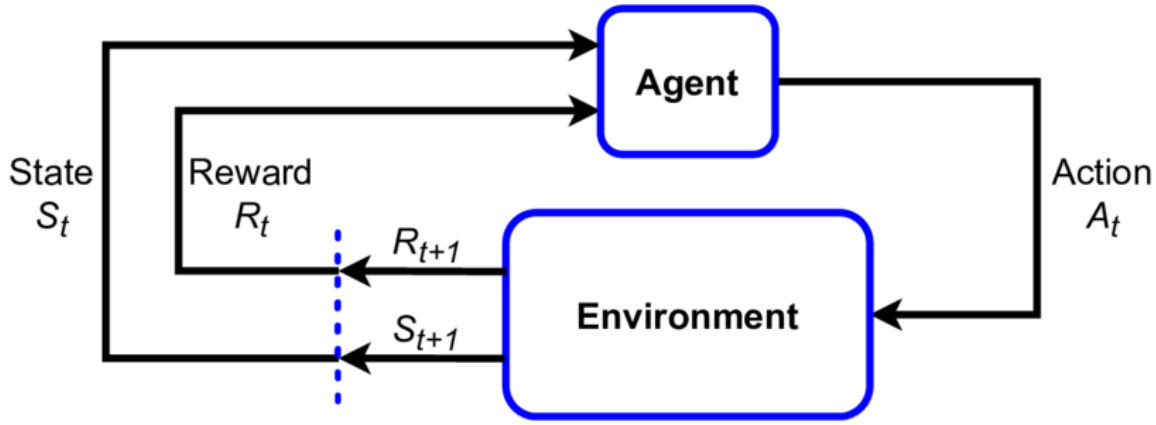


Figure 2.1: Interaction between an agent and its environment. The agent observes the state S_t of the environment and selects an action A_t based on its policy. The environment then transitions to a new state S_{t+1} and provides a reward R_t to the agent.

Let's assume an environment $\mathcal{E}$ which provides the agent with an observation S_t, the agent selects an action A_t, and then the environment responds by providing the agent with the next state S_{t+1}. The interactions between the agent and environment are formalized by *MDPs* which are reinforcement learning tasks that satisfy Markov property [15]. It is defined by the tuple $< \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{T}, \gamma >$ where, $\mathcal{S}$ represents the set of states, $\mathcal{A}$ is the set of actions, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ denotes the reward function. $\mathcal{T} : \mathcal{S} \times \mathcal{A} \rightarrow Dist(\mathcal{S})$ represents the transition function mapping state-action pairs to a distribution over next states $Dist(\mathcal{S})$ and $\gamma \in [0, 1]$ is the discount factor. An abstract visualization for a MDP can be seen in Figure 2.1.

2.2.2 Q-Learning

Imagine an agent traversing a maze, represented by a set of states S and a set of actions A. At each state s, the agent can choose an action a, incurring a transition to a new state s' with probability $P(s', a|s)$. This transition also produces a reward $r(s, a)$, reflecting the immediate consequence of taking action a in state s.

Q-Learning seeks to estimate the optimal Q-value, $Q*(s, a)$, which represents the expected discounted sum of future rewards obtained by taking action a in state s and following an optimal policy thereafter. Bellman's equation elegantly expresses this relationship:

$Q(s, a) = E[r(s, a) + \gamma \sum Q(s', a')P(s', a'|s, a)]$ where γ is the discount factor, balancing immediate rewards with future gains. The core of Q-Learning lies in iteratively updating estimates of $Q(s, a)$ towards $Q*(s, a)$ using the observed transitions and rewards:

$Q(s,a) = Q(s,a) + \alpha[r(s,a) + \gamma max'_a Q(s',a') - Q(s,a)]$, where α is the learning rate, controlling the update step size.

2.2.3 Deep Q-Networks

For complex environments with numerous states, the Q-table approach becomes impractical. DQN addresses this by replacing the Q-table with a deep neural network, $Q(s,a;\theta)$, parametrized by weights θ. This network estimates Q-values for all actions given the current state, eliminating the need for explicit state-action pairs.

DQN employs experience replay, storing past transitions in a buffer and randomly sampling batches for training. This decorrelates training examples, reducing variance and improving network stability. Further, a separate "target network" with slowly-updated weights is used to compute "bootstrapped" Q-values during updates, injecting stability and combating overfitting.

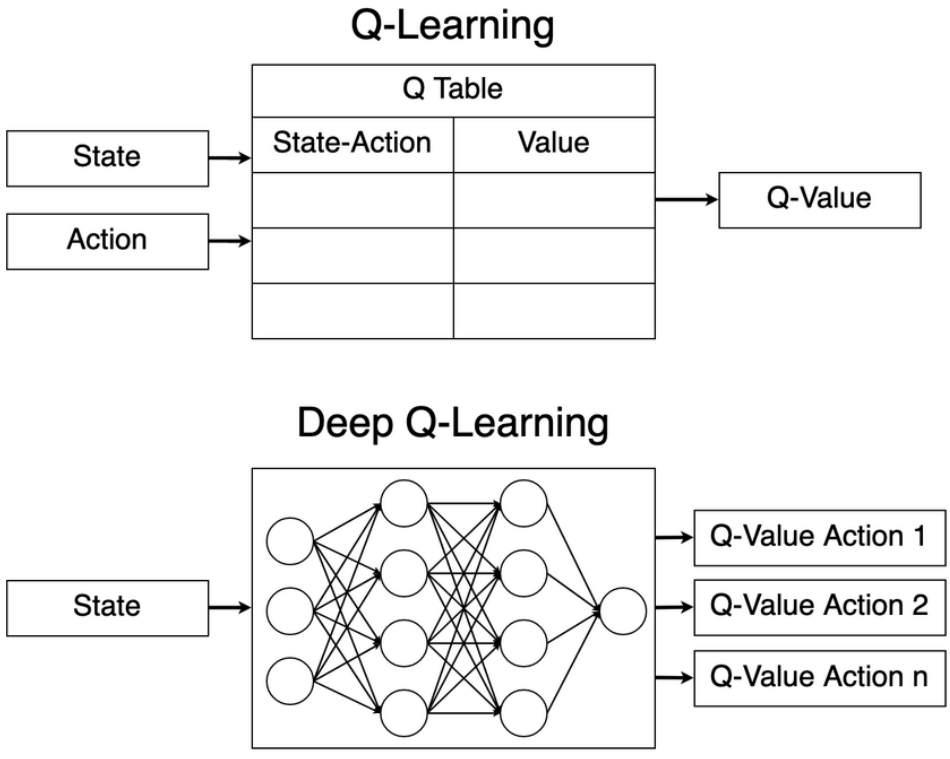


Figure 2.2: The image illustrates the difference between traditional Q-learning and Deep Q-learning. The Q-learning approach uses a lookup table to store the Q-values for each state-action pair, while Deep Q-learning employs a neural network to approximate the Q-values. Taken from [1].

Both Q-Learning and DQN utilize Bellman's equation as their guiding principle, but DQN leverages the flexibility of neural networks to handle vast state spaces and learn complex relationships between states and actions (Figure 2.2). This empowers agents to tackle real-world challenges, making them valuable tools for diverse applications in robotics, resource management, and even healthcare.

2.2.4 Actor Critic Methods

Actor-Critic methods form a prominent class of reinforcement learning algorithms that integrate both value-based and policy-based approaches to optimize decision-making in sequential decision problems. The fundamental idea behind Actor-Critic methods is to simultaneously learn a policy (the actor) and a value function (the critic). The policy is responsible for selecting actions in the environment, while the value function evaluates the desirability of states or state-action pairs.

Let $\pi(a|s, \theta)$ represent the policy, where a is an action, s is the state, and θ denotes the policy parameters. The critic approximates the state-value function $V(s, w)$ or the action-value function $Q(s, a, w)$, where w represents the critic's parameters.

The actor updates its policy parameters to maximize expected cumulative rewards, typically using gradient ascent. Simultaneously, the critic refines its value estimates based on the temporal difference error, capturing the difference between predicted and actual rewards.

Mathematically, the policy update is expressed as: $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$, where $J(\theta)$ is the expected cumulative reward, and α is the learning rate. The critic's parameters w are updated using the temporal difference error: $w \leftarrow w + \beta \nabla_w (R + \gamma V(s', w) - V(s, w))$ Here, R is the immediate reward, γ is the discount factor, and s' is the next state.

Actor-Critic methods offer a balance between exploration and exploitation, making them versatile for various reinforcement learning tasks. The combination of policy and value function updates enables efficient learning in complex environments.

2.2.5 IMPALA

In the grand theatre of reinforcement learning algorithms, IMPALA (Importance Sampling Actor-Critic Learning Algorithm) steps onto the stage, not as a solitary performer, but as a conductor, leading an orchestra of actors and critics to accelerate learning in vast and complex environments. Imagine, instead of a single agent exploring a labyrinth, IMPALA deploys a team of agents, each traversing their own copy, sharing their discoveries and refining their knowledge collectively. This distributed asynchronous nature is the cornerstone of IMPALA's speed and efficiency.

2.2.5.1 Independent Learners

Unlike traditional actor-critic methods, IMPALA employs a team of actors and critics running concurrently. Each learner interacts with its own copy of the environment, collecting experiences and inde-

pendently updating its local actor and critic networks. This parallel exploration significantly increases data collection and speeds up learning compared to a single agent.

2.2.5.2 Information Exchange

But how do these independent teams contribute to a unified understanding? IMPALA utilizes periodic parameter updates, where each local network transmits its learned parameters to a central "global" network. This global network is then used to generate importance sampling weights, guiding future local updates towards valuable, policy-relevant experiences.

2.2.5.3 Importance Sampling

This is where the "I" in IMPALA shines. Each learner's experience is weighted based on how likely it is to be generated by the current, global policy. This prioritizes valuable data, preventing learners from getting lost in irrelevant corners of the environment and ensuring efficient convergence towards the optimal policy.

2.3 Motion Perception

Motion perception, the ability to sense and interpret the movement of objects around us, plays a crucial role in navigating our environment, interacting with objects, and understanding the actions of others. Yet, accurately deciphering motion poses significant challenges, both for our biological visual systems and for artificial intelligence algorithms. The dot motion [16] task is a classic paradigm for studying motion perception. It presents subjects with a field of randomly moving dots, some coherently moving in a specific direction and others moving randomly. The task challenges the observer to identify the direction and strength (number of coherently moving dots) of the collective motion. This seemingly simple task requires integrating information across space and time, extracting subtle patterns from noise, and factoring in prior knowledge about motion dynamics.

2.3.1 Middle Temporal Visual Area (MT)

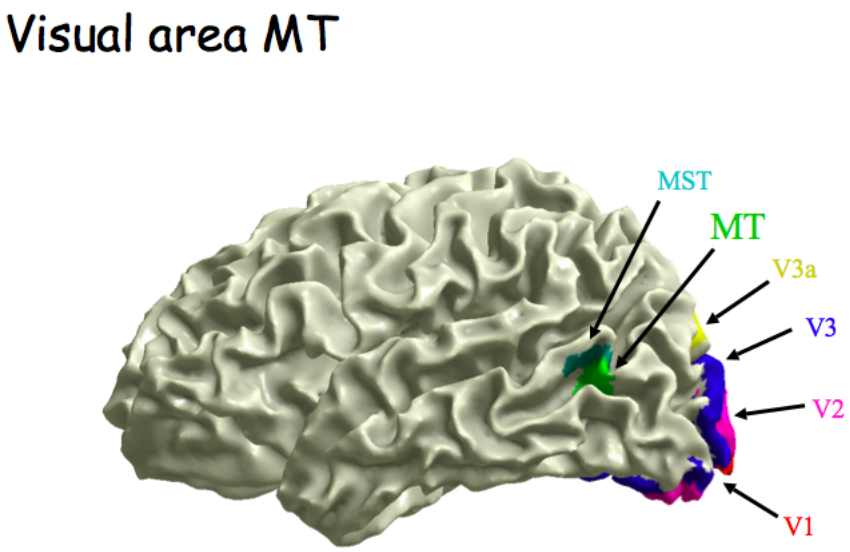


Figure 2.3: Visual area MT. The figure shows a 3D rendering of the macaque brain, with the visual areas labeled. Taken from [2].

Neurons in the middle temporal visual area (MT, also called V5, as shown in Figure 2.3) are activated by large-scale motion stimuli. Different neurons in MT respond to different directions of motion, but just as in other parts of visual cortex, area MT has a columnar structure so that clusters of neighboring neurons share receptive fields with a similar preferred direction of motion. Also, the firing rate of the MT regions depends on the amount of coherence, higher the coherence, larger is the activation of the neurons in that region.

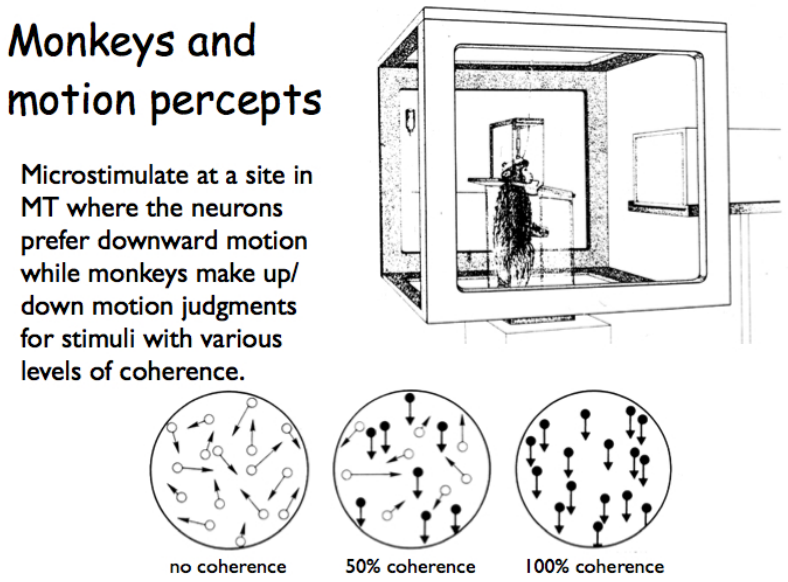


Figure 2.4: Monkey performing a motion discrimination task. Taken from [2].

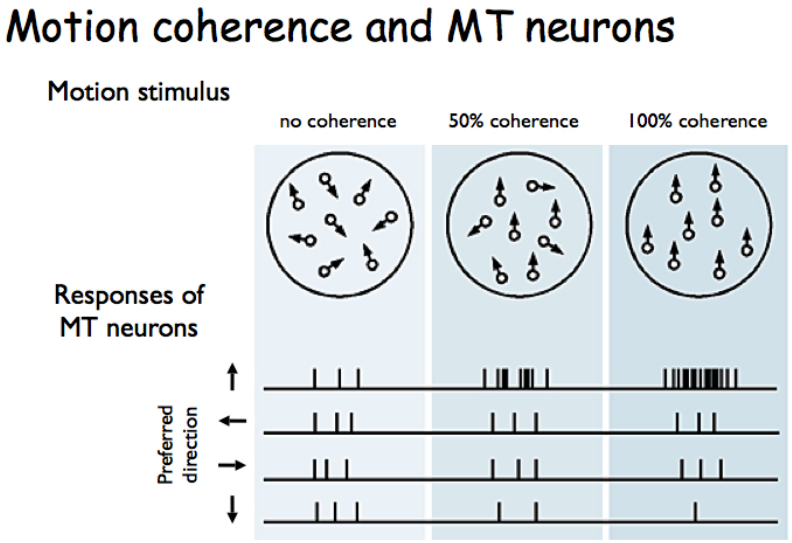


Figure 2.5: Responses of MT neurons to stimuli with different levels of motion coherence. Taken from [2].

[2] did an experiment on Monkeys on the dot motion task. The task involved looking at a streams of dots moving with a certain level of coherence in a particular direction and figure out the direction of

movement of the majority of the dots. Figure 2.4 shows a monkey performing a motion discrimination task. The monkey is seated in a chair and is looking at a screen. The screen displays a random dot kinematogram, and the monkey is required to indicate whether the dots are moving up or down. The monkey's performance is shown as a psychometric curve, which plots the percentage of correct trials as a function of the level of motion coherence. The curve shows that the monkey's performance is better at higher levels of motion coherence.

Figure 2.5 shows the responses of MT neurons to stimuli with different levels of motion coherence. The stimulus is a random dot kinematogram, in which a percentage of the dots move in a coherent direction (upward) while the remaining dots move in random directions. The responses of MT neurons are shown, with each dot representing the firing of a single neuron. The neurons are sorted by their preferred direction of motion, with neurons that prefer upward motion shown at the top of the plot and neurons that prefer downward motion shown at the bottom. The plots show that the neurons respond more strongly to stimuli with higher levels of motion coherence.

2.4 Skewed and Rare Experiences

2.4.1 Importance Sampling

Within the ever-evolving landscape of Reinforcement Learning (RL), navigating rare yet crucial experiences can be akin to searching for a needle in a haystack. Imagine trying to estimate the average income of a city by randomly interviewing pedestrians. However, if most residents are students, your sample will be heavily skewed towards low incomes. Here's where importance sampling shines. Instead of random sampling, we prioritize interviewing individuals with higher incomes (say, CEOs) who, despite being fewer, contribute more significantly to the average. We then adjust our estimates to account for this biased sampling, ensuring a more accurate representation of the true average income.

Similarly, in RL, we encounter situations where the optimal policy leads to rare, but impactful, experiences. These could be scenarios like winning a complex game or discovering a hidden treasure in an environment. Relying solely on the agent's exploration might take an eternity to encounter these events, hindering learning progress.

This is where importance sampling [17, 18] steps in. We modify the agent's policy to deliberately increase the probability of experiencing these rare events. Just like interviewing CEOs, we prioritize actions that lead to these valuable situations. Think of it as giving the agent a "flashlight" to illuminate the hidden corners of the environment.

2.4.2 Prioritized Experience Replay (PER)

However, simply increasing the occurrence of rare events isn't enough. We need to efficiently utilize the information gleaned from them. This is where Prioritized Experience Replay (PER) comes into play.

PER [18] stores the agent's experiences in a memory buffer, but not all experiences are treated equally. Experiences with higher "priority", typically those associated with rare events or large errors, are stored with a higher probability and revisited more frequently during training. This ensures that the agent focuses on learning from the most valuable experiences, accelerating its understanding of the environment and the optimal policy.

Rare experiences pose unique challenges in RL. Firstly, their stochastic nature makes them difficult to replicate consistently, hindering the agent's ability to learn from them effectively. Secondly, the agent might not even recognize the value of a rare experience until it encounters it multiple times. This is known as the "hindsight bias", where the agent overestimates the importance of an experience after knowing the outcome.

To overcome these challenges, researchers are exploring various techniques, such as:

- **Off-policy learning:** Decoupling the learning process from the exploration policy allows the agent to learn from all its experiences, regardless of how they were generated.

- **Exploration bonuses:** Rewarding the agent for venturing into uncharted territory encourages it to explore and potentially discover rare events.

- **Curiosity-driven learning:** Motivating the agent to seek out novel and informative experiences can naturally lead it to encounter rare events.

2.5 Episodic Memory

2.5.1 Hippocampus

The hippocampus (shown in Figure 2.6), nested deep within the medial temporal lobe, reigns supreme as the conductor of episodic memory. It orchestrates the complex task of encoding, storing, and retrieving past experiences, weaving together a tapestry of sights, sounds, smells, and emotions that defines our personal narrative. Within its intricate neuronal circuits, fleeting sensory inputs are transformed into durable representations, allowing us to relive the joy of a childhood birthday, navigate the familiar streets of our hometown, or recount the details of a pivotal life event.

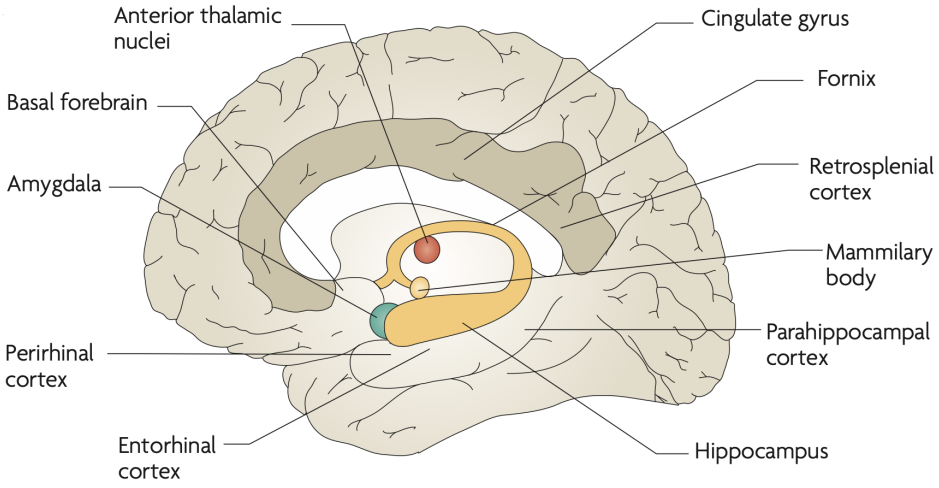


Figure 2.6: Anatomy of brain with Hippocampus marked. Taken from [3].

This remarkable property hinges on the hippocampus's unique anatomical organization. Its layered structure, with the dentate gyrus, cornu ammonis, and subiculum playing distinct roles, facilitates the intricacies of information processing. The dentate gyrus acts as a gateway, filtering and transforming incoming sensory data into sparse representations. These representations then flow through the cornu ammonis, where they are meticulously encoded and integrated with existing memories, forging rich and contextually-bound associations. Finally, the subiculum acts as a bridge, relaying these consolidated memories to the broader cortical network for storage and retrieval.

Beyond its structural elegance, the hippocampus employs a dynamic interplay of cellular mechanisms. Excitatory neurons, fueled by the neurotransmitter glutamate, fire in precise patterns, forming the building blocks of memory representations. Interneurons, meanwhile, modulate these signals, ensuring the fidelity of encoding and preventing runaway excitation. This delicate balance allows the hippocampus to capture the essence of an experience, filtering out irrelevant details while preserving the core emotional and sensory components.

In essence, the hippocampus stands as a testament to the brain's remarkable ability to weave a tapestry of past experiences. Its intricate architecture and cellular dance orchestrate the intricate processes of encoding, storage, and retrieval, ensuring that our personal journeys are not lost to the ephemeral flow of time.

2.5.2 Novel and Rare Experiences

Our memories aren't just made of ordinary things, but also special moments that stand out. These rare and unexpected experiences, whether a breathtaking sunset or a chance encounter with a fascinating stranger, leave an indelible mark on our memory and shape the very fabric of our understanding. But

how does the brain, and by extension, the nascent field of reinforcement learning (RL), navigate this landscape of the unfamiliar?

From a neuroscientific perspective, novelty acts as a potent neuromodulator, triggering the release of dopamine and other neurotransmitters that enhance learning and memory consolidation. These ”aha!” moments activate the hippocampus, the brain's maestro of episodic memory, leading to the formation of rich and vivid representations of the rare experience. Additionally, the prefrontal cortex, the orchestra conductor of executive functions, engages in heightened processing, analyzing the unexpected event and integrating it into existing knowledge frameworks.

In the realm of RL, novelty presents both a challenge and an opportunity [19]. Traditional algorithms, trained on frequent, predictable interactions, often struggle to recognize and learn from rare events. This can lead to suboptimal performance in real-world environments, where unexpected situations are inevitable. However, recent advancements in RL offer promising avenues for tackling this challenge.

One approach involves intrinsically motivated exploration [20], where agents are encouraged to actively seek out novel experiences. This can be achieved through curiosity-driven mechanisms that reward exploration for its own sake, mimicking the intrinsic drive humans have to explore and learn. Additionally, hierarchical reinforcement learning can be employed, where agents learn high-level policies that guide them towards exploring promising areas, ultimately leading them to encounter and learn from rare events.

Understanding how the brain and RL algorithms navigate the landscape of novelty is crucial for developing robust and adaptable agents that can thrive in the ever-changing world. By bridging the gap between neuroscience and RL, we can unlock the potential of these rare experiences, not just as fleeting moments, but as catalysts for learning, adaptation, and ultimately, progress.

Chapter 3

Dot Motion (Perception)

3.1 Introduction

Perceptual decision [21, 22] making process refers to selecting a course of action from a fixed number of possible alternatives based on available sensory information to a system. This information is analyzed and translated into behavior through making high-accuracy decisions at each time step. These decisions made in this process are based on statistically optimal procedures involving the integration of extracted information and further computations to form a categorical choice. The probabilistic choice of the system is investigated with variance in the presented stimulus to understand its effect on these perceptual decisions.

There has been a long line of work in modeling cognitive faculties using neural networks, but these networks do not use raw sensory data [23, 24]. In recent times, Deep Neural Networks (DNN) [25] have proved a useful model to study visual systems in neuroscience, and here we extend it and put forward a Deep Reinforcement Learning (DRL) [26] model to study the random dot motion task [27]. The visual random dot motion task is simulated to test the organism's ability to perceive coherent motion. Since a decision is made at the end of this perceptual task, this is ideally suited to be modeled using reinforcement learning. In this experiment, we adopt a naturalistic stimulus approach where the inputs to the random dot motion task are the same as those used in psychological settings. We explore if a deep reinforcement learning architecture can provide insights into how animals would be solving this paradigm of tasks. In this task, the variability in the physical dot motion from one frame to the next is directly quantified to determine the direction of the dots in the frame.

Even though the network has not been specifically trained to replicate features of the Middle Temporal (MT) [5] neural population, we find the emergence of their properties when the neural network is trained to maximize its objective of gaining rewards. The model exhibits directional selectivity corresponding to the direction of movement as well as coding its motion strength, features which are found in the MT neural population while animals do the random dot motion task. Furthermore, the model also predicts the existence of coherence selective neurons irrespective of the direction, which could form an integral component in solving such perceptual tasks.

3.2 Related Work

3.2.1 Simoncelli and Heeger Model

The model constructed by Simoncelli and Heeger [5] was designed with MT receptive field weights that were tuned to respond to a visual stimulus moving with a certain direction and speed. This computational model of the middle temporal physiology was based on previous studies that have indicated that the neurons present in the primate brain at the Middle Temporal Area were associated specifically with the velocity of the visual stimuli to the system. In the model, each complex cell response had its own space-time orientation and phase, which were computed using the simple cell afferents' weighted sum which were scattered across a local spatial region.

3.2.2 Actor-Critic Model

Framework for investigating cognitive and value-based computations has been presented by Song et al. [28]. The reinforcement learning paradigm for a long time has been used to study the cognitive and value-based computations in the brain. Expanding on the significance of including biological constraints in the training of neural networks, the value-based recurrent neural network guides the network to learn through predicting the future reward by the decisions of the network. The 'actor-critic' architecture was developed with a value network using the decision network's specified actions and activities to forecast future rewards. The baseline value network got the same inputs as that of the decision network and the selected actions, implying that they shared recurring units. The predicted reward from the system is directly tied to choice confidence, and the value network nonlinearly delivers the decision network consistent confidence information.

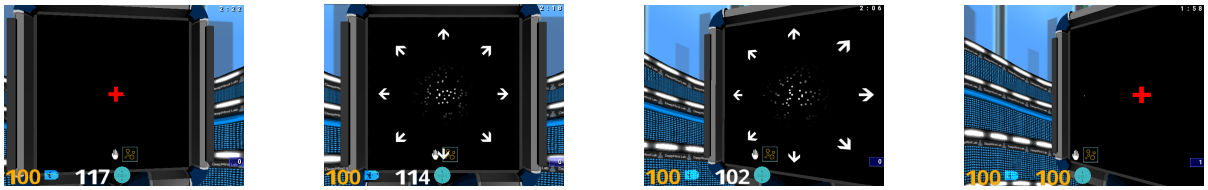
3.2.3 Law and Gold Model

Law and Gold [22] had implemented a reinforcement learning model that, on a visual motion direction discrimination task, showed improved perceptual performance. The sensory information extracted by this system imitated the representation of this information in the brain to form decisions. The model was trained over gradual trials guided by reward prediction to establish the connections between the sensory neurons and the decision-making neurons. This feedback-driven system has been employed for associative and perceptual learning tasks for both motion and response directions.

In this thesis, we have adopted a normative approach of combining Deep Reinforcement Learning and high dimensional inputs, unlike previous models, to see what kind of representations are supported when reward-related maximization is done on naturalistic stimuli.

3.3 Methods

3.3.1 Random Dot Motion Discrimination Task



(a) (b) (c) (d)

Figure 3.1: Example of a single trial in which left dot motion stimulus is presented. Images are taken from the "Random Dot Motion Discrimination" provided by Psychlab [4]. **(a)** Fixation. **(b)** Dot motion displayed. **(c)** Saccade towards the decision arrow. **(d)** The decision arrow is chosen.

Random dot motion discrimination tasks are used to investigate motion perception in the presence of noise [29, 16]. The direction in which the majority of dots are going must be determined by the subjects. To assess motion perception-related entities, the experimenter can change the percentage of dots moving coherently vs randomly. We use the "Random dot motion discrimination task" provided by Psychlab [4], which inside the first-person 3D game world of DeepMind Lab [30] is a simulated psychology laboratory.

Visualization of a trial presented with left dot motion stimulus can be seen in Figure 3.1. The task starts with a black screen with a red cross (fixation point) at the center of the experiment screen, as shown in Figure 3.1a. The eye observation point is denoted with a small white dot at the center of the screen. This observation point's position is randomly initialized somewhere close to the fixation point on the horizontal axis of dot motion. The observer is required to saccade to the fixation point for the trial to start. Once the trial starts, the dots start moving in a random direction with a random coherence value (see Figure 3.1b). $\{0\%, 6.4\%, 12.8\%, 25.6\%, 51.2\%\}$ are the coherence values that we used for the task. Eight arrows are displayed around the dot motion, denoting the eight directions in which the dots might be moving in. The observer then has to move towards that arrow and place the white dot over it to make a decision (as indicated in Figures 3.1c & 3.1d). For this work, we consider only left/right dot motions similar to primate experiments, and the Psychlab environment has been adjusted accordingly. A simple reward scheme has been used to stipulate what the agent should accomplish. A reward of +1 is given for fixation, +5 for a correct decision taken during a trial, and -5 for a wrong decision taken during a trial.

The order of coherence values of the dot motion in this task, during trials, is determined by the adaptive staircase procedure mentioned in [5, 31]. It is a sort of curriculum learning approach where the easier task of higher coherence is introduced first, and then the more difficult task of lesser coherence is introduced later. When the agent first learns, it does not finish many trials every episode, and when it does, it acts randomly. As a result, it continues to be exposed to the most basic version of the task. It

can only acquire level $c+1$ i.e lower coherence when it has learned to solve level c of higher coherence. Trials at all lesser difficulty levels remain interleaved thanks to the probing of trials. This reduces the number of trials that are available for the analysis at each of the difficulty levels while also preventing catastrophic forgetting.

Let $p_1, .., p_K$ be a series of probability distributions across trials in ascending complexity order. In our task, p_0 would be a trial-by-trial distribution of dot motion stimuli with the greatest coherence setting. A distribution across the weakest coherence stimuli would be p_K. At difficulty level c, let p_c be the distribution of trials For each new trial, do one of the following with equal probability: a) Base Case: sample a trial from p_c. b) Advance Case: sample a trial from p_{c+1} OR c) Probe Case: The difficulty level p_p is sampled first from $p_1, .., p_K$, and then the individual trial from p_p. If the agent can achieve a score of 75 percent correct or higher after c trials sampled p_{c+1} (the advance scenario), increase the basic difficulty level to $c+1$. Reduce the difficulty level to $c-1$ if the agent scored less than 50 percent correct after c trials sampled from p_c.

3.3.2 Asynchronous Advantage Actor-Critic (A3C)

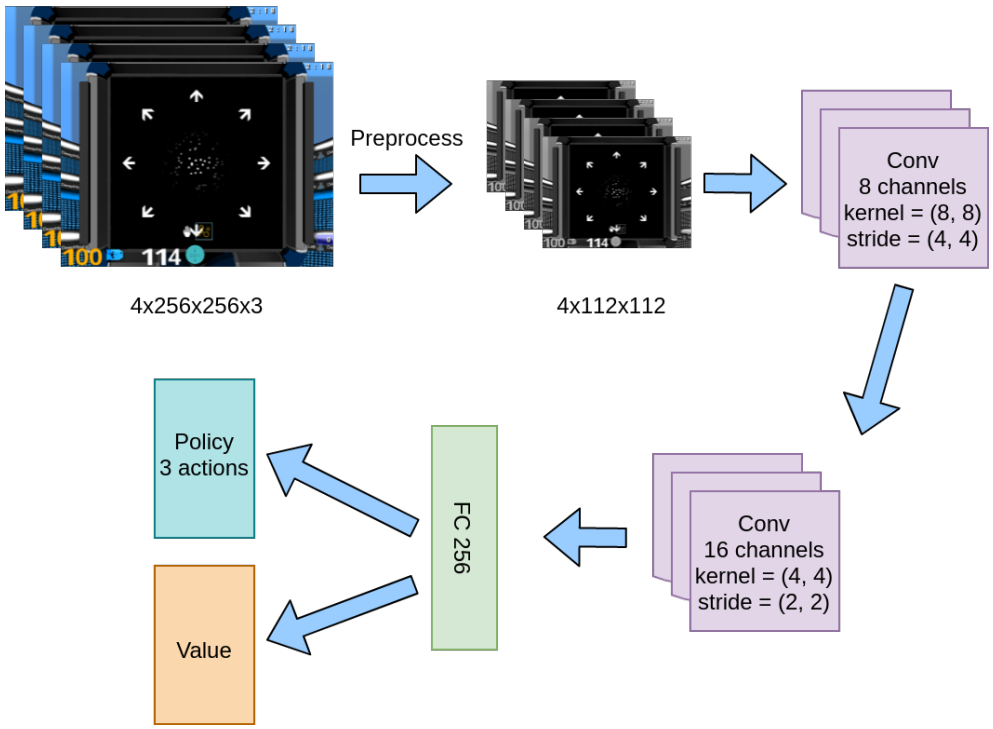


Figure 3.2: Overview of the adapted Asynchronous Advantage Actor Critic (A3C) architecture for this task.

The agent in the Reinforcement Learning framework strives to maximize its overall reward by performing actions in the environment. The agent's activities have an impact on the next state as well as

the reward that it receives from the environment. The MDP formalism is used to model this behavior which is represented as a 5-tuple $\langle \mathcal{S}, \mathcal{A}, r, p, \gamma \rangle$ where at a timestep *t*, the agent takes action $a_t \in \mathcal{A}$, in state $s_t \in \mathcal{S}$, that leads to the next state $s_{t+1} \in \mathcal{S}$ with the transition probability $p\left(\boldsymbol{s}' \mid s, a\right)$.

The agent maximizes the expected discounted reward which is defined as $R_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}$ where γ is the discount factor and also it denotes the tradeoff between future and immediate rewards. Here T is the terminal timestep. The optimal action-value function,

$$Q^*(s, a) = \max_\pi \mathbb{E}\left[R_t \mid s_t = s, a_t = a, \pi\right],$$

is the maximum discounted expected reward obtained after following policy π which is a relationship between states and actions.

This model employs the A3C [32] architecture, in which the critic learns the value function and the actor learns the policy. Unlike traditional systems such as DQN [32], which use a single agent to communicate with a single environment, A3C [33] uses several agents, each with their own copy of the environment. Each agent chooses an action a_t depending on its policy at each time step t. Under the policy π, the value function of state s_t is defined as

$$V_\pi(s_t) = E[R_t|s_t] \tag{3.1}$$

where E is the expected value. The state-action value function:

$$Q_\pi(s_t, a_t) = E[r_{t+1} + \gamma V_\pi(s_{t+1})|s_t, a_t] \tag{3.2}$$

where, r_t is the reward at time step t, as part of the advantage function:

$$A_\pi(s_t, a_t) = Q_\pi(s_t, a_t) - V_\pi(s_t) \tag{3.3}$$

tells how good it is in a state s_t to take action a_t with respect to the average. The quality of the policy π is determined by the following objective function:

$$J(\theta) = E[V_\pi(s_o)] \tag{3.4}$$

where θ are the parameters of the policy function. The loss function to be minimized is given by:

$$L(\theta, \theta_v) = -log\pi(a_t|s_t, \theta)(R_t - V(s_t, \theta_v)) - \beta H(\pi(a_t|s_t, \theta)) \tag{3.5}$$

where θ and θ_v are the parameters of the policy and value functions, respectively. H represents the entropy of the policy.

3.3.3 Model Architecture

We use the A3C architecture to teach our agents how to learn this task. The overview of this model architecture can be seen in Figure 3.2. We work directly with raw frames, which are 256x256 RGB images. During preprocessing, we convert this 256x256x3 sized image into a downscaled grayscale image

of dimension 112x112. Inspired by the frame stacking technique used in [34], we stack four consecutive frames and treat them as an observation. This is very important, as tasks like this require not just the pixel information but also the movement information of these pixels. The final input (observation) to our model is a vector of 4x112x112 dimension.

Parameter	**Value**
Learning rate	0.0001
Gamma (discount factor)	0.99
Beta (β)	0.01
Lambda (λ)	0.995
# of training agents	10
Size of FC layer	256
Episode length	20s
Frames stacked	4

Table 3.1: Parameters for the optimal neural network training.

The first 2d convolutional layer convolves 8 (8x8) filters with a stride of 4 over the input image of dimension 4x112x112. This output is followed by a ReLU activation [35, 36]. This intermediary output is then fed to another 2d convolutional layer that convolves 16 (4x4) filters with a stride of 2, which is then passed through a ReLU activation. The output is then flattened to get a vector of size 2304. This 2304-sized vector is passed through a linear layer with 256 units, followed by ReLU again. The actor and critic layers then use this as the input and give as the output the policy and value, respectively. The policy network predicts 3 actions, namely - no operation, left and right. The different hyperparameter values of the model can be seen in Table 3.1. In A3C, a smoothing parameter called lambda is utilized to make training more stable by lowering its variance. We varied this value in the bucket [0.9, 1.0] and found 0.995 to be the optimal value that provides the agent with the benefit of taking action both in the long and short term.

3.4 Results

3.4.1 Performance Variation with Coherence

From Figure 3.3, it is evident that the agent's performance highly varies with the coherence of the dot motion stimuli presented. It is observed that the agent is able to collect more rewards for higher coherence stimuli, and also performs them accurately, leading to stronger confidence. The percentage of trials done correctly by the agent for 0% coherence is just 50%, implying that the agent's decisions

are mostly chance-based and lack any performance-based decision-making. This finding reveals that the agent's performance increases with increasing coherence and decreases with decreasing coherence.

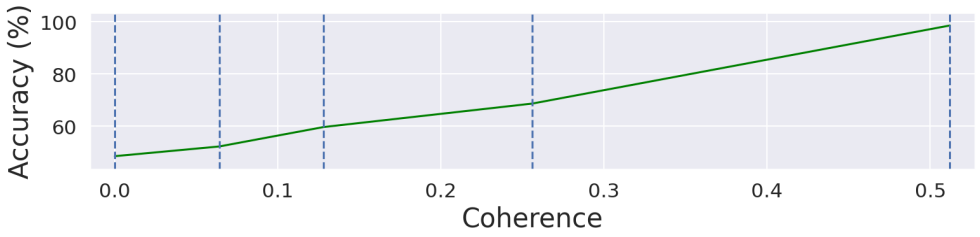


Figure 3.3: Performance curve of the model for different motion strengths. Depicts the percentage of trials done correctly during an episode for set coherence values.

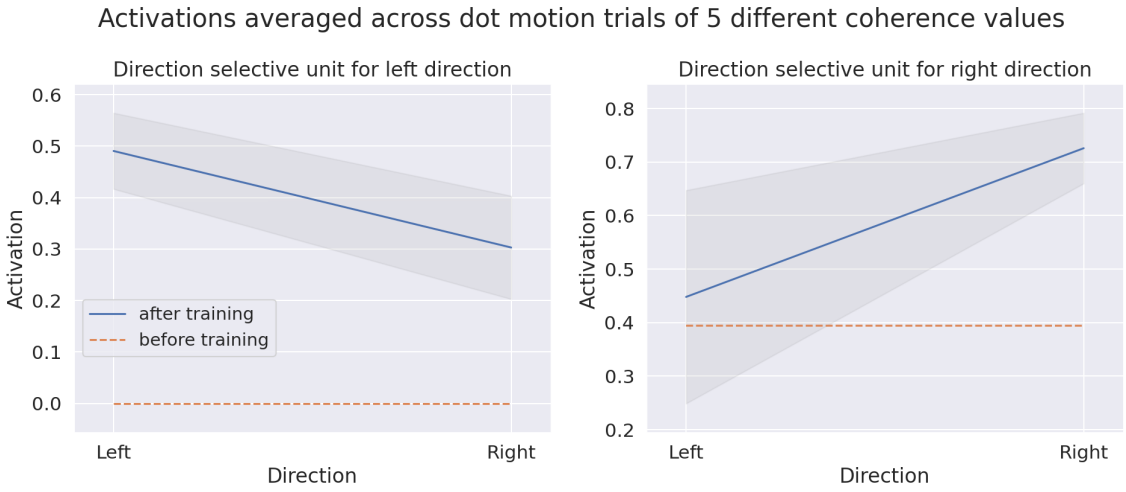


Figure 3.4: Response of direction-selective neurons from the last convolutional layer of the model (MT region) averaged across approximately 67 trials. The standard deviation is shown in light blue color around the mean. Around 14.58% units in this layer were found to be left-direction-selective, and 15.15% were right-direction-selective. The left and right figures show the activation of neurons that prefer the left and right directions, respectively. The dotted line represents the activation of the same neuron before the model was trained to do the task. We can see an increase in the mean response of the unit and also see a decrease in variance for the preferred direction.

The data further corroborates that the agent's performance can be improved by providing more coherent stimuli. When compared to randomness, the agent's accuracy in performing tasks relatively increases with higher levels of coherence. This implies that agents are highly sensitive to environmental changes and can leverage their performance based on such changes. To test this further, experiments

involving varying levels of coherence can be conducted to observe how agents respond under certain conditions. In general, these findings provide evidence that agents are sensitive to their environment and can optimize their performance based on changes in coherence.

3.4.2 Emergence of Direction Selectivity

Before Training				
	Direction Selectivity			
	Left		Right	
	Trial Direction			
Coherence (%)	Left	Right	Left	Right
0.0	0.73	0.73	0.30	0.30
6.4	0.73	0.73	0.30	0.30
12.8	0.73	0.73	0.30	0.30
25.6	0.73	0.73	0.30	0.30
51.2	0.73	0.73	0.30	0.30

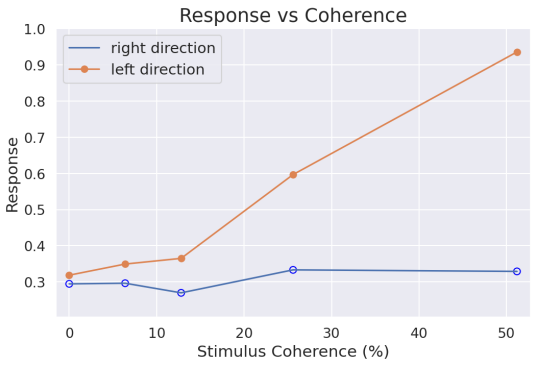
After Training				
	Direction Selectivity			
	Left		Right	
	Trial Direction			
Coherence (%)	Left	Right	Left	Right
0.0	0.32	0.29	0.49	0.65
6.4	0.35	0.29	0.47	0.69
12.8	0.41	0.26	0.48	0.73
25.6	0.60	0.30	0.47	0.80
51.2	0.94	0.28	0.49	0.93

Table 3.2: Direction selectivity for different coherence values. Computed by averaging the response of the units across approximately 67 trials. Contains activation during left and right trials for left and right direction-selective neurons. We can see the difference in activation for the left and right directions, showing a preference for the direction. Graded firing is being shown for dot motion stimuli of different coherence strengths when the motion strength is aligned with the preferred direction of the unit. The left and right tables contain the activations before and after training, respectively.

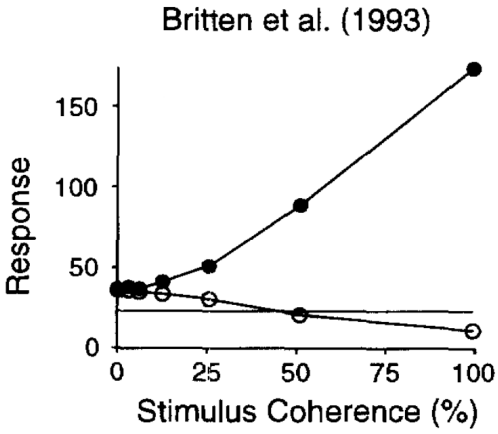
Even though the network has not been explicitly trained to have direction-selective neurons, there is an emergence of direction selectivity corresponding to the direction chosen in the task, as shown in figures 3.4 and 3.5. The units visualized were taken from the last convolutional layer of the model. The left figure in figure 3.4 shows a left direction-selective unit, and we can clearly see a higher response for the preferred direction. The right figure, on the other hand, shows a similar property but is selective towards the motion direction in the opposite direction. Furthermore, this direction selectivity is not found in untrained neural networks, as shown using dotted lines. The response of the units is almost constant no matter what direction the dots are moving in. Also, this phenomenon arises purely through training, which indicates that representations of this kind are critical to completing the dot motion task.

The presence of direction selectivity is a key feature of representations that aid in completing specific tasks such as the dot motion task. It is evident that the trained network is able to respond differently depending on the direction of motion, while an untrained network struggles to do so successfully. Therefore, it is critical to comprehend and consider how representations affect performance when designing powerful AI systems.

3.4.3 Graded Firing Corresponding to Coherence



(a) Direction selective unit that's active only for left direction.



(b) MT response as a function of motion signal strength, taken from figure 11 of [5].

Figure 3.5: (a) Graded firing of units for dot motion of different strength averaged across approximately 67 trials. The blue line shows the activation for motion during the right trial and the orange line for the left trial. Symbols that are filled correspond to movement in the preferred direction. Open symbols represent movement in the anti-preferred (opposite) direction. (b) An MT neuron's response as a function of the percentage of dots moving coherently. (re-plotted from Britten et al., 1993 [5]). Note: We only plot response till the coherence value of 51.2%

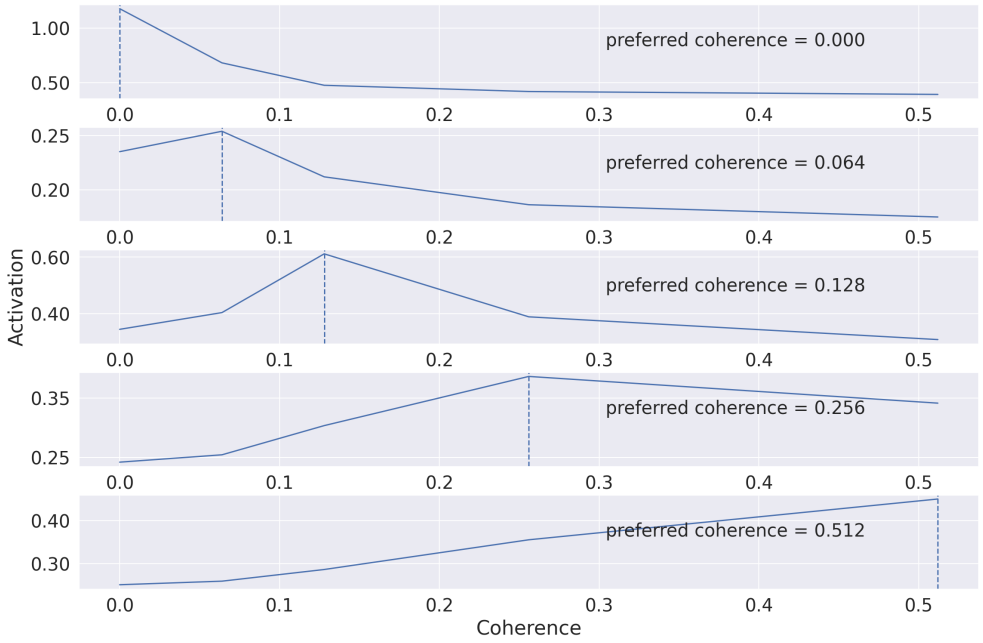


Figure 3.6: Preferred coherence plot computed by averaging unit responses across approximately 67 trials. Top to Bottom: preferred coherence plots in increasing order of coherence of the dot motion stimuli. We can see a higher response for the preferred coherence value of the unit.

The model shows a clear correlation between the coherence of MT neurons and the graded firing. As seen in Table 3.2 and Figure 3.5, the stronger the motion of the stimulus, the stronger the response of the model. This is further evidenced by the activation of the direction-selective units, as seen in Table 3.2 and Figure 3.5a. The left direction-selective unit shows higher activation for higher coherence stimuli and the same can be seen for the right direction unit. These findings are consistent with previous research, as seen in [5].

It is interesting to note that the activations shown in the top part of Table 3.2 are the same, regardless of the direction in which the dots are moving. This highlights the importance of the learned representations for successfully completing the dot motion task. The model's ability to effectively replicate the graded firing of MT neurons suggests that it has a good understanding of the underlying neural mechanisms involved in the processing of visual motion.

Additionally, the model's ability to show direction-selectivity highlights its capability to simulate the properties of MT neurons accurately. These properties play a crucial role in the visual perception of motion and the model's replication of them is a testament to its validity as a tool for studying the visual system. In conclusion, the model's replication of the graded firing and direction-selectivity of MT neurons highlights its usefulness as a tool for understanding the neural mechanisms involved in visual perception. The model's results provide further support for the role of MT neurons in the processing of

visual motion and reinforce the importance of accurate representations for the successful completion of the dot motion task.

3.4.4 Coherence Selective Neurons

The results from the study show a clear preference for certain coherence values in some units of the network, as shown in Figure 3.6. This can be seen in the activation patterns of these units, with higher firing observed for stimuli with a preferred coherence value. However, when the coherence value deviates from the preferred value, the response of these units declines. It is important to note that this effect was observed even when the coherence was set to 0. This suggests that the neural network encodes some neurons in such a way that they have a high default activation, which gradually declines as the coherence is increased. This provides insight into the encoding strategy used by the network and highlights the importance of these units in solving the task efficiently.

These results also demonstrate the network's ability to predict the perceived coherence of the stimuli. By encoding neurons in such a way that they have a high activation for preferred coherence values and declining activation for deviating coherence values, the network is able to solve the task more effectively. It is worth mentioning that these findings align with the previously established role of MT neurons in the processing of visual motion. The network's ability to replicate this effect highlights the validity of the model as a tool for studying the neural mechanisms involved in visual perception. In conclusion, the preference for certain coherence values in some units of the network provides insight into the encoding strategies used by the network and highlights the importance of these units in solving the dot motion task efficiently.

3.5 Discussion

The model, even though it has only been specifically trained to maximize rewards, exhibits several properties of MT neurons. The model exhibits direction selectivity in the last CNN layer, and these direction-selective neurons showed graded firing for different coherence values indicating that these neurons code for motion strength. The model also makes a specific testable prediction that there also could exist a subset of neurons tuned for coherence selection in the MT population. It could imply that the coding of coherence selectivity is essential for motion processing, and extracting such statistics on coherence helps perceptual decision-making.

The model uses a naturalistic stimulus setting on the random dot motion task. The motivation for adopting such stimuli was to find if there would be an emergence of MT-like representations in the neural network trained to optimize rewards. The emergence of direction-selective neurons and neurons coding for motion strength suggest that these representations are critical in perceptual decision-making and indicate convergence with the brain brought through by evolution. This model uses a Deep RL

approach, which could also provide an avenue to further study representations in the brain similar to supervised learning.

3.6 Conclusion

The current work uses a feed-forward neural network approach with reward-based feedback to find similar representations in the random dot-moving task in the MT neural population. This could provide a new avenue for training other psychological experiments in an end-to-end manner using naturalistic stimuli to discover how the brain deals with complex high-dimensional stimuli and make decisions using Deep Reinforcement Learning. We hope to extend this work by finding links between psychophysical manipulations of this task and finding their corresponding neural signature to figure out what representations are being used to perform this task, for example, manipulating the speed of the dot motion or the numerosity, and finding out how it influences perceptual decision making.

Chapter 4

Learning from Zipfian Environments (Episodic Memory)

4.1 Introduction

Navigating the complexities of real-world environments presents a formidable challenge for Reinforcement Learning (RL) algorithms. Unlike the neatly structured and statistically balanced scenarios often employed in theoretical settings, real-world data distributions are characterized by long-tailed phenomena. Here, a small subset of states dominates the data landscape, while a vast majority of crucial experiences occur only infrequently [13, 7]. This characteristic poses a significant hurdle for conventional RL algorithms, leading to:

- **Sample Inefficiency:** Frequent states overwhelm the learning process, rendering rare encounters statistically insignificant and hindering their effective exploration and exploitation.

- **Learning Bias:** Biases emerge towards high-frequency states, neglecting the potentially critical information contained within infrequent encounters.

- **Generalization Challenges:** Learned policies struggle to generalize effectively to unseen or underrepresented states, limiting their real-world applicability.

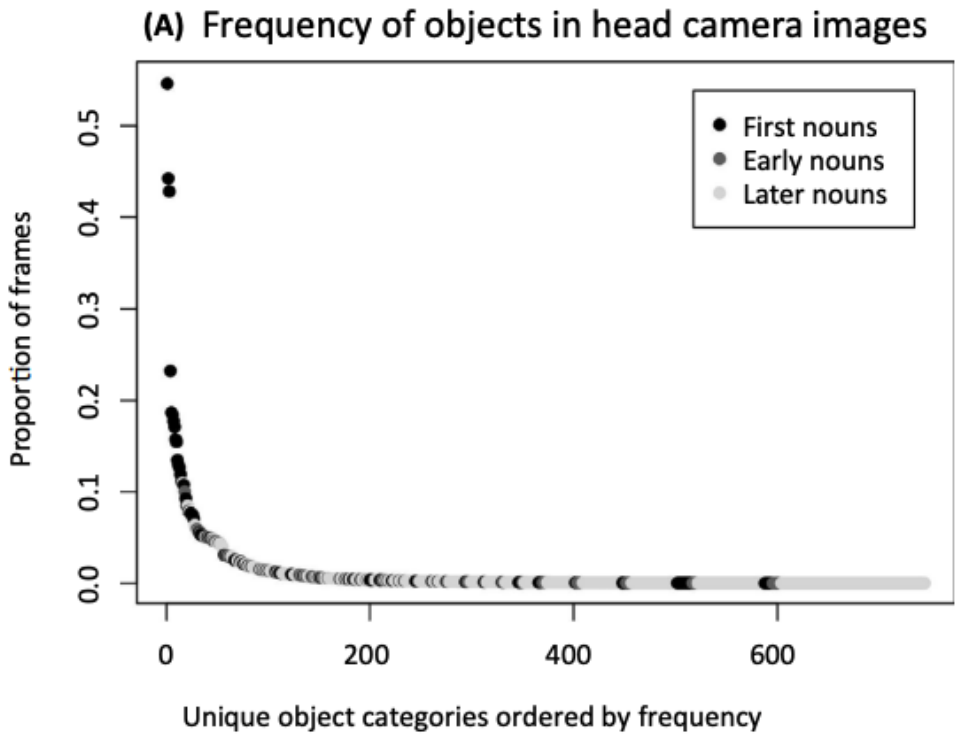


Figure 4.1: Studies using head cameras on 8-10 month old infants reveal a fascinating phenomenon: the frequency of objects they encounter follows a long-tailed distribution. Cups, spoons, and bowls appear far more often than other items, shaping their early learning and highlighting the importance of understanding imbalanced data in cognitive development. Taken from [6] and has been reproduced from [7].

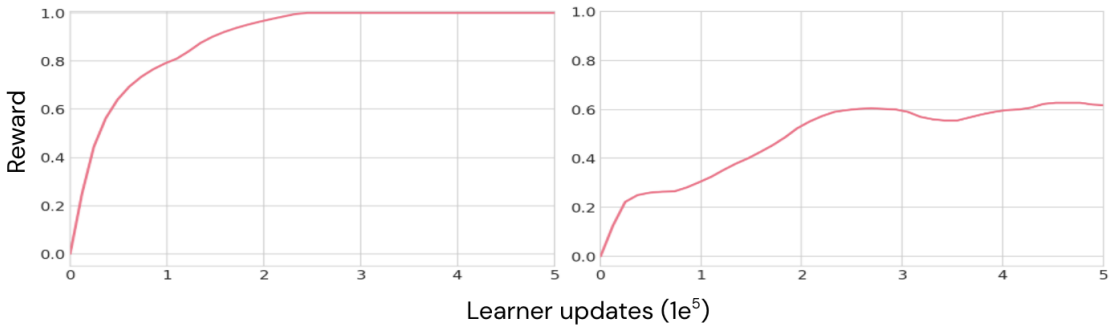


Figure 4.2: An agent trained with the IMPALA algorithm on in a Zipfian environment (Zipf's Playroom) [6] with 50 objects. **Left** evaluated on all objects presented according to the training distribution **Right** on the rarest 20% of objects in the training distribution. Taken from [6].

Imagine a deer foraging in a forest filled with abundant resources but also harboring lurking predators. Encountering a predator while drinking from a specific water source is a rare yet critical event. Learning from this infrequent experience and generalizing it to similar situations is paramount for the deer's survival. Similarly, in autonomous driving, the rare instances of accidents or unusual traffic patterns hold invaluable insights for safety and robust navigation. We can see in Figure 4.1 a real world example where the image shows that the most frequent object categories are present in a high proportion of frames, while the less frequent object categories are present in a low proportion of frames. The results of training a RL agent (IMPALA [8]) on such a setting can be seen in Figure 4.2.

4.1.1 Memory Systems in RL

Memory systems in humans allow them to retrieve the relevant set of experiences for decision-making in case of unseen circumstances. In neuroscience, some of the types of memories studied are – *Working Memory and Episodic Memory*. Working memory is short-term temporary storage while episodic memory is a non-parametric or semi-parametric long-term storage memory. Deep Reinforcement Learning agents with episodic memory, in particular, a combination of non-parametric and parametric networks have shown improved sample efficiency and are suitable for decision making in rare events. [37] used a non-parametric model to keep the best Q-values in tabular memory. [11] in Neural Episodic Control proposed a differentiable-neural-dictionary to keep the representations and Q-values in a semi-tabular form. [38] took up a trajectory-centric approach to model such systems. This thesis looks up to [39]'s state-centric formulation of an Episodic Memory (MEM) which implements working memory using a latent recurrent neural network and an episodic memory.

This thesis delves into the "Momentum Boosted Episodic Memory", which proposes a novel architecture for tackling the challenges of long-tailed RL settings. Inspired by the complementary learning systems framework in neuroscience [40, 14], the paper posits that efficient learning in long-tailed environments necessitates the interplay between a fast learning system and a slow learning system.

- **Fast Learning System:** We leverage the capabilities of a deep RL algorithm equipped with an episodic buffer for rapid generalization across experiences. This allows the agent to quickly adapt to frequently encountered states and build an initial understanding of the environment.

- **Slow Learning System:** A familiarity memory module employing a contrastive momentum loss acts as the slow learning system. This mechanism identifies rare states based on their recurrent activation within the episodic buffer, effectively prioritizing their valuable information content amidst the dominance of frequent encounters [41].

These prioritized rare state experiences are then retained for a longer duration in the buffer, enabling further analysis and learning. Crucially, the hidden activations corresponding to these rare states are also reinstated within the recurrent layers of the RL network. This novel technique leverages the insights gleaned from infrequent experiences to influence the future learning process, promoting a more balanced and informative representation of the environment.

4.1.2 Main Contributions:

- **Pioneering Solution for Long-Tailed RL Navigation:** This work presents the first dedicated solution for navigating to objects with a long-tailed distribution using deep RL. This opens doors for tackling real-world tasks characterized by significant data scarcity and imbalanced state occurrences.

- **Unsupervised Long-Tail State Discovery:** The application of contrastive momentum loss within the familiarity memory module provides a unique approach for automatically identifying rare states without requiring additional supervision. This significantly simplifies the learning process and allows the agent to autonomously prioritize valuable information within the long-tailed data distribution.

- **Prioritization and Reinstatement for Efficient Learning:** The innovative combination of prioritizing rare states in the buffer and reinstating their corresponding hidden activations fosters a more efficient learning process. This enables the agent to effectively leverage the insights from infrequent experiences, leading to a more robust and generalizable policy.

By delving into the theoretical foundations of Long-Tailed RL, meticulously dissecting the "Momentum Boosted Episodic Memory" architecture, and conducting comprehensive empirical evaluations, this thesis aims to shed light on the following key questions:

- **Effectiveness of the proposed framework in addressing the challenges of long-tailed RL environments:** Can the architecture effectively overcome the limitations of sample inefficiency, learning bias, and generalization difficulties often encountered in long-tailed settings?

- **Contribution of individual components to overall performance:** How do the fast learning system, familiarity memory module, and reinstatement technique each contribute to the overall learning efficiency and policy performance?

- **Generalizability across diverse Long-Tailed RL problems:** Can the proposed approach be effectively applied to various Long-Tailed RL tasks beyond the specific domain explored in the original paper?

- **Potential limitations and avenues for further improvement:** What are the potential limitations of the current framework, and what avenues can be explored for further enhancement and refinement?

Through a rigorous investigation of the "Momentum Boosted Episodic Memory" architecture, this thesis endeavors to illuminate the path towards effective learning from scarcity in RL environments. This advancement holds significant implications for empowering RL agents to tackle the complexities of the real world.

4.2 Benchmarks

4.2.1 Zipf's Gridworld

The Zipf's Gridworld task introduced in [6] was investigated, which presents mul tiple distinct tasks consisting of skewed data distributions along various dimensions that challenge conventional architectures to generalize to rare states and events. The task (as shown in Figure 4.3) presents the agent with

a 2D map containing multiple rooms and diverse objects. At the top-left corner of its visual input, the agent receives a visual cue depicting the target object it must find. A fixed set of 20 maps, each with 9 rooms and 20 objects, ensures consistency across trials. Importantly, start locations, object shapes, colors, and locations remain fixed within each map.

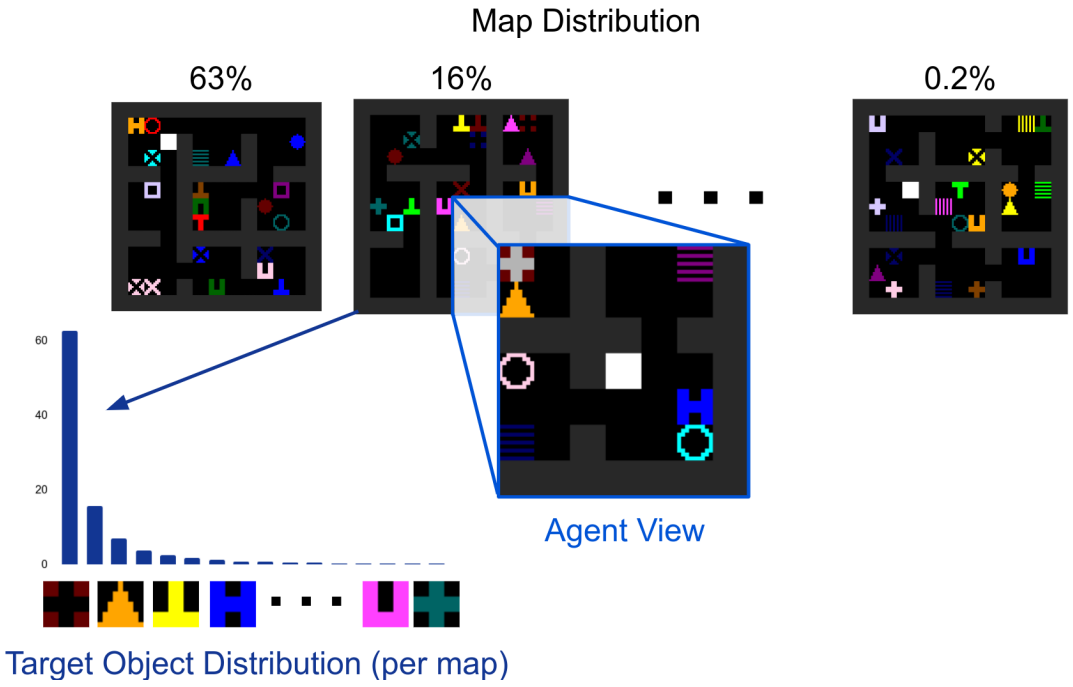


Figure 4.3: **Zipf's Gridworld:** This 2D environment presents a double dose of challenge: hierarchical skew in both map selection and target object choice. Each episode, the agent (white square) navigates a map chosen from a Zipfian distribution across 20 possibilities (top). Within each map, objects always start in the same locations, and one is designated the target, again following a Zipfian distribution (bottom left). Local observations (bottom right) guide the agent, with the target highlighted in the top left corner (here, the red squares against the light grey background).

$$zp(k, n, e) = \frac{1/k^e}{\sum_{i=1}^{n}(1/i^e)} \tag{4.1}$$

Each episode begins with the agent receiving a visual cue for a specific target object (Figure). Positive rewards are granted solely for reaching the intended object; any interactions with other objects result in no reward and episode termination. The probability of occurrence of the maps is governed by Zipf's power law (Equation 4.1), where n is the number of maps, k is the map index $(1 <= k <= n)$ and e is the Zipf's exponent. While object and location configurations are static within maps, both map selection and target object designation follow a Zipfian distribution (e = 2). This introduces inherent data imbalance, with some maps and objects encountered significantly more frequently than others.

Hence, Zipf's Gridworld challenges agents to generalize from hierarchically skewed experiences to uniform distributions over both maps and target objects. This includes testing their ability to adapt to the rarest 20% of data points, encompassing the four rarest objects within the four least frequently encountered maps. By evaluating performance on this challenging spectrum of data frequencies, Zipf's Gridworld provides valuable insights into an agent's generalizability and robustness in the face of long-tailed experience distributions. Instead of (20 maps, 20 objects), the experiments were done on (10 maps, 10 objects) which is equally challenging due to the reasons mentioned below:

1. Reducing the number of maps and objects doesn't make the task easier, but might be more difficult because we train the agent using just 50 actors and for 4e7 steps due to computational constraints.

2. Not able to reproduce the exact results on 20 maps and 20 objects using our implementation of IMPALA and the original code for training on Deepmind's environments is not publicly available due to some issues on their side, at the time of experimentation and writing this thesis .

4.2.2 Zipf's Labyrinth

While "Zipf's Gridworld" delves into object identification and manipulation under long-tailed data, "Zipf's Labyrinth" (Figure 4.4) tackles a different challenge: heavily skewed experiences of tasks and situations within specific goals. To craft this intricate environment, they leverage the existing DM-Lab benchmark [30], a diverse collection of 30 distinct tasks set in a 3D, first-person environment built on Quake 3 Arena.

The key twist lies in rebalancing the task exposure during training using a Zipfian distribution (exponent 1). This distribution assigns probabilities to each task, with some encountered much more frequently than others, mimicking real-world data scarcity. Importantly, we base this Zipfian distribution on the original ordering of tasks, which are grouped by type (e.g., navigation, manipulation, resource collection). This potentially amplifies the difficulty, as entire clusters of similar tasks may become rare or common, further straining the agent's generalizability across diverse skills.

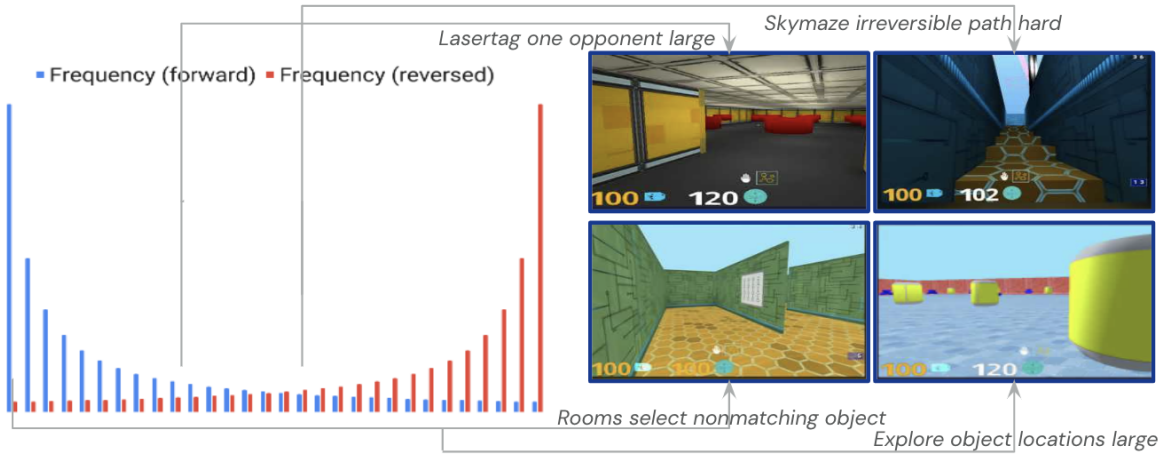


Figure 4.4: **Left** the frequency with which the learner experiences each of the DM-Lab 30 tasks in Zipf's Labyrinth. **Right** Frames of several tasks from the agent's perspective.

To ensure the findings are not an artifact of this specific ordering, they adopt a two-pronged approach. Agents are trained on Zipfian distributions with the skew either descending across tasks ("forward Zipf") or ascending across them ("reversed Zipf"). This allows to analyze the impact of data skewness patterns on agent performance.

Evaluating the agent's success is multifaceted. They assess its performance across all tasks sampled uniformly, providing a holistic view of generalizability. However, the true test lies in its ability to excel on the rarest 20% of tasks (six tasks), representing the crucial yet infrequent situations it may encounter in the real world. By analyzing performance across these diverse data frequencies, we gain valuable insights into the agent's robustness and adaptability in the face of long-tailed task distributions.

Zipf's Labyrinth, therefore, offers a powerful tool for unlocking the secrets of generalizability in complex, real-world scenarios. By studying how agents navigate this labyrinth of skewed task exposure, we can push the boundaries of artificial intelligence and empower agents to learn, adapt, and excel even when information is scarce and unevenly distributed.

4.2.3 Zipf's 3DWorld

Drawing inspiration from the Zipf's Gridworld task [6], this thesis presents Zipf's 3DWorld task, a challenging 3D environment designed to assess an agent's ability to navigate in the face of partial observability and skewed data distributions. This 3D benchmark differentiates itself from its predecessor by offering a more complex and realistic setting, pushing the boundaries of generalizable reinforcement learning.

4.2.3.1 A Multifaceted Challenge:

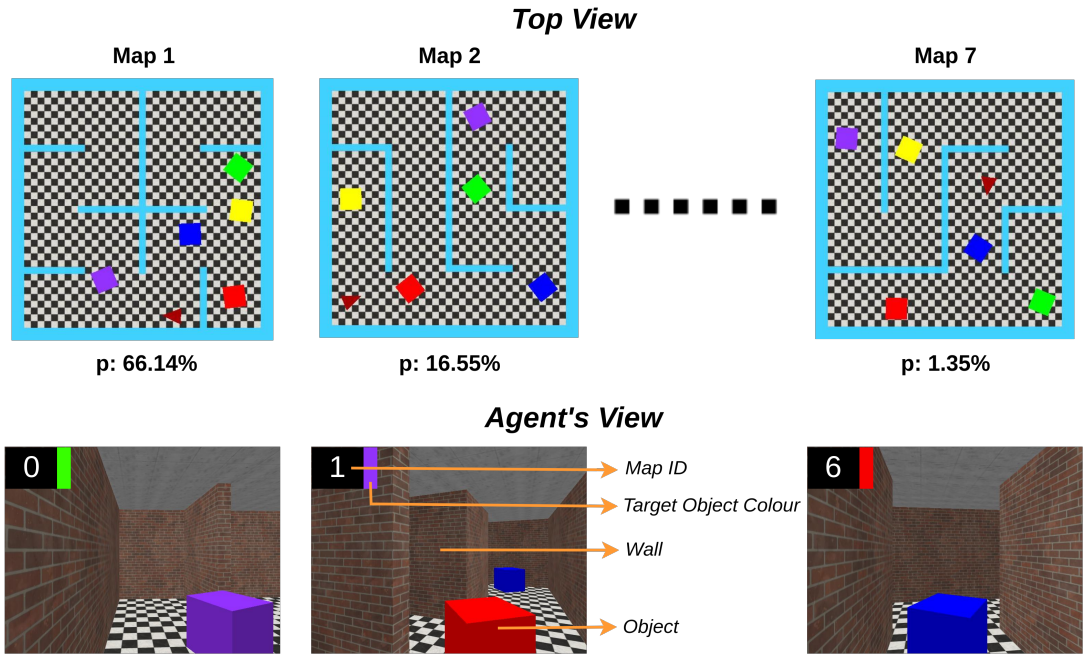


Figure 4.5: **Zipf's 3DWorld Task:** Contains 7 maps, each with 5 objects placed at random locations. The location of these objects does not change during trials. The agent (Red triangle in top view) starts at a fixed location in each trial and has to navigate towards the target object, whose color is shown in the top-left corner along with the current map ID (0 indexed). The agent's first-person view of each map is shown in the bottom images. The details of the environment experienced can be seen in the annotated image. The value 'p' below each map shows the probability of occurrence of the map in a trial, highlighting the skew in the distribution. A similar skew occurs for the distribution of objects in these maps. We can see this in Figure 4.6, which shows the distribution of objects for the first map (most common).

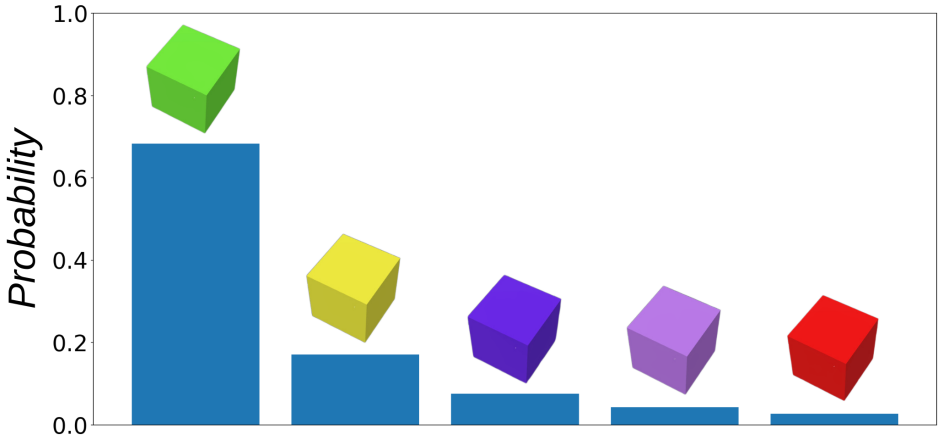


Figure 4.6: The probability distribution for objects to appear as the target object in a map during a trial. This example shows the distribution of objects for Map 1 in Figure 4.5.

Zipf's 3DWorld (Figure 4.5) comprises seven distinct maps, each presenting a unique layout and spatial arrangement. Within each map, five uniquely identifiable objects, represented by colored square boxes, are randomly positioned. The agent, starting from a fixed location, must then navigate the partially observable environment, relying solely on its immediate visual input and past experiences to locate a specific target object within a limited number of steps (200).

4.2.3.2 Partial Obscurity and Informative Cues:

The agent's visual field provides only a glimpse of its surroundings. This partial observability mimics real-world scenarios where complete information is rarely available, forcing the agent to rely on memory, exploration, and efficient decision-making to succeed. However, crucial navigational cues are embedded within the agent's visual input. The upper left corner displays the target object's color, highlighting its key distinguishing feature. Additionally, the map identifier adjacent to the color cue facilitates map-specific learning and context-aware navigation.

4.2.3.3 Action Space and Object Interactions:

The agent possesses a well-defined action space, enabling it to move forward and backward, turn left and right, and ultimately interact with objects by picking them up. This simple yet effective action set allows the agent to explore the environment extensively and engage with its constituent elements. Notably, all objects within a map are distinct, ensuring that the agent can reach any object using the available set of five actions, eliminating ambiguities and focusing attention on efficient navigation and target identification.

4.2.3.4 Evaluation Metrics and Generalizability:

The agent's performance in Zipf's 3DWorld is evaluated based on its success in identifying the correct target object within the step limit. Reaching the target object yields a positive reward, while incorrect choices or exceeding the step limit result in no reward. However, the true test of generalizability lies in the agent's ability to consistently perform well across all seven maps, even for the target objects encountered less frequently. By introducing a Zipfian distribution to the map selection during training, we can ensure that some maps and their corresponding target objects become significantly rarer than others, mimicking real-world data scarcity and challenging the agent's adaptability (skew as show in Figure 4.6).

4.2.3.5 Beyond Gridworlds: A Bridge to Real-World Challenges:

Zipf's 3DWorld bridges the gap between the simplicity of grid-world navigation tasks and the complexities of real-world scenariose. The 3D environment, partial observability, and skewed data distribution present a demanding yet realistic challenge for RL agents, pushing them to develop generalizable and efficient learning strategies. By analyzing their performance in this controlled environment, we gain valuable insights into their capability to handle data scarcity, navigate with limited information, and adapt to unseen situations. Ultimately, Zipf's 3DWorld serves as a stepping stone towards the development of robust and reliable AI systems that can operate effectively in the often-uncertain and information-scarce real world.

4.3 Model Architecture

Given an image observation (im), IMPALA's [8] feature extractor gives a pixel input embedding (p) which is further passed to an LSTM network with the hidden state (h) to get the new hidden state, policy, and value. As done in [39], we have a MEM module (orange-red buffer in Figure 4.8) that is used to find the memory (m), which is then additionally fed as input to the LSTM network along with the pixel input embedding p. We only keep rare states in the MEM by introducing an additional 'familiarity' buffer (light cyan buffer in Figure 4.8) that uses boosted contrastive learning to prioritize and filter rare states for storing in MEM.

4.3.1 State familiarity using Boosted Contrastive Learning

A 'familiarity' buffer is a circular buffer that contains states over which we can get a prediction on whether a state is rare or how rare the states are. To achieve this, we take inspiration from [41], where they improve performance on a long-tailed self-supervised learning task by proposing a momentum loss that can predict which samples among the dataset are long-tail samples.

The 'familiarity' buffer (light cyan buffer in Figure 4.8) contains the input image (im), key (k), pixel input embedding (p), and LSTM hidden state (h). We will define these terms in the following section. In each learning step of IMPALA [8], the batch of trajectories is sent to this buffer. The entire trajectory isn't sent to the buffer, but a subset of it (states at a hop of hp from the start of the trajectory) is sent. For a trajectory $\mathrm{Tr} = (s_1, s_2, ..., s_k)$ of size k consisting of states s_i, the subset of trajectory is defined as

$$\mathrm{Tr}_{\mathrm{subset}} = \left(s_1, s_{(1+hp)}, ..., s_{\left(1+\lfloor \frac{k-1}{hp} \rfloor * hp\right)} \right) \tag{4.2}$$

The same feature extractor of IMPALA is trained using an additional auxiliary contrastive loss (Equation 4.6) on states present in the familiarity buffer. Once the circular buffer is full of states, the contrastive learning process starts to help the learning process. For each image sample im_i in the buffer, we find its pixel input embedding p_i after IMPALA's feature extractor. The same feature extractor is used on the augmented image to get another embedding p_i^{aug}. The image im_i is augmented using two augmentations, namely gaussian noise [42] and random cutout [43] as shown in Figure 4.7. We use these two augmentations because they are simple and don't affect the task. For example, using something like random conv [44] will change the color of all objects in the observation, and this might result in the agent confusing it to be an image (observation) from some other trial.

For adding gaussian noise (Figure 4.7) to the image, we first generate an image of the same dimension as that of the original image that is filled with random numbers from $N(0, \sigma^2)$. This image is then added to the original image after amplifying by a factor of σ. For random cutout (Figure 4.7), we take a random location in the image and cut a rectangular area of random size. By cutting, we mean replacing the pixels in that rectangular region with black pixels.

During the training process, we consider the embeddings p_i and p_i^{aug} as positive pairs for contrastive learning. The NT-Xent loss ([45], Equation 4.6) is used to calculate the loss per sample. For each sample i in the circular buffer, we track its momentum loss following [41]. The momentum loss assists in knowing which samples in the circular buffer are long-tail samples. For a sample i, if for T consecutive epochs the contrastive losses are $\{\ell_{i,1}^T, \ell_{i,2}^T, ..., \ell_{i,T}^T\}$, then the moving average momentum loss is defined as follows:

$$\ell m_{i,1}^m = \ell_{i,1}^T; \;\; \ell m_{i,t}^m = \beta \ell m_{i,t-1}^m + (1-\beta)\ell_{i,t}^T \tag{4.3}$$

where β is a hyperparameter that controls the degree smoothed by the historical losses. The final normalized momentum used to find the familiarity of states is defined as

$$M_{i,t} = \frac{1}{2}\left(\frac{\ell m_{i,t}^m - \bar{\ell m}_t^m}{max\{|\ell m_{j,t}^m - \bar{\ell m}_t^m|\}_{j=1,...,N}} + 1 \right) \tag{4.4}$$

where $\bar{\ell m}_t^m$ is the average momentum loss of the dataset at the t^{th} training step of the algorithm and N is the number of samples. The higher the momentum value $M_{i,t}$, the higher the rareness of the sample in the circular buffer. The model is trained end to end by optimizing both IMPALA's loss and the auxiliary

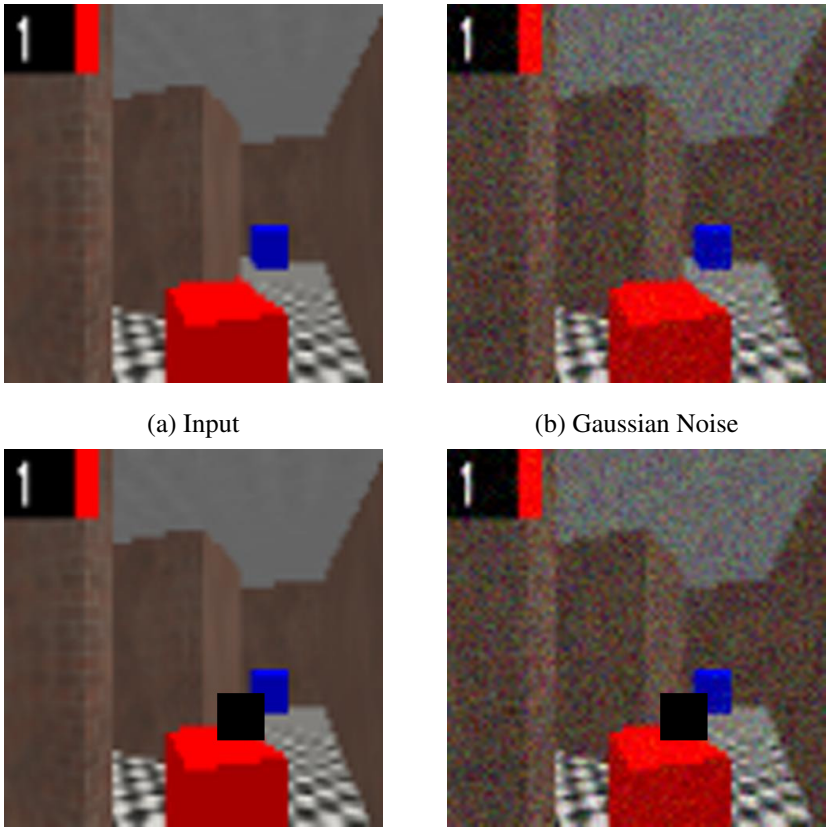
contrastive loss. Let the loss given by IMPALA be $\mathcal{L}_{\text{impala}}$ and that given by the contrastive learning branch be $\mathcal{L}_{\text{contrastive}}$, then we define the final loss to be the one shown in Equation 4.5 below.

$$\mathcal{L} = \mathcal{L}_{\text{impala}} + \gamma * \mathcal{L}_{\text{contrastive}} \tag{4.5}$$

where γ is a hyperparameter. The contrastive loss is given by:

$$\mathcal{L}_{\text{contrastive}} = \frac{1}{N} \sum_{i=1}^{N} -\log \frac{\exp\left(\frac{p_i^\top . p_i^{aug}}{\tau}\right)}{\sum_{p_i' \in X'} \exp\left(\frac{p_i^\top . p_i'}{\tau}\right)} \tag{4.6}$$

where N is the number of samples, $X^{'}$ represents $X^- \cup \{p_i^{aug}\}$, (p_i, p_i^{aug}) is the positive sample sample pair, X^- is the negative sample set of p and τ is the temperature.



(a) Input (b) Gaussian Noise

(c) Random Cutout (d) Final Image

Figure 4.7: **Image augmentations for contrastive learning: (a)** Shows downsampled input image for a trial. **(b)** Input image after adding gaussian noise to it. **(c)** Input image after applying random cutout augmentation. The black rectangle near the agent's position is the area cutout. **(d)** Final augmented image after adding gaussian noise and random cutout.

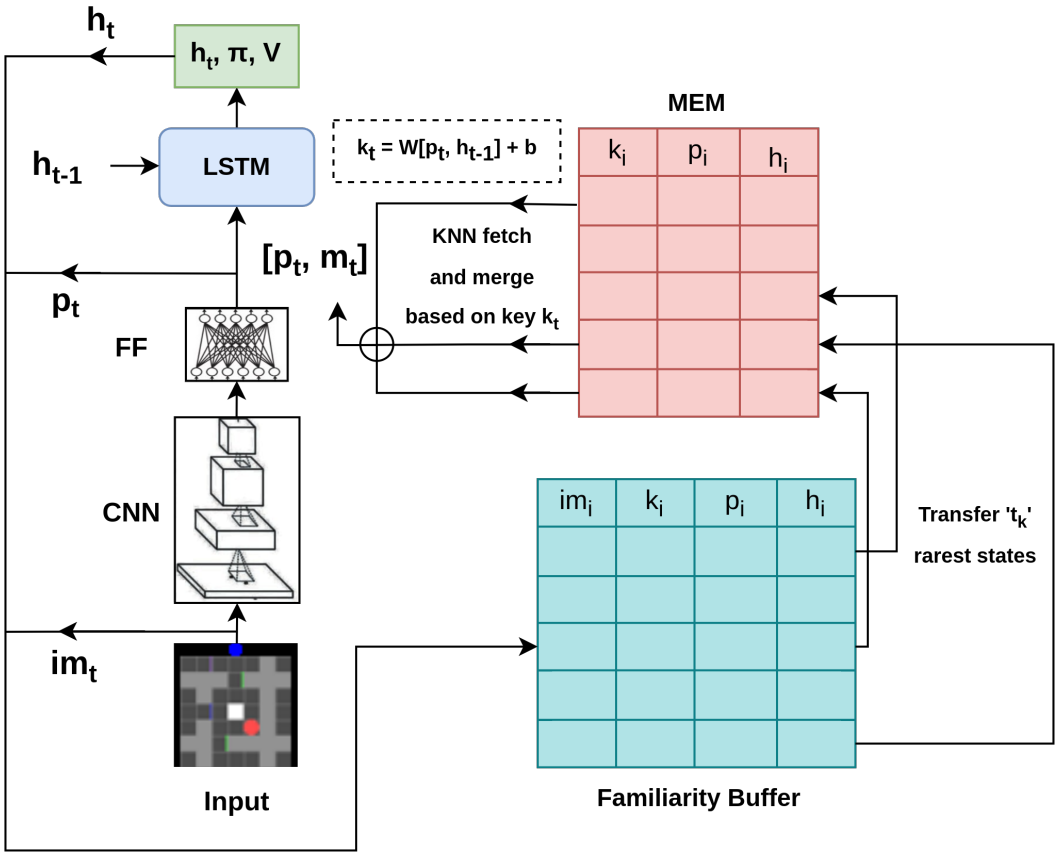


Figure 4.8: **Model Architecture:** The figure shows the momentum-boosted episodic memory architecture pipeline. The IMPALA [8] backbone consists of a CNN feature extractor followed by a Feed Forward layer that gives the embedding. This embedding is concatenated with the one hot action encoding, reward & memory to get pixel embedding p_i and then given to the LSTM network for further processing with working memory. The LSTM network additionally takes the past hidden state h_{t-1} as input. During training, the input image, pixel embedding, LSTM hidden states, and keys are stored in the familiarity buffer. The momentum loss tracked on this buffer during contrastive learning is then used to prioritize long-tail states. The MEM is then periodically updated with top t_f states from the familiarity buffer. The memory (m_t) is computed from the MEM using a weighted sum ($\bigoplus$) after fetching using a KNN similarity search on the keys present in the MEM using the query key k_t (Equation 4.8).

4.3.2 Combining Familiarity with Episodic Memory

The base architecture used is IMPALA with MEM proposed in [39]. Note that we do not use the one-step contrastive predictive coding (CPC) module that they have introduced. In the paper, an Episodic

Memory (MEM) is introduced on top of the IMPALA architecture, which is a circular buffer that stores the pixel embedding (p), LSTM hidden state (h), and the key (k), which is calculated using

$$k = W[p, h] + b \tag{4.7}$$

where W and b are learnable parameters and $[p, h]$ denotes the concatenation of p and h along the dimension axis. When IMPALA+MEM comes across situations with a similar context, it uses a method to learn how to save summaries of previous experiences and extract crucial data. The neural network (controller), which generates the model predictions, also receives extra inputs from the readings from memory. Learning long-term dependencies, which can be challenging when depending solely on back-propagation in recurrent architectures, is made simpler by successfully enhancing the controller's working memory capacity with experiences from various time scales received from the MEM.

In the original IMPALA+MEM algorithm, the pixel embeddings p_i, LSTM hidden states h_i, and keys k_i were computed after the feature extraction layer of IMPALA and then added to the MEM buffer. But, we instead add this to the familiarity buffer, which computes the familiarity of states based on the momentum loss and then periodically sends it to the MEM buffer to facilitate learning during rare trials. The agent is able to perform well on frequent states even if the MEM module is not used and only the simple IMPALA algorithm is used. Hence, only the relatively rare states are maintained in the MEM buffer to help learn on rare states that actually require the help of an external episodic memory module. The MEM gets t_k most rare states from the familiarity buffer after every t_f training epochs of contrastive learning, where t_k and t_f are hyperparameters.

The overall architecture can be seen in Figure 4.8. For a stimulus p_t and previous hidden state h_{t-1}, the agent chooses the most pertinent events to give as input m_t to the LSTM network using a type of dot-product attention [46] over its MEM. Using the key k_t, formed by p_t and h_{t-1} using Equation 4.7, a K Nearest Neighbour (KNN) search is done over the keys in MEM to find the most relevant K keys. The hidden states for these K relevant items are combined using the below-weighted sum (Equation 4.8) to get additional input m_t to be given to the LSTM network.

$$m_t = \frac{\sum_{i=1}^{K} w_i h_i}{\sum_{i=1}^{K} w_i} \quad w_i = \frac{1}{||k_t - W[p_i, h_i] - b||_2^2 + \epsilon} \tag{4.8}$$

where ϵ is a small constant and $||x||_2^2$ represents the squared $L2$ norm of x.

The pseudo-code for the algorithm is provided in Algorithm 1 below:

Inputs: Familiarity memory $\boldsymbol{fm}$, MEM $\boldsymbol{mem}$, transfer frequency $\boldsymbol{t_f}$, number of rare instances to transfer $\boldsymbol{t_k}$, number of IMPALA training epochs $\boldsymbol{T}$, within trajectory hop $\boldsymbol{hp}$ and the contrastive loss weight $\boldsymbol{\gamma}$.

Initialize: $fm.buffer, mem.buffer \leftarrow \{\}$ ▷ clear buffers

for t in $1, \cdots, T$ **do**
  $tr \leftarrow get\_impala\_batch(t)$ ▷ Get trajectories
  $im, k, p, h, impala\_loss \leftarrow train(tr, hp, mem)$
  $fm.add(im, k, p, h)$ ▷ Subset added - Equation 4.2
  $cl \leftarrow fm.contr\_train()$ ▷ Contr. Loss - Equation 4.6
  if $t \mod t_f = 0$ **then**
    $mv \leftarrow fm.calculate\_normalized\_momentum()$ ▷ Equation 4.4
    $rare\_experiences \leftarrow fm.get\_rare\_k(t_k, mv)$
    $mem.add(rare\_experiences)$
  end if
  $final\_loss \leftarrow impala\_loss + \gamma * cl$ ▷ Equation 4.5
  $final\_loss.backward()$ ▷ Backpropogate loss
end for

Algorithm 1: Pseudocode for the algorithm

4.4 Results

We compare the results of our method with mainly four different types of architecture. The first architecture we compare is IMPALA [8], which is an off-policy actor-critic framework and has shown substantial improvements over baselines like [47, 33]. The second one is the IMPALA+MEM introduced in [39]. The third experiment that we do is IMPALA with visual reconstruction. [6] have experimented with visual reconstruction [48], and similarly we add an extra task for visual reconstruction on top of IMPALA with a CNN-based autoencoder. The fourth one is where we included contrastive learning to learn good embeddings. It is to be noted that this approach does not find the rare states in the familiarity buffer, i.e. it samples k states uniformly randomly from the familiarity buffer. In contrast, in the proposed approach, we pass the rare k states to MEM from the familiarity buffer.

From the experiments (Figure 4.9 & Table 4.1), we can see clearly that contrastive learning (feature representation learning) alone cannot give good performance and we also need to add a familiarity buffer to prioritize rare states and pass them on to MEM. The training curves (Training/Zipfian Accuracy) and different ablations can be seen in the appendix section A.1. We can see that the Zipfian mean episode return for our method increases faster than all the other methods in the initial phase of training and also

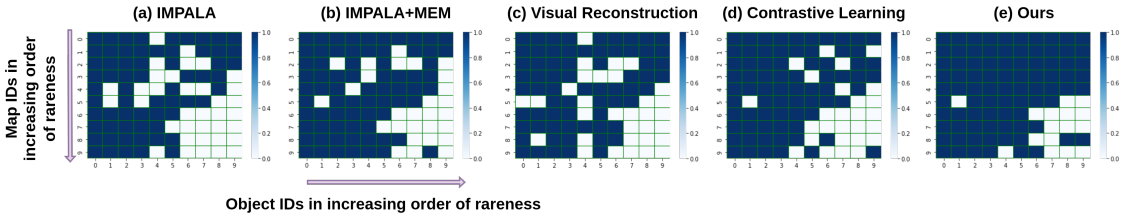


Figure 4.9: **Performance plots (Zipf's Gridworld): (a)** Performance of IMPALA agent on each map and object. The y-axis denotes the map axis, and the x-axis denotes the object axis. Value at (i, j) shows the performance (0-1 scale) of the agent on the trial where the object with ID j is chosen at the map with ID i. An increase in i and j means an increase in the rareness of the map and object respectively according to the Zipf's distribution (Equation 4.1). **(b)** Performance of IMPALA with MEM added. We can see that are some medium-rare trials in which the agent has learned to navigate and learn the task. **(c)** IMPALA with Visual Reconstruction using CNN-based autoencoder. **(d)** Performance of IMPALA+MEM with contrastive learning. **(e)** Performance of our agent consisting of familiarity buffer that highlights long tail samples for MEM using modified boosted contrastive learning.

converges later to have the highest accuracy. The hyperparameters used for our model across all tasks are listed in the appendix section A.2.

Having a higher training accuracy is not what we are looking for; instead, we want to have good accuracy when tested uniformly or in rare instances. Figures 4.9(a-e) show the performance of the five architectures on all the maps and objects of our environment for Zipf's Gridworld task. For each (map, object) combination, we plot the average performance across 50 trials. Figure 4.9a shows the performance of the IMPALA agent. We can see that the agent is not able to learn the extremely rare trials and also some medium-rare trials are not learned by the agent. Figures 4.9 (b) & (d) show the performance of IMPALA+MEM with and without contrastive learning respectively. In the case of IMPALA+MEM with contrastive learning, the familiarity buffer is sampled uniformly randomly to fetch states for the MEM. We see that the performance is almost similar, but still, the medium-rare trials are not learned. Figure 4.9c shows the performance of IMPALA with added visual reconstruction using CNN based autoencoder. This performs slightly better than the baseline (IMPALA) but fails to match the performance of other agents. Lastly, in Figure 4.9e we can see the performance of our agent with the familiarity module. The medium-rare trials are being learned, and also some very rare trials are being done successfully by our agent.

	IMPALA (shallow)	IMPALA (deep)	**Ours**
alien	1536.05	15962.10	**21948.24**
amidar	497.62	**1554.79**	324.34
assault	12086.86	19148.47	**25142.24**
asterix	29692.50	300732.00	**382749.53**
asteroids	3508.10	108590.05	**128394.94**
atlantis	773355.50	849967.50	**927883.39**
bank_heist	1200.35	**1223.15**	1193.34
battle_zone	13015.00	**20885.00**	19238.34
beam_rider	8219.92	32463.47	**44938.83**
berzerk	888.30	1852.70	**1928.46**
bowling	35.73	**59.92**	20.53
boxing	96.30	**99.96**	97.34
breakout	640.43	787.34	**928.64**
centipede	5528.13	11049.75	**28183.64**
chopper_command	5012.00	28255.00	**28442.89**
crazy_climber	136211.50	**136950.00**	136212.7
defender	58718.25	185203.00	**192837.78**
demon_attack	107264.73	**132826.98**	135454.52
double_dunk	-0.35	**-0.33**	-37
enduro	0.00	0.00	**0.00**
fishing_derby	32.08	**44.85**	42.45
freeway	0.00	0.00	**0.33**
frostbite	269.65	**317.75**	310.23
gopher	1002.40	66782.30	**62838.46**
gravitar	211.50	359.50	**419.64**
hero	33853.15	33730.55	**33854.35**
ice_hockey	-5.25	**3.48**	3.21
jamesbond	440.00	601.50	**728.35**
kangaroo	47.00	**1632.00**	1536.64
krull	9247.60	8147.40	**10293.93**
kung_fu_master	42259.00	**43375.50**	42474.4
montezuma_revenge	0.00	0.00	**86.34**
ms_pacman	6501.71	**7342.32**	7293.43
name_this_game	6049.55	21537.20	**23847.83**
phoenix	33068.15	**210996.45**	208494.24
pitfall	-11.14	**-1.66**	-15.35
pong	20.40	**20.98**	20.58
private_eye	92.42	98.50	**99.64**
qbert	18901.25	351200.12	**362748.22**
riverraid	17401.90	**29608.05**	28793.53
road_runner	37505.00	57121.00	**59304.52**
robotank	2.30	**12.96**	1.34
seaquest	1716.90	**1753.20**	1743.64
skiing	-29975.00	**-10180.38**	-29999.24
solaris	2368.40	2365.00	**2483.64**
space_invaders	1726.28	43595.78	**46383.91**
star_gunner	69139.00	200625.00	**218373.24**
tennis	-1.89	**0.55**	-2.44
time_pilot	6617.50	48481.50	**50283.15**
tutankham	267.82	292.11	**299.34**
up_n_down	273058.10	**332546.75**	315554.39
venture	0.00	0.00	**0.00**
video_pinball	228642.52	572898.27	**603947.23**
wizard_of_wor	4203.00	**9157.50**	8923.42
yars_revenge	80530.13	84231.14	**85039.92**
zaxxon	1148.50	32935.50	**34923.10**

Table 4.2: Atari scores after training for 200M steps in environment. Existing results taken from [8].

Table 4.1 gives more insights into our agent's performance on different tasks. Our agent is able to beat other compared agents by a large margin on all three evaluation metrics (Training/Zipfian accuracy, Uniform accuracy, and Rare accuracy). We also report the results of our methods on the Atari Learning Environment [49] in Table 4.4. Doing this shows that our architecture not only improves performance, but also retains or boosts performance on simple baseline tasks. The increase in perfomance for some tasks could suggest that they might exhibit long-tail behavior. We also tried using just the single value of contrastive loss instead of using the momentum loss on the history of contrastive losses, but the results were worse. This is because the momentum additionally captures the change in the contrastive losses, which predicts how fast/well it is able to learn on those samples. We also report results for Variational Autoencoder (VAE) [50] & Hierarchical Chunk Attention Memory (HCAM) [51] in appendix section A which have very different methods of solving and memory.

4.5 Discussions

Our work deals with the problem of long-tailed distribution in reinforcement learning. Inspired by the theory of complementary systems, which states that an intelligent agent requires a fast and slow learning system acting complementary to each other. Here, the momentum loss of contrastive learning provides a mechanism to detect long tail samples in an unsupervised manner. These samples are then prioritized to be stored in a separate buffer that stores hidden activation associated with such states. When a rare sample comes, a similarity search is done to find relevant keys, and the corresponding hidden activations are merged to be reinstated in the recurrent layers.

This architecture relates to how the hippocampus which is a fast learning system acts in tandem with the slow learning cortex of the organism to store relevant experiences and replay them to overcome the statistics of the environment the organism is subjected to [52]. The episodic memory relies on the network to discover long-tail data from the incoming data stream. The network relies on the episodic memory for identifying the relevant memory of the long tail data in order to reinstate it in the recurrent layers of the working memory system to execute the episodic sequence. Similarly, the brain could reinstate episodic sequences from the hippocampus to the working memory when animals execute a task.

Furthermore, dopamine neurotransmitter has been found to detect novel states and relay them to the hippocampus [53]. The firing of dopamine neurons is also related to curiosity and learning progress which is analogous to momentum loss here [54, 55]. The neural network prioritizes its representations and persistence of memory based on how informative the state is for executing the task. This architecture can also potentially execute episodic generalization as proposed by [39] but adapted for long-tailed data. The resulting embedding could be different, but the underlying dynamics of executing the task could be the same. Future work could look at how we could extend this to more realistic 3D environments and include systematic generalization to compose temporally separated rare states to solve tasks [56].

Method	Accuracy (%) | **Zipf's Gridworld**		
	Zipfian	Uniform	Rare
IMPALA	88.3 ± 2.1	41.1 ± 1.8	0.0 ± 0.0
IMPALA + Visual Reconstruction	90.2 ± 2.4	45.9 ± 1.1	0.0 ± 0.0
IMPALA + MEM	92.9 ± 3.2	51.2 ± 2.3	25.0 ± 2.0
IMPALA + MEM + Contrastive Learning	94.8 ± 2.7	52.3 ± 1.1	25.1 ± 2.4
Ours	**98.5 ± 1.2**	**66.3 ± 1.0**	**25.2 ± 1.1**

Method	Accuracy (%) | **Zipf's 3DWorld**		
	Zipfian	Uniform	Rare
IMPALA	95.9 ± 4.1	65.6 ± 8.2	33.3 ± 2.1
IMPALA + Visual Reconstruction	96.0 ± 2.2	68.6 ± 10.4	37.1 ± 3.1
IMPALA + MEM	97.3 ± 2.2	74.3 ± 6.0	42.6 ± 8.3
IMPALA + MEM + Contrastive Learning	97.5 ± 1.8	74.4 ± 5.5	43.0 ± 1.2
Ours	**99.2 ± 1.3**	**80.1 ± 2.0**	**55.6 ± 2.4**

Method	Accuracy (%) | **Zipf's Labyrinth** | Forward Zipf		
	Zipfian	Uniform	Rare
IMPALA	63.3 ± 3.0	27.9 ± 3.1	5.0 ± 4.2
IMPALA + Visual Reconstruction	65.6 ± 8.5	31.2 ± 1.9	8.7 ± 4.2
IMPALA + MEM	68.5 ± 7.6	45.3 ± 2.5	23.1 ± 3.5
IMPALA + MEM + Contrastive Learning	69.1 ± 4.6	49.3 ± 4.4	27.5 ± 2.2
Ours	**77.5 ± 7.0**	**54.2 ± 2.2**	**32.2 ± 2.1**

	Accuracy (%) | **Zipf's Labyrinth** | Reversed Zipf		
Method	Zipfian	Uniform	Rare
IMPALA	53.8 ± 7.1	21.3 ± 3.3	4.1 ± 3.1
IMPALA + Visual Reconstruction	55.6 ± 13.2	25.2 ± 5.1	9.6 ± 3.3
IMPALA + MEM	66.7 ± 11.3	38.6 ± 2.0	16.7 ± 2.7
IMPALA + MEM + Contrastive Learning	68.9 ± 9.4	39.0 ± 7.7	18.8 ± 5.1
Ours	$\mathbf{71.3 \pm 3.1}$	$\mathbf{47.1 \pm 2.3}$	$\mathbf{25.3 \pm 1.9}$

Table 4.1: **Evaluation Performance:** We compare four different methods with our algorithm namely IMPALA, IMPALA+Visual Reconstruction using a simple CNN-based autoencoder, IMPALA+MEM, and IMPALA+MEM with only contrastive learning. We report median results across three runs ($\pm$ absolute median deviation across runs) with distinct random seeds for models trained for 4×10^7 steps. Our method (IMPALA+MEM+Contrastive Learning+Rare State Prioritization using Familiarity Buffer) beats the remaining methods on all three evaluation metrics. **LABELS: Zipfian:** Maps and objects are chosen according to the Zipfian distribution, **Uniform:** Maps and objects are chosen uniformly randomly. **Rare:** Rarests 20% objects on rarest 20% maps are both uniformly randomly chosen.

Chapter 5

Conclusion and Future Work

In this thesis, a profound exploration into the integration of cognitive building blocks within the realm of Reinforcement Learning (RL) is undertaken, with the overarching aim to narrow the existing chasm between artificial intelligence and human cognition. This endeavor is pivotal as it paves the way for the development of RL systems that are not only efficient but also capable of mirroring the complex adaptability inherent in human cognitive functions.

One of the most significant contributions of this research is the emphasis on the critical role of cognitive processes such as perception, memory, and reasoning, within the context of RL. The thesis demonstrates how incorporating these elements into RL algorithms markedly enhances their adaptability and decision-making prowess, thereby making these systems more effective and applicable in real-world scenarios. A key highlight is the investigation into the emergence of direction selectivity and motion strength in deep RL networks. This aspect is crucial as it illustrates that RL agents, much like humans, can learn to interpret and respond to directional movements and varying motion strengths within their environment.

Furthermore, the thesis introduces an innovative approach to episodic memory in RL, particularly effective in long-tailed environments characterized by infrequent, yet significant events. Drawing inspiration from human memory systems, this model significantly boosts the capability of RL agents to learn from past experiences and adapt to new situations. This approach is particularly evident in the proposed Momentum Boosted Episodic Memory (MBEM) model, which shows superiority in managing skewed distributions and complex tasks, outperforming existing methods.

Another cornerstone of this thesis is the focus on reasoning and decision-making within RL algorithms. This integration is shown to be essential for complex problem-solving and planning, echoing the higher cognitive processes found in humans. The use of platforms like DeepMind's PsychLab and Zipfian Environments for controlled testing provides invaluable insights into the performance and adaptability of these enhanced RL algorithms.

Looking towards the future, this research sets a foundational stage for further investigation in the field of RL. The potential of RL to reach human-level cognitive capabilities is underscored, especially when fundamental cognitive processes are integrated. Future research directions could explore more intricate

cognitive tasks, expand on the MBEM model, and translate these theoretical findings into practical, real-world applications. We also looked at some additional experimentation using other memory structures like Hopfield Networks and Modern Hopfield Networks in appendix section B. These are powerful and more biologically plausible networks that can be further investigated to see if they can be tailored for solving problems with such skew.

In conclusion, this thesis represents a significant stride towards the creation of RL systems that not only learn and adapt but also possess the ability to think and perceive in ways akin to human cognition. By embedding human-like cognitive processes into RL, the research opens new vistas in artificial intelligence, edging it closer to human understanding and adaptability. This work contributes substantially to both AI and cognitive science and heralds a new era of intelligent system design, with far-reaching implications for diverse real-world applications.

Appendix A

Zipfian Environments (Additional Experiments)

We compare our method with two more architectures with very different methods of solving and memory. They are:

- Variational Autoencoder (VAE) [50]

- Hierarchical Chunk Attention Memory (HCAM) [51]

The results on all three tasks are provided in Table A.1. We notice that both these methods perform poorly compared to our method and even the baseline IMPALA. Also, these methods perform worse when tested on the rare 20% of the tasks.

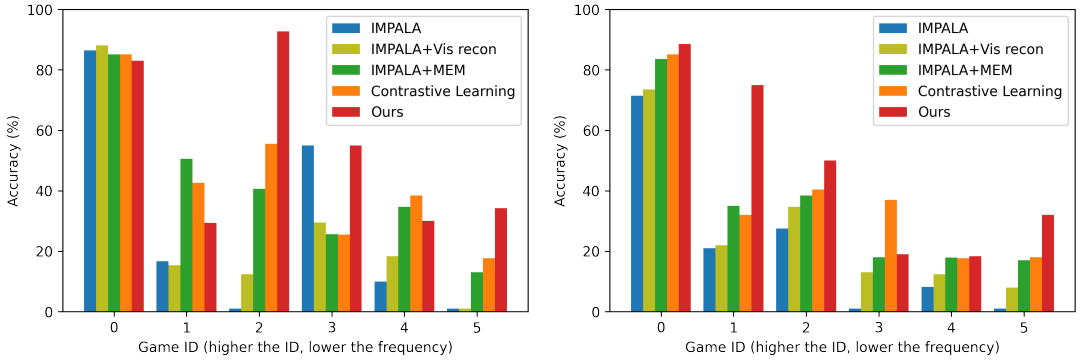
A.1 Supplementary Analyses

For Zipf's Labyrinth task, we plot the accuracies for each of the methods on all tasks in Figure A.1. We can clearly see that our method (red bar) performs consistently well on most of the rare and medium rare trials. On the extremely rare trials (tasks), our method is the best across all architectures which shows the effectiveness of discovering long-tailed states using our proposed method.

Method	Accuracy (%) | **Zipf's Gridworld**		
	Zipfian	Uniform (All maps and objects)	Rare (Rarest 20% objects on rarest 20% maps)
VAE	45.3 ± 5.6	5.7 ± 4.5	0.0 ± 0.0
HCAM	75.9 ± 1.1	25.6 ± 0.9	0.0 ± 0.0
Ours	$\mathbf{98.5 \pm 1.2}$	$\mathbf{66.3 \pm 1.0}$	$\mathbf{25.2 \pm 1.1}$

Method	Accuracy (%) | **Zipf's Labyrinth**					
	Forward Zipf			Reversed Zipf		
	Zipfian	Uniform (All maps and objects)	Rare (Rarest 20% objects on rarest 20% maps)	Zipfian	Uniform (All maps and objects)	Rare (Rarest 20% objects on rarest 20% maps)
VAE	35.0 ± 6.8	3.6 ± 2.6	0.0 ± 0.0	36.2 ± 6.4	5.3 ± 3.2	0.0 ± 0.0
HCAM	53.1 ± 2.3	20.3 ± 3.4	0.0 ± 0.0	56.2 ± 4.3	25.6 ± 2.5	0.0 ± 0.0
Ours	$\mathbf{71.3 \pm 3.1}$	$\mathbf{54.2 \pm 2.2}$	$\mathbf{32.2 \pm 2.1}$	$\mathbf{77.5 \pm 7.0}$	$\mathbf{47.1 \pm 2.3}$	$\mathbf{25.3 \pm 1.9}$

Table A.1: **Evaluation Performance:** We additionally compare our method with different varying types of methods namely Variational Autoencoder (VAE) and Hierarchical Chunk Attention Memory (HCAM).



(a) Forward Zipf.

(b) Reversed Zipf.

Figure A.1: **Performance Plots (Zipf's Labyrinth):** Performance of agents in Zipfs Labyrinth Foward & Reversed on all tasks.

The training curve for the main experiments (Zipf's Gridworld) can be seen in Figure A.2.

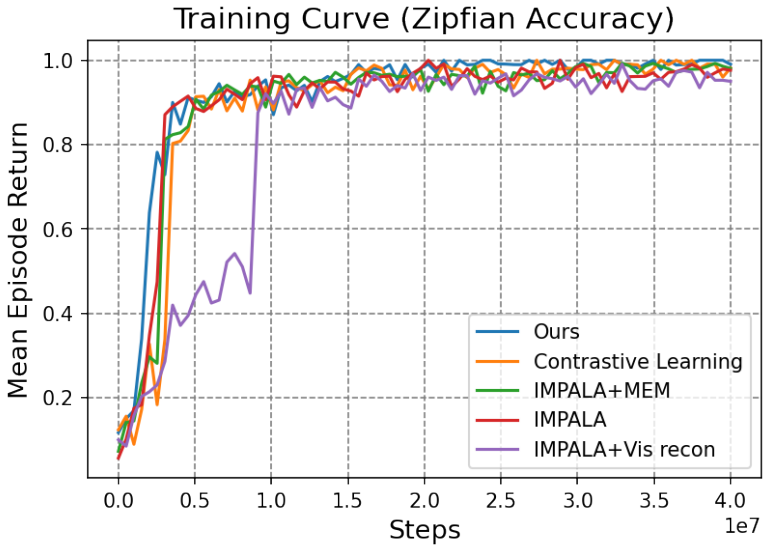


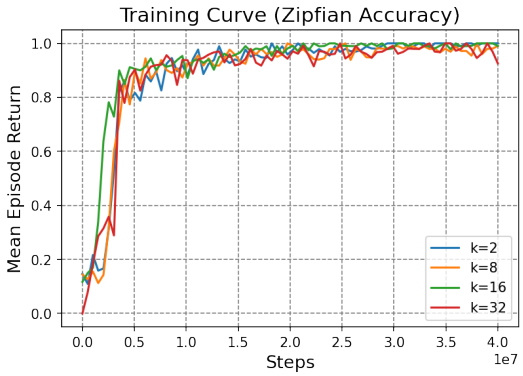
Figure A.2: **Performance plots:** Training curves for different experiments.

For the Zipf's Gridworld task, we only consider 10 maps and 10 objects. In the paper where the task is introduced, they have considered 20 maps and 20 objects. Reducing the number of maps and objects doesn't make the task easier, but might be more difficult because we train the agent using just 50 actors and for 4e7 steps due to computational constraints. We were not able to reproduce the exact results on 20 maps and 20 objects using our implementation of IMPALA. The original code for training on

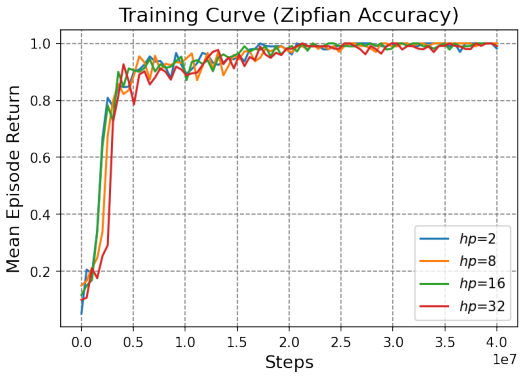
Deepmind's environments is also not publicly available yet due to some issues on their side. For similar reasons we only consider 6 tasks in Zipf's Labyrinth environment. The 6 tasks in order are listed below:

```
LEVELS_DMLAB30_FORWARD = [
    'rooms_collect_good_objects_train', 'rooms_exploit_deferred_effects_train',
    'rooms_select_nonmatching_object', 'explore_object_locations_small',
    'explore_goal_locations_small', 'explore_object_rewards_few',
]
```

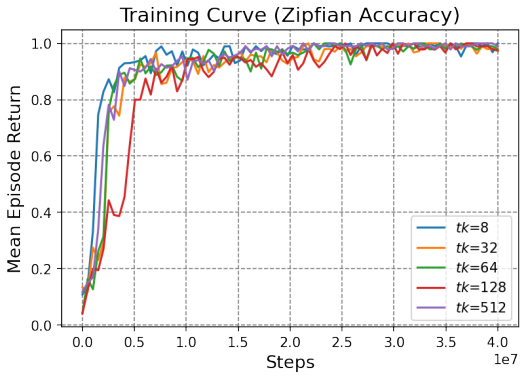
The effects of changing different hyperparameters of our architecture are explained here. We look at the 'K' value of K Nearest Neighbour, Trajectory Hop (hp), Rare State Transfer amount (t_k), and Rare State Transfer Frequency (t_f) for our ablation study. The effect on the training of these hyperparameters can be seen in Figure A.3. Tables A.2- A.5 give more insights into the effect of different hyperparameters on the training.



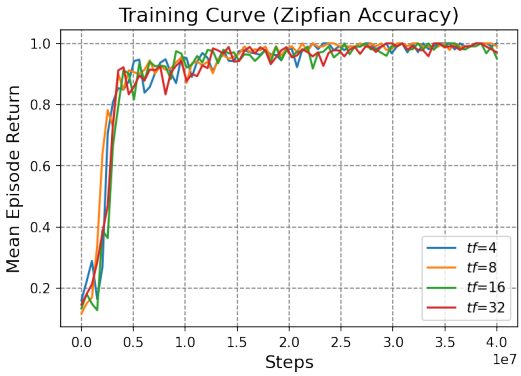
(a) Effect of KNN 'K' value



(b) Effect of trajectory hop (hp)



(c) Effect of transfer amount (t_k)



(d) Effect of transfer frequency (t_f)

Figure A.3: **Training Performance:** Effect of changing different hyperparameters on training.

Table A.2: Effect of KNN 'K' value.

K Value	Accuracy (%)		
	Zipfian	Uniform (All maps and objects)	Rare (Rarest 20% objects on rarest 20% maps)
2	99.4	79.0	0
8	98.5	67.6	0
16	99.9	84.4	49.9
32	97.3	74.9	24.9

Table A.3: Effect of trajectory hop (hp).

(hp)	Accuracy (%)		
	Zipfian	Uniform (All maps and objects)	Rare (Rarest 20% objects on rarest 20% maps)
2	99.1	78.4	24.8
8	99.3	79.1	24.9
16	99.9	84.4	49.9
32	99.1	77.7	25.2

Table A.4: Effect of transfer amount (t_k).

(t_k)	Accuracy $(\%)$		
	Zipfian	Uniform (All maps and objects)	Rare (Rarest 20% objects on rarest 20% maps)
8	99.2	81.1	24.9
16	99.9	84.4	49.8
32	99.2	76.0	0
64	98.9	77.9	0
128	98.7	69.8	0

Table A.5: Effect of transfer frequency (t_f).

(t_f)	Accuracy $(\%)$		
	Zipfian	Uniform (All maps and objects)	Rare (Rarest 20% objects on rarest 20% maps)
4	99.1	79.9	25.0
8	99.9	84.4	49.8
16	99.3	82.0	49.9
32	99.1	76.9	74.8

We also run our experiments on the Atari Learning Environment [49] whose results are shown in Table 4.4). We specifically run it across the set of 57 Atari games previously examined by [18]. This collection of games presents a compelling array of tasks characterized by multifaceted challenges, encompassing aspects such as sparse reward structures and considerable variations in scoring scales across the diverse game set.

A.2 Experiment Hyperparameters

	Zipf's Gridworld	Zipf's 3DWorld	Zipf's Labyrinth
Image Width	84	84	84
Image Height	84	84	84
Action Repeats	1	3	1
Unroll Length	32	32	32
Discount (γ)	0.99	0.99	0.99
Baseline loss scaling	0.5	0.6	0.5
Entropy cost	0.01	0.00001	0.01
Optimizer	RMSProp	RMSProp	RMSProp
Learning rate	$3e-4$	$3e-4$	$3e-4$
Number of training steps	$4e7$	$4e7$	$4e7$
Maximum steps in a trial	100	200	500

	Additional Parameters	Zipf's 3DWorld	Zipf's Labyrinth
Zipf's Exponent (e)	2	2	2
Number of Actors	50	50	50
Trajectory Hop (hp)	16	16	16
Average Momentum Beta (β)	0.97	0.97	0.97
Loss Gamma (γ)	0.5	0.5	0.5
MEM Buffer capacity	1024	2048	2048
Familiarity Memory Buffer capacity	1024	2048	2048
Rare State Transfer Amount (t_k)	512	512	512
Rare State Transfer Frequency (t_f)	8	8	8
KNN (K)	16	32	32
Epsilon (ϵ)	$1e-3$	$1e-3$	$1e-3$
Sigma (σ)	0.05	0.05	0.05

Appendix B

Auto-associative Memory

B.1 Tolman-Eichenbaum Machine (TEM)

The Tolman-Eichenbaum Machine (TEM) proposes that the hippocampus [9] isn't just for remembering places, but also for understanding connections between things. It uses past experiences to help us navigate new situations, like a brain-built map for life beyond just physical space. Studies suggest that the Tolman-Eichenbaum machine can explain several phenomena related to hippocampal function, including the ability to generalize across similar experiences and the ability to rapidly form new memories. The model also suggests that the hippocampus plays a critical role in learning abstract concepts and understanding the relationships between them. Experimental validation of TEM's capabilities across diverse tasks requiring spatial encoding, relational inference, and rapid memory formation underscores its effectiveness in both domains. TEM provides a valuable computational framework for further exploration of these intricate cognitive processes.

TEM's proficiency in spatial encoding and retrieval was demonstrated through the Object-Location memory task. Participants were tasked with memorizing object positions and subsequently identifying object location changes. It exhibited high accuracy in this task, corroborating its ability to process and store spatial information effectively. To evaluate it's relational reasoning abilities, the Transitive Inference task was employed. Participants deduced relationships between indirectly presented objects based on learned hierarchies (e.g., A $>$ B $>$ C, infer A $>$ C). TEM achieved high accuracy in this task, demonstrating its capacity for processing relational information and effectively inferring novel relationships. In Figure B.1: **A, B** are examples of relational reasoning tasks. **C** is an example of a spatial reasoning task. **D–G** are relational and spatial graph representations of the tasks in **A–C**. **H** is an illustration of the graph embedding task. Each node in the graph represents a word, and the edges between the nodes represent the semantic relations between the words. The task is to predict the missing word (indicated by the question mark) in the graph. In **I**, learning curves for the graph embedding task with different numbers of training examples are shown.

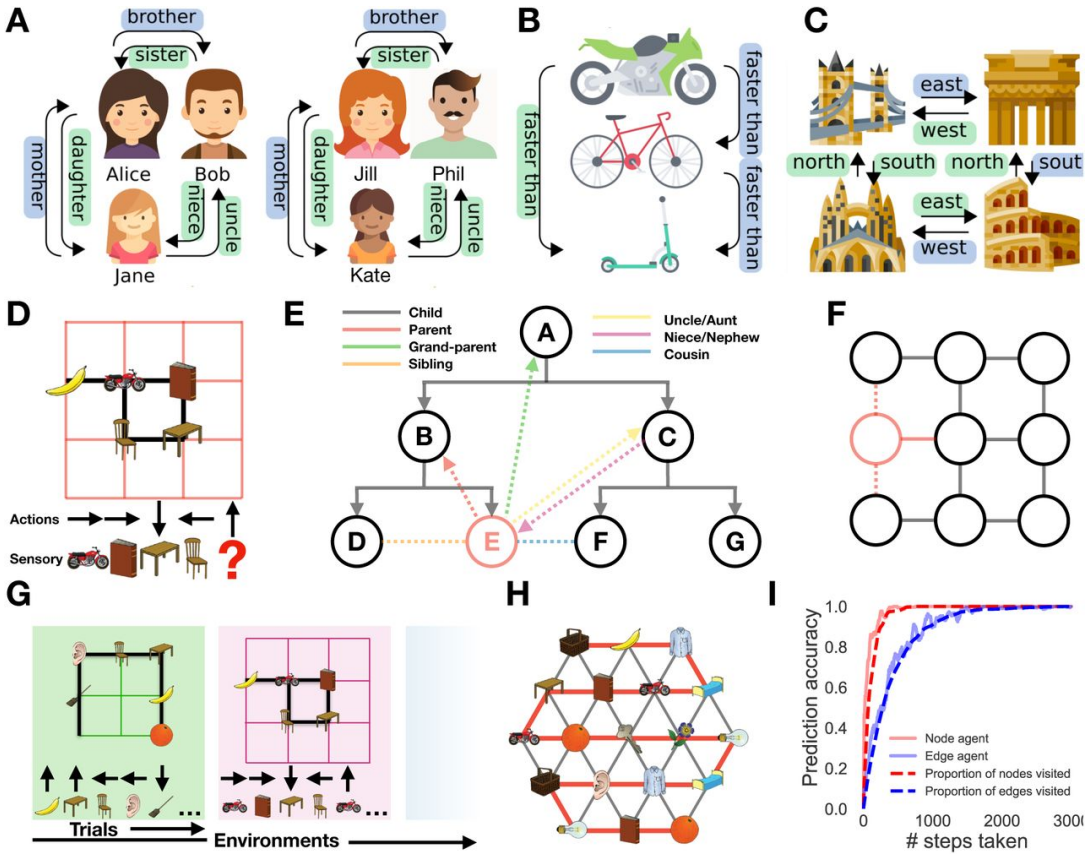


Figure B.1: Overview of the different types of relational reasoning tasks and the environments used to evaluate them. Taken from [9].

TEM's efficacy in rapid memory formation was further assessed through the Rapid Learning task. Participants associated object pairs with rewards, demanding swift memory consolidation. TEM excelled in this task, rapidly encoding and retrieving rewarded associations with high accuracy, highlighting its capabilities in dynamic memory formation. It's performance across diverse tasks encompassing spatial encoding, relational reasoning, and rapid memory formation underscores its effectiveness in both domains. This multifaceted capability establishes TEM as a valuable framework for further probing the intricate neural mechanisms underlying these complex cognitive processes.

The model (Figure B.2) consists of two main components: an inference model and a generative model. The inference model (left) takes as input sensory data and infers the animal's current position and the path it has traveled. The generative model (right) uses the inferred path to predict the sensory data that the animal will experience at the next time step. The model is trained using a combination of supervised learning and unsupervised learning (to learn the generative model parameters). At each time step, the model makes a prediction about the next sensory input and updates its internal state accordingly. The model can also recall memories from previous experiences by reactivating the attractor states that correspond to those memories.

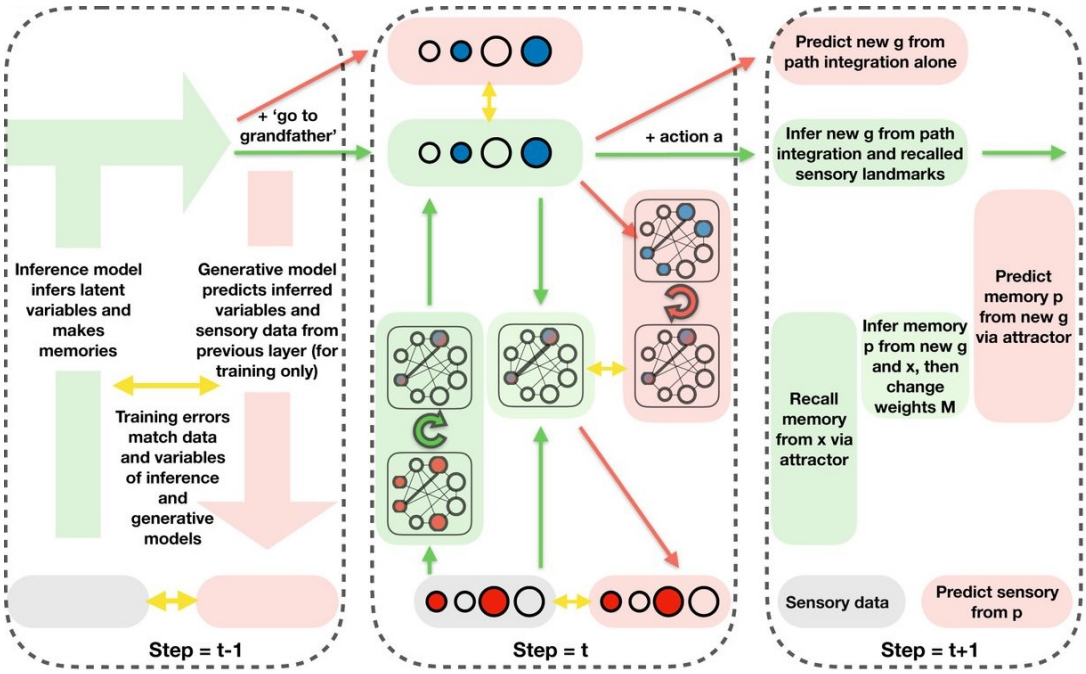


Figure B.2: Overview of the proposed generative model for spatial learning and memory recall in the brain. Taken from [9].

B.2 Hopfield Networks (HN)

Hopfield Networks, a class of recurrent artificial neural networks, are widely recognized for their application in associative memory and optimization problems. Introduced by John Hopfield in 1982 [57], these networks consist of symmetrically connected nodes, where the state of each node is determined by the weighted sum of inputs from other nodes. The network is trained to store patterns as stable states, allowing it to recall and reconstruct complete patterns from partial or noisy input. The energy function associated with Hopfield Networks facilitates convergence to stable states during recall. While Hopfield Networks excel in content-addressable memory and pattern recognition tasks, they have limitations in scalability and susceptibility to spurious states. Nonetheless, their simplicity and effectiveness in certain applications make them a valuable tool in the realm of neural network research and computational neuroscience.

Let $X \in -1, 1_{DxM}$ represent the set of target memories (binary patterns) that we want to store in the HN, i.e. the memory dataset. Then, X is a DxM matrix where M (indexed by m) is the number of targets and D (indexed by d) is the number of bits in each pattern, with each $X_{dm} \in -1, 1$. The synaptic connections in a HN follow Hebbian learning and thus the weight matrix, W, is set to be $W = (1/M)X.X_T$. Hence, the synaptic weight connecting nodes v_i and v_j, W_ij, will become

increasingly positive as the set of memory patterns for these nodes become more highly correlated (i.e., the more the nodes share the same sign).

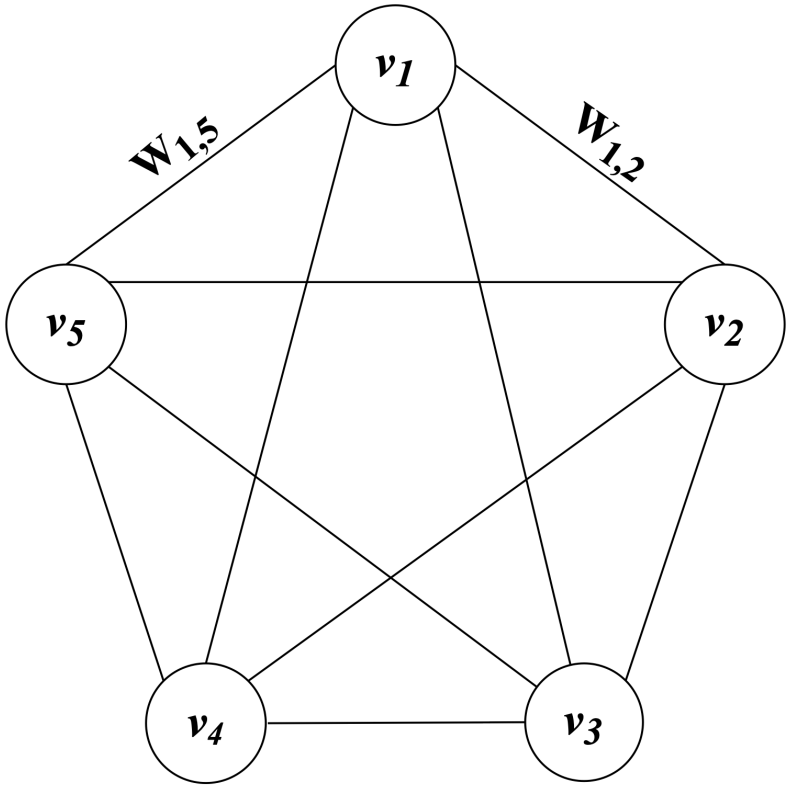


Figure B.3: A diagram of a traditional HN with N = 5 nodes. Taken from [10].

B.3 Modern Hopfield Networks (MHN) and Biological Plausibility

Modern Hopfield Networks [12] represent an evolution of the classic Hopfield architecture, incorporating advancements to address limitations and enhance performance. Recent developments focus on scalability, robustness, and efficient learning mechanisms. Techniques such as sparsity constraints, structural modifications, and advanced learning rules have been introduced to mitigate issues associated with the classical model. Additionally, the integration of modern neural network concepts, including deep learning principles and novel activation functions, contributes to the improvement of Hopfield Networks for various applications, ranging from associative memory to combinatorial optimization problems. By embracing these contemporary enhancements, Modern Hopfield Networks strive to overcome traditional constraints and offer a more versatile and powerful tool in the landscape of neural network research and practical applications. Although modern HNs improved some aspects of HNs, it moved

the implementation further from being biologically plausible. This is due to the many-body synapses introduced from the polynomial interaction function.

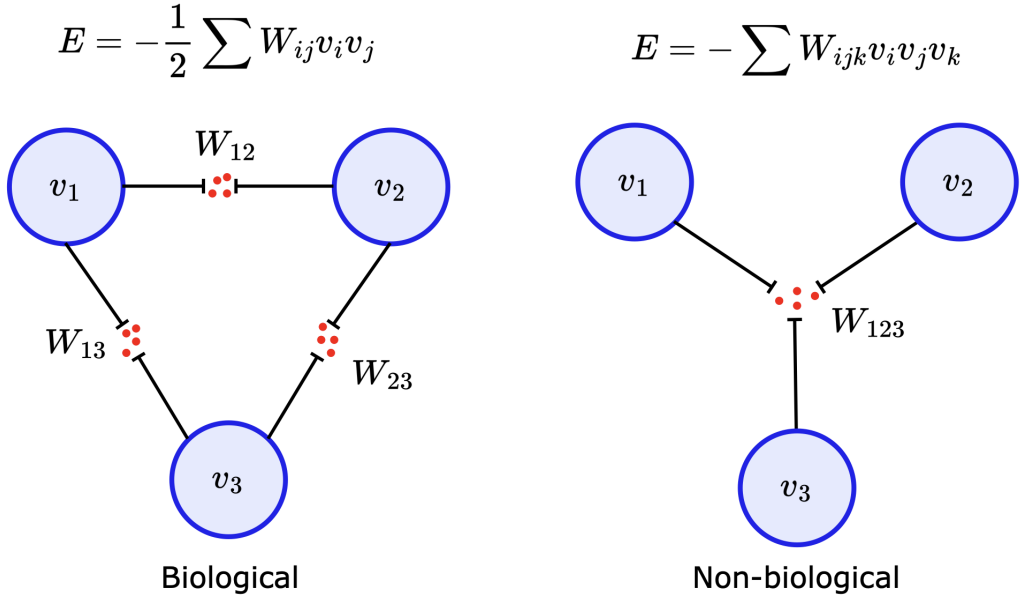


Figure B.4: Traditional Hopfield Network (HN) diagram (left). Modern HN configuration with the interaction function $F(x) = x^3$ (right). This representation depicts weight dependence on three nodes in the modern HN, which is biologically implausible. Taken from [10].

We can see that the weights here (Right part in Figure B.4) depend on three nodes, and since a biological synapse can only connect two nodes, these many-body synapses are biologically implausible!!!

To address this many-body synapse issue, [10] proposed a model of large associative memory that moves closer to being biologically plausible. Their approach was to couple a set of N_h hidden nodes, h (indexed by j), to a set of N_v feature nodes, v (indexed by i). The full network of $N_v + N_h$ nodes forms a bipartite graph, which behaves like a classifier and each target pattern in the feature nodes corresponds to a one-hot setting of the hidden nodes, and vice versa.

One more biological implausibility in MHNs is due to the use of softmax, $S(h)$. The output of hidden node j will depend on all other hidden nodes, even though they are not connected. This non-local computation poses a degree of biological implausibility. It is also biologically suspect that the hidden nodes ideally represent the target patterns using a one-hot representation.

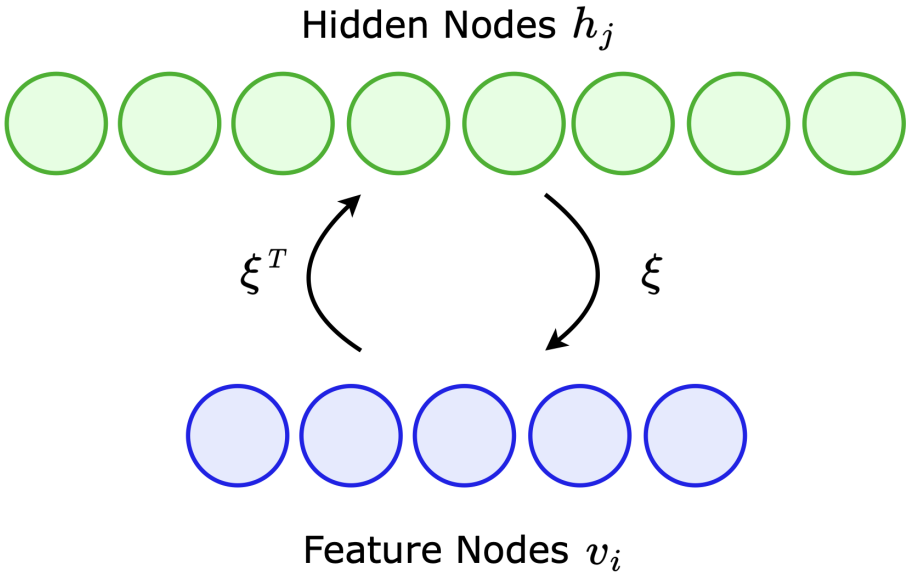


Figure B.5: Additional Hidden Nodes. Taken from [10].

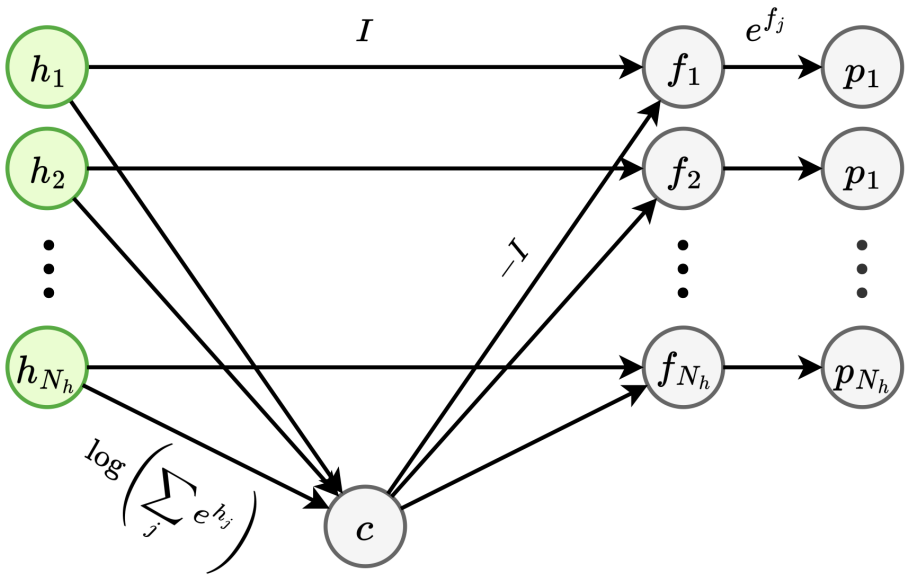


Figure B.6: Computation of softmax using locLSE network. Taken from [10].

To tackle this they present a more biologically plausible version of the LSE network that only uses local computations - the locLSE network (Figure B.6). Recall that to be biologically plausible, the output of each node must only depend on its own input, i.e. involve local computation. Computing $S(h)j$, i.e. the softmax of hidden node j (Figure B.5), involves normalization with respect to all other

hidden node activities, and is thus a non-local computation. Hence, $S(h)_j$ cannot be computed by a single node alone (h_j) - it must be implemented by a network of nodes.

B.4 Experiments

B.4.1 Experiment 1 (Hopfield Network Memory Test)

An experiment was conducted where the Hopfield Network was trained on ‘n_patterns‘ number of samples of the MNIST dataset [58] and the memory capacity was tested by reconstructing the noisy input patterns. The above plot shows the memory ratio plotted for different numbers of training samples. The memory ratio is the number of noisy samples correctly reconstructed divided by the total number of the input patterns. The red dotted line shows the expected memory i.e. 0.138*N, where N is the number of neurons in the associative memory. The blue dotted line shows the observed memory, which is the average of all memory ratios for different values of ‘n_patterns‘.

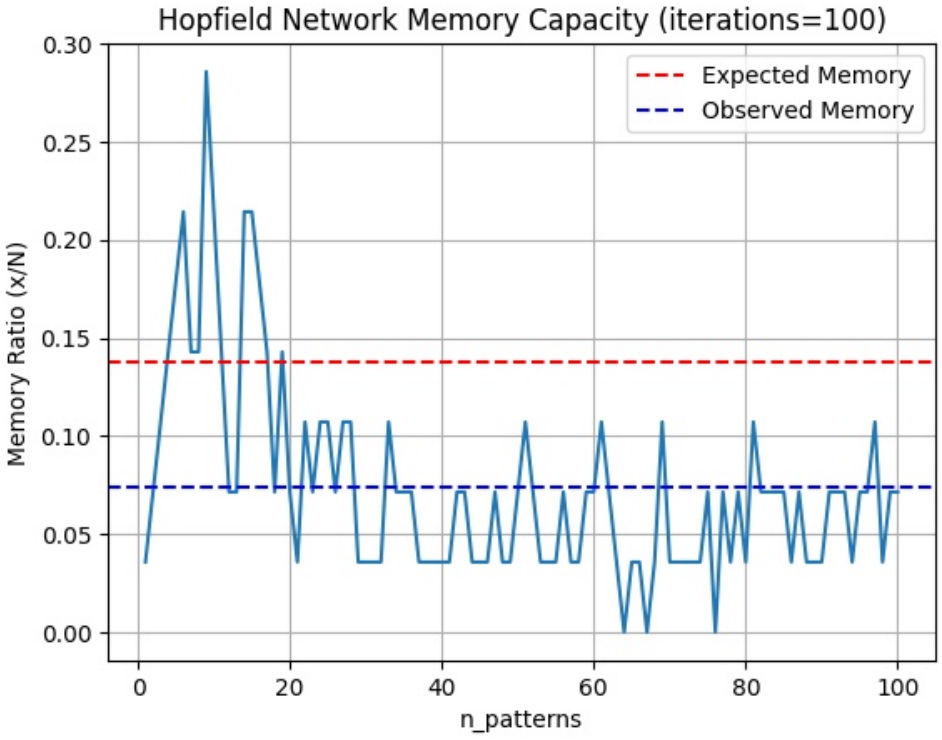


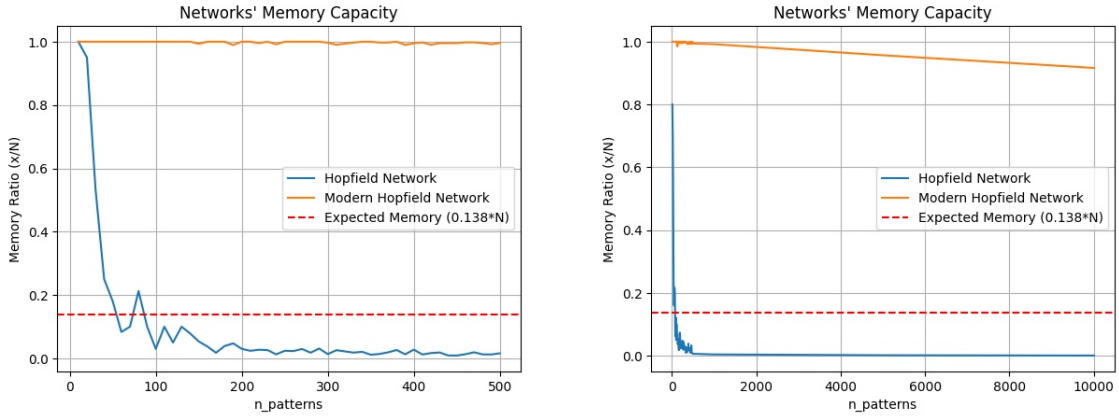
Figure B.7: Hopfield Network Memory Capacity.

We see that the observed memory is lesser than the expected value of 0.138N (B.7). One reason could be that the memory is affected by the quality of the input patterns. If the input patterns are not orthogonal or are too similar to each other, the network may have difficulty in retrieving the correct

memory and may retrieve spurious attractors instead. The MNIST dataset consists of image samples for 10 digits (0-9), and since these images are very similar to each other, the results that we have are justified.

B.4.2 Experiment 2 (Hopfield Network vs Modern Hopfield Network Memory)

A similar experiment to Experiment 1 was conducted, but now we compare both Hopfield Network and Modern Hopfield Network (Figure B.8). We can see that the hopfield network fails badly when the number of samples gets pretty huge, which is completely expected. Modern Hopfield Networks also seem to show a decrease in memory capacity performance, which is a bit surprising because for d=784, the memory capacity for MHN is $2^{d/2} = 2^{784/2} = 2^{392}\ 1e118$, hence for such a small dataset size (even though there is less orthogonality), the performance shouldn't have been so bad.



(a) Plot showing memory capacity.

(b) Zoomed version

Figure B.8: Comparison of different networks' memory capacity.

B.4.3 Experiment 3 (DQN vs DQN+NEC vs DQN+MHN)

We trained multiple algorithms on an Atari task (Montezuma's Revenge) and compared their performance (Figure B.9). We experimented with 3 algorithms namely DQN [1], DQN+NEC [11] and DQN+MHN [12]. DQN failed to solve the task, which is evident from its architectural setting as it is missing a major component of long term memory which is required to solve the task. DQN+NEC performed better and was able to give an average reward of around 40.1 on this task. DQN+MHN performed the best and gave an average reward of 50.0, but was computationally more expensive than both DQN and DQN+NEC. This is due to the huge matrix product that's involved in each retrieve phase of the MHN.

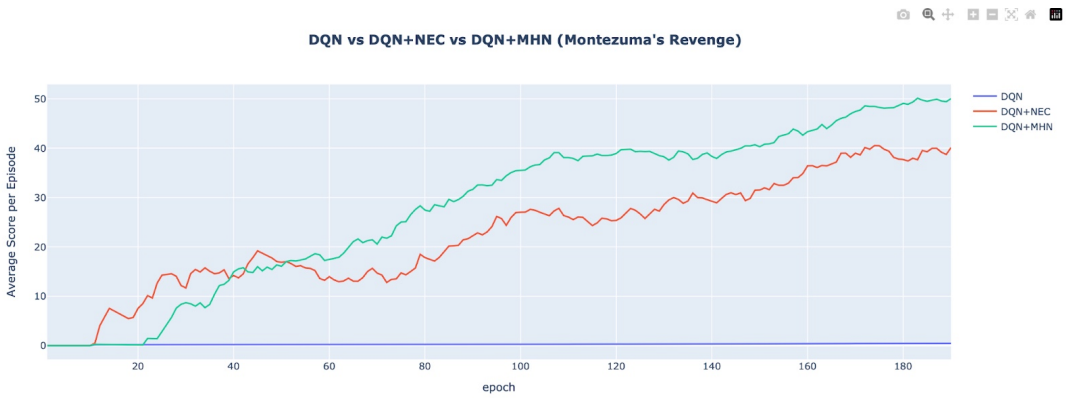


Figure B.9: Performance plot for DQN [1], DQN+NEC [11] and DQN+MHN [12] on Montezuma's Revenge.

Related Publications

1. **Dolton Fernandes**, Pramod Kaushik, Bapi Raju Surampudi. 2023, June. **Emergence of Direction Selectivity and Motion Strength in Dot Motion Task Through Deep Reinforcement Learning Networks.** *In Proceedings of International Joint Conference on Neural Networks (IJCNN), Gold Coast, Australia, 2023*.

2. **Dolton Fernandes**, Pramod Kaushik, Nallapu Bharghv Teja, Bapi Raju Surampudi. 2022, August. **Emergence of Direction Selectivity and Motion Strength in Dot Motion Task Through Deep Reinforcement Learning Networks.** *In Proceedings of 2022 Conference on Cognitive Computational Neuroscience*.

3. **Dolton Fernandes**, Pramod Kaushik, Harsh Shukla, and Bapi Raju Surampudi. 2022, December. **Momentum Boosted Episodic Memory for Improving Learning in Long-Tailed RL Environments**. *In Proceedings of Deep Reinforcement Learning Workshop NeurIPS 2022.*

Bibliography

[1] Alessandro Sebastianelli, Massimo Tipaldi, Silvia Liberata Ullo, and Luigi Glielmo. A deep q-learning based approach applied to the snake game. In *2021 29th Mediterranean Conference on Control and Automation (MED)*, pages 348–353. IEEE, 2021.

[2] C Daniel Salzman, Kenneth H Britten, and William T Newsome. Cortical microstimulation influences perceptual judgements of motion direction. *Nature*, 346(6280):174–177, 1990.

[3] Chris M Bird and Neil Burgess. The hippocampus and memory: insights from spatial processing. *Nature reviews neuroscience*, 9(3):182–194, 2008.

[4] Joel Z Leibo, Cyprien de Masson d'Autume, Daniel Zoran, David Amos, Charles Beattie, Keith Anderson, Antonio García Castañeda, Manuel Sanchez, Simon Green, Audrunas Gruslys, et al. Psychlab: a psychology laboratory for deep reinforcement learning agents. *arXiv preprint arXiv:1801.08116*, 2018.

[5] Eero P. Simoncelli and David J. Heeger. A model of neuronal responses in visual area MT. *Vision Research*, 38(5):743–761, March 1998.

[6] Stephanie C. Y. Chan, Andrew Kyle Lampinen, Pierre H. Richemond, and Felix Hill. Zipfian environments for reinforcement learning. *1st Conference on Lifelong Learning Agents*, abs/2203.08222, 2022.

[7] Linda B Smith, Swapnaa Jayaraman, Elizabeth Clerkin, and Chen Yu. The developing infant creates a curriculum for statistical learning. *Trends in cognitive sciences*, 22(4):325–336, 2018.

[8] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International conference on machine learning*, pages 1407–1416. PMLR, 2018.

[9] James C. R. Whittington, Timothy H. Muller, Shirley Mark, Guifen Chen, Caswell Barry, Neil Burgess, and Timothy Edward John Behrens. The tolman-eichenbaum machine: Unifying space and relational memory through generalization in the hippocampal formation. *Cell*, 183:1249 – 1263.e23, 2019.

[10] Mallory Snow. Biological plausibility in modern hopfield networks. Master's thesis, University of Waterloo, 2022.

[11] Alexander Pritzel, Benigno Uria, Sriram Srinivasan, Adrià Puigdomènech Badia, Oriol Vinyals, Demis Hassabis, Daan Wierstra, and Charles Blundell. Neural episodic control. *CoRR*, abs/1703.01988, 2017.

[12] Dmitry Krotov and John J Hopfield. Dense associative memory for pattern recognition. *Advances in neural information processing systems*, 29, 2016.

[13] George Kingsley Zipf. *The psycho-biology of language: An introduction to dynamic philology*. Routledge, 2013.

[14] Dharshan Kumaran, Demis Hassabis, and James L McClelland. What learning systems do intelligent agents need? complementary learning systems theory updated. *Trends in cognitive sciences*, 20(7):512–534, 2016.

[15] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA, 2018.

[16] William T Newsome, Kenneth H Britten, C Daniel Salzman, and J Anthony Movshon. Neuronal mechanisms of motion perception. In *Cold Spring Harbor symposia on quantitative biology*, volume 55, pages 697–705. Cold Spring Harbor Laboratory Press, 1990.

[17] Alberto Maria Metelli, Matteo Papini, Nico Montali, and Marcello Restelli. Importance sampling techniques for policy optimization. *Journal of Machine Learning Research*, 21(141):1–75, 2020.

[18] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[19] Audrey Houillon, RC Lorenz, Wendelin Böhmer, MA Rapp, Andreas Heinz, J Gallinat, and Klaus Obermayer. The effect of novelty on reinforcement learning. *Progress in brain research*, 202:415–439, 2013.

[20] Mingqi Yuan. Intrinsically-motivated reinforcement learning: A brief introduction. *arXiv preprint arXiv:2203.02298*, 2022.

[21] Timothy D. Hanks and Christopher Summerfield. Perceptual decision making in rodents, monkeys, and humans. *Neuron*, 93(1):15–31, January 2017.

[22] Chi-Tat Law and Joshua I Gold. Reinforcement learning can account for associative and perceptual learning on a visual-decision task. *Nature Neuroscience*, 12(5):655–663, April 2009.

[23] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4):115–133, December 1943.

[24] David E Rumelhart, James L McClelland, PDP Research Group, et al. *Parallel distributed processing*, volume 1. IEEE New York, 1988.

[25] Gwangsu Kim, Jaeson Jang, Seungdae Baek, Min Song, and Se-Bum Paik. Visual number sense in untrained deep neural networks. *Science Advances*, 7(1), January 2021.

[26] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6):26–38, 2017.

[27] Praveen K. Pilly and Aaron R. Seitz. What a difference a parameter makes: A psychophysical comparison of random dot motion algorithms. *Vision Research*, 49(13):1599–1612, 2009.

[28] H Francis Song, Guangyu R Yang, and Xiao-Jing Wang. Reward-based training of recurrent neural networks for cognitive and value-based tasks. *Elife*, 6:e21492, 2017.

[29] Kenneth H Britten, Michael N Shadlen, William T Newsome, and J Anthony Movshon. The analysis of visual motion: a comparison of neuronal and psychophysical performance. *Journal of Neuroscience*, 12(12):4745–4765, 1992.

[30] Charles Beattie, Joel Z Leibo, Denis Teplyashin, Tom Ward, Marcus Wainwright, Heinrich Küttler, Andrew Lefrancq, Simon Green, Víctor Valdés, Amir Sadik, et al. Deepmind lab. *arXiv preprint arXiv:1612.03801*, 2016.

[31] Bernhard Treutwein. Adaptive psychophysical procedures. *Vision research*, 35(17):2503–2522, 1995.

[32] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[33] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR, 2016.

[34] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[35] Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.

[36] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Icml*, 2010.

[37] Charles Blundell, Benigno Uria, Alexander Pritzel, Yazhe Li, Avraham Ruderman, Joel Z Leibo, Jack Rae, Daan Wierstra, and Demis Hassabis. Model-free episodic control, 2016.

[38] Steven Hansen, Pablo Sprechmann, Alexander Pritzel, André Barreto, and Charles Blundell. Fast deep reinforcement learning using online adjustments from the past, 2018.

[39] Meire Fortunato, Melissa Tan, Ryan Faulkner, Steven Hansen, Adrià Puigdomènech Badia, Gavin Buttimore, Charles Deck, Joel Z Leibo, and Charles Blundell. Generalization of reinforcement learners with working and episodic memory. *Advances in neural information processing systems*, 32, 2019.

[40] James L McClelland, Bruce L McNaughton, and Randall C O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 102(3):419, 1995.

[41] Zhihan Zhou, Jiangchao Yao, Yan-Feng Wang, Bo Han, and Ya Zhang. Contrastive learning with boosted memorization. In *International Conference on Machine Learning*, pages 27367–27377. PMLR, 2022.

[42] Ajay Kumar Boyat and Brijendra Kumar Joshi. A review paper: noise models in digital image processing. *arXiv preprint arXiv:1505.03489*, 2015.

[43] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[44] Michael Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. *CoRR*, abs/2004.14990, 2020.

[45] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. *Advances in neural information processing systems*, 29, 2016.

[46] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[47] Alfredo V Clemente, Humberto N Castejón, and Arjun Chandra. Efficient parallel methods for deep reinforcement learning. *arXiv preprint arXiv:1705.04862*, 2017.

[48] Felix Hill, Olivier Tieleman, Tamara von Glehn, Nathaniel Wong, Hamza Merzic, and Stephen Clark. Grounded language learning fast and slow. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[49] Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

[50] Yunchen Pu, Zhe Gan, Ricardo Henao, Xin Yuan, Chunyuan Li, Andrew Stevens, and Lawrence Carin. Variational autoencoder for deep learning of images, labels and captions. *Advances in neural information processing systems*, 29, 2016.

[51] Andrew Lampinen, Stephanie Chan, Andrea Banino, and Felix Hill. Towards mental time travel: a hierarchical memory for reinforcement learning agents. *Advances in Neural Information Processing Systems*, 34:28182–28195, 2021.

[52] Joseph O'Neill, Barty Pleydell-Bouverie, David Dupret, and Jozsef Csicsvari. Play it again: reactivation of waking experience and memory. *Trends in neurosciences*, 33(5):220–229, 2010.

[53] Adrian J Duszkiewicz, Colin G McNamara, Tomonori Takeuchi, and Lisa Genzel. Novelty and dopaminergic modulation of memory persistence: a tale of two systems. *Trends in neurosciences*, 42(2):102–114, 2019.

[54] Alexandr Ten, Pramod Kaushik, Pierre-Yves Oudeyer, and Jacqueline Gottlieb. Humans monitor learning progress in curiosity-driven exploration. *Nature communications*, 12(1):1–10, 2021.

[55] Matthias J Gruber and Charan Ranganath. How curiosity enhances hippocampus-dependent memory: the prediction, appraisal, curiosity, and exploration (pace) framework. *Trends in cognitive sciences*, 23(12):1014–1025, 2019.

[56] Jerry A Fodor and Zenon W Pylyshyn. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28(1-2):3–71, 1988.

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

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