Exploration of Recurrent Neural Network Dynamics in Simple Working Memory Models

Christian Wawrzonek

Advised by Timothy Buschman

Collaboration: Pavlos Kollias, Matthew Panichello

This paper represents my own work in accordance with University Regulations.

Department of Computer Science

Princeton University

April 29, 2016

Abstract

How do populations of neurons encode information over very short time scales? Given extensive training, neurons are able to change the weighted connections between them in order to encode information [CITE]. However, very short timescales of only a few seconds are far too short to change neural weights [CITE]. Still, humans and higher functioning animals possess the ability to encode and maintain small amounts of information presented over very short timescales [CITE]. This is the problem of working memory, the transient holding, processing, and manipulation of information used in higher cognitive functioning. Previous computational models of working memory typically have a solution in mind and attempt to construct an architecture and training pattern that pushes a model towards the intended solution [CITE]. We attempted to train a relatively simple, unconstrained neural network on a complex working memory tasks and analyze the natural solution space found by the network "naturally." Through a range of analyses, it is clear that even a simple, single layer recurrent network is capable of dynamic, generalized solutions without deliberate solution paths presented.

1 Introduction and Background

1.1 Motivation and Goal

The motivation and style of this project follows very closely to the work of Mante et al. in their exploration of context dependent information integration in prefrontal cortex. In their words, "most often in computational studies in neuroscience a network model is designed by hand to implement a specific function, such as a decision [CITE], or auto completion of memories [CITE]. This study was different [CITE]." Like Mante et al. we trained various networks to solve the same working memory using an unconstrained, machine learning approach.

1.2 Working Memory

Todo.

1.3 Delayed Saccade Task

Todo.

2 Model

2.1 Dynamics

Todo.

2.2 Architecture

Todo.

2.3 Training Structure

Todo.

2.4 Hessian Free Optimization

Todo.

3 Methods

3.1 Linear Regressions

Todo.

3.2 Principle Component Analysis

Todo.

3.3 t-Distributed Stochastic Neighbor Embedding

Todo. (t-SNE)

3.4 Representational Similarity Analysis

Todo.

3.5 Fixed and Slow Point Analysis

Todo.

4 Results

4.1 Generalization of Model

Theta error plots. Projection Through tsne space.

The previous literature follows a strong consensus that excessively large hidden layers lead to overfitting and poor generalizability [CITE], a phenomenon readily observed in our network as well. Given that we have observed relatively consistent generalizability with a high number of locations and a small hidden layer, numerous questions arise immediately that merit exploration. What exactly is it about this "sweet spot" in model size to data ratio that leads to a more appropriately generalized model? What features can we observe in the generalized networks that are not present in the failed networks?

4.2 Dimensionality of Solution Space

Todo.

4.3 Dynamics of Solution

Todo.

4.4 Combined Network Solutions

Todo.

5 Discussion

5.1 Conclusions

While the analyses speak for themselves, there are a few closing comments. Given the simplistic architecture of our network and the unconstrained nature of our training, there was significant uncertainty whether the model would be able to find a meaningful solution, or really a solution at all. Not only was a simple recurrent neural layer able to find a meaningful solution, many aspects of the solution, such as stability, are usually seen in deliberately modeled architectures that predispose stable solutions (such as bump attractor networks) [CITE]. This exploration shows that even complex, overlapping stable representations can emerge even in simple, single layer networks. Some aspects of the model which which are less interesting are likely a consequence of the simplicity. For example, in the combined network, the two unique solutions thing?

5.2 Behavioral Similarity

As stated previously, fitting behavioral data was not an explicit goal of this model. However, similarity to behavioral data can be observed in a few cases, though the significance of the model dynamics to cortical network solutions remains to be seen. [MENTION: PC projections of attention task to FEF firing rates]

5.3 Future Work

There are an endless number of directions future explorations could take, and quite a few more that I thought I could include, but ultimately were left out. The most obvious early inclusions are variability in stimuli presentation and noise.

By variation in stimulation time I mean variation in the delay periods, stimuli pre-

sentation, and trial lengths. Most behavior tasks measuring working memory involve uncertainty in timing while this model uses constant time steps. This likely is the reason for the observed oscillatory nature of the selection parameter in the selection task. Moving the selection parameter forward or backward one time step in some trials resulted in a saccade close to the non-selected location. Given fixed delay periods, the network was able to time oscillations between two location representations to match the selection presentation. This rather inflexible solution would fail given uncertainty in the delay period. However, it is interesting to note that studies of attention note an oscillation between attended location items, resulting in oscillating response times [CITE].

In the case of noise, most network models include uncertainty terms in the activation update equations (Eq ?.) [CITE] or in the input data. Our initial goal was to explore the most simple solutions in unconstrained learning, and thus adding additional dimensions to learning such as uncertainty in the input seemed extraneous. Given that most models attempt to directly model behavioral data, noise is essential given the highly noise in neural data. [CITE] Initially, I assumed noise would be necessary to push the network towards a general solution. That is, in the absence of consistent combinations of inputs and outputs, the network would be forced to learn the underlying patterns in the data instead of simple input-output mappings. This perhaps could have aided during learning with over-fitted networks. However, generalization occurred rather easily by changing the size of the input layer, which alleviated the need to explore noise.

Given that fitting behavioral data was not a primary motivation of this study, there are numerous features that were left out that in future studies, could lead to more relevant insights into possible solutions implemented in the brain. For my second Junior Paper last year, I implemented a form of local connectivity in a similar, though much more simple, model. Constrained by physical limitations, cortical networks necessarily follow local connectivity rules. It isn't a stretch to assume that such connectivity heuristics change solutions

implemented by cortical networks. These heuristics can be modeled with simple distributions on the existence of connection weights over distance between neurons. Previous studies have found that neural connectivity tends to follow a lognormal distribution over distance in the cortex [CITE].

There are numerous hyper-parameters to both the dynamics and training of the network that could be altered, such as the choice of saturation function $h(x)$.

6 References

References

7 Appendix

7.1 Training Strategies

Describe in detail the way training sets were divided.