Complex Logical Action-State Prediction

by

Justin Schlauwitz

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Software and Intelligent Systems Engineering

Department of Electrical and Computer Engineering

University of Alberta

Abstract

This thesis proposes three novel improvements to the Actor-Critic State-Action-Reward-State-Action algorithm while considering potential biologically equivalent mechanisms. The algorithms are optimized via a Particle Swarm Algorithm, tested on a unigram character prediction problem, and evaluated on bit-wise accuracy and character exactness. Some non-unique changes include, kerneling for flexibility in state encoding options, and mixing historical and predictive information into the algorithm's logical input to supplement non-observable elements. The first contribution is a more flexible δ calculation method which better emulates how neurotransmitters are released, recovered, and lost. The second contribution is w.r.t. the implementation of complex weights and states using a trigonometric interpretation, allowing the algorithm to more clearly distinguish between non-observability and non-existence. The last contribution, bounded error, restricts the maximum output magnitude of the logical predictions in a way that improves weight stability and filtration of influence from states with weak relations to the output.

To God above

I see what you have made and frequently ask: why or for what purpose? You have planted in me the desire to emulate my own likeness and experience the frustrations, joys, and sorrows that come with.

A good name is more desirable than great riches; to be esteemed is better than silver or gold.

– Proverbs 22:1, NIV

Whoever loves discipline loves knowledge, but he who hates correction is stupid.

– Proverbs 12:1, NIV

When pride comes, then comes disgrace, but with humility comes wisdom.

– Proverbs 11:2, NIV

A fool gives full vent to his anger, but a wise man keeps himself under control.

– Proverbs 29:11, NIV

Even a fool is thought wise if he keeps silent, and discerning if he holds his tongue.

– Proverbs 17:28, NIV

Do you see a man wise in his own eyes? There is more hope for a fool than for him.

– Proverbs 26:12, NIV

Acknowledgements

I would like to thank my supervisor, Dr. Petr Musilek, for granting me the opportunity to thrive and study under him. In the time I have spent under his care, he has provided me opportunities to grow as a researcher, public speaker, and teaching assistant. I thank my supervisory committee for being the voice of reason and confrontation. While I was trying to figure out the scope and focus for my research; if I attempted to accomplish as much as initially thought I could have, I would probably need another 20 years. I would also like to acknowledge the Queen Elizabeth II PhD. Scholarship offered by the University of Alberta as it has provided significant financial support. I thank my close relatives for their moral support even though they could not easily grasp the concepts and perspectives of my work — If you're reading this dissertation, I hope what has been written makes it clearer than what was conveyed orally. I will also extend a thank you to my fellow work colleagues who made my time on campus lively and enjoyable. I could start thanking all the people I have learned from inside the classroom, on the job site, *etc.*; however, it would be too long winded, and I am sure they know how sharing their experiences and knowledge can, have, and will create better opportunities for those who receive it.

Contents

List of Tables

List of Figures

List of Symbols

A	The set of all possible actions.
S	The set of all possible states.
$\vec{a}$	The column vector representation of the action space.
$\vec{s}$	The column vector representation of the state space.
a	The scalar/singular action representation.
s	The scalar/singular state representation.
r	The scalar/singular reward.
$\square_t$	The arbitrary item(s) of what is present or available at time t.
$\square_{t+0.5}$	The arbitrary item(s) of what is present or available while transitioning from time t to $t+1$.
$\square_{t+n-0.5}$	The arbitrary summary of what is *expected* to be present or available while transitioning from time t to $t+n$.
$\square_{t-\tilde{m}-0.5}$	The arbitrary summary of what has occurred while transitioning from time $t-\tilde{m}-0.5$ to t.
$\mathbb{E}\{x\}$	Expected value of expression x.
$[\square]$	The default 2 dimensional matrix.
$[\square^n]$	A tensor matrix of n dimensions.
$\square^T$	Transpose of an item.

Chapter 1

Introduction

The focus of this research is to modify the State-Action-Reward-State-Action (SARSA) algorithm in a way that more closely resembles a biological neuron. These modifications are done with the intention of increasing the Learning Algorithm's (LA's) flexibility, stability, and memory density. SARSA was designed to work with state-spaces that have a fixed number of active binary states at any given time, *i.e.* Gray-encoding methods. Similarly, it expects the action space to produce only one active output while all others are inactive. Through some small tricks, it is possible to make SARSA work with Binary-encoded or similar state/action-spaces with limited success. However, a poor choice of reward scale, parameters that do not sufficiently restrain weight growth, and encoding methods other than Gray-coding can cause weights to diverge without limits. The restriction to only use Gray-coding for states and actions is not an issue for small state-spaces and even helps in lowering the learning difficulty in most cases, however, it can become impractical for problems with a large number of states.

Biological neurons are the foundational source of algorithms such as Neural Networks (NNs) and Temporal Difference Reinforcement Learning (TDRL), albeit from different perspectives. This thesis will continue along this avenue because there are aspects of the biological neuron that have been excluded or simplified, but may also offer hints of how to reach this research's objective. Some hints lay in the fact that they operate in an environment where resources are finite and mediums of information come in different forms.

W.r.t. this research, it may be possible to make similar improvements to a NN model, however, certain aspects of TDRL are more favorable for making improvements. TDRL tends to focus on how neurotransmitters that function as rewards are processed within a single layered cluster of neurons. NNs focus on signal processing and transmission between sequentially connected neurons. Non-hybridized NNs usually do not use qualitative rewards in conjunction with prediction error to learn because state/output quality is a generalized measure w.r.t. the current state and all outputs. Instead, methods such as backwards-propagation convert the logical error into a neuron specific qualitative reward for all but the output layer. For an output layer, the total logical error can be applied as a qualitative reward, but getting logical error measures from qualitative reward is not necessarily possible as they may not have a strong relation to each other. As a visual interpretation, neurons inside a NN have cell membranes and inter-cellular connections with the method for processing signals within the cell depending on the chosen flavor of neuron. The inter-cellular interactions and processing methods used in NNs are relatively developed and rigid. Alternatively, TDRL severely lacks the signal processing capabilities found in NNs as well as the cell walls that define the boundary between one cell and the next. Instead, it models the overall interactions of an arbitrary arrangement of competing 'skinless' cells with vague environmental feedback. Given that NNs have been researched for a relatively long time compared to TDRL, there seems to be more potential for unique biologically inspired improvements in TDRL.

Each modification is applied step-by-step to demonstrate the progressive effects they have on the algorithm, gradually bringing about the new algorithm: Complex-Logical-Action-State-Prediction (CLASP). The problem to be used in this research is unigram character prediction. Character-level prediction allows us to use any text-based file as a dataset and is structured w.r.t. sequence, while also being largely stochastic [10]. The difficulty w.r.t. predictability can typically be adjusted by changing the number of characters presented as inputs or including relevant state information about higher level data. The LAs will be optimized with a focus on hit-rate while bit-wise accuracy is second, *i.e.*

getting the exact binary sequence for a character as much as possible is more important than getting a character that is similar in binary.

1.1 Motivation

Most algorithms tend to focus on using presented observations and internally set states to decide which actions to take. Some may even use internal states with dual properties (*i.e.* negative state values), but one would be hard pressed to find an algorithm that explicitly considers non-dual predictions of its observed states and attempts to strike a balance between observation and prediction. This is not without reason: adding too much useless or conflicting information may lead to a case of 'garbage-in-garbage-out.' A learning method that mitigates, if not entirely filters out the effects of noisy state data, will be desirable. A method that better ensures learning stability w.r.t. reward would be worth including. Weight transparency is another aspect to be considered, but not strictly focused on in this research. The primary objective of this research is not to produce a perfected product; however, it is expected to show new ways that TDRL can be improved through biologically inspiration. Though adding to SARSA is expected to increase time and memory requirements, if the flexibility of encoding, compression of information, stability of learning, and weight clarity can be improved, it may offer a desirable starting point for further research focusing on rigor and clarity.

1.2 Objectives

To improve SARSA, a series of modifications will be progressively implemented and tested. The first step is to try and improve the δ calculation method to reduce the likelihood of having state-action quality predictions diverge — a common way that SARSA has been found to fail in highly non-stationary environments. This change will consider additional aspects of how neurotransmitters can be manipulated. The next objective will be to implement complex weights. This will serve to increase the amount of information that can be stored for each state. As complex logical states already deviate from

the norm of logic and probability, an alternative perspective will be proposed: logical values will be assumed to have units in the form of energy or amplitude, *i.e.* they will no longer be considered unit-less. This change in perspective is expected to allow weights to be viewed as trigonometric relations with high degrees of similarity to conventional logical operations, instead of as purely logical or probabilistic components. Combining trigonometry with complex numbers is expected to increase the potential flexibility of a given weight element by permitting intermediate forms of the fundamental logical operations. The last objective is to improve the logical stability. This will be done with consideration of how neurons are limited in size by the number of resources at their disposal, *i.e.* limiting roughly how large the weights used can become. A limitation on the size of the logical error will also be imposed in expectation of improving the stability of logical error calculations — making the algorithm more compatible with non-gray-coding methods in chaotic state-spaces. In short, w.r.t. SARSA, this research aims to:

- improve on the δ calculation method to gain better stability,
- allow for alternative weight interpretations, and
- improve on the logical error calculation's stability to better handle chaotic densely coded state information.

Chapter 2

Background

2.1 Conventional Logic Faults When Expectation Deviates From Application

There are two faults to be addressed in this chapter: the assumption of knowledge about the dual H of a subset T when only T is given without guarantee that $X = \{0\}$; and the oversight that, denial and non-observability are both mapped into 0. These issues are usually ignored because we turn a blind eye to X, assuming it will likely never happen, and focus on the duality of interest. However, for Learning Algorithms (LAs) that are required to develop their own logical circuits, it is very unlikely for the internal logic equations to abide by the rules and restrictions provided in the field of probability, *e.g.* producing values outside the range of $[0, 1]$ if an explicit bounding function is not imposed. The consequence of this is that the LA's X set is not the empty set $\{0\}$ and is not insignificant — as is the case learning has converged to a deterministic policy. Because X is not insignificant, it is necessary to address the problems that arise in our conventional interpretations before working to find a resolution.

2.1.1 Inherent Assumption of Complete Knowledge

One of the requirements for probability to work is that '*all*' outcomes must be known/represented in some form, *i.e.* for all elements in the universe of

discourse Z[4], [23]:

$$1 = \int_{Z} \mathbb{P}(Z), \tag{2.1}$$

however, though possible in theory, this is often impossible in practice, *i.e.* for $S \subseteq Z$ such that $Z = S \cup X$:

$$1 \geq \sum_{S} \mathbb{P}(S). \tag{2.2}$$

To make this easier to comprehend, a coin flip example will be used to represent a simple two state problem. Theoretically, the ideal coin flip game has two outcomes $S = \{H, T\}$ with claimed equal probability $\{0.5, 0.5\}$ where:

$$T = \neg H \tag{2.3}$$

and

$$H = \neg T. \tag{2.4}$$

I.e. H and T are mutually opposed elements that cannot or at least should not occur together or be entirely absent [4], [23]. However, given all the possibilities of reality, we must acknowledge that there is some possibility for an unknown and unaccounted for state $X = Z - S$, *e.g.* the coin landing on its side, which causes the inequalities:

$$(1 - \mathbb{P}(T)) \geq \mathbb{P}(\neg T) \tag{2.5}$$

and

$$(1 - \mathbb{P}(H)) \geq \mathbb{P}(\neg H) \tag{2.6}$$

or the equalities

$$(1 - \mathbb{P}(H)) = \mathbb{P}(X) + \mathbb{P}(T) \tag{2.7}$$

and

$$(1 - \mathbb{P}(T)) = \mathbb{P}(X) + \mathbb{P}(H). \tag{2.8}$$

To rectify this, one must use a more elaborate equation that accounts for the anomaly:

$$((1 - \mathbb{P}(X)) - \mathbb{P}(T)) = \mathbb{P}(H) \tag{2.9}$$

where $\mathbb{P}(H \cup T) = 1 \rightarrow (1 - \mathbb{P}(X))$. Given $\mathbb{P}(X)$ is unknown, the assumption:

$$T = \neg H \implies \mathbb{P}(T) = (1 - \mathbb{P}(H)) \tag{2.10}$$

only holds under the assumption that all states are accounted for, *i.e.* the 1 in $\mathbb{P}(T) = (1 - \mathbb{P}(H))$ implies that $S = Z$ and T and H are mutually exclusive, *i.e.* $X = \{0\}$. Let us return to the original problem of the coin flip, but with a case where the assumption of exclusivity is false, *e.g.* a second identical coin gets mixed in during the flip and it is now possible for H and T to appear simultaneously. It would imply the original rules of the game were violated, but this is also a valid state outside the expected set of outcomes should it come into existence. If the probability of such an outcome were to be accounted for, then one could write the probabilistic sum:

$$1 = \mathbb{P}(H) + \mathbb{P}(T) - \mathbb{P}(T \cap H). \tag{2.11}$$

which can be rearranged to get equation (2.7) or (2.8) if it is permitted for $-\mathbb{P}(T \cap H) \subseteq \mathbb{P}(X)$. The consequence of not establishing rules that account for violations of exclusivity, but do consider all other possibilities, would result in:

$$1 \leq \sum_{\forall S} \mathbb{P}(S). \tag{2.12}$$

The range for $\mathbb{P}(X)$, when accounting for unexpected outcomes and contradictions, is $-\infty < \mathbb{P}(X) \leq 1$ if $0 \leq \#S < \infty$. The lower bound of $\mathbb{P}(X)$ is determined by the correctness of the rules w.r.t. the relations of the elements within S while the upper bound is determined by how closely S equates to Z.

In summary, conventional logic has little to no tolerance for incomplete or inaccurate rule sets — there exists an inherent assumption of infallible prior knowledge; however, if the completeness and accuracy is sufficiently close to the ideal theoretical scenario, it can be regarded as, and often is acceptable for implementation. It should be noted that w.r.t. intelligent systems that are required to learn their own logical policies, $|\mathbb{P}(X)|$ can only be regarded as being minimized after learning has converged — even then, it may still not be insignificant enough to be ignored. Regarding whether or not a probability

should be allowed to exceed the range $[0, 1]$ is another issue that must be addressed separately.

2.1.2 Grouping of Denial and Non-Observability; and Value and Inconclusivity

Through the research thus far, attempts have been made to distinguish between logical values and probabilities. This is because, though similar, they represent two very different aspects: all logical values, and only two probabilistic values, $\{0, 1\}$, suggest a definite outcome, while probabilities in the range $(0, 1)$ suggest indefinite outcomes. Even then, the interpretation of $\mathbb{P}(Y) = 0$ and $\mathbb{V}(Y) = 0$ depend on the observability of the duality of Y.

Logical values state the strength of a conclusive statement, *e.g.* the degree to which something observably, in part or whole, exists. The duality of $\mathbb{V}(Y) > 0$ pertains to the non-confidence in the statement, *e.g.* the lack of supporting evidence, while the duality of $\mathbb{V}(Y) = 0$ is a full declaration of non-existence; however, a lack of supporting evidence is *not* the same as a degree of confidence an observation's dual. *E.g.* on a moderate day, one person asks: is it hot outside? The one being asked could potentially answer with: it is bright or it is cold. Of the two answers, one is irrelevant, *i.e.* a lack of supporting evidence for affirmation, while the other is true on the premise that 'cold' is the dual of 'hot,' and both cannot be *directly* relied upon and thus should be mutable, *i.e.* 0, because both responses are disjointed from the question which expects a truthful and direct reply regarding heat — yes or no. The word 'directly' is placed under emphasis because, if there exists additional rules governing the relation between radiance and duality of the source respectively to the prompt, one may be able to make a conclusion; however, such considerations are not made within the logic compiled by the question, and if the premise of duality is established, there is still the problem of potentially lost information during the interpretation of 'cold' to 'not hot.' The loss comes from the fact that, linguistically speaking, there are several degrees of relative 'hot' and 'cold' that do not have an equivalent in the opposing spectrum. This is to say that, w.r.t. conclusive statements made with conventional logic in response to 'is it

hot outside,' *i.e.* $S_{\mathbb{V}} = \{Y, N\}$, using values in the spectrum of 'hot' can be interpreted with:

$$\mathbb{V}(Y) > 0 \implies Y_h = 1 \implies N_h = 0 \tag{2.13}$$

and

$$\mathbb{V}(Y) = 0 \implies Y_h = 0 \implies N_h = 1, \tag{2.14}$$

i.e. a statement of even the smallest non-zero magnitude in affirmation constitutes Y otherwise it is N. Naturally, this also applies when swapping N and Y.[1] However, from this information, it can only be concluded if it is hot or not and cannot claim if it is cold or lukewarm, unless there are additional questions related to said temperature groups. For a discrete case where the direct inverse states are observable, it can be an acceptable answer with w.r.t. the inverse statement, but for a case where the dual is not observable, *e.g.* existence and non-existence, a *direct* claim that something does not exist simply because it is not observable is erroneous, *i.e.*:

$$\mathbb{V}(Y_h) = \begin{cases} 1 & \text{If } \mathbb{V}(Y) > 0, \\ 0 & \text{Otherwise,} \end{cases} \tag{2.15}$$

and

$$\mathbb{V}(Y_h) = x \implies N_h = 1 - x. \tag{2.16}$$

From these sets of implications, when regarding the matter of temperature, there is more information relevant to the question contained in the response directly coinciding with the spectrum of said question than in a dual value that is not diametrically opposed.

If logic represents a conclusive statement, then probabilistic values represent the possibility that something will/does/did exist/occur. Probability is not concerned with the value of a statement, but more concerned about the possibility and reliability of said value, *e.g.* the possibility of $Y = 1$ or false negative $N = 1$ of a photo-voltaic's activity in a circuit given brightness level B. It is important to note that the set of claims and conclusions are similar but different, *i.e.* $S_{\mathbb{P}} = \{Y, N\} \neq S_{\mathbb{V}} = \{Y, N\}$. The probability that something

[1] This set of implications does not work for X because it is not part of the duality relation.

will occur with $\mathbb{P}(Y) = x$ implies that it will not occur with $\mathbb{P}(N) = 1 - x$. The implicit part of this duality relation is that one always implies the opposite for the other which is characteristic of leading questions, *e.g.* if one claims that Schrödinger's Cat is dead you imply that it is not alive; and the use of a non-vectorized value implies that one is not fully excluded/included unless the other is fully accepted/denied respectively, *e.g.* a bird is not a cat nor a dog and white contains both red and green but each question can only be answered with one of the two candidate solutions: cat or dog and red or green. The problem perceived from these cases is when we rely on $\mathbb{P}(Y) = \mathbb{P}(N) = 0.5$ as a mutable case and assume $\mathbb{P}(Y) + \mathbb{P}(N) = 1$ when not all dualities are proper duals — as mentioned in the previous section where $X \neq \{0\}$. To examine this further, let us scrutinize the two questions: is it hot and does it exist? The first question is an example of 'cold' not being a proper dual to 'hot,' and the second question, though more subtle, implies that non-observable equates to non-existent. For clarity, it is necessary to state the relation between the soft/continuous $\mathbb{P}(Y_s)$ and hard/discrete $\mathbb{P}(Y_h)$ as:

$$\mathbb{P}(Y_h) = \int_{Y_s \in Y} \mathbb{P}(Y_s), \tag{2.17}$$

which also applies to N and X. Unfortunately, $\mathbb{P}(X_h)$ and $\mathbb{P}(X_s)$ are unknown because they include outcomes outside of our expectation S, however, this does not stop us from learning some information about $\mathbb{E}(X_h)$ and $\mathbb{E}(X_s)$ through indirect means. The expected logical value of affirmation can be calculated as:

$$\begin{aligned} \mathbb{E}(Y_s) &= \int_{Y_s \in Y} \mathbb{V}(Y_s) \times (\mathbb{P}(Y_s|Y) + \mathbb{P}(Y_s|N) + \mathbb{P}(Y_s|X)) \\ &= \int_{Y_s \in Y} \mathbb{V}(Y_s) \times \mathbb{P}(Y_s) \end{aligned} \tag{2.18}$$

while

$$\mathbb{E}(Y_s) \leq \mathbb{E}(Y_h) = \mathbb{V}(Y_h) \times \mathbb{P}(Y_h) = \mathbb{V}(Y_h) \times \int_{Y_s \in Y} \mathbb{P}(Y_s). \tag{2.19}$$

This inequality can also be presented more broadly as:

$$\begin{aligned}
\mathbb{E}(Y_s) \leq & \quad \mathbb{E}(\neg N_h) - \mathbb{E}(X_h) \\
\leq & \quad \mathbb{V}(\neg N_h) \times (\mathbb{P}(\neg N_h|N) + \mathbb{P}(\neg N_h|Y) + \mathbb{P}(\neg N_h|X)) - \mathbb{E}(X_h) \\
\leq & \quad \mathbb{V}(\neg N_h) \times \mathbb{P}(\neg N_h) - \mathbb{E}(X_h) \\
\leq & \quad (1 - \mathbb{V}(N_h)) \times (1 - \mathbb{P}(N_h)) - \mathbb{E}(X_h) \\
\leq & \quad (\mathbb{V}(Y_h) + \mathbb{V}(X_h)) \times (\mathbb{P}(Y_h) + \mathbb{P}(X_h)) - \mathbb{E}(X_h) \\
\leq & \quad \mathbb{V}(Y_h) \times \mathbb{P}(Y_h) + \mathbb{E}(X_h) + \mathbb{V}(Y_h) \times \mathbb{P}(X_h) + \mathbb{V}(X_h) \times \mathbb{P}(Y_h) - \mathbb{E}(X_h) \\
\leq & \quad \mathbb{E}(Y_h) + \mathbb{V}(Y_h) \times \mathbb{P}(X_h) + \mathbb{V}(X_h) \times \mathbb{P}(Y_h), \qquad (2.20)
\end{aligned}$$

given the relations:

$$\mathbb{V}(\neg N_h) = (1 - \mathbb{V}(N_h)) = \mathbb{V}(Y_h) + \mathbb{V}(X_h). \qquad (2.21)$$

The relation in equation (2.21) in reasonable because, anything that is not N_h is expected to be Y_h, however, should it be neither or both, then $\mathbb{V}(X_h)$ would be non-zero to enforce the sum of values equal to 1 as is done for the probability in the previous section. If both methods of calculating $\mathbb{E}(Y_s)$ are acceptable, then:

$$\mathbb{E}(Y_h) - \mathbb{E}(Y_s) \geq -\mathbb{V}(Y_h) \times \mathbb{P}(X_h) - \mathbb{V}(X_h) \times \mathbb{P}(Y_h), \qquad (2.22)$$

Using the same approach for N_s, we get the inequality:

$$\mathbb{E}(N_h) - \mathbb{E}(N_s) \geq -\mathbb{V}(N_h) \times \mathbb{P}(X_h) - \mathbb{V}(X_h) \times \mathbb{P}(N_h), \qquad (2.23)$$

Given that the left hand side is the region to which a declaration is lacking, we can expect that they are part of X_s or the respective dual component, given by the other equation. For regions outside of S and $Y_h \cap N_h \neq 0$, X_h takes on the necessary value to ensure these inequalities. To define $\mathbb{E}(X_s)$ w.r.t. Z give:

$$\begin{aligned}
\mathbb{E}(X_s) = & \int_{X_s \in X} \mathbb{V}(X_s) \times \mathbb{P}(X_s) \\
= & \quad \mathbb{E}(N_h) - \mathbb{E}(N_s) + \mathbb{E}(Y_h) - \mathbb{E}(Y_s) + \mathbb{E}(X_h), \qquad (2.24)
\end{aligned}$$

which is in accordance with equation (2.21). Considering the other side of the inequalities, gives:

$$\mathbb{E}(X_s) \geq \mathbb{E}(X_h) - \mathbb{V}(Y_h) \times \mathbb{P}(X_h) - \mathbb{V}(X_h) \times \mathbb{P}(Y_h) \\ - \mathbb{V}(N_h) \times \mathbb{P}(X_h) - \mathbb{V}(X_h) \times \mathbb{P}(N_h), \quad (2.25)$$

which can be rearranged into:

$$\mathbb{V}(X_h) \times \mathbb{P}(X_h) = \mathbb{E}(X_h) \leq \mathbb{V}(Y_h) \times \mathbb{P}(X_h) + \mathbb{V}(X_h) \times \mathbb{P}(Y_h) + \mathbb{E}(X_s) \\ + \mathbb{V}(N_h) \times \mathbb{P}(X_h) + \mathbb{V}(X_h) \times \mathbb{P}(N_h). \quad (2.26)$$

A problem that arises here is that $\mathbb{E}(X_s)$ can only be calculated with the sum of unknown values and probabilities or by requiring $\mathbb{E}(X_h)$ which also has the unknown component $\mathbb{P}(X_h)$. What is known is that $\mathbb{E}(X_h) = 0$ when $Y_h = \neg N_h$ and $\mathbb{E}(X_h) = \mathbb{E}(X_s) = 1$ if it is observed that $Y_h = N_h = 0$, however, this does not give the value of $\mathbb{E}(X_s)$ when $\mathbb{E}(X_h) \neq 1$, thus the inequality.

Going back to equation (2.21), if there is any instance where $\mathbb{V}(X_h) \neq 0$ it is guaranteed that inequality (2.26) will be influenced by $\mathbb{P}(Y_h) = \mathbb{P}(N_h) = x > 0$, *i.e.* a non-zero value suggesting indifference is not mutable if $\mathbb{P}(X_h) = 1 - 2 \times x \neq 0$. If we apply these expected value equations to the question: does it exist, since the dual, *i.e.* non-existence, cannot be observed to any degree and realistic sensors are not infallible and omniscient, we must conclude that $N \subset X$ and $\mathbb{E}(X_h) = 1 - \mathbb{E}(Y_h)$. This can be implemented by setting $\mathbb{P}(N) = 0$ and $\mathbb{V}(N) = 0$ which allows equation (2.24) to reduce to:

$$\begin{aligned} \mathbb{E}(X_s) - \mathbb{E}(X_h) = & \quad \mathbb{E}(Y_h) - \mathbb{E}(Y_s) \\ = & \quad (1 - \mathbb{E}(X_h)) - (1 - \mathbb{E}(X_s)). \end{aligned} \quad (2.27)$$

This allows a reduction for the inequality between $\mathbb{E}(X_s)$ and $\mathbb{E}(X_h)$ to become:

$$\mathbb{V}(X_h) \times \mathbb{P}(X_h) = \mathbb{E}(X_h) \leq \mathbb{V}(Y_h) \times \mathbb{P}(X_h) + \mathbb{E}(X_s) + \mathbb{V}(X_h) \times \mathbb{P}(Y_h) \quad (2.28)$$

or

$$\begin{aligned} \mathbb{V}(Y_h) \times \mathbb{P}(X_h) + \mathbb{V}(X_h) \times \mathbb{P}(Y_h) \geq & \quad \mathbb{E}(X_h) - \mathbb{E}(X_s) \\ \geq & \quad \mathbb{E}(Y_s) - \mathbb{E}(Y_h). \end{aligned} \quad (2.29)$$

Due to X being diametrically opposed to Y, we can claim that:

$$\mathbb{V}(X_h) = 1 - \mathbb{V}(Y_h) \tag{2.30}$$

and

$$\mathbb{P}(X_h) = 1 - \mathbb{P}(Y_h), \tag{2.31}$$

therefore:

$$\begin{aligned}
\mathbb{E}(X_s) \geq & \quad \mathbb{E}(X_h) - \mathbb{V}(Y_h) \times \mathbb{P}(X_h) - \mathbb{V}(X_h) \times \mathbb{P}(Y_h) \\
\geq & \quad (1 - \mathbb{V}(Y_h)) \times (1 - \mathbb{P}(Y_h)) - \mathbb{V}(Y_h) \times \mathbb{P}(X_h) - \mathbb{V}(X_h) \times \mathbb{P}(Y_h) \\
\geq & \quad 1 + \mathbb{E}(Y_h) - \mathbb{V}(Y_h) \times (2 - \mathbb{P}(Y_h)) - (2 - \mathbb{V}(Y_h)) \times \mathbb{P}(Y_h).
\end{aligned} \tag{2.32}$$

From this inequality, even though N was removed from the scope of the problem, it is still not possible to find a solution for the expected value of X_s.

The conclusion regarding conventional logic and probability is that, it is viable only in cases where the claims and conclusions consider all possibilities and establish proper dualities w.r.t. interpretation. If the established duality is improper or incomplete, there is a risk of encountering logical errors when relying on states derived from the inverse of a prior claim/conclusion. In cases where $X \neq \{0\}$, there are situations where $\mathbb{V}(X)$ and $\mathbb{P}(X)$ are required to exceed the range $[0, 1]$ to maintain the range for valid responses in S which further implies that rules of conventional logic cannot be guaranteed to hold when situations beyond the limits of expectation occur.

2.2 Complex Logic and Signed Probability

Due to the necessity to be able to handle conflicting, fallible and/or irrelevant logical information, it was necessary to implement a method of representing $\mathbb{V}(Y)$ and $\mathbb{P}(Y)$ and their claimed duals $\mathbb{V}(N)$ and $\mathbb{P}(N)$ separately but in a way that can still demonstrate their relations. The requirement for logical representation is that the observable/real value must reflect what is seen in conventional logic, however, the process of switching between duals must be

lossless, *i.e.* no data compression. The requirements for probabilities build off of the understanding of logic, but must either be applicable to the conventional operations or have an alternative that displays equivalent behavior. It should be noted that the primary interest of this chapter is not to give a thorough proof, as would be found in a dissertation from a department of mathematics, but to give sufficient explanation of the concept for justifiable implementation and interpretation.

2.2.1 Complex Valued Logic

To improve on conventional logic, the properties of complex numbers are believed to have the best matching characteristics as a way to implement $\mathbb{V}(Y)$ and $\mathbb{V}(N)$:

$$\mathbb{L}(Y) \equiv \mathbb{L}(S) = \mathbb{V}(Y) + \hat{\imath}\mathbb{V}(N) \tag{2.33}$$

and

$$\mathbb{L}(N) \equiv \mathbb{V}(N) + \hat{\imath}\mathbb{V}(Y), \tag{2.34}$$

where $\mathbb{L}(S)$ is the more generalized expression, taking Y as an affirmation and N as a denial. If N is used to represent the dual of Y, the relation between $\mathbb{L}(Y)$ and $\mathbb{L}(N)$ is established as a rotation about $1 + \hat{\imath}1$:

$$\mathbb{L}(N) = \sqrt{-\mathbb{L}^2(Y)}. \tag{2.35}$$

or more generally:

$$\mathbb{L}(\overline{S}) = \sqrt{-\mathbb{L}^2(S)^*} = \hat{\imath}\mathbb{L}(S)^*. \tag{2.36}$$

As with conventional logic, the *inverse* value relative to the focus is summarized as zero w.r.t. the real value; however, unlike conventional logic, the data stored on the imaginary axis is not lost when considering the entirety of the logic value. The consequence of this is that $N = Y$ — a commonly found property in logical paradoxes — places the logic vector on the axis of rotation, and though less meaningful, still gives a non-zero logic value which suggests that a conclusion exists, *e.g.* admitting that it does not know [23]. Such a property is useful for differentiating between a non-statement and a conflicting statement. It is also possible to claim $N = \{0\}$, *i.e.* the dual literally does

not exist, to which the inverse, $\overline{S}$, is properly non-observable while the real component of $\mathbb{L}(S)$ projected to the imaginary axis is expressed in a way that does not interfere with the real axis. Discarding the relation:

$$1 = \mathbb{V}(Y_h) + \mathbb{V}(N_h) + \mathbb{V}(X_h), \tag{2.21}$$

is necessary as it is not capable of performing as it did before in this context. In its stead, the following equality will be used:

$$1^2 = \mathbb{L}^2(X) + \mathbb{L}(S) \times \mathbb{L}(S)^* = \mathbb{L}^2(X) + \mathbb{V}^2(Y) + \mathbb{V}^2(N), \tag{2.37}$$

will be used. This new relation is based on the radial distance, where $\mathbb{L}^2(X)$ takes on the deviation from the unit circle assuming all values of N and Y are positive. This method gives the implication that the real and imaginary components of $\mathbb{L}(X)$ represent regions where S is deficient or excessive respectively. In the ideal scenario, $\mathbb{L}(X) = 0$ as this would imply that N and Y together are able to fully and properly represent Z, *i.e.* they are diametrically opposed in accordance with the equation for the unit circle. If we wish to specifically identify the contradiction $Y \cap N$, we can use the imaginary part of $\mathbb{L}^2(S)$, *i.e.*:

$$\begin{aligned} \mathbb{V}^2(Y \cap N) = &\quad \mathbb{I}\{\mathbb{L}^2(S)\} = 2 \times \mathbb{V}(Y) \times \mathbb{V}(N) \\ \mathbb{V}(Y \cap N) = &\quad \sqrt{2} \times \sqrt{\mathbb{V}(Y)} \times \sqrt{\mathbb{V}(N)}, \end{aligned} \tag{2.38}$$

which gives the largest combined value $\mathbb{V}(Y \cap N) = 1$ at $\mathbb{V}(Y) = \mathbb{V}(N) = 1/\sqrt{2}$ given $\mathbb{L}(X) = 0$. It should be clarified that $\mathbb{L}^2(X)$, assuming it can be found, only tells us if the logical values are exaggerated or reserved/deficient, but do not tell us if they are perfect duals; thus, equation (2.38) is not something that should be carelessly discarded when evaluating a set of logical operations. This statement of conflicting information between Y and its assumed dual N inherently suggests the degree to which the logical input is incomplete. The difference between X here and X from chapter 2.1 is that the resulting $\mathbb{L}(X)$ here may not be well contained. However, to ensure its magnitude is bounded, it is sufficient to use a squashing function to establish a tolerable limit w.r.t. exaggeration for which all values greater are considered equally

large. This same squashing/bounding function can be applied to ensure that output values remain in the valid range when they are to be used again for further calculations, *e.g.*:

$$\mathbb{R}\{\mathbb{L}(S)\}\begin{cases}\dfrac{\mathbb{R}\{\mathbb{L}(S)\}}{||\mathbb{R}\{\mathbb{L}(S)\}||} & \text{If } ||\mathbb{R}\{\mathbb{L}(S)\}|| \geq 1,\\ \mathbb{R}\{\mathbb{L}(S)\} & \text{Otherwise,}\end{cases} \tag{2.39}$$

and

$$\mathbb{I}\{\mathbb{L}(S)\}\begin{cases}1 & \text{If } \mathbb{I}\{\mathbb{L}(S)\} \geq 1,\\ 0 & \text{If } 0 \geq \mathbb{I}\{\mathbb{L}(S)\},\\ \mathbb{I}\{\mathbb{L}(S)\} & \text{Otherwise.}\end{cases} \tag{2.40}$$

2.2.2 Signed Probability

Assuming logic and probability can be handled similarly, what was uncovered in the previous section can carry over, however, a distinction should be made:[2]

$$\mathbb{P}(Y \oplus N) = \mathbb{R}\{\mathbb{L}^2(S)\} = \mathbb{R}\{\mathbb{P}(S)\} = \mathbb{P}(Y) - \mathbb{P}(N). \tag{2.41}$$

This is acceptable because all conflicting information is placed on the imaginary axis, becoming mutable when only the real component is used. If information regarding conflict is relevant, it is possible to restate equation (2.38) as:

$$\mathbb{P}(Y \cap N) = \mathbb{I}\{\mathbb{L}^2(S)\} = \mathbb{I}\{\mathbb{P}(S)\} = 2 \times \sqrt{\mathbb{P}(Y)} \times \sqrt{\mathbb{P}(N)}. \tag{2.42}$$

To match with equation (2.37), probability $\mathbb{P}(S_h)$ will have to be defined as:[3]

$$\mathbb{P}(||S_h||) = \int_{S_s \in S} ||\sqrt{\mathbb{P}(S_s)}||^2 = \mathbb{V}^2(Y_h) + \mathbb{V}^2(N_h) = \mathbb{L}(S_h) \times \mathbb{L}(S_h)^*. \tag{2.43}$$

If conflicting information exists, it would be prudent to keep in mind that:

$$||\mathbb{P}(Y \oplus N)|| \leq \mathbb{P}(||S_h||) \leq ||\mathbb{P}(Y \oplus N)|| + \mathbb{P}(Y \cap N) = \mathbb{P}(||S||). \tag{2.44}$$

This inequality is worth noting as the XOR probability $\mathbb{P}(Y \oplus N)$ removes all conflicting information while said information is retained in $\mathbb{P}(S_h)$, and if

[2]For this work it is assumed that all logical/probabilistic values of Y and N are positive.

[3]$||\sqrt{\mathbb{P}(S_s)}||^2 \equiv \sqrt{\mathbb{P}(S_s)} \times \sqrt{\mathbb{P}(S_s)}^*$. Where subscript s and h denote the soft or continuous probability space and hard or discrete space.

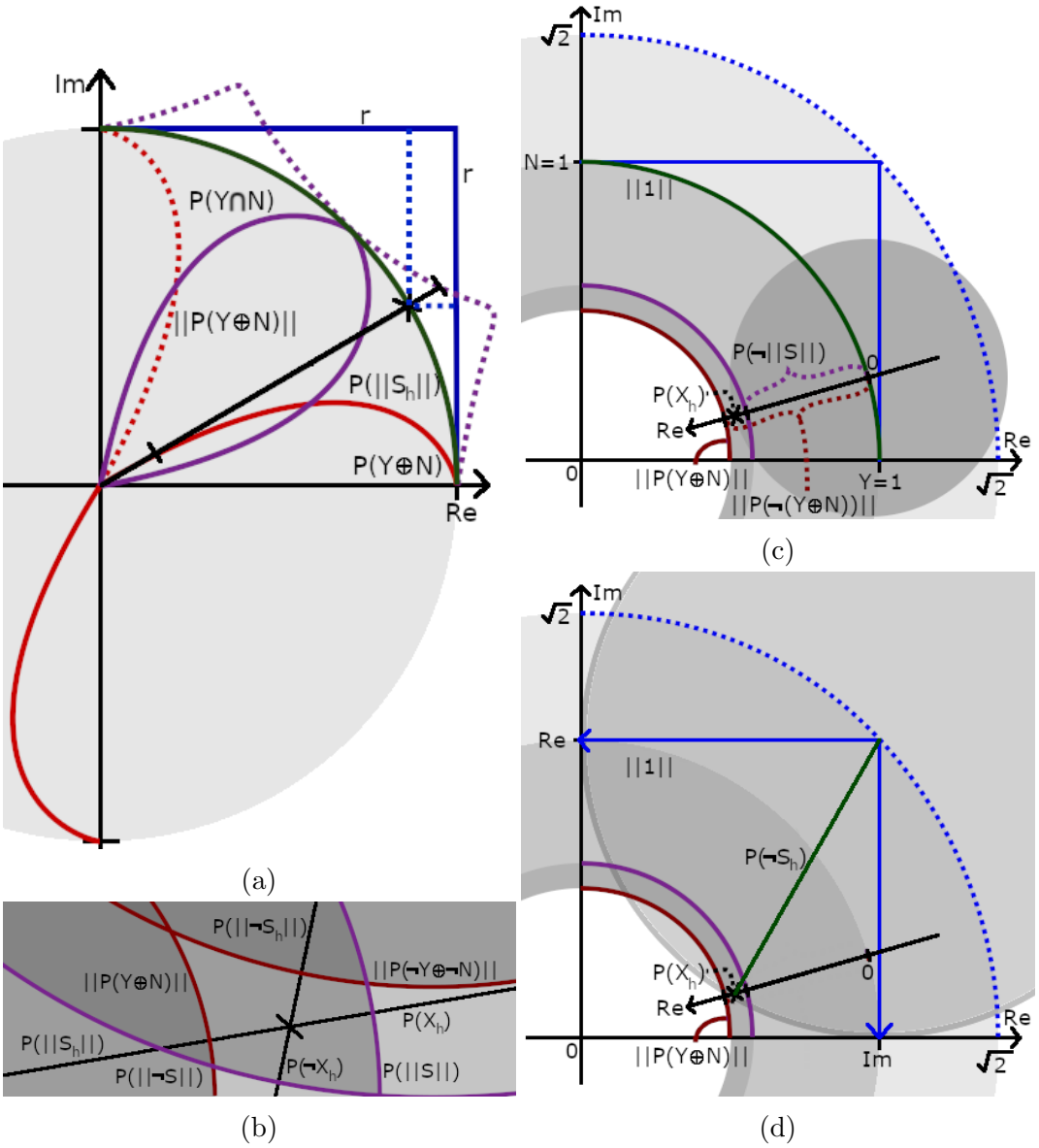


Figure 2.1: These figures demonstrate several basic properties of signed probability given complex logical values. (a) demonstrates the different probabilistic values (given as the radius) that result from a set of logical values (given as real and imaginary values). r is an arbitrary maximum value that allows for a circle to be formed with $\mathbb{L}(S)$, *i.e.* $r = \sqrt{Y^2 + N^2}$. This figure also demonstrates how $\mathbb{P}(||S_h||)$ (the green line on the circle) is bounded by $||\mathbb{P}(Y \oplus N)||$ (the red solid and dotted line in the positive quadrant) and $||\mathbb{P}(Y \oplus N)|| + \mathbb{P}(Y \cap N)$ (the dotted purple line outside the circle). (c) shows how $\mathbb{P}(X_h)$ can be approximated by its respective upper and lower bound demonstrated in (a). (d) shows how the negation $\mathbb{P}(\neg S)$ relates to the elements of $\mathbb{P}(S)$ to which (a) and (c) can be reapplied to get what is shown in (b). (c) demonstrates how the intersections of inequalities for S and $\neg S$ form a closed region surrounding point S_h. The two radial lines (black intersecting lines) also demonstrate the disjoint relation between S and $\neg S$.

the logical AND probability $\mathbb{P}(Y \cap N)$ is added to the lower bound $||\mathbb{P}(Y \oplus N)||$, a value greater or equal to the probability of the hard value S_h is guaranteed. A rough visualization has been provided in figure 2.1(a). This means that, if all conflicting information is removed, the set of inequalities will become equalities. It should also be noted that, for $0 \leq Y$ and $N \leq 1$, we get $0 \leq ||\mathbb{P}(Y \oplus N)|| \leq 1$ and $0 \leq \mathbb{P}(||S||) \leq 2$.

Reinterpreting equation (2.37) to probabilities:

$$1^2 = \int_{S_s \in S} ||\sqrt{\mathbb{P}(S_s)}||^2 + \int_{X_s \in Z} \mathbb{P}(X_s) = \mathbb{P}(||S_h||) + \mathbb{P}(X_h). \tag{2.45}$$

X is processed differently here compared to before because S contains signed information pertaining to Y and N while X is only concerned with the magnitude of S and the areas where it is deficient or excessive.

Operations: Inverse and Negation

The first operation to be proposed will be a simple extension of the logical inverse:

$$\mathbb{P}(\overline{S}) = -\mathbb{P}(S)^*. \tag{2.46}$$

It is also necessary to establish a claim that the inverse is distinctly different from the negations:

$$\mathbb{P}(\neg||Y \oplus N||) = 1 - ||\mathbb{P}(Y \oplus N)||, \tag{2.47}$$

$$\mathbb{P}(\neg||S||) = 1 - \mathbb{P}(||S||), \tag{2.48}$$

and

$$\begin{aligned} \mathbb{P}(\neg S) = &\quad (1 + \hat{\imath}1 - \sqrt{\mathbb{P}(Y \oplus N) + \hat{\imath}\mathbb{P}(Y \cap N)})^2 \\ = &\quad (1 + \hat{\imath}1 - \sqrt{\mathbb{P}(S)})^2, \end{aligned} \tag{2.49}$$

which can be separated into its elements:

$$\mathbb{P}(\neg Y \oplus \neg N) = \mathbb{R}\{\mathbb{P}(\neg S)\} \tag{2.50}$$

and

$$\mathbb{P}(\neg Y \cap \neg N) = \mathbb{I}\{\mathbb{P}(\neg S)\}. \tag{2.51}$$

The key difference between inversion and negation is that the inverse serves to swap the interpretation of Y and N which may not be true dualities while the negation considers the true dual of S or its elements (see figure 2.1(d)).

When considering equation (2.44), it can also be claimed that:

$$\mathbb{P}(\neg||Y \oplus N||) \geq \mathbb{P}(\neg||S_h||) = \mathbb{P}(X_h) \geq \mathbb{P}(\neg||S||). \tag{2.52}$$

If $0 \leq Y$ and $N \leq 1$, then $1 \geq \mathbb{P}(X_h) \geq -1$ is known. Equation (2.49) is distinctly different from the other two negations because it deals with what is not being declared, *e.g.* if the elements of $\sqrt{\mathbb{P}(Y)}$ are the probabilities of two supposedly dual states being activated, then the elements of $\sqrt{\mathbb{P}(\neg S)}$ represent the probability of inactivation/deactivation for said states. A rough explanation as to why the distinction is not made in conventional logic would be that when $N = 1 - Y = \neg Y$, the equivalence $\mathbb{P}(\neg S) \equiv \mathbb{P}(\overline{S})$ comes about, *i.e.* the distinction cannot be made. The properties of equations (2.45) and (2.37) should not be confused with the one found in equation (2.49) as the former consider S as a whole, which allows for consideration of possibility X, while the latter is only concerned with its elements, which rightly assumes $X \in \neg S$ but does not allow for a proper separation of X from $\neg S$.

It may be easier to understand the complications of using $\mathbb{P}(\neg S)$ by comparing its inverse $\mathbb{P}(\neg\overline{S})$ with $\mathbb{P}(S)$. An important aspect that must be pointed out, is that the real and imaginary components of $\mathbb{P}(\neg\overline{S})$ are disjointed. As the respective origins lay at $Y = 1$ and $N = 1$ respectively, it is implied that $\neg N$ on the real axis and $\neg Y$ on the imaginary axis are largely independent declarations despite Y and N being supposed duals. Likewise, $\mathbb{P}(X_h)$ and $\mathbb{P}(\neg X_h)$ are disjoint (see figure 2.1(b)).

Though equation (2.49) makes it more apparent that Y and N are disjointed, it distinctly represents what is not presented in the elements of S.[4] To draw a conclusion w.r.t. the negation and inverse of S, if N is not directly processed as the absence of Y, *i.e.* $N \subseteq X$ such that $N = \neg Y$, it is not permissible to assume that N is diametrically opposed to Y, *e.g.* when N is derived from an alternative source; even then, $N = X$ cannot be assumed

[4] A more general method of implementation for $\mathbb{P}(\neg S)$ will be introduced later.

as this only occurs under ideal circumstances. Another remark that can be made to help understand how this integrates with set theory is that sets are primarily concerned with what does and does not exist and are not concerned about their placement much like $\mathbb{P}(Y)$, $\mathbb{P}(N)$, $||\mathbb{P}(Y \oplus N)||$, and $\mathbb{P}(Y \cap N)$ which are strictly positive and real [16]. Though values of said sets can be considered as unstable, there are applicable methods for establishing upper and lower bounds. For the probability of outcomes X, there is $||\mathbb{P}(X)||$ which can be acquired with some manipulation of equation (2.37). The values of $||\mathbb{P}(X)||$ can also be considered a valid set which contains what is lacking from set S and/or overly covered by Y and N. Lastly, it should not be forgotten that the 1 and $1 + \hat{\imath}1$ in the negations, as well as the 1 in equations (2.45) and (2.37) are representative of the complete universe of discourse Z.

Operations: Product and Summation

From equation (2.42) and from a reinterpretation of the traditional method, there are two likely proposals for the product:

$$\mathbb{P}(B \cap C) = 2 \times \sqrt{\mathbb{P}(B)} \times \sqrt{\mathbb{P}(C)} \tag{2.53}$$

and

$$\mathbb{P}(B \cap C) = \sqrt{\mathbb{P}(B)} \times \sqrt{\mathbb{P}(C)}. \tag{2.54}$$

As is, the second method is more desirable since this would imply that, for a product of a set with itself, $\mathbb{P}(A \cap A) = \mathbb{P}(A)$ while the first method implies $\mathbb{P}(A \cap A) = 2 \times \mathbb{P}(A)$, however, there are several other aspects to consider before making a conclusion, *e.g.* re-scaling and clipping of exaggerations. Two other operations that can be built off of these two product equations are XOR and OR:

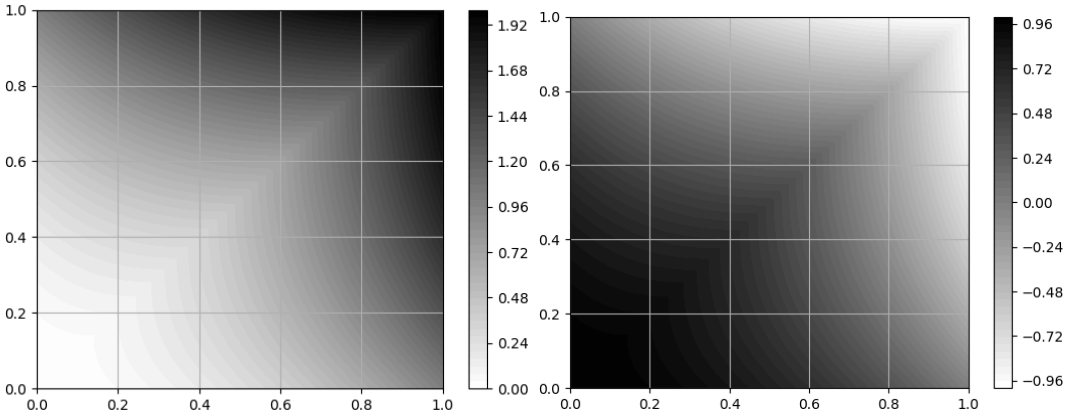
$$\mathbb{P}(B \oplus C) = \mathbb{P}(B) + \mathbb{P}(C) - 2 \times \sqrt{\mathbb{P}(B)} \times \sqrt{\mathbb{P}(C)} \tag{2.55}$$

and

$$\mathbb{P}(B \cup C) = \mathbb{P}(B) + \mathbb{P}(C) - \sqrt{\mathbb{P}(B)} \times \sqrt{\mathbb{P}(C)}. \tag{2.56}$$

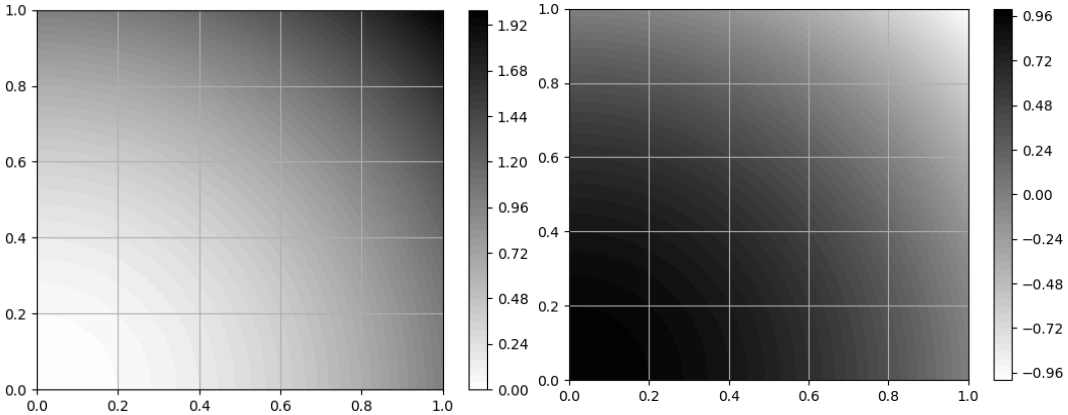
These equations have some similarity to the Law of Cosines equation:

$$a^2 = b^2 + c^2 - 2 \times b \times c \times cos(\theta_{bac}), \tag{2.57}$$



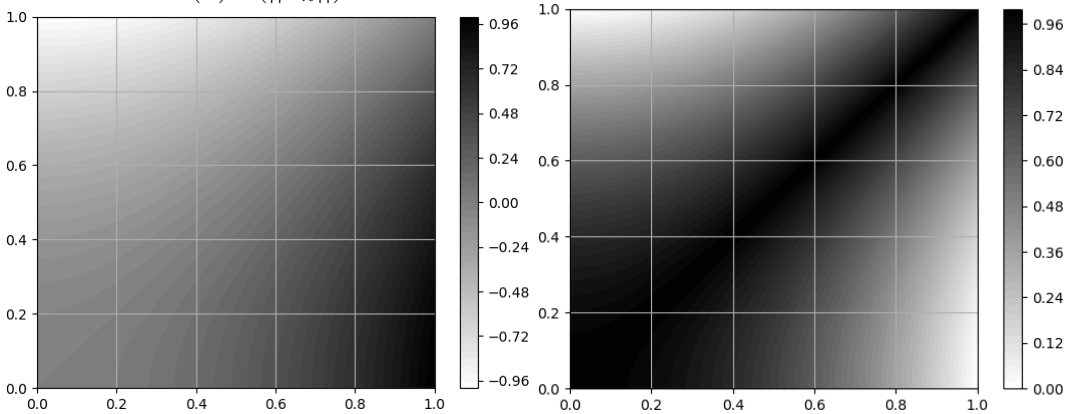
(a) $\mathbb{P}(||S||)$

(d) $\mathbb{P}(\neg||S||)$



(b) $\mathbb{P}(||S_h||)$

(e) $\mathbb{P}(X_h)$



(c) $\mathbb{P}(Y \oplus N)$

(f) $\mathbb{P}(\neg||Y \oplus N||)$

Figure 2.2: Where the y-axis is N and the x-axis is Y. The corresponding plots for $\mathbb{P}(\neg S)$ can be demonstrated by rotating about the line formed by $N = 1 - Y$ and the plots for the inverse can be found by rotating about the line formed by $N = Y$.

when $a \equiv \mathbb{L}(A)$, $b \equiv \mathbb{L}(B)$, and $c \equiv \mathbb{L}(C)$. $cos(\theta_{bac})$ is of interest because it is the angle between the logical hyperplanes for which B and C lay on respectively. By manipulating θ_{bac} subject to $0 \leq \theta_{bac} \leq \pi/2$, it is possible to get $\mathbb{P}(B \oplus C)$ at $\theta_{bac} = 0$; $\mathbb{P}(B \cup C)$ at $\theta_{bac} = \pi/3$ for non-mutually exclusive sets of B and C; $\mathbb{P}(B \cup C)$ at $\theta_{bac} = \pi/2$ for mutually exclusive sets of B and C; and every fine combination that lays between each of these. Unfortunately, this only applies under the assumption that all probability values are positive and real or at least lay on a single axis and not a plane. To better encompass the expression, it would require extending the Law of Cosines to include an arbitrary degree of flexibility for the angles between input planes and their relative orientations to the output.

Before going further, it may be useful to address whether the results should be treated as an aggregation of its input sets or as a directly dependent but separate event. For equation (2.53), it is likely more appropriate to apply the latter while equation (2.54) would apply the former, allowing the use of:

$$\mathbb{P}(A|B \cap C) = 2 \times \sqrt{\mathbb{P}(B)} \times \sqrt{\mathbb{P}(C)} \times cos(\theta_{bac}), \tag{2.58}$$

and since $(A \cap B) = (A|B) \times B$ for A dependent on B, it can be implied that:

$$\begin{aligned} \mathbb{P}(A \cap B \cap C) = & \quad \sqrt{\mathbb{P}(A|B \cap C)}\sqrt{\mathbb{P}(B \cap C)} \\ = & \quad \sqrt{2 \times cos(\theta_{bac})} \times \sqrt{\mathbb{P}(B)} \times \sqrt{\mathbb{P}(C)}. \end{aligned} \tag{2.59}$$

This further implies that $2 \times cos(\theta_{bac})$, given the Law of Cosines, determines the degree of dependence of A on $B \cap C$, even if $A \equiv B \cap C$, *i.e.* it can be claimed that: $A = B \cap C$ if $\theta_{bac} = \pi/3$; A is exclusive from $B \cap C$ if $\theta_{bac} = \pi/2$; and $A \propto B \cap C$ otherwise. Nitpicking about B and C being potentially dependent, the case of $B \propto C$ can be considered:

$$\begin{aligned} \mathbb{P}(B \cap C) = & \quad \sqrt{\mathbb{P}(B|C)} \times \sqrt{\mathbb{P}(C)} \\ = & \quad \sqrt{2 \times cos(\theta_{bc}) \times \mathbb{P}(C)} \times \sqrt{\mathbb{P}(C)} \\ = & \quad \sqrt{2 \times cos(\theta_{bc})} \times \mathbb{P}(C), \end{aligned} \tag{2.60}$$

which implies that the probability of $B \cap C$ is a scaled up/down value of the probability of C. More explicitly, the relation between $A|B$ and $(A \cap B)$, for

A that is dependent on B, can be given as:

$$\begin{aligned}\mathbb{P}(A|B) = &\quad 2 \times cos(\theta_{bab}) \times \mathbb{P}(B) \\ = &\quad \sqrt{2 \times cos(\theta_{ab})} \times \mathbb{P}(A \cap B). \end{aligned} \tag{2.61}$$

These relations imply that $2 \times cos(\theta_{bab}) = \mathbb{P}(A|B)/\mathbb{P}(B) \equiv \mathbb{P}(A \cap B)$ and $\sqrt{2 \times cos(\theta_{ab})} \equiv 1/\mathbb{P}(B)$ which further implies $\sqrt{2} \times cos(\theta_{bab})/\sqrt{cos(\theta_{ab})} \equiv \mathbb{P}(A|B)$. They are only equivalent and not equal because $2 \times cos(\theta)$ is unitless while the probabilities are not necessarily so. Carrying this over to our probabilistic sum, we can claim the general expression is based on conditional statements:

$$\begin{aligned}\mathbb{P}(A|B \bigcirc C) = &\quad \mathbb{P}(A|B) + \mathbb{P}(A|C) - 2 \times cos(\theta_{bac}) \times \mathbb{P}(A|B \cap C) \\ = &\quad 2 \times cos(\theta_{ab}) \times \mathbb{P}(B) + 2 \times cos(\theta_{ac}) \times \mathbb{P}(C) \\ &\quad -2 \times cos(\theta_{bac}) \times \sqrt{\mathbb{P}(B)} \times \sqrt{\mathbb{P}(C)}. \end{aligned} \tag{2.62}$$

In this form, the angles can be manipulated to form an assortment of logical expressions as well as there intermediates. This equation is more favorable due to A not being entirely dependent on one component or the other and because it is more compatible with the expression for the XOR operation. Making the replacement of $2 \times cos(\theta)$ in equation (2.62) with the learned weights of our learning algorithm only further enforces the utility of this interpretation.

Apply this new equation to the special case of $\mathbb{P}(B) = 1$, $\sqrt{2 \times cos(\theta_{1ac})} \leq 0$, $\sqrt{2 \times cos(\theta_{ac})} \leq 0$, a range of $\mathbb{P}(A|1 \bigcirc C)$ where A is an inverse form of C become possible. In short, the Law of Cosines with an orientation of inputs w.r.t. the output, though with different methods of representing the angles, can be considered equivalent to a generalized logical operator which can include several core logical operations — namely AND, OR, and INV. If this is acceptable, it is possible to analyze the complex valued weights of the learning algorithm as the scaling components produced by the orientations of each hyperplane relative to the output space and each other.

Calculating Expected Values

For the topic of expected values, the claim that they can be expressed by:

$$\begin{aligned}\mathbb{E}\{A|B\} = & \quad \mathbb{V}(A|B) \times \mathbb{P}(A|B) \\ = & \quad 2 \times \cos(\theta_{bab}) \times \mathbb{V}(A|B) \times \mathbb{P}(B) \\ = & \quad \sqrt{2 \times \cos(\theta_{ab})} \times \mathbb{V}(A|B) \times \mathbb{P}(A \cap B), \end{aligned} \tag{2.63}$$

will be proposed. In this form $\mathbb{V}(A|B)$ is more generalized than its previous usage, *i.e.* it includes logical values. If the expected value associated with equation (2.62) is the sum of its products, then:

$$\begin{aligned}\mathbb{E}\{A|(B \bigcirc C)\} = & \quad \mathbb{E}\{A|B\} + \mathbb{E}\{A|C\} - \mathbb{E}\{A|B \cap C\} \\ = & \quad \mathbb{V}(A|B) \times \mathbb{P}(A|B) + \mathbb{V}(A|C) \times \mathbb{P}(A|C) \\ & \quad -\mathbb{V}(A|B \cap C) \times \mathbb{P}(A|B \cap C), \end{aligned} \tag{2.64}$$

which implies the expected value A can depend on the respective input set, *i.e.* a sum of its parts. It should be noted that the effective units of $\mathbb{E}\{\}$ and $\mathbb{V}()$ are distinctly different because $\mathbb{E}\{\}$ includes the assumed units of $\mathbb{P}()$. The simplification, given $A = B \bigcirc C$ and the relation between logical values and probability values, as well as equations (2.59) and (2.60):

$$\begin{aligned}\mathbb{V}(A) = & \quad \mathbb{V}(A \cap A) = \sqrt{\mathbb{V}(A|A) \times \mathbb{V}(A)} \\ = & \quad \mathbb{V}(A|A) = \mathbb{V}(A|B \bigcirc C), \end{aligned} \tag{2.65}$$

for which it is assumed $\theta_a = \pi/3$. An expected value calculation that depends on but is strictly separate from its inputs can be given as:

$$\mathbb{E}\{A|(B \bigcirc C)\} = \mathbb{E}\{A\} = \mathbb{V}(A) \times \mathbb{P}(A). \tag{2.66}$$

To distinguish between $\mathbb{E}\{A|(B \bigcirc C)\}$ from the one from equation (2.64) and this shorter expression, the former accounts for relations between B and C according to the operation $\bigcirc$ while the latter's value is independent of this upstream aspect even if it is represented by A.

Operations: Reorienting, Re-scaling, and Clipping

Reorienting has to do with cases were logical values are partially negative on the their respective axis. Though encountering negative values can be prevented with appropriate bounding functions, it would be best to at least propose methods of handling them. Firstly, due to the nature of how phase angles are affected by operations of the form x^n, going from $\mathbb{L}(A)$ to $\mathbb{P}(A)$ for negative real or imaginary values would yield overlapping or contradictory results. Therefore, before squaring logical values it is necessary to convert them to their equivalent positive values. This may be done with:

$$2\mathbb{L}(A) = \mathbb{L}(A) + \hat{\imath}\mathbb{L}(-A)^*, \tag{2.67}$$

which adds the magnitude of negative denials and affirmations to the real and imaginary axis respectively. Following this conversion, the negative values can be clipped with:

$$\sqrt{\mathbb{P}(A)} \equiv \mathbb{R}\{2\sqrt{\mathbb{P}(A)}\} \times u(\mathbb{R}\{2\sqrt{\mathbb{P}(A)}\}) + \hat{\imath}\mathbb{I}\{2\sqrt{\mathbb{P}(A)}\} \times u(\mathbb{I}\{2\sqrt{\mathbb{P}(A)}\}). \tag{2.68}$$

This method cannot be performed in the probability space due to the overlapping regions for which two different phase angles, upon being doubled, give the same result.

To ensure $\mathbb{P}(X_h) \geq 0$, the total probability should not be allowed to exceed 1^2. One option is to immediately re-scale the values using euclidean normalization whenever they exceed the unit circle boundary, *i.e.* ensuring $0 \leq ||\sqrt{\mathbb{P}(A)}|| \leq 1$, which preserves the ratios of the real and imaginary components; however, this may render the less dominant statement mute simply because the counter-statement was excessively large. The alternative is to forgo preserving the phase angle and clip exaggerated values to 1 before performing normalization. Clipping, followed by re-scaling, has an interesting property whereby, the more something is exaggerated for or against a statement to which there is a conflicting alternative, the greater the distortion of phase angle away from said exaggeration, *e.g.* when something is too good to be true, doubts regarding credibility gain more influence. These three op-

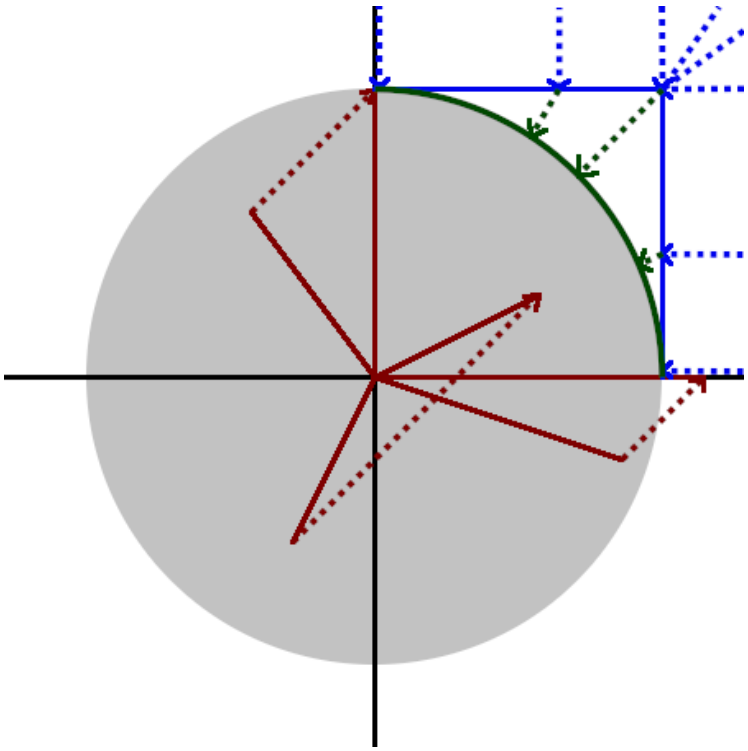


Figure 2.3: Following the operations: logical components can first be reoriented such that they lie in the complex plane; then values exceeding 1 can be clipped to remove exaggerations, causing skepticism of exaggerated values; finally, the ratio is preserved while exaggerations of the total logic space S are projected onto the unit circle. It should be noted that no matter how large the exaggeration is, it will not cause the conclusion to cross the axis for inversion, *i.e.* $1 + \hat{\imath}1$.

erations can also be used in order of reorienting, clipping, and re-scaling to maintain the desired range of values within the probability space.

2.3 Ions in the Neuron Cell

By focusing on the four major ions, *i.e.* Calcium, Sodium, Potassium, and Chloride, and getting a sufficient understanding of their interactions with the neuron cell membrane (see tables 2.1 to 2.3), it is possible to differentiate their effects and roles in neuron signal transmission and processing.[5] It quickly be-

[5]Characteristics that were researched include resting concentrations, method of transport across the membrane, influence on membrane voltage, conditions for associated activation/inhibition, order of movement during cell activation, and refractory period duration. Factors that affect cellular development/learning are not covered as the application of this

Ion	**In-Cell** $[mMol]$	**Out-Cell** $[mMol]$	**Equilibrium Potential** $[mV]$	**Permeability** [%]
Ca^{2+}	0.0001	1.2	+125	≈0*
Na^{+}	18	145	+56	0.04
K^{+}	135	3	-102	1
Cl^{-}	7**	120	-76	0.45

* The numerical value could not be found, however, it can be expected to be extremely small.
** This value tends to be higher in immature cells, typically 25-40 $[mMol]$ [30].

Table 2.1: Ion properties at equilibrium [15], [21], [26]

$In \rightarrow Out$	Ratio	$In \leftarrow Out$	$:ATP$*
Ca^{2+}	1 : 3	Na^{+}	:1
Ca^{2+}	1 : 0	n/a	:1
Na^{+}	3 : 2	K^{+}	:1
H^{+}	1 : 1	Na^{+}	:0
HCO_3^{-}	1 : 1	Cl^{-}	:0
Cl^{-}	1 : 0	n/a	:1

* ATP is an energy source from inside the cell required for active transport.

Table 2.2: A sample of ion exchangers and anti-porters [7], [30], [32]

comes apparent that Ca^{2+}, and Na^{+} play a major role in representing immediate information, while the activities of K^{+} and Cl^{-} persist much longer after activation and focus more on the final clean-up and stabilization after periods of high cellular activity, suggesting a role with more historical significance w.r.t. impulse activity.

Ca^{2+} is only active for extremely short periods of time and largely responsible for the release of neurotransmitters [34]. It can be seen from table 2.1 that, at equilibrium, the concentration of Calcium inside the cell is minuscule in comparison to the concentration outside, and from table 2.3, the gating

information will focus on signal processing w.r.t. the cell membrane and not the cell as a whole.

Target	Direction	Gating	Inhibitors
Cl^-	$In \rightarrow Out$	Pressure	
Cl^-	$In \leftarrow Out$	Voltage*	
Cl^-	$In \leftrightarrow^{\dagger} Out$	GABA, Membrane Tension	
K^+ ***	$In \rightarrow Out$	Medium/Low-Voltage, Ca^{2+}	
Na^+	$In \leftarrow Out$	Low-Voltage	Inactivation
Ca^{2+}	$In \leftarrow Out$	High/Medium/Low-Voltage	Mg^{2+}

* Acts to rectify during hyper-polarization.

† Direction is primarily regulated by the chloride concentration gradient, Osmotic Pressure, and relative concentrations of chloride transporters and channels [30].

***These tend to have a delayed and very slow transition between open and closed states and serve to re-polarize the cell.

Table 2.3: A sample of regulated ion channels [3], [9], [12], [13], [24], [25], [30].

method typically used functions only in the presence of a sufficient positive voltage stimulation. The nature of these two factors suggest that neurotransmitter release occurs when the Ca^{2+} enter the cell which only occurs when a positive voltage is induced. There are three voltage ranges that specifically induce gate reaction: low (T-Type), medium (R-Type), and high (N-, P/Q-, and L-type) voltage stimuli. Given sufficient membrane permeability to Ca^{2+}, the induced voltage can result in activation, however, this does not necessarily mean complete membrane activation must/will occur solely because of Ca^{2+} [9], [26]. Another point to keep in mind is that the presence of Ca^{2+} inside the cell is only for a relatively short amount of time as the Mg^{2+} in the cell act as inhibitors, preventing more from entering while the ion pumps work to quickly remove the inter-cellular Ca^{2+} using ATP and available Na^+. Though there are a number of other interesting functions that Ca^{2+} is responsible for — such as influencing gene expression and long term potentiation — the segment this work is primarily interested in is the fact that, given the membranes reactivity to voltage, Ca^{2+} is responsible for neurotransmitter release and is utilized during the rising edge of a positive voltage spike. [25].

The Sodium cation, Na^+, is one of two ions — the other being K^+ — that are commonly attributed to neural activity. Similarly to Ca^{2+}, the majority of

Set	Ratio	Direction
Na^+:$Gluclose$	2 : 1	$In \leftarrow Out$
Na^+:P_i*	3 : 1	$In \leftarrow Out$
Na^+:I^-	2 : 1	$In \leftarrow Out$
Na^+:Cl^-:$GABA$	2 : 1 : 1	$In \leftarrow Out$
Na^+:Cl^-	1 : 1	$In \leftarrow Out$
Na^+:K^+:Cl^-	1 : 1 : 2	$In \leftarrow Out$
Na^+:HCO_3^-	1 : 1	$In \leftarrow Out$
Na^+:HCO_3^-	1 : 2	$In \leftarrow Out$
Na^+:HCO_3^-	1 : 3	$In \rightarrow Out$
K^+:Cl^-	1 : 1	$In \rightarrow Out$
H^+:Pep**	1 : 1	$In \leftarrow Out$
H^+:$R - COO^-$***	1 : 1	$In \rightarrow Out$

* P_i refers to inorganic Phosphate.
** Pep refers to Di/Tripeptides.
*** $R - COO^-$ refers to Monocarboxylates.

Table 2.4: A sample of ion symporters and co-transporters [30], [32].

the Na^+ concentration is outside the cell at resting equilibrium and is typically gated by a low voltage threshold. The primary difference in the gating mechanism is that the gates are closed and enter an inactivation phase after a short period of time and are not inhibited by other ions or molecules during normal operation [1]. From the information in tables 2.2 and 2.4, it can be gathered that Na^+ is largely responsible for maintaining the condition of the cell as it is used to remove Ca^{2+} that causes neurotransmitter release and HCO_3^-, which is a byproduct of regular cellular activity. It is also responsible for carrying in neurotransmitters once released for signal emission such as $GABA$ and molecular compounds and ions necessary for maintaining regular cell health [30], [32]. Based on how involved Na^+ is w.r.t. cell activity, this information also suggests that a sufficiently high cellular metabolism can also cause self activation. It is also worth noting that most of its applications involve moving from outside to inside the cell, therefore the opening of the voltage-gated Na^+ channel to balance the ion concentrations can cause a temporary interruption and, if active for prolonged periods of time, may even cause the cell to become exhausted. The last point to mention is that Na^+ typically has the greatest influence on the membrane voltage during cell activation and is the primary signal carrying medium [24].

The potassium ion, K^+, is the second ion that plays a notable role in neuron activation. K^+ is most active during the depolarization phase which returns the cell back to its resting potential. In contrast to Sodium and Calcium, the resting membrane concentrations for K^+ are such that the concentration inside the cell is much higher than the concentration outside the cell, and during activation, they flow out of the cell. Another notable difference is that the permeability of K^+ is much larger than any other ion during the resting phase, allowing for relatively easy movement in and out of the cell in the presence of sufficient osmotic pressure. K^+ gates also have an interesting characteristic in that they tend to have some amount of delay before opening to allow K^+ out of the cell [1]. Given that the primary task of K^+ is removing Na^+ and, to a lesser degree Cl^- from inside the cell after polarization, it can be understood that K^+ plays a specialized role in maintaining the concentration

gradients of other ions that are critical for cellular operation, *i.e.* primarily Cl^- and Na^+ [1], [30].

The Chloride ion — though not as influential as the other ions w.r.t. neuron spiking — is comparable to Ca^{2+} in that it has a significant influence on cellular development. Cl^- is primarily responsible for maintaining PH and limiting osmotic pressure, but also influences internal structures such as micro-tubules — typically by binding with free Mg^{2+}. The role of Cl^- also carries some similar characteristics to the roles of other ions, such as: removing cellular waste; maintaining the overall concentration gradient as well as the specific gradients of Na^+ and K^+; and drawing in neurotransmitters. From table 2.3, it can be gathered that Cl^- gating is mostly reactive. Additionally, Cl^- is the only ion whose concentration gradient changes as the cell matures, starting as an excitatory ion and shifting to an inhibitory ion. Lastly, relative to the other ions, Cl^- has the longest period of activity after a neuron has fired [30].

Based on the information gathered, there are several aspects of interest. The order of initial dominant activity is roughly Ca^{2+}, Na^+, Cl^-, and K^+, while the order of recovery is roughly Ca^{2+}, Na^+, K^+, and Cl^-. The activity threshold and duration tend to be gradual and short for Ca^{2+}; crisp and short for Na^+; delayed, crisp and moderate for K^+; and conditionally crisp and long for Cl^-. Given Na^+ is strongly tied to cellular activity, it can be assumed that it is related to immediate current information while K^+ and Cl^-, which are responsible for cleanup have some ties to short term historical record keeping. In this way it can also be expected that Ca^{2+} is related to predictions of the immediate future.

2.4 The Generalized Sigmoid Function

The sigmoid function comes in many forms, with each being built with some specific purpose in mind. Similarly, the generalized sigmoid function proposed here is formed with the specific purpose of offering the flexibility to optimize its curvature in a continuous fashion alongside the learning algorithm's parameters. There are some circumstances where a piece-wise linear sigmoid

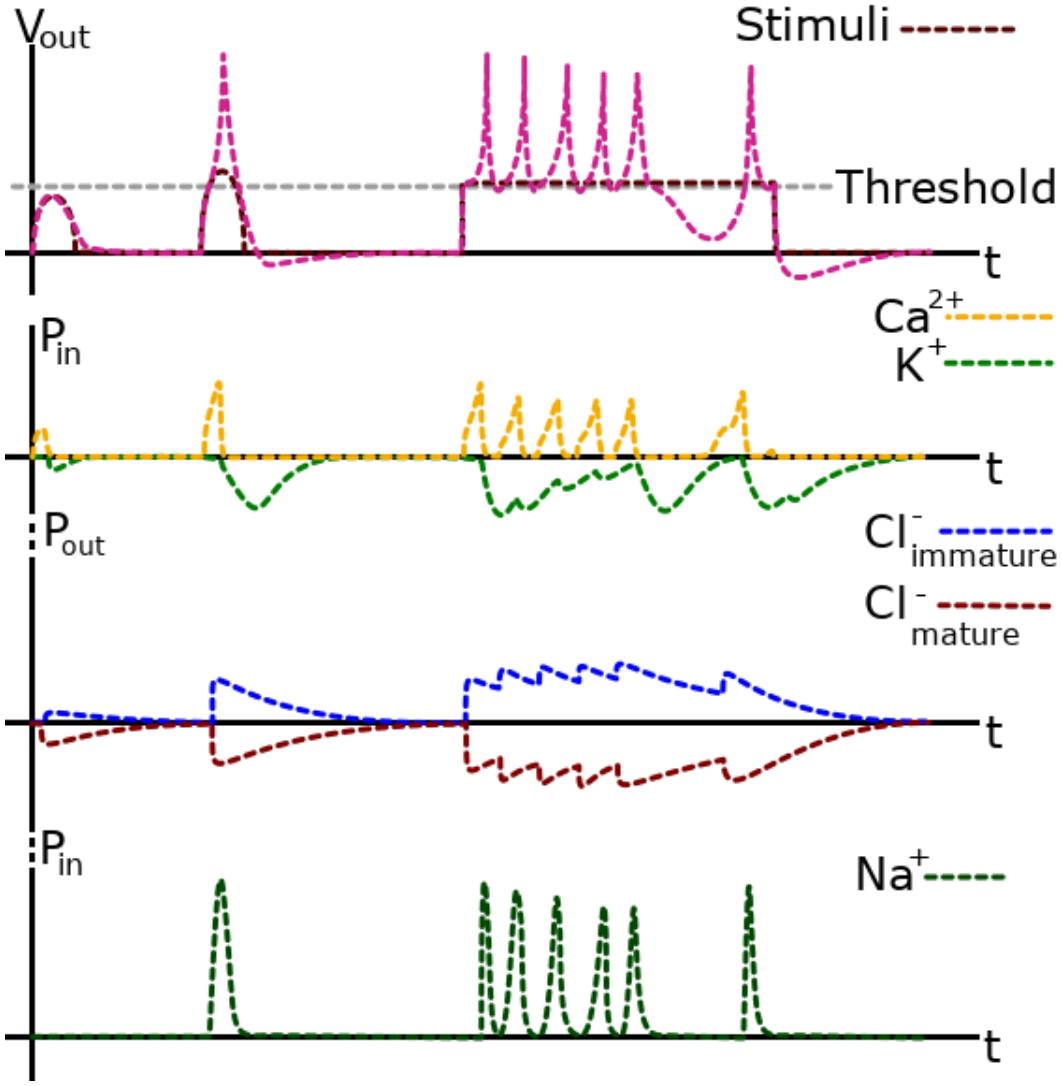


Figure 2.4: Approximate plots of the expected signal medium permeabilities of interest given an input stimuli.

is preferred over the exponential equivalent, but there are also many other forms that have the potential to better suite the problem. Flexibility allows for a progressive optimization of the threshold function itself. The proposed function is defined as:

$$sig^{-}(x, a, b, c) = \frac{|x|^{a \times b}}{(|x|^{a \times c} + 1)^{b/c}} \tag{2.69}$$

for the unsigned case and:

$$sig^{+}(x, a, b, c) = \frac{x}{|x|} \times \frac{|x|^{a \times b}}{(|x|^{a \times c} + 1)^{b/c}} \tag{2.70}$$

for the signed case.[6] Starting from the two relatively simple signed forms, it is possible to see how these equations come about:

$$sig(x) = \frac{x}{\sqrt{x^2 + 1}} \tag{2.71}$$

and

$$sig(x) = \frac{x}{|x| + 1}. \tag{2.72}$$

Focusing on the denominator first, they can be likened to the Euclidean and Hamming distance measures respectively. Given the similar structure, we can write a more generalized function that, under specific conditions, can achieve one, the other, something in-between, or something different altogether:

$$sig(x, c) = \frac{x}{\sqrt[c]{|x|^c + 1}}. \tag{2.73}$$

Before moving on, consideration of all possible values of $c \in \mathbb{R}$ including when $c \to \pm\infty$ should be taken. For $c \to \infty$, we will rely on two equivalent forms by restructuring the denominator as needed:

$$\sqrt[c]{|x|^c + 1} = |x| \times \sqrt[c]{1 + \left|\frac{1}{x}\right|^c}. \tag{2.74}$$

given that, $|x|^c > 1$ for $c > 1$ and $x > 1$ we can expect (2.74) to reduce to:

$$\lim_{c \to \infty} \sqrt[c]{|x|^c + 1} = 1 \tag{2.75}$$

[6]These two functions allow us to cover requirements for both odd and even forms respectively while maintaining equivalence.

for $0 < x \leq 1$ and:

$$\lim_{c \to \infty} \sqrt[c]{|x|^c + 1} = \infty \tag{2.76}$$

for $x \geq 1$. These limits apply equally to $c \to -\infty$ as $x^{-c} = (x^{-1})^c$. The last extreme value is when $c = 0$ as $|x|^0\big|_{x \neq 0} = 1$ but 0^0 is undefined. Based on the fact that the largest finite value follows the aforementioned relation, we also expect $\infty^0 = 1$. The known limits of the denominator approach ∞ which will result in the sigmoid function going to zero. Since the function is continuous in all other cases, an expectation of $0^0 = 1$ will be held to for now and thus:

$$\lim_{c \to 0} \sqrt[c]{|x|^c + 1} = \infty. \tag{2.77}$$

The second value considered is a as a power of x which, if applied to equation (2.73) under normal circumstances, would result in:

$$sig(x^a, c) = \frac{x^a}{\sqrt[c]{|x^a|^c + 1}}; \tag{2.78}$$

however, the value of a will affect both the magnitude and sign of the result, for which the signed function will transition between a signed and unsigned function depending on the value of a. To preserve the sign without affecting the influence on magnitude, it is preferable to reapply it as an internal value:

$$sig(x, a, c) = \frac{x \times |x|^{a-1}}{\sqrt[c]{|x|^{a \times c} + 1}}. \tag{2.79}$$

For the limit case where $a \to \infty$ and assuming a finite positive c, we get three solutions depending on the magnitude of x, *i.e.*:

$$\lim_{a \to \infty} \frac{x \times |x|^{a-1}}{\sqrt[c]{|x|^{a \times c} + 1}} = 0 \tag{2.80}$$

for $|x| < 1$,

$$\lim_{a \to \infty} \frac{x \times |x|^{a-1}}{\sqrt[c]{|x|^{a \times c} + 1}} = \frac{x}{\sqrt[c]{2} \times |x|} \tag{2.81}$$

for $|x| = 1$, and

$$\lim_{a \to \infty} \frac{x \times |x|^{a-1}}{\sqrt[c]{|x|^{a \times c} + 1}} = \frac{x}{|x|} \tag{2.82}$$

for $|x| > 1$. If $a < 0$, we can rearrange equation (2.79) to:

$$sig(x, a, c) = \frac{x}{|x| \times \sqrt[c]{1 + |x|^{-a \times c}}} \tag{2.83}$$

for which the limits become:

$$\lim_{a\to-\infty} \frac{x}{|x| \times \sqrt[c]{1+|x|^{-a\times c}}} = 0 \tag{2.84}$$

for $|x| > 1$,

$$\lim_{a\to-\infty} \frac{x}{|x| \times \sqrt[c]{1+|x|^{-a\times c}}} = \frac{x}{\sqrt[c]{2} \times |x|} \tag{2.85}$$

for $|x| = 1$, and

$$\lim_{a\to-\infty} \frac{x}{|x| \times \sqrt[c]{1+|x|^{-a\times c}}} = \frac{x}{|x|} \tag{2.86}$$

for $|x| < 1$. Should $a \to 0$, we get the limit:

$$\lim_{a\to 0} \frac{x}{|x| \times \sqrt[c]{1+1^{c}}} = \frac{x}{\sqrt[c]{2} \times |x|} \tag{2.87}$$

which applies for all values of x. As can be seen by these limits, the phase of the input is preserved for instances that do not go to zero.

To complete the generalized sigmoid functions, an exponential will also be applied to the output such that it does not interfere with the phase, *i.e.* equation (2.70). Should a and c both be finite and positive, the limits as $b \to \infty$ can be obtained by:

$$\lim_{b\to\infty} \frac{x}{|x|} \times |sig(x,a,c)|^{b} = 0 \tag{2.88}$$

for $|sig(x,a,c)| < 1$,

$$\lim_{b\to\infty} \frac{x}{|x|} \times |sig(x,a,c)|^{b} = \frac{x}{|x|} \tag{2.89}$$

for $|sig(x,a,c)| = 1$, and

$$\lim_{b\to\infty} \frac{x}{|x|} \times |sig(x,a,c)|^{b} = \infty \tag{2.90}$$

for $|sig(x,a,c)| > 1$. With these limits known, the solution for $b \to -\infty$ becomes trivial. Should $b \to 0$, the limit is:

$$\lim_{b\to\infty} \frac{x}{|x|} \times |sig(x,a,c)|^{b} = \frac{x}{|x|} \tag{2.91}$$

for all values of x. These limits allow us to understand the result given a single parameter approaches a limit, however, it is also necessary to grasp how they

interact with each other at their limits. For this, an order of operation giving priority to resolving the parameters closest to the input first and work to the output will be used, thus for $|x| < 1$ we get:

$$\lim_{\substack{a\to\infty\\ b\to\infty\\ c\to\infty}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to\infty\\ c\to\infty}} \frac{x}{|x|}\times\frac{0^b}{(0^c+1)^{b/c}} = \lim_{b\to\infty}\frac{x}{|x|}\times\frac{0^b}{(1)^b} = 0, \quad (2.92)$$

$$\lim_{\substack{a\to\infty\\ b\to\infty\\ c\to 0}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to\infty\\ c\to 0}} \frac{x}{|x|}\times\frac{0^b}{(0^c+1)^{b/c}} = \lim_{b\to\infty}\frac{x}{|x|}\times\frac{0^b}{(\infty)^b} = 0, \quad (2.93)$$

$$\lim_{\substack{a\to\infty\\ b\to\infty\\ c\to-\infty}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to\infty\\ c\to\infty}} \frac{x}{|x|}\times 0^b\times(\infty^c+1)^{b/c}$$
$$= \lim_{\substack{b\to\infty\\ c\to\infty}} \frac{x}{|x|}\times(1+\frac{1}{\infty^c})^{b/c} = \lim_{b\to\infty}\frac{x}{|x|}\times(1)^b = \frac{x}{|x|}, \quad (2.94)$$

$$\lim_{\substack{a\to\infty\\ b\to 0\\ c\to\infty}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to 0\\ c\to\infty}} \frac{x}{|x|}\times\frac{0^b}{(0^c+1)^{b/c}} = \lim_{b\to 0}\frac{x}{|x|}\times\frac{0^b}{(1)^b} = \frac{x}{|x|}, \quad (2.95)$$

$$\lim_{\substack{a\to\infty\\ b\to 0\\ c\to 0}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to 0\\ c\to 0}} \frac{x}{|x|}\times\frac{0^b}{(0^c+1)^{b/c}} = \lim_{b\to 0}\frac{x}{|x|}\times\frac{0^b}{(\infty)^b} = \frac{x}{|x|}, \quad (2.96)$$

$$\lim_{\substack{a\to\infty\\ b\to 0\\ c\to-\infty}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to 0\\ c\to\infty}} \frac{x}{|x|}\times 0^b\times(\infty^c+1)^{b/c}$$
$$= \lim_{\substack{b\to 0\\ c\to\infty}} \frac{x}{|x|}\times(1+\frac{1}{\infty^c})^{b/c} = \lim_{b\to 0}\frac{x}{|x|}\times(1)^b = \frac{x}{|x|}, \quad (2.97)$$

$$\lim_{\substack{a\to\infty\\ b\to-\infty\\ c\to\infty}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to-\infty\\ c\to\infty}} \frac{x}{|x|}\times\frac{0^b}{(0^c+1)^{b/c}} = \lim_{b\to-\infty}\frac{x}{|x|}\times\frac{0^b}{(1)^b} = \frac{x}{|x|}\times\infty, \quad (2.98)$$

$$\lim_{\substack{a\to\infty\\ b\to-\infty\\ c\to 0}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to-\infty\\ c\to 0}} \frac{x}{|x|}\times\frac{0^b}{(0^c+1)^{b/c}} = \lim_{b\to-\infty}\frac{x}{|x|}\times\frac{0^b}{(\infty)^b} = \frac{x}{|x|}\times\infty, \quad (2.99)$$

$$\lim_{\substack{a\to\infty\\ b\to-\infty\\ c\to-\infty}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to-\infty\\ c\to\infty}} \frac{x}{|x|}\times 0^b\times(\infty^c+1)^{b/c}$$
$$= \lim_{\substack{b\to-\infty\\ c\to\infty}} \frac{x}{|x|}\times(1+\frac{1}{\infty^c})^{b/c} = \lim_{b\to-\infty}\frac{x}{|x|}\times(1)^b = \frac{x}{|x|}, \quad (2.100)$$

$$\lim_{\substack{a\to 0\\ b\to\infty\\ c\to\infty}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to\infty\\ c\to\infty}} \frac{x}{|x|}\times\frac{1^b}{(1^c+1)^{b/c}} = \lim_{b\to\infty}\frac{x}{|x|}\times\frac{1^b}{(2)^b} = 0, \quad (2.101)$$

$$\lim_{\substack{a\to 0\\ b\to\infty\\ c\to 0}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to\infty\\ c\to 0}} \frac{x}{|x|}\times\frac{1^b}{(1^c+1)^{b/c}} = \lim_{b\to\infty}\frac{x}{|x|}\times\frac{1^b}{(\infty)^b} = 0, \quad (2.102)$$

$$\lim_{\substack{a\to 0\\ b\to\infty\\ c\to-\infty}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to\infty\\ c\to\infty}} \frac{x}{|x|}\times 1^b\times(1^c+1)^{b/c} = \lim_{b\to\infty}\frac{x}{|x|}\times(2)^b = \frac{x}{|x|}\times\infty, \quad (2.103)$$

$$\lim_{\substack{a\to 0\\ b\to 0\\ c\to\infty}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to 0\\ c\to\infty}} \frac{x}{|x|}\times\frac{1^b}{(1^c+1)^{b/c}} = \lim_{b\to 0}\frac{x}{|x|}\times\frac{1^b}{(2)^b} = \frac{x}{|x|}, \quad (2.104)$$

$$\lim_{\substack{a\to 0\\ b\to 0\\ c\to 0}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to 0\\ c\to 0}} \frac{x}{|x|}\times\frac{1^b}{(1^c+1)^{b/c}} = \lim_{b\to 0}\frac{x}{|x|}\times\frac{1^b}{(\infty)^b} = \frac{x}{|x|}, \quad (2.105)$$

$$\lim_{\substack{a\to 0\\ b\to 0\\ c\to-\infty}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to 0\\ c\to\infty}} \frac{x}{|x|}\times 1^b\times(1^c+1)^{b/c} = \lim_{b\to 0}\frac{x}{|x|}\times(2)^b = \frac{x}{|x|}, \quad (2.106)$$

$$\lim_{\substack{a\to 0\\ b\to-\infty\\ c\to\infty}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to-\infty\\ c\to\infty}} \frac{x}{|x|}\times\frac{1^b}{(1^c+1)^{b/c}} = \lim_{b\to-\infty}\frac{x}{|x|}\times\frac{1^b}{(2)^b} = \frac{x}{|x|}\times\infty, \quad (2.107)$$

$$\lim_{\substack{a\to 0\\ b\to-\infty\\ c\to 0}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to-\infty\\ c\to 0}} \frac{x}{|x|}\times\frac{1^b}{(1^c+1)^{b/c}} = \lim_{b\to-\infty}\frac{x}{|x|}\times\frac{1^b}{(\infty)^b} = \frac{x}{|x|}\times\infty, \quad (2.108)$$

$$\lim_{\substack{a\to 0\\ b\to-\infty\\ c\to-\infty}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to-\infty\\ c\to\infty}} \frac{x}{|x|}\times 1^b\times(1^c+1)^{b/c} = \lim_{b\to-\infty}\frac{x}{|x|}\times(2)^b = 0, \quad (2.109)$$

$$\lim_{\substack{a\to-\infty\\ b\to\infty\\ c\to\infty}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to\infty\\ c\to\infty}} \frac{x}{|x|}\times\frac{\infty^b}{(\infty^c+1)^{b/c}}$$
$$= \lim_{\substack{b\to\infty\\ c\to\infty}} \frac{x}{|x|}\times\frac{1}{(1+\frac{1}{\infty^c})^{b/c}} = \lim_{b\to\infty}\frac{x}{|x|}\times\frac{1}{(1)^b} = \frac{x}{|x|}, \quad (2.110)$$

$$\lim_{\substack{a\to-\infty\\ b\to\infty\\ c\to0}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to\infty\\ c\to0}} \frac{x}{|x|}\times\frac{\infty^b}{(\infty^c+1)^{b/c}}$$

$$= \lim_{\substack{b\to\infty\\ c\to0}} \frac{x}{|x|}\times\frac{1}{(1+\frac{1}{\infty^c})^{b/c}} = \lim_{b\to\infty}\frac{x}{|x|}\times\frac{1}{(\infty)^b} = 0, \quad (2.111)$$

$$\lim_{\substack{a\to-\infty\\ b\to\infty\\ c\to-\infty}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to\infty\\ c\to-\infty}} \frac{x}{|x|}\times\frac{\infty^b}{(\infty^c+1)^{b/c}}$$

$$= \lim_{\substack{b\to\infty\\ c\to\infty}} \frac{x}{|x|}\times(1+\infty^c)^{b/c} = \lim_{b\to\infty}\frac{x}{|x|}\times(1)^b = \frac{x}{|x|}, \quad (2.112)$$

$$\lim_{\substack{a\to-\infty\\ b\to0\\ c\to\infty}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to0\\ c\to\infty}} \frac{x}{|x|}\times\frac{\infty^b}{(\infty^c+1)^{b/c}}$$

$$= \lim_{\substack{b\to0\\ c\to\infty}} \frac{x}{|x|}\times\frac{1}{(1+\frac{1}{\infty^c})^{b/c}} = \lim_{b\to0}\frac{x}{|x|}\times\frac{1}{(1)^b} = \frac{x}{|x|}, \quad (2.113)$$

$$\lim_{\substack{a\to-\infty\\ b\to0\\ c\to0}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to0\\ c\to0}} \frac{x}{|x|}\times\frac{\infty^b}{(\infty^c+1)^{b/c}}$$

$$= \lim_{\substack{b\to0\\ c\to0}} \frac{x}{|x|}\times\frac{1}{(1+\frac{1}{\infty^c})^{b/c}} = \lim_{b\to0}\frac{x}{|x|}\times\frac{1}{(\infty)^b} = \frac{x}{|x|}, \quad (2.114)$$

$$\lim_{\substack{a\to-\infty\\ b\to0\\ c\to-\infty}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to0\\ c\to-\infty}} \frac{x}{|x|}\times\frac{\infty^b}{(\infty^c+1)^{b/c}}$$

$$= \lim_{\substack{b\to0\\ c\to\infty}} \frac{x}{|x|}\times(1+\infty^c)^{b/c} = \lim_{b\to0}\frac{x}{|x|}\times(1)^b = \frac{x}{|x|}, \quad (2.115)$$

$$\lim_{\substack{a\to-\infty\\ b\to-\infty\\ c\to\infty}} \frac{x\times|x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to-\infty\\ c\to\infty}} \frac{x}{|x|}\times\frac{\infty^b}{(\infty^c+1)^{b/c}}$$

$$= \lim_{\substack{b\to-\infty\\ c\to\infty}} \frac{x}{|x|}\times\frac{1}{(1+\frac{1}{\infty^c})^{b/c}} = \lim_{b\to-\infty}\frac{x}{|x|}\times\frac{1}{(1)^b} = \frac{x}{|x|}, \quad (2.116)$$

<table>
<tr><td>$a \to \infty$</td><td>$b \to -\infty$</td><td>$-\infty < b < 0$</td><td>$b \to 0$</td><td>$0 < b < \infty$</td><td>$b \to \infty$</td></tr>
<tr><td>$c \to -\infty$</td><td colspan="5" rowspan="2"></td></tr>
<tr><td>$-\infty < c < 0$</td></tr>
<tr><td>$c \to 0$</td><td colspan="2" rowspan="2">$\frac{x}{\lvert x \rvert} \times \infty$</td><td rowspan="2">$\frac{x}{\lvert x \rvert}$</td><td colspan="2" rowspan="2">0</td></tr>
<tr><td>$0 < c < \infty$</td></tr>
<tr><td>$c \to \infty$</td><td colspan="5"></td></tr>
</table>

Table 2.5: The sigmoid limits for $a \to \infty$ assuming $|x| < 1$ which includes the limit $|x| \to 0^{\pm}$. This is also the case for $a \to -\infty$ assuming $|x| > 1$ which includes the limit $|x| \to \infty$

$$\lim_{\substack{a\to-\infty \\ b\to-\infty \\ c\to0}} \frac{x \times |x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to-\infty \\ c\to0}} \frac{x}{|x|} \times \frac{\infty^b}{(\infty^c+1)^{b/c}}$$
$$= \lim_{\substack{b\to-\infty \\ c\to0}} \frac{x}{|x|} \times \frac{1}{(1+\frac{1}{\infty^c})^{b/c}} = \lim_{b\to-\infty} \frac{x}{|x|} \times \frac{1}{(\infty)^b} = \frac{x}{|x|} \times \infty, \quad (2.117)$$

$$\lim_{\substack{a\to-\infty \\ b\to-\infty \\ c\to-\infty}} \frac{x \times |x|^{a\times b-1}}{(|x|^{a\times c}+1)^{b/c}} = \lim_{\substack{b\to-\infty \\ c\to-\infty}} \frac{x}{|x|} \times \frac{\infty^b}{(\infty^c+1)^{b/c}}$$
$$= \lim_{\substack{b\to-\infty \\ c\to\infty}} \frac{x}{|x|} \times (1+\infty^c)^{b/c} = \lim_{b\to-\infty} \frac{x}{|x|} \times (1)^b = \frac{x}{|x|}, \quad (2.118)$$

For $x > 1$, it is equivalent to exploring $x < 1$ when changing $a \to -1 \times a$; likewise — for the purpose of continuity and symmetry — $a \to 0 \times a$ is equivalent to exploring $x^0 = 1$. Given these limits are w.r.t. equation (2.70), the equivalent for the unsigned equation (2.69) would have the replacement $|x/|x|| = |0/|0|| = 1$ in difference to the signed equation which assumes:

$$\left|\frac{x}{|x|}\right| = \begin{cases} 0 & \text{If } |x| = 0, \\ 1 & \text{Otherwise.} \end{cases} \quad (2.119)$$

The phase values are considered dominant within the equation, therefore, for the signed case $0/|0| \times \infty = 0$. If these limits are tabulated, we obtain what is shown in tables 2.5 to 2.9.

The generalized functions are much more powerful than the traditional forms because they can be properly and justifiably applied to vectorized inputs, *e.g.* complex numbers, without having to make special exceptions or

$0 < a < \infty$	$b \to -\infty$	$-\infty < b < 0$	$b \to 0$	$0 < b < \infty$	$b \to \infty$
$c \to -\infty$					
$-\infty < c < 0$	0	$\frac{x}{\lvert x\rvert} \times \frac{1}{(1+\lvert x\rvert^{-a\times c})^{b/c}}$		$\frac{x}{\lvert x\rvert} \times (1+\lvert x\rvert^{-a\times c})^{-b/c}$	$\frac{x}{\lvert x\rvert} \times \infty$
$c \to 0$		$\frac{x}{\lvert x\rvert} \times \infty$	$\frac{x}{\lvert x\rvert}$	0	
$0 < c < \infty$		$\frac{x}{\lvert x\rvert} \times \left(1+\left\lvert\frac{1}{x}\right\rvert^{a\times c}\right)^{-b/c}$		$\frac{x}{\lvert x\rvert} \times \frac{\lvert x\rvert^{a\times b}}{(1+\lvert x\rvert^{a\times c})^{b/c}}$	
$c \to \infty$					

Table 2.6: The sigmoid limits for $0 < a < \infty$ assuming $|x| < 1$. This is also the case for $0 > a > -\infty$ assuming $|x| > 1$.

$a \to 0$	$b \to -\infty$	$-\infty < b < 0$	$b \to 0$	$0 < b < \infty$	$b \to \infty$
$c \to -\infty$					
$-\infty < c < 0$	0	$\frac{x}{\lvert x\rvert} \times \frac{1}{2^{b/c}}$		$\frac{x}{\lvert x\rvert} \times 2^{-b/c}$	$\frac{x}{\lvert x\rvert} \times \infty$
$c \to 0$		$\frac{x}{\lvert x\rvert} \times \infty$	$\frac{x}{\lvert x\rvert}$	0	
$0 < c < \infty$		$\frac{x}{\lvert x\rvert} \times 2^{-b/c}$		$\frac{x}{\lvert x\rvert} \times \frac{1}{2^{b/c}}$	
$c \to \infty$					

Table 2.7: The sigmoid limits for $a \to 0$ which applies for all values of $|x|$, including its limits.

$-\infty < a < 0$	$b \to -\infty$	$-\infty < b < 0$	$b \to 0$	$0 < b < \infty$	$b \to \infty$
$c \to -\infty$					
$-\infty < c < 0$	0	$\frac{x}{\lvert x\rvert} \times \frac{\lvert x\rvert^{a\times b}}{(1+\lvert x\rvert^{a\times c})^{b/c}}$		$\frac{x}{\lvert x\rvert} \times \left(1+\left\lvert\frac{1}{x}\right\rvert^{a\times c}\right)^{-b/c}$	$\frac{x}{\lvert x\rvert} \times \infty$
$c \to 0$		$\frac{x}{\lvert x\rvert} \times \infty$	$\frac{x}{\lvert x\rvert}$	0	
$0 < c < \infty$		$\frac{x}{\lvert x\rvert} \times (1+\lvert x\rvert^{-a\times c})^{-b/c}$		$\frac{x}{\lvert x\rvert} \times \frac{1}{(1+\lvert x\rvert^{-a\times c})^{b/c}}$	
$c \to \infty$					

Table 2.8: The sigmoid limits for $-\infty < a < 0$ assuming $|x| < 1$. This is also the case for $\infty > a > 0$ assuming $|x| > 1$.

<table>
<tr><td>$a \to -\infty$</td><td>$b \to -\infty$</td><td>$-\infty < b < 0$</td><td>$b \to 0$</td><td>$0 < b < \infty$</td><td>$b \to \infty$</td></tr>
<tr><td>$c \to -\infty$</td><td colspan="5"></td></tr>
<tr><td>$-\infty < c < 0$</td><td colspan="2" rowspan="2">$\frac{x}{\lvert x \rvert} \times \infty$</td><td rowspan="2">$\frac{x}{\lvert x \rvert}$</td><td colspan="2" rowspan="2">0</td></tr>
<tr><td>$c \to 0$</td></tr>
<tr><td>$0 < c < \infty$</td><td colspan="4" rowspan="2"></td></tr>
<tr><td>$c \to \infty$</td></tr>
</table>

Table 2.9: The sigmoid limits for $a \to -\infty$ assuming $|x| < 1$ which includes the limit $|x| \to 0^{\pm}$. This is also the case for $a \to \infty$ assuming $|x| > 1$ which includes the limit $|x| \to \infty$.

modifications. They can approximate/replicate a large number of threshold functions even outside the genre of sigmoid functions just by changing the parameters a, b, c, and the parameters and resulting output are unit-less regardless of the input's units. If the signed and unsigned functions are used together, it is also possible to increase the variety of representable curves further, *e.g.* using $sig^{+}(x, a_1, b_1, c_1) \times sig^{-}(x, a_2, b_2, c_2)$. For completeness, the first order derivatives are defined as:

$$\begin{aligned} \frac{\delta sig^{-}(x,a,b,c)}{\delta x} &= \frac{x \times |x|^{a \times b - 1}}{(|x|^{a \times c} + 1)^{b/c}} \times \frac{a \times b}{(|x|^{a \times c} + 1) \times |x|} \\ &= sig^{+}(x,a,b,c) \times \frac{a \times b}{(|x|^{a \times c} + 1) \times |x|} \end{aligned} \tag{2.120}$$

and

$$\begin{aligned} \frac{\delta sig^{+}(x,a,b,c)}{\delta x} &= \frac{|x|^{a \times b}}{(|x|^{a \times c} + 1)^{b/c}} \times \frac{(a \times b \times x^2 - x^2 \times |x|^{a \times c} + |x|^{a \times c + 2})}{(|x|^{a \times c} + 1) \times |x|^3} \\ &= sig^{-}(x,a,b,c) \times \frac{(a \times b \times x^2 - x^2 \times |x|^{a \times c} + |x|^{a \times c + 2})}{(|x|^{a \times c} + 1) \times |x|^3}. \end{aligned} \tag{2.121}$$

It is not surprising that the derivative of one has the other cleanly present in its derivative, but this does further suggest that they are a pair. Figures 2.5 to 2.7 are samples of several variations of a, b, and c in the signed function, showing how they distort the curve. From these graphs, it can be concluded that a causes a rotation $x = 1$, the decrease in magnitude of b causes a compression along the x-axis towards the y-axis, and the magnitude of c determines the

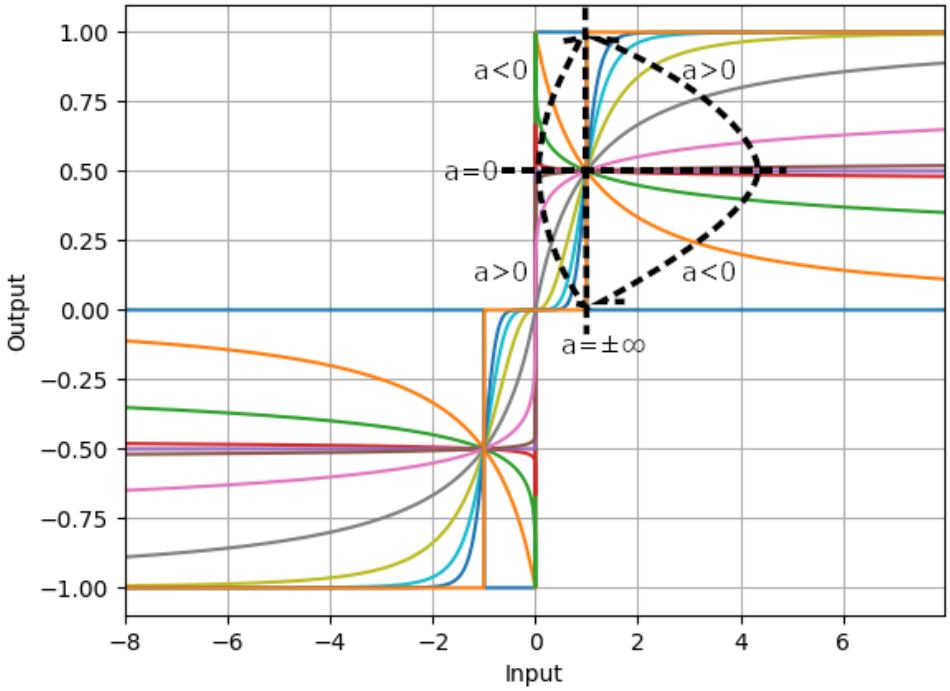


Figure 2.5: The signed general sigmoid function given that a varies while $b = c = 1$. The arrows show how the curve is distorted w.r.t. changes in a.

crispness w.r.t. the corner point at $x = 1$, *i.e.* how close the output approaches point $(1, 1)$ for $x = 1$. The triangular relationship between a, b, and c also allows for some flexibility in the choice of values, *e.g.* if $b/c = 1$ then $b_{new} = c_{new} = 1$ and $a_{new} = a \times b$.

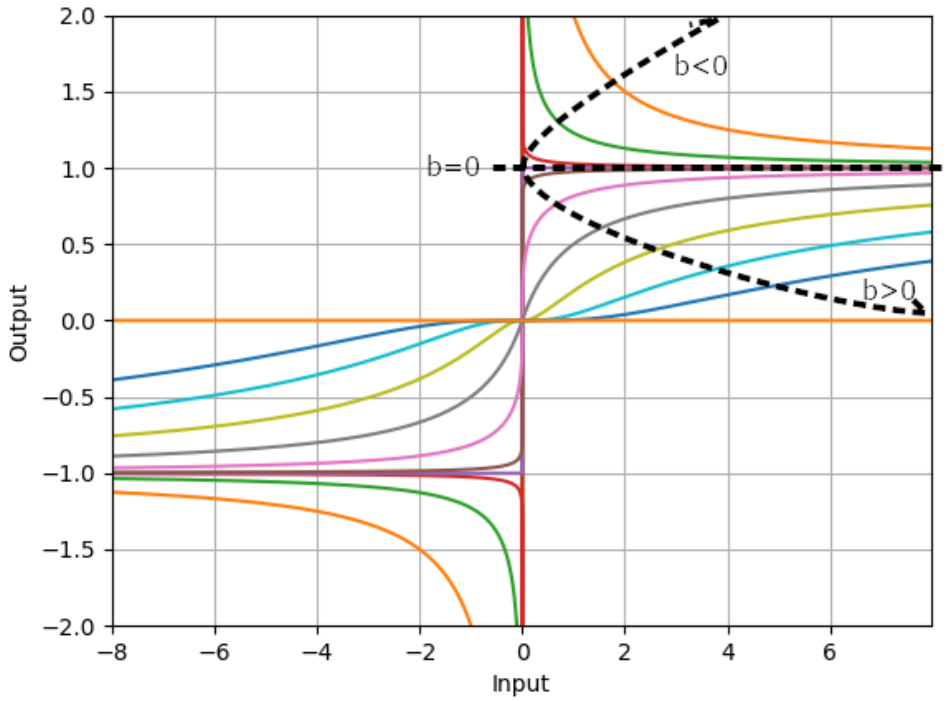


Figure 2.6: The signed general sigmoid function given that b varies while $a = c = 1$. The arrows show how the curve is distorted w.r.t. changes in b.

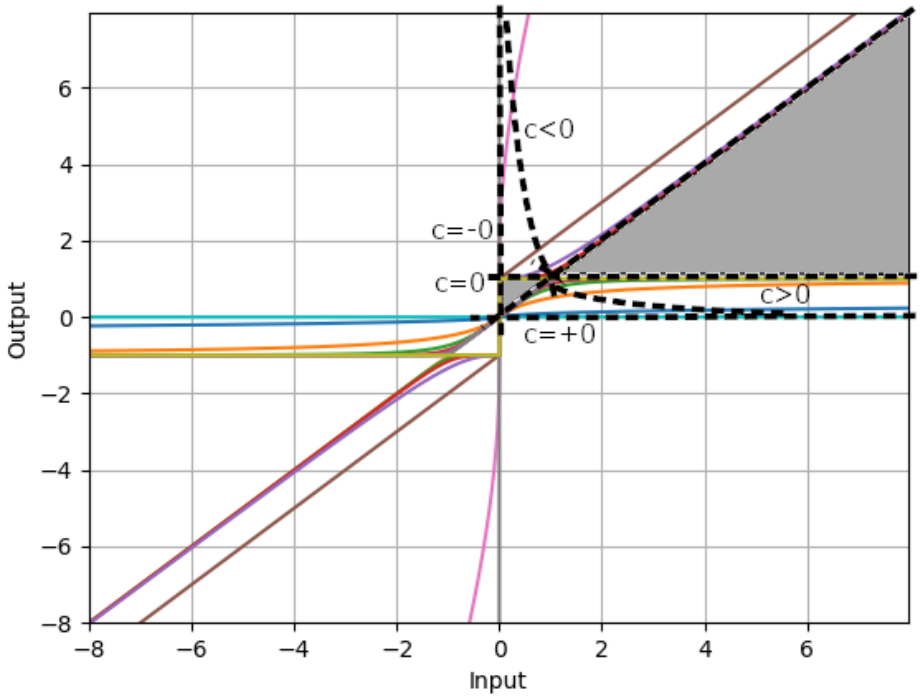


Figure 2.7: The signed general sigmoid function given that c varies while $a = b = 1$. The arrows show how the curve is distorted w.r.t. changes in c and the filled in regions are the areas that the curve will never pass through with the given values of a and b. It should be noted that $c = 0$ is the step function while $c = \pm 0$ results in a strait horizontal or vertical line respectively.

Chapter 3

Baseline: SARSA

As SARSA (State-Action-Reward-State-Action) is closely aligned with the fundamentals of Temporal Difference Reinforcement Learning (TDRL), this research will be using it as the starting point and baseline for improvement. Since most TDRL algorithms are biologically inspired and designed with the intention of emulating the human brain, attempts will be made to clarify the insights and perspectives regarding the behavior/habits of the algorithm and analogies of the components. NNs attempt to model the brain by emulating individual neurons and their connections, while TDRL seeks to model policy and valuation aspects seen on a behavioral level. TDRL does this by iteratively adding expected value with an inversely proportional relation to the distance from the state or states that give rewards, creating an expected value gradient that can be followed to the goal [36]. If a positive reward can be equated to a neurotransmitter that incites an individual to act, then TDRL models the dispersion of neurotransmitters among a cluster of neurons, or more specifically their dendrites which are stimulated by state inputs. To further demonstrate SARSA's similarity to a NNs, by reducing the algorithm to having only one action, it becomes similar to a perceptron in how it converts inputs to outputs, but with a different learning rule applied.[1]. This chapter will start with the foundation of SARSA and progressively work though the algorithm to en-

[1]As the action selection process for SARSA is usually based on a probability distribution across all actions, *i.e.* competitive activation, using only one action output is functionally useless as it can only choose one action, but still serves to demonstrate the similarity between TDRL and NNs.

sure later changes will have a proper basis of interpretation with which to be compared.

3.1 Bellman Equation

As the basis for SARSA, Q-Learning, and Monte Carlo, it is important to give an acceptable interpretation for the algorithm's underlying principles. The Bellman Equation is defined using these two fundamental equations [36]:

$$v(s_t) = \sum_{a_{t+0.5}}^{A_{t+0.5}} \big(\mu(a_{t+0.5}|s_t) \times q(a_{t+0.5}, s_t)\big), \tag{3.1}$$

and

$$q(a_{t+0.5}, s_t) = \sum_{r_{t+0.5}} \sum_{s_{t+1}}^{S_{t+1}} \Big(\rho(r_{t+0.5}, s_{t+1}|s_t, a_{t+0.5}) \times \big(r_{t+0.5} + \gamma_{t+0.5} \times v(s_{t+1})\big)\Big). \tag{3.2}$$

From equation (3.1), $v(s_t)$ is the approximation of the state's expected value; $\mu(a_{t+0.5}|s_t)$ is the probability of taking action $a_{t+0.5}$ out of all valid actions $A_{t+0.5}$ given state s_t, *i.e.* μ is the agent's action policy; and $q(a_{t+0.5}, s_t)$ is the quality or expected value approximation of the action and state combination.[2] The state-action value $q(a_{t+0.5}, s_t)$ can be calculated using equation (3.2), where $p(r_{t+0.5}, s_{t+1}|s_t, a_{t+0.5})$ describes the next state — existing within S_{t+1} — and reward probabilities given the current state and intended action. The probability values $p(r_{t+0.5}, s_{t+1}|s_t, a_{t+0.5})$ can be calculated/approximated using a model, however, model-based algorithms tend to strongly rely on the premise that the model is accurate. Learning Algorithms (LAs) such as SARSA are classified as model-free methods. They integrate $p(r_{t+0.5}, s_{t+1}|s_t, a_{t+0.5})$ into the state-action value $q(a_{t+0.5}, s_t)$ approximation and by extension into the state value $v(s_t)$ through their gradual learning processes. The reward, $r_{t+0.5}$, naturally is the measure of goodness/badness that is experienced for a given action or state. The discounting factor, γ, is used to determine the degree of

[2]Taking action $a_{t+0.5}$ results in setting the elements of the action space encoding $a_{t+0.5}$ to 1 while all others are set to 0. Typically, SARSA only has one element of $\vec{a}_{t+0.5}$ represent a corresponding element in the action space A, thus setting one $a_{t+0.5}$ to 1 while all vector elements are 0 is equivalent to selecting one action from A.

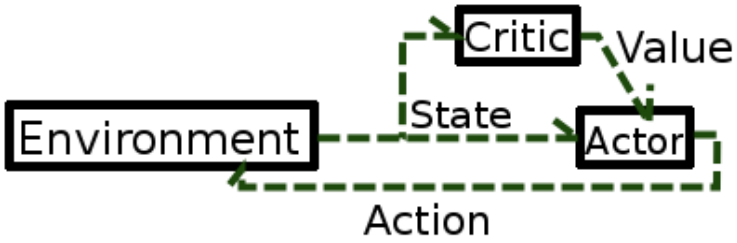


Figure 3.1: The Actor-Critic Learning Model.

short-sightedness in the face of rewards, where, for $\gamma \in [0, 1)$, a larger value gives greater weight to a future reward. In some algorithms, γ will be a function based on factors such as time or state.

Given a deterministic solution for every step, *i.e.* the Monte Carlo solution, the cumulant is [37]:

$$G_{t+N-0.5} = \sum_{n=0}^{N-1} \big((\prod_{k=0}^{n} \gamma_{k+0.5}) \times r_{t+n+0.5} \big). \tag{3.3}$$

$\gamma_{k+0.5}$ is the discounting factor at transition $t + k + 0.5$, and $r_{t+n+0.5}$ is the reward received when transitioning from time step $t+n$ to time step $t+n+1$. The cumulant G_∞ is what the Monte Carlo solution converges to and Bootstrapped methods approximate [36].

3.2 Discrete SARSA Actor-Critic

Discrete SARSA refers to the algorithm structure designed to give outputs for a discrete/logic-based action space as opposed to a continuous/value-based action space. Actor-Critic methods are designed to calculate and learn the policy $\mu(a_{t+0.5}|s_t)$ independently from the state-action value $q(a_{t+0.5}, s_t)$. In contrast, non-AC methods use the state-action value as an approximation of the policy when using greedy single action selection methods. Though non-AC methods are simpler, they inherently assume a fixed number of actions — typically only one — will be selected. AC methods divide the burden of learning state/action quality and action policies, making each step more efficient as the actor and critic strictly focus on learning policies and expected values respectively [36].

As a part of the foundational research, a comparison of the linear and gradient methods was carried out, SARSA and SARSA(λ) respectively. For the SARSA(λ) algorithm, the update equations are defined as:[3]

$$q(\vec{a}_{t+0.5}, \vec{s}_t) = \vec{a}^T_{t+0.5} \cdot [Q] \cdot \vec{s}_t, \tag{3.4}$$

$$\vec{a}_{t+0.5} = [\mu] \cdot \vec{s}_t, \tag{3.5}$$

$$\delta_t = r_{t-0.5} - \bar{r}_t + \gamma_p \times q(\vec{a}_{t+0.5}, \vec{s}_t) - q(\vec{a}_{t-0.5}, \vec{s}_{t-1}), \tag{3.6}$$

$$\bar{r}_{t+1} = \bar{r}_t + \delta_t \times \alpha_r, \tag{3.7}$$

$$[e] = \gamma_p \times \lambda \times [e] + \frac{\delta_t \times \vec{a}_{t-0.5} \cdot \vec{s}^T_{t-1}}{\max(1, \sum \vec{a}_{t-0.5} \cdot \vec{s}^T_{t-1})}, \tag{3.8}$$

and

$$[Q] = [Q] + \alpha_q \times [e], \tag{3.9}$$

where $[e]$ is the trace of the gradient of $q(\vec{a}_{t+1.5}, \vec{s}_{t+1})$, and λ is the forgetting factor of the trace. To make this an Actor-Critic method, a separate term is updated for the policy along side the weight update as:

$$[\tau] = \gamma_p \times \lambda \times [\tau] + \frac{(\vec{a}_{t-0.5} - \vec{p}(A_{t-0.5}|\mathbf{s}_{t-1})) \cdot \vec{s}^T_{t-1}}{\max(1, \sum |\vec{s}^T_{t-1}|)} \tag{3.10}$$

$$[\mu] = [\mu] + \alpha_\mu \times \delta_t \times [\tau], \tag{3.11}$$

where $\vec{p}(A_{t+0.5}|\vec{s}_t)$ is the probability distribution (or prediction of $\vec{a}_t$) over the action set A_t given state $\vec{s}_t$, and $[\tau]$ is the trace of the policy gradient. In difference to SARSA(λ), SARSA does not have a trace and equations (3.9) and (3.11) are replaced with:

$$[Q] = [Q] + \frac{\alpha_q \times \delta_t \times \vec{a}_{t-0.5} \cdot \vec{s}^T_{t-1}}{\max(1, \sum \vec{a}_{t-0.5} \cdot \vec{s}^T_{t-1})} \tag{3.12}$$

and

$$[\mu] = [\mu] + \frac{\alpha_\mu \times \delta_t \times (\vec{a}_{t-0.5} - \vec{p}(A_{t-0.5}|\vec{s}_{t-1})) \cdot \vec{s}^T_{t-1}}{\max(1, \sum |\vec{s}^T_{t-1}|)} \tag{3.13}$$

respectively. It should be noted that the action probabilities $\vec{p}(A_{t+0.5}|\vec{s}_t)$ will be calculated based on what is suitable for the LA: Soft-Max for SARSA(λ)

[3]Some elements are shown in matrix form for clarity, *i.e.* demonstrating the linear properties of the equations. The functions are broken down into steps for visual ease and consistency w.r.t. syntax and labels. Parametric symbols γ_p and γ_q are used for consistency in function with the modified algorithms.

and Piece-wise linear for SARSA (refer to section 5.2.3 for details). It should also be noted that, for the linear SARSA method, the initial weights of $[Q]$ will be set to zero and the initial weights of $[\mu]$ will be set to $1/(\#S \times \#A)$. The non-zero initial values of $[\mu]$ allow the policy to start off as a uniform random policy as a way of generating the push needed for the agent to start learning. In a prior work, it was found that SARSA(λ) is not suitable for this type of non-stationary problem, however, we have included its equations for completeness.

3.3 Analysis

Since the purpose of this research is to derive a new algorithm based on SARSA, it is best to have a firm grasp of how its foundation can be interpreted. In section 3.1, the Bellman Equation was defined using inputs/outputs s_t, $a_{t+0.5}$, and $r_{t+0.5}$; functions $p(r_{t+0.5}, s_{t+1}|a_{t+0.5}, s_t)$, $\mu(a_{t+0.5}|s_t)$, $q(a_{t+0.5}, s_t)$, and $v(s_t)$; and the constant $\gamma_{t+0.5}$. This section will give alternative interpretations that are more closely aligned with physiology and psychology. By drawing relations between the algorithm and biological neurons, it becomes easier to identify components that have been simplified or left out. From the elements that have been included, identifying how they may have been simplified can lend some explanation as to why some issues are present in the algorithm. Regarding components that have been excluded, the biological interpretation lends some explanation of what their purpose is and suggests how they can be integrated into the algorithm.

To make the relations between the algorithm and the biological neuron clear, several conjectures will be proposed. The neuron — like a state or action — tends to have two key modes activity, namely being active or resting. From the perspective of neurobiology, it can be concluded that an active action/state corresponds to an activated neuron or set of neurons in part or in whole, *i.e.* their cell body and/or their axon at a corresponding time:

Conjecture 3.1. Neural State and Action — *State* $s_t = 1$ *and action* $a_{t-0.5} = 1$ *correspond to activated axons of neuron* s *and* a*, while state* s_{t+1}

and action $a_{t+0.5}$ equate to the stimulated cell body of neuron s and a in the following time step.

If this is to be accepted, it could then be concluded that state value and state-action values would correspond to the neurotransmitters released during activation/stimulation:

Conjecture 3.2. Neural State Value — *The state value $v(s_t)$ corresponds to the neurotransmitters released when the axon and/or cell body of neuron s is stimulated/activated.*

Conjecture 3.3. Neural State-Action Value — *The state-action value $q(a_{t+0.5}, s_t)$ corresponds to the neurotransmitters released during neuron a's stimulation by neuron s, likely from the dendrite of neuron a.*

The assumption that these values are neurotransmitters is further reinforced by their relation with the $r_{t+0.5}$ which is a direct representation of rewards and punishments. If these statements are to be accepted, then the neural responses to stimuli, being an activation probability associated with the next time step matches the function of $p(r_{t+0.5}, s_{t+1}|a_{t+0.5}, s_t)$ and $\mu(a_{t+0.5}|s_t)$. As the stimulation of a given cell s or a at time t increases, its likelihood of activation in the next time step — $s_{t+1} = 1$ or $a_{t+0.5} = 1$ respectively — also increases. Given the policy and state prediction are related to the strength of stimulation is within reason to expect:

Conjecture 3.4. Neural Policy Value — *The policy value $\mu(a_{t+0.5}|s_t)$ corresponds to how well neuron s's axon can stimulate neuron a's dendrites or how easily neuron s can stimulate neuron a.*

Conjecture 3.5. Neural Prediction Value — *The prediction value $p(r_{t+0.5}, s_{t+1}|s_t, a_{t+0.5})$ has two parts. The first part, $p(s_{t+1}|s_t, a_{t+0.5})$, corresponds to how well neuron $s_{@t}$'s axon and stimulated neuron $a_{@t+0.5}$ can stimulate neuron $s_{@t+1}$'s dendrites. The second part, $p(r_{t+0.5}|s_t, a_{t+0.5})$, corresponds to the probability of however much neurotransmitter will be expected to be present in the vicinity in the immediate future.*[4]

[4]As a probability, it is unable to clarify how much and would likely represent the probability of $\mathbb{E}\{r_{t+0.5}|s_t, a_{t+0.5}\}$.

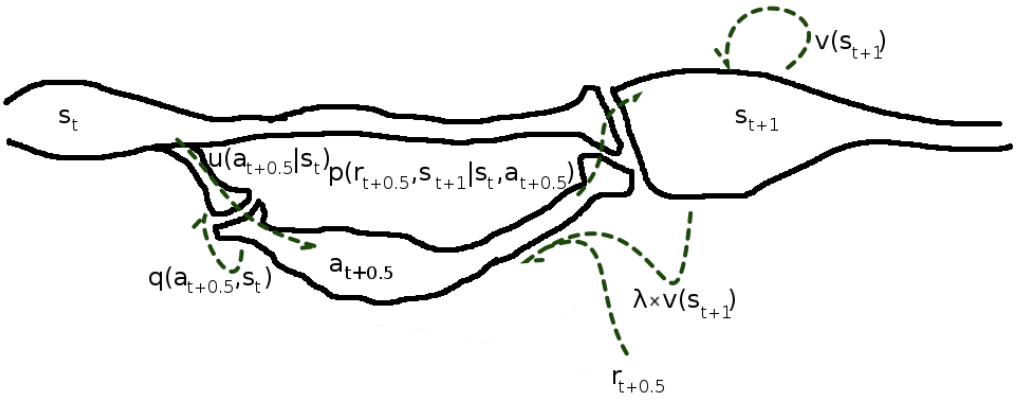


Figure 3.2: An interpretation of the Bellman Equation where signals flow forward and neurotransmitters flow backward and cyclically.

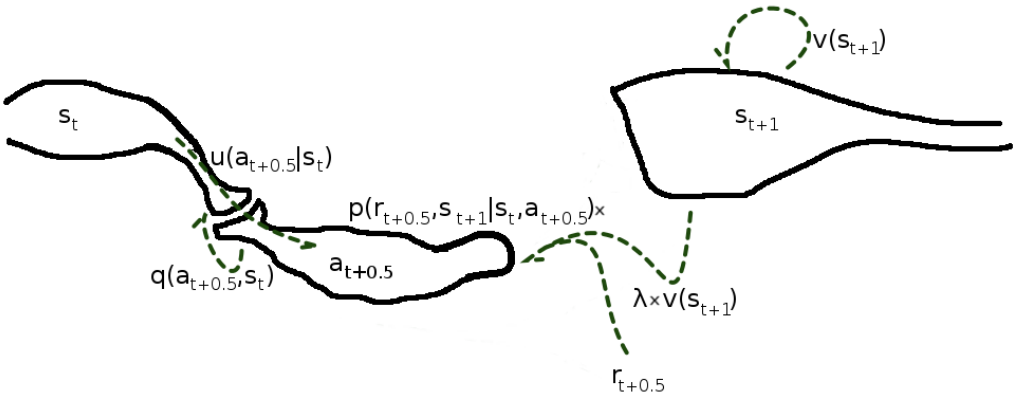


Figure 3.3: An interpretation of SARSA where the model component is removed.

The last component to be interpreted is the discounting factor $\gamma_{t+0.5}$. With the conjectures thus far, it is reasonable to conclude:

Conjecture 3.6. Neural Discounting Factor — *The discounting factor $\gamma_{t+0.5}$ corresponds to the percentage of neuron $s_{@t+1}$'s neurotransmitter release that will be carried back to neuron $s_{@t}$.*[5]

Drawing a simple diagram of a neuron model to show the locations associated with each of the Bellman Equation's components, it would probably as shown in figure 3.2. This model carries all the components described in the Bellman Equation such that neurotransmitters flowing together are added

[5]This does not clarify if the difference is covered by production of neurotransmitters by neuron $s_{@t+1}$ or $s_{@t}$.

then multiplied by the counter-flowing signal activity. Given that SARSA has the world model removed or included in the absorption of the associated neurotransmitters, the resulting model will be closer to what is shown in figure 3.3. If these statements and diagrams give a proper representation of the Bellman Equation and SARSA, then some areas for improvement become rather easy to identify.

Chapter 4

Modifications: CLASP

Building off of the biological interpretations derived when analyzing SARSA, the first revision to be considered is improving symmetry between states and actions (see figure 4.1). Symmetry between states and actions allows the algorithm to create action-to-state, state-to-state, and action-to-action relations in addition to the original state-to-action relation. This allows us to group states within the set S_t and actions within the set $A_{t-0.5}$ together, *e.g.* $N_{t+1} \equiv S_{t+1} \cap A_{t+0.5}$ for generic state-action predictions. This model-based form removes one of the major benefits offered by SARSA, however, it will be necessary for future modifications which introduce complex logical states [36]. Additionally, it will allow for a more thorough understanding of where the rewards are being administered from and how it can regulate its own policy. Later modifications address other aspects such as input space size compression, inclusion of historical and predictive elements in the input space, and restricting the scale of each output. It should be noted that, with the inclusion of an internal model, only the bias term for each logical output will be initialized with a non-zero value.

4.1 Model and Policy Matching

By including an internal model and reducing the algorithm to one state prediction or one action prediction, the most distinct difference between states and actions is whether or not the output can cause a change in the environ-

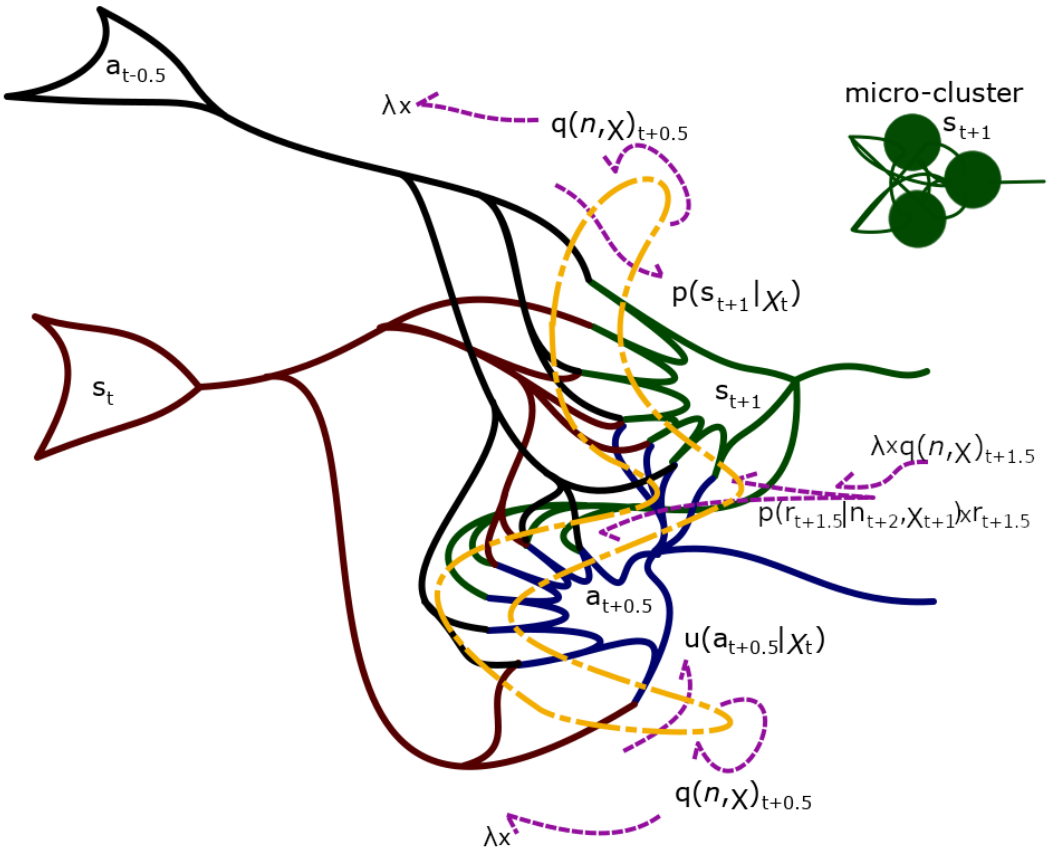


Figure 4.1: A Simplified interpretation of SARSA with a model and including logical actions. In contrast to the visual interpretations for the model-free SARSA and Bellman Equation, logical information is expected to flow seamlessly into neurons representing future states or actions. The dispersion of neurotransmitters — represented by quality prediction — is preserved, however, it occurs at the dendrites and is largely contained within the localized area of activity (circled in yellow) instead of at the cell body. γ also no longer represents the percent of neurotransmitters that disperse from the stimulated cell body, but the percent diffusing away from the synaptic region. This suggests neurotransmitters can be released even if the stimulated dendrites do not carry the signal to the cell body. If state and action elements are allowed to stimulate themselves in this model, it would be more accurate to say that each element is represented by a micro-cluster with internal connections and the output is an aggregate of all representatives within.

ment.[1] As state and action predictions are so similar, it is easier to regard both as next state predictions where actions represent transitional states.[2] By extension, the update method for both can be expected to be similar. When considering these aspects, it is acceptable to say that:

Claim 4.1. Actions are transitional states — *Just as states are used to indicate the condition of the environment, logical actions can be used to indicate the current state of the LA and the intended method of transition between the current state and the next state.*

If the agent will be trained to predict the next character in the words 'caterpillar' and 'catamaran,' it is possible for the fourth letter, *e.g.* 'e,' to be consistently predicted as 'a.' If the erroneous prediction is consistent, the agent's selected action, given as an input, allows for an embedded understanding of the known error in the policy which can be exploited to improve the prediction of the next letter, *i.e.* $\vec{s} \cap \vec{a} \in \{\text{'ee', 'ea'}\} \implies$ 'r' has more potential for convergence than $\vec{s} \in \{\text{'e'}\} \implies$ 'r'. To implement state-action symmetry, the equations for the state-action value and policy will be rewritten as:

$$q(a,s)_t = \begin{bmatrix} \vec{s}_{t+1} \\ \vec{a}_{t+0.5} \end{bmatrix}^T \cdot [Q] \cdot \begin{bmatrix} \vec{s}_t \\ \vec{a}_{t-0.5} \end{bmatrix} \tag{4.1}$$

$$\vec{a}_{t+0.5} = [\mu] \cdot \begin{bmatrix} \vec{s}_t \\ \vec{a}_{t-0.5} \end{bmatrix}, \tag{4.2}$$

and the unbounded prediction method will be define as:

$$\vec{s}_{t+1} = [p] \cdot \begin{bmatrix} \vec{s}_t \\ \vec{a}_{t-0.5} \end{bmatrix}. \tag{4.3}$$

Unfortunately, there is a critical flaw in this method, *i.e.* all functions are linear and there is no mixing of inputs.[3] If there are any outcomes that are dependent on more than one state and/or action, these methods can only give

[1] This interpretation implies that both state and action probabilities are correlated to the environment's future, but the output is only regarded as an action if it has a causal relationship with the future, *i.e.* correlation does not mean causation for states but does for actions.

[2] Though the policy distribution is the cause for activation, it also serves as a probability w.r.t. the associated random number being sufficiently low to cause activation, *i.e.* this view reverses the perspective of cause and effect for convenience of symmetry.

[3] This will be resolved through the kernel trick addressed in section 4.3

a linear approximation. For compactness, state-action pairs will be presented as:

$$\vec{\eta}_{t+1} = \begin{bmatrix} \tilde{1} \\ \vec{s}_{t+1} \\ \vec{a}_{t+0.5} \end{bmatrix} \tag{4.4}$$

for non-thresholded future values,

$$\vec{\chi}_t = \begin{bmatrix} 1 \\ \vec{s}_t \\ \vec{a}_{t-0.5} \end{bmatrix} \tag{4.5}$$

for logical percent values, *i.e.* facts, and

$$\vec{p}(N_{t+1}|\chi)_t = \vec{p}(\begin{bmatrix} S_{t+1} \\ A_{t+0.5} \end{bmatrix}|\chi_t) \tag{4.6}$$

for thresholded predictions of future logical values. With these new representations, the quality will be expressed as:

$$q(\eta, \chi)_{t+0.5} = \vec{\eta}_{t+1} \cdot [Q^2] \cdot \vec{\chi}_t; \tag{4.7}$$

and policy as:

$$\vec{\eta}_{t+1} = \left[p^2\right] \cdot \vec{\chi}_t, \tag{4.8}$$

bounded by:

$$p(N_{t+1}|\vec{\chi})_t = \begin{cases} 1 & \text{if } \eta_{t+1} \geq 1 \ , \\ \eta_{t+1} & \text{if } 1 > \eta_{t+1} > 0 \ , \\ 0 & \text{if } 0 > \eta_{t+1} \ ; \end{cases} \tag{4.9}$$

The benefit of these policy and prediction methods are that it is possible to identify if reward values are caused by the state transitioned into, the action used, the state departed from, or a combination of two or more of these cases. This will grant the LA a more detailed understanding about the happenings between time step t and $t+1$ and not just a vague summary of the results assigned to the action and previous state. The obvious demerit that should be kept in mind is that, for horde architectures, the inbuilt state-models can be redundant and very wasteful. Though it is beyond the scope of this research, it will be desirable to separate each output without affecting overall performance and stability, *i.e.* converting the TDRL algorithm into a neural network model which steps within the deadly triad [36].

As the only change is w.r.t. how the policy and state-action value are calculated, the error and mean reward calculations can remain as:

$$\delta_t = r_{t-0.5} - \bar{r}_t + \gamma_p \times q(\eta, \chi)_{t+0.5} - q(\eta, \chi)_{t-0.5} \tag{4.10}$$

and

$$\bar{r}_{t+1} = \bar{r}_t + \delta_t \times \alpha_r. \tag{4.11}$$

An interesting property that crops up because of the new method is that, similarly to $\bar{r}_{t+1}$, $q(\tilde{1}, 1)_{t+0.5}$ will effectively be the approximation of the mean state-action value, *i.e.* equation (4.11) will be redundant. Now that the model and policy have been defined, the update equations can also be redefined to accommodate the changes in policy and the addition of the model:[4]

$$\partial_t = \begin{cases} \alpha_p \times (\chi_t - p(N_t|\chi)_{t-1}) & \text{for state predictions,} \\ \alpha_\mu \times \alpha_p \times (\chi_t - p(N_t|\chi)_{t-1}) \times \delta_t & \text{for policy learning;} \end{cases} \tag{4.12}$$

$$[Q^2] = [Q^2] + \frac{\alpha_q \times \delta_t \times \vec{\chi}_t \otimes \vec{\chi}_{t-1}}{\max(1, (\sum |\vec{\chi}_t|) \times (\sum |\vec{\chi}_{t-1}|))}; \tag{4.13}$$

and

$$[p^2] = [p^2] + \frac{\vec{\partial}_t \otimes \vec{\chi}_{t-1}}{\max(1, (\sum |\vec{\chi}_t|) \times (\sum |\vec{\chi}_{t-1}|))}. \tag{4.14}$$

The policy update and model update will be different from each other because the policy is expected to change with reward, while the model is only mapping out what will be possible/likely regardless of the reward.

4.1.1 Equation Interpretation

The inclusion of an action feedback suggests that some cells will require capacity for more than one active source. The inclusion of a model allows all dendrites to have a corresponding axon, *i.e.* each χ_t has an associated η_{t+1}. This also means that any state or action element can influence the neurotransmitter release as $q(\eta, \chi)_{t+0.5}$. Consequently, the $\gamma_p \times q(\eta, \chi)_{t+0.5}$ component in equation (4.10) must then be a non-targeted emission much like the reward

[4] $\otimes$ denotes the outer product, which allows us to construct the tensor for the state and error update information.

$r_{t+0.5}$. Reviewing the conjectures made in 3.3, several revisions will be in order. Firstly, Conjecture 3.1 must be revised to:

Conjecture 4.1. Neural State and Action — *State/action χ_t corresponds to the activated axon of a representative/dominant neuron χ within a micro-cluster of neurons that collectively represent χ, while state/action $p(\eta_{t+1}|\chi)_t$ equates to stimulated dendrites and/or cell bodies of micro-cluster η.*

This correction is required because $p(\eta_{t+1}|\chi_t)$, where $\chi_t|_{\chi_t=\eta_{t+1}}$, can be non-zero; however, most biological neurons are not self-activating, instead operating in groups to increase reliability through a majority vote [20]. Though it is not likely to require detailing the internal interactions of these micro-clusters, they will likely be more correct for our models and may also carry some preferred features that singular neurons cannot offer.

Because the model will be included in the calculations, the original equation for the state value $v(s)$ will not carry over to the condition value $v(\chi)$ and/or $v(\eta|_{\eta=\chi})$. Interestingly, there are three values, $q(\tilde{1},\chi)_{t+0.5}$, $q(\eta|_{\eta=\chi},1)_{t-0.5}$, and $q(\eta|_{\eta=\chi},\chi)_{t\pm0.5}$, that carry similar meaning to $v(\chi)$ and/or $v(\eta|_{\eta=\chi})$. The difference lay in the transition relative to χ/η: leaving, entering, or staying respectively. The value function can be defined as:[5]

$$v(\chi)_t = q(\tilde{1},\chi)_{t+0.5} + \frac{q(\eta|_{\eta=\chi},\chi)_{t+0.5} + q(\eta|_{\eta=\chi},\chi)_{t-0.5}}{2} + q(\eta|_{\eta=\chi},1)_{t-0.5}. \tag{4.15}$$

$q(\tilde{1},\chi)_{t+0.5}$ describes the expected average value for leaving a particular state; and $q(\eta|_{\eta=\chi},1)_{t-0.5}$ is the expected value of entering the state or using the action. The middle term must be regarded as a result of intra-micro-cluster interactions, describing the value of transitioning to and from the same state or action. The values $q(\tilde{1},\chi)_{t+0.5}$ and $q(\eta,1)_{t+0.5}$ are somewhat interesting: they imply a diffusive property exists along side $q(\eta,\chi)_{t+0.5}$, meaning some neurotransmitters can be emitted from or drawn in by neuron χ without a targeted dendrite; similarly, neuron η can release/absorb neurotransmitters passively.

[5]This equation deliberately does not distinguish between state and action within χ as the value of a state-action pair or just the action — a transitional state — can also be of interest.

Though the equation for $v(\chi)$ changes significantly, the interpretation given by conjecture 3.2 remains more or less correct, however, a subtle difference must be clarified:

Conjecture 4.2. Neural Circumstance Value — *The circumstance value $v(\chi)_t$ corresponds to the neurotransmitters released purely due to neuron or micro-cluster χ's stimulation or activation.*

This is acceptable despite not multiplying and summing with the policy as in equation (3.1) because the mean value will already be relative to the policies that were learned up to time t. Additionally, the calculation in equation (4.15) will not be directly dependent on the details regarding other states/actions, *i.e.* it is not based on conditional statements regarding the previous or next state/action such as $a_{t+0.5}|s_t$, $a_{t+0.5}|a_{t-0.5}$, $s_{t+1}|s_t$, or $s_{t+1}|a_{t-0.5}$.

Though the state-action value definition is unchanged, it would be prudent to rephrase it in a way that will be more acceptable, *i.e.* conjecture 3.3 will be revised to:

Conjecture 4.3. Neural Transition Quality — *The Transition quality $q(\eta, \chi)_{t+0.5}$ corresponds to the neurotransmitters released during neuron/micro-cluster η's stimulation at its dendrites by neuron/micro-cluster χ.*

As implied, this function describes the quality attributed to transitions between states which may or may not include the specific means used to cause the transition, *e.g.* taking action $a_{t+0.5}$ while in state s_t to arrive at state s_{t+1} or transitioning from state s_t to state s_{t+1} irrespectively of the action taken. The primary reason t, $t+0.5$, and $t+1$ are being used so explicitly is to clarify that the action and reward will be realized at some point between time t and $t+1$ in accordance with the transition of states.

Similarly to the transition quality, it will be useful to generalize the interpretation for the policy to include all predictions, *i.e.* conjecture 3.4 should be revised to:

Conjecture 4.4. Neural Policy Value — *The policy value that can be defined as $p(\eta|\chi)$ corresponds to how well neuron χ can stimulate neuron η.*

Since the model is included in this statement, *i.e.* $p(\chi_{t+1}|\eta_{t+1}, \chi_t)$ is learned, the prediction of the reward will also require redefinition. The redefinition of conjecture 3.5:

Conjecture 4.5. Neural Prediction Value — *$p(r_{t+0.5}|\eta_{t+1}, \chi_t)$ corresponds to the probability for the neurotransmitter, in an approximate concentration corresponding to $r_{t+0.5}$, will be present in the vicinity in the immediate future.*

The pleasant part about the logical model being separated from the reward model is that both $p(r_{t+0.5}|\eta_{t+1}, \chi_t)$ and $r_{t+0.5}$ will become external factors entirely determined by the whims of the environment. The expected value $\mathbb{E}\{r_{t+0.5}|\eta_{t+1}, \chi_t\}$ can be calculated as:

$$\mathbb{E}\{r_{t+0.5}|\eta_{t+1}, \chi_t\} = p(r_{t+0.5}|\eta_{t+1}, \chi_t) \times r_{t+0.5}, \tag{4.16}$$

while the approximation $q(\eta_{t+1}, \chi_t)$ relies on the expected values dependent factors χ and η. This also means that a change in policy — η — will not necessarily overwrite the expected quality of a previous learned policy. By extension, it can also be said that $v(\chi)$ for a given policy will remain relatively unchanged when using a different action unless the reward mechanism changes. In such cases where the rewards will be partially or entirely stochastic, the problem can be considered non-stationary.

As a final statement, conjecture 3.6, is still correct but will be generalized to:

Conjecture 4.6. Origin of the Discounting Factor — *The discounting factor γ_p corresponds to the percentage of neuron η_{t+1}'s neurotransmitter release that will be carried back to neuron χ_t.*[6]

To some degree, this discounting factor can be considered a neurotransmitter dispersion factor since the requirement/expectation for the transition quality to be constrained to specific targets — axons needing to be bound to

[6]This does not clarify but suggests that the lost neurotransmitters are replaced by η_{t+1}'s production.

specific dendrites — no longer needs to be strongly adhered to. Given these conjectures, it is expected that a simplified connection model will look similar to what is shown in figure 4.1.

4.2 Quality Error: Modified δ

In the simplest case restricted to immediate rewards, the reward error can be calculated as:

$$\delta_t = r_{t-0.5} - q(\eta, \chi)_{t-0.5}. \tag{4.17}$$

Unfortunately, this is insufficient for emulating the adaptive behavior of the biological neuron. By considering the reward to be a neurotransmitter whose magnitude corresponds to the logarithm of the concentration in a fixed volume, the error becomes a discrepancy in the prediction creating a deficit or surplus of chemical stimuli. This will in turn result in a shortage or excess of recovered neurotransmitters and correspondingly influence the dispersion gradient of extracellular neurotransmitters. These consequences in turn, influence the dynamics of the cell growth which seeks to adapt to the ever-changing environment. Using steps, the first step will be to represent the neurotransmitter sources and amounts, *e.g.*:

$$\mathfrak{R}_{t-0.5} = r_{t-0.5} + (1 - \gamma_b) \times (\gamma_q \times q(\eta, \chi)_{t-0.5} + \gamma_{\mathfrak{R}} \times \mathfrak{R}_{t-1}) \tag{4.18}$$

for what will be kept and:

$$\mathfrak{r}_t = \gamma_{\mathfrak{R}} \times \gamma_b \times (\gamma_q \times q(\eta, \chi)_{t-0.5} + \gamma_{\mathfrak{R}} \times \mathfrak{R}_{t-1}) \tag{4.19}$$

for amounts that will 'bleed' out beyond the network's area of influence. γ_b represents the percentage of neurotransmitters that will disperse from the local region [27]; $\gamma_{\mathfrak{R}}$ corresponds to the life expectancy of the neurotransmitter that will remain from half of an iteration [6]; and γ_q is the percent total stored concentration that will be utilized per time step [35]. As neurotransmitters will be considered a relatively limited resource, γ_q will serve to indicate roughly how many activations would be required to exhaust a particular connection

w.r.t. rewards. The implication of these equations is that rewards can be self-administered under certain conditions and excess rewards will be pushed off to later iterations.

$\mathfrak{r}_t$ will become important for systems composed of sub-networks where a summary of local information must be shared with other regions, *i.e.* as a part of $r_{t-0.5}$, increasing the algorithm's scalability.[7] Based on how rewards are learned, one of the primary differences between Temporal Difference Reinforcement Learning (TDRL) and Neural Networks (NNs) is that the former primarily relies on interactions at a group level while the latter uses an individual or layer-by-layer level of processing. By allowing some amount of bleeding of rewards to the surroundings, some amount of flexibility is being permitted in how rewards are processed, *i.e.* by individual nodes, small clusters or as an entire network. Though it is beyond this research, node-level processing will be kept in mind for future work.

Moving forward to the next full iteration, the residual neurotransmitter will be calculated as:

$$\mathfrak{R}_t = \gamma_{\mathfrak{R}} \times \mathfrak{R}_{t-0.5} - \gamma_q \times q(\eta, \chi)_{t-0.5} \tag{4.20}$$

where γ_q will also serve as a recovery efficiency parameter. $\mathfrak{R}_t$ closely resembles the prediction error δ_t found in SARSA, however, it lacks the cumulant, therefore the preferred error measure:

$$\delta_t = \mathfrak{R}_t + \gamma_p \times \gamma_q \times q(\eta, \chi)_{t+0.5}, \tag{4.21}$$

will be used. In this instance, γ_p is a percent of the quality that is released prematurely. When sorting through these equations, it can be seen that excessive rewards will be pushed into later iterations while predictions of the next iteration will be drawn to the current quality prediction. This will force the learning process of a given state to be restricted, temporarily blur the region for which the reward is attributed, and yet not have the consequences fully suppressed. The effect of spreading a fraction of the reward around its source

[7] It is assumed that, if $\mathfrak{r}_t$ is the only source of reward, the relation would be $r_{t+0.5} = \gamma_{\mathfrak{R}} \times \mathfrak{r}_t$ as the neurotransmitters decay over time and require time to propagate.

will also be expected to grant some tolerance to rewards that are correctly administered but temporally inconsistent.

For clarity, if equation (4.21) is expanded out, the delta equations will become:

$$\begin{aligned}\delta_t = \gamma_{\mathfrak{R}} \times (r_{t-0.5} + \gamma_{\mathfrak{R}} \times (1-\gamma_b) \times \mathfrak{R}_{t-1}) \\ + \gamma_p \times \gamma_q \times q(\eta,\chi)_{t+0.5} \\ - \gamma_q \times (1 - \gamma_{\mathfrak{R}} \times (1-\gamma_b)) \times q(\eta,\chi)_{t-0.5}\end{aligned} \tag{4.22}$$

for the error,

$$\mathfrak{r}_t = \gamma_{\mathfrak{R}} \times \gamma_b \times (\gamma_q \times q(\eta,\chi)_{t-0.5} + \gamma_{\mathfrak{R}} \times \mathfrak{R}_{t-1}) \tag{4.23}$$

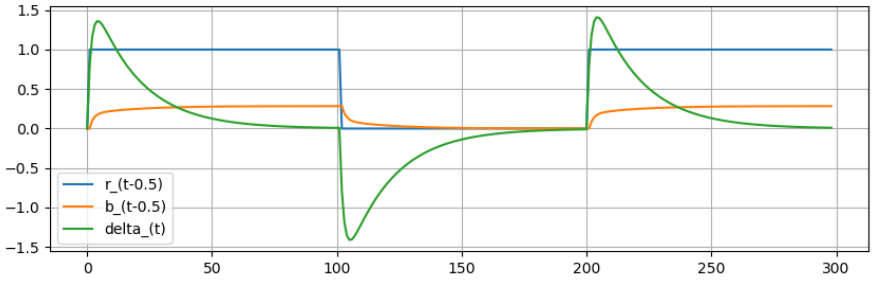
for the dispersed reward, and

$$\begin{aligned}\mathfrak{R}_t = \gamma_{\mathfrak{R}} \times (r_{t-0.5} + \gamma_{\mathfrak{R}} \times (1-\gamma_b) \times \mathfrak{R}_{t-1}) \\ - \gamma_q \times (1 - \gamma_{\mathfrak{R}} \times (1-\gamma_b)) \times q(\eta,\chi)_{t-0.5}\end{aligned} \tag{4.24}$$

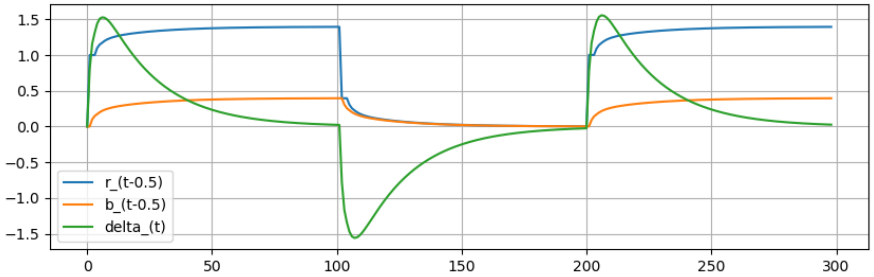
for the on-iteration residual reward. The result of testing with a reward that switches between one and zero is shown in figure 4.2. It is worth noting that under-, critically, and over-damped responses are possible by modifying these γ parameters, allowing for a better range of potential optimal combinations. One case worth pointing out is where $\gamma_b = 1$ such that excess values will be removed and $\gamma_{\mathfrak{R}} = \gamma_q = 1$, allowing all quality values to be fully used. This specific configuration gives the δ_t calculation from SARSA less $\bar{r}$.

4.3 Kerneling

Energy variant coding methods, such as binary-coding, will be a more compact alternative to constant energy coding methods, such as gray-coding. For energy variant methods, it is possible to interpolate based on shared active states, but two similar states-spaces with drastically different policies may hinder the agent's ability to converge for either of them. Kerneling largely remedies this by increasing the number of states available using $\vec{\chi}_t \otimes \vec{\chi}_t$. This



(a) $\mathfrak{r}_t$ is not fed back as part of the reward.



(b) $\mathfrak{r}_t$ is fed back as part of the reward with retention $\gamma_{\mathfrak{R}} = 0.8$.

Figure 4.2: Two Plots of the reward input $r_{t-0.5}$, reward bleed out $\mathfrak{r}_{t-0.5}$ (shown as $b_{t-0.5}$), and error calculation over 300 iterations when $\gamma_b = 0.2$, $\gamma_{\mathfrak{R}} = 0.8$, $\gamma_q = 0.9$, $\gamma_p = 0.1$, $\alpha_q = 0.2$ and all initial values are set to zero. In both cases, the administered reward was 0 for $100 < t \leq 200$ and 1 otherwise. The only active state was the internal-bias value.

change will affect the quality and policy equations as follows:

$$\vec{\eta}_{t+1} = \left[\left[p^3\right] \cdot \vec{\chi}_t\right] \cdot \vec{\chi}_t; \tag{4.25}$$

$$p(N_{t+1}|\chi)_t) = \begin{cases} 1 & \text{if } \mathbb{R}\{\eta_{t+1}\} \geq 1, \\ \mathbb{R}\{\eta_{t+1}\} & \text{if } 1 > \mathbb{R}\{\eta_{t+1}\} > 0, \\ 0 & \text{if } 0 \leq \mathbb{R}\{\eta_{t+1}\}; \end{cases} \tag{4.26}$$

$$q(\eta, \chi)_{t+0.5} = \vec{\eta}_{t+1}\left[\left[Q^3\right] \cdot \vec{\chi}_t\right] \cdot \vec{\chi}_t; \tag{4.27}$$

$$[Q^3] = [Q^3] + \frac{\alpha_q \times \delta_t \times \vec{\chi}_t \otimes \vec{\chi}_{t-1} \otimes \vec{\chi}_{t-1}}{\max(1, (\sum |\vec{\chi}_t|) \times (\sum |\vec{\chi}_{t-1}|)^2)}; \tag{4.28}$$

and

$$[p^3] = [p^3] + \frac{\vec{\partial}_t \otimes \vec{\chi}_{t-1} \otimes \vec{\chi}_{t-1}}{\max(1, (\sum |\vec{\chi}_t|^2) \times (\sum |\vec{\chi}_{t-1}|)^2)}. \tag{4.29}$$

Though it is not yet relevant, to reflect the change in units, logical error will be calculated as:

$$\partial_t = \begin{cases} \alpha_p \times (\mathbb{R}\{\chi_t\}^2 - \mathbb{R}\{p(N_t|\chi)_{t-1})\} & \text{for state predictions,} \\ \alpha_\mu \times \alpha_p \times (\mathbb{R}\{\chi_t\}^2 - \mathbb{R}\{p(N_t|\chi)_{t-1}\}) \times \delta_t & \text{for policy learning;} \end{cases} \tag{4.30}$$

and action selection will be based on $\mathbb{R}\{\sqrt{p(N_{t+1}|\chi)_t}\}$.

Given the cardinalities $\#A$, $\#S$ and $\#X = \#A + \lceil log_2(\#S) \rceil$ for input sets X and S and output sets X and A, the Kerneled method will significantly reduce the number of required states at the cost of changing the order of time complexity from $O(\#A \times \#S)$ to $O(\#X \times \#X^2/2)$. CLASP's order of complexity becomes relatively smaller at $500 < \#S < 512$ and $672 < \#S$ given $\#A = 1$.[8] If the model was removed, the situation $O(\#A \times \#S) < O(\#A \times \#X^2/2)$ occurs when $14 \leq \#S \leq 16$ and $19 \leq \#S$ for $\#A = 1$, *i.e.* by removing the model, it is possible to significantly reduce the algorithms complexity for most larger state-spaces.

4.4 Fleshing out State Information

W.r.t. logical information, machine learning algorithms are designed to learn what is presented within a given set of data, either by being given a set of logical targets as in supervised learning or by identifying clusters of samples by their similar properties. Near the decision boundary dividing logical outcomes, the algorithm can still assume the probability of a point belonging to either side of the decision boundary depending on the threshold function used. However, from the standpoint of conventional logical states, it is impossible to learn about information that is not there.

Lets take the trivial case where an input space has zero points within and is being evaluated on by a simple algorithm with one binary output, *e.g.* a perceptron or modified SARSA algorithm with only one output. An algorithm cannot learn without data, but the initial weights will still establish a decision boundary, thus making the assumption that the input space is divided into

[8] A larger action space will require a larger $\#S$ value to flip the inequality but the fact that is CLASP is more compact for a sufficiently large state-space remains unchanged.

distinct sub-regions equal to the number of possible outputs. The most correct answer that can be given in such a circumstance is a prediction of ≈ 0.5, however, given a lack of additional information, this also implies that the algorithm is confident in this prediction and its relevance to the input space. In reality, it is impossible for algorithms such as SARSA and the perceptron to admit that they don't know what is supposed to happen in this search space because they are not designed to account for ambiguity caused by missing information. The significance of this trivial case becomes more apparent when it is taken as a sub-space of the complete input space. Interpolating between two sub-spaces already exists within the learning algorithm to different effects relative to the encoding method used, and can allow the algorithm to make valid conclusions with certainty. However, the problem becomes more apparent when an algorithm extrapolates to edge cases without supporting evidence — *e.g.* having no surrounding sub-spaces containing sample points — and yet claims the same degree of certainty regarding its output.

Learning about information that is absent from the data, is not as simple as taking the inverse of the logical value because a logical zero embodies two fundamental concepts depending on circumstance — either 'is not' or 'don't know and/or don't care.' To learn about both of these aspects will be greatly beneficial to decision making, but that will only be the case if one can distinguish between the two (see chapter 2.1). To resolve this limitation, it will be necessary to assess what can extracted for a given state then consider how to arrange the information.

4.4.1 Traces: Intangible Information

One of the ironic aspects of control systems is that observed state information will almost always be assumed to be correct. In contrast, if the predictions contradict the presented state, they will be assumed to be wrong. Such assumptions, though often true, can be rather bold when considering real world problems where sensor information can be fallible and very localized. Though the prediction cannot be claimed as being entirely correct, it would be improper to fully disregard what the Learning Algorithm (LA) will have

concluded to be correct after having learned for a period of time. The gradual learning process reduces the likelihood of erroneous predictions preventing convergence, but does not solve the problem of having incomplete information potentially propagate from one input state to the next. To prevent a build up of errors, a ratio of mixing predictions of the current state with its observations should be implemented. This will also shift a portion of the learning rate away from the designated policy learning parameter, but this is also acceptable. As the proposal involves modifying the input state space, the modified space will be defined as:

$$\tilde{\chi}_t = \zeta_p \times \chi_t + (1 - \zeta_p) \times \mathbb{R}\{\sqrt{p(\eta_t|\tilde{\chi})_{t-1}}\}. \tag{4.31}$$

Equation (4.31) will combine the logical observation and prediction under the assumption that the output units match the input units. As both are signal amplitudes, there should be no issue. This form of mixing can be done because both sources represent the exact same source; the difference being that one is based on non-omniscient observations, while the other is based on prior experience. By replacing χ_t with $\tilde{\chi}_t$, the process of re-learning in non-stationary environments will be softened. This method is not unreasonable as even the human brain — that CLASP seeks to emulate — uses predictions and observations together. If $\zeta_p \neq 1$, it will be reasonable to claim:

Claim 4.2. Partial Reliance on Predictions — *The agent should be able to rely on predictions and observations with partial independence. In other words: for a short period after the loss of observability, the agent should still be able to act on the expectation of what the current state should have been, giving it time to shift its focus to states that are still reliable [33].*

4.4.2 Traces: Past and Present

As a compliment to the current state information χ_t, a trace of the historical activity will also be kept. Adding traces of past states will serve a few purposes:

- generating a state transition vector,
- determining the 'age' of an active state, and

- providing a record of past active states.

For some problems, having knowledge of the path used to get to the current state can be as important as the current state itself [29]. This applies to many sequential and time-series problems, including k-gram prediction. Something else to note is that: if the sample frequency for the state-space is considered very low and based on accumulated values since the last observation, a portion of $\mathbb{R}\{\tilde{\chi}_t\}$ may already be expired historical data. With these considerations, the proposed candidate solution will be:[9]

$$\chi_{t-\tilde{m}-0.5} = \beta_h \times (\beta_p \times \mathbb{R}\{\tilde{\chi}_t\} + (1-\beta_p) \times \mathbb{R}\{\tilde{\chi}_{t-1}\}) + (1-\beta_h) \times \chi_{t-\tilde{m}-1.5}, \quad (4.32)$$

and its integration with the prediction information will be done via:

$$\tilde{\chi}_{t-\tilde{m}-0.5} = \zeta_h \times \chi_{t-\tilde{m}-0.5} + (1-\zeta_h) \times \mathbb{I}\{\sqrt{p(\eta_t|\tilde{\chi})_{t-1}}\}. \quad (4.33)$$

In total χ_t is the fully active signal, $\mathbb{R}\{\sqrt{p(\eta_t|\tilde{\chi})_{t-1}}\}$ is the excitatory signal, $\mathbb{I}\{\sqrt{p(\eta_t|\tilde{\chi})_{t-1}}\}$ is the inhibitory signal, and $\chi_{t-\tilde{m}-0.5}$ is the refractory/recovery signal, all of which having a strong resemblance to the respective ion concentration roles found in the biological neuron (see chapter 2.3).

As a state's duration of activity typically causes a depreciation in positive signal strength, it is tempting to use:

$$\tilde{\tilde{\chi}}_t = \tilde{\chi}_t - \tilde{\chi}_{t-\tilde{m}-0.5}, \quad (4.34)$$

however, this equation cannot properly express conflicts between observation and expectation. Therefore, it will be more practical to use:

$$\tilde{\tilde{\chi}}_t = \frac{\tilde{\chi}_t + \hat{\imath}\tilde{\chi}_{t-\tilde{m}-0.5}}{\max\left(1, |\tilde{\chi}_t + \hat{\imath}\tilde{\chi}_{t-\tilde{m}-0.5}|\right)}, \quad (4.35)$$

while knowing that the historical information negates positive signals when using $\tilde{\tilde{\chi}}_t^2$. Setting the limit of the magnitudes upper bound to 1 is partially to reduce the risk of causing instability while also reducing the scale of the vector components in a way that will not adversely affect the phase. The phase of

[9]It should be noted that, the bias state's trace value is fixed to 1, as it can be assumed that it has been active since $t = -\infty$; however, not fixing the bias state may also have its use in determining the neurons age.

the $\tilde{\vec{\chi}}_t$ is important when the algorithm must decide how to resolve conflicting information. With the inclusion of traces, the new algorithm's policy and quality equations will be:[10]

$$\vec{\eta}_{t+1} = \left[[p^3] \cdot \tilde{\vec{\chi}}_t \right] \cdot \tilde{\vec{\chi}}_t; \tag{4.36}$$

$$q(\eta, \chi)_{t+0.5} = \sqrt{p(\eta_{t+1}|\tilde{\vec{\chi}}_t)_t} \cdot \left[[Q^3] \cdot \tilde{\vec{\chi}}_t \right] \cdot \tilde{\vec{\chi}}_t; \tag{4.37}$$

$$[Q^3] = [Q^3] + \frac{\alpha_q \times \delta_t \times \vec{\chi}_t \otimes \tilde{\vec{\chi}}_{t-1} \otimes \tilde{\vec{\chi}}_{t-1}}{\max(1, \left(\sum |\tilde{\vec{\chi}}_t|^2 \right) \times \left(\sum |\tilde{\vec{\chi}}_{t-1}| \right)^2)}; \tag{4.38}$$

and

$$[p^3] = [p^3] + \frac{\vec{\delta}_t \otimes \tilde{\vec{\chi}}_{t-1} \otimes \tilde{\vec{\chi}}_{t-1}}{\max(1, \left(\sum |\tilde{\vec{\chi}}_t|^2 \right) \times \left(\sum |\tilde{\vec{\chi}}_{t-1}| \right)^2)}. \tag{4.39}$$

The maximum value for a state being dependent on duration of activation will enable conditions where persisting states have slightly less emphasis on the policy. A point of caution regarding $\tilde{\vec{\chi}}_t$: though it will provide additional state information, it is also possible for it to become a source of garbage information. Granted, the configuration of $\tilde{\vec{\chi}}_t$ will allow for χ_t to fully suppress the other sources of information if the PSO determines it is better.

4.4.3 Bifurcation: Affirming and Denial States

To lend further justification for complex state information, it can be divided into two dimensions within its respective logic space, *i.e.* affirmation and denial. These attributes of a state will lay on the real and imaginary axes — extracted via $\mathbb{R}\{\square\}$ and $\mathbb{I}\{\square\}$ respectively. Unfortunately, denial states are more abstruse due to their inherently non-observable nature. It is important to note that the assumption:

$$\mathbb{I}\{\chi_t\} = 1 - \mathbb{R}\{\chi_t\}, \tag{4.40}$$

[10]The usage of $\vec{\eta}_{t+1}$ in the quality prediction was replaced with $\sqrt{p(\eta_{t+1}|\tilde{\chi})_t}$ as a consideration to limit the effects of exaggerated values in $\vec{\eta}_{t+1}$. The primary difference is that the former would be based on ion activity while the latter is based on membrane activity caused by stimuli.

is erroneous as the denial state distinctly claims that something is not only absent from observation but also not present within the state's scope. Instead, the negation of the affirmation only assert that something is not observed and/or predicted. The difference between negation and denial becomes apparent when affirmation and denial are not dualities. In the event that both denial and affirmation occur as a result of a logical operation, it distinctly admits the logic does not know. Similarly, if neither affirmation nor denial occur, it distinctly admits the logic does not care. For conventional logic, both of these instances would be represented by $p(y) = 0.5$.

Non-dual state logic can be expected to occur when two supposedly dual options are unable to properly cover the entire logic space or are somewhat independent of each other, *e.g.* 'hot or cold' lacks intermediate temperatures and 'happy or sad' does not include 'happy and sad'. For traditional logic, it would be required to provide additional states which compartmentalize the corresponding analog range, *i.e.* depending on how states are presented, some form of problem-specific pre-processing of logical states may be required (see chapter 2.1).

Denial states must be independent from their corresponding observable dual, but simultaneously have some method of opposing the affirmation state. Complex numbers will allow the state to achieve independence between affirmation and denial, mutability without removal of conflicting information, and flexibility while maintaining the opposing nature of dualities (see chapter 2.2). The new interpretation of logic, asserts the complete expression of state x can be given minimally by:

$$x = \mathbb{R}\{x\} + \hat{\imath}\mathbb{I}\{x\}. \tag{4.41}$$

The logical information regarding affirmation is unchanged, however, it will be necessary to develop the agent's ability to actively and meaningfully deny statements. Looking at equation (4.33), the first element to consider will be the historical trace $\chi_{t-\tilde{m}-0.5}$. The implication of this component is that the agent should not care as much about states that will be active more frequently or for longer durations. This assumption follows the refractory

characteristic found with Cl^- and K^+ in matured cells: functioning as a form of inhibitory historical information w.r.t. Na^+ and Ca^{2+} which compose $\tilde{\chi}$ in equation (4.31). $\mathbb{I}\{\sqrt{min(0, max(-1, \mathbb{R}\{\eta_t\}))}\}$ is the result of learning what will cause exceptions in the observed states or more formally the declaration of denial. This element is also associated with Cl^- and K^+ and similarly inhibits positive activation; however, it is distinctly different from the historical trace as it will be the result of prior knowledge in the form of a logical induction of unobservable facts/speculations. The consequence of this layout will be that, if there is no observable state value, the agent will be able to rightly deny the associated state, and if there is an observation, it can only state that it does not know the truth of said statement, *i.e.* it could have been a faulty observation or an erroneous prediction. It must be noted that not knowing does not mean that the agent will not be able to come to a decision based on its indeterminate knowledge; but that it must develop a preference — by changing the weight phase — in the face of uncertainty.

Another aspect that must be addressed is that the imaginary state value does not have a target to pursue and will thus lack a method of calculating error. Applying a direct update method will not be possible as the learning of what is not there can only be accomplished as a natural consequence of learning the exceptions for observations. However, the update method and probability value must be adjusted to properly handle the learning of complex valued weights:

$$\tilde{\eta}_{t+1} = \frac{(\mathbb{R}\{\eta_{t+1}\} - 0.5) \times \lambda_h}{1 - \lambda_h} + 0.5; \tag{4.42}$$

$$\mathbb{R}\{p(N_{t+1}|\tilde{\tilde{\chi}}_t)_t\} = \begin{cases} -1 & \text{if } -1 \geq \mathbb{R}\{\eta_{t+1}\}, \\ \mathbb{R}\{\eta_{t+1}\} & \text{if } 0 > \mathbb{R}\{\eta_{t+1}\} > -1, \\ 0 & \text{if } \lambda_p \geq \mathbb{R}\{\eta_{t+1}\} \geq 0 \text{ or } 0 \geq \tilde{\eta}_{t+1}, \\ 0 & \text{if } 0.5 \geq \mathbb{R}\{\eta_{t+1}\} \geq \lambda_p \text{ and } 0 \geq \tilde{\eta}_{t+1}, \\ 1 & \text{if } \mathbb{R}\{\eta_{t+1}\} \geq 1 - \lambda_p \text{ or } \tilde{\eta}_{t+1} \geq 1, \\ \tilde{\eta}_{t+1} & \text{otherwise;} \end{cases} \tag{4.43}$$

$$\mathbb{I}\{p(N_{t+1}|\tilde{\tilde{\chi}}_t)_t\} = \begin{cases} 1 & \text{if } \mathbb{I}\{\eta_{t+1}\} \geq 1, \\ \mathbb{I}\{\eta_{t+1}\} & \text{if } 1 > \mathbb{I}\{\eta_{t+1}\} > 0, \\ 0 & \text{if } 0 \geq \mathbb{I}\{\eta_{t+1}\}; \end{cases} \tag{4.44}$$

$$p(N_{t+1}|\tilde{\tilde{\chi}})_t = \frac{\mathbb{R}\{p(N_{t+1}|\tilde{\tilde{\chi}})_t\} + \hat{\imath}\mathbb{I}\{p(N_{t+1}|\tilde{\tilde{\chi}})_t\}}{\max\left(1, |\mathbb{R}\{p(N_{t+1}|\tilde{\tilde{\chi}})_t\} + \hat{\imath}\mathbb{I}\{p(N_{t+1}|\tilde{\tilde{\chi}})_t\}|\right)}; \tag{4.45}$$

and

$$\partial_t = \begin{cases} \alpha_p \times \left(\mathbb{R}\{\tilde{\tilde{\chi}}_t\}^2 - \max\left(0, \mathbb{R}\{p(N_t|\tilde{\tilde{\chi}})_{t-1}\}\right)\right) & \text{for state predictions,} \\ \alpha_\mu \times \left(\mathbb{R}\{\tilde{\tilde{\chi}}_t\}^2 - \max\left(0, \mathbb{R}\{p(N_t|\tilde{\tilde{\chi}})_{t-1}\}\right)\right) \times \delta_t & \text{for policy learning.} \end{cases} \tag{4.46}$$

These new equations will allow a bounded form of the denial predictions, affirming predictions, and historical traces to carry into the prediction and learning error. An additional item included will be the piece-wise linear threshold function defined by equations (4.44) and (4.45). The ability to optimize the threshold function will allow the PSO to include a measure of tolerance for build-up of historical traces should it be desired. Parameter λ_h will set the sensitivity of the neuron to stimuli between the thresholds determined primarily by λ_p and $1 - \lambda_p$.

4.5 Logical Error

The logical error function used for equation (4.46) is simple and effective, however, both logical and distance error have their shortcomings. Conventional logical error:

$$\vec{\partial}_t^{logc} = \vec{\chi}_t - \vec{p}(N_{t-0.5}|\vec{\chi})_{t-1} \tag{4.47}$$

has a tolerance for exaggerations in logical statements due to the thresholded nature of $p(N_{t-0.5}|\vec{\chi})$, thus making it very stable; but it does not prevent weights from growing excessively which can cause issues with divergence. Distance error:

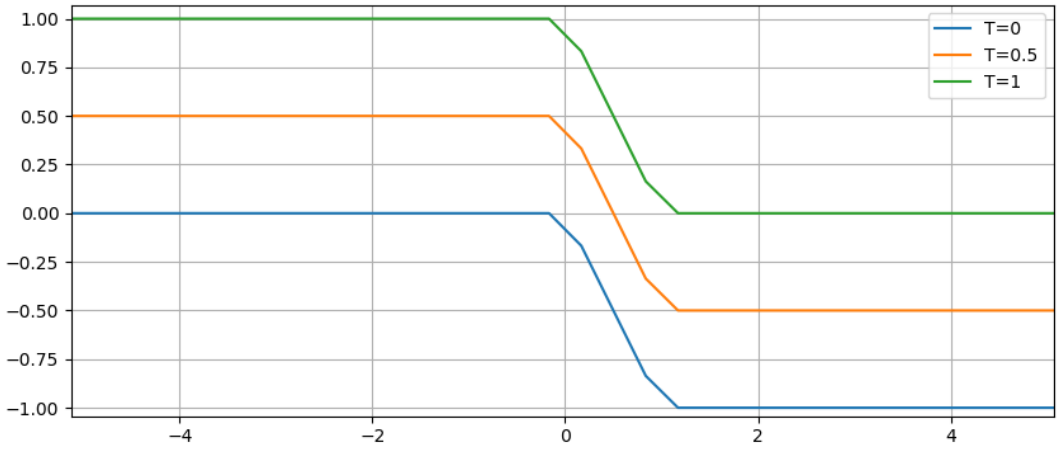
$$\vec{\partial}_t^{dist} = \vec{\chi}_t - \vec{\eta}_t \tag{4.48}$$

has no tolerance for exaggerations, demanding exact values for every prediction, thus making it more unstable for highly stochastic problems. It also severely restricts the algorithm's learning ability for points very close to the decision boundary. The logical error will always be a safe, albeit potentially unstable default for learning, however, a combination of these two methods is expected to not only allow the agent to learn about observable states, but

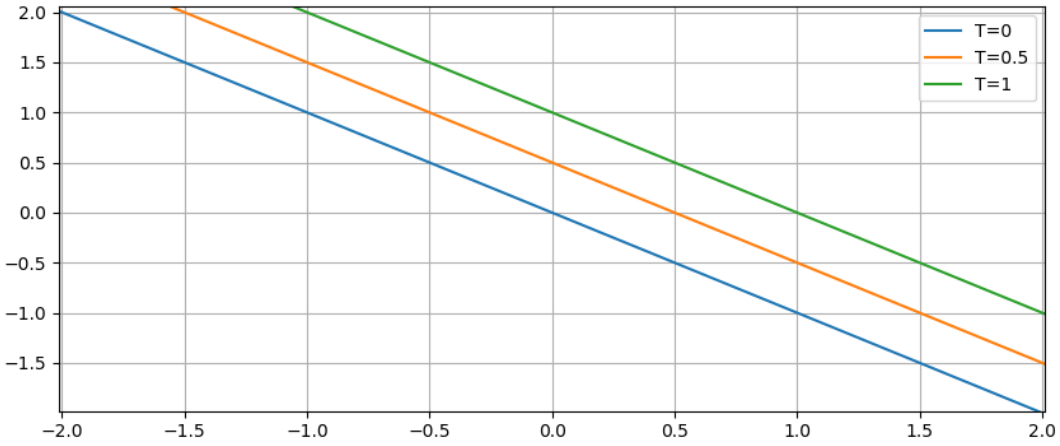
also allow it to exploit logical exaggerations as a means to gain additional information regarding unobserved states. The initial proposal will be to restrict the error such that it forms a slacked region:

$$\partial_{raw} = \begin{cases} \tau + \chi_t - \eta_t & \text{If } \chi_t \geq 1 \text{ and } \eta_t > \tau + \chi_t, \\ 0 & \text{If } \chi_t \geq 1 \text{ and } \tau + \chi_t \geq \eta_t \geq \chi_t, \\ \chi_t - \eta_t & \text{If } \chi_t \geq 0 \text{ and } \chi_t > \eta_t, \\ \chi_t - \eta_t & \text{If } 1 > \chi_t \text{ and } \eta_t > 0, \\ 0 & \text{If } 0 \geq \chi_t \text{ and } 0 \geq \eta_t \geq -1 - \tau, \\ \chi_t - \eta_t - 1 - \tau & \text{If } 0 \geq \chi_t \text{ and } \chi_t - 1 - \tau > \eta_t. \end{cases} \tag{4.49}$$

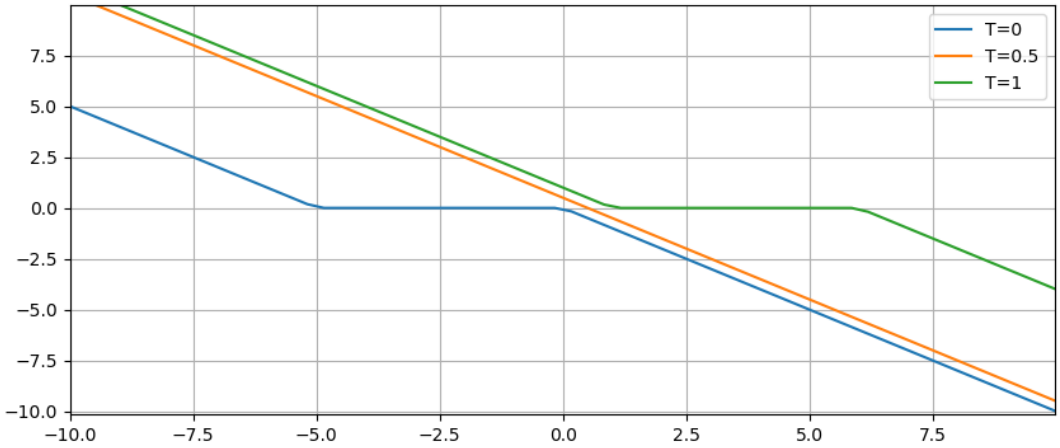
This set of conditional statements will allow for strict learning when misclassification occurs or when η_t exceeds the target χ_t by more than τ which is reminiscent of equation (4.48). If the prediction is correct and within the tolerance specified by τ, the error will be 0, producing a result similar to equation (4.47). This method's more notable characteristic is that it will create three distinct regions whose edges run parallel to the decision boundary: the outermost region spanning $(\infty, \tau + \chi_t]$ and $(-\infty, \tau - 1]$ on either side affects weights that have grown excessively large, *i.e.* imposing a material/resource constraint on the neurons size; the intermediate region encloses values that are considered reasonable in magnitude and will thus be handled according to the rules of logical error; lastly, the region surrounding the decision boundary spanning to either side. The two logical error methods are virtually identical for the boundary region, however, it is also a region where the solution will likely remain stochastic or simply be too vague to make a definite decision given the scale of the input space. This region is where learning will not be fully supported due to the lack of sufficient resources to fine-tune the results, *e.g.* τ or the number of useful inputs need to be increased. This approach is similar in principle to the one used in Support Vector Machines (SVMs) but also distinctly different. The support vectors in SVMs lay relatively close to the boundary, but the slacked error method produces supporting points laying at the outermost regions of the problem space relative to the decision boundary [11], [28].



(a)



(b)



(c)

Figure 4.3: Relative to the associated target value $T \in \{0, 0.5, 1\}$ and the unbounded prediction value (*i.e.* the x-axis): (a) Probabilistic error, (b) Value error, and (c) Slacked error with $\tau = 5$.

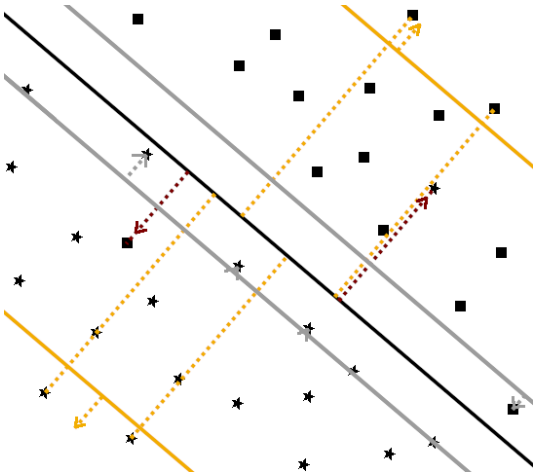


Figure 4.4: The effects of slacked error is demonstrated by the yellow and gray lines. Assuming correct classification, points beyond the yellow line will cause the respective weights to shrink while points between the black decision boundary and the gray line will attempt to increase the weight value. For incorrect classification, the result is based on distance error.

In reference to figure 4.4, the solid yellow line, representative of the threshold τ, will restrict the maximum cumulative weight size — logical exaggeration — w.r.t. correctly classified data by shifting all the lines slightly in the direction of the stimulus point, *i.e.* reducing the distance between the respective point and the decision boundary (the black line) in the weight space. The region between the yellow and gray line serves as the slack permitted for exaggerated activation. The exhibited behavior can also be regarded from a biological perspective of the membrane as a range of elastic deformation, *i.e.* correctly classified points that lay within do not affect the weight values [18]. The region between the two gray lines serves as the non-deterministic area. Correctly classified data points within this region will attempt to increase their respective weights, thus reducing the distance between the threshold τ and the decision boundary line w.r.t. the input space. Though poorly classified data does not fully oppose the effects of τ, combining said effects with those of misclassified data is sufficient for ensuring a proper decision boundary is formed. To give the analysis from another perspective, figure 4.3 shows the different error curves for a given set of targets and a range of 'predicted' values. It can be seen from this that, if $\tau = 0$, slacked and value error will be identical

while, if $\tau = \infty$, slacked and prediction error will become identical for correctly classified data.

4.5.1 Bounded Errors

Before testing this new error method, it will be preferable to more closely match slacked error with logical error, *i.e.* restricting the error range to $(-1, 1)$ or $[-1, 1]$ is expected to reduce the likelihood of divergence than a potentially unbounded error. This gives a better guarantee of maintaining the stability w.r.t. error values inherent to the original logical error method. W.r.t. biological neurons, this is establishing a limit to the degree of plasticity a cell will be able to employ per learning cycle. Additionally, by bounding the error, infrequent large errors will not be able to influence the learning process too strongly relative to smaller but more frequent errors. This minimizes the likelihood of getting a large number of misclassified data points near the decision boundary in a futile attempt to rectify the misclassification of potential outliers.[11] Table 4.1 shows the two ways of classifying the concerned erroneous data. Though one could further divide these cases into those that will be better resolved with further learning and those that will not, such cases would require information about the data set as a whole and is not within the scope of this research.

Errors that occur often but with a relatively short distance to the decision boundary can be considered as small frequent errors. If such errors are present, it would usually imply the learning algorithm is close to convergence but still requires more learning. If the small errors occur relatively intermittently, this would suggest that the algorithm has converged to a reasonable but non-perfect solution. Such cases will tend to have a balanced amount of cumulative error from each side of the decision boundary. Other cases where small errors would likely occur also exist: if the training data poorly represents the data along the true decision boundary, and/or if the algorithm will not or cannot balance the distance between the training data that neighbors the decision boundary.

[11] If the grossly miss-classified point is relevant, it is more likely to be a problem of lacking sufficient resources.

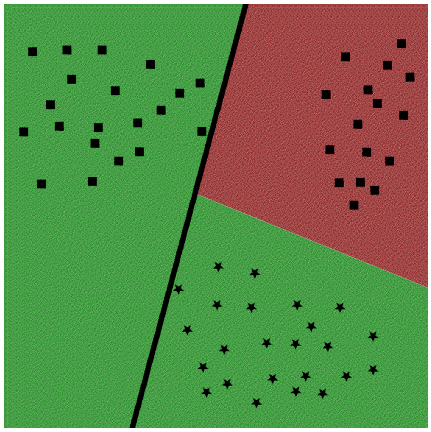
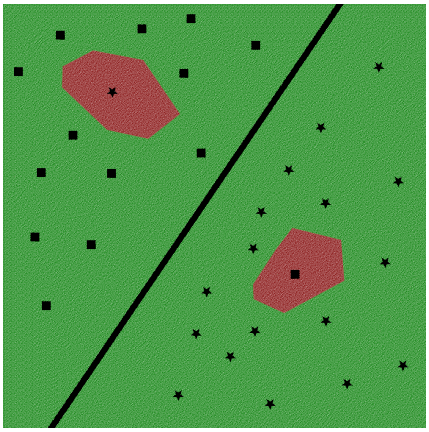
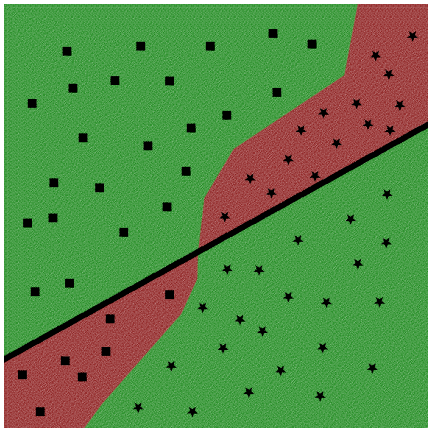
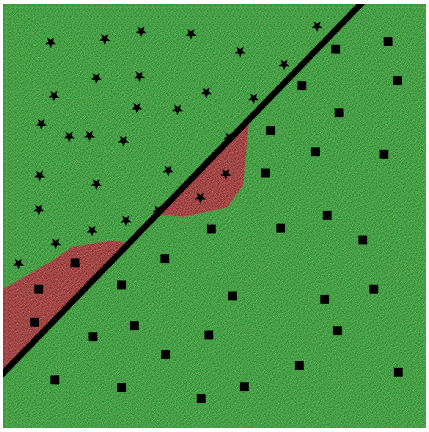
	Frequent	Intermittent
Large		
Small		

Table 4.1: The four categories for which errors can be categorized into.

Similarly to the small errors, if a large error occurs relatively frequently, it would suggest the algorithm has yet to converge. If it has converged, then it may be a case where the problem is non-separable to a severe degree. If a large error occurs infrequently, there are a number of likely system causes: faulty labeling, erroneous information, and/or a result that occurs due to a highly non-linear but mostly separable problem space. For infrequent errors, they could be outliers that should be ignored or genuine points that require additional resources to be properly represented, *i.e.* exceptions to the currently learned rules. Regardless, large infrequent errors should not be allowed to shift the entire decision boundary too far. It would be advisable to establish a new boundary instead, using excess resources should they be available. To achieve a bounded error, equation (4.49) will be replaced with:

$$\partial_{raw} = \begin{cases} sig^{+}\left(z + x - y, a, b, c\right) & \text{If } x \geq 1 \text{ and } y > z + x, \\ 0 & \text{If } x \geq 1 \text{ and } \tau + x \geq y \geq x, \\ sig^{+}\left(x - y, a, b, c\right) & \text{If } x \geq 0 \text{ and } x > y, \\ sig^{+}\left(x - y, a, b, c\right) & \text{If } 1 > x \text{ and } y > x, \\ 0 & \text{If } 0 \geq x \text{ and } x \geq y \geq x - 1 - z, \\ sig^{+}\left(x - y - 1 - z, a, b, c\right) & \text{If } 0 \geq x \text{ and } x - 1 - z > y, \end{cases} \tag{4.50}$$

where:

$$\begin{aligned} x &= \mathbb{R}\{\tilde{\tilde{\chi}}_t\}^2 \\ y &= \mathbb{R}\{\eta_t\} \\ z &= \tau \times \#X^2 \\ b &= 1 \end{aligned}$$

The curve for $sig^{+}(x, a, b, c)$ is comparable to the traditional sigmoid functions when $a = b = 1$ and $c > 0$ (see chapter 2.4). Fixing $b = 1$ serves to reduce parameter redundancy as a, b, and c are inter-related but the roles of a and c are more distinct. The sigmoid function will largely influence the effective learning rate which would otherwise be solely regulated by α_μ and α_p. An approximation for the effective logical learning rate:

$$\alpha = \frac{|\partial_t| + \epsilon}{|\partial_t^{dist}| + \epsilon}, \tag{4.51}$$

where ∂_t is the final error value after all learning rates are applied and ∂_t^{dist} is the distance error calculated from equation (4.48).[12] Though it is not relevant to the learning process, it gives a useful measure of the per iteration learning limitation imposed on the algorithm via the error method. The final equation for calculating ∂_t will be given as:

$$\partial_t = \begin{cases} \alpha_p \times \partial_{raw} & \text{for state predictions,} \\ \alpha_\mu \times \partial_{raw} \times \delta_t & \text{for policy learning.} \end{cases} \tag{4.52}$$

[12] ϵ is an infinitely small number included to prevent instances of $0/0 = NaN$ when it is sufficient for $0/0 = x/x = 1$.

Chapter 5

System Model

In this chapter, the details of the simulator setup starting with the environment model followed by the agent components will be covered. How these models are configured largely determine the problem difficultly, practicality, as well as the extensibility outside of research. The section on agent models goes into moderate detail about preprocessing of state information and post-processing of probabilistic action distributions. While making changes to the algorithm, it will be worth while to consider the effects of sub-optimal real world conditions. In reality, it is not uncommon for there to be limitations and external factors forcing the configuration to be sub-optimal, *e.g.* large data, noisy/faulty data, and delays. After clarifying the details of the system model, a description of the simulation setup itself will be in order, followed by what additional information will be recorded as methods of comparison, and how the performance/score will calculated for later evaluation.

5.1 World/Problem Models

The problem of choice is character prediction. The reasons why this type of problem is of interest are that:

- there is a wide variety of raw text to be used as data sets,
- the state-space is moderately large given the number of valid ASCII characters and symbols available,
- there are several options for encoding and decoding,

- there are two distinct ways of evaluating performance (*i.e.* hit rate and bit-wise accuracy), and
- unigram prediction from a single character is largely stochastic or highly non-stationary.

A more popular form of this type of problem is at the level of predicting the next word using a length of words following behind, *i.e.* an n-gram formed with words, to build sentences and paragraphs from an initial phrase. Word prediction is easier because the order follows the rules for grammatical structure of the given language, but the set of all states is also large, *i.e.* there are as many unique states as the number of word n-grams in the language used [8], [38]. Character level n-grams are more difficult unless the word construction also has a set of rules, but it has relatively fewer unique states. English does have a few rules for spelling, but it also has a lot of exceptions, making it difficult to predict the next character. As the number of characters/words used for prediction increases so does the uniqueness of the n-gram, thus decreasing the number of likely next characters/words [19]. Given that each character represents its own state, the unigram problem can also be viewed as a highly connected graph with as many edges leading to the next state as there are states. The catch being that there is only one correct edge with a positive reward, *i.e.* the one corresponding to the destination state.

The highly non-stationary problem of unigram character prediction is used in this research to test the ability of the LA in what can be considered a very difficult but compact problem. It is also expected to better demonstrate the effects of internally tracking historical information and filtering out irrelevant states than if it were applied to a simple grid-world problem. Character prediction with binary state information is also likely to cause the algorithm to become unstable if it is not constructed with learning stability in mind. By demonstrating that the algorithm can not only learn but also remain stable in a relatively chaotic environment, it will likely be more easily accepted in real-world applications. In English text, it is not easy to guess the next character when only presented with the current character, especially when the binary

m	c_0	c_1	τ_1	c_2	τ_2	$v_{lim}^{\%}$
0.537582	0.768016	0.999995	0.629405	0.999383	0.872157	0.418376

Table 5.1: Chosen PSO parameters.

encoding does not have elements that represent shared features within the character set. This is especially so when you do not know any of the higher level rules, *e.g.* cat, catapult, caterpillar, catfish, and catamaran have the same first 3 letters, but the fourth letter would likely be easier to predict if some knowledge of the context was present. For the encoder, some elements such as upper/lower-case, vowel/consonant, and symbols will be embedded into the binary state representation. This is akin to rearranging the order of characters such that their hexadecimal representations are more convenient for our learning problem than what is given using ASCII.

5.1.1 In-Sample: Parameter Optimization

To ensure a satisfactory combination of parameters are selected for the Learning Algorithm (LA), a Dimension-wise Particle Swarm Optimization (PSO) algorithm is used with the parameters shown in table 5.1. These parameters were chosen from the results of a previous work which demonstrated that they were effective in moderately high dimensional problems of approximately 9 dimensions, some of which being of much greater difficulty than others [31]. For the in-sample data, the Temporal Difference Reinforcement Learning (TDRL) algorithm will predict the next character's binary value with the same configuration that would be used in the out-of-sample problem. The first 987 characters — including whitespace — from the foreword of *Lord of the Ring — Fellowship of the Ring* book:

> This tale grew in the telling, until it became a history of the Great War of the Ring and included many glimpses of the yet more ancient history that preceded it. It was begun soon after The Hobbit was written and before its publication in 1937; but I did not go on with this sequel, for I wished first to complete and set in order the

> mythology and legends of the Elder Days, which had then been taking shape for some years. I desired to do this for my own satisfaction, and I had little hope that other people would be interested in this work, especially since it was primarily linguistic in inspiration and was begun in order to provide the necessary background of 'history' for Elvish tongues.
>
> When those whose advice and opinion I sought corrected little hope to no hope, I went back to the sequel, encouraged by requests from readers for more information concerning hobbits and their adventures. But the story was drawn irresistibly towards the older world, and became an account, [22]

This text is expected to be long enough for the LA to show a notable difference in performance as its parameters are varied, yet short enough to prevent the PSO from requiring excessive amounts of time to finish. So long as the training and test scripts are different enough that the samples do not excessively overlap and a bag of n-grams containing all sample data from both sets can be generated, the choice of source data is not limited to English text [8], [19], [38].

5.1.2 Out-of-Sample

To test the optimized algorithms, the Shakespearean play titled '*All's Well That Ends Well*' has been selected for the out-of-sample evaluation [2]. After making minor adjustments in format, this play produces 135,795 raw text data samples. Due to the length of the data set, it will be evaluated thrice to ensure sufficient time for convergence (totaling 407,385 characters).

5.2 Agent Models

For this research, the problem will be considered to be an isolated system composed of multiple elements — characters, rules, topics, semantics, *etc.* known or otherwise — with select modes of interaction between the environment and

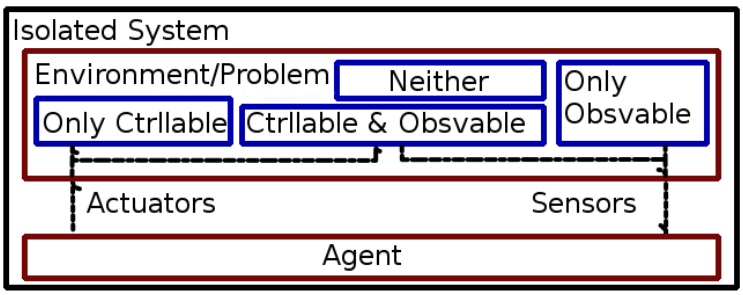


Figure 5.1: The basic system model

agent as shown in figure 5.1. Opening up the general interpretation of an agent, potential internal components commonly found in control systems can be filled in (see figure 5.2). Should these elements be relevant, they will be likely sources of noise, delay, and/or data manipulation occurring independently of the problem. The agent will not be considered to have actuators as the problem set will be strictly observable. The sensor data will be considered as the raw character observed.[1]

5.2.1 Reward

Based on the type of the problem, the reward is considered a result of preprocessing looped back action information with the most recent sensor data. The reward function takes the character being passed as the next input performing a bitwise comparison with the predicted character, summing the resulting binary sequence, and multiplies it by -1. The reward will be placed into the first of two reward channels. If the characters match, a '+1' will be passed to the second channel.[2] With this configuration, the first channel's value relays how close the agent is to getting the predictions right, and the second channel relays the agent's ability to get the right answer. This setup also prevents the LA from encountering large regions that may lack a reward stimulus.

[1]It should also be noted that sensors can be assumed to be infallible, of infinite resolution, and without delay, *i.e.* the data from the environment is accurate and precise.

[2]These channels are combined with the logical information channel in figure 5.2 to reduce clutter and branch off within the AI block, pointing towards the learning block that needs the rewards. If the learning algorithm used is intended to operate with only one reward value, the results of the two channels will be summed during the LAs execution.

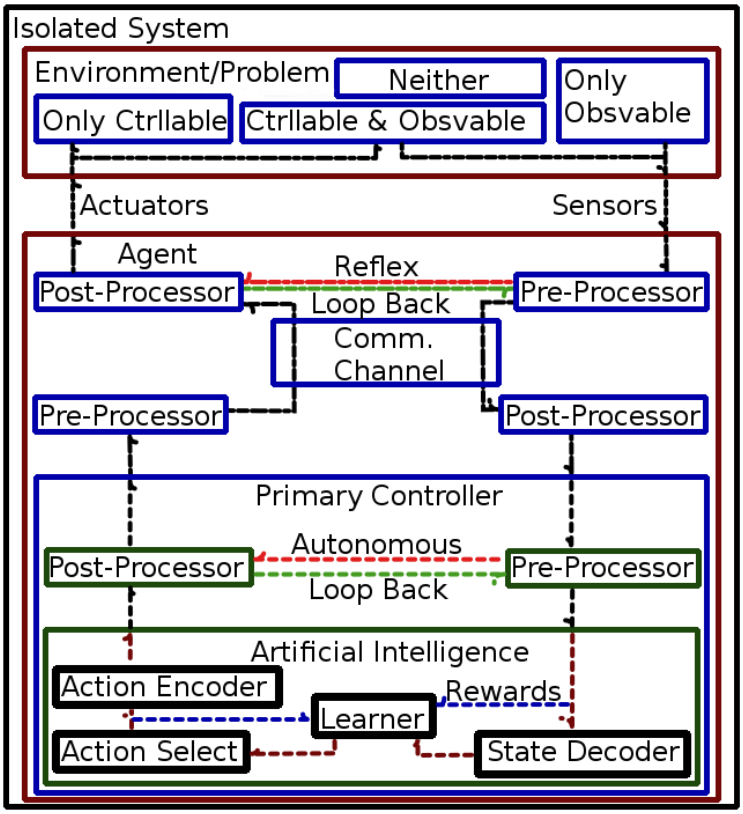


Figure 5.2: The system model, including a complete layout of the agent's general components and a stereotypical layout for the AI.

5.2.2 Transmission Channels

Though the communication channels, shown in figure 5.2 and labeled as 'Comm. Channel,' will not be directly evaluated in this thesis, the problems associated with transmission channels are probably one of the most significant factors present in real world problems. The major issues with communication channels include: delay (a known cause of instability), packet loss and corruption, and limited bandwidth [5], [14]. The practicality of an algorithm for real world problems will often be heavily influenced by these factors. Though these will be beyond the scope of this research, they will still be acknowledged as hurdles that should be examined before promoting a new algorithm for implementation outside the lab.

5.2.3 Artificial Intelligence

To add clarity to the scope of this research, the state decoder, action encoder, action selection method, and Learning Algorithm (LA) will be considered as entirely separate from each other.[3] Given the large variety of LAs available and the veritable number of combinations that can be made with available encoders, decoders, and selection methods, the scope of this research has been limited to only comparing with the Temporal Difference Reinforcement Learning (TDRL) algorithm SARSA (State-Action-Reward-State-Action). The Linear SARSA with an Actor-Critic implementation will be desirable because of its close alignment with the Bellman Equation. SARSA will also serve as a good starting point and benchmark for making changes and evaluating performance improvement.

Decoding and Encoding

Decoding of data into logical information involves translating input values into a series/vector of logical values that can be more easily processed in the next step, *i.e.* where each resulting element is in the range $[0, 1]$ or set $\{0, 1\}$. The common method for TDRL is to use decoders that have a fixed number of

[3]Everything will be defined, as much as possible, from the perspective of the LA as a standard, thus state information is decoded and actions are encoded.

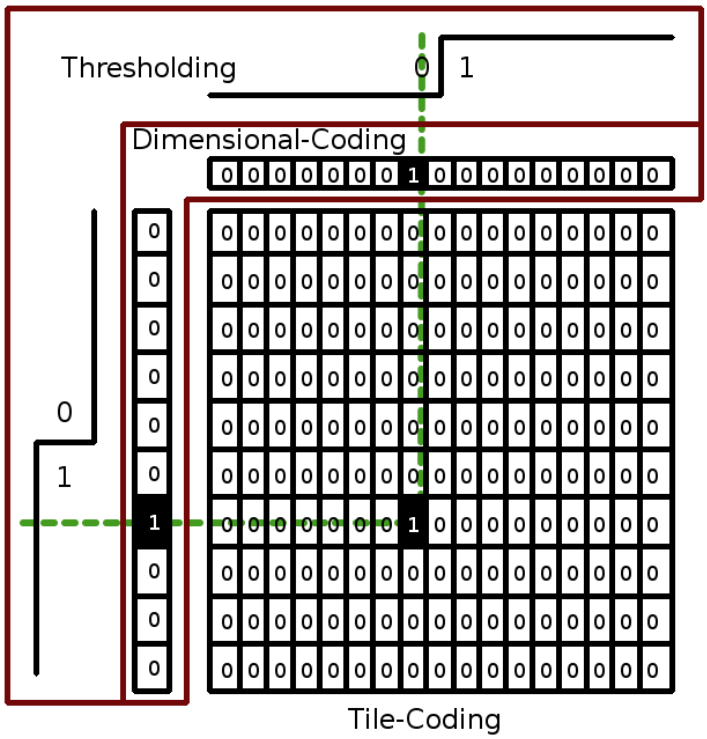


Figure 5.3: A visual comparison of three different discrete gray-coding decoding methods. For clarity: this can also be translated to fuzzy decoding methods.

active states among a group that represents the entire state-space, *i.e.* gray-coding. This may be suitable for small scale operations, however, for large scales which may require an exorbitant number of states to represent the entire state-space, gray-coding methods may no longer be cost-effective.[4] Other problems that can arise with a larger state space include: an increased number of dead states, *i.e.* ones that cannot be explored properly if at all; a lack interpolation between similar states; and an increased likelihood to encounter under-explored regions after convergence on frequently seen states. Gray-code methods should not be discredited as they have their own strengths as well — the state-space learning problem becomes highly linearized and is often more stable; however, in some extreme circumstances it may not be practical to have 1,114,111 or more input states.[5] The modifications will mostly work around binary-coding methods, making them more compact than the gray-code alternatives. Binary-coding is also more in line with how the brain functions, because different amounts of information can be conveyed at different times, and there is no fixed number of active neurons at any given time. The primary demerits to binary-code methods are that the number of active states will not be fixed and any given state value will have a different interpretation based on which other states are also active, *i.e.* learning is less stable and more likely to be highly non-orthogonal in nature. If there is any consolation, it would be that, if the learning algorithm can perform well with such a decoder, it would likely also have some degree of stability when data corruption and partial data loss are possible (see section 5.2.2). Strictly considering the character space, for the number of states $\#S$ that will be available, the largest representable value is found by:

$$K_{Gray} = \#S \tag{5.1}$$

[4]There are ways of making the code more efficient in some algorithms over others by exploiting certain aspects such as indexing, however, this research will not be delving into hardware optimizations. Different platforms may also largely mitigate the requirements for processing time — graphic processing units being one — but the overall increase in memory requirements remains.

[5]This is the number of unique characters for Unicode based on Python3.6's system reading as of July 2018. There are still other languages that have more characters than this, and having multiple languages would only cause it to grow exponentially.

and

$$K_{Binary} = 2^{\#S-1}. \tag{5.2}$$

This means that, for ranges that require higher resolutions or cover a larger span, binary-coding becomes exponentially more attractive. Before moving on, it should be noted: for the decoding method, an additional bit will be included — being set to 1 when all other bits are 0 and set to 0 for any other case. This will prevent cases where $0x00$ cannot stimulate the input should it exist in the character set.

Action Selection

As TDRL algorithms are designed to produce a policy with a probabilistic distribution, it is necessary to discretize the policy in accordance with the distribution and encoding method. If the action logic is intended to be gray-coded, then it will likely be sufficient to use a roulette-style method such as Soft-Max. For this type of distribution method, the algorithm's output values are coupled such that the sum totals to 1. However, in pursuing a more compact and non-orthogonal binary-coding method, the action selection method must be decoupled for each probabilistic action value without having any drastic impact on how SARSA's error gradient is calculated. Given that Soft-Max is calculated by:[6]

$$p(a_{t+0.5}|\vec{s}_t) = \frac{e^{(\vec{a}^T_{t+0.5}\cdot[\mu]\cdot\vec{s}_t)}}{\sum_{b\in A} e^{(\vec{b}^T_{t+0.5}\cdot[\mu]\cdot\vec{s}_t)}}, \tag{5.3}$$

it can be considered a form of coupled sigmoidal thresholding — sigmoids being another equation that relies on the same full range $[-\infty, \infty]$ and form. The equivalent decoupled sigmoid will be defined as:

$$p(a_{t+0.5}|\vec{s}_t) = \frac{1}{1 + e^{-(\vec{a}^T_{t+0.5}\cdot[\mu]\cdot\vec{s}_t)}}. \tag{5.4}$$

The coupled pair for the implied second action $\vec{b}_{t+0.5}$ from the sigmoid function can also be generated from:

$$p(\neg a_{t+0.5}|\vec{s}_t) = \frac{1}{1 + e^{(\vec{a}^T_{t+0.5}\cdot[\mu]\cdot\vec{s}_t)}}\ . \tag{5.5}$$

[6]These equations assume that $[\mu]$ is the policy matrix and only action $a_t = 1$ in the action set A_t while all others are zero.

In both equations (5.4) and (5.5), the 1 value is generated by the continuous assumption that $\pm(\vec{b}^T_{t+0.5} \cdot [\mu] \cdot \vec{s}_t) = 0$. Because of the coupled nature of Soft-Max, this assumption is not present in its calculations for cases with more than one action value being set. However, by using equation (5.4), the gradient calculation for each action is preserved. Apart from decoupling the actions, it should be kept in mind that the sigmoid's knee point becomes fixed, *i.e.* the reference point for the probability calculation is not affected by other action probabilities. In this way, a random number in the range of $[0, 1)$ can be produced for each probabilistic action value $p(a_{t+0.5}|s_t)$, and if the random number is less than the probabilistic value, the corresponding output will be set to 1, otherwise it will be set to 0.

In the event that the algorithm is intended to learn in a linear fashion, *i.e.* the action policy values are expected to be within the range $[0, 1]$, the computational demand will be reduced further by using a piece-wise linear function such as:

$$p(a_{t+0.5}|\vec{s}_t) = \begin{cases} 1 & \text{If } 1 \leq \vec{a}^T_{t+0.5} \cdot [\mu] \cdot \vec{s}_t \ , \\ \vec{a}^T_{t+0.5} \cdot [\mu] \cdot \vec{s}_t & \text{If } 0 < \vec{a}^T_{t+0.5} \cdot [\mu] \cdot \vec{s}_t < 1. \\ 0 & \text{Otherwise.} \end{cases} \tag{5.6}$$

To reduce confusion when incorporating changes, the thresholding methods will be included as part of the LA code while the random number generator and selection occur separately. This also allows a distinct separation of a likely source of noise from the LA itself.[7]

Learning Algorithm (LA)

The LA will be described as a component that receives logical state data and qualitative reward data to guide the agent's actions towards maximizing reward. The first step of the LA will be to process state data for extraction of information relevant to the action policy. This information will then be filtered, scaled, and/or shifted to acquire a probability distribution for the

[7]The noise caused by random activation/selection is useful for exploration but harmful for exploitation, having it as a distinctly separate module from the LA itself was deemed to be preferable so as to make it clear that random number generation is not necessarily part of the LA though it is still part of the AI.

action set. After passing out the action probabilities, *i.e.* on the next time step, the results will be gathered and compared with prior expectations, generating a measure of error. The error will then be used to adjust the weights with the expectation of said error being reduced. With the new weight matrix, the process repeats. Of the types of LAs available, *e.g.* on-line or off-line; batch or iterative; on-policy or off-policy; supervised, semi-supervised, or unsupervised, this research will be restricted to on-line on-policy TDRL. Off-policy methods such as Q-learning assume each action is represented by a single element and rely on this to select the maximum $q(a_{t+0.5}, s_t)$ produced for a given state. Off-line learning methods such as batch learning require the algorithm to run on a set of training data, gathering the errors attributed for each weight, and applying the average of the errors at the end of one training epoch. Off-policy and off-line learning each add more complications to modifying the algorithm and will thus be avoided altogether. As for algorithms outside of TDRL, they are currently beyond the scope of our research. An important point to note for SARSA w.r.t. the type of problem testing will be conducted on is that, for highly non-stationary problems, there are often cases where INF or NaN occur because of the finite range and resolution of the numerical data-types used. To counter these cases that may result in runtime errors, checks will be imposed for operations likely to return INF. These INF values will be replaced with the python value $sys.float_info.max$ of matching sign, *i.e.* $\pm 1.7976931348623157 \times 10^{308}$. A similar check will be imposed for NaN which will be replaced with zero if it is a result of multiplication and one if it is due to division, *i.e.* $0 = 0 \times INF$ and $1 = INF/INF$ respectively. From prior trials, these occurrences have tended to happen during the policy and value calculations after a period of frustrated learning; replacing $\pm INF$ and NaN ensures that the outputs remain processable and that the learning process will not freeze before results will be gathered. That said, diverging weights usually mean the parameter combination is a poor choice — largely ruled out by the PSO — or the LA itself is unable to learn properly with the given problem configuration. If it fails for any reason during the in-sample optimization, the PSO algorithm will automatically assign a score of $-\infty$ and move on. The

modified LAs will automatically be marked as failed for any runtime errors or warnings they throw, *i.e.* there will be no attempts to push a parameter combination through if it fails even once.

5.3 System Simulation

The simulator will be designed to maximize parallelization, minimizing overall runtime. Each agent will process the problem entirely before the results are aggregated and the performance metrics are determined. Every in-sample and out-of-sample test will be initialized with 30 identical individuals, each solving the problem independently. Each of the 6 particles in the PSO will also be evaluated in parallel to further minimize processing time. Additionally, the PSO will terminate if the global best does not improve after 30 PSO iterations.[8]

There are a few options for evaluating the performance of the algorithm based on the resulting reward values. The method of choice is to calculate the running mean of rewards R_t^+ and R_t^-:[9]

$$\bar{R}_{t+1} = \bar{R}_t \times (1 - \zeta) + R_t \times \zeta, \tag{5.7}$$

for each agent. For grid-world problems, the performance measure used is based on how short the algorithm can make its path or the average reward given upon ending a training episode [36]. This method relies on the fact that the algorithm can choose which states to move into, affecting the length of the path traversed, and that there are terminal states.[10] Using a regular mean of the prediction accuracy across all samples is also based on the assumption that the algorithm is not learning during the out-of-sample evaluation phase [8], [19], [38]. However, SARSA and CLASP are continuous learning algorithms, meaning the same logical inputs at different points in time can yield different results as they make and learn from errors in between. The running

[8]The simulation will be run on an ASUS PC with Windows 10 and an intel i5-4690K CPU. Its maximum clock speed is 4.5GHz and it has access to 32GB of RAM.

[9]R_t^+ and R_t^- refer to the reward given for getting the exact character and bit-wise dissimilarity respectively.

[10]Terminal states are positions that move the algorithm back to the starting position to start the next episode.

average approach does not discredit older rewards in the online learning process, but also does not value them as much as the most recent results. After the simulations have completed, the mean and standard deviation across all 30 agents will then be processed to get:

$$R_{sim} = (\bar{R}^{+}_{ave@T+1} + \bar{R}^{-}_{ave@T+1}) - (\bar{R}^{+}_{std@T+1} - \bar{R}^{-}_{std@T+1}). \tag{5.8}$$

As there is a large difference in data set size for in-sample and out-of-sample problems, $\zeta = 0.011$ will be used for the in-sample data while $\zeta = 0.001$ will be used for the out-of-sample data. Considering weighted significance, only the $\lfloor 1/\zeta \rfloor$ most recent samples will be regarded as most relevant to the resulting score, *i.e.* the last 90 samples for the in-sample set and the last 1,000 samples for the out-of-sample set.

The hit-rate and bit-wise accuracy — the LA's ability to guess the character correctly and how close it is to the correct character — will be the primary methods of evaluation; however, per-character processing time via `time.time()`, LA memory via `pympler.asizeof.asizeof()`, and the $3 - \sigma$ evaluation results will also be collected. The hit-rate reward, originally $\in \{0, 1\}$, will be multiplied by 7 to get R^{+}_{t} while the bit-wise reward $\in \{0, -1, -2, -3, -4, -5, -6, -7\}$ will be divided by 7 to get R^{-}_{t}; placing emphasis on guessing more characters correctly over getting a close binary approximation. With these alterations in mind, accuracy will calculated using:

$$\bar{X}^{-}_{ave}\% = 100 \times (1 + \bar{R}^{-}_{ave@T+1}) \tag{5.9}$$

and the associated standard deviation is calculated using:

$$\bar{X}^{-}_{std}\% = 100 \times \bar{R}^{-}_{std@T+1}. \tag{5.10}$$

The corresponding hit rate is calculated as:

$$\bar{X}^{+}_{ave}\% = 100 \times \frac{\bar{R}^{+}_{ave@T+1}}{7} \tag{5.11}$$

and the associated standard deviation is calculated as:

$$\bar{X}^{+}_{std}\% = 100 \times \frac{\bar{R}^{+}_{std@T+1}}{7}. \tag{5.12}$$

In these calculations, $\bar{R}^{-}_{ave@T+1}$ and $\bar{R}^{-}_{std@T+1}$ refer to the mean and standard deviation of negative reward running average values, and $\bar{R}^{+}_{ave@T+1}$ and $\bar{R}^{+}_{std@T+1}$ refer to the mean and standard deviation of positive hit-rate running average values.

5.4 Summary

In this chapter, the simulation setup regarding the problem format and general agent configuration have been covered. With this system model, it will be possible to get a rough evaluation and comparison regarding the trade-offs of deviating from the SARSA model w.r.t. highly non-stationary problems. From the measures w.r.t. computational resources as well as detailed statistical performance, it should be possible to roughly determine how the changes affect learning and if the modifications are worth implementing in highly non-stationary environments.

Chapter 6

Testing CLASP and SARSA

The previous chapters covered the setup of the problem and the iterative changes made to improve the Learning Algorithm (LA) while using biological neurons as a source of inspiration. The algorithms were optimized on the first 987 characters from the forward of *Lord of the Ring — Fellowship of the Ring*. The experiments were conducted with only 1 character inputs while allowing or restricting the γ parameters. In forcing select γ values equal to 1, it was expected that a clearer comparison of how each modification affected performance would be possible. The results are based on the mean value of 30 identically initialized learning algorithms.

6.1 Results

The optimization process resulted in the parameter selections shown in tables 6.1, 6.3, and 6.5. For replicating the old δ method, *i.e.* modifications with the subscript 'old γ,' $\gamma_{\Re}$, γ_b, and γ_q were set to 1. Seeing as the chosen γ parameters are not always at extreme values of 1 or 0 and given the difference in the resulting rewards, the new δ method proved to be a major factor in improving the algorithm's scores. Looking at the chosen parameters for CLASP in table 6.1, $\lambda_h \approx 0$ and $\lambda_p \approx 0.2$ suggest that the algorithm preferred a probability of approximately 0.5 for activation when predictions fell within the gray zone of the decision boundary, *i.e.* $0.2 \leq \eta_{t+1} \leq 0.8$. Similar results can be seen in tables 6.3 and 6.5, suggesting that it was able to confidently predict some action elements, however, any states that fell within the gray zone were likely

Modification	$bias$	α_p	α_μ	α_q	$\gamma_{\Re}$	γ_b	γ_p	$\gamma_q \vert \alpha_r$	β_p	β_h	ζ_p	ζ_h	λ_p	λ_h	τ	a	c
SARSA	0.0954	—	0.2759	0.0350	—	—	0.6341	0.1081	—	—	—	—	—	—	—	—	—
Model$_{\text{old }\gamma}$	0.0561	0.2028	0.4892	0.5816	1.0000	1.0000	0.3062	1.0000	—	—	—	—	—	—	—	—	—
Model	0.3133	0.6061	0.7630	0.0986	0.6113	0.6395	0.6883	0.5565	—	—	—	—	—	—	—	—	—
Kernel$_{\text{old }\gamma}$	0.0518	0.6170	0.3091	0.1646	1.0000	1.0000	0.3413	1.0000	—	—	—	—	—	—	—	—	—
Kernel	0.0391	0.6395	0.5516	0.5277	0.8527	0.8200	0.2528	0.4480	—	—	—	—	—	—	—	—	—
Traces$_{\text{old }\gamma}$	0.2777	0.5131	0.7177	0.5441	1.0000	1.0000	0.5312	1.0000	0.9910	1.0000	0.0000	0.0000	0.4926	0.6342	—	—	—
Traces	0.5273	0.8081	0.7541	0.7367	0.7743	0.8423	0.8148	0.6074	0.3719	0.8205	0.0000	0.0000	0.4909	0.7720	—	—	—
Complex Error$_{\text{old }\gamma}$	0.0381	0.6432	0.3203	0.5254	1.0000	1.0000	0.6713	1.0000	0.4132	0.7892	0.0331	0.4621	0.5136	0.1187	—	—	—
Complex Error	0.0180	0.3634	0.9646	0.5517	0.4588	0.5588	0.3239	0.4601	0.0000	0.6917	0.0000	0.5308	0.6956	0.2359	—	—	—
Bounded Error$^{\dagger}_{\text{old }\gamma}$	0.9433	0.1543	0.3205	0.0000	1.0000	1.0000	0.3348	1.0000	0.0372	0.8263	0.4164	0.7292	0.3185	0.0000	2.7112	1.0812	90.1921
CLASP$^{\dagger *}$	0.9476	0.4952	0.3346	0.000000017	0.9188	0.6596	0.3420	1.0000	0.0628	0.8177	0.4662	0.5954	0.2037	0.000004	1.1267	1.1634	85.3354

† Bounded error used 6,442 characters to get properly optimized results.

* The PSO was having difficulty when trying to optimize all 17 parameters. To improve the PSO results, the first particle was seeded with the parameters from Bounded Error$_{\text{old }\gamma}$ with the extreme values being shifted from the boundaries by 0.00001.

Table 6.1: Optimal parameters based on Particle Swarm Optimization for 1 input character.

Method	R_{sim}	$\bar{R}_{ave@T}$	$\bar{R}_{std@T}$	$\bar{X}^{-}_{ave}\%$	$\bar{X}^{-}_{std}\%$	$\bar{X}^{+}_{ave}\%$	$\bar{X}^{+}_{std}\%$	$memory[bytes]$	$t^{max}_{ave}[s]$	$t^{max}_{std}[s]$	σ_{acc1}	σ_{acc2}	σ_{acc3}
SARSA	0.3473	0.7129	0.3655	59.1055	1.4253	16.0258	5.2082	18,704	0.0166	0.0040	86.6667	90.0000	96.6667
Model$_{\text{old }\gamma}$	0.6019	0.8161	0.2142	59.3508	1.2819	17.4650	3.2432	603,008	0.0611	0.0131	96.6667	96.6667	96.6667
Model	0.8558	0.8558	0.0000	59.1128	0.0000	18.0672	0.0000	603,008	0.0542	0.0078	100.0000	100.0000	100.0000
Kernel$_{\text{old }\gamma}$	-0.1666	0.4305	0.5971	58.1778	2.7929	12.1242	8.3354	83,280	0.0330	0.0060	80.0000	90.0000	100.0000
Kernel	0.3035	0.7009	0.3974	58.9233	1.8756	15.8807	5.6180	83,280	0.0332	0.0063	86.6667	90.0000	96.6667
Traces$_{\text{old }\gamma}$	-0.5661	0.0128	0.5788	53.4201	4.6890	6.8370	7.7014	83,280	0.0368	0.0058	63.3333	96.6667	100.0000
Traces	0.3127	0.7201	0.4073	58.1827	2.7902	16.2605	5.4202	83,280	0.0321	0.0051	90.0000	90.0000	100.0000
Complex Error$_{\text{old }\gamma}$	-0.5110	-0.3253	0.1857	54.3533	6.9800	1.8737	1.8803	83,152	0.0383	0.0042	56.6667	96.6667	100.0000
Complex Error	-0.5107	-0.3639	0.1468	52.6695	4.9291	1.5631	1.5372	83,152	0.1344	0.0419	70.0000	93.3333	96.6667
Bounded Error$^{\dagger}_{\text{old }\gamma}$	0.3230	0.5596	0.2366	58.1876	3.1629	13.9671	2.9557	83,152	0.0355	0.0034	73.3333	93.3333	100.0000
CLASP	0.5583	0.7993	0.2410	58.8007	1.5996	17.3040	3.2147	83,152	0.0363	0.0039	96.6667	96.6667	96.6667

$t^{max}_{ave}[s]$ and $t^{max}_{std}[s]$ are based on the maximum time spent to process 1 character for 1 agent's run for which the average and standard deviation are taken across the 30 agent's with identical parameters.

Table 6.2: The out-of-sample results for each modification. The final CLASP algorithm is the result of Bounded Error.

Modification	$bias$	α_p	α_μ	α_q	$\gamma_{\mathfrak{R}}$	γ_b	γ_p	$\gamma_q \vert \alpha_r$	β_p	β_h	ζ_p	ζ_h	λ_p	λ_h	τ	a	c
SARSA	0.2407	0.1104	0.0533	0.5847	0.3439	–	–	–	–	–	–	–	–	–	–	–	–
CLASP	0.4082	0.6629	0.5414	0.3426	0.3082	0.5722	0.2651	0.5408	0.6876	1.0000	0.2290	0.5886	0.3273	0.0000	5.5865	0.0000	99.1463

Table 6.3: Optimal parameters based on Particle Swarm Optimization for 1 input character with Bit-wise accuracy scaled down such that the total reward is in range $[1, -1]$.

Method	R_{sim}	$\bar{R}_{ave@T}$	$\bar{R}_{std@T}$	$\bar{X}^-_{ave}\%$	$\bar{X}^-_{std}\%$	$\bar{X}^+_{ave}\%$	$\bar{X}^+_{std}\%$	$memory[bytes]$	$t^{max}_{ave}[s]$	$t^{max}_{std}[s]$	σ_{acc1}	σ_{acc2}	σ_{acc3}
SARSA	-0.2250	-0.1640	0.0610	67.7450	3.0312	15.8559	5.4978	18,736	0.4164	0.0860	90.0000	90.0000	100.0000
CLASP	-0.2439	-0.2231	0.0208	62.9727	1.9104	14.7185	3.2169	83,152	0.3595	0.1945	63.3333	100.0000	100.0000

$t^{max}_{ave}[s]$ and $t^{max}_{std}[s]$ are based on the maximum time spent to process 1 character for 1 agent's run for which the average and standard deviation are taken across the 30 agent's with identical parameters.

Table 6.4: The out-of-sample results for each modification with Bit-wise accuracy scaled down such that the total reward is in range $[1, -1]$.

Modification	$bias$	α_p	α_μ	α_q	$\gamma_{\mathfrak{R}}$	γ_b	γ_p	$\gamma_q \vert \alpha_r$	β_p	β_h	ζ_p	ζ_h	λ_p	λ_h	τ	a	c
SARSA	0.3379	0.0356	0.0398	0.3993	0.3817	–	–	–	–	–	–	–	–	–	–	–	–
CLASP	0.3309	0.5821	0.5715	0.1133	0.2384	0.7336	0.0494	0.5645	0.6909	0.7987	0.2259	0.5470	0.4219	0.0000	7.0685	0.4855	71.5274

Table 6.5: Optimal parameters based on Particle Swarm Optimization for 1 input character with Bit-wise accuracy scaled such that the total reward is in range $[1, -7]$.

Method	R_{sim}	$\bar{R}_{ave@T}$	$\bar{R}_{std@T}$	$\bar{X}^-_{ave}\%$	$\bar{X}^-_{std}\%$	$\bar{X}^+_{ave}\%$	$\bar{X}^+_{std}\%$	$memory[bytes]$	$t^{max}_{ave}[s]$	$t^{max}_{std}[s]$	σ_{acc1}	σ_{acc2}	σ_{acc3}
SARSA	-2.0708	-2.0210	0.0498	69.8436	0.5218	8.9940	2.8793	18,736	0.3452	0.0640	76.6667	93.3333	96.6667
CLASP	-2.2924	-2.1957	0.0967	67.5966	1.1326	7.2558	3.2795	83,152	0.2987	0.2239	76.6667	93.3333	100.0000

$t^{max}_{ave}[s]$ and $t^{max}_{std}[s]$ are based on the maximum time spent to process 1 character for 1 agent's run for which the average and standard deviation are taken across the 30 agent's with identical parameters.

Table 6.6: The out-of-sample results for each modification with Bit-wise accuracy scaled such that the total reward is in range $[1, -7]$.

too intermingled to be allocated a reliable action policy. The action values too challenging to learn with one TDRL algorithm would likely require additional layers of processing to resolve. It is also worth noting that mixing parameters β_h, β_p, ζ_h, and ζ_p were not pushed to 0 or 1 as is with Traces$_{old\gamma}$. This suggests that moderately mixing prediction information and historical information into the input state space — what would be expected to result in a garbage-in-garbage-out scenario in most cases — was preferred during optimization, *i.e.* bounded error was able to filter out unwanted data. In general, the sigmoid function for bounded error seems to have had a tendency to be crisp, either being comparable to a piecewise linear function or a step function. For the case where hit rate was given more weight than bit-wise accuracy, a very small α_q — the critic's quality learning rate — was preferred, while more moderate values were selected in the other two test cases.

The Out-of-Sample results are shown in tables 6.2, 6.4, and 6.6, as well as in appendices B and C. Of the modifications shown in table 6.2, only Model$_{old\gamma}$, Model, and CLASP achieved R_{sim} values greater than SARSA. Again w.r.t. table 6.2, CLASP achieved a bit-wise accuracy ($\bar{X}^{-}_{ave}\%$) that was lower than SARSA's by 0.3 of a percent with its standard deviation being higher by 0.17 of a percent, but had a hit rate ($\bar{X}^{+}_{ave}\%$) that was higher by 1.3 of a percent with the deviation being 2.0 of a percent lower. The 3-σ result for the 30 separate runs was produced by checking the percentage of how many runs were within the given standard deviation for bit-wise accuracy. These also suggest that CLASP's results are more tightly grouped w.r.t. 1-σ and 2-σ. Given that the new δ method seems to have relatively consistent improvements in the 3-σ results, it is likely to be partially responsible for CLASP's smaller deviations from the mean. The result of testing with different scales for bit-wise accuracy, given in tables 6.4 and 6.6, show the parameters chosen by PSO were not sufficient for CLASP to perform better than SARSA. Whether it is because the number of parameters is too many for PSO to reliably find the optimal set or that the algorithm does not learn as effectively with frequent negative rewards, there are still areas that can be improved on. These results may be improved by slightly deviating from the pure binary state decoding method

in favor of a state distribution that is easier to interpret by the algorithm.[1] Regardless, CLASP has potential to be used in real world applications with its inherent stability.

The two focal contributions of this research, the new method of calculating δ and the bounded error, worked very favorably. The new δ method increased the learning stability by better regulating reward damping characteristics. The results show bounded error restricts the amount of over-training that can occur, forcing excessively large predictions to be reduced by flushing out superfluous weights. A visible consequence is that the standard deviation was notably reduced (see figure B.11). The stability shown by CLASP is very appealing for problems where the LA cannot be allowed to fail during runtime or where the nature of the state-space or rewards may change over time. Obtaining results that indicate the LA is performing poorly is better than that plus the risk of a forced reset because of an overflow runtime error. Additionally, given that stability cannot be guaranteed when combining function approximation, bootstrapping, and off-policy learning, this added stability may open up an avenue for further development within the region of the deadly triad [17]. For CLASP, it was difficult for the PSO to find the best values for τ, a, and c partially because of how significantly the new δ method influenced performance as well as because of the large number of searchable parameters relative to the population size. This was accommodated for by seeding the first individual of the first PSO iteration with the optimal result for Bounded $\text{Error}_{\text{old }\gamma}$ — or with the result of a prior run that did not have a seeded individual; encouraging particles to move into the general neighborhood of the optimal τ, a, and c before working on the γ parameters.

The addition of a purely logical model, increased memory and time requirements by factors of ≈ 34 and ≈ 4 respectively. Applying the kernel trick with binary coding countered the memory and time demands required for including the model component, reducing the factors to ≈ 4 and ≈ 2 respectively. The order of complexity for the kerneled method with binary encoding was found

[1] It should not be forgotten that SARSA is using Gray-coded states which distribute information over a notably larger number of logical values.

to be better than using gray-coding for sufficiently large state-spaces. Traces of historical values and predictions in the input-space served to increase the amount of information available to the algorithm, including: state activation duration, predictions of non-observable data w.r.t. the input-space, as well as predictions that may enforce or contradict potentially fallible observations. The complex error modification was an attempt to evaluate and learn with the full proposed range of complex probabilistic values; unfortunately, its results were worse than those of just adding the trace information. A better result may be found by changing how $p(N_{t+1}|\tilde{\tilde{\chi}})_t$ is calculated or applied in the logical error.[2]

Given the PSO focused on maximizing $\bar{X}^+$, it is understandable that $\bar{X}^-_{ave}$ did not change more than 1.4 of a percent for modifications that did not result in notably poor R_{sim} values, which is smaller than SARSA's corresponding standard deviation. Part of this was likely because the algorithm chose to compromise the best policy of a relatively smaller set of states in favor of making better judgments for a relatively larger set of predictions. In general, it would appear that there is some restriction preventing the algorithms from breaching a hit-rate of 20% and accuracy of 70%. It is likely that the information density for the given states have reached a saturation point or the remaining bits are too chaotic to be predicted from a single character. To exceed 20%, it is expected to require multiple algorithms, *i.e.* daemons, which are able focus on different character and rule sets. These sets would entirely depend on the similarity of rules for a given character to come next; some characters may even belong to multiple sets — a form of distributed duplicate memory — if there are more than one set of rules. That said, it would require another agent to choose which daemon, *i.e.* rule set, to follow. The 3-σ results are not as useful as hoped because there were only 30 samples of each agent, however, from what can be seen, CLASP seems to be relatively consistent with

[2]The notably larger time was likely due to a background program requiring enough processing power to force at least one of the algorithms to pause during the processing cycle for several seconds. Additionally, the parameters α_μ and λ_h were too small to be represented with only 4 significant digits, but would be similar to the optimal parameters found for CLASP (Bounded Error).

only one abnormal agent result. In almost all instances, there was at most, only one agent whose results fell outside the range of 3-σ, suggesting that the agents' learning within the simulations were relatively consistent.

Chapter 7

Conclusions, Contributions, and Future Work

7.1 Conclusions

In this research project, the SARSA algorithm was modified into what is now presented as the Complex-Logical-Action-State-Prediction (CLASP) algorithm. At the expense of increased memory and time requirements for the unigram problem covered in this research, several things were achieved:

- reduced order of complexity for large state-spaces by allowing binary encoding;
- increased the flexibility for encoding options;
- included potentially relevant information (*i.e.* historical and predictive) without a loss of stability;
- enabled the algorithm to make predictions for and act on unobservable state information derived by its internal model;
- emulated in more detail, how the mediums for rewards and punishments are transmitted and recovered;
- provided a method to maintain algorithmic stability by restricting weight growth during learning;

- proposed a potential avenue for weight transparency via exploitation of complex number properties and considering a weight space that permits trigonometric-like relations;
- and incorporated a more detailed model of how neurotransmitters would likely be influenced during emission.

Overall, CLASP is successful in improving on SARSA in these aspects.

7.2 Contributions

The unique contributions to the formation of CLASP are the revised δ calculation, the application of complex weights, and the bounded error learning method. The δ calculation in SARSA focuses strictly on how neurons release neurotransmitters as a way of suggesting a states expected value, and recover them plus the reward neurotransmitter injected into the algorithm during learning. The new δ method considers the aspects of release and recovery in addition to dispersion and decomposition outside the cell. This change improves the stability of learning state-action qualities. Complex weights allow for changes in magnitude and phase, offering more flexibility in the number of logical operations that can be executed. The complex values also allow the algorithm to account for non-dual action and state predictions/observations. Bounded error learning method limits the total size of a given 'neuron' by imposing a material limitation. This differs from methods that involve weight decay as it takes effect only when the unbounded prediction distance error exceeds a predetermined amount and only affects weights used for said prediction. By limiting the input weights' size, CLASP prevents divergent behavior. However, this also limits the effective coverage of data points to those sufficiently far from the decision boundary. Imposing a form of limited plastic deformation via the generalized sigmoid function ensures the bounded error's stability. Restricting the maximum and minimum possible error to ± 1 respectively ensures the error itself cannot become unstable and form a positive feedback loop. The logical values in this research were applied with the as-

sumption that they have units in the form of energy or amplitude, deviating from the logic space expectation that they are unit-less. This change in perspective allows the weights to be viewed as trigonometric relations with high degrees of similarity to conventional logical operations, instead of as purely logical or probabilistic components. W.r.t. the original objectives, this research has:

- produced a new δ calculation method that improves the stability of state-action quality predictions,
- proposed an alternative weight interpretation based on complex values and trigonometry, and
- implemented a bounded logical error method which prevents weight divergence by filtering out less relevant weights, stabilizing the logical error calculation process.

7.3 Future Work

In future research, the number of explorable parameters will need to be reduced as the optimization difficultly for the PSO was showing signs of struggle when handling more than 14 dimensions. It would also be desirable to remove the model, or at least separate it from the rest of the policy component, to recover the perks of being model-free. Going a step further, it would be of interest to reduce the output scale to a single neuron per algorithm to see if stability can be maintained within the deadly triad, *i.e.* bridging the gap between TDRL and NN models. Testing CLASP in a horde architecture would also be worth pursuing as the new δ method presents a potential method of propagating rewards through $\mathfrak{r}_t$. Its ability to handle fallible observations and delay is also worth conducting more in-depth research as CLASP is expected to partially consider for these non-ideal scenarios. Given how bounded error operates, it may be worth testing if CLASP can exploit the availability of extra action spaces to develop its own internal states and rules independently of its external environment. Lastly, It will be necessary to evaluate CLASP's

weight interpretation in more detail to properly verify the validity of using trigonometric relations as a basis for interpretation.

The problem chosen for this research was a challenge for SARSA and CLASP. It would be worth testing these algorithms using word level n-gram prediction which would likely be easier. Grid-world problems would also work to check the difference in convergence speed and how different encoding methods affect state-space interpret-ability w.r.t. the LA. Some checks with simple logical operations were done to ensure LA was operating as intended, but more rigorous examinations of what CLASP can do should be conducted to more thoroughly demonstrate how its weights change over time.

References

[1] B. Alberts, A. Johnson, J. Lewis, P. Walter, M. Raff, and K. Roberts, *Molecular biology of the cell 4th edition*, Ion Channels and the Electrical Properties of Membranes, New York: Garland Science, 2002. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK26910/. 30, 31

[2] (2018). All's well that ends well. George Mason University; Open Source Shakespeare, [Online]. Available: https://www.opensourceshakespeare.org/views/plays/playmenu.php?WorkID=allswell (visited on 08/14/2018). 83

[3] D. Atlas, "The voltage-gated calcium channel functions as the molecular switch of synaptic transmission," *Annual Review of Biochemistry*, vol. 82, no. 1, pp. 607–635, 2013, PMID: 23331239. DOI: 10.1146/annurev-biochem-080411-121438. eprint: https://doi.org/10.1146/annurev-biochem-080411-121438. [Online]. Available: https://doi.org/10.1146/annurev-biochem-080411-121438. 28

[4] (2019). Axiomatic probability and point sets. Lecture 2, [Online]. Available: https://www.le.ac.uk/users/dsgp1/COURSES/LEISTATS/SAMPLINF.html (visited on 05/02/2019). 6

[5] J. Baillieul and P. J. Antsaklis, "Control and communication challenges in networked real-time systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 9–28, Jan. 2007, ISSN: 0018-9219. DOI: 10.1109/JPROC.2006.887290. 86

[6] V. Bhatt-Mehta and M. C. Nahata, "Dopamine and dobutamine in pediatric therapy," *Pharmacotherapy: The Journal of Human Pharmacology and Drug Therapy*, vol. 9, no. 5, pp. 303–314, Oct. 1989, ISSN: 1875-9114. DOI: 10.1002/j.1875-9114.1989.tb04142.x. [Online]. Available: https://doi.org/10.1002/j.1875-9114.1989.tb04142.x. 61

[7] M. P. Blaustein and W. J. Lederer, "Sodium/calcium exchange: Its physiological implications," *Physiological Reviews*, vol. 79, no. 3, pp. 763–854, 1999, ISSN: 0031-9333. eprint: http://physrev.physiology.org/content/79/3/763.full.pdf. [Online]. Available: http://physrev.physiology.org/content/79/3/763. 27

[8] A. Z. Broder, S. C. Glassman, M. S. Manasse, and G. Zweig, "Syntactic clustering of the web," *Computer Networks and ISDN Systems*, vol. 29, no. 8, pp. 1157–1166, 1997, Papers from the Sixth International World Wide Web Conference, ISSN: 0169-7552. DOI: https://doi.org/10.1016/S0169-7552(97)00031-7. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0169755297000317. 81, 83, 92

[9] W. A. Catterall, "Structure and regulation of voltage-gated ca2+ channels," *Annual Review of Cell and Developmental Biology*, vol. 16, no. 1, pp. 521–555, 2000, PMID: 11031246. DOI: 10.1146/annurev.cellbio.16.1.521. eprint: https://doi.org/10.1146/annurev.cellbio.16.1.521. [Online]. Available: https://doi.org/10.1146/annurev.cellbio.16.1.521. 28

[10] (2018). Character-level language model. Imad Dabura, [Online]. Available: https://towardsdatascience.com/character-level-language-model-1439f5dd87fe. 2

[11] D.-R. Chen, Q. Wu, Y. Ying, and D.-X. Zhou, "Support vector machine soft margin classifiers: Error analysis," *J. Mach. Learn. Res.*, vol. 5, pp. 1143–1175, Dec. 2004, ISSN: 1532-4435. [Online]. Available: http://dl.acm.org/citation.cfm?id=1005332.1044698. 73

[12] E. S. L. Faber and P. Sah, "Physiological role of calcium-activated potassium currents in the rat lateral amygdala," *Journal of Neuroscience*, vol. 22, no. 5, pp. 1618–1628, 2002, ISSN: 0270-6474. eprint: http://www.jneurosci.org/content/22/5/1618.full.pdf. [Online]. Available: http://www.jneurosci.org/content/22/5/1618. 28

[13] ——, "Calcium-activated potassium channels: Multiple contributions to neuronal function," *The Neuroscientist*, vol. 9, no. 3, pp. 181–194, 2003, PMID: 15065814. DOI: 10.1177/1073858403009003011. eprint: http://dx.doi.org/10.1177/1073858403009003011. [Online]. Available: http://dx.doi.org/10.1177/1073858403009003011. 28

[14] D. Fiala, F. Mueller, C. Engelmann, R. Riesen, K. Ferreira, and R. Brightwell, "Detection and correction of silent data corruption for large-scale high-performance computing," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, ser. SC '12, Salt Lake City, Utah: IEEE Computer Society Press, 2012, 78:1–78:12, ISBN: 978-1-4673-0804-5. [Online]. Available: http://dl.acm.org/citation.cfm?id=2388996.2389102. 86

[15] J. L. Fitzakerley. (2014). Ion permeabilities, [Online]. Available: http://www.d.umn.edu/~jfitzake/Lectures/DMED/IonChannelPhysiology/MembranePotentials/Permeabilities.html (visited on 09/17/2018). 27

[16] P. R. Halmos, *Measure theory / [by] paul r. halmos*, English. Springer-Verlag New York, 1974, xi, 304 p. ISBN: 0387900888. [Online]. Available: http://www.loc.gov/catdir/enhancements/fy0814/74010690-t.html. 20

[17] H. van Hasselt, Y. Doron, F. Strub, M. Hessel, N. Sonnerat, and J. Modayil, *Deep reinforcement learning and the deadly triad*, 2018. arXiv: 1812.02648 [cs.AI]. 99

[18] D. S. Jeong, I. Kim, M. Ziegler, and H. Kohlstedt, "Towards artificial neurons and synapses: A materials point of view," *RSC Advances*, vol. 3, no. 10, pp. 3169–3183, 2013. 75

[19] D. Jurafsky and M. J. H., *Speech and language processing (3rd ed. draft)*. Stanford University, 2020. [Online]. Available: https://web.stanford.edu/~jurafsky/slp3/ed3book_dec302020.pdf. 81, 83, 92

[20] V. V. Klinshov, J.-n. Teramae, V. I. Nekorkin, and T. Fukai, "Dense neuron clustering explains connectivity statistics in cortical microcircuits," *PLOS ONE*, vol. 9, no. 4, pp. 1–12, Apr. 2014. DOI: 10.1371/journal.pone.0094292. [Online]. Available: https://doi.org/10.1371/journal.pone.0094292. 58

[21] D. Kroeger, "Brain activity patterns in deep anesthesia," PhD thesis, Université Laval, 2008. [Online]. Available: http://archimede.bibl.ulaval.ca/archimede/fichiers/25863/25863.html. 27

[22] (2015). Lord of the ring– the fellowship of the ring. Internet Archive, [Online]. Available: https://archive.org/stream/TheLordOfTheRing1TheFellowshipOfTheRing/The%20Lord%20Of%20The%20Ring%201-The%20Fellowship%20Of%20The%20Ring_djvu.txt (visited on 04/26/2019). 83

[23] E. Mendelson, *Introduction to mathematical logic 4th edition*, New York: Queens College, Aug. 1997. [Online]. Available: https://www.karlin.mff.cuni.cz/~krajicek/mendelson.pdf. 6, 14

[24] (2011). Neuronal action potential, [Online]. Available: http://www.physiologyweb.com/lecture_notes/neuronal_action_potential/neuronal_action_potential_important_features.html (visited on 10/01/2018). 28, 30

[25] E. Pchelintseva and M. B. A. Djamgoz, "Mesenchymal stem cell differentiation: Control by calcium-activated potassium channels," *Journal of Cellular Physiology*, n/a–n/a, 2017, ISSN: 1097-4652. DOI: 10.1002/jcp.26120. [Online]. Available: http://dx.doi.org/10.1002/jcp.26120. 28

[26] (2011). Resting membrane potential, [Online]. Available: http://www.physiologyweb.com/lecture_notes/resting_membrane_potential/resting_membrane_potential_in_real_cells_multiple_ions_contribute_to_the_membrane_potential.html (visited on 09/17/2018). 27, 28

[27] M. Rice, G. Gerhardt, P. Hierl, G. Nagy, and R. Adams, "Diffusion coefficients of neurotransmitters and their metabolites in brain extracellular fluid space," *Neuroscience*, vol. 15, no. 3, pp. 891–902, 1985, ISSN: 0306-4522. DOI: https://doi.org/10.1016/0306-4522(85)90087-9. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0306452285900879. 61

[28] C. Rieger and B. Zwicknagl, "Deterministic error analysis of support vector regression and related regularized kernel methods," *J. Mach. Learn. Res.*, vol. 10, pp. 2115–2132, Dec. 2009, ISSN: 1532-4435. [Online]. Available: http://dl.acm.org/citation.cfm?id=1577069.1755856. 73

[29] M. Ring, "Representing knowledge as predictions (and state as knowledge)," An unpublished paper used with conditional permission of the author, Mar. 2016. 68

[30] I. Rinke, "Chloride regulatory mechanisms and their influence on neuronal excitability," PhD thesis, Ludwig-Maximilians-Universität München, Dec. 2010. 27–31

[31] J. Schlauwitz and P. Musilek, "A dimension-wise particle swarm optimization algorithm optimized via self-tuning," in *2020 IEEE Congress on Evolutionary Computation (CEC)*, 2020, pp. 1–8. 82

[32] (2011). Secondary active transport, [Online]. Available: http://www.physiologyweb.com/lecture_notes/membrane_transport/secondary_active_transport.html (visited on 10/01/2018). 27, 29, 30

[33] A. Seth. (2017). Your brain hallucinates your conscious reality, [Online]. Available: https://www.ted.com/talks/anil_seth_how_your_brain_hallucinates_your_conscious_reality (visited on 09/25/2018). 67

[34] T. C. Südhof, "Calcium control of neurotransmitter release," *Cold Spring Harbor perspectives in biology*, vol. 4, no. 1, a011353, 2012. 27

[35] T. C. Südhof and J. Rizo, "Synaptic vesicle exocytosis," *Cold Spring Harbor perspectives in biology*, vol. 3, no. 12, a005637, Dec. 2011. DOI: 10.1101/cshperspect.a005637. 61

[36] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*, ser. Computational neuroscience series. Cambridge (Mass.): MIT Press London, 2013. 45–47, 53, 56, 92

[37] A. White, "Developing a predictive approach to knowledge," PhD thesis, University of Alberta, 2015. 47

[38] K. Wołk, K. Marasek, and W. Glinkowski, "Telemedicine as a special case of machine translation," *Computerized Medical Imaging and Graphics*, vol. 46, pp. 249–256, 2015, Information Technologies in Biomedicine, ISSN: 0895-6111. DOI: https://doi.org/10.1016/j.compmedimag.2015.09.005. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0895611115001275. 81, 83, 92

Appendix A

CLASP Algorithm Summary

This appendix contains the series of equations used for CLASP described throughout Chapter 4. The algorithm starts with mixing information into the logical inputs, followed by policy generation. These values are used to determine the quality prediction which is needed for generating the quality prediction error. The final steps are updating the weights used for prediction and generating the new policy distribution. The complete CLASP algorithm is written as:

$$\tilde{\chi}_t = \zeta_p \times \chi_t + (1 - \zeta_p) \times \mathbb{R}\{\sqrt{p(\eta_t|\tilde{\chi})_{t-1}}\}; \tag{4.31}$$

$$\chi_{t-\tilde{m}-0.5} = \beta_h \times (\beta_p \times \mathbb{R}\{\tilde{\chi}_t\} + (1-\beta_p) \times \mathbb{R}\{\tilde{\chi}_{t-1}\}) + (1-\beta_h) \times \chi_{t-\tilde{m}-1.5}; \tag{4.32}$$

$$\tilde{\chi}_{t-\tilde{m}-0.5} = \zeta_h \times \chi_{t-\tilde{m}-0.5} + (1 - \zeta_h) \times \mathbb{I}\{\sqrt{p(\eta_t|\tilde{\chi})_{t-1}}\}; \tag{4.33}$$

$$\tilde{\tilde{\chi}}_t = \frac{\tilde{\chi}_t + \hat{\imath}\tilde{\chi}_{t-\tilde{m}-0.5}}{\max\left(1, |\tilde{\chi}_t + \hat{\imath}\tilde{\chi}_{t-\tilde{m}-0.5}|\right)}; \tag{4.35}$$

$$\vec{\eta}_{t+1} = \left[\left[p^3\right] \cdot \vec{\tilde{\tilde{\chi}}}_t\right] \cdot \vec{\tilde{\tilde{\chi}}}_t; \tag{4.36}$$

$$\tilde{\eta}_{t+1} = \frac{(\mathbb{R}\{\eta_{t+1}\} - 0.5) \times \lambda_h}{1 - \lambda_h} + 0.5; \tag{4.42}$$

$$\mathbb{R}\{p(N_{t+1}|\tilde{\tilde{\chi}})_t\} = \begin{cases} -1 & \text{if } -1 \geq \mathbb{R}\{\eta_{t+1}\}, \\ \mathbb{R}\{\eta_{t+1}\} & \text{if } 0 > \mathbb{R}\{\eta_{t+1}\} > -1, \\ 0 & \text{if } \lambda_p \geq \mathbb{R}\{\eta_{t+1}\} \geq 0 \text{ or } 0 \geq \tilde{\eta}_{t+1}, \\ 0 & \text{if } 0.5 \geq \mathbb{R}\{\eta_{t+1}\} \geq \lambda_p \text{ and } 0 \geq \tilde{\eta}_{t+1}, \\ 1 & \text{if } \mathbb{R}\{\eta_{t+1}\} \geq 1 - \lambda_p \text{ or } \tilde{\eta}_{t+1} \geq 1, \\ \tilde{\eta}_{t+1} & \text{otherwise;} \end{cases} \tag{4.43}$$

$$\mathbb{I}\{p(N_{t+1}|\tilde{\tilde{\chi}})_t\} = \begin{cases} 1 & \text{if } \mathbb{I}\{\eta_{t+1}\} \geq 1, \\ \mathbb{I}\{\eta_{t+1}\} & \text{if } 1 > \mathbb{I}\{\eta_{t+1}\} > 0, \\ 0 & \text{if } 0 \geq \mathbb{I}\{\eta_{t+1}\}; \end{cases} \tag{4.44}$$

$$p(N_{t+1}|\tilde{\tilde{\chi}})_t = \frac{\mathbb{R}\{p(N_{t+1}|\tilde{\tilde{\chi}})_t\} + \hat{\imath}\mathbb{I}\{p(N_{t+1}|\tilde{\tilde{\chi}})_t\}}{\max\left(1, |\mathbb{R}\{p(N_{t+1}|\tilde{\tilde{\chi}})_t\} + \hat{\imath}\mathbb{I}\{p(N_{t+1}|\tilde{\tilde{\chi}})_t\}|\right)}; \tag{4.45}$$

$$q(\eta, \chi)_{t+0.5} = \sqrt{p(\eta_{t+1}|\tilde{\tilde{\chi}})_t} \cdot \left[[Q^3] \cdot \vec{\tilde{\tilde{\chi}}}_t\right] \cdot \vec{\tilde{\tilde{\chi}}}_t; \tag{4.37}$$

$$\partial_{raw} = \begin{cases} sig^+(z + x - y, a, 1, c) & \text{If } x \geq 1 \text{ and } y > z + x, \\ 0 & \text{If } x \geq 1 \text{ and } \tau + x \geq y \geq x, \\ sig^+(x - y, a, 1, c) & \text{If } x \geq 0 \text{ and } x > y, \\ sig^+(x - y, a, 1, c) & \text{If } 1 > x \text{ and } y > x, \\ 0 & \text{If } 0 \geq x \text{ and } x \geq y \geq x - 1 - z, \\ sig^+(x - y - 1 - z, a, 1, c) & \text{If } 0 \geq x \text{ and } x - 1 - z > y; \end{cases} \tag{4.50}$$

where:

$$\begin{aligned} x &= \mathbb{R}\{\tilde{\tilde{\chi}}_t\}^2; \\ y &= \mathbb{R}\{\eta_t\}; \\ z &= \tau \times \#X^2; \end{aligned}$$

$$\partial_t = \begin{cases} \alpha_p \times \partial_{raw} & \text{for state predictions,} \\ \alpha_\mu \times \partial_{raw} \times \delta_t & \text{for policy learning;} \end{cases} \tag{4.52}$$

$$\begin{aligned} \delta_t = \gamma_{\mathfrak{R}} \times (r_{t-0.5} + \gamma_{\mathfrak{R}} \times (1 - \gamma_b) \times \mathfrak{R}_{t-1}) \\ + \gamma_p \times \gamma_q \times q(\eta, \chi)_{t+0.5} \\ - \gamma_q \times (1 - \gamma_{\mathfrak{R}} \times (1 - \gamma_b)) \times q(\eta, \chi)_{t-0.5}; \end{aligned} \tag{4.22}$$

$$\mathfrak{r}_t = \gamma_{\mathfrak{R}} \times \gamma_b \times (\gamma_q \times q(\eta, \chi)_{t-0.5} + \gamma_{\mathfrak{R}} \times \mathfrak{R}_{t-1}); \tag{4.23}$$

$$\mathfrak{r}_{t+0.5} = \gamma_{\mathfrak{R}} \times \mathfrak{r}_t;$$

$$\begin{aligned} \mathfrak{R}_t = \gamma_{\mathfrak{R}} \times (r_{t-0.5} + \gamma_{\mathfrak{R}} \times (1 - \gamma_b) \times \mathfrak{R}_{t-1}) \\ - \gamma_q \times (1 - \gamma_{\mathfrak{R}} \times (1 - \gamma_b)) \times q(\eta, \chi)_{t-0.5}; \end{aligned} \tag{4.24}$$

$$[Q^3] = [Q^3] + \frac{\alpha_q \times \delta_t \times \vec{\chi}_t \otimes \vec{\tilde{\chi}}_{t-1} \otimes \vec{\tilde{\chi}}_{t-1}}{\max(1, \left(\sum |\vec{\tilde{\chi}}_t|^2\right) \times \left(\sum |\vec{\tilde{\chi}}_{t-1}|\right)^2)}; \tag{4.38}$$

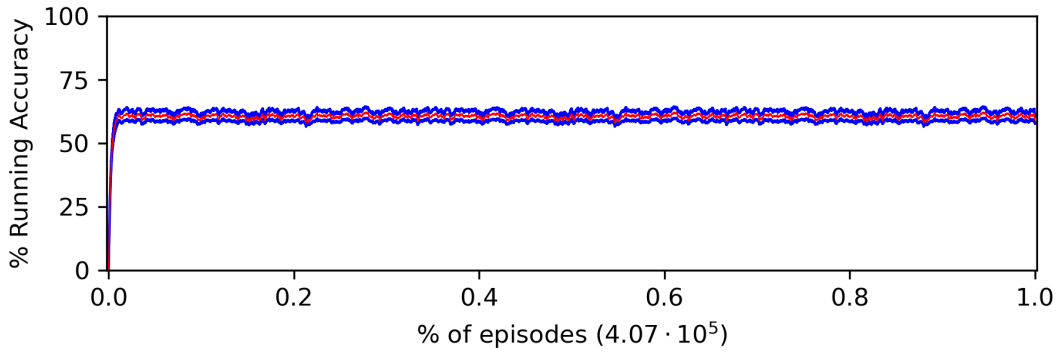
and

$$[p^3] = [p^3] + \frac{\vec{\partial}_t \otimes \vec{\tilde{\chi}}_{t-1} \otimes \vec{\tilde{\chi}}_{t-1}}{\max(1, \left(\sum |\vec{\tilde{\chi}}_t|^2\right) \times \left(\sum |\vec{\tilde{\chi}}_{t-1}|\right)^2)}. \tag{4.39}$$

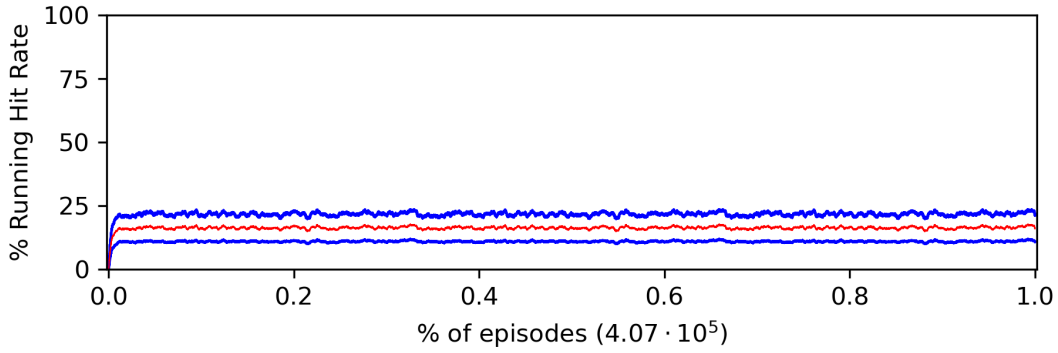
Appendix B

Out-of-Sample Plots

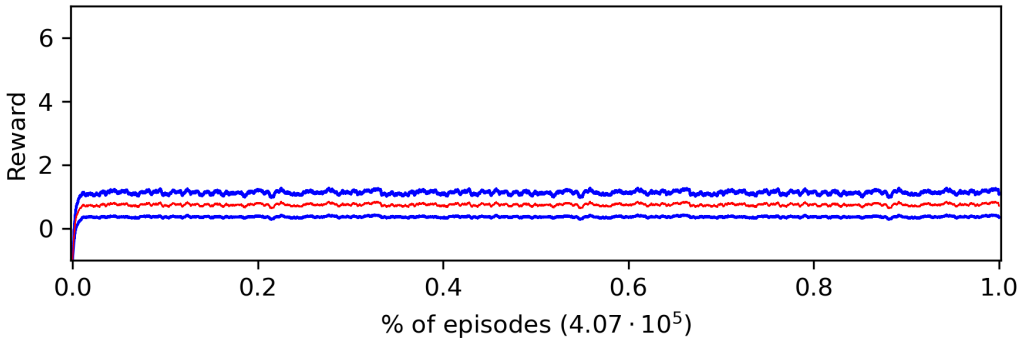
The figures in this appendix are the resulting plots for the running average (red) and the upper and lower running standard deviation (blue) over the out-of-sample character data for SARSA and the modifications leading up to CLASP. These tests use a weight of 1 for bit-wise accuracy (% Running Accuracy) and 7 for hit rate (% Hit Rate) when fed to the learning algorithm, *i.e.* hit rate is prioritized over bit-wise accuracy. From these plots, it can be seen that the new δ error method improves the mean and standard deviation in every instance that did not suddenly worsen over time. The Bounded Error, *i.e.* the final modification for CLASP, in figures B.10 and B.11 also show that it was able to restrict the worsening of the rewards.



(a)

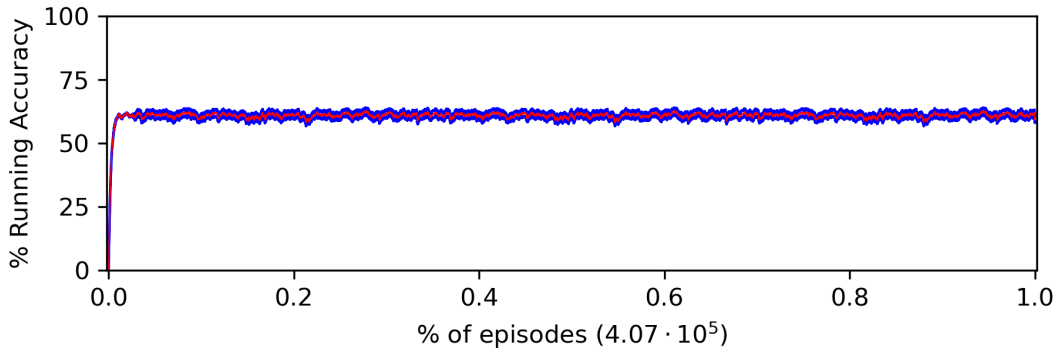


(b)

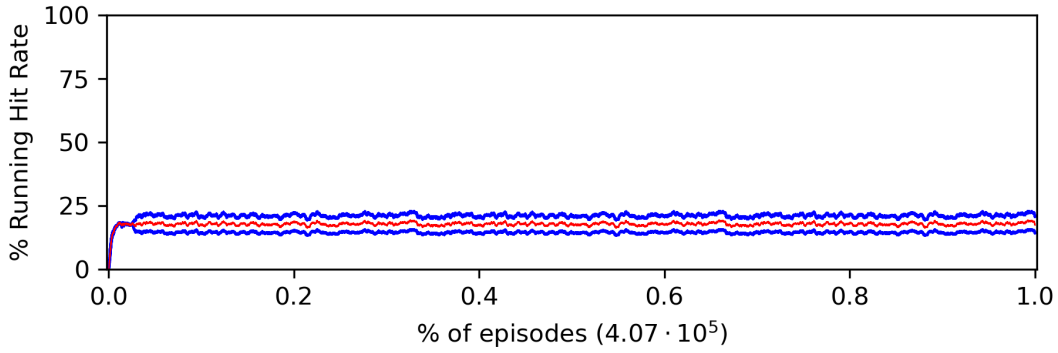


(c)

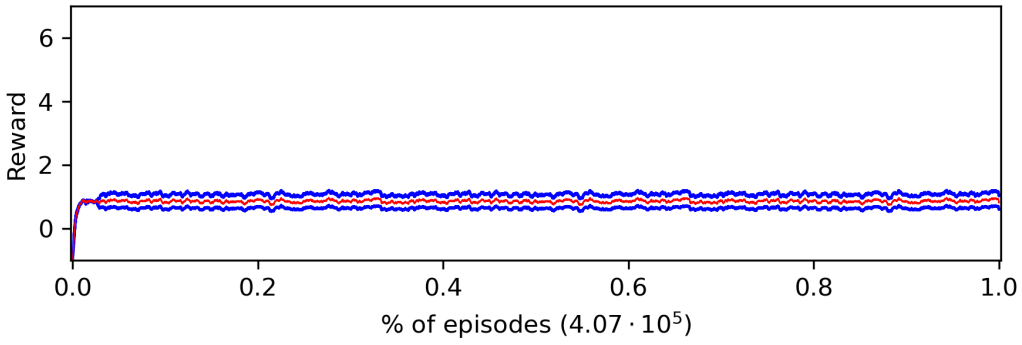
Figure B.1: The out-of-sample results for SARSA where (a) corresponds to the negative values associated with accuracy, (b) corresponds to the positive binary value associated with hit rate, and (c) corresponds to the reward used to evaluate the algorithms overall performance.



(a)

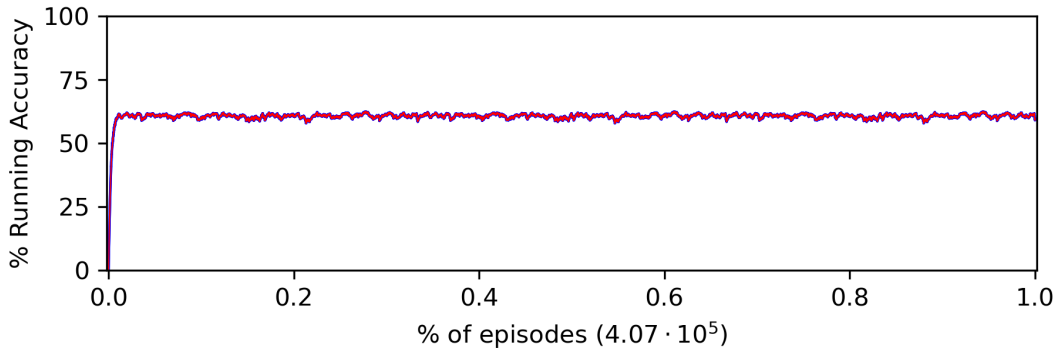


(b)

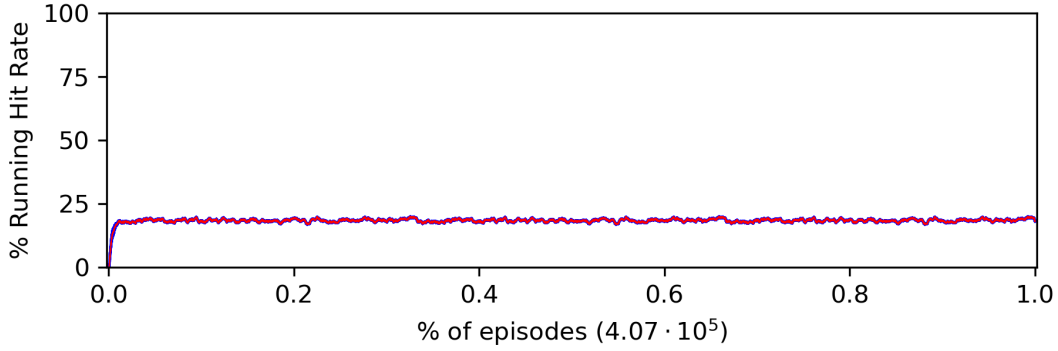


(c)

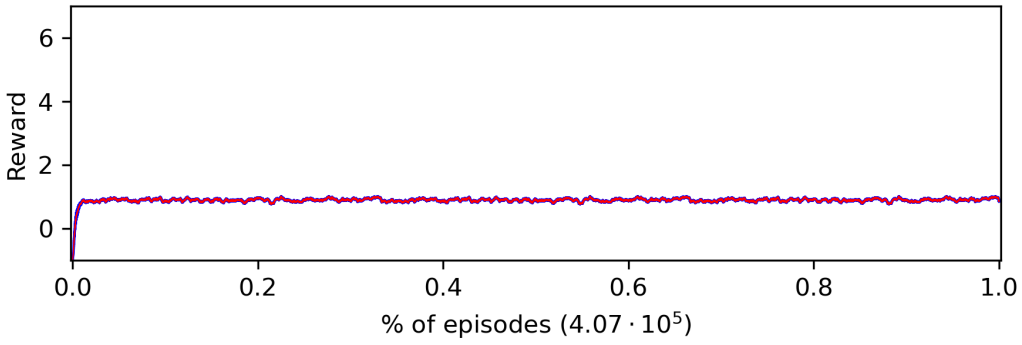
Figure B.2: The out-of-sample results for $\text{Model}_{\text{old }\gamma}$ where (a) corresponds to the negative values associated with accuracy, (b) corresponds to the positive binary value associated with hit rate, and (c) corresponds to the reward used to evaluate the algorithms overall performance.



(a)

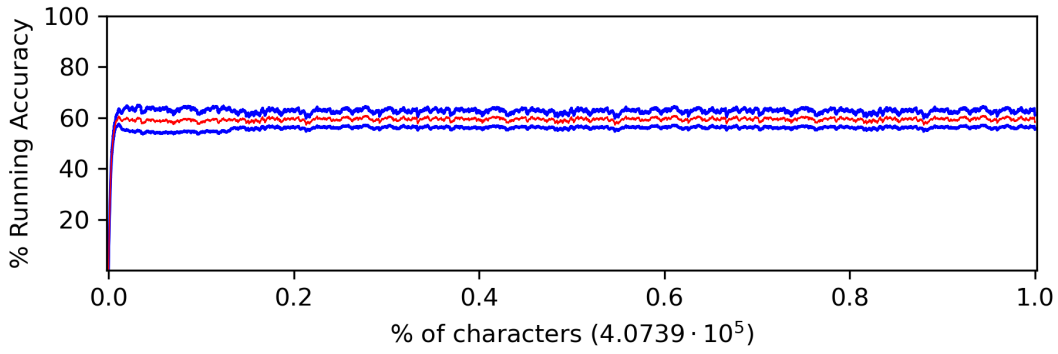


(b)

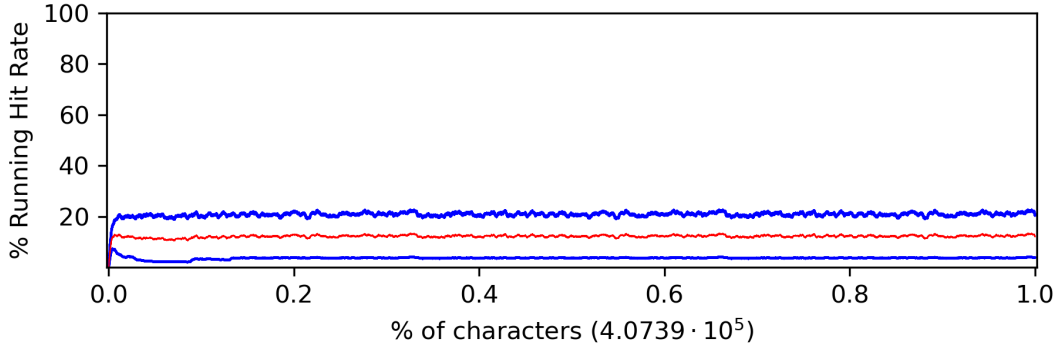


(c)

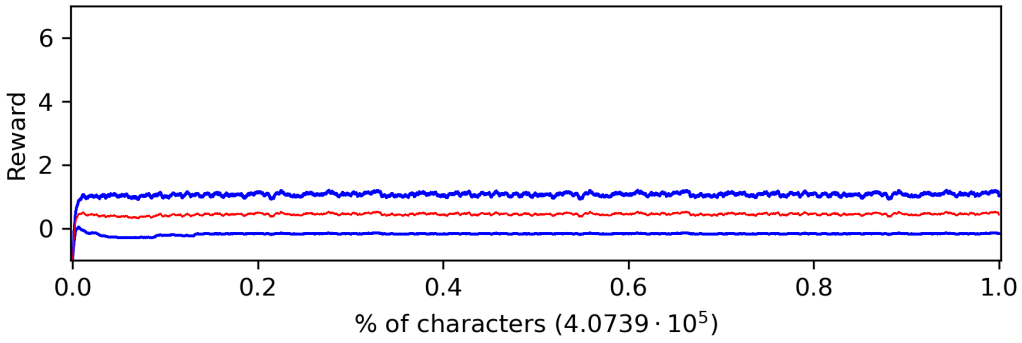
Figure B.3: The out-of-sample results for Model where (a) corresponds to the negative values associated with accuracy, (b) corresponds to the positive binary value associated with hit rate, and (c) corresponds to the reward used to evaluate the algorithms overall performance.



(a)

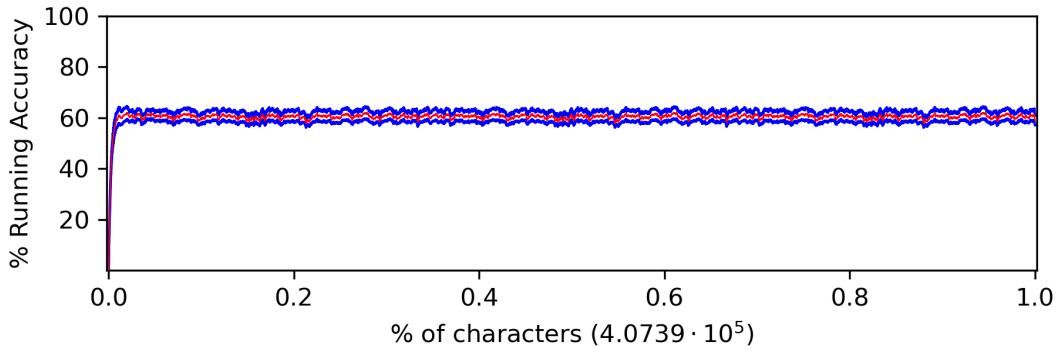


(b)

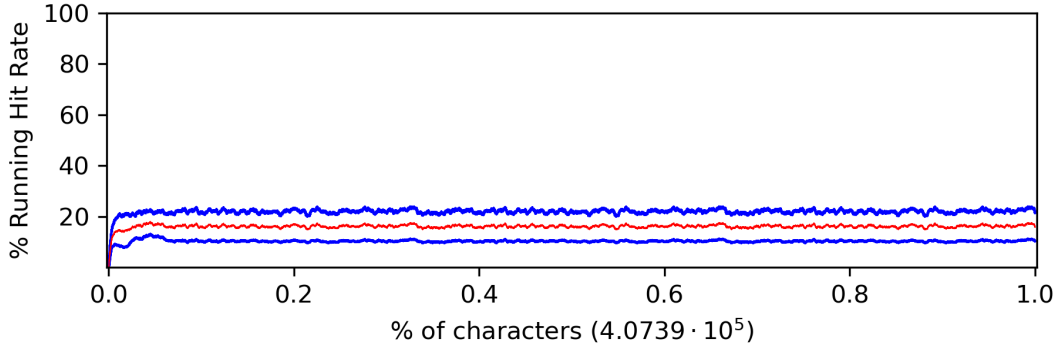


(c)

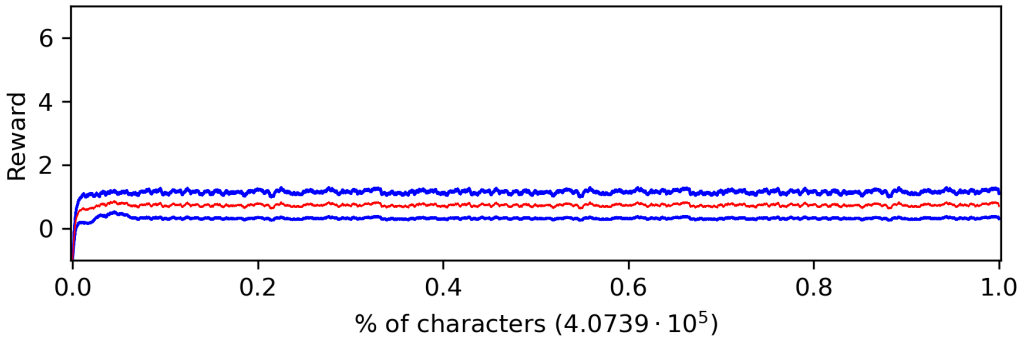
Figure B.4: The out-of-sample results for $\text{Kernel}_{\text{old }\gamma}$ where (a) corresponds to the negative values associated with accuracy, (b) corresponds to the positive binary value associated with hit rate, and (c) corresponds to the reward used to evaluate the algorithms overall performance.



(a)

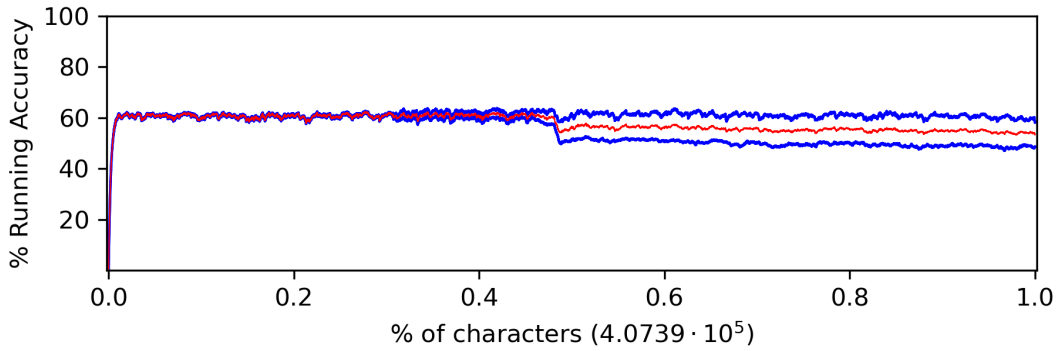


(b)

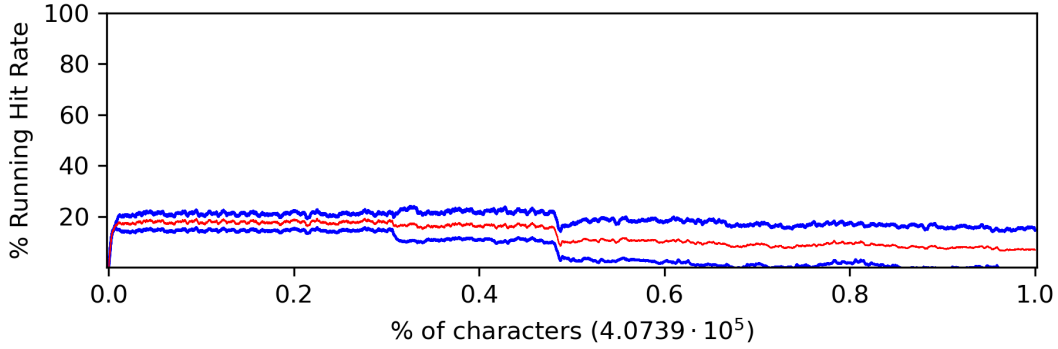


(c)

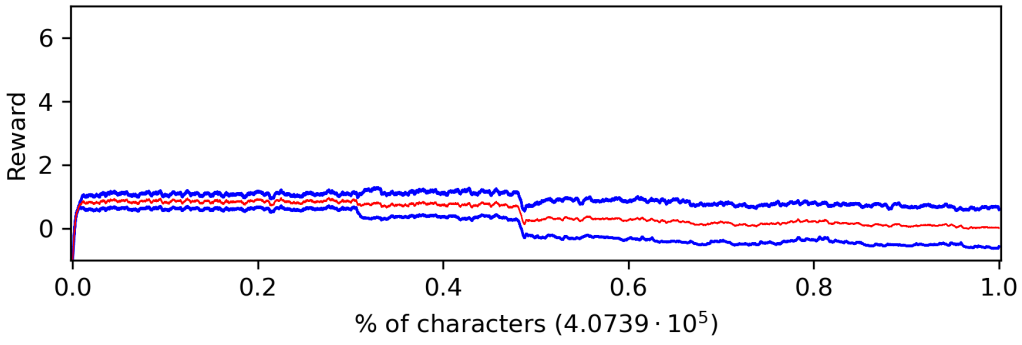
Figure B.5: The out-of-sample results for Kernel where (a) corresponds to the negative values associated with accuracy, (b) corresponds to the positive binary value associated with hit rate, and (c) corresponds to the reward used to evaluate the algorithms overall performance.



(a)

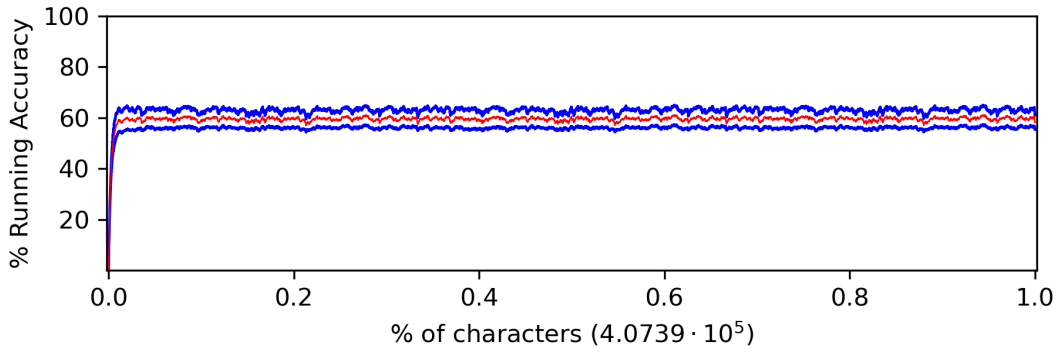


(b)

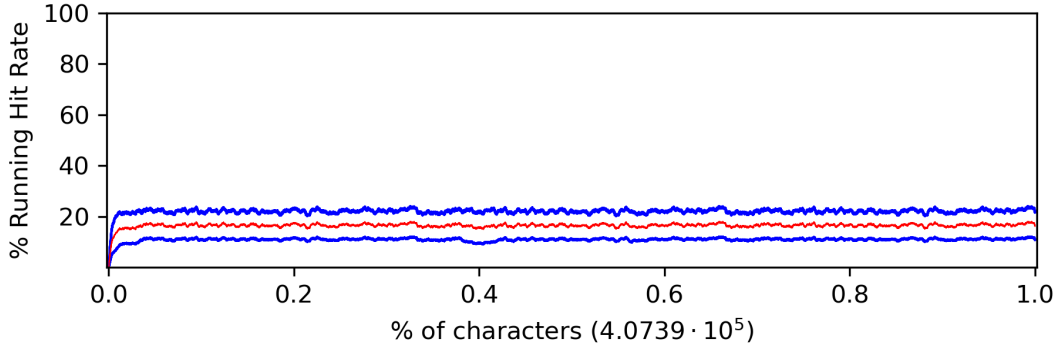


(c)

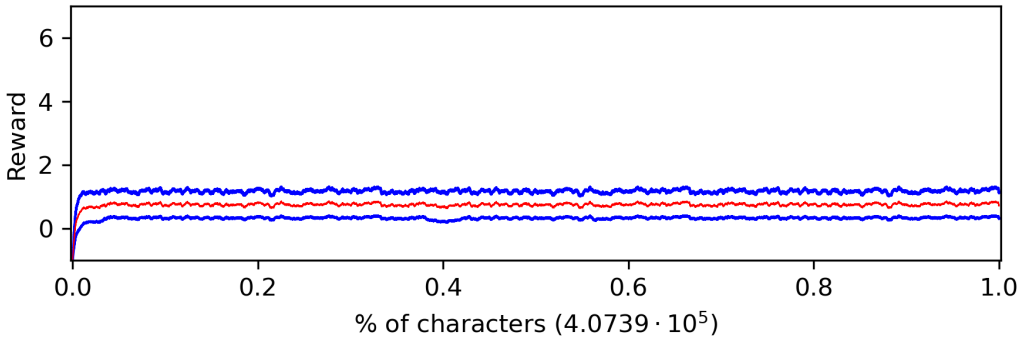
Figure B.6: The out-of-sample results for $\text{Traces}_{\text{old }\gamma}$ where (a) corresponds to the negative values associated with accuracy, (b) corresponds to the positive binary value associated with hit rate, and (c) corresponds to the reward used to evaluate the algorithms overall performance.



(a)

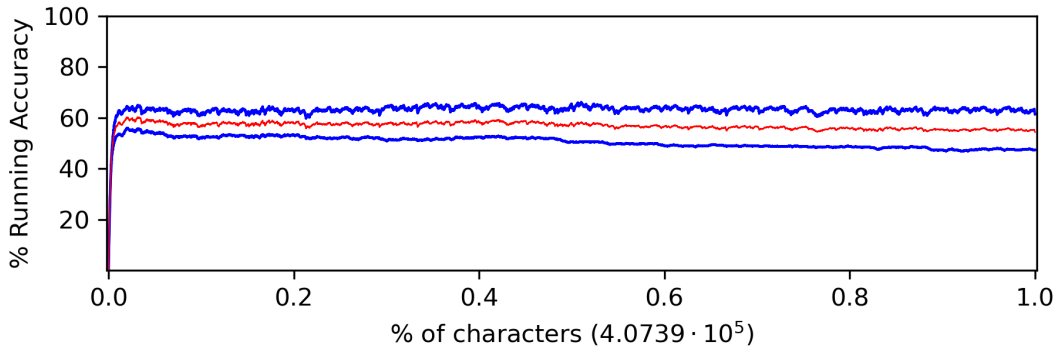


(b)

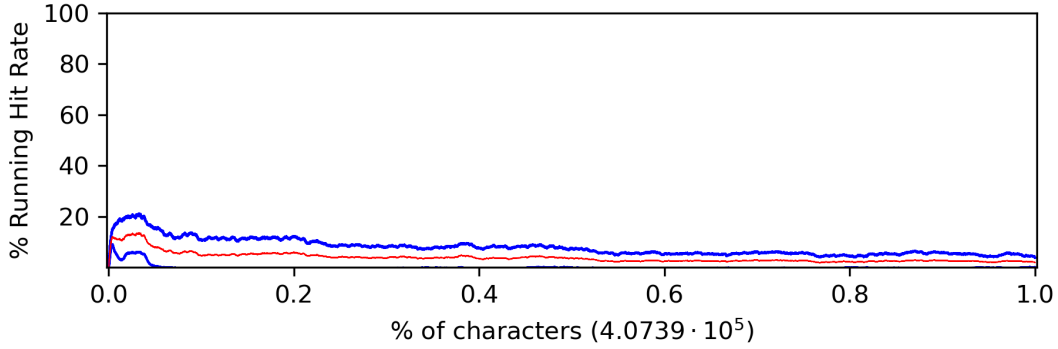


(c)

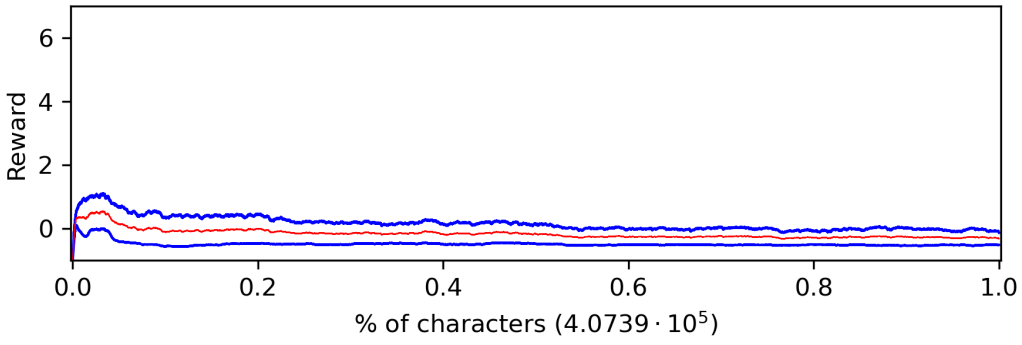
Figure B.7: The out-of-sample results for Traces where (a) corresponds to the negative values associated with accuracy, (b) corresponds to the positive binary value associated with hit rate, and (c) corresponds to the reward used to evaluate the algorithms overall performance.



(a)

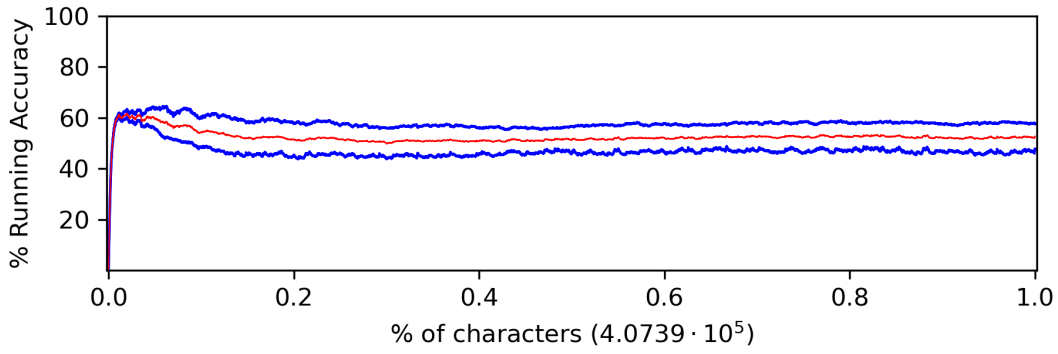


(b)

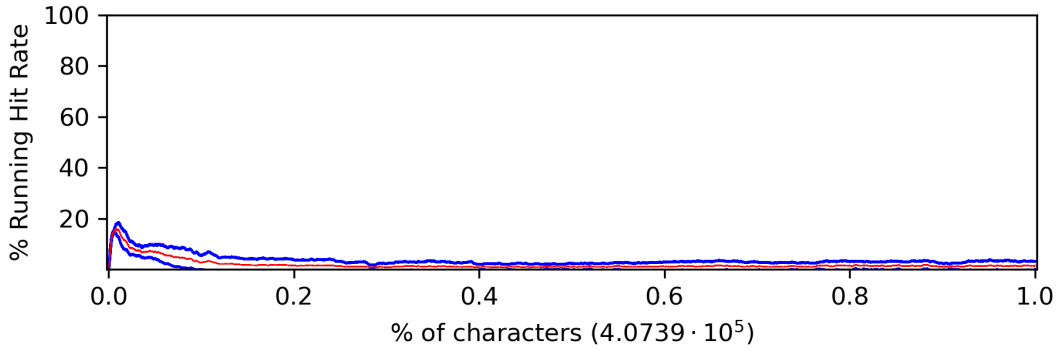


(c)

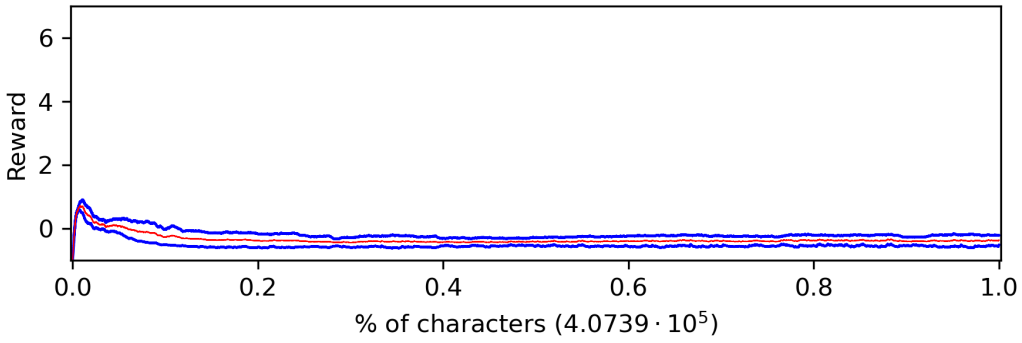
Figure B.8: The out-of-sample results for Complex $\text{Error}_{\text{old }\gamma}$ where (a) corresponds to the negative values associated with accuracy, (b) corresponds to the positive binary value associated with hit rate, and (c) corresponds to the reward used to evaluate the algorithms overall performance.



(a)

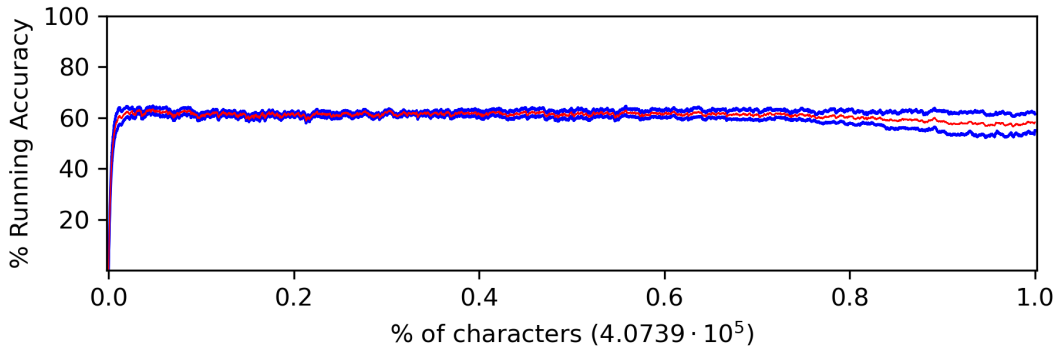


(b)

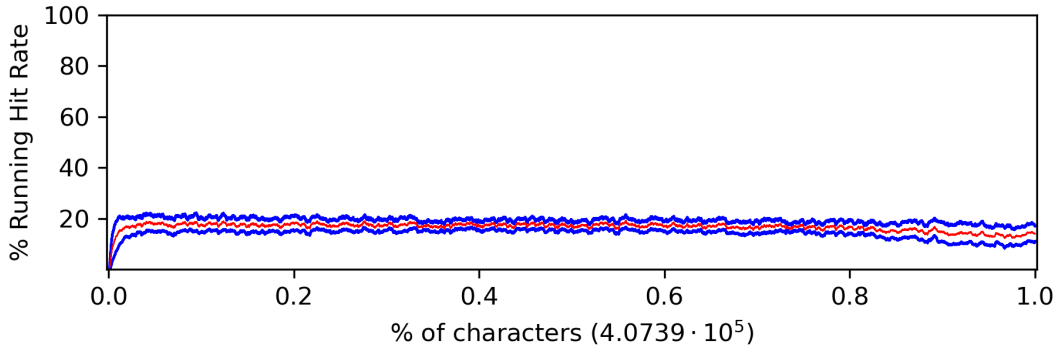


(c)

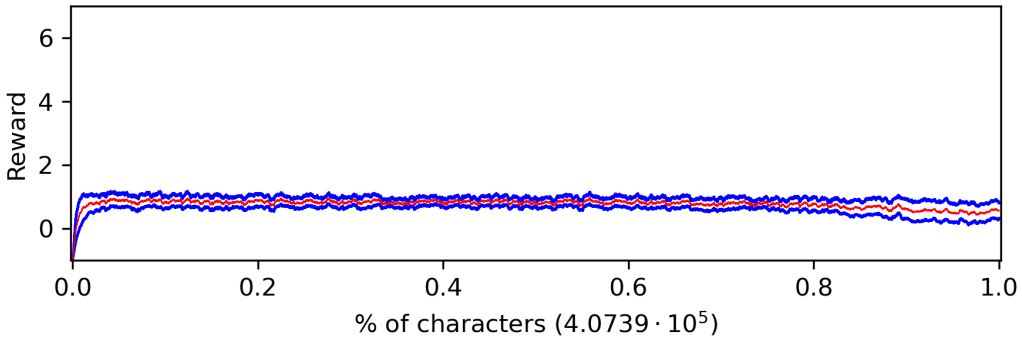
Figure B.9: The out-of-sample results for Complex Error where (a) corresponds to the negative values associated with accuracy, (b) corresponds to the positive binary value associated with hit rate, and (c) corresponds to the reward used to evaluate the algorithms overall performance.



(a)

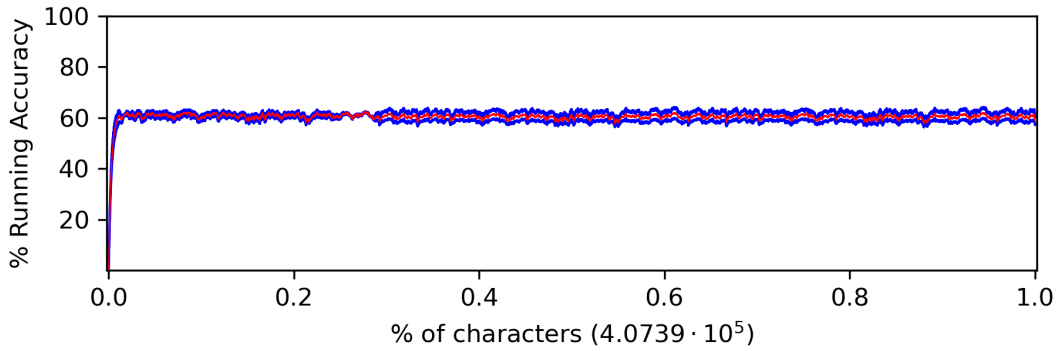


(b)

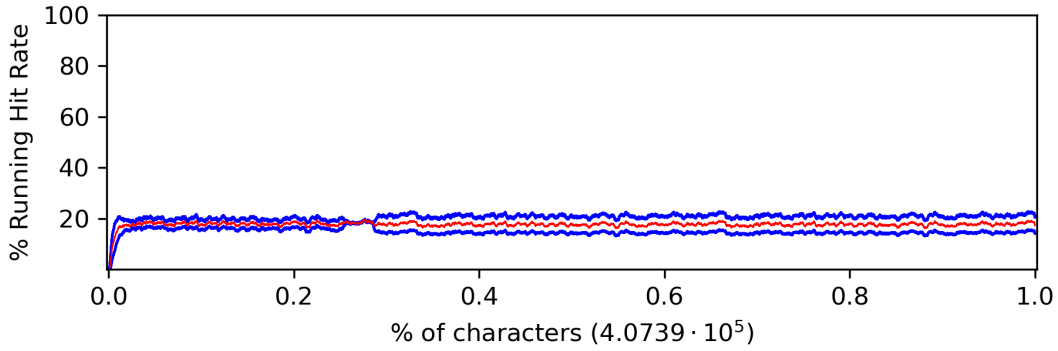


(c)

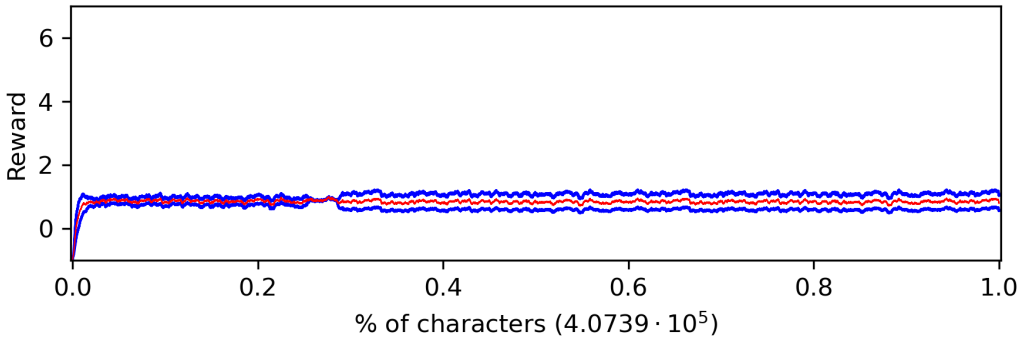
Figure B.10: The out-of-sample results for Bounded $\text{Error}_{\text{old }\gamma}$ where (a) corresponds to the negative values associated with accuracy, (b) corresponds to the positive binary value associated with hit rate, and (c) corresponds to the reward used to evaluate the algorithms overall performance.



(a)



(b)



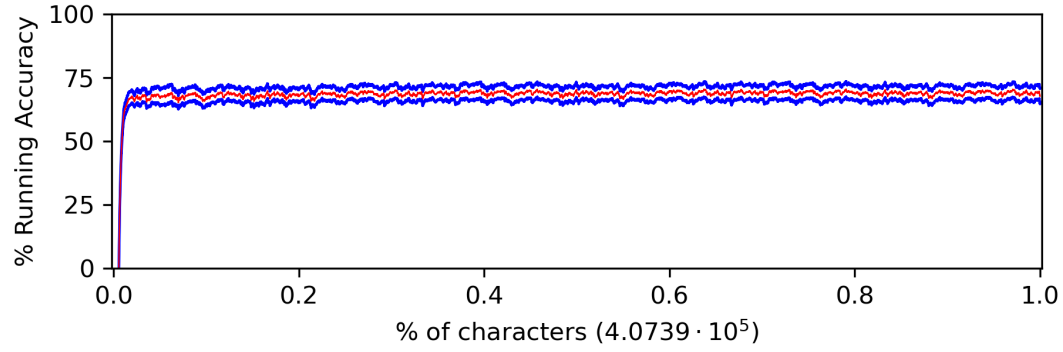
(c)

Figure B.11: The out-of-sample results for CLASP (Bounded Error) where (a) corresponds to the negative values associated with accuracy, (b) corresponds to the positive binary value associated with hit rate, and (c) corresponds to the reward used to evaluate the algorithms overall performance.

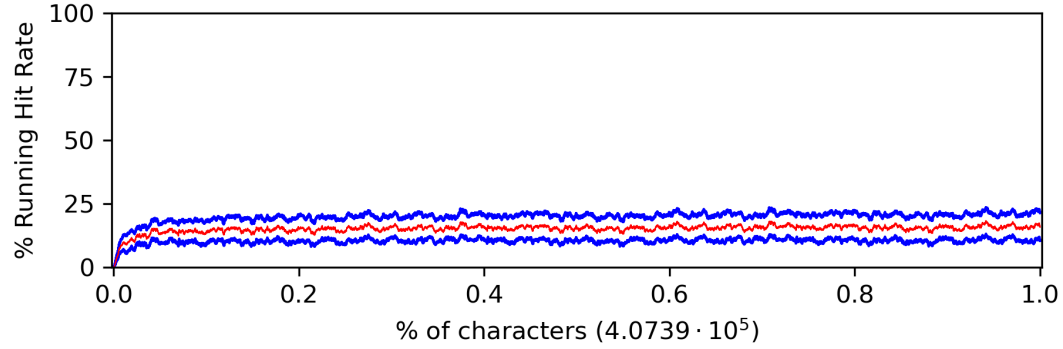
Appendix C

Variant Out-of-Sample Plots

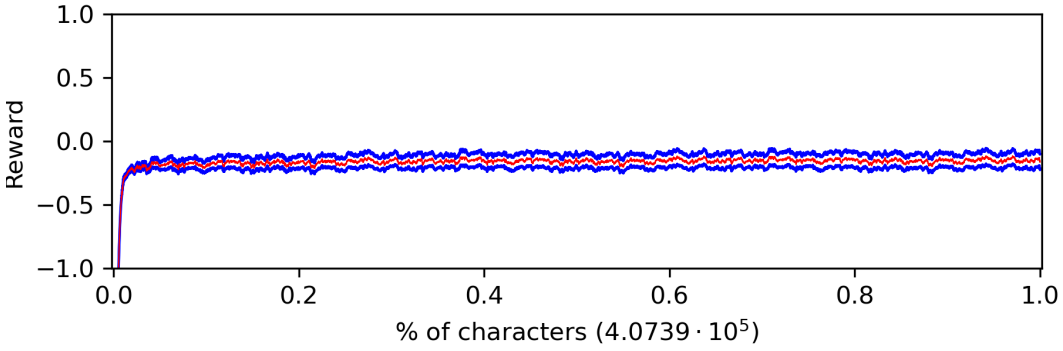
The figures in this appendix are the resulting plots for the running average (red) and the upper and lower running standard deviation (blue) over the out-of-sample character data for SARSA and CLASP. These tests vary the weight of bit-wise accuracy (% Running Accuracy) and hit rate (% Hit Rate) fed to the learning algorithm to see how it affects the learning performance. The cases shown here are when the largest magnitude for hit-rate and bit-wise accuracy are 1 and 1 or 1 and 7, *i.e.* the largest values are given equal consideration or bit-wise accuracy is given more respectively.



(a)

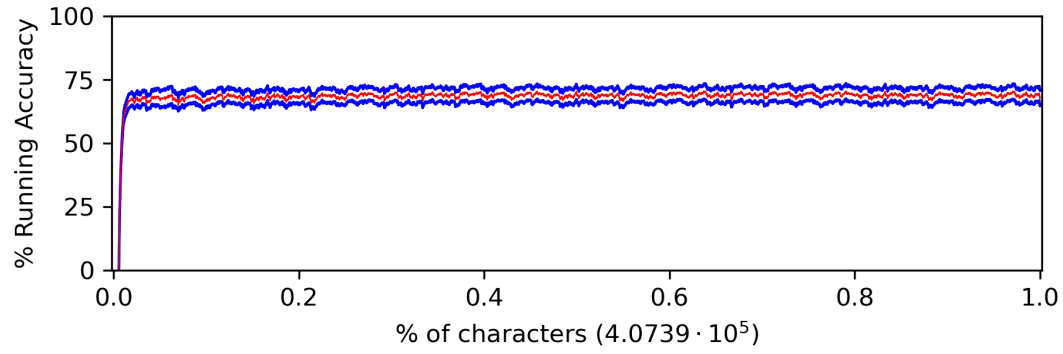


(b)

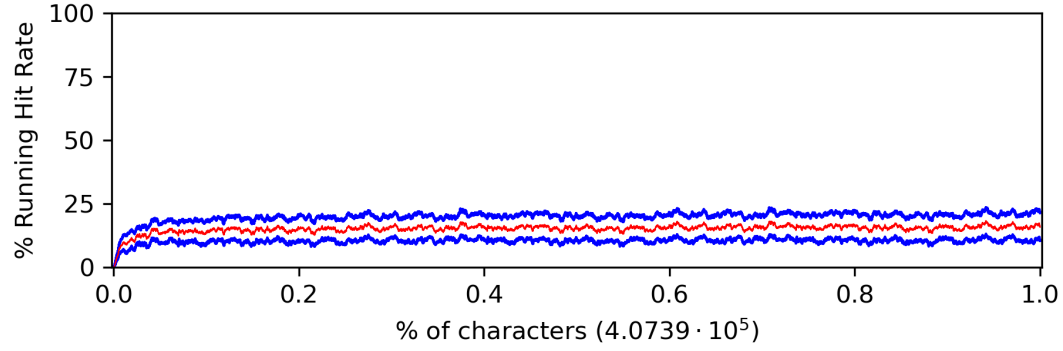


(c)

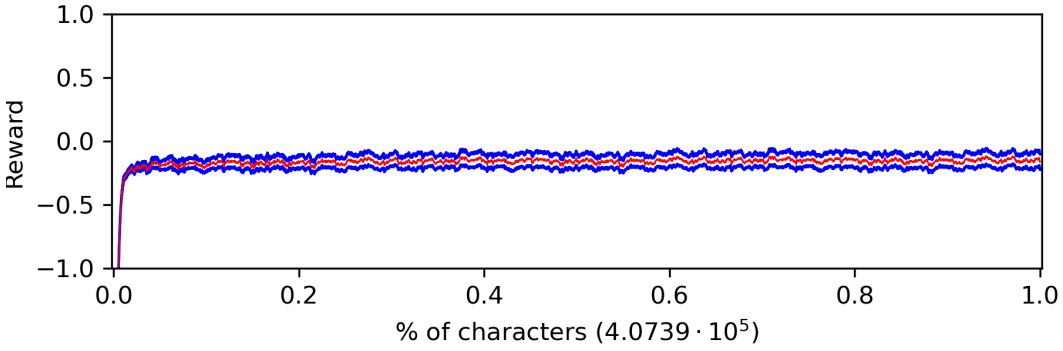
Figure C.1: The out-of-sample results for SARSA where (a) corresponds to the negative values associated with accuracy, (b) corresponds to the positive binary value associated with hit rate, and (c) corresponds to the reward used to evaluate the algorithms overall performance within the range $[1, -1]$.



(a)

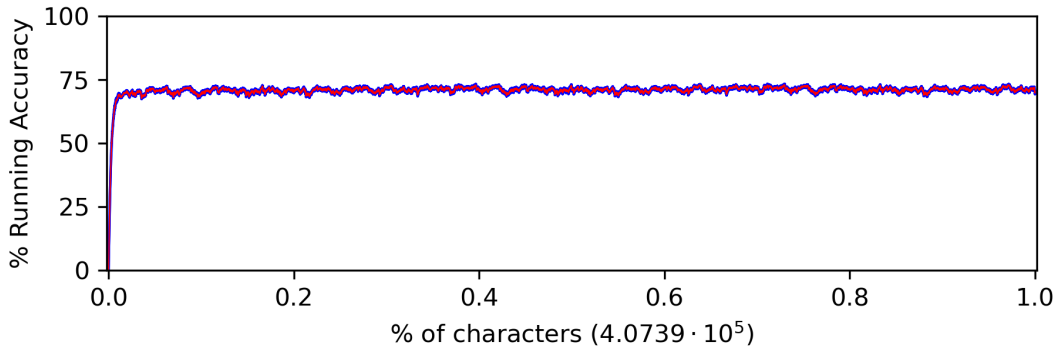


(b)

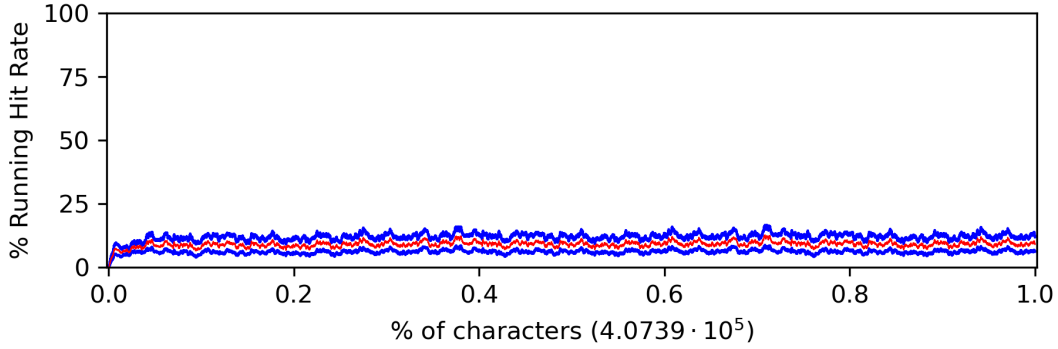


(c)

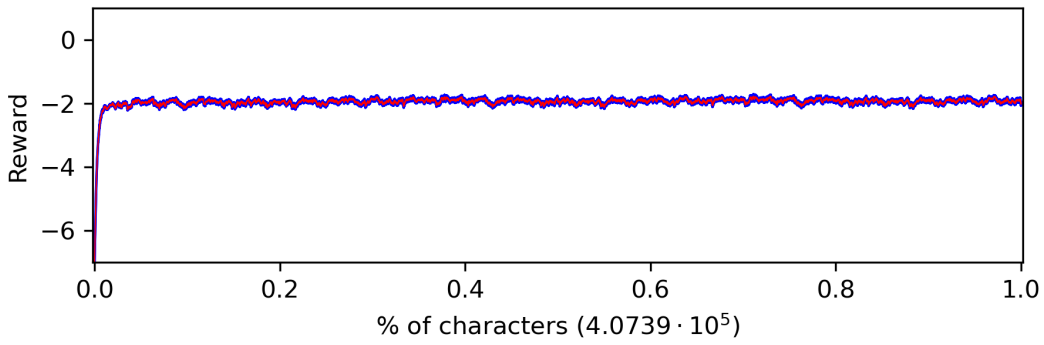
Figure C.2: The out-of-sample results for CLASP where (a) corresponds to the negative values associated with accuracy, (b) corresponds to the positive binary value associated with hit rate, and (c) corresponds to the reward used to evaluate the algorithms overall performance within the range $[1, -1]$.



(a)

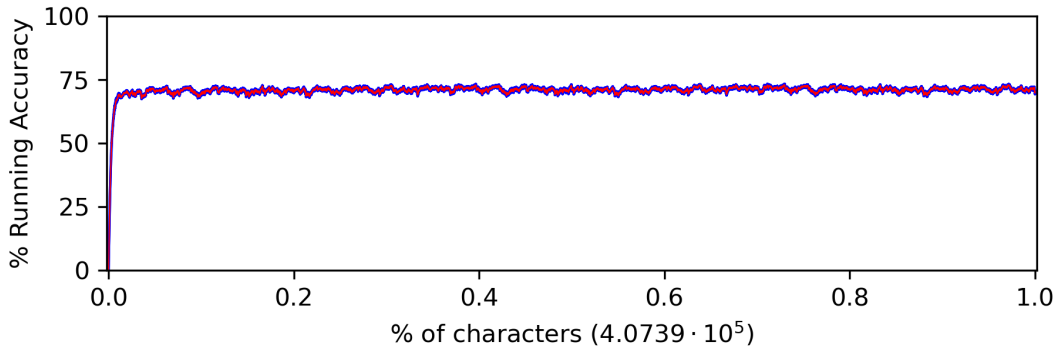


(b)

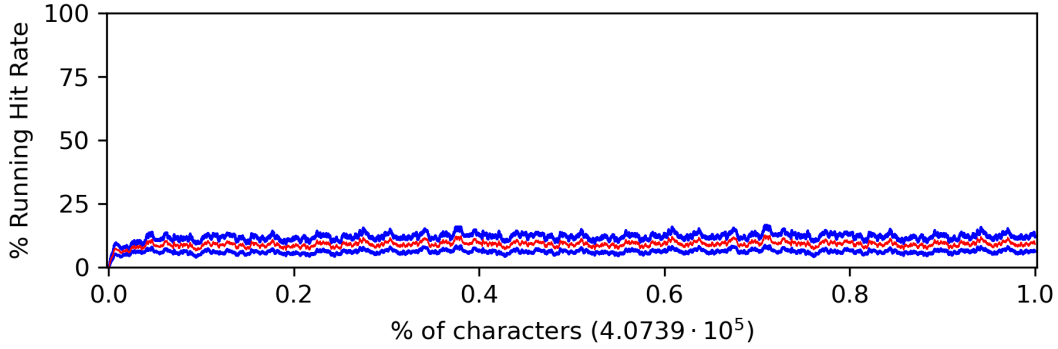


(c)

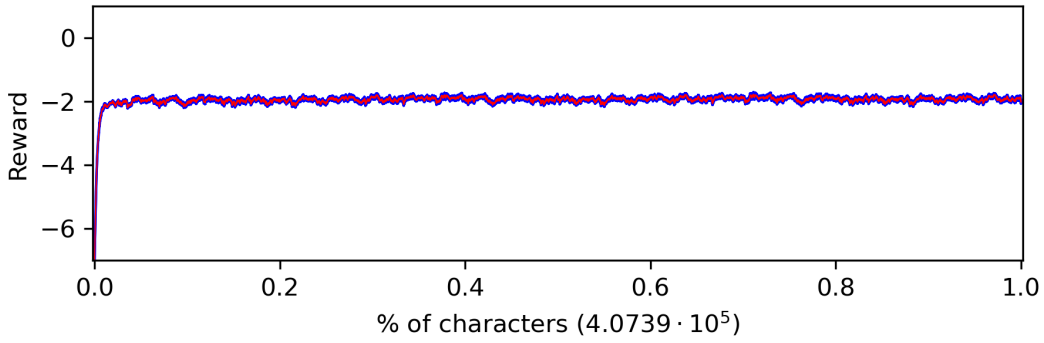
Figure C.3: The out-of-sample results for SARSA where (a) corresponds to the negative values associated with accuracy, (b) corresponds to the positive binary value associated with hit rate, and (c) corresponds to the reward used to evaluate the algorithms overall performance within the range $[1, -7]$.



(a)



(b)



(c)

Figure C.4: The out-of-sample results for CLASP where (a) corresponds to the negative values associated with accuracy, (b) corresponds to the positive binary value associated with hit rate, and (c) corresponds to the reward used to evaluate the algorithms overall performance within the range $[1, -7]$.