Spiking Neural Networks as a Model for the Nervous System of *Hydra*

1 Introduction

An artificial neural network is a simplification of a biological neural network, generally based on a directed weighted graph. The vertices of the graph represent neurons, and the edges represent connections between them. Typically, these artificial neurons are only vaguely inspired by biological neurons, and therefore are unsuitable for use in modeling biological nerve networks. We hope to work past some of the biological limitations of traditional artificial neural networks using what are known as spiking neural networks, and aim to apply them in creating a model of the very primitive nervous system of the metazoans *Hydra*.

2 The nervous system of *Hydra*

Hydra is a genus of small, freshwater cnidarians which have long been studied in laboratories, due to both their ease of culturing and small size (typically 10 mm in length). They have one of the simplest nervous systems known among metazoans, yet they are still capable of nontrivial behaviors, unlike more basal clades such as *Porifera* (sponges). Due to this simplicity, as well as the large knowledge base surrounding *Hydra*, we aim to develop a basic model of the *Hydra* nervous system.

Like other cnidarians, *Hydra* do not have a central nervous system, but rather a nerve net, which is characterized by neurons distributed throughout the body with little centralization (i.e. no brain). That said, the neurons are still not distributed homogeneously across the body of *Hydra*, but assuming as much is a decent approximation. Extending our model to consider more complex neuronal distributions would be an interesting topic for further exploration.

Being such primitive organisms, *Hydra* have a very limited set of behaviors. *Hydra* are capable of movement, but they are primary sessile, with their primary behaviors being elongation and contraction. Interestingly, Han et al., 2018, examined freely behaving *Hydra* and found that the specific set of behaviors observed is independent of environmental conditions. Dupre and Yuste, 2017, completed the first ever full nervous system activity map using *Hydra*, and found that the primary behaviors of *Hydra* are controlled by disjoint circuits in the nerve net. That is, each neuron belongs to a single connected group of neurons (a circuit), and each circuit controls a specific behavior. However, at least two of the circuits have an antagonistic relationship, so they are not completely independent of each other.

We plan to use a spiking neural network as the basis for our model. Unfortunately, we do not have any numerical data to use, so we will have to train the network on estimated data points. The exact values should not matter as long as orders of magnitude are consistent across the data set – all that should matter is correlation. For example, as mechanical pressure is applied to the tentacles, the *Hydra* should contract longitudinally (as observed experimentally), and this negative correlation should be reflected in the data set.

3 Spiking Neural Networks

Traditional neural networks used today in deep learning have their neurons fire only during what is known as a propagation cycle. That is, if the neural network is to act as an agent in some environment, then the timesteps must be discrete. On one hand, making the simplification that neurons fire only during a propagation cycle makes the simulation of networks much easier, and is perfectly fine for applications such as image recognition. However, it is far less realistic when trying to simulate an actual biological nervous systems which runs on a continuous time scale.

Spiking neural networks aim to solve this problem in part. Firstly, this type of neural network includes continuous time as a factor in the model. This leads artificial neurons to behave more like biological neurons, only firing once a predetermined 'membrane potential' has been reached. If the input is not strong enough, the current potential will degrade over time. If the neuron fires, the signal travels to other neurons but not instantaneously as in traditional networks. Finally, a refractory period can be added so that the neuron does not fire immediately after having fired, giving it a sort of cool down period as is observed in biological neurons. All these properties together make spiking neural networks a better choice for our model than any other traditional neural network design.

4 Appendix: Optimization through graph tensor products

We assume that the graph structure of a biological neural network at some time t can be modeled by a time dependent function $\mathbb{M}$ that transforms the edge and vertex sets of an initial graph G in some fashion. Further, we will model networks under the assumption that the initial structure of G before any edge weights are modified can be approximated by the tensor product of other graphs:

$$G = (H_1 \otimes H_2 \otimes \cdots \otimes H_L)$$

The reason for this assumption is as follows. For the tensor product $C = A \otimes B$ of two graphs A, B with $|V_A| = h, |V_B| = k$, the size of the vertex set of C is $h \cdot k$. However, we can simulate a random walk on C with only $(h + k)$ memory for the vertices, despite the vastly increased complexity of the graph. This is a fundamental property of graph tensor products. For our applications, of course, we are not trying to simulate a random walk, but there are computational savings nonetheless. See section 4.1 for more details.

An important distinction is that a neural networks changes with time as it learns, and this is where our function $\mathbb{M} : (G, t) \mapsto G_t$ comes into play. In most artificial feed-forward neural networks, the function would simply modify the weights of the edges of G in accordance to some fitness function. However, for our purposes, modification of the underlying product components $H_1, H_2, \ldots, H_L$ may be advantageous, particularly in the early stages of learning. This is because modifications to the underlying product components create large changes in the structure of the network at little computational expense.

4.1 Code notes

- When storing the graph G, we store the tensor product components $H_1, H_2, \dots, H_L$ where L is the number of layers. To specify a vertex v in G, we store in an L-element array the vertices $v_i \in H_i$ to which v corresponds to.
- To store an edge e that is an addition to G or the change of an existing edge weight in Gs we store the vertices v_1, v_2 which it connects, and store the edge in a data structure sorted by the input vertex v_1. Given a vertex v_1, this lets us easily access the edges leading away from v_1.
- To avoid having the repetitive weights from the original graph G, the weights of $M(G, 0)$ are determined by hash function until the vast majority of edges are changed, and then we begin to store the edges directly.

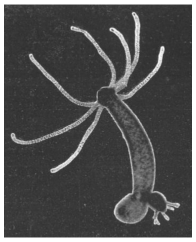
5 Artifical Neural Networks

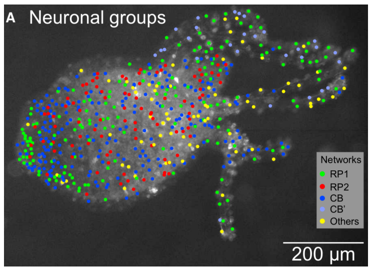
5.1 Artificial Neural Network (ANN) Introduction

- Neurons represented by vertices in a weighted directed graph
- Output of a neuron is a function of the weighted sum of inputs
- Usually have multiple hidden layers of neurons
- Learning accomplished through modifying edge weights according to some algorithm
- Originally inspired by biological nervous systems, but most implementations exchange biological accuracy for simplicity and computational efficiency

5.2 Spiking Neural Network (SNN) Introduction

- More closely resemble biological neural nets than other ANNs
- Fire not during a propagation cycle, but when a specified 'threshold membrane potential' crossed (recall Hodkin-Huxley)
- A neuron's state is modeled by a differential equation that depends on input strength and as well as timing
- More promise compared to other types of ANNs, but much more computationally intensive to both train and run
- It is thought a primary advantage of SNNs comes from their ability to encode information in the frequency/rate of pulses and not just in their intensity, as with traditional networks





6 Biology of Hydra vulgaris

6.1 Evolutionary History of H. vulgaris

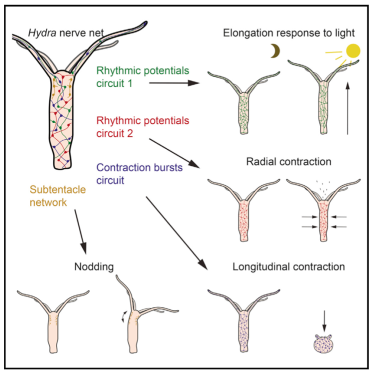
- *Hydra* are small, freshwater hydrozoans (family Cnidaria)
- Believed to have originated around 60 Mya
- Cnidarians first appeared around 580 Mya, haven't changed much since
- Very small repertoire of behaviors which is consistent across individuals and environments

6.2 Nerve Nets

- *Hydra* have a diffuse nerve net rather than a CNS
- Comprised of ganglia and a few types of sensory neurons
- Once mature, a constant density gradient of neurons is maintained. Adults have no more than a few thousand neurons
- Turns out nerve net is actually composed of non-overlapping circuits that correspond to specific sets of behaviors

6.3 Separate Circuits

- RP1 associated with longitudinal elongations
- RP2 associated with radial contractions
- CB associated with longitudinal contractions


Hydra nerve net
Elongation response to light
Rhythmic potentials circuit 1
Rhythmic potentials circuit 2
Contraction bursts circuit
Subtentacle network
Radial contraction
Nodding
Longitudinal contraction

- STN associated with ‘nodding’ behavior
- RP1 and CB are antagonistic, no other interactions

6.4 Summary

Advantages of *Hydra* as a model organism:

- Very small size, so currents travel quickly and therefore the topology of the nerve net can be ignored at first approximation
- Distinct behaviors appear to be controlled by distinct circuits, so analysis can be broken into many parts
- Small amount of neurons and fixed neuronal density make simulation much easier

7 Network Framework

7.1 Neuron Model

- Leaky integrate-and-fire (LIF) model:
$$\frac{dV_m}{dt} = \frac{1}{C_m}\left(-\frac{(V_m - V_m^{eq})}{R_m} + I_{ext}\right)$$
- Computationally simpler than Hodgkin-Huxley
- Models neuron as RC circuit with leak term
- Doesn’t explicitly specify spiking behavior or refractory period, but easy to implement using iterative ODE methods
- Possible implementation:

```
[fontsize=\scriptsize]
if V(t+1) > threshold:
   V(t)   <- spike
   V(t+1) <- hyperpolarize
if t in refractory period:
   V(t+1) <- hyperpolarize
```

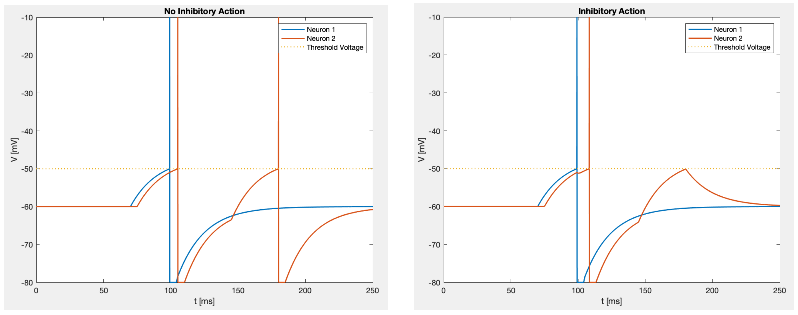


Figure 1: Simulation of two LIF neurons not acting on each other, and acting antagonistically, respectively. Input current is same in both simulations.

7.2 Antagonistic Neural Circuits

- Assume each neuron of RP1 emits an inhibitory neurotransmitter E_{RP1} when spiking, and similarly for CB with E_{CB}. Using concentrations, the model is:

$$\frac{dV_{RP1}}{dt} = \frac{1}{C_{RP1}}\left(-\frac{(V_{RP1} - V_{RP1}^{eq})}{R_{RP1}} + I_{ext}\,(1 - E_{CB})\right)$$
$$\frac{dV_{CB}}{dt} = \frac{1}{C_{CB}}\left(-\frac{(V_{CB} - V_{CB}^{eq})}{R_{CB}} + I_{ext}\,(1 - E_{RP1})\right)$$
$$\frac{dE_{RP1}}{dt} = d_{RP1}E_{RP1} \qquad \frac{dE_{CB}}{dt} = d_{CB}E_{CB}$$

7.3 Antagonistic Neural Circuits (cont.)

8 Optimizations

8.1 SNN simplifications

In our nerve network model, instead of fully simulating a continuous waveform for the state of the neuron, we keep track of the spikes at discrete timesteps instead. The primary advantage of spiked neural networks comes from the temporal coding, and this can be preserved with small enough discrete timesteps.

One advantage of this approach is that the timestep size can be decreased if we find that information is not being encoded temporally as expected, or can be increased to save on processing power.

8.2 Tensor Product Optimization

We propose imposing non self-similar structure on the "blank" (before learning) nerve nets in the following fashion:
Where $H_1, H_2, \dots, H_L$ are all graphs, we set

$$G = H_1 \otimes H_2 \otimes \cdots \otimes H_L$$

Where "$\otimes$" is the graph tensor product, equivalent to the matrix tensor product of the adjacency matrices of the graphs.
Tensor product $H \otimes K$ is equivalent to replacing every vertex in H with a copy of K. Vertices have connections between them if their counterparts in either H or K have edges between them. This means a random walk on $H \otimes K$ can be simulated by a random walk on H and K at the same time.

8.3 Tensor Product Optimization

- Note that the space required to store a graph as an adjacency matrix is $O(n^2)$ on the number of vertices
- If we assume H_i has n_i vertices, then we only require $\sum_{i \in [1,L]} (n_i)^2$ space to store the component graphs
- On the other hand, storing G by itself requires space $\prod_{i \in [1,L]} (n_i)^2$

8.4 Tensor Product Optimization, continued

For large networks, the aforementioned size differences become very significant

- Network with 5 layers of 100 neurons each will take only $5 \cdot 10^4$ units of memory to store by adjacency matrices
- the above 5-layer network describes a complex network with $100^5 = 10^{10}$ neurons that would take an adjacency matrix of size 10^{20} to describe
- assuming 1 byte memory per connection (very low estimate), the 5-layer network takes up about 50MB, while it's counterpart takes around a million TB to store directly.
- Simulating and training this network is still computationally difficult, but space complexity of is reduced

8.5 Biological Motivation

Making no assumptions about which genes affect the structure of the human brain, it is clearly impossible for the exact structure of the $> 10^{10}$ neurons and $> 10^{12}$ connections between them to be encoded in the mere $4 \cdot 10^9$ base pairs of the human genome.
This tells us that there must be *some* sort of repeated structure in most nerve networks. Granted, graph tensor

products might not be the most realistic way to model this repeated structure, but we hope that our work is a small step in that direction.

9 Future Work

9.1 Short term goals

- Using existing data in the literature, finish creating out model of the *Hydra* antagonistic nerve nets
- After creating a network that exhibits temporal signalling behavior, we hope to find the largest timestep Δt that preserves this behavior. Knowing the largest possible timestep lets us run simulations faster, and grants insights about the nature of the temporal signaling.
- Extend the tensor product optimization to allow replacement of vertices in H_i with different graphs, not just H_{i+1}. This will in theory allow the representation of a wider variety of graph patterns, notably allowing differences in local structure between larger regions

9.2 Long term future work

- Train the framework described here for *Hydra* on actual behavioral data from specimens. With sufficient training, the network described should be able to, in more or less real time, predict the behavior of *Hydra* in response to external stimuli.
- Chemical signalling in the human brain plays a far more complex role than the relatively simple antagonistic networks in *Hydra*. Extending our model to the human brain is far outside the scope of this project or modern technology, but perhaps some insight into chemical signalling systems in nerve nets can be gained.
- With more information about the nature of the temporal signalling, it may be possible to model the frequency domain instead of the time domain. This would be both easier for a human to read, and less computationally expensive.