Computational Model of Basal Ganglia

Written by David McDougall, 2017

Introduction

This is a computational model of the Basal Ganglia, which is a piece of the brain and is thought to perform Reinforcement Learning. This model is a proof of concept. It accepts input from a Hierarchical Temporal Memory which is a computational model of the Cortex. The model is tested and analyzed. It appears to function correctly however the problem it is tasked with is simple and does not use real world data.

In theory the Basal Ganglia is the part of the brain that decides whether or not to take an action. However the primary output of the Basal Ganglia is to the Thalamus, not the motor cortex which performs the action. The Basal Ganglia is only able to reach the motor cortex indirectly through the Thalamus. I do not have a computational model of the Thalamus and as such I am unable to use the Basal Ganglia to control a motor. Instead I have constructed an artificial problem which tests the Cortex and Basal Ganglia without needing any motors or decisions from the model. In the future I hope to research and model the mechanisms by which the Basal Ganglia executes control over the motor cortex via the Thalamus.

This is not the first computational model of the Basal Ganglia. This work is inspired by the work of (Sungur, August 2017, HIERARCHICAL TEMPORAL MEMORY BASED AUTONOMOUS AGENT FOR PARTIALLY OBSERVABLE VIDEO GAME ENVIRONMENTS).

Model

This section describes how the computer models the Cortex and Basal Ganglia (BG). The Cortex is modeled using a Hierarchical Temporal Memory (HTM), see (Hawkins J and Ahmad S (2016) Why Neurons Have Thousands of Synapses, a Theory of Sequence Memory in Neocortex. Frontiers in Neural Circuits 10:23. doi: 10.3389/fncir.2016.00023) and numenta.org for more information regarding HTMs. The Basal Ganglia has two pieces, the Striatum and the Globus Pallidus (GP). In this model the Striatum processes cortical input and responds to valuable things in the cortex. The GP processes Striatal input and determines the net value of the things which the Striatum finds. The purpose of the Striatum is to take its cortical input and reduce the size of it without losing information which is critical for the GP's operation. The GP uses the temporal difference (TD) method to estimate the expected value of future rewards.

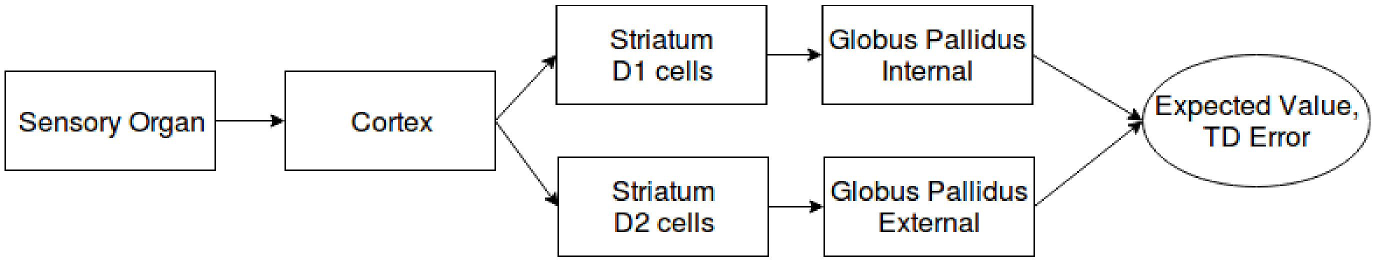

Figure 1: Block Diagram of Model.

The computer executes the following 7 steps in order in a loop:

Step 1) Sample and encode the current sensory input. Receive any rewards which were given in the time since the previous input was sampled.

Step 2) Compute the state of the Cortex. This model uses a single layer of spatial pooling and temporal memory to represent patterns in the input.

Step 3) Compute the state of the Striatum. The Striatum is composed of a type of neuron called a medium spiny neuron (MSN), of which there are two distinct varieties named D1 and D2. MSNs accept cortical input to their dendrites. MSNs use the same dendrite model as basal dendrites on pyramidal neurons in the cortex. MSNs compete for a fixed number of activations, only the most excited neurons are chosen to activate. A MSNs excitement is the number of predictive segments it has. The D1 and D2 MSNs only inhibit their own type which means that they don't compete with each other for activations.

Step 4) Compute the state of the Globus Pallidus (GP). The GP is split into two parts: the external (GPe) and internal (GPi) portions. The GPi accepts input from only D1 MSNs, and the GPe accepts input from only D2 MSNs. GP neuron dendrites work in the same way as do Cortical and Striatal dendrites except that they contain weighted synapses instead of binary synapses. A GP neurons excitement is the number of active segments it has. The GPe and GPi compete as a single population for a fixed number of activations, only the most excited neurons are allowed to activate.

Step 5) Calculate the expected valued and TD error. The expected value is the fraction of GP activations which are GPi cells and not GPe cells. It is calculated using the equation:

[eq1] $\text{Expected Value} = (|GPi| - |GPe|) / (|GPi| + |GPe|)$

Where $|x|$ refers to the number of activations in x. The range of the expected value is [-1, 1].

The TD error is the calculated by the following two equations:

[eq2] $\text{Updated Expected Value}(\text{time} - 1) = \text{Rewards} + \text{Future Discount} * \text{Expected Value}(\text{time})$

[eq3] $\text{TD Error} = \text{Updated Expected Value}(\text{time} - 1) - \text{Expected Value}(\text{time} - 1)$

Step 6) The Globus Pallidus learns from the TD error with the objective of minimizing the TD error in the future. The GP accomplishes this as follows:

Step 6A) The model changes the number of activations in the GPe and GPi such that the new expected value of the GP is the Updated Expected Value(time-1) instead of the Expected Value(time - 1). This step involves calculating the inverse expected value function. In the case where neurons are activated, the most excited inactive neurons are selected. In the case where neurons are inhibited, the least excited active neurons are selected.

Step 6B) GP Neurons which were activated by step 6A learn about their new state. These newly activated neurons reinforce their active segments and initialize new segments if they have too few learning segments. Segments use Hebbian learning, which is to say that segments learn by increasing the connection strength of a synapse if its presynaptic neuron is active and decreasing the connection strength if the presynaptic neuron is inactive. The magnitude of the change is proportional to the magnitude of the TD error.

Step 6C) Neurons which were inhibited by step 6A learn about their new state. Active segments on inhibited neurons weaken their active segments by decreasing the connection strength of synapses with active presynaptic neurons. Synapses with inactive presynaptic neurons are not modified. The magnitude of the change is proportional to the magnitude of the TD error.

Step 7) The Striatum learns from the updated expected value. D1 cells are allowed to learn if the updated expected value is positive, while D2 cells are allowed to learn if the if is negative. In this way the two populations of MSNs learn to represent valuable things in the world.

Results

This section describes how I tested the model. The source code is posted at github.com/ctrl-z-9000-times/HTM_experiments.

Parameter Search

The model has many parameters whose values are determined through trial and error. A simple evolutionary algorithm automates the process of finding optimal values for all of the parameters in the model. The evolution is guided by a linear combination of the following fitness metrics:

1) The RMS of the TD error, divided by the baseline. The baseline is the RMS of the TD error of an agent which always outputs an expected value of zero. The baseline serves to rescale this metric into a more consistent range of values. This metric is minimized.

2) The Temporal Memory sequence classification accuracy. A statistical classifier is trained to classify the current sequence using the Temporal Memory activations. This metric is maximized.

3) The anomaly metric of the temporal memory during predictable sequences. This metric is minimized.

4) The anomaly metric of the temporal memory during the unpredictable filler between sequences. This metric is maximized.

Dataset

The model is tested on artificial datasets which are randomly generated at the programs start. All tests use time series datasets consisting of sequences of inputs leading up to rewards. Each input is mapped to a randomly generated sets of neural activations coming from a hypothetical sensory organ. Each sequence consists 4 to 20 inputs and a reward. The reward is given on the final step of the sequence, all preceding steps have zero rewards. The rewards are uniform random numbers in the range [-1, 1]. The sequences are played in a random order in an infinite loop. 2 to 4 random inputs are inserted between sequences which separates the sequences. The dataset contains 50 sequences. Every iteration of the model (called a step hereafter) consumes a single input. On average, the dataset plays one complete sequence every 15 steps and plays every sequence once every 750 steps. The models parameters were optimized for this dataset and except where otherwise stated, this dataset was used for all experiments in this report.

The Basal Ganglia learns the expected value function.

The primary output of this model is the expected value of future rewards. The model was run for 20,000 steps and produced the following two figures.

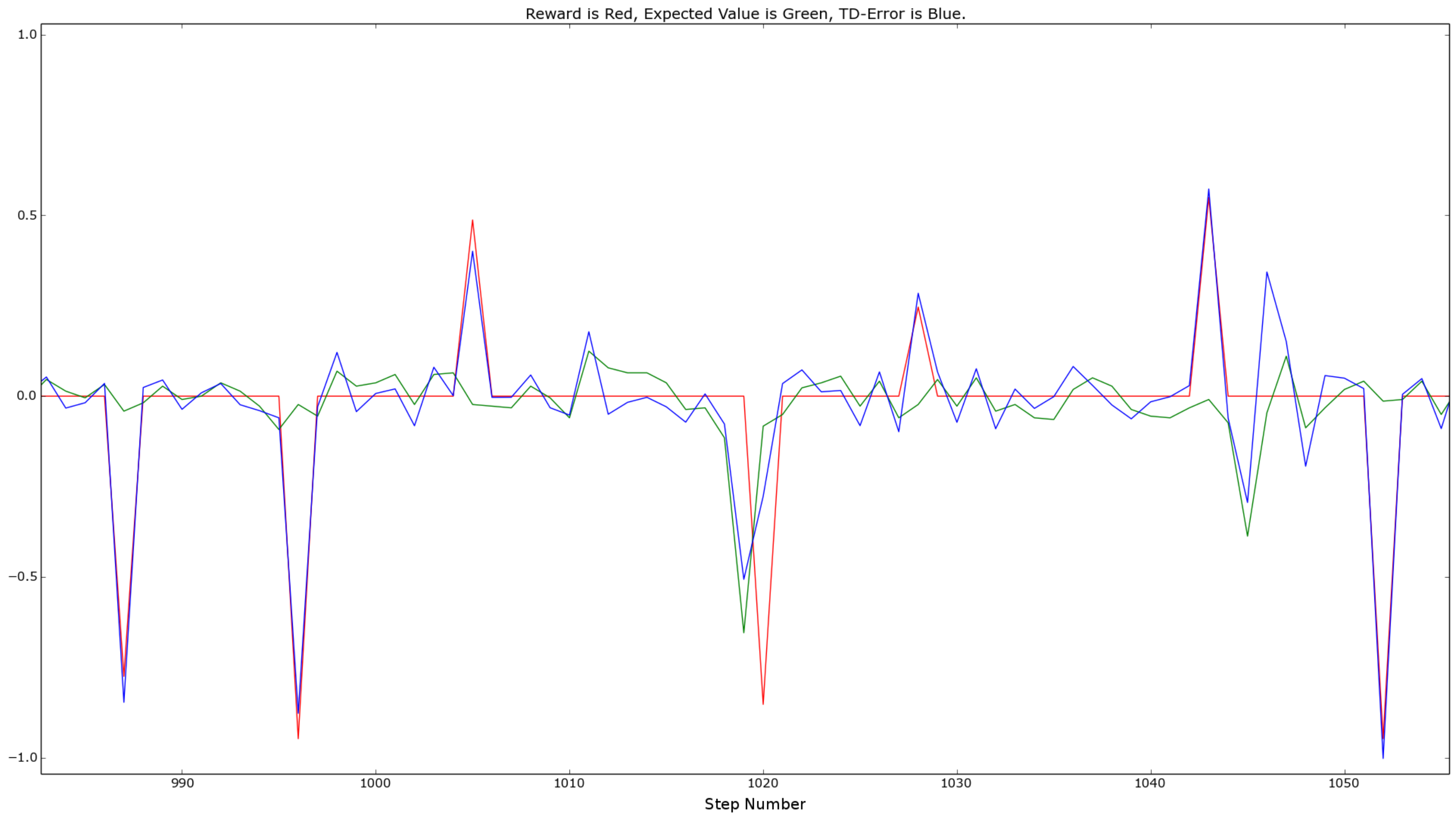

Figure 2: Untrained behavior. The BG successfully predicted the reward (red line) given at step number 1020, which was preceded by the expected value (green line). The BG failed to predict all other rewards in this figure.

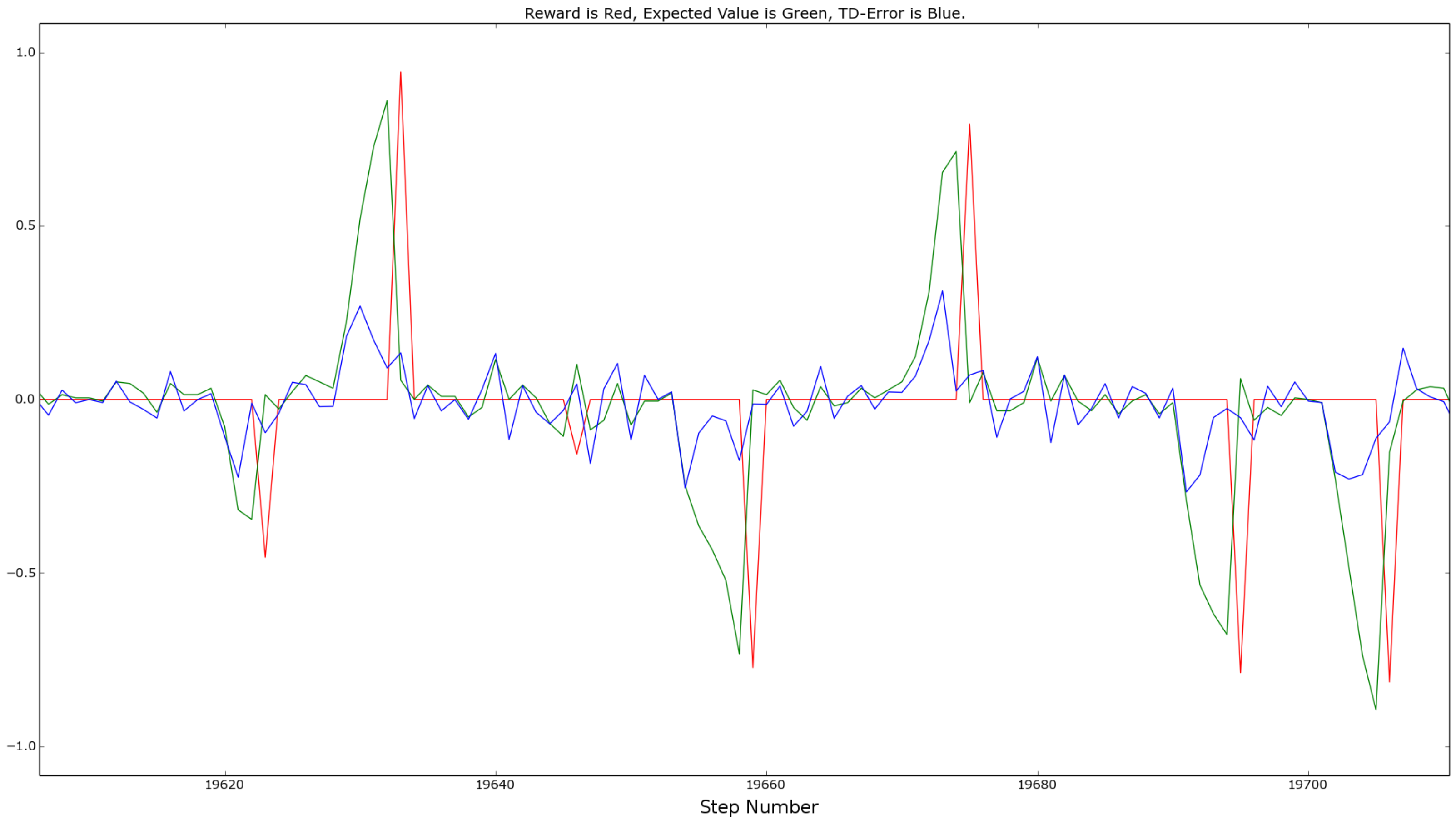

Figure 3: Trained behavior. The BG correctly predicted all of the large rewards. It is unclear if the small reward at step number 19645 was predicted.

The Striatum learns valuable sequences.

In order to see what the Striatum does and does not know, statistical classifiers are trained to use D1 or D2 activations to recognize the current sequence. The sequence classification accuracy reflects the number of MSNs which respond to the sequence. A low accuracy will not necessarily cause issues for the GP but zero accuracy will. The model was run for 20,000 steps and produced the following figure.

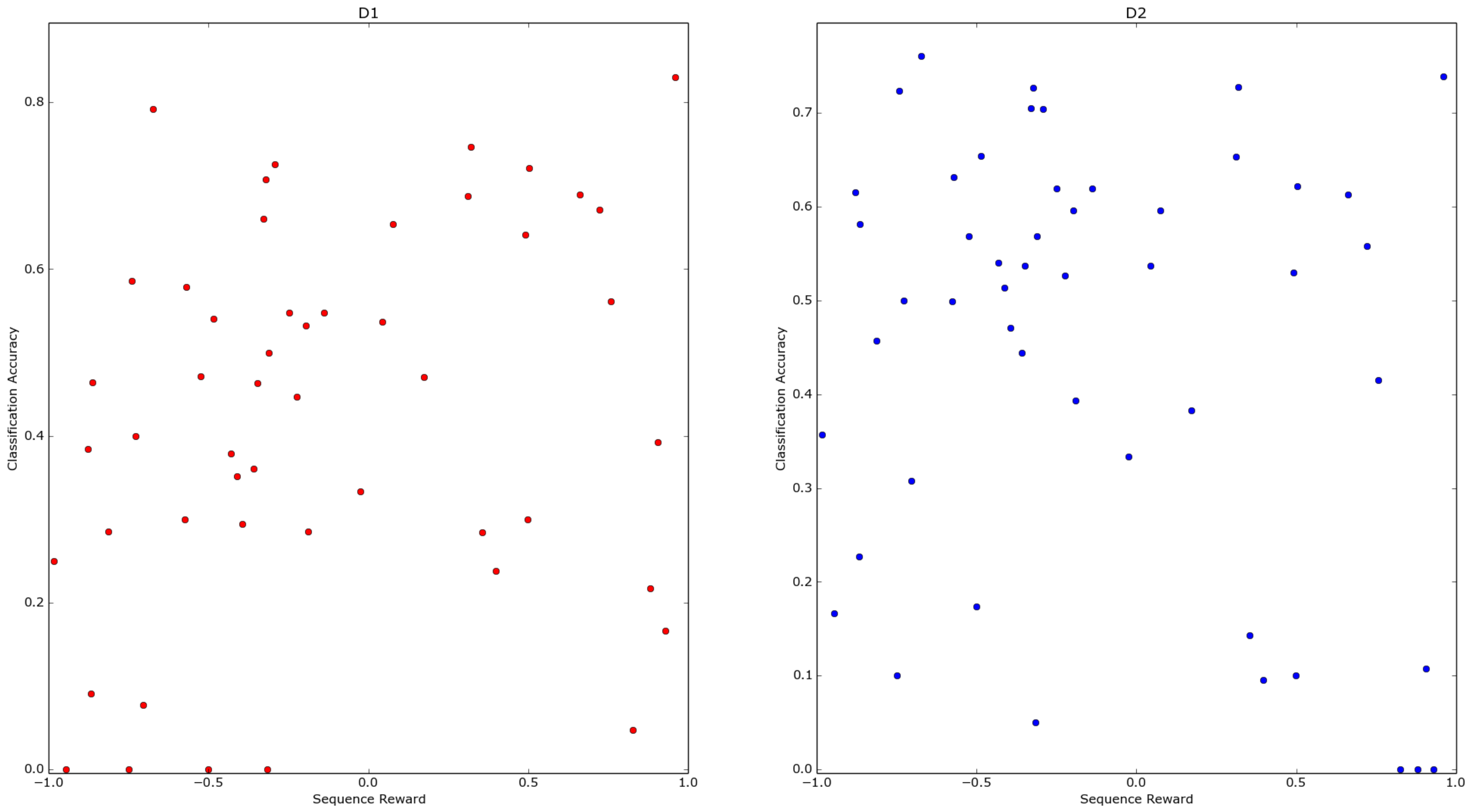

Figure 4: Striatum vs Reward. This figure shows that D1 and D2 neurons learn sequences with positive and negative values (respectively). D1 and D2 neurons have learned some but not all sequences which the other population is tasked with representing.

The Globus Pallidus learns sequences values.

The purpose of this experiment is to isolate learning to the GP. This is accomplished by taking an existing model and modifying its dataset by changing all reward magnitudes (but not the signs) to a new uniform random number. Since the sign of the value of each sequence has not changed, each sequence still needs to be represented by the same population in the Striatum. This experiment was performed on the same instance of the model which created figure 4. After figure 4 was created the reward magnitudes were changed, the model was run for another 20,000 steps, and it produced the following two figures.

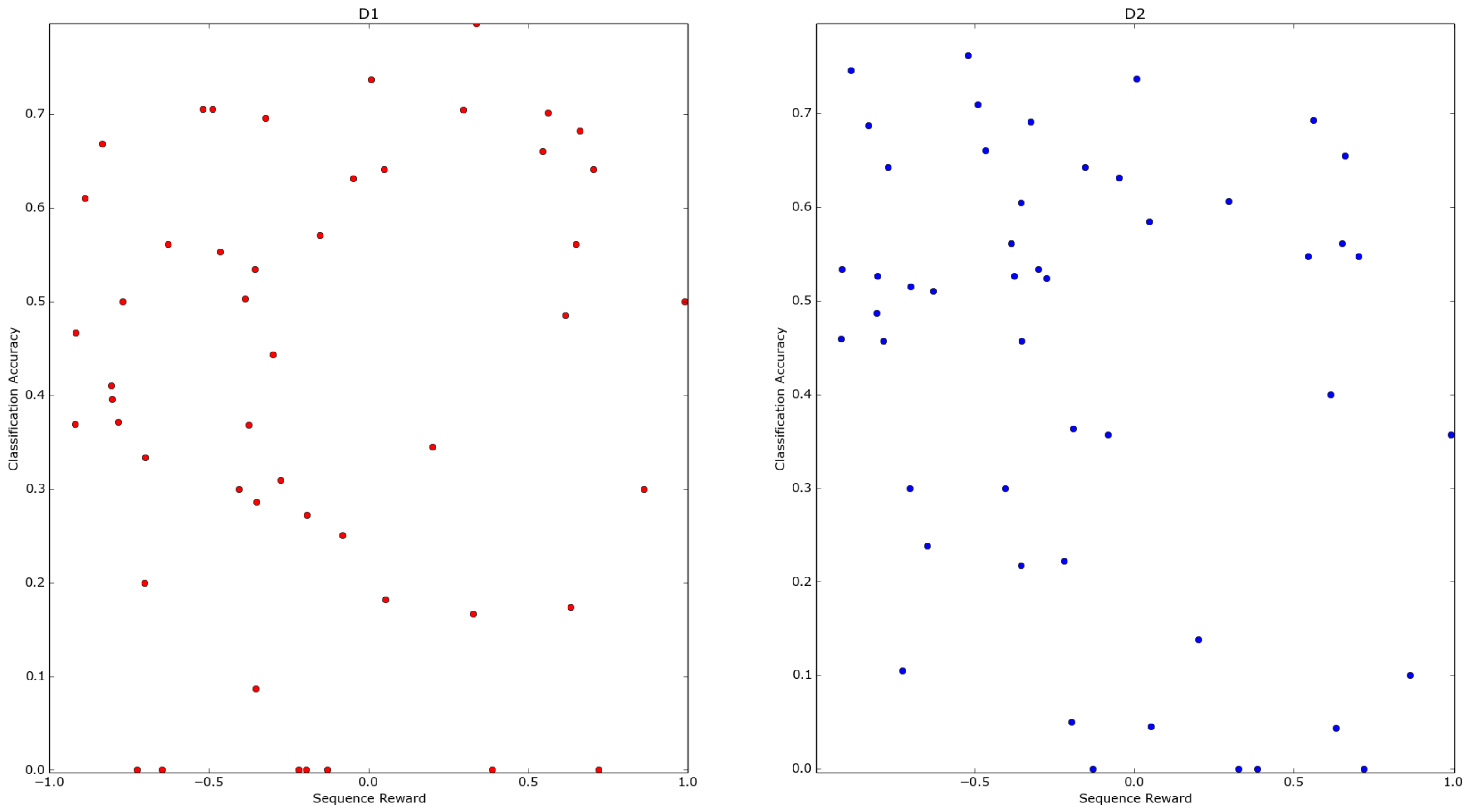

Figure 5: Striatum vs Reward, after changing reward magnitudes and retraining. What each population knows and doesn't know is mostly unchanged from before. Several data points fared worse after additional training, which is an unexplained issue.

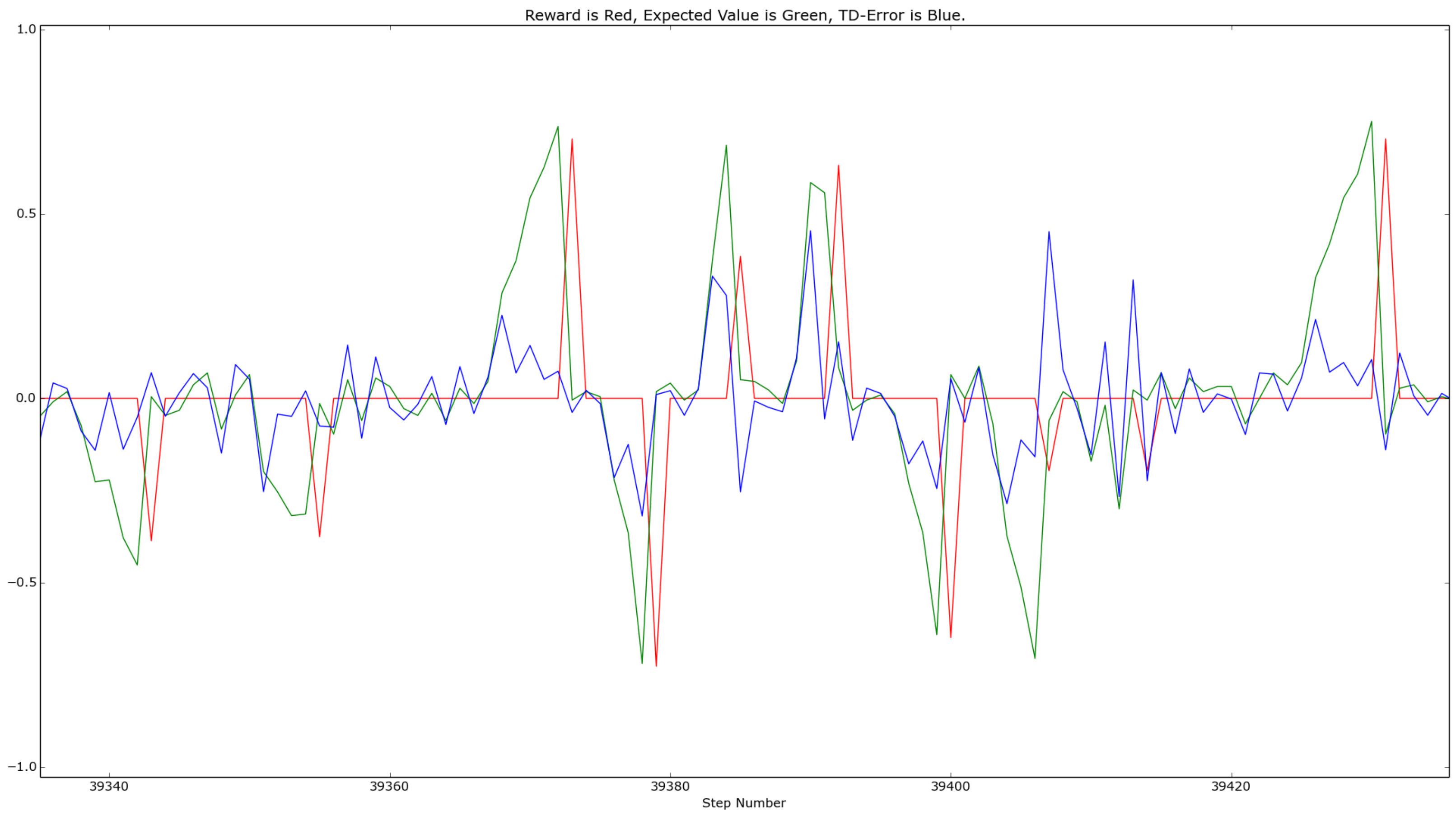

Figure 6: The GP has adjusted in response to the rewards changing magnitude. Notice that the GP incorrectly predicted the magnitude of rewards at step 39385 and 39408 and it failed to predict the reward at step 39415. The 7 other rewards shown are correctly predicted.

Model Capacity Experiment.

The purpose of this experiment is to determine how many different things the BG can learn. To this end, the number of sequences in the dataset is systematically changed. The model was run for 400 steps per sequence in its dataset and produced the following table.

Number of Sequence	Number of Steps	TM Classification Accuracy	RMS TD Error, Measured / Baseline = Score
25	10,000	0.7285	0.121407 / 0.162398 = 0.7475
50	20,000	0.6951	0.118016 / 0.164056 = 0.7193
75	30,000	0.6774	0.109295 / 0.143401 = 0.7621
100	40,000	0.6462	0.123619 / 0.167526 = 0.7379
150	60,000	0.5992	0.117043 / 0.153766 = 0.7611
200	80,000	0.5735	0.116667 / 0.149121 = 0.7823
300	120,000	0.5229	0.122871 / 0.158770 = 0.7738
400	160,000	0.4782	0.125451 / 0.157042 = 0.7988

500	200,000	0.3924	0.129353 / 0.156192 = 0.8281
600	240,000	0.2798	0.133169 / 0.152568 = 0.8728
700	280,000	0.1931	0.142850 / 0.156322 = 0.9138
800	320,000	0.1245	0.149613 / 0.154589 = 0.9678
900	360,000	0.08715	0.159435 / 0.154235 = 1.0337
1000	400,000	0.05979	0.168212 / 0.155019 = 1.0851

Table 1: Sequence Capacity Results. The models performance decreased as the size of the dataset increased. It performs well when tasked with more data than it was optimized for. The models parameters were optimized for 50 sequences.

Cell Death Experiment.

This experiment verifies that the model can withstand some of its cells dying. For this experiment, all cells in the TM, Striatum, and GP have the same probability of dying. The model was run for 40,000 steps and at step number 20,000 a fraction of its cells were killed. The following table shows the result of a single run of the model at various cell death percentages.

Cell Death	TM Classification Accuracy	RMS TD Error, Measured / Baseline = Score
0 %	0.7256	0.106169 / 0.159591 = 0.6652
5 %	0.7088	0.102747 / 0.144989 = 0.7086
10 %	0.7102	0.108201 / 0.165675 = 0.6530
15 %	0.7039	0.112893 / 0.160865 = 0.7017
20 %	0.7056	0.115260 / 0.154725 = 0.7449
25 %	0.7026	0.112065 / 0.162948 = 0.6877
30 %	0.7122	0.114592 / 0.152431 = 0.7517
35 %	0.6865	0.118236 / 0.145238 = 0.8140
40 %	0.7148	0.118947 / 0.144206 = 0.8248
45 %	0.7016	0.123064 / 0.138494 = 0.8885
50 %	0.6643	0.211897 / 0.185994 = 1.1392
75 %	0.4795	0.243096 / 0.130531 = 1.8623
95 %	0.6445	0.204631 / 0.154466 = 1.3247

Table 2: Cell Death Results. A cell death of 25% or less caused little to no loss in functionality. Between 25% and 50% cell death performance deteriorated and beyond 50% cell death the model performed worse than the baseline.

Works cited

Hawkins J and Ahmad S (2016) Why Neurons Have Thousands of Synapses, a Theory of Sequence Memory in Neocortex. Frontiers in Neural Circuits 10:23. doi: 10.3389/fncir.2016.00023

Sungur, August 2017, HIERARCHICAL TEMPORAL MEMORY BASED AUTONOMOUS AGENT FOR PARTIALLY OBSERVABLE VIDEO GAME ENVIRONMENTS