Contents

1 Guest Lecture

What region of the brain is doing motor functions autonomously?

1.1 What could generate the "wiggles"

We can take a dynamical system, in this case a recurrent neural network, which is a collection of units that interact with each other with a linearity. The network itself allows in a small amount of input (start and stop signals), and along with a slight variation of the classical back propagation algorithm that is used to train most neural networks, achieves great results for some scenarios but not for others. The autonomous system that satisfies changes i n a specific variable is a function of itself.

1.2 Dynamical Systems in the Brain

We come to see that dynamical systems have limitations that are worth noting. For one, no autonomous dynamical system can satisfy a sine and cosine wave and nothing else. The reason for this is that if you plot both functions in time as functions of themselves, the

dynamical system would create an infinite loop that was shown to make a figure eight, the point where the two lines intersect is problematic. We consider this system a "tangled" dynamical system. In order to "untangle" a dynamical system that has a figure eight, you need to introduce a third signal, which would allow the system as a whole to "lift" the crossing point in three dimensions. Researchers have used this model to study dynamical systems in the brain, specifically neural systems that are autonomous, ie. their output is a system that only depends on itself in isolation. If a neural system exhibits the figure eight pattern, then they are being fed information from some outside entity, but "lifted" areas are interesting to study because it means that some part of the system is producing information autonomously.

1.3 Experiments

In order to test autonomous systems in the brain, researchers set up an experiment where they trained a monkey to visually identify a goal and use a crank to move themselves toward the goal for a reward. During the task they noticed that the muscles that move the arm, which are part of the spinal cord neurons, exhibit figure 8 patterns and thus are not an autonomous system. Researchers then concluded that the motor cortex neurons provide the "lifting" within the system. Thus, the neurons in the motor cortex are autonomous dynamical systems. However as we see in the visualization, the cycles observed in the motor cortex sit on top of each other, so there is no concept of timing the cycles - the system requires one pulse to tell it to start, and another to tell it to stop. However, when we look in the supplementary motor area, we see that the cycles no longer sit on top of one another, they spiral back up in a helix. So, the depth of the start pulse decides how many cycles there will be since each cycle will move back towards the starting point at a fairly constant rate. Thus in the supplementary motor area we have an autonomous dynamical system that requires only a start signal.

1.4 Conclusions

In conclusion, it was restated that studying autonomous dynamical systems involves studying whether a group of neurons where being driven by some outside force or not. The motor neurons in the spinal cord do not have the structure required to be autonomous, but the motor cortex did relying on start and stop signals. The supplementary motor cortex displayed minimal autonomous behavior that only required a start signal. In terms of thinking about artificial neural networks, using findings within autonomous neural systems could aid in improving performance or providing a better perspective in which we can study recurrent neural networks.

2 Today

(a) Assembly Hypothesis

(b) Reinforcement Learning

(c) Conditioning vs Temporal differences

(d) Reinforcement Learning in the brain

(e) The math of RL

(f) Deep RL and Alpha GO

3 Assembly Hypothesis

The basic operations are they plausible and the similarities of assemblies is preserved under projection. Furthermore, there is an algorithmic mode that is Turing complete, a patter completion mode that is associative and uses probabilistic computation. Finally, they can implement syntax trees in the brain. These modes improve and extend simulations and there are still things about them that we are unsure about such as can they do predictive learning.

4 Reinforcement Learning in the Brain

The brain also exhibits parallels to reinforcement learning, since we are constantly learning about the world through reward systems. These rewards are always a constant struggle between being positive or negative. Prior to reinforcement learning, researchers used conditioning in order to study how past rewards effect animal expectations in certain scenarios.

5 Conditioning

Classical conditioning, also known as Pavlovian conditioning, is a learning procedure a potent stimuli, such as food, is paired with a previously neutral stimuli, such as ringing a bell. In classical conditioning experiments, animals learn to expect a reward when performing specific neutral actions.

5.1 Examples

The different forms of classical conditioning are the following:

- Pavlovian
- Extinction

- Partial
- Blocking
- Inhibition
- Overshadowing
- Aristotelian

For example extinction does not get reward even though it performs the task and blocking is when two actions together lead to a reward, in isolation one of them does not

5.2 Rescola-Wagner Update (Delta Rule)

The Rescola-Wagner model is a model of classical conditioning, in which how an agent learns is models in terms of the associations between conditioned and unconditioned stimuli. The delta rule is as follows: There exists a stimulus u, prediction $x = u \cdot w$, weights w, reward R. Weights are computed as follows:

$$x = u \cdot w$$

Rescola-Wagner Plasticity:

$$w \to w + \varepsilon(R - x) \cdot w$$

Gradient Descent Error:

$$2(R - x)^2$$

$R - x$ explains classical conditioning.

A caveat to this model is that it does not take into account timing. Animals make decisions based on present and future behaviours. The Rescola-Wagner formulation can be improved by adding a time constraint to the update rule. Let stimulus $u(t)$, prediction $x(t)$, weights $w(t)$. The update rule becomes $x(t+1) = R(t) + \gamma x(t)$. The downside is that we get this by solving a "wishful thinking" equation. Classical conditioning is the case where $\gamma = 0$. We can minimize δ^2 using gradient descent.

This method was used in TD-Gammon (Tesauro 1992), an automaton designed to play backgammon using temporal differences. One could think of it as the predecessor to Alpha Go.

5.3 Sutton and Barto Reinforcement Learning

Same set up as before, but x is different as it is predicted value of all future rewards. xt = ut wt x(t)=R(t+1)+gammaR(t+2)+gamma2 R(T +3) This discounts the future based on how far it is. There was another model but it was wishful thinking, it will not be correct, and the deviation is called delta. So they take the equation delta = R(t) + gammax(t) x(t

+ 1) This minimizes Delta squared via gradient descent. This is the temporal difference algorithm. It allows the neural network to predict future reward. Get a stimulus at time 100 and reward at time 200, with a gamma of 1. The animal will learn that there is a reward (correctly predicts the future rewards)

5.4 Conditional Learning in the Brain

The brain uses Pavlovian learning from stimuli, where you learn from your own actions and dopamine serves as the reward.

6 Reinforcement Learning Math

6.1 Multiarmed Bandits

In multiarmed bandits, you have m actions, each with an unknown reward distribution. You can take two different routes, either exploration or exploitation, where one machine has the best expectation, at any time you can only choose one, and you want to minimize your regret overall. You keep trying the machines, evaluating their mean by sampling. You can take a greedy approach, where the regret is linear in T, or you can take a noisy approach, which unfortunately is also linear in T.

6.2 Is linear regret inevitable?

Linear regret is not inevitable, however, we can punish machines that have a good sample mean but has a low number of samples. We punish using the Hoeffding bound. Achieving log linear regret is exhibited in the UCB1 algorithm.

6.3 Can we do better?

An even better algorithm uses posterior sample, and maintains a parameterized model of reward distributions. You pick a machine based on sampling the reward distributions according to the current parameters, and pick the best action. This implementation is called Thompson's algorithm, which achieves a lower log bound than UCB1.