Summary

My project has two main parts which I introduce separately. The first part involves wrapping up a project I worked on last semester, PsychRNN: An open source Python package for cognitive task modeling using recurrent neural networks. The second part involves using PsychRNN to investigate the strategies and limitations of fixed- and fast-weight neural networks when trained on a task previously investigated in monkeys.

PsychRNN: An open source Python package for cognitive task modeling using recurrent neural networks

Motivation

Modeling using recurrent neural networks (RNNs) is quickly becoming a popular strategy to probe the mechanisms behind cognitive tasks. RNNs are able to accomplish many popular cognitive tasks but are vastly more manipulable and observable than other classic model organisms such as monkeys and humans. Modeling is a fast, inexpensive way to investigate the abilities and limitations of proposed mechanisms behind cognitive tasks, and can lead to deeper understanding of current proposed mechanisms, as well as propose new mechanistic hypotheses. These insights can in turn guide experimental focus to better distinguish between current proposals.

Prior Work

In Fall 2018, I worked in the Murray lab to finish developing and release PsychRNN, an accessible and extensible Python package for training RNNs on a variety of cognitive tasks. The deep learning details are handled by the TensorFlow-based backend and are extensible for projects that require additional customization. We provide multiple initializations, loss functions and regularizations as well as a framework to easily add more. PsychRNN focuses on facilitating specification of neurobiological constraints on synaptic architecture and on learning.

Proposed Work

PsychRNN has not yet been thoroughly tested. I have started writing tests in the currently open pull request on the repository, and plan to finish writing tests for rnn.py at the beginning of Spring 2019 to validate the package.

---

1 Code is available at https://github.com/dbehrlich/PsychRNN, and as a prerelease on PyPi https://pypi.org/project/PsychRNN/.
I will implement closed-loop training, training that is adjusted in closed loop based on performance, in PsychRNN to allow researchers to implement task shaping. Experimentalists regularly train animals to perform tasks by shaping tasks -- PsychRNN will then allow investigation of how shaping trajectory choice could affect the observed results.

Additionally, I will draft a methods paper on PsychRNN for submission to a journal so that other researchers find out about PsychRNN, and so that they have a paper to cite when they use it.

**Characterizing the strategies and limitations of fixed- and fast-weight neural networks on a monkey memory task.**

*Motivation*

Hoshi and Tanji (2000) present a task in which monkeys use a specified arm (right or left) to select a specified target (right or left). A peripheral cue indicates right vs. left, while a central cue indicates whether the peripheral cue refers to the target or the arm selection (Hoshi & Tanji, 2000). Each trial consists of two cues, one which indicates which arm to use and one which indicates which target to select, each followed by a delay (Hoshi & Tanji, 2000). The delay requires the monkey to hold in memory information from prior cues (Hoshi & Tanji, 2000). There are multiple strategies for doing this, for example, the right / left indication from the first cue could be held in memory identically for both arm and target specificity, or the right/left indication could be stored combined with the information for arm/target specificity (Hoshi & Tanji, 2000). Hoshi and Tanji (2000) found neurons representing each of these strategies, as well as others, in monkeys.

However, we don’t understand the mechanisms by which these neurons learn to store the information using the various strategies discussed by Hoshi & Tanji (2000). By training different neural networks on the task presented by Hoshi & Tanji (2000) and characterizing the solutions reached by the neural networks, we hope to better understand what mechanisms could lead to strategies observed in monkeys by Hoshi & Tanji (2000).

Fast-Weights networks, RNNs with variables that change at a rate between that of activities and that of standard weights, are able to store temporary memories of the recent past (Ba, Hinton, Mnih, Leibo, & Ionescu, 2016). With the same training data on the same tasks, RNNs are only able to learn one task at a time, whereas Fast-Weights networks are able to learn multiple tasks (Chen, Lu, Beukers, Baldassano, & Norman, 2018). So, there is reason to expect Fast-Weights networks to produce qualitatively different results / strategies than traditional (fixed-weight) RNNs. Thus, comparing Fast-Weights and Fixed-Weights RNNs should provide some insight into how different network structures result in different strategies, hopefully providing some insight into what potential mechanisms result in the strategies found by Hoshi and Tanji (2000) in monkeys.
**Proposed Work**

I will first implement a Fast-Weights network for PsychRNN since PsychRNN does not yet include a Fast-Weights network implementation. Then I will implement the task introduced by Hoshi and Tanji (2000) in the PsychRNN framework. This will allow me to train a traditional RNN and a Fast-Weights network on the task. Then, I will compare the two networks’ solutions, and reverse engineer the networks’ solutions to some degree to better understand the solutions. Time permitting, I will additionally design and implement an expansion of the task that Fast-Weights succeeds on but that fixed-weight RNNs do not succeed on.

**List of Deliverables**

- **PsychRNN**
  - Finish implementing tests for rnn.py
  - Implement closed loop training
  - Release package update on pypi with the above developments
  - Draft methods paper for submission to a journal on PsychRNN

- Characterizing the strategies and limitations of fixed- and fast-weight neural networks on a monkey memory task.
  - Implement a fast-weights network in the PsychRNN framework
  - Implement Hoshi and Tanji (2000) task with both fast and fixed weights using PsychRNN
    - Compare the two networks
    - (Stretch Goal) Reverse engineer the different networks’ strategies
  - (Stretch Goal)
    - Design an expansion of the task where fixed-weights fails but fast-weights succeeds in order to better understand the failure modes of the different networks.

- Lab Meeting Presentation on the semester’s work in the Murray Lab.

**Works Cited**

