EMERGENCE OF PREDICTION IN DELAYED REACHING TASK THROUGH NEUROEVOLUTION

An Engineering Honors Thesis

by

WILLIAM KANG[1], CHRIS ANAND[2]

Submitted at
Texas A&M University
in partial fulfillment of the requirements for the designation as an

ENGINEERING HONORS IN COMPUTER SCIENCE AND ENGINEERING

Approved by
Faculty Research Advisor: Dr. Yoonsuck Choe

December 2023

Major: Computer Science[1]
Computer Science[2]

RESEARCH COMPLIANCE CERTIFICATION

Research activities involving the use of human subjects, vertebrate animals, and/or biohazards must be reviewed and approved by the appropriate Texas A&M University regulatory research committee (i.e., IRB, IACUC, IBC) before the activity can commence. This requirement applies to activities conducted at Texas A&M and to activities conducted at non-Texas A&M facilities or institutions. In both cases, students are responsible for working with the relevant Texas A&M research compliance program to ensure and document that all Texas A&M compliance obligations are met before the study begins.

We, William Kang[1], and Chris Anand[2], certify that all research compliance requirements related to this thesis have been addressed with our Faculty Research Advisor Dr. Yoonsuck Choe prior to the collection of any data used in this final thesis submission.

This project did not require approval from the Texas A&M University Research Compliance & Biosafety office.

TABLE OF CONTENTS

ABSTRACT

Emergence of Prediction in Delayed Reaching Task Through Neuroevolution

William Kang[1] and Chris Anand[2]
Department of Computer Science and Engineering[1]
Department of Computer Science and Engineering[2]
Texas A&M University

Faculty Research Advisor: Dr. Yoonsuck Choe
Department of Computer Science and Engineering
Texas A&M University

Prediction is an important foundation of cognitive and intelligent behavior. Recent advances in deep learning heavily depend on prediction, in the form of self-supervised learning based on prediction and reinforcement learning (reward prediction). However, how such predictive capabilities emerged from simple organisms has not been investigated fully. Prior works have shown the relationship between input delay and predictive function to compensate for such delay.

In this thesis, we investigate the emergence of predictive capabilities in simple evolving neural network controllers, where not only the connection weights but also the network topology evolves. We focus on two main research questions: (1) what fitness criterion promotes predictive behavior? and (2) what changes in the neural network structure correlate with predictive function? To test this, we set up a delayed reaching task, where a two-segment arm is controlled to reach a moving target where the target's location arrives at the arm's controller with a period of delay. The arm also has an option of picking up a stick (a tool) to extend its reach. We tested several factors to be included in the fitness function: (1) energy usage, (2) tracking target, and (3) number of tool pick-ups. Our results show that minimizing energy usage is a key to the emergence of

prediction. As for the evolved network structure, we found that controllers with more recurrent loops perform better in the task, i.e., tracking the predicted location of the moving target. These results lend us two important insights regarding the evolutionary emergence of prediction: (1) energy minimization is a key driving force, without which random strategy (wasteful in terms of energy) can potentially perform equally, and (2) recurrent loops in the neural network controller not only play the traditional role of memory, but they also serve the purpose of prediction. We expect our results to shed new light on the origin of neural architectures supporting prediction and the energetic constraints.

DEDICATION

To our families, professor, and peers who supported us throughout the research process.

ACKNOWLEDGMENTS

Contributors

We would like to thank our faculty advisor, Dr. Yoonsuck Choe, for his guidance and support throughout the course of this research.

Thanks also go to our friends and colleagues and the department faculty and staff for making our time at Texas A&M University a great experience.

Finally, thanks to our families for their encouragement, patience, and love.

The NEAT algorithm used for this research was originally developed by Kenneth O. Stanley and Risto Miikkulainen. The Java implementation of NEAT (ANJI) we used was developed by Derek James and Philip Tucker. The implementation of the task environment in Java was by Qinbo Li. We would also like to thank Jiyoon Hwang and Vijay Seetharam for their ideas, discussions, and efforts in the research.

All other work conducted for the thesis was completed by the students independently.

Funding Sources

No funding was provided for this research.

NOMENCLATURE

NEAT	NeuroEvolution of Augmenting Topologies
E	Energy Factor
T	Tool Factor
R	Reach Factor; also used as fitness function with sole factor
ER	Fitness function of E $\times$ R
RT	Fitness function of R $\times$ T
ERT	Fitness function of E $\times$ R $\times$ T

1. INTRODUCTION

In our introduction, we cover our focus of research and give some clarification as to why it might be interesting.

1.1 Research Focus

Prediction is fundamental to many species allowing animals to hunt food and find shelter. Primitive animals functioned reactively with little memory of the past [1], similar to a feed-forward neural network (FNN). Through natural selection, memory persisted as a crucial part of their survival. At first, memory may have used external markers such as pheromones for ants. Later animals internalized, or "privatized", memories that led to the anticipation of the present based on the past or prediction of the future based on the present [2]. Such phenomena were promoted with reward systems such as food and safety in the "survival of the fittest" environment. This form of internalized memory is similar to that of recurrent neural networks (RNN).

For instance, Rats are known for their evolved cognitive map and ability to understand the position of their target in space, as opposed to blank memorization of the path [3]; with cognitive maps, rats can find the most efficient path to their target. As Humphrey suggested, one of the most promising theories for the possibility of spatial cognition is an internal memory system represented in neural networks as recurrent neural networks [4]. Previous research has shown that networks with high internal state predictability (ISP) performed better than networks with low ISP in a novel pole-balancing task [5]. In our research, we further expand on this research, by increasing the focus from spatial cognition to spatiotemporal cognition and analyzing key evolutionary constraints and the neural network's topological traits responsible for prediction in a delayed control task environment.

To expand the scope of our research, we introduce delays to our agents which are also commonly found in nature. According to Nijhanwan and Wu, "neural delays, rather than being a passive consequence of the anatomy and physiology of the nervous system, are a result of a basic

feature and active strategy used by the nervous system for information processing" [6]. For example, the flash-lag effect (FLE), a phenomenon where the position of a moving object is perceived ahead of the current state due to changing stimuli, shows that animals account for the delay due to their nervous system through prediction [7]. Due to its prevalence, it raises the question of how the neural system can compensate for the delayed information. To answer this question, we utilized NeuroEvolution of Augmenting Topologies (NEAT) to evolve chromosomes in an environment with delayed input which are then analyzed to find evidence for constraints and features responsible for the compensation of delayed information.

1.2 The NeuroEvolution of Augmenting Topologies Algorithm (NEAT)

To evolve the neural networks, we used a genetic algorithm called NEAT [8]. NEAT is an algorithm that trains a neural network for both the weights of the network and the topology of the network. NEAT can be configured in many different ways such as allowing recurrent loops inside the network which was quintessential to the experiment.

We used NEAT for this experiment rather than other types of learning algorithms for two main reasons. The first is that NEAT allows for the analysis of the network during the evolution of the topology of the network. That allows for the observation of various traits of the network during evolution. Secondly, NEAT has been proven to be efficient in reinforcement learning. NEAT also allows for newly evolved species to find niches in the environment and determine the viability of the potential path they have taken. This mechanism is known as speciation and allows for more variety in the chromosomes evolved through training rather than settling for a uniform, locally optimal population.

1.3 Approach

This paper investigates the evolutionary aspect of prediction in a neural network tasked with controlling a two-degree-of-freedom articulated limb. Employing a modified version of an existing task [9], our experiment involves a dynamic setup comprising a limb, a tool, and a moving target object. In the previous work, the target was stationary. Furthermore, in our experiments the input was delayed, so that the delayed target location was provided to the agent. The agent's mission is

to "catch" the moving target by predicting its trajectory, utilizing a tool (a stick) to extend its reach as needed if the target moves out of the limb's reach. This exploration aims to uncover the nuances of predictive evolution in neural networks as previous work [10] has shown delay forces the neural network to evolve prediction. We tested three main factors in our evolved neural networks using their fitness functions: energy usage, number of tool pick up, and frequency of touching the target. Our metrics involve the number of loops developed in the neural network topology, fitness based on fitness functions, and success rates. We discuss the metrics further in chapter 2.

Our results indicate the agents with the ability to minimize their energy usage are more successful than other networks without the ability to account for energy. This indicates that the ability to account for energy allows for the network to evolve more efficient methods to track the true target given the delayed location, and in the process, exhibit predictive behavior.

We also discovered networks involving a greater amount of recurrent loops have a higher success rate. This indicates that there is a strong correlation between memory and prediction in neural networks as recurrent loops are considered to serve as memory of past events in neural networks. This is in line with findings of memory being necessary for prediction in humans as explained in the memory-framework theory [11].

1.4 Thesis Structure

We organize the rest of this thesis as follows: details of our methods and the task given to our agent (chapter 2), results and analysis of experiments (chapter 3), discussion of future works and possible implications (chapter 4), and conclusion of work (chapter 5).

2. METHODS

In this section, we discuss the details of the experiment: changes to the environment, different fitness functions used to evolve the agents, and performance metrics.

2.1 Modified Environment

The original task/environment in [9] focused on the agent's ability to reach targets outside its reach as efficiently as possible. To research the key features of a network during success cases, the environment was set up as follows: 8 input nodes, 2 output nodes, a static target, a static tool, and two semi-circles of which the radii of the semi-circles corresponded to the sum of both arm components and a sum of both arm components + the length of the tool (l1 + l2, l1 + l2 + l3). An example of the original environment can be seen in Figure 1. In each trial, the agent controlled the two arm components centered at the bottom center of the semi-circles, joined together to form figures similar to that of a human arm. The agents would find the optimal way to find the randomly placed tool and ultimately the target. As the original task intended to perform analysis on the network's efficiency, each trial was performed for a maximum of t timesteps:

$$t = min(MAX\_TRIAL\_LENGTH, time\_step\_to\_reach\_target).$$

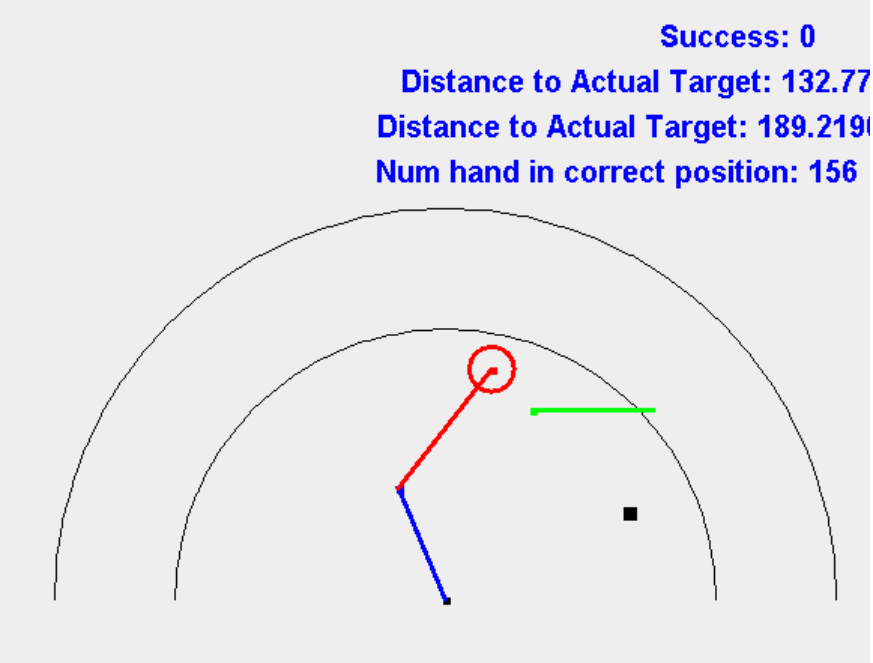

Figure 1: Original tool-use task environment. Static target and tool available to the agent [9].

While the original environment provided sufficient challenges to tool use, it lacked spatiotemporal challenges that would induce prediction. As a result, we have modified the environment accordingly:

1. Add a 9th input node, a touch sensor.
2. Replace the static target with a moving target (3 directions).
3. Provide delayed information (500 timesteps).
4. Add a time buffer to let target move before the agent (500 timesteps).
5. Run each trial to the MAX_TRIAL_LENGTH.

These changes together, would bolster the need for prediction, especially in the light of genetic algorithms such as NEAT.

By replacing the static target with a moving target and only providing delayed information on the target's positions and angles, the agent can no longer rely on information from a single time frame, but must collect, reason, and act based on a series of data points in the past. To further enhance the network's capabilities to collect and reason about the data, we have added a time buffer at the beginning of each trial to allow the networks to observe the movement of the targets and base their actions accordingly.

To make sure that we collect accurate data, each trial was extended to run a constant MAX_TRIAL_LENGTH amount of timesteps. Using the original method of ending the trial as soon as the target is reached, there would be no way to clearly distinguish a successful predictive trial from a "lucky" trial in which the agent would thoughtlessly swerve as much area as it can to hopefully hit the target. Extending our trial length helps us determine if the agent was persistent, indicating a successful predictive behavior, or if the result was out of pure luck. The final change to the environment was the addition of the 9th input node, a touch sensor. As the scope of our research expanded to the persistence of the agent on the target, the touch sensor provides critical and useful information that helps the network act more fittingly into our criteria. An example of the modified environment can be seen in the Figure 2.

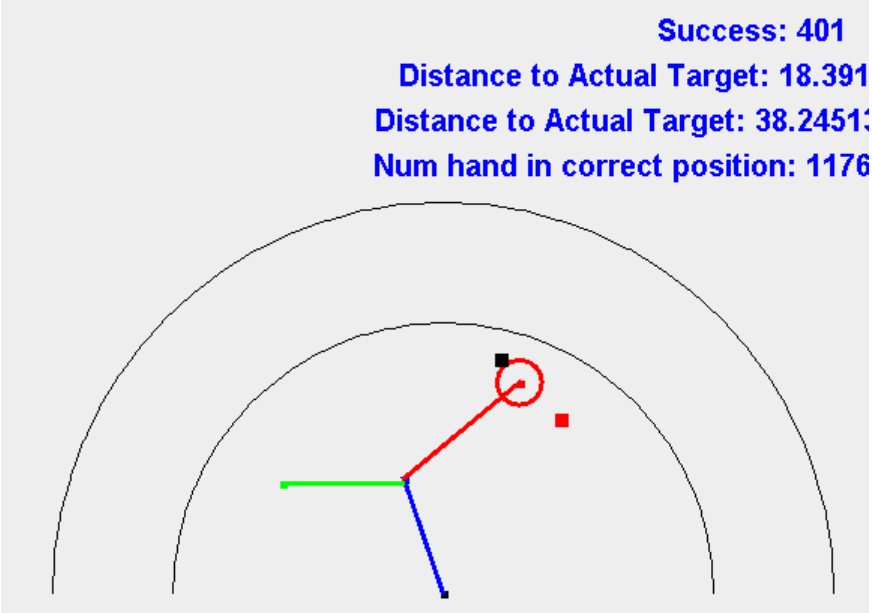

Figure 2: Modified tool-use task environment. Moving target (shown by black square but not used as input to the agent), target positions inputted to the agent (shown by the red square), and tool available to the agent. Here, the target is moving from the lower right to the upper left.

2.2 Fitness Functions

Using a genetic algorithm such as NEAT, fitness functions define the success criteria of the networks thereby shaping the behavior of the agents. Hence, after modifying our environment, we tested 4 different fitness functions, testing what information they bring and how each factor may affect the behavior of the agents.

The fitness functions are formed in the format of: $F = \alpha \times P$; where α is the constant set of factors found critical and was commonly used for all fitness functions and P is the permutation of factors E (energy), R (predictive reaches - percent of trials with 5 or more consecutive reaches), and T (tool use).

The α component consists of the average MSE (Mean Squared Error) between the target and the hand, Turn Factor which is a penalty given to discourage repeated oscillations, and Count Factor which holds the information of the relative number of reaches in each trial. Although other

combinations of different factors may result in similar behavior, in the scope of our research, these three factors together provided the fundamental information to the agents that could be used as our control group.

2.3 Performance Metrics

As the design of the experiment changes the fitness functions, using the fitness scores alone does not provide enough context nor does it serve as a standard metric to compare different evolutions. Hence, we have derived multiple performance metrics that help us gain insight into the networks' success, especially regarding predictability.

The first metric is the absolute number of reaches. Since the goal of the agents can be abstracted to touching the target by prediction, the number of reaches serves as a general measurement of how well the network performs. The number of reaches is computed with equation 1.

$$number\ of\ reaches\ =\ \sum_{n=1}^{num\_trials}\ number\ of\ reaches\ in\ trial\ n \tag{1}$$

Metric 1_hit measures the number of trials in which the agent had reached the target at least once. The 1_hit is computed with equation 2. While this metric is standardized and can show the effectiveness of a network, it cannot distinguish between an accidental hit of the target compared to a purposeful reach.

$$1\_hit\ =\ \sum_{n=1}^{num\_trials}\ x = \begin{cases} 1 & \text{if hit target in trial } n \\ 0 & \text{else} \end{cases} \tag{2}$$

Hence, we also introduce the 5_hit metric which counts the number of trials in which the agent had reached the target at least 5 consecutive timesteps. This is to ensure that the reach of a target was intentional and therefore is used as our main metric to measure the predictive capabilities of the agents. The metric is computed with equation 3.

$$5\_hit = \sum_{n=1}^{num\_trials} x = \begin{cases} 1 & \text{if hit target for 5+ consecutive steps in trial } n \\ 0 & \text{else} \end{cases} \tag{3}$$

All the metrics mentioned above apart from the fitness scores, act as a standard metric to compare the successes of different agents. Unlike fitness scores, other metrics share common extremes (0 – 500,000 for the number of reaches and 0 – 100 for 1_hit or 5_hit) and are computed using the same equations resulting in standardized scores.

c

3. RESULTS & ANALYSIS

In this section, we discuss our findings for our 4 fitness functions (ERT, ER, RT, and R). We first discover statistics and insights in predictive behavior by comparing different success criteria (fitness score, number of reaches, number of trials with 1 or more hits out of 100 trials, number of trials with 5 or more consecutive hits out of 100 trials) as it brings more context to otherwise less interpretable numbers. Then, we further test and prove the significance of some factors to its conclusion.

The statistics are computed for each fitness function for each of the successes. The total number of runs (where 1 run = a set of 100 individual trials) to compute the statistics was 31 $(n = 31)$.

3.1 Performance Metrics vs Generations

The fitness functions of ERT, ER, and R have evolved exactly 150 generations. The fitness function of ER (color green) has evolved 120 generations. There exists a threshold within NEAT of which if the fitness function is above the threshold, t, the evolution stops. In the case of ER, the 120^{th} evolution crossed the threshold, halting its evolution.

We first measure the average **fitness score per generation** for each fitness function. As per Figure 3, we find that the fitness function of ER greatly outperforms any other fitness function, especially after 30 generations. However, as we see by comparing other performance metrics, performance comparisons between fitness functions should not be made with Figure 3 alone as the evaluation functions themselves are different. Rather, we should look at the trends within each class and draw comparisons among the trends.

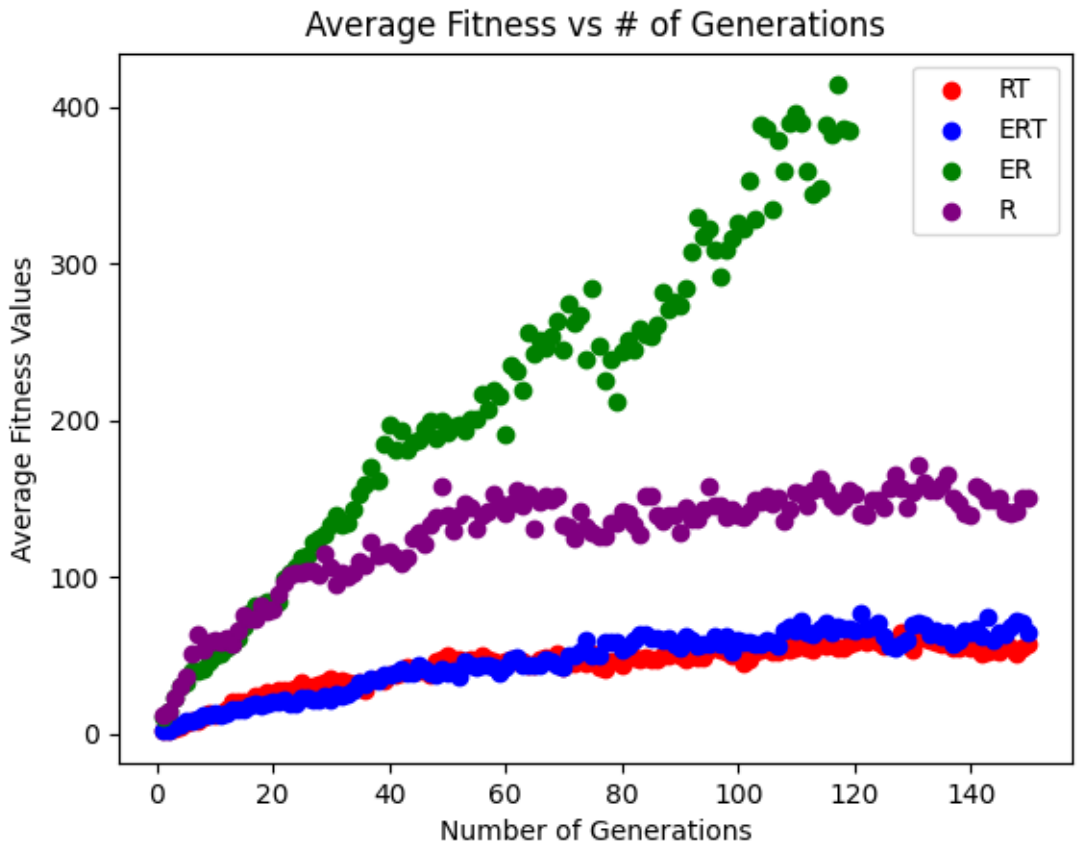

Figure 3: The average fitness values for each fitness function across the generations. The average was found between the 100 trials computed per generation

Fitness functions RT and ERT follow very similar growth patterns of a gentle increase in performance over time. Fitness function R follows a log-like curve in which there are large increases in performance in earlier generations but converges to a value at around 60 generations. These three fitness functions show convergence to a single value as the number of generations increases. The function ER shows a very different pattern; the average fitness values continue to increase at a steep rate even suggesting if the threshold were to be larger, the fitness value would continue to increase. This suggests that the function ER continues to improve its strategy and when compared to other functions that converge at around or before the 60^{th} generation, we expect the performance and predictability of ER to be greater than other functions.

We then measure the average **number of reaches per generation** for each fitness function. Figure 4, is resemblant to Figure 3. The functions ERT, ER, and R are closer together in the lower bounds of the plot while the function ER differentiates itself from the group at around 30 generations. Inferring from striking similarity, we justify the behavior of the fitness function is derived from the absolute number of reaches especially because the alpha component includes the normalized variable of the number of reaches.

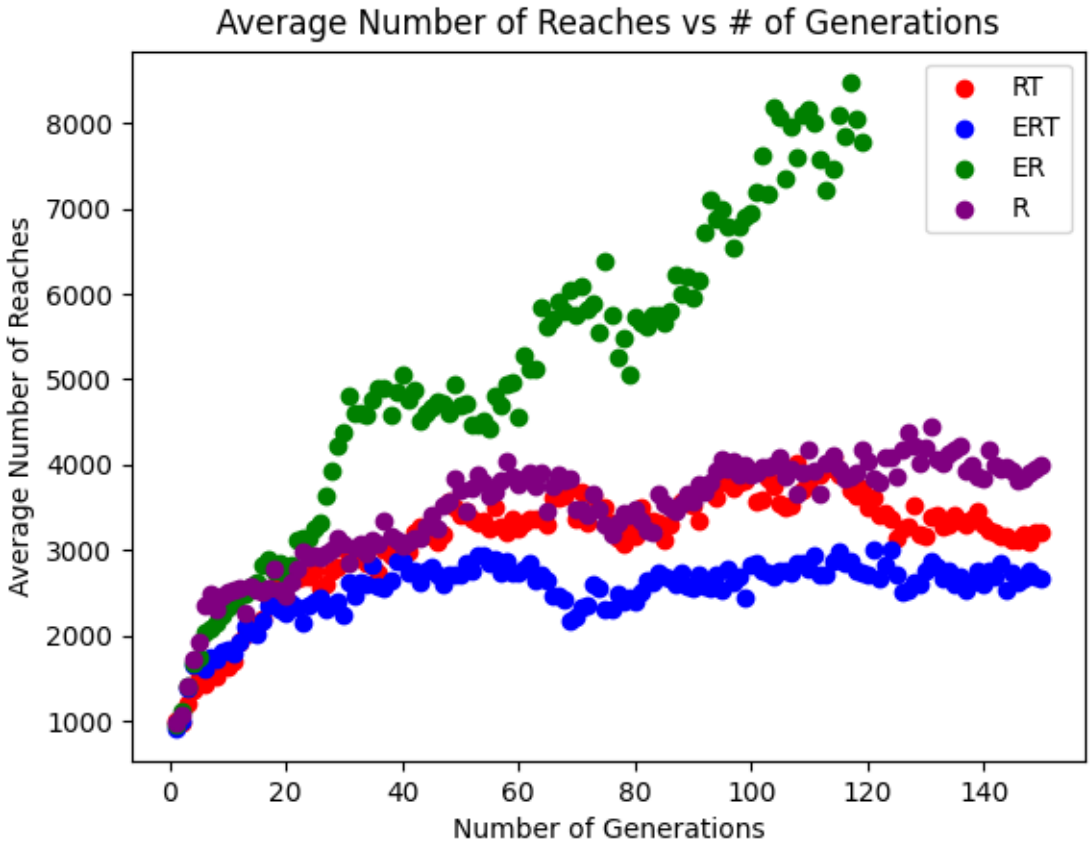

Figure 4: The average number of reaches for each fitness function across the generations. The average was found between the 100 trials computed per generation

While the number of reaches is a standardized measure and can be used as a metric to compare between different fitness classes, we realized a small number of trials could easily manipulate

the data. We have found that there are multiple instances in which a small number of extremely successful cases (i.e. tracking the target for most of the timestep) are influential enough to have a greater number of reaches than those we classified as more successful. To mitigate such issues, we compute the number of trials with a defined success threshold in a single run.

Figure 5 shows the **number of trials with the first defined success threshold of 1_hit over the generations**. Unlike other figures above, we observe that all fitness functions follow a very similar pattern of a log-like curve. In the case of function ER, despite having a much larger total reach, the amount of trials that touched the target at least once is very similar to that of other functions. This first proves our conjecture that small amounts of outliers can lead to a significant difference in the number of reaches criteria but also suggests a prominent sign of prediction.

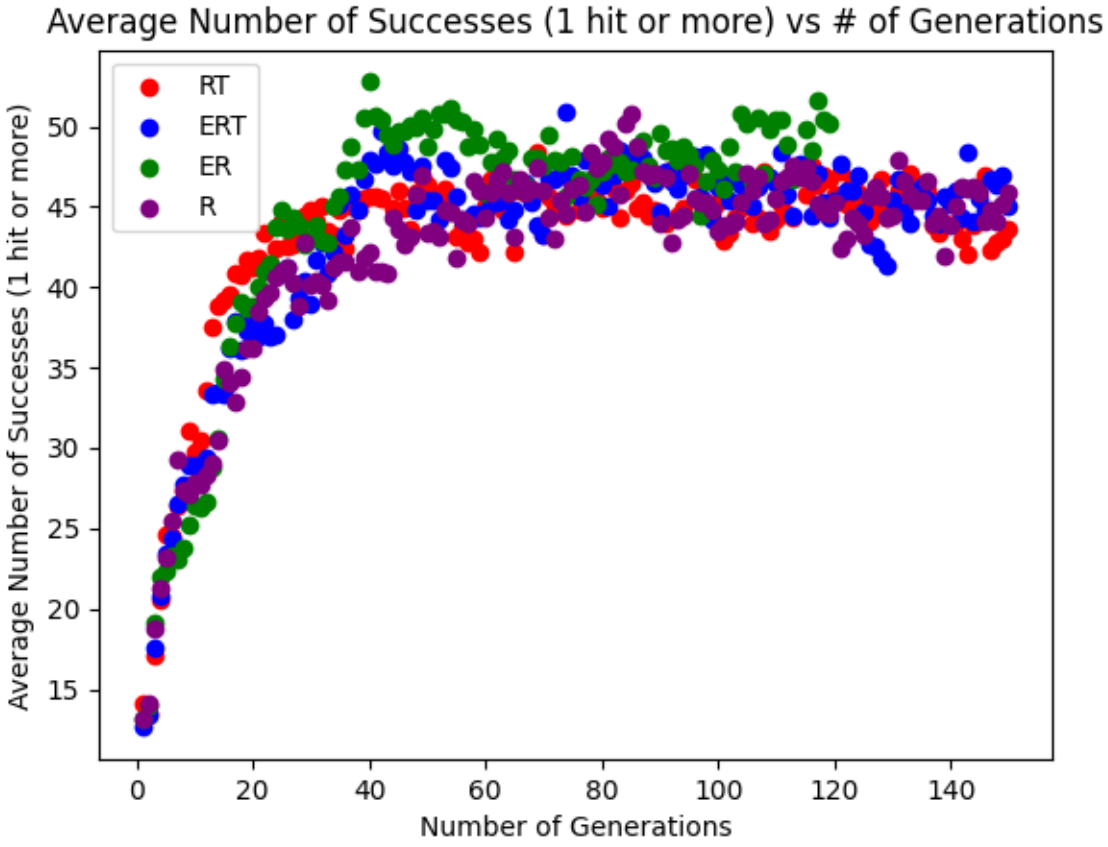

Figure 5: The number of trials with at least 1_hit on target out of 100 trials per generation.

Figure 6 shows the number of trials with the second defined success threshold of 5 or more consecutive hits per generation. The reason for the choice to define a second threshold, especially if the threshold is connected to consecutive hits, and why fitness ER may show signs of prediction is due to the possibility of accidental touches. One of the first expected behaviors in unsuccessful predictive networks was a thoughtless waving motion. Since the abstracted goal was to reach the target, the fitness function had to contain a reward for reaching the target. When configuring the fitness functions improperly, the agents would often swerve around its grounds in an attempt to hit the target as much as possible. Hence, to distinguish between accidental hits from intentional hits, we propose a threshold containing consecutive hits.

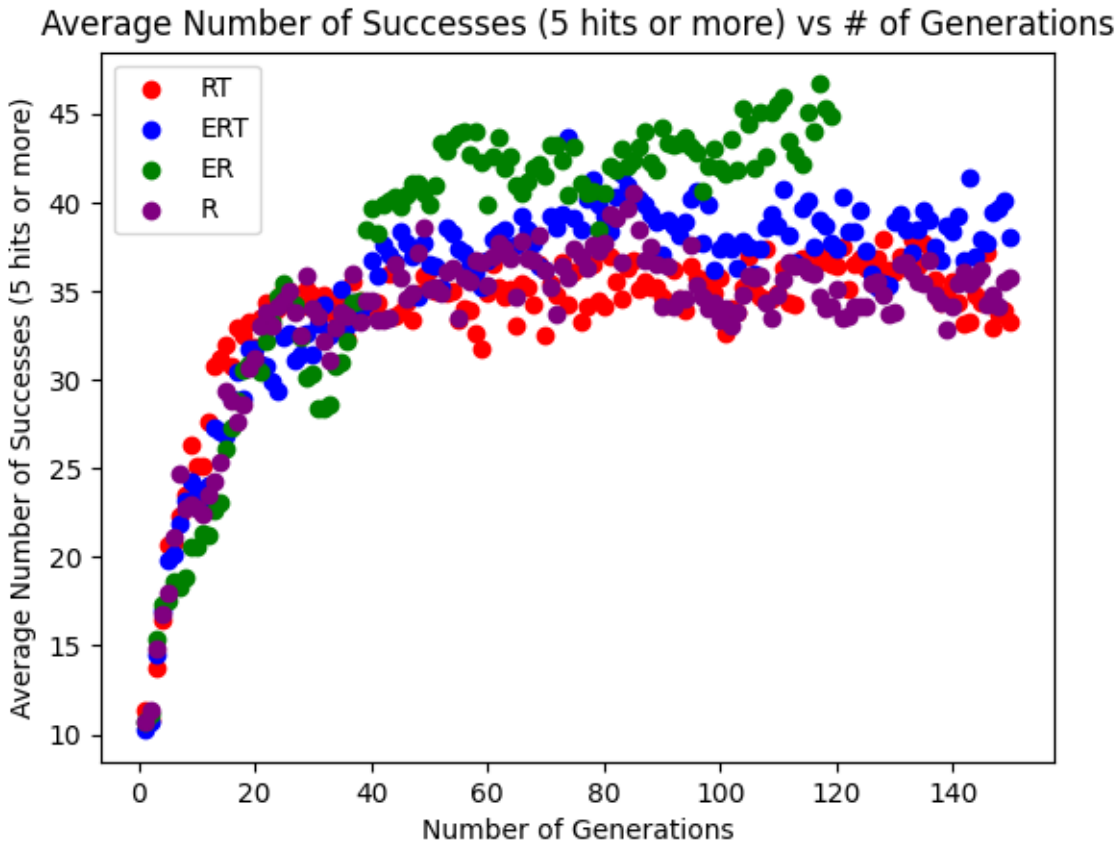

Figure 6: The number of trials with at least 5_consecutive hits on target out of 100 trials per generation.

In Figure 6, we observe a similar pattern to that of Figure 5 except that the fitness classes are more separated, with ER showing the highest value. As this metric measures the predictive capabilities of the network, we expect the predictive capabilities of the networks to follow a similar ranking as those shown in Figure 6.

3.2 Comparing Performance Metrics on Best Chromosomes

In this subsection, we discuss the results and the analysis of the performance metrics based on 31 runs of each fitness function class.

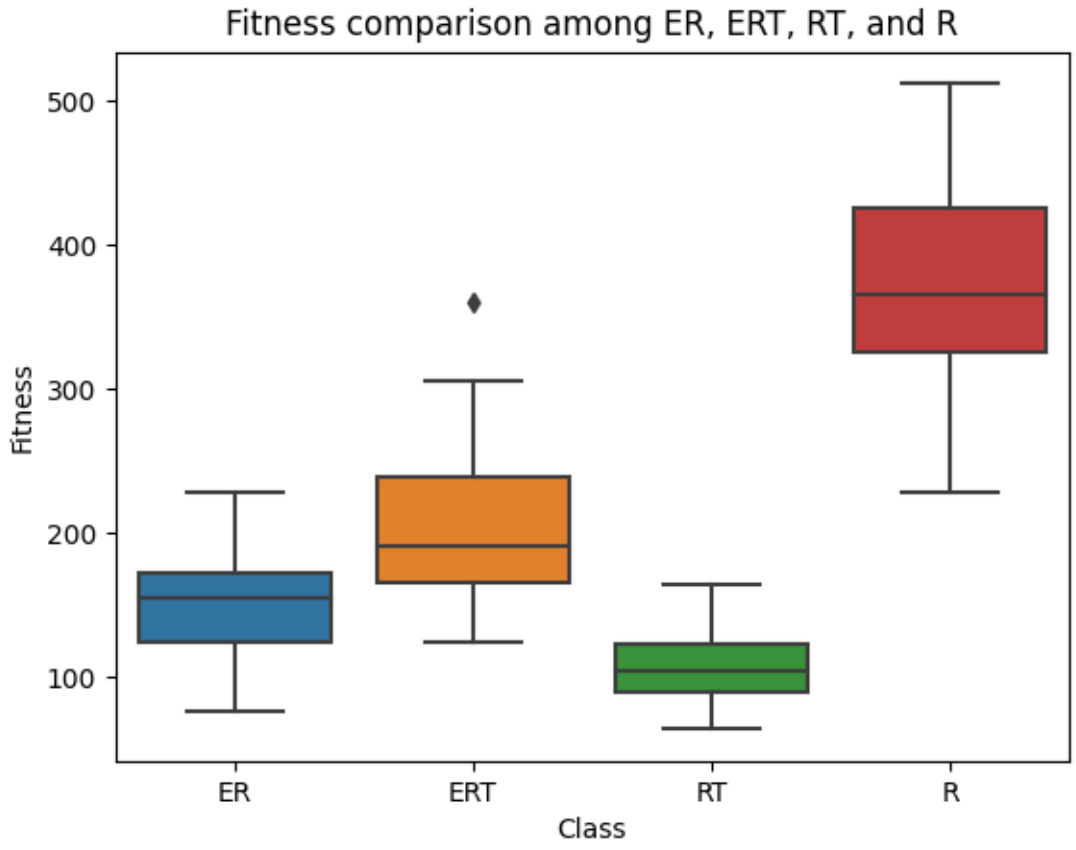

Figure 7: Comparison of fitness values of each fitness function class. The Box-whisker plot shows the mean, range, and standard deviation of each fitness class with trials n = 31.

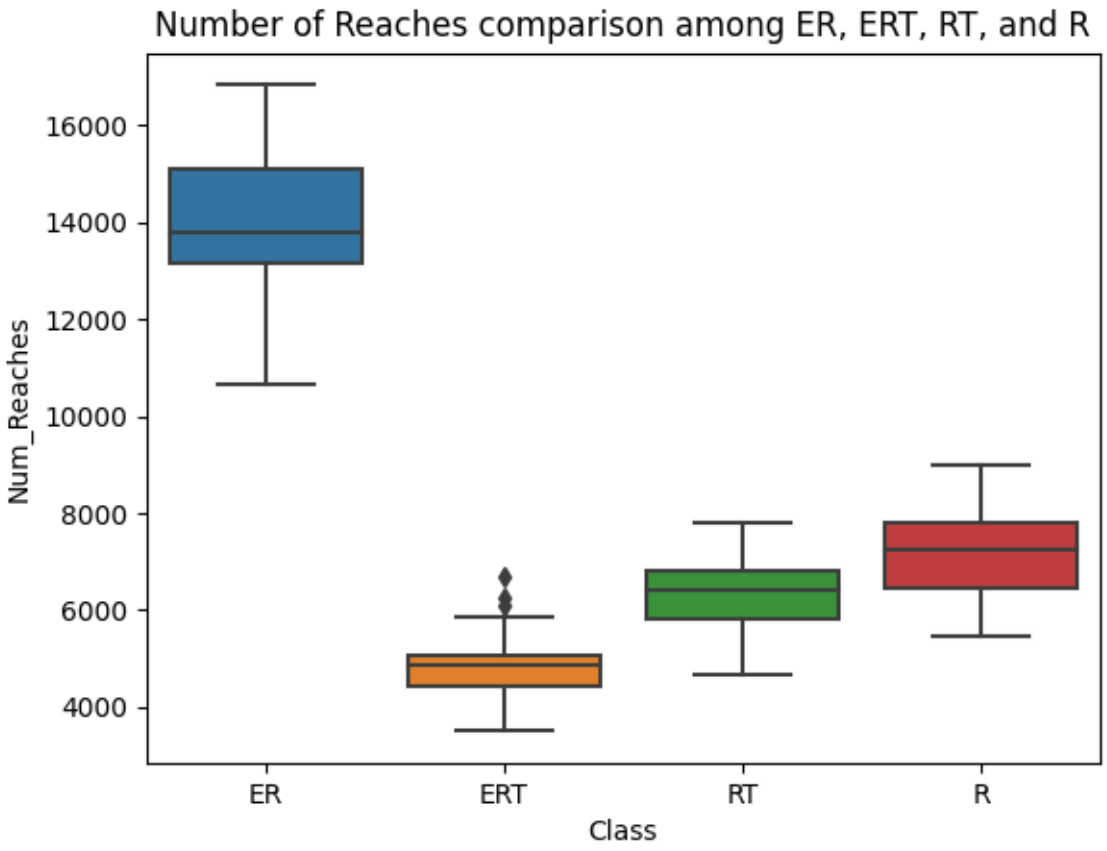

Figure 8: Comparison of the number of reaches of each fitness function class. n = 31.

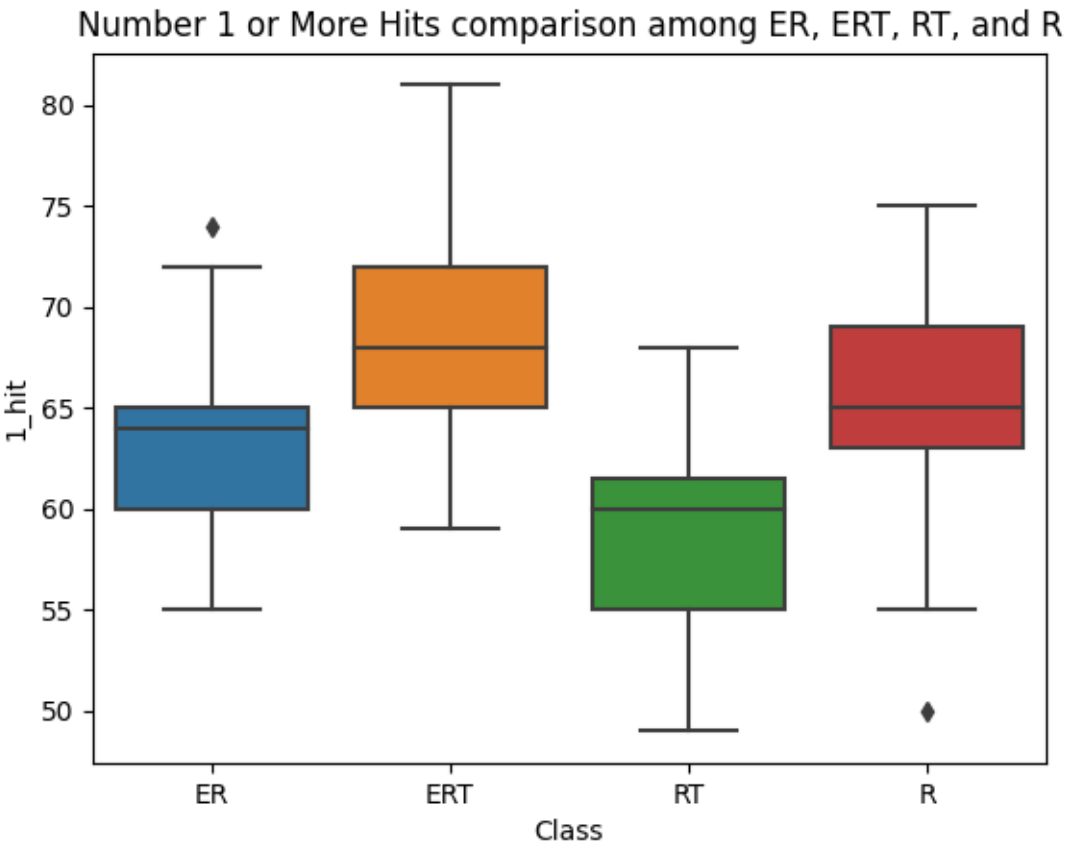

Figure 9: Comparison of the number of at 1_hit of each fitness function class. n = 31.

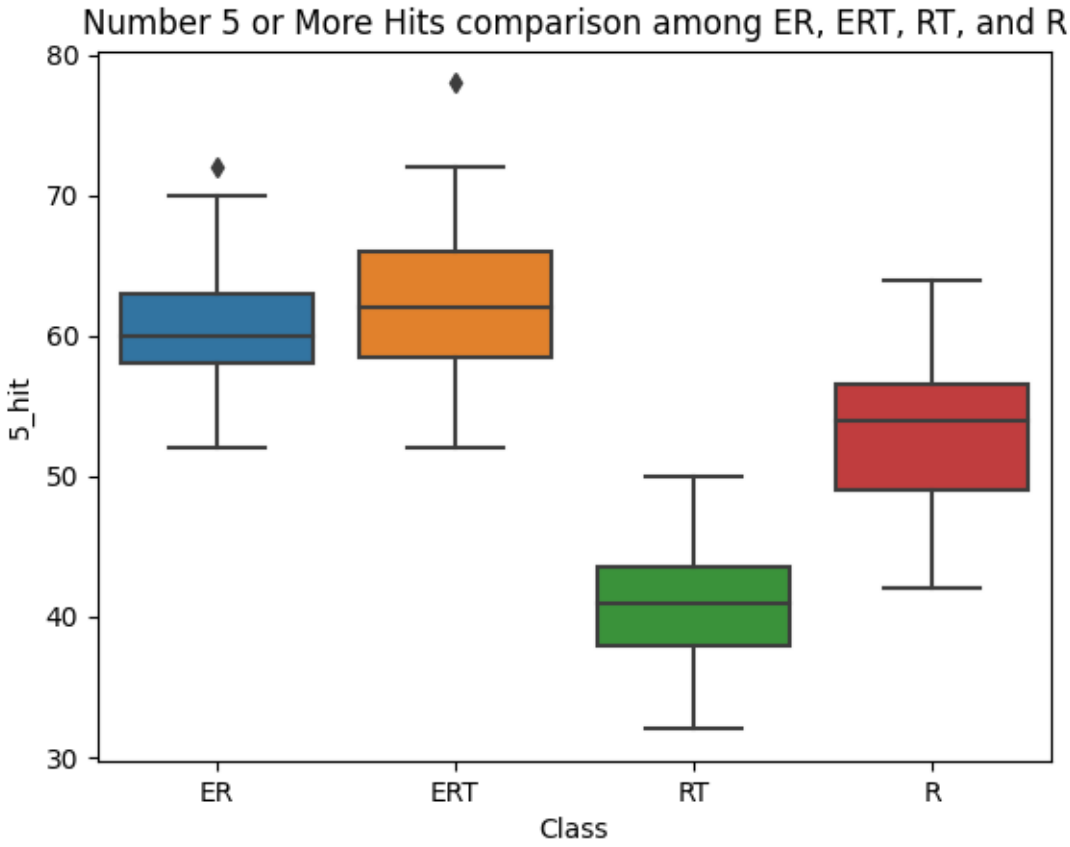

Figure 10: Comparison of the number of 5_hit of each fitness function class. n = 31.

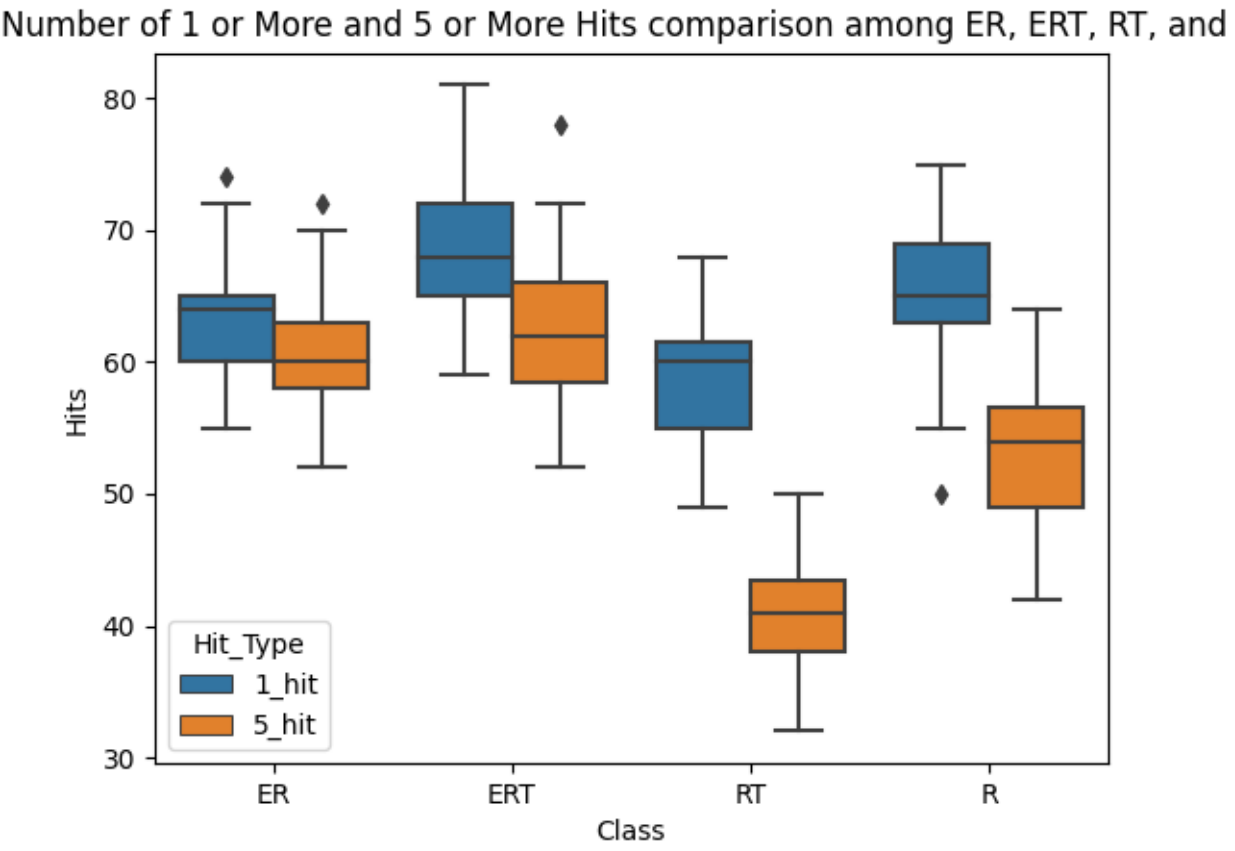

Figure 11: Combination of Figures 9 and 10. n = 31.

Figure 7 summarizes the range of each fitness function class based on their number of reaches. We expected the greatest number of reaches for fitness function R as the fitness function is solely dependent on the predictive reach factor increasing its dominance over the agents' behavior. The trend exists when comparing the plots of ERT, RT, and R. However, the fitness function ER showed much greater total reaches indicating that the energy factor may increase the predictability of an agent.

To further evaluate the hypothesis of energy factor increasing predictability, we can look at the combination of Figures 9, 10, and 11. In Figure 9, which counts the number of trials with at least 1 hit, the number of success trials increases in the order of RT, ER, R, and ERT. However,

looking at Figure 10, we observe that our predictive metric increases in the order of RT, R, ER, then ERT. suggesting that the agent evolved with the fitness function of R hits the target more frequently, but a lot of those reaches are accidental. Such description can be more prominently seen in Figure 11 which compares the metric of Figures 9 and 10. In the figure, the discrepancies between the 1_hit range and the 5_hit range are the largest for the fitness of RT, followed by R, then ERT, and finally ER. The numerical difference can be calculated using the statistics in section 3.4.

We reason such phenomena as following: The energy factor is a penalty factor that increases its punishment the less steady an agent is. In a genetic algorithm, where the fitness score determines its "survival", agents with the energy factor are narrowed into a strategy that involves less movement. In such circumstances, the ideal strategy may be to compute the target's approximate location via prediction and follow the target; this is the case for agents in the ER and ERT classes.

3.3 Statistical Significance

To rigorously test if the energy factor was indeed a significant factor deriving such actions, we used the Mann-Whitney U test to compute the statistical significance of the energy factor's advantage. We chose to use the Mann-Whitney U test as the fitness functions fail the homoscedasticity assumption and we are computing the difference between 2 classes at once: ERT vs RT and ER vs R. The reasoning for the failed homoscedasticity assumption can be computed with the tables in 3.4.

The numeric values of the statistics were rounded to the hundredth digit.

Table 1: Comparing ERT vs RT (The null hypothesis: ERT == RT, alternative hypothesis: ERT != ER). See Figure 11 for the mean and standard deviation.

P-value for 5_hits	1.36e-11

Table 2: Comparing ER vs R (The null hypothesis: RT == R, alternative hypothesis: RT != R). See Figure 11 for the mean and standard deviation.

P-value for 5_hit	9.71e-07

As it can be observed, the p-values for all the success metric is less than the standard alpha of 0.05. Hence, we can determine that there is a significant difference between a fitness function using the energy factor and those who don't and further can say that the energy factor is responsible for the symptoms of predictive capabilities.

3.4 Raw Descriptive Statistics (N = 31)

The numeric values of the statistics were rounded to the hundredth digit.

Table 3: Descriptive Statistics for Fitness = ERT

Mean Fitness	205.25
Median Fitness	191.42
Fitness Standard Deviation	56.47
Mean Number of Reaches	4882.00
Median Number of Reaches	4879.00
Number of Reaches Standard Deviation	700.57
Mean 1_hit	68.55
Median 1_hit	68.00
Standard Deviation 1_hit	5.38
Mean 5_hit	62.97
Median 5_hit	62.00
Standard Deviation 5_hit	6.12

Table 4: Descriptive Statistics for Fitness = ER

Mean Fitness	153.64
Median Fitness	154.19
Fitness Standard Deviation	38.85
Mean Number of Reaches	13892.45
Median Number of Reaches	13790.00
Number of Reaches Standard Deviation	1583.76
Mean 1_hit	63.06
Median 1_hit	64.00
Standard Deviation 1_hit	5.03
Mean 5_hit	60.42
Median 5_hit	60.00
Standard Deviation 5_hit	4.92

Table 5: Descriptive Statistics for Fitness = RT

Mean Fitness	105.94
Median Fitness	103.43
Fitness Standard Deviation	23.98
Mean Number of Reaches	6254.74
Median Number of Reaches	6409.00
Number of Reaches Standard Deviation	763.33
Mean 1_hit	58.35
Median 1_hit	60.00
Standard Deviation 1_hit	4.94
Mean 5_hit	41.00
Median 5_hit	41.00
Standard Deviation 5_hit	4.45

Table 6: Descriptive Statistics for Fitness = R

Mean Fitness	374.24
Median Fitness	366.20
Fitness Standard Deviation	76.79
Mean Number of Reaches	7139.65
Median Number of Reaches	7266.00
Number of Reaches Standard Deviation	1001.36
Mean 1_hit	65.06
Median 1_hit	65.00
Standard Deviation 1_hit	5.36
Mean 5_hit	52.97
Median 5_hit	54.00
Standard Deviation 5_hit	4.98

3.5 Correlation of Hidden Neurons

In this subsection we discuss the time series correlation matrix and the activation charts of the hidden neuron activation patterns.

The three heatmaps here show the time series correlation of the ERT fitness function organized into three categories: predictive, lucky, and fail. In the predictive case the limb was able to accurately predict where the target would be ahead of time and catch the target. For the lucky case the arm was waving around and managed to touch the target by accident for ten or less timesteps for the initial touch then latched onto the path of the target that was considered lucky. Then for fail the limb was wildly waving around with no set path or intended target.

In the predictive case Figure 12 the heatmap shows there are 5 clusters of neurons that are highly correlated. The clusters are as follows: cluster 1 (0,3,8,9,16), cluster 2 (10, 5), cluster 3 (12,18), cluster 4 (19,17), and cluster 5 (7, 2). In the lucky case Figure 13 there are 4 clusters of

neurons that are highly correlated. The clusters are as follows: cluster 1 (2,7), cluster 2 (12,18,19), cluster 3(0,3,8), cluster 4 (10,5). In the failure case Figure 14 there are also 4 clusters. The clusters are as follows: cluster 1 (1, 14, 6), cluster 2(18, 12), cluster 3 (8,0,16), and cluster 4(7,2). As we can see in all three cases there are 3 subclusters that are prevalent in all of them: subcluster 1(7,2), subcluster 2 (12,18), and subcluster 3 (0,8). Additionally there is one shared cluster between the lucky and predictive case which consist of (10,5) We can hypothesize that those 4 subclusters are necessary for the movement of the limb or any other basic computation that all three cases require. However, we can observe the unshared clusters in the predictive case to discover the hidden neurons that are working to successfully predict. With that knowledge we can distinguish the main unshared cluster (17,19) and hypothesize that there are predictive capabilities in the ERT function using those clusters, which we will discuss further in chapter 5.

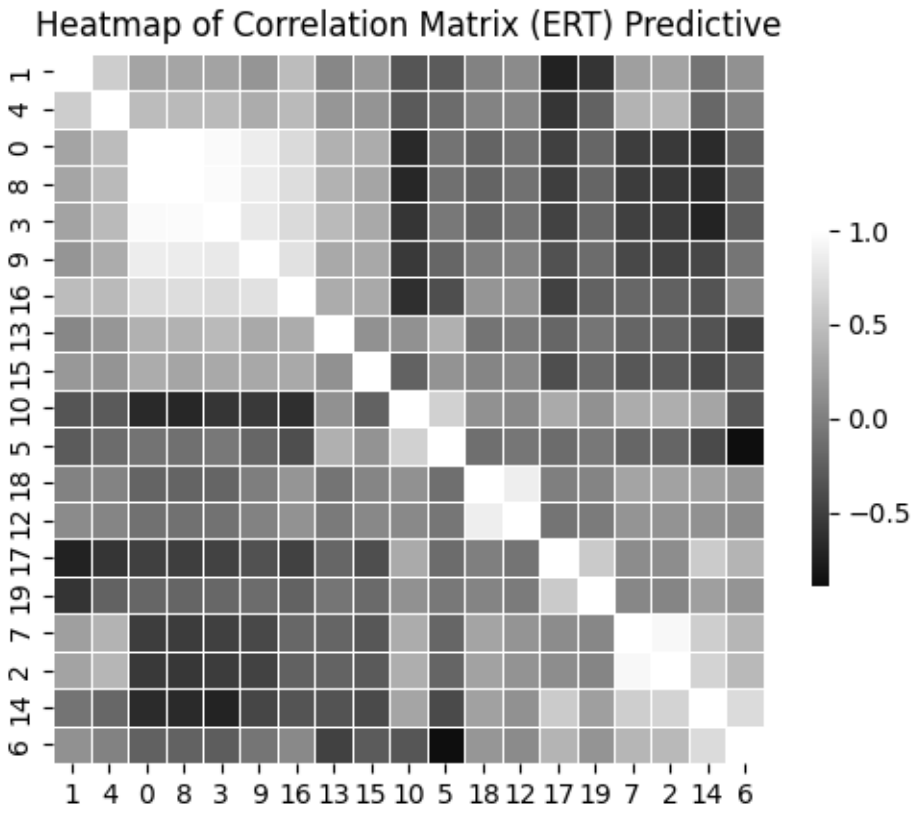

Figure 12: Correlation matrix of hidden neuron activations in ERT in a trial that showcased prediction.

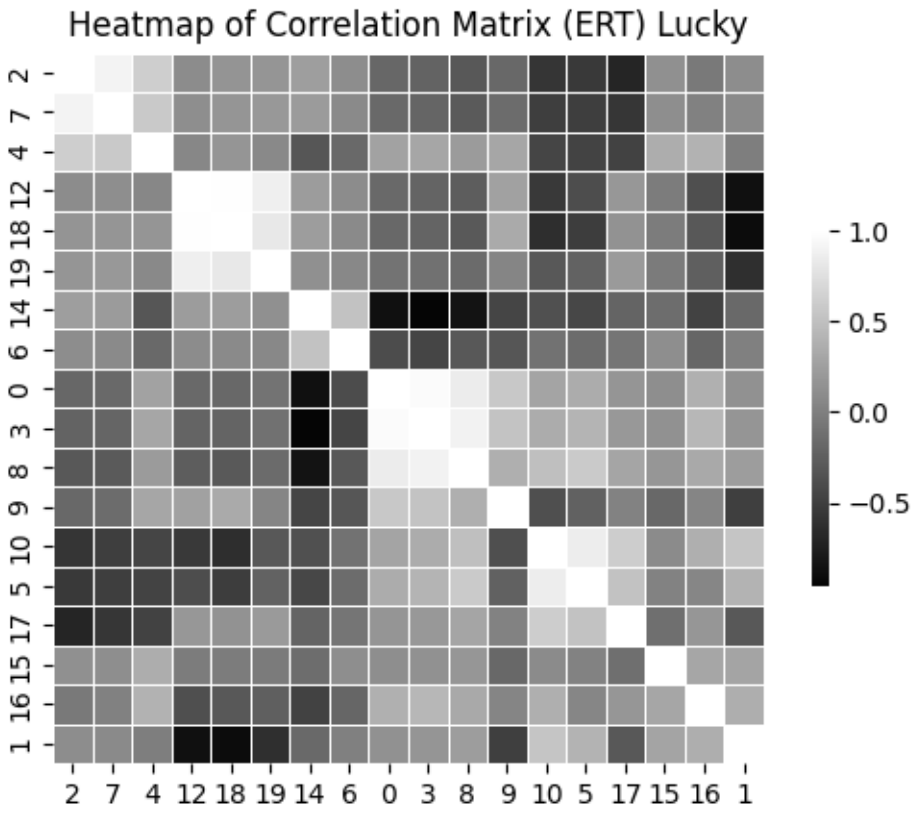

Figure 13: Correlation matrix of hidden neuron activations in ERT in a trial that showcased luck

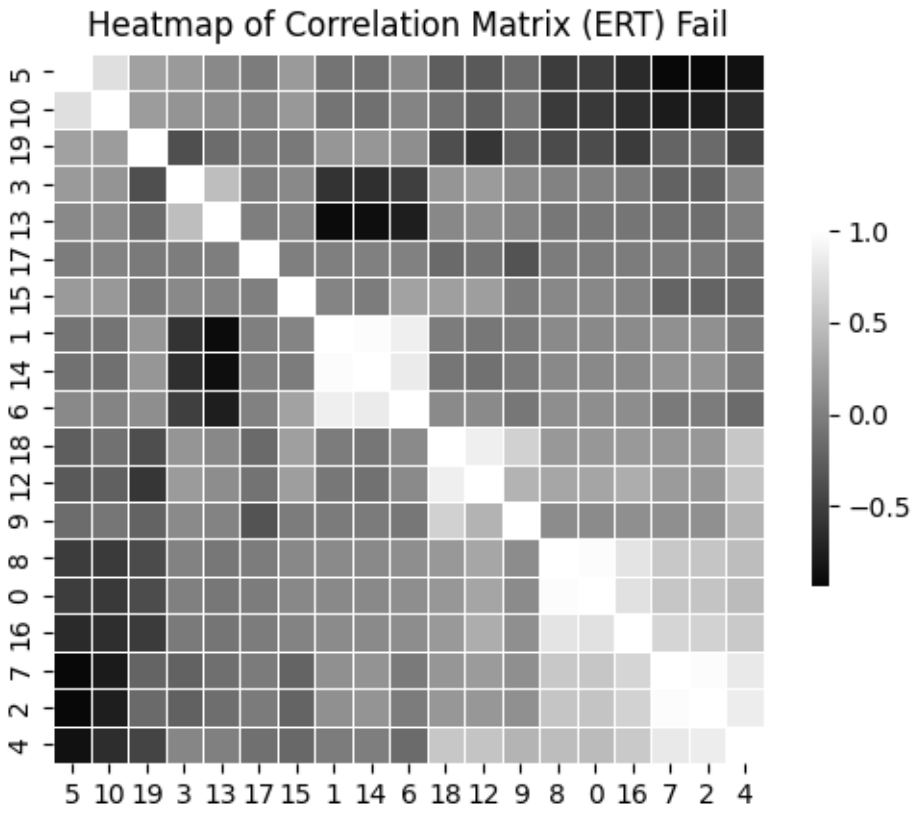

Figure 14: Correlation matrix of hidden neuron activations in ERT in a trial that showcased failure

The three figures shown below are the activation graphs of the hidden neurons for each case provided above. The information provided in these graphs are the same as the correlation plots; however, they show the numerical values and when the neurons spiked. This can provide valuable insight such as what neurons were active when the agent picked up the tool.

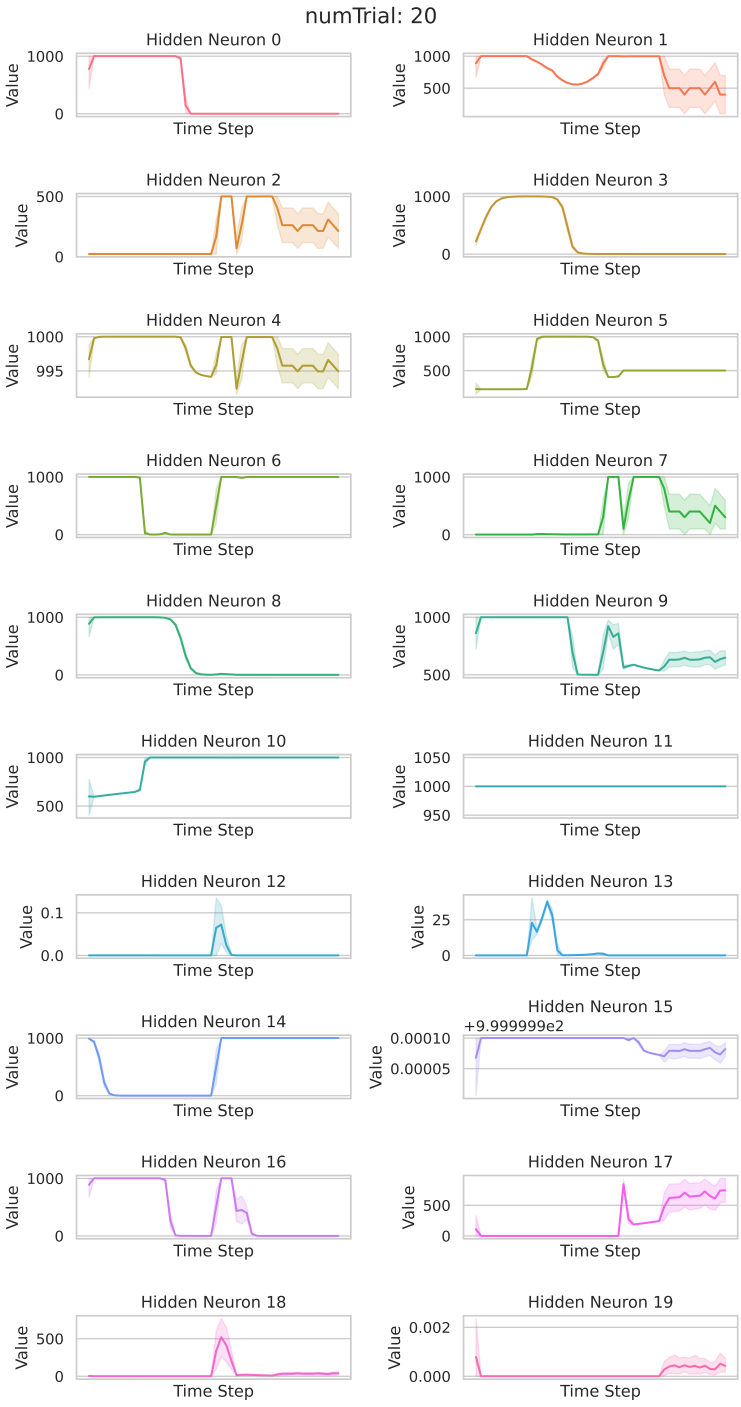

Figure 15: Activation graphs of hidden neurons in a predictive case.

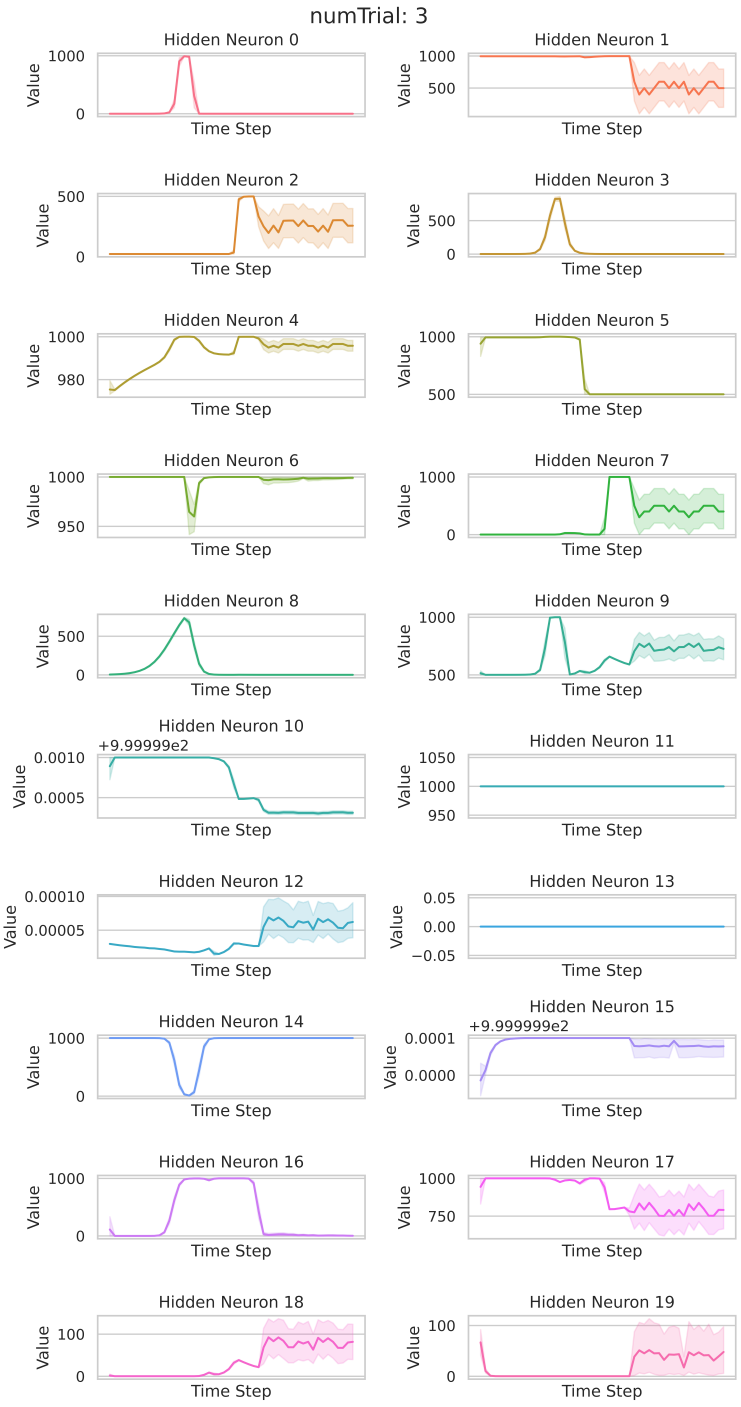

Figure 16: Activation graphs of hidden neurons in a lucky case.

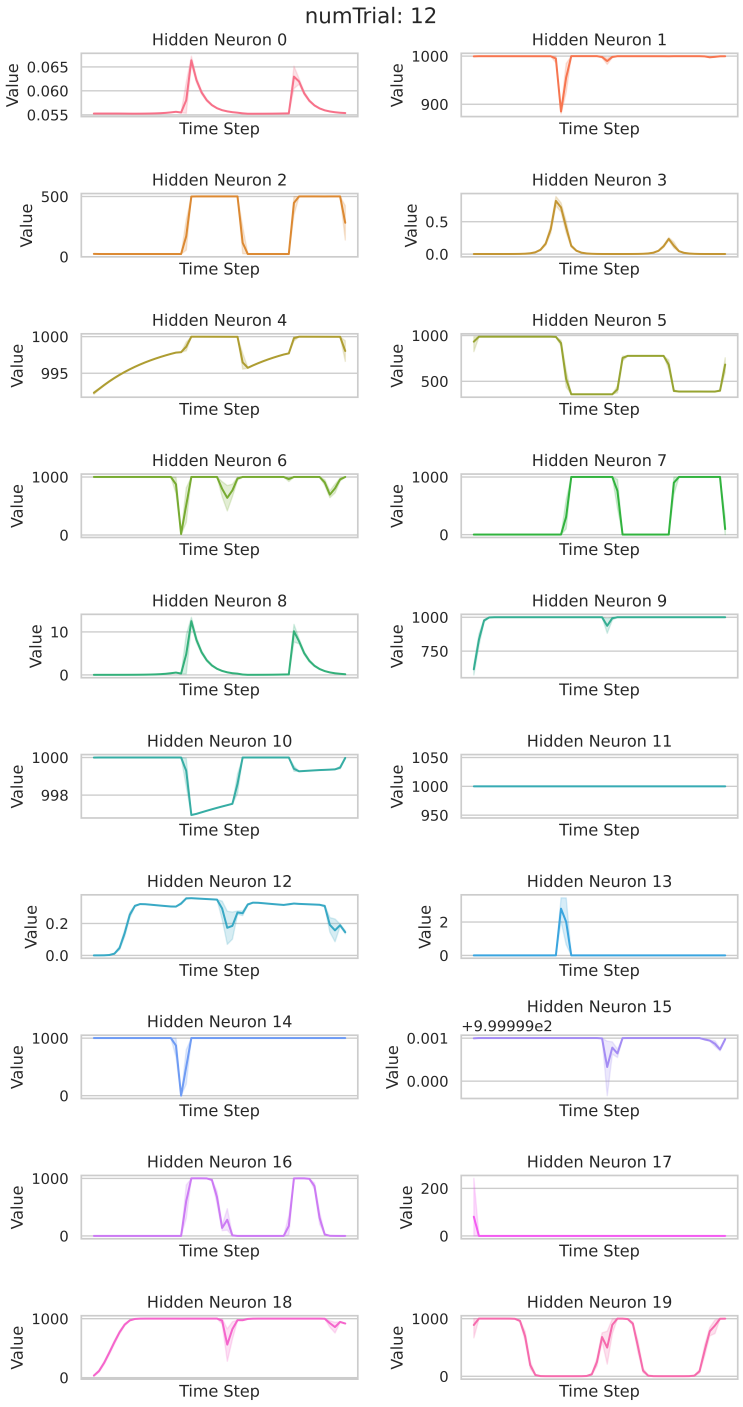

Figure 17: Activation graphs of hidden neurons in a failure case.

The three figures here show the time series correlation of the ER fitness function organized

into the same three categories: predictive, lucky, and fail. As we can see here all three cases for the ER function share a common cluster (3,8,5,4) so we can assume that as above is for the basic functions of the limb. Unlike the ERT fitness function there is not a subcluster we can isolate in the predictive case as the only other cluster in the ER predictive case (0,9) is also in the failure case. However, as we can see the complexity of the agent decreases with the removal of the tool factor.

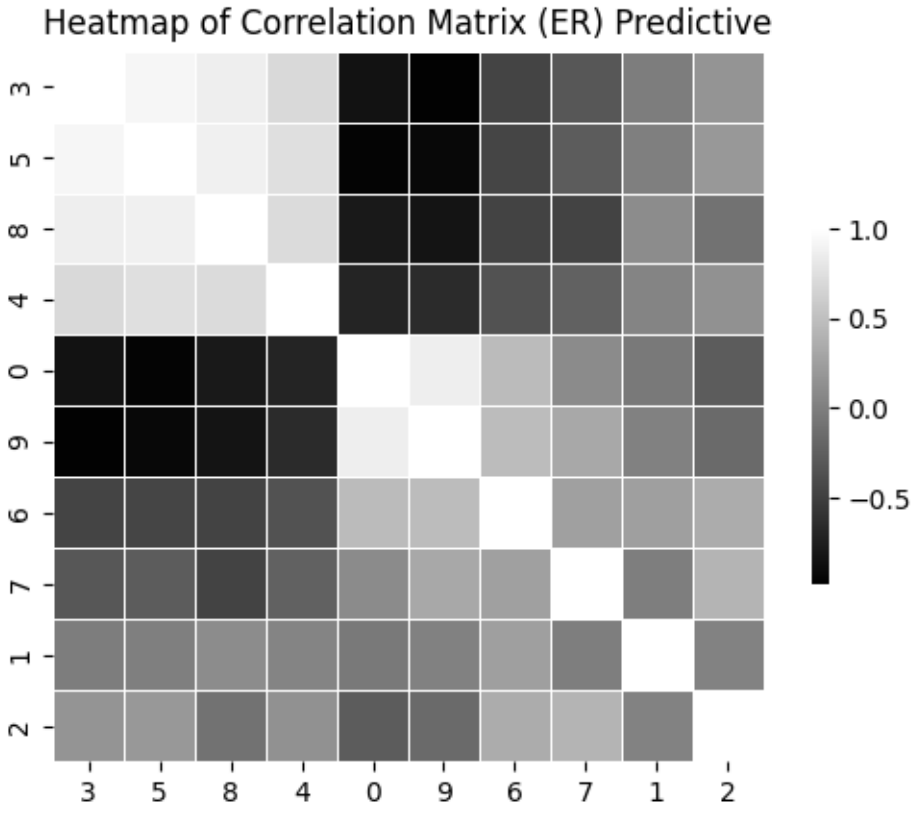

Figure 18: Correlation matrix of hidden neuron activations in ER in a trial that showcased prediction.

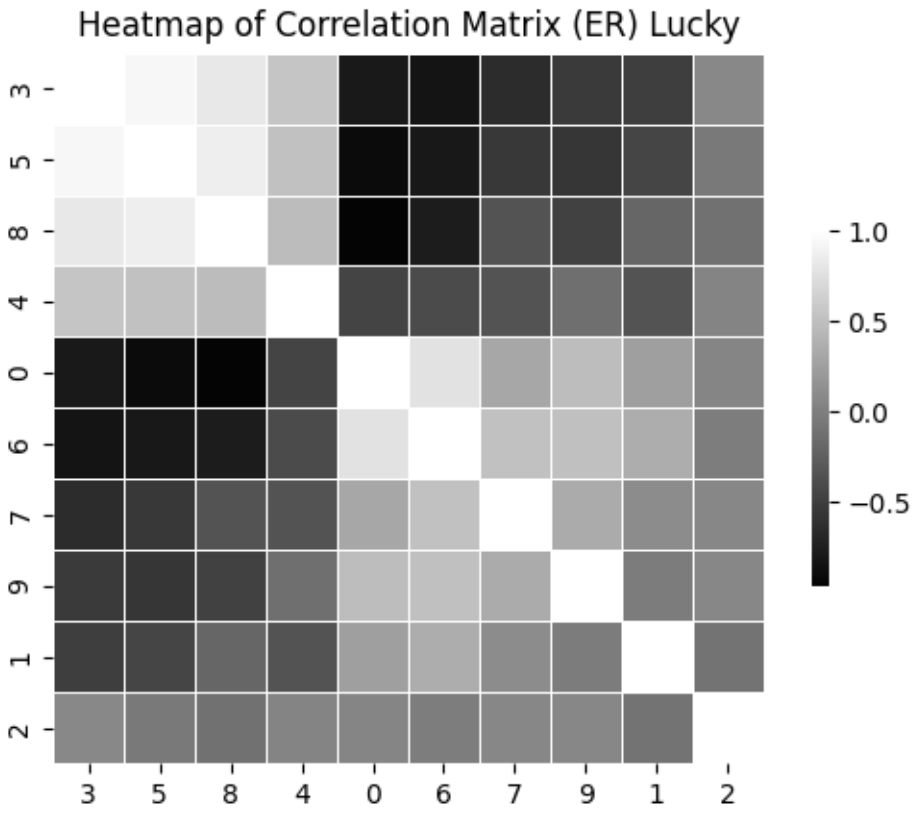

Figure 19: Correlation matrix of hidden neuron activations in ER in a trial that showcased luck

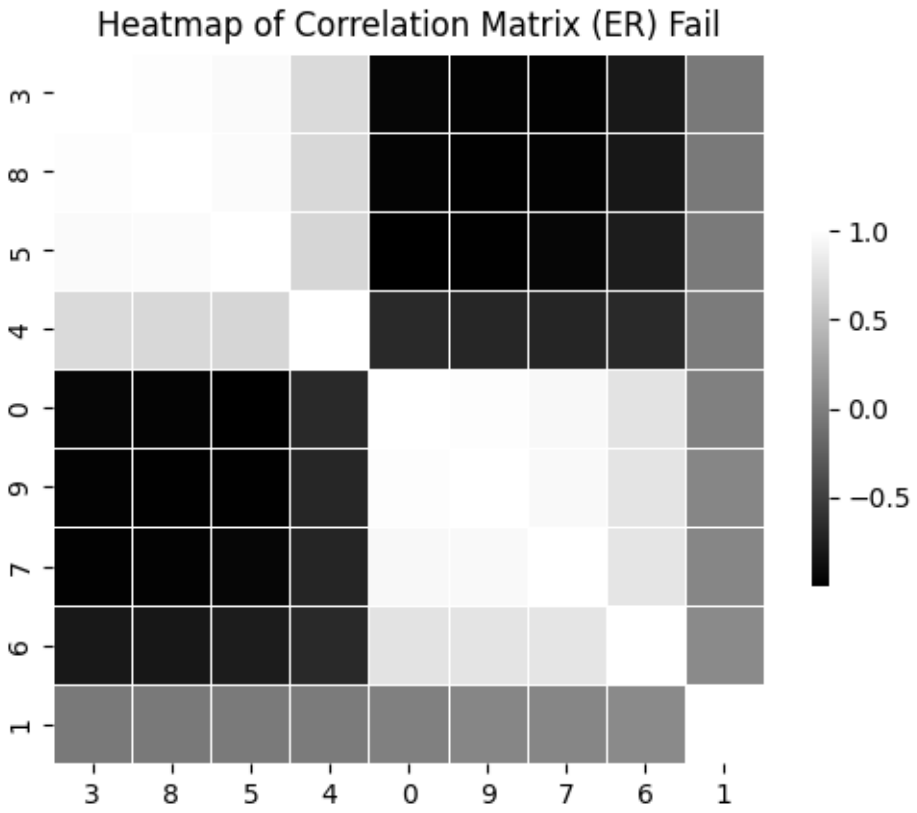

Figure 20: Correlation matrix of hidden neuron activations in ER in a trial that showcased failure

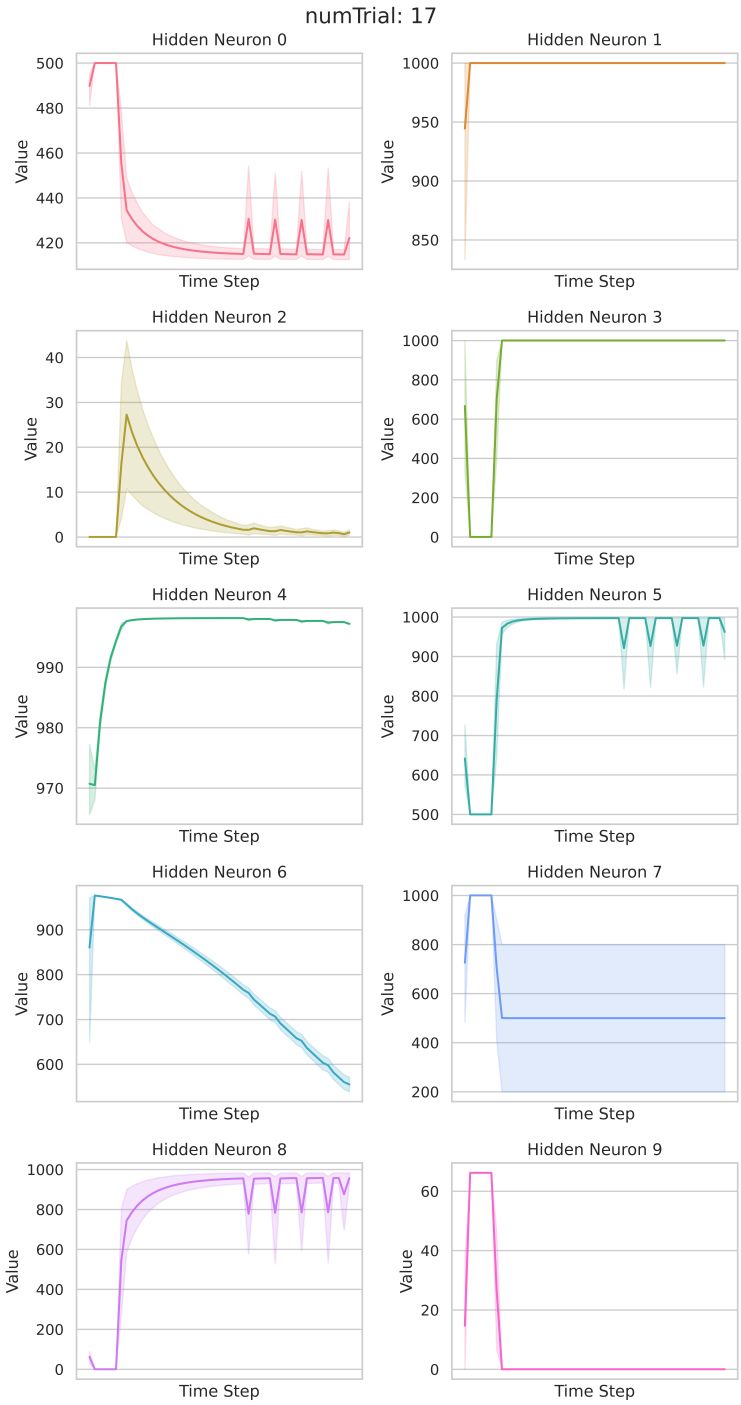

Figure 21: Activation graphs of hidden neurons in a predictive case.

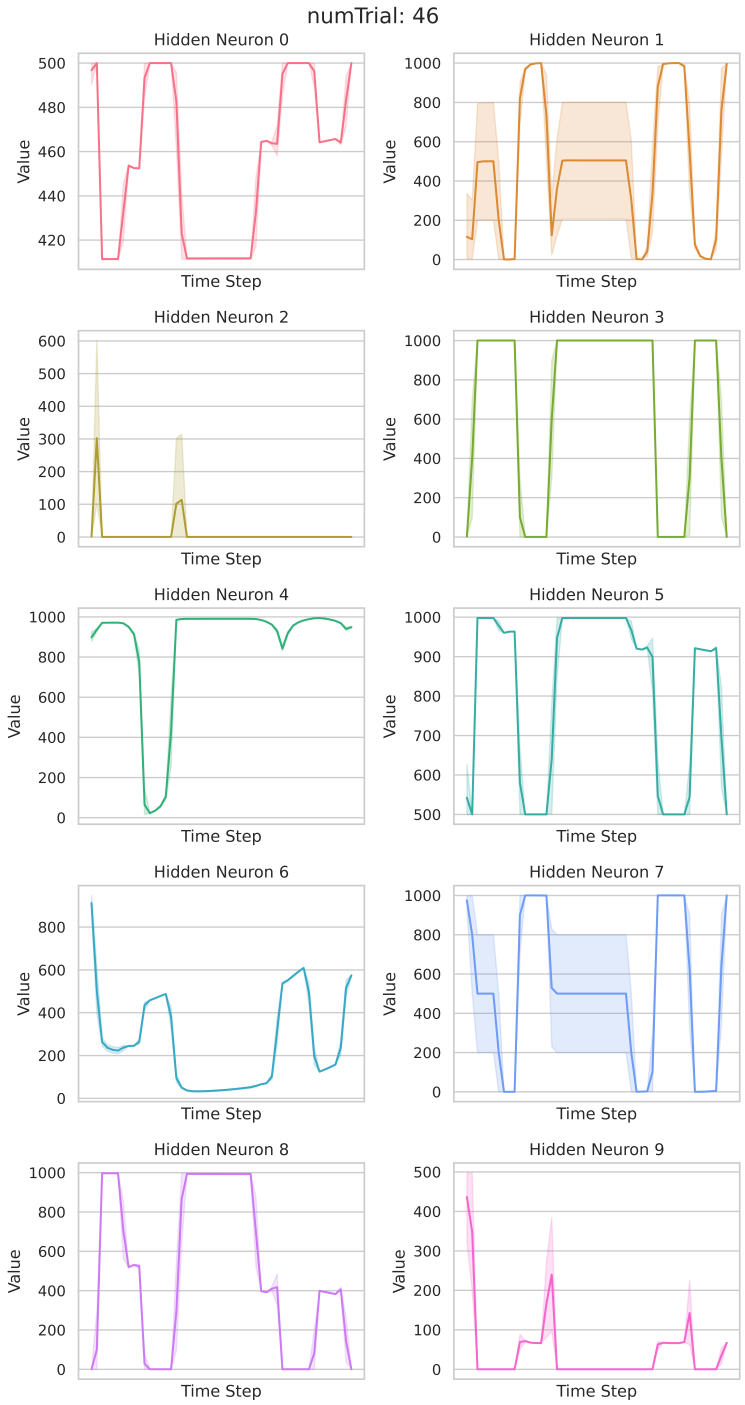

Figure 22: Activation graphs of hidden neurons in a lucky case.

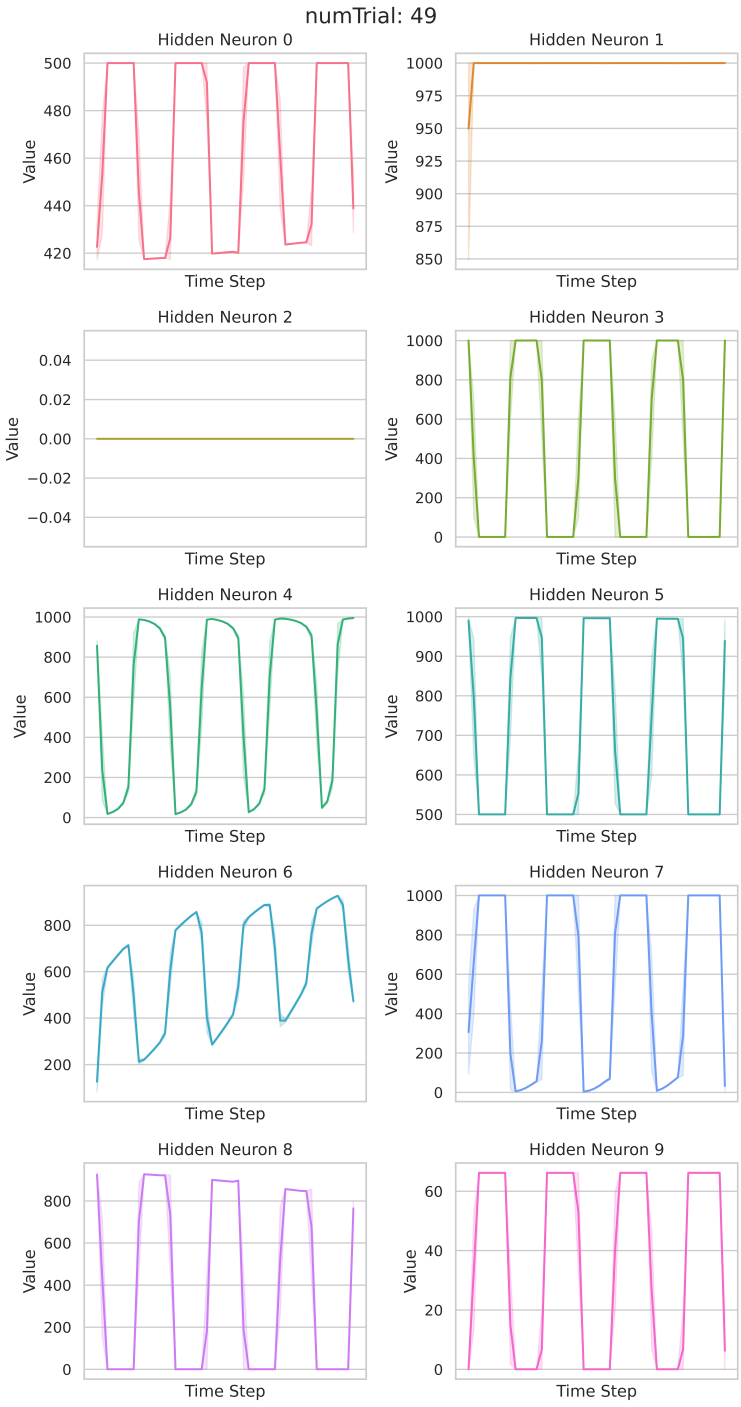

Figure 23: Activation graphs of hidden neurons in a failure case.

The three figures below show the time series correlation between the hidden neurons for

the RT function. As we can see here the agent is even more simplistic than the ER function. Unfortunately we have a paucity of information regarding the correlation between the three cases. However, we can still conclude that there is a common cluster between all three cases (1,3).

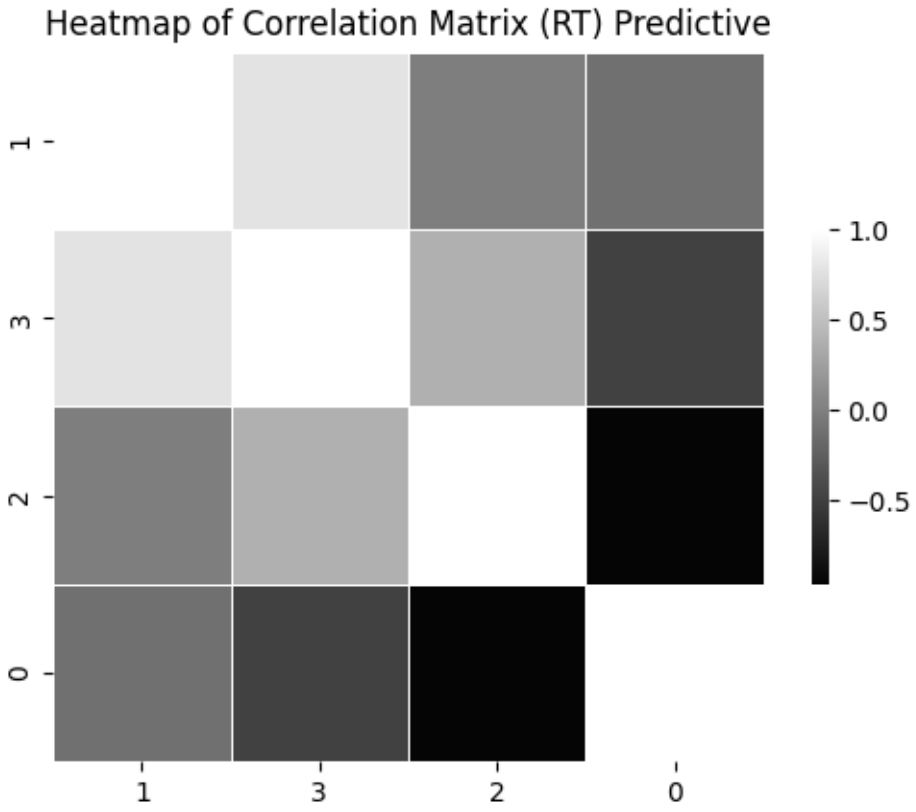

Figure 24: Correlation matrix of hidden neuron activations in RT in a trial that showcased prediction.

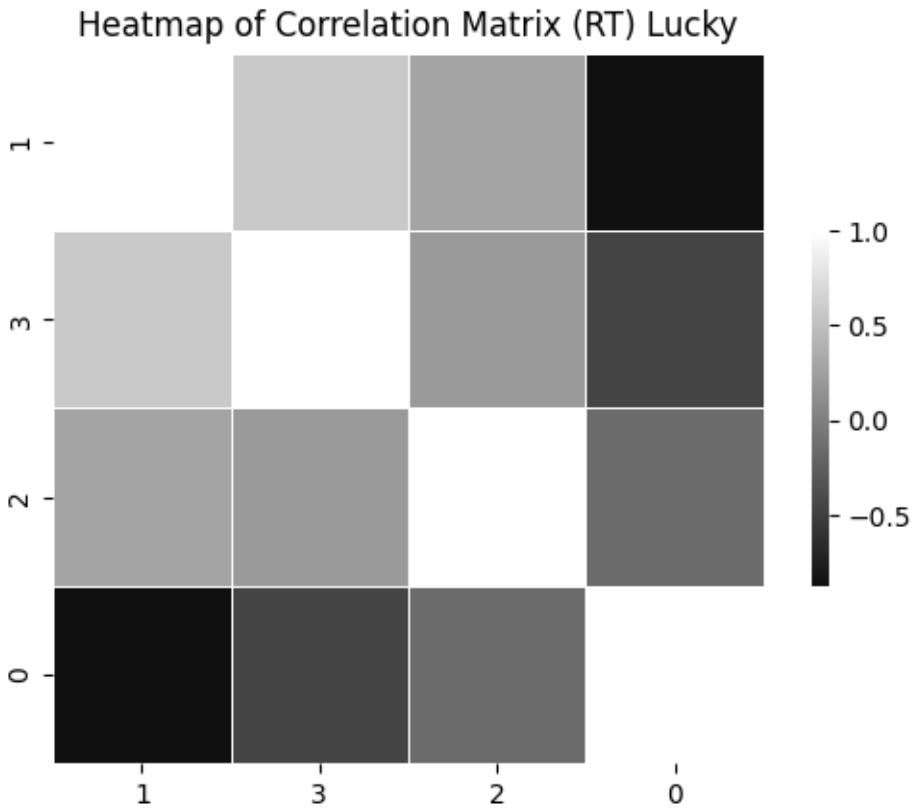

Figure 25: Correlation matrix of hidden neuron activations in RT in a trial that showcased luck

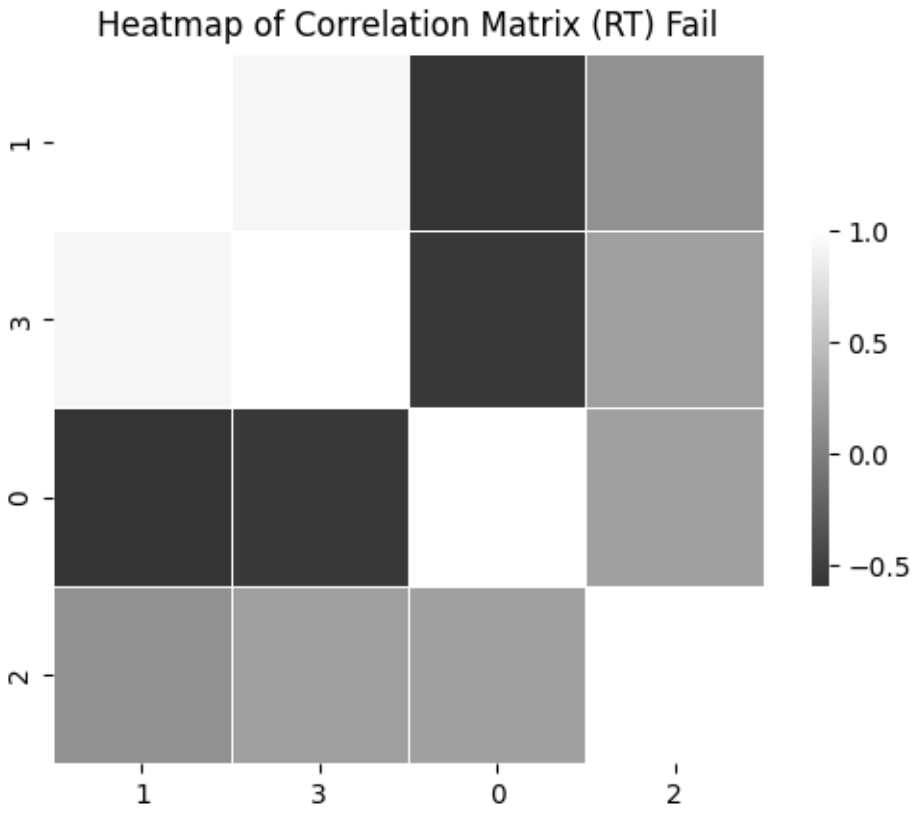

Figure 26: Correlation matrix of hidden neuron activations in RT in a trial that showcased failure

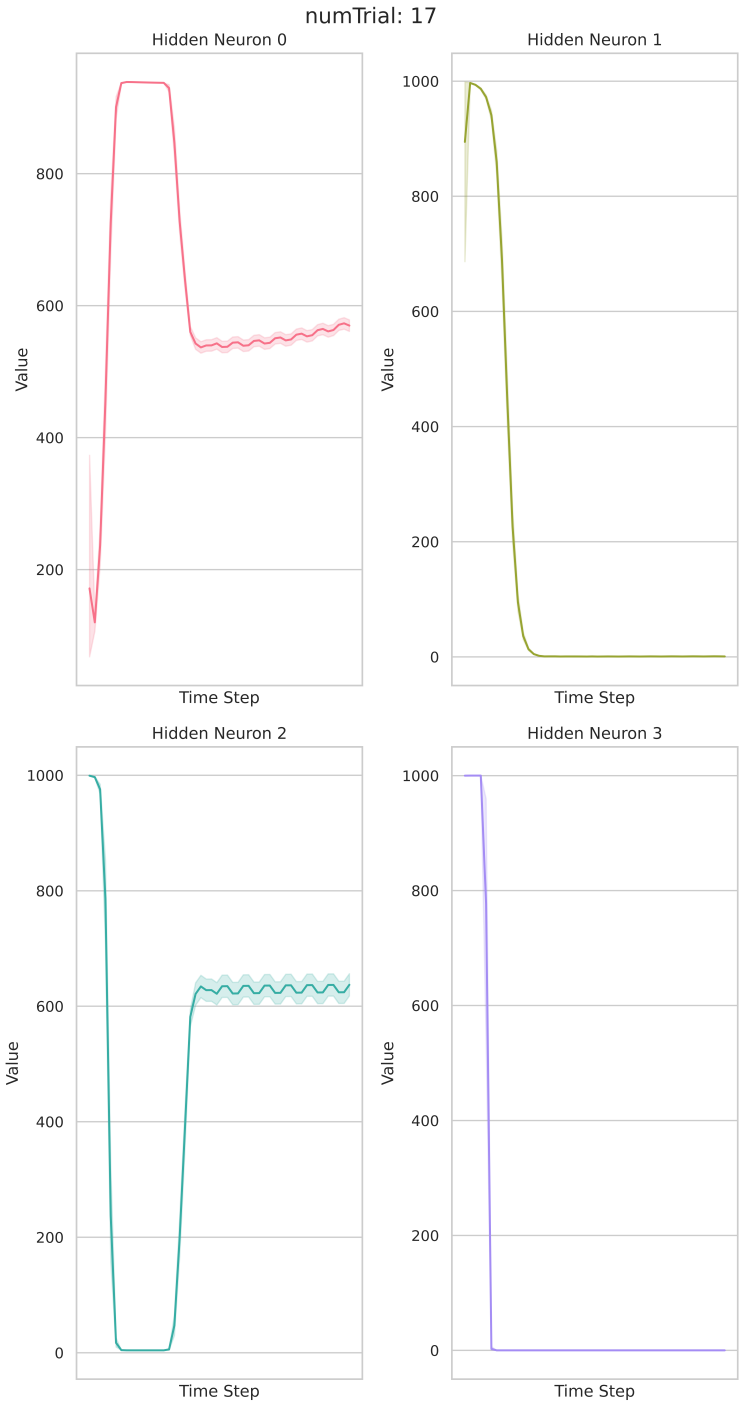

Figure 27: Activation graphs of hidden neurons in a predictive case.

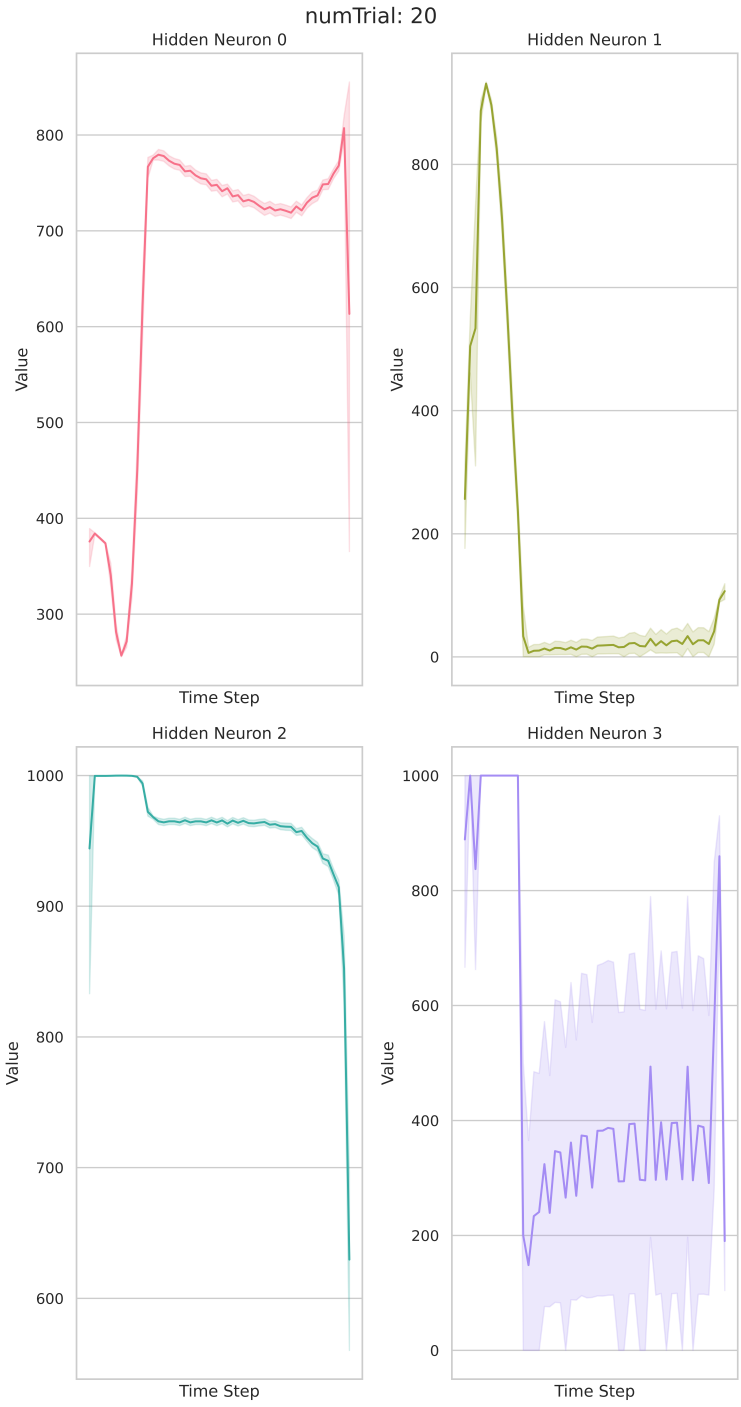

Figure 28: Activation graphs of hidden neurons in a lucky case.

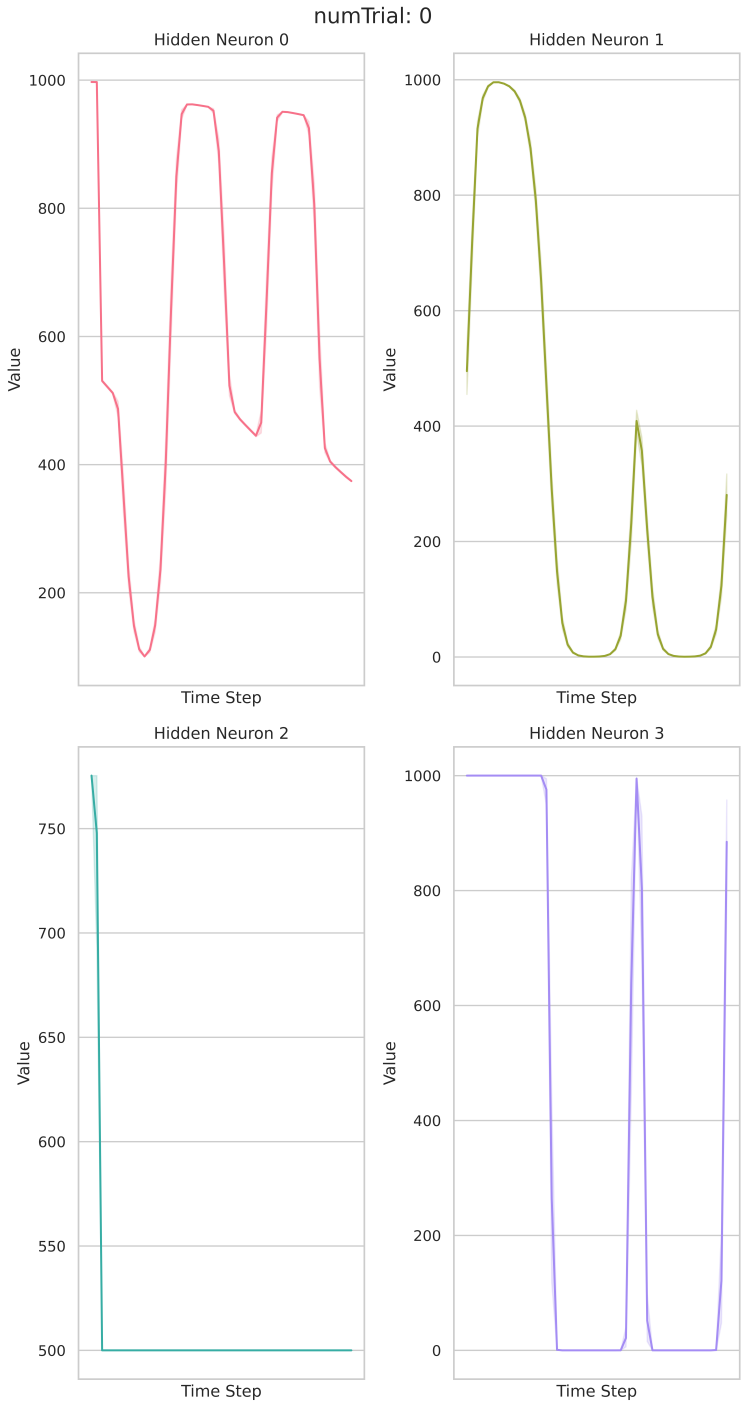

Figure 29: Activation graphs of hidden neurons in a failure case.

As we can see with ERT, ER, and RT we can observe that the increasing number of fac-

tors leads to an increase in the complexity of the agents. We can also hypothesize that with the increase of complexity agents there are more specialized hidden neurons as shown with the time series correlation matrices. With the correlations especially in Figures 18,19, and 20 we can see there is a higher correlation between smaller amounts of neurons. We can reason using the specialized hidden neurons to allow for the agent to perform better. On the other hand, we can also hypothesize that the increase in the number of neurons allows for the agent to have more complex memory which has been shown to be beneficial to prediction in animals. While we can look at the correlation between the hidden neurons we should also observe the relationship between the number of loops in each best performing agent and the different metrics for success.

3.6 Comparison of Number of Loops

In this subsection we discuss the correlation between the number of loops and the performance of the agents.

As we proposed in the introduction, our hypothesis suggests that increasing the number of loops in the agent positively influences its performance using two metrics: success rate and fitness. Here we define success rate as the average number of 5_ hits that the agent achieved. Figure 30 shows a typical evolved neural network topology. We can see several loops in the network. Figure 31 illustrates the fitness calculation using the formula explained in Chapter 2 for the ERT function. It is evident from the results that as the chromosomes evolve and the number of loops increases, there is a noticeable improvement in the fitness of each agent. There is also a corresponding enhancement in the success rate of the agent as shown in Figure 32.

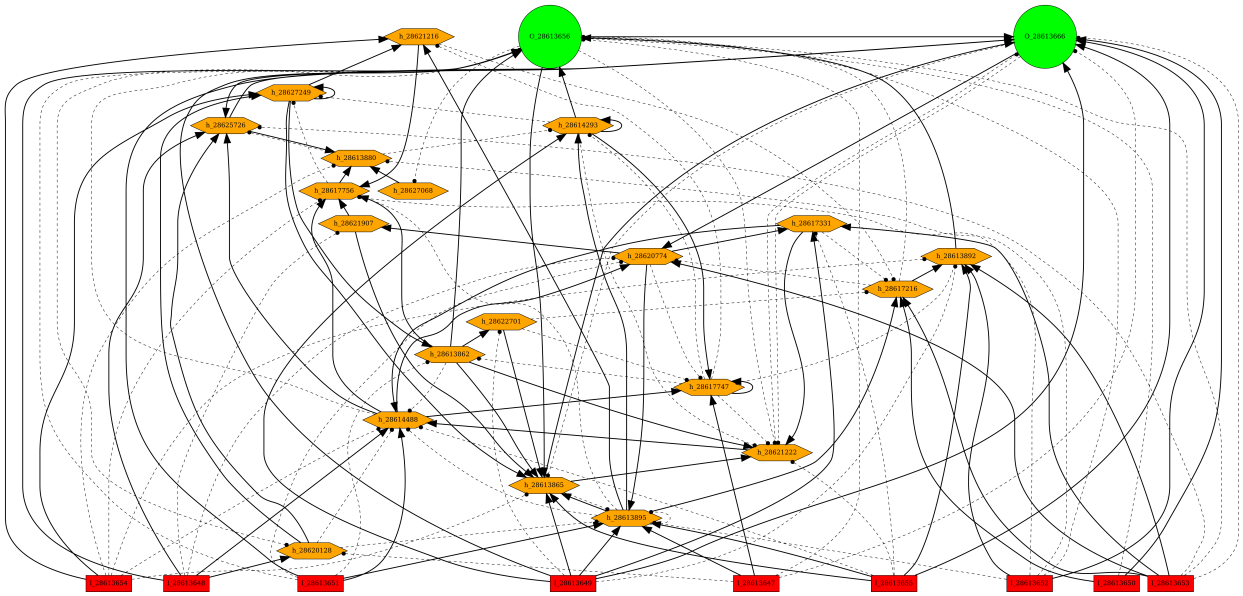

Figure 30: Topology of an agent trained using the ERT function

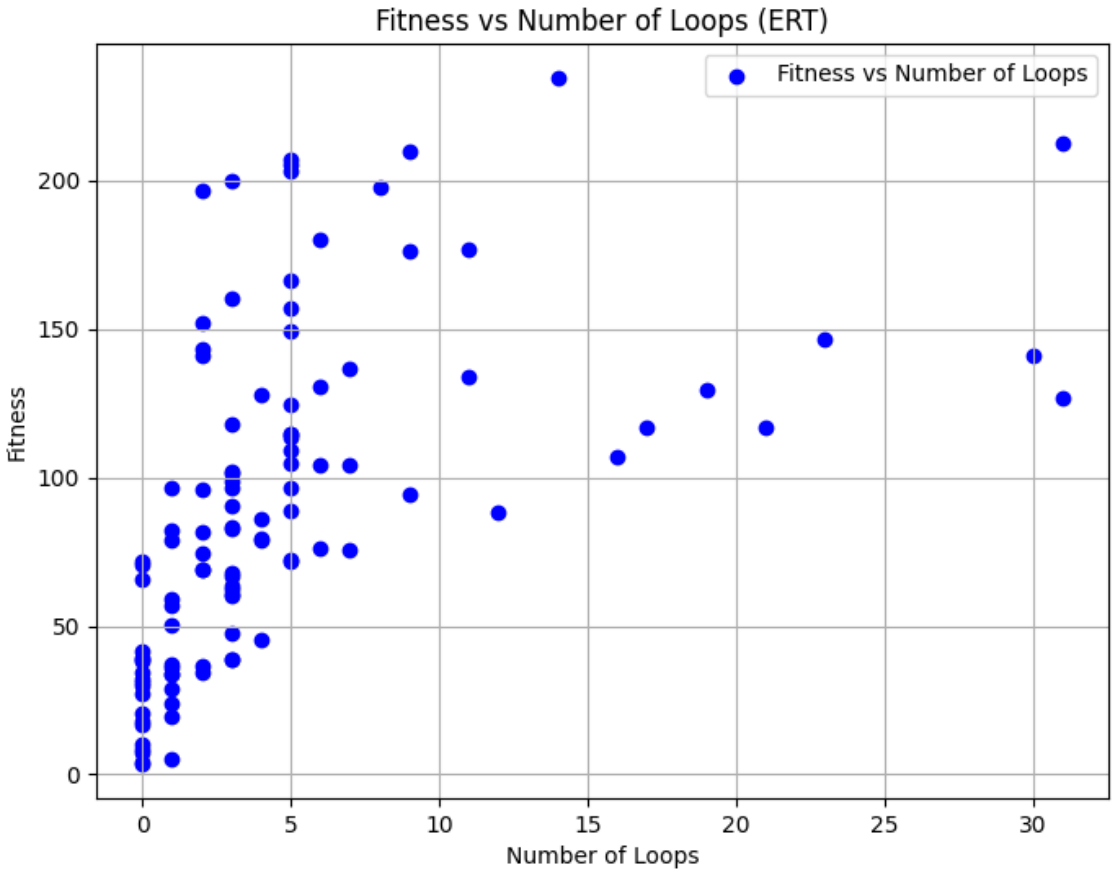

Figure 31: Scatterplot of average fitness vs number of loops in the agents trained using the ERT function

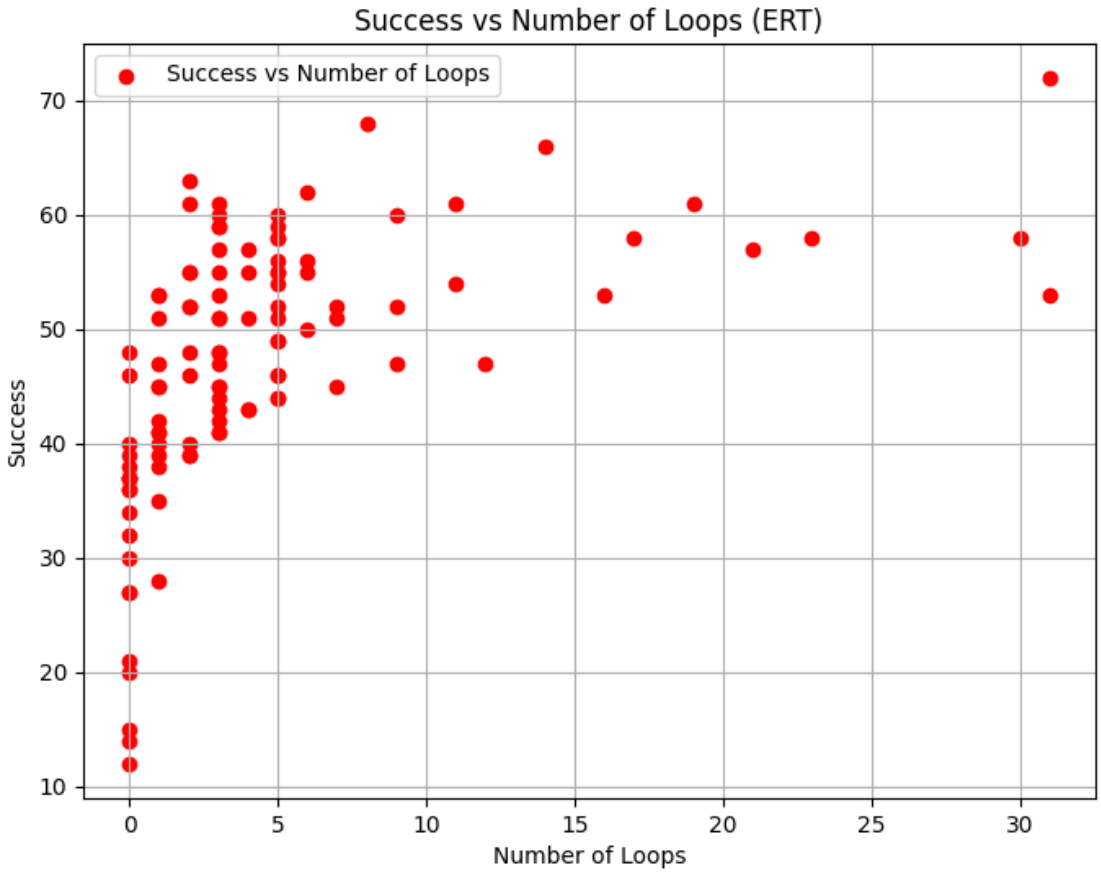

Figure 32: Scatterplot of average success rates vs number of loops in the agents trained using the ERT function

In Figure 34, a pronounced correlation persists between the escalating number of recurrent loops and the agent's fitness. Furthermore, Figure 35 reveals a robust association between the quantity of recurrent loops and the success rate achieved by the agent. Figure 33 shows a typical evolved neural network topology. We can see several loops in the network.

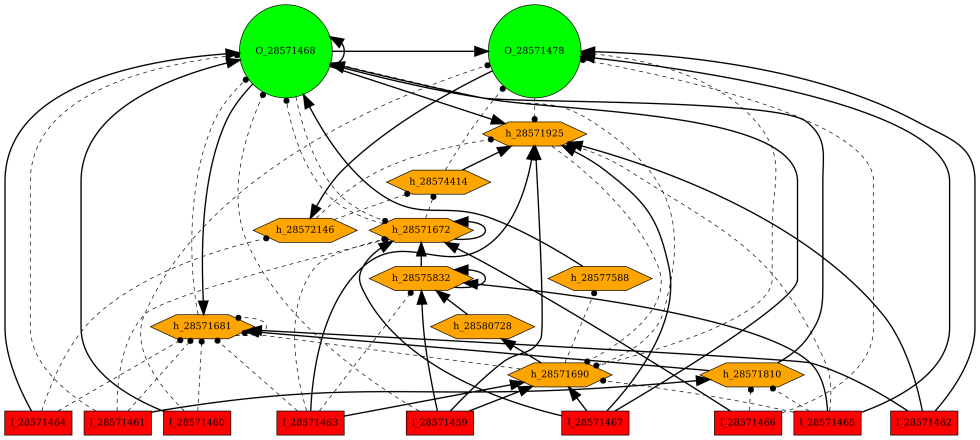

Figure 33: Topology of an agent trained using the ER function

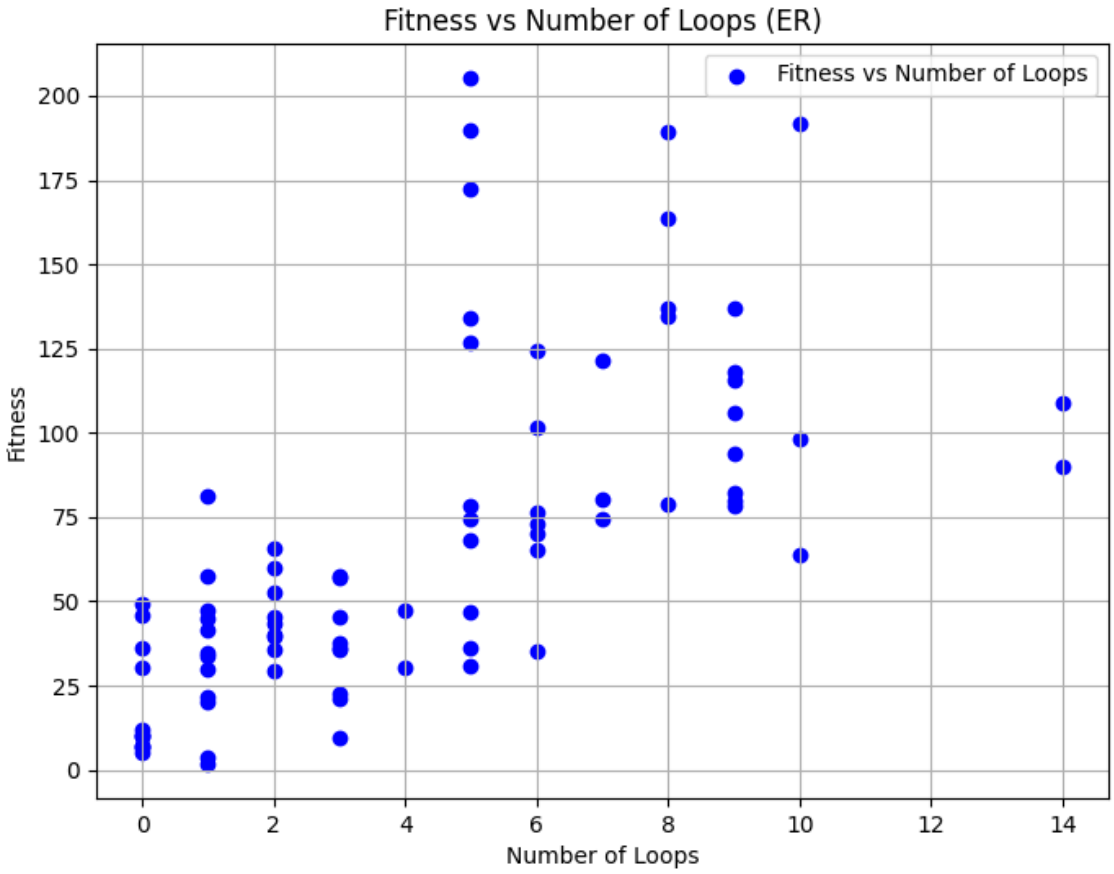

Figure 34: Scatterplot of average fitness vs number of loops in the agents trained using the ER function

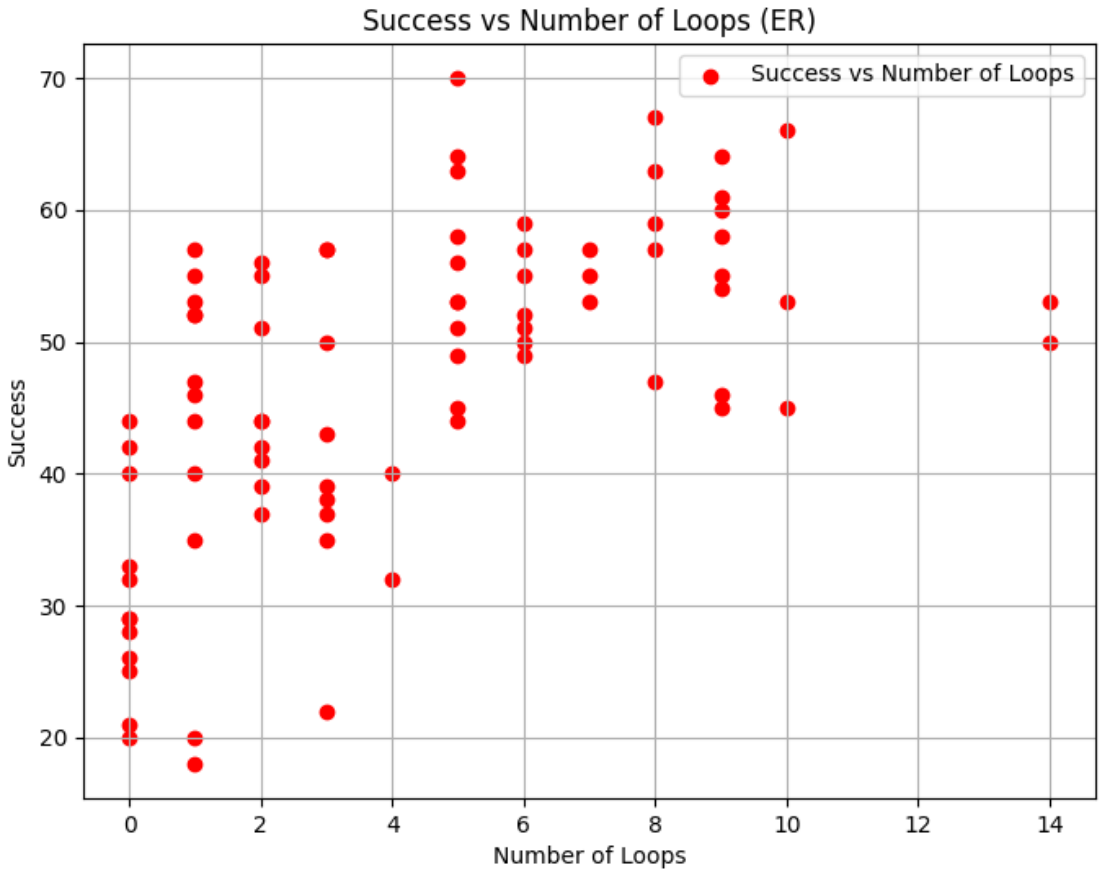

Figure 35: Scatterplot of average success rates vs number of loops in the agents trained using the ER function

Upon observation, our dataset for the RT functions is constrained by the simplicity of the agents, resulting in a limited amount of available data. Nevertheless, discernible correlations emerge, notably between the number of loops and the fitness rate of the agents, as depicted in Figure 37, and similarly, between the number of loops and the success rate of the agents, as illustrated in Figure 38. Given the prevalent use of recurrent loops for memory retention within neural networks, it is reasonable to infer a correlation between memory capacity and predictive capabilities in the agents due to the increase in performance given the number of loops. Figure 36 shows a typical evolved neural network topology. We can see several loops in the network.

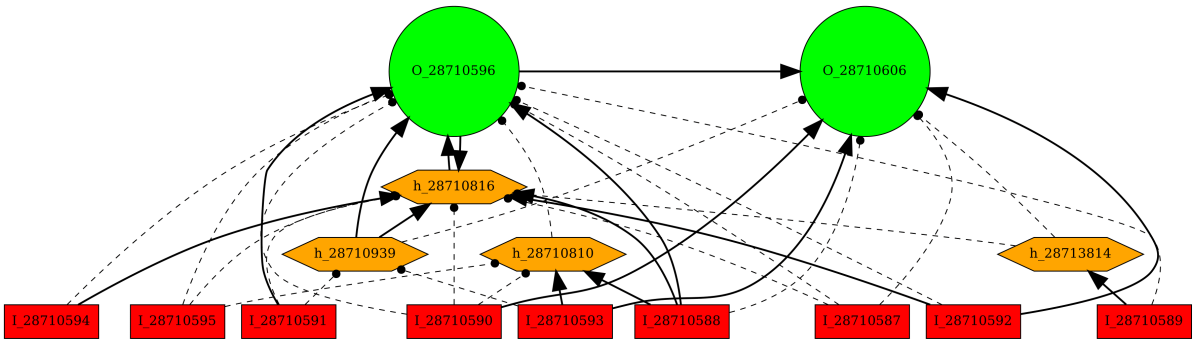

Figure 36: Topology of an agent trained using the RT function

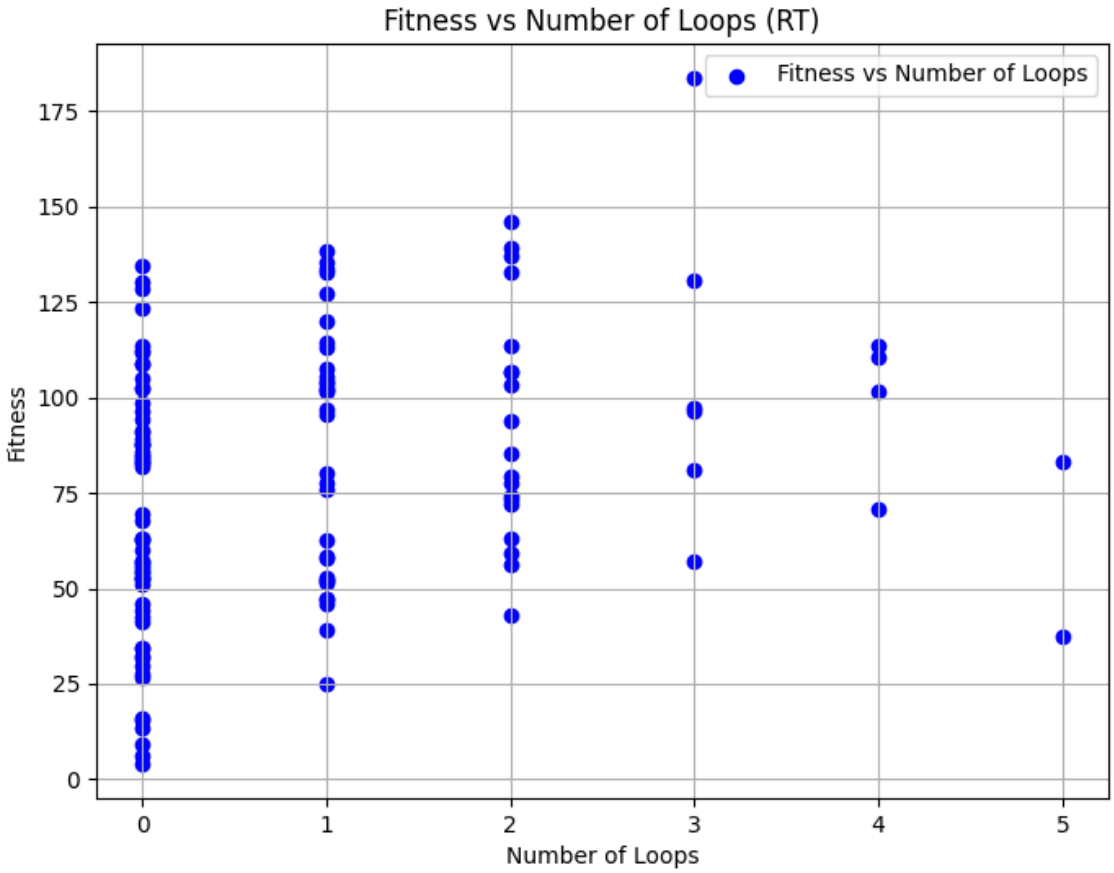

Figure 37: Scatterplot of average fitness rates vs number of loops in the agents trained using the RT function

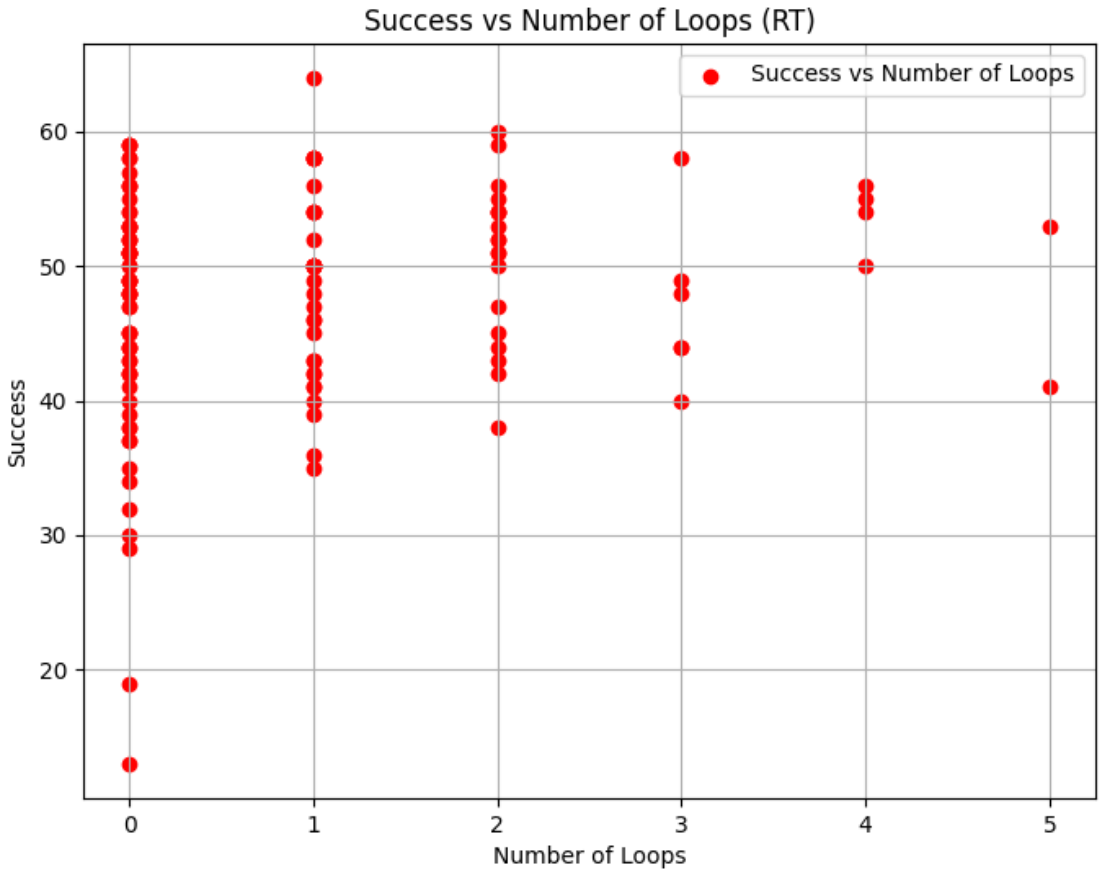

Figure 38: Scatterplot of average success rates vs number of loops in the agents trained using the RT function

The findings unmistakably reveal a progressive enhancement in both the fitness of each agent and the corresponding success rate as the chromosomes evolve and the number of loops increases, as illustrated in Figure 41. However, it is noteworthy that the data points of the function exhibit increased variability with the rising number of loops. This variability stems from the nature of the R fitness function, as discussed in earlier Chapter 3, which tends to explore more possibilities and occasionally achieves success through lucky rather than prediction. Despite this apparent variability, the data consistently underscores the positive correlation between a higher number of loops and the agents' improved predictive capabilities. Figure 39 shows a typical evolved neural

network topology. We can see several loops in the network.

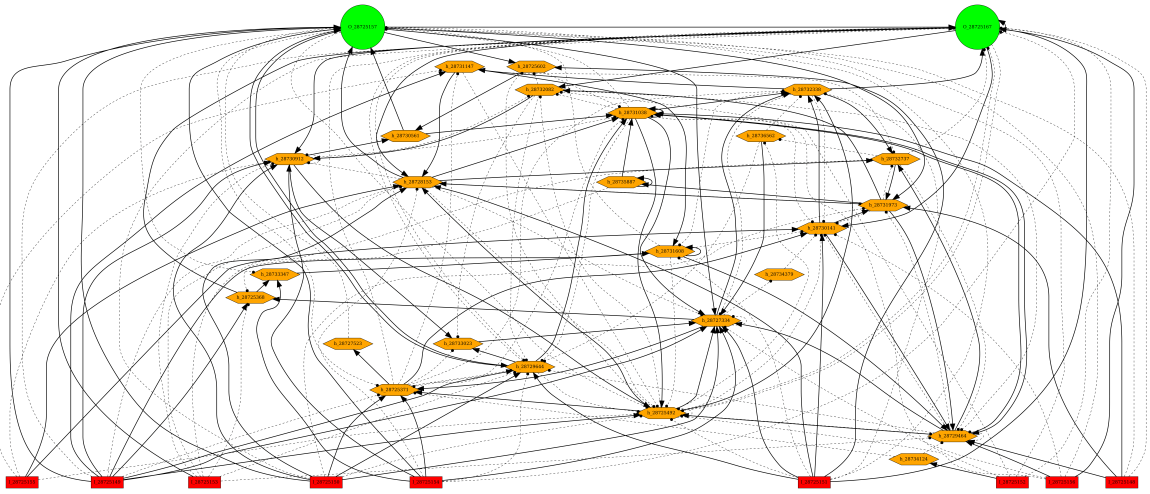

Figure 39: Topology of an agent trained using the R function

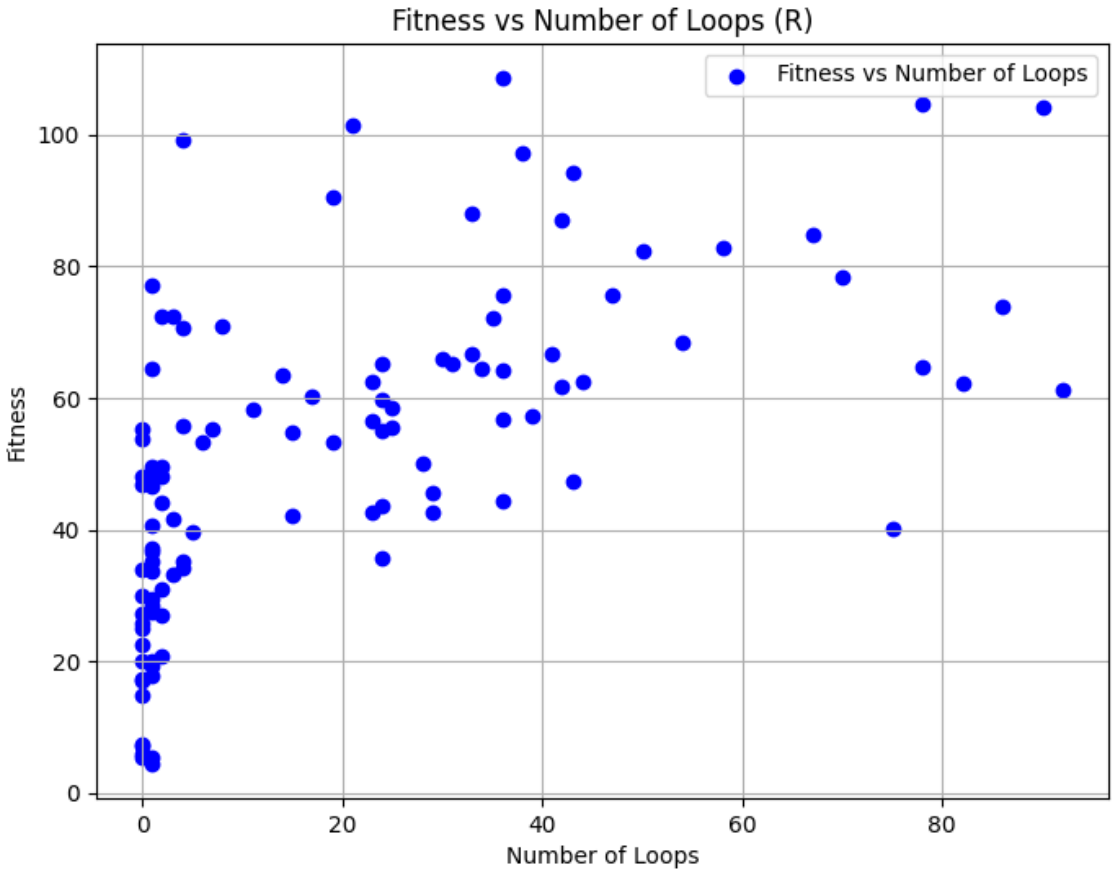

Figure 40: Scatterplot of average fitness vs number of loops in the agents trained using the R function

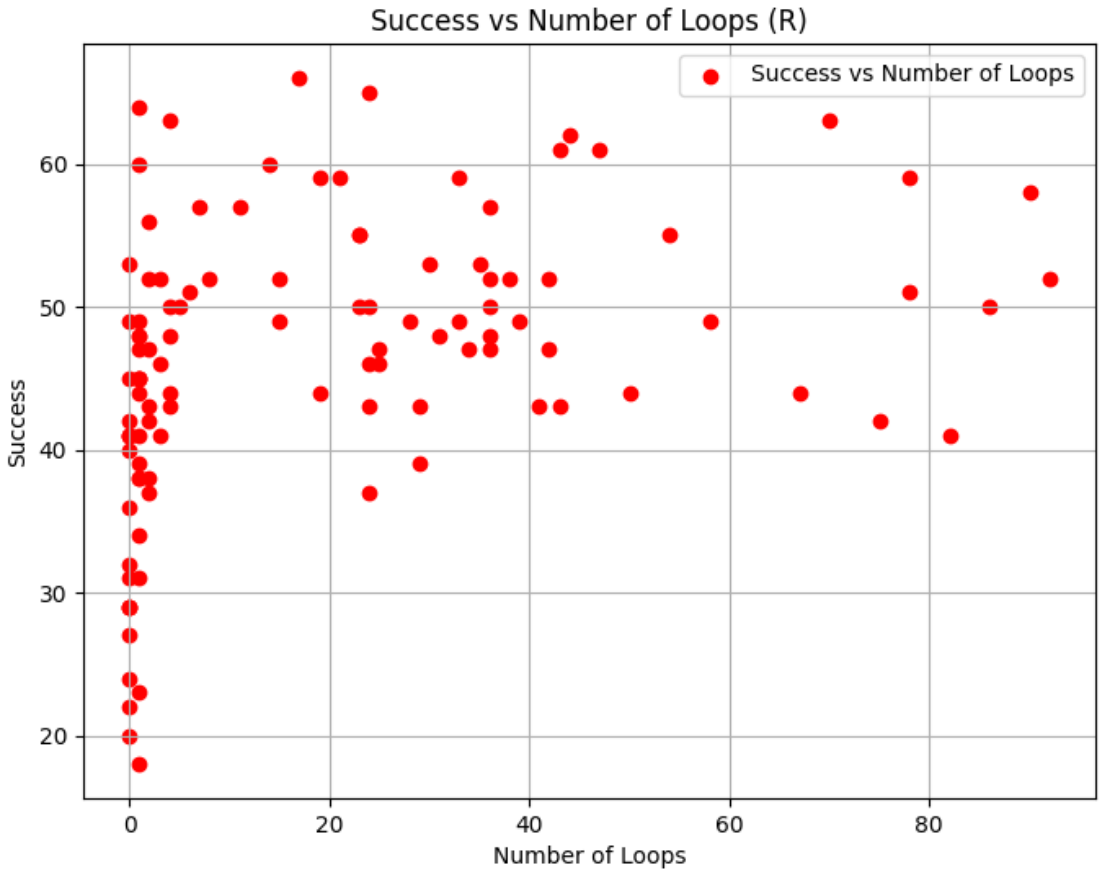

Figure 41: Scatterplot of average success rates vs number of loops in the agents trained using the R function

4. DISCUSSION

In this section, we discuss some unique and interesting preliminary findings of our experiments that can lead to future research.

4.1 Individual Hidden Neuron's Role

In this subsection we discuss possible future works.

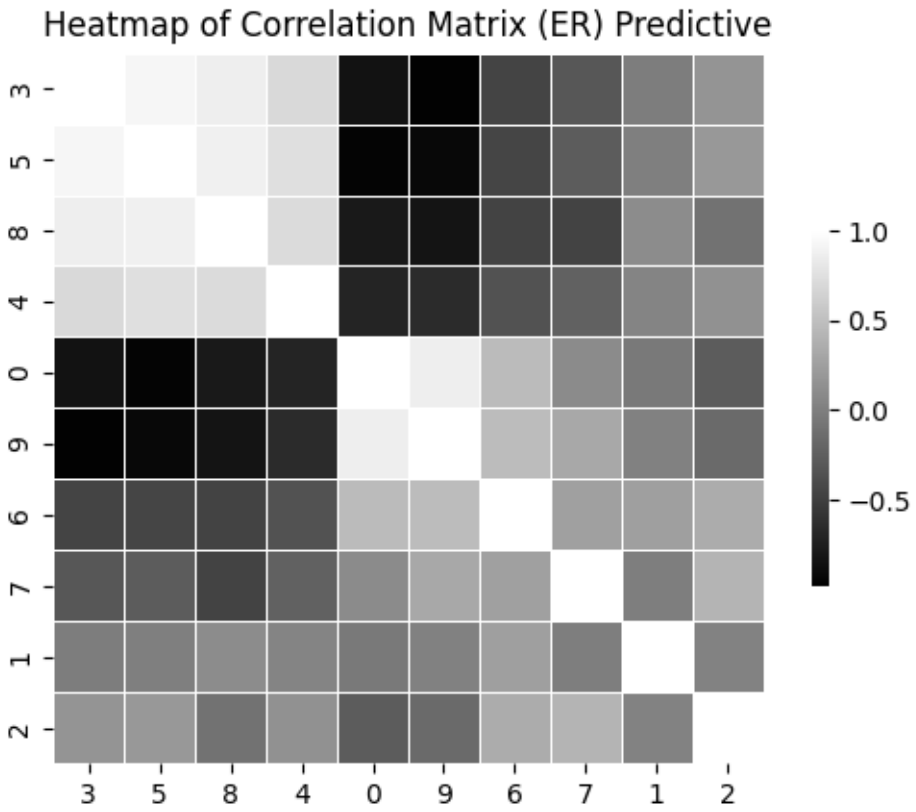

Figure 42: Correlation matrix of hidden neurons activations in ERT in a trial that showcased prediction.

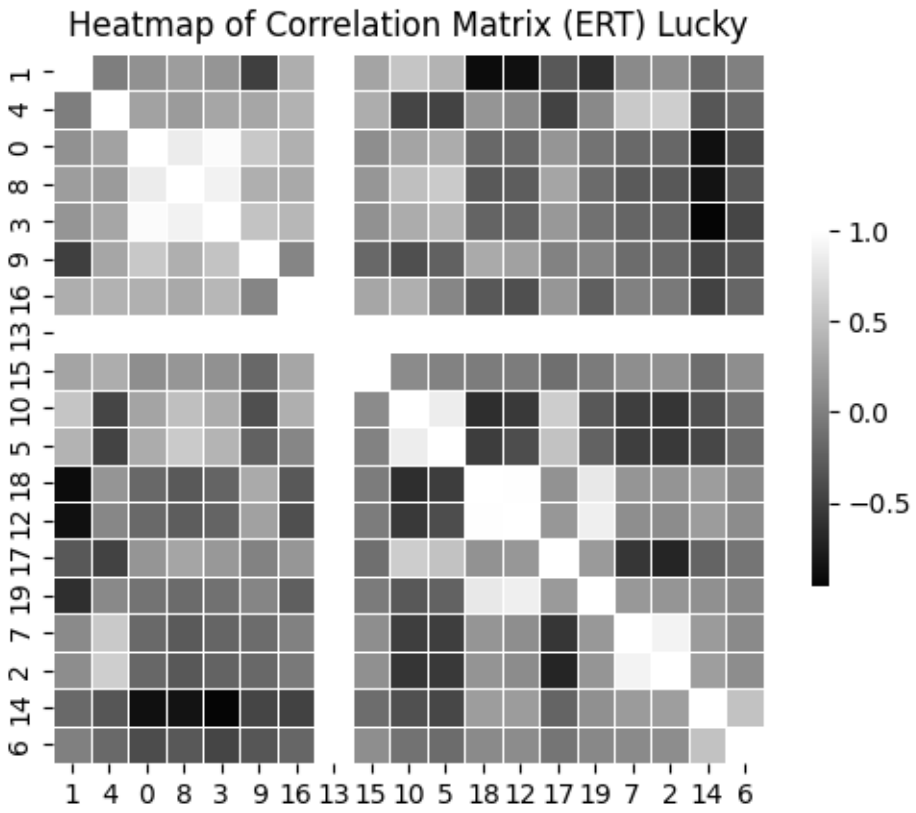

Figure 43: Correlation matrix of hidden neurons activations in ERT in a trial that showcased luck re-ordered in respect to prediction trial.

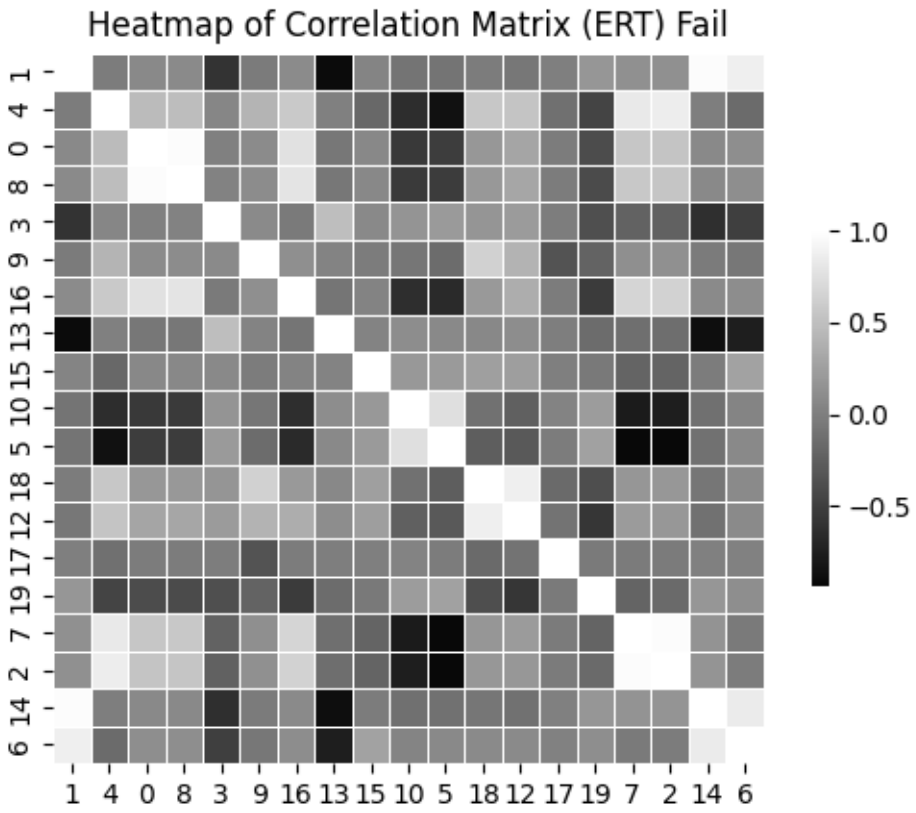

Figure 44: Correlation matrix of hidden neurons activations in ERT in a trial that showcased failure re-ordered in respect to prediction trial.

In section 3.5, we discuss the possible correlation between the main unshared cluster (17, 19) and the prediction capabilities of the ERT agent. Future works may involve removing all connections to both hidden neurons and possibly even other hidden neurons and observing how the agent navigates through the environment. Theoretically, the removal of all connections to both neurons would lead to the model following the delayed target.

To draw comparisons between different cluster groups, however, it may be easier to do so if we keep the order of the mapped neurons. Figures 42,43, and 44 re-ordered the neuron correlation matrix based on the ordering of the predictive case's cluster. As a result, we are able to better see the differences in overall activation levels between different trials. Such information paired

with specific activation levels and the agent's movement could help pinpoint topological features in prediction and intent.

4.2 Predictive Capabilities of Hidden Neuron Clusters

In this subsection we discuss the performance of the ERT agent without it's predictive cluster.

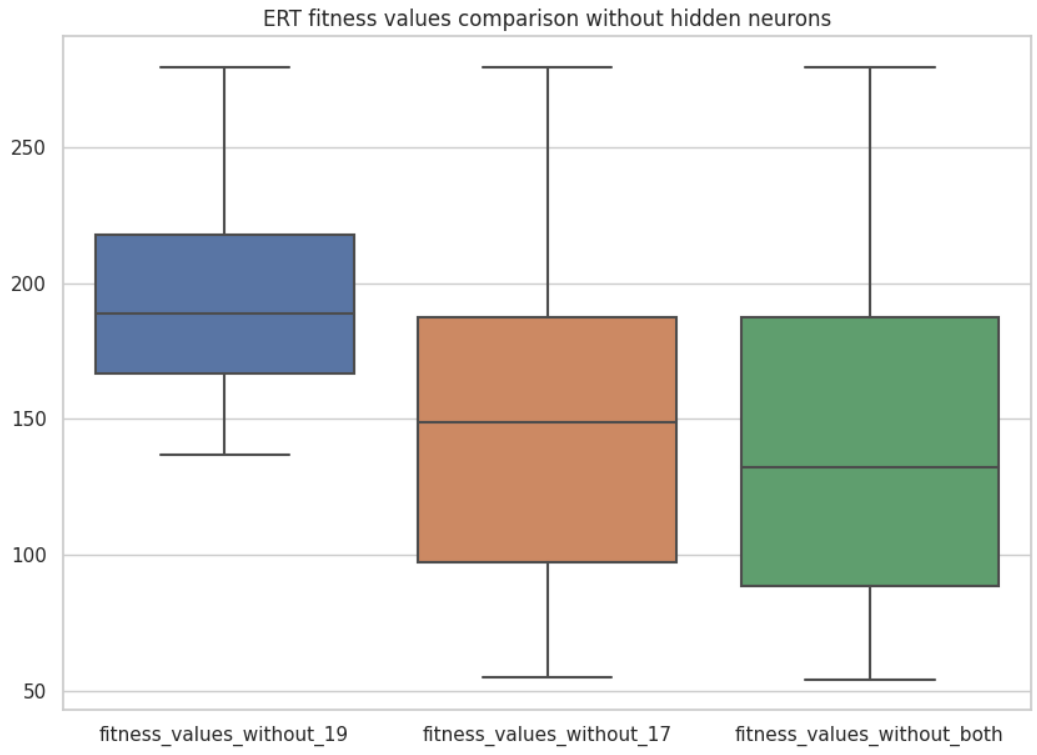

Figure 45: Comparison of fitness values of the ERT agent without part or all of the hidden neurons in the predictive cluster n = 15.

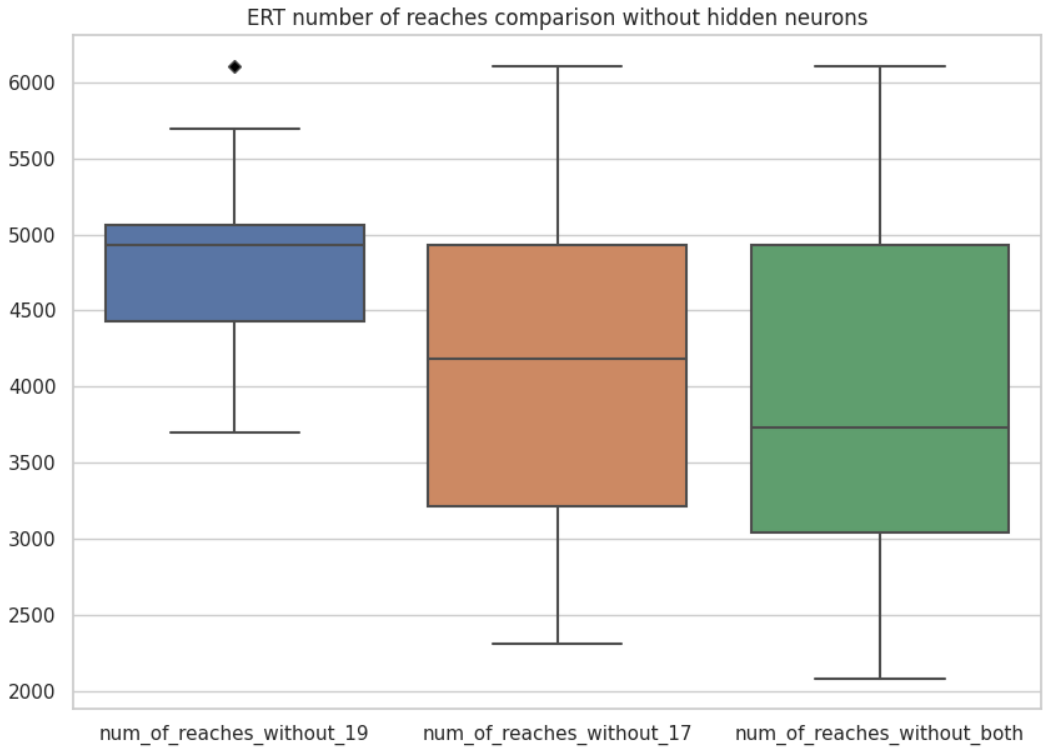

Figure 46: Comparison of number of reaches of the ERT agent without part or all of the hidden neurons in the predictive cluster n = 15.

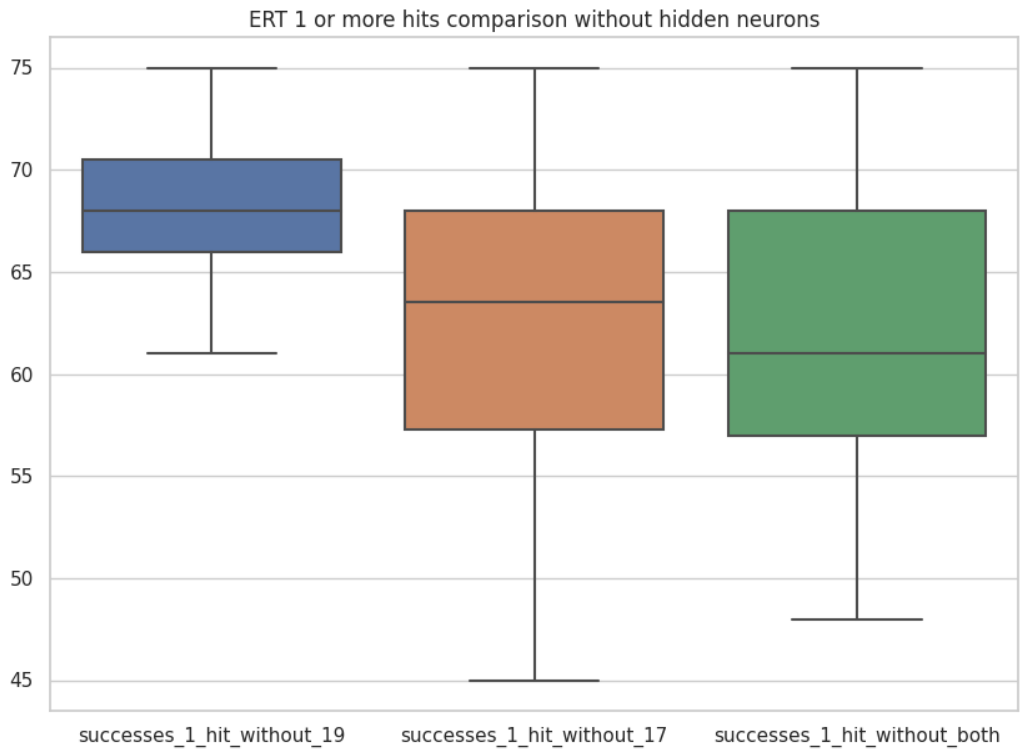

Figure 47: Comparison of the number of at least 1 hit of ERT agent without part or all of the hidden neurons in the predictive cluster n = 15.

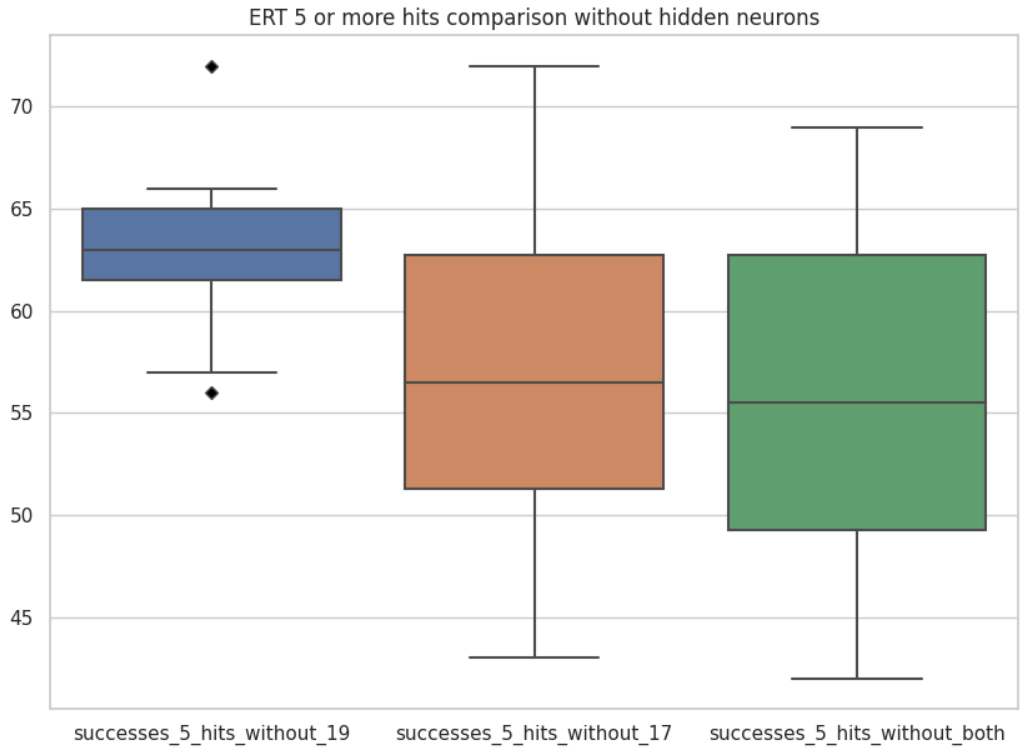

Figure 48: Comparison of the number of at 5_hits of ERT agent without part or all of the hidden neurons in the predictive cluster n = 15.

As mentioned above it is crucial to observe the performance and relationships between the normal ERT agent, the ERT agent with hidden neuron 17 removed, the ERT agent with hidden neuron 19 removed, and the ERT agent with both hidden neurons removed. Which we have done here.

In Figures 42 to 45 since all agents use the same fitness function we can compare the values of the fitness. As shown the agent without hidden neuron 19 shows similar performance to the normal ERT agent shown in graphs above except for one key area which is number of reaches, where the average number of reaches for the agent without hidden neuron 19 is higher than the normal agent. With this we can state that hidden neuron 19 monitors the energy output of the

agent. As a result when it is missing the agent reaches more for the target.

Now observing the agent without hidden neuron 17 we can see performance has degraded throughout all metrics for the agent. The biggest performance degradation has occurred at the number of 5_hits where the average number of hits is the minimum number of hits for the agent without hidden neuron 19. As we use number of 5_hits as our main criteria for prediction, we can conclusively state the prediction capabilities of the agent has degraded due to the removal of hidden neuron 17.

As stated earlier in Chapter 3 the assumed predictive cluster of the ERT agent was comprised of hidden neurons 17 and 19 and when removing both neurons we can see predictive capabilities have severely degraded. Additionally as we have previously stated an element of prediction is the energy criteria of the fitness function which is monitored by hidden neuron 19. So between the removal of what we can state is the energy monitoring neuron and the main predictive neuron prediction has degraded greatly with the removal of both neurons.

5. CONCLUSION

In this thesis, we explored the influence of different fitness factors and topological features of prediction in evolved neural network controllers. We state that prediction is an important foundation of cognitive and intelligent behavior but its evolutionary origin has received little attention. To investigate prediction in cognitive settings, we changed the environment so that it forces an alternative strategy to a simple reaction. By evolving neural networks with 4 fitness function classes (ERT, ER, RT, and R), we found that the energy factor, or a factor that penalizes the use of excessive energy use, contributes significantly to inducing prediction. We also found that there are strongly correlated hidden neurons that highly correlate with the predictive ability of the overall agent. To further investigate prediction we observed the relationship between the number of loops observed in the chromosomes, the fitness functions, and the success rates. As we observed, with an increasing number of loops there was a highly correlated increase in success rates and fitness rates. With that information, we can state that with more loops the predictive capabilities of the agents increase, which also supports the memory-framework theory we discussed in Chapter 1. With these results, we hope to emphasize the network's predictive architecture and bring new insights into the relationship between energy dynamics and predictive behavior in an evolutionary context.

REFERENCES

[1] L. W. Swanson, *Brain Architecture*. Oxford University Press, 2003.

[2] P. D. W Schultz and P. R. Montague, "A neural substrate of prediction and reward," tech. rep., Science (New York, N.Y.), 1997.

[3] E. C. Tolman, "Cognitive maps in rats and men," *Psychology Review*, 1948.

[4] N. Humphrey, "The invention of consciousness," *Topoi*, 2017.

[5] J. Kwon and Y. Choe, "Predictive internal neural dynamics for delay compensation," *IEEE*, 2008.

[6] R. Nijhawan and S. Wu, "Compensating time delays with neural predictions: are predictions sensory or motor?," *The Royal Society Publishing*, 2009.

[7] H. Lim and Y. Choe, "Extrapolative delay compensation through facilitating synapses and its relation to the flash-lag effect," *IEEE Trans Neural Network*, 2008.

[8] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *The MIT Press Journals*, 2002.

[9] J. Y. Qinbo Li and Y. Choe, "Emergence of tool use in an articulated limb controlled by evolved neural circuits," *Proceedings of the International Joint Conference on Neural Networks*, 2015.

[10] T. A. Mann and Y. Choe, "Neural conduction delay forces the emergence of predictive function in simulated evolution," *Nineteenth Annual Computational Neuroscience Meeting*, 2010.

[11] J. Hawkins and S. Blakeslee, *On Intelligence: How a New Understanding of the Brain Will Lead to the Creation of Truly Intelligent Machines*. St. Martin's Griffin, 2005.