Exploring Recurrent Neural Network Dynamics in Simple Working Memory Models

...

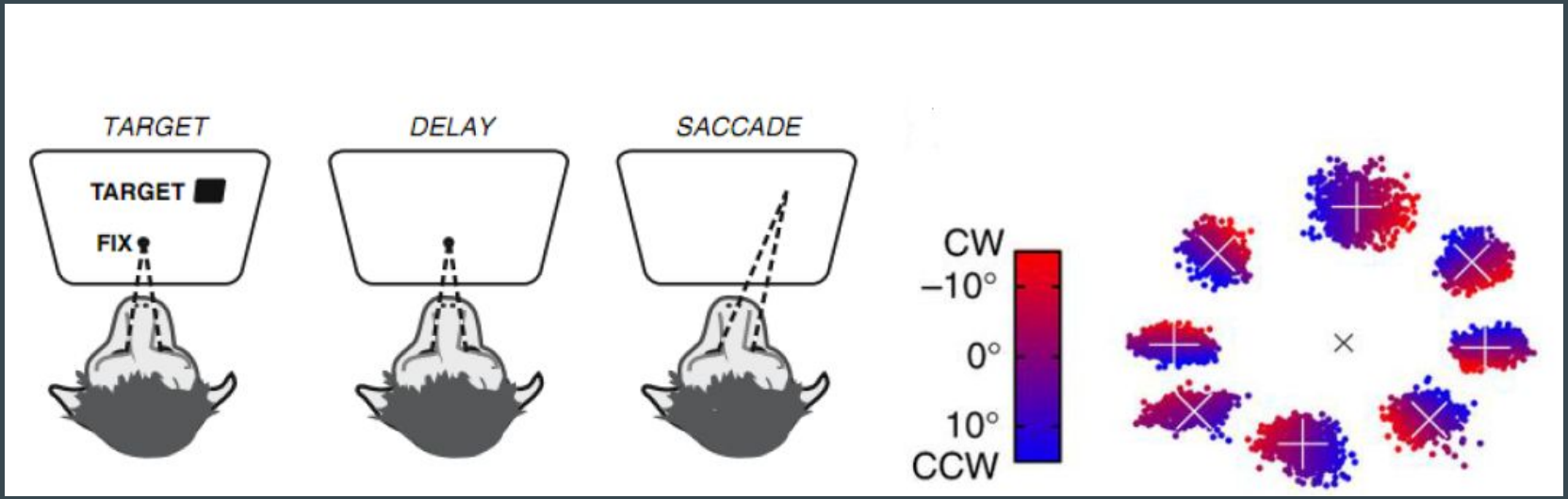
Christian Wawrzonek
AB – 2016
Advisor: Timmothy Buschman
Collaborators: Matthew Panichello, Pavlos Kollias

Understanding Working Memory (Motive and Goal)

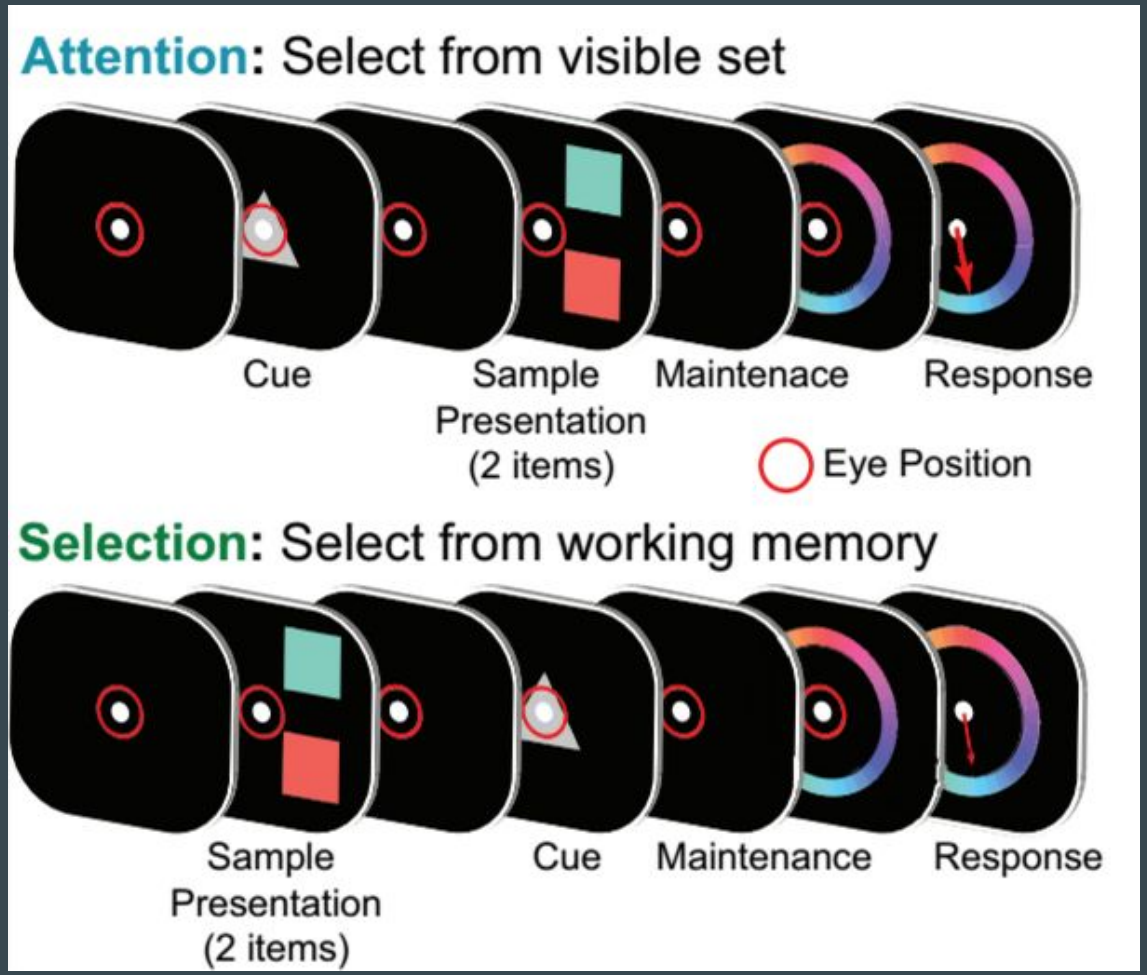
- System responsible for short term encoding and retention of small amount of information
- Extremely robust, but very limited capacity
- Conversely, working memory representations are highly flexible but very short lived
- How exactly does the brain solve this problem? Why is working memory so limited? How does the brain reconcile competing representations in working memory?
- Questions about a functioning model include: What is the dimensionality of the representation? Is the representation conjunctive or featural? What is the difference between generalized and non-generalized networks?

Delayed Saccade Task

- Classic experiment of working memory in monkeys
- Lots of experimental data, theories, and proposed models of working memory involving this task

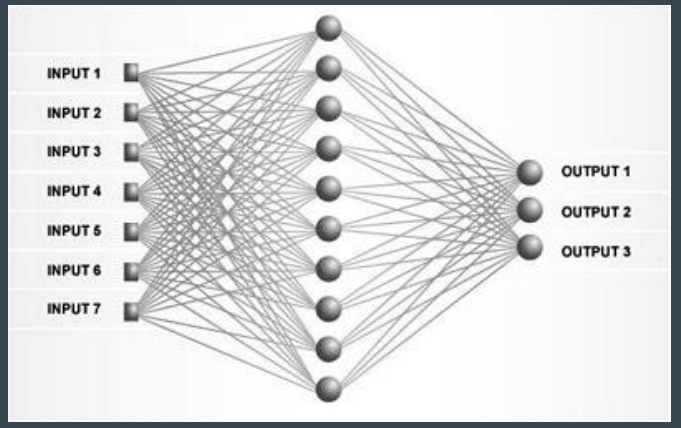


Delayed Saccade Continued



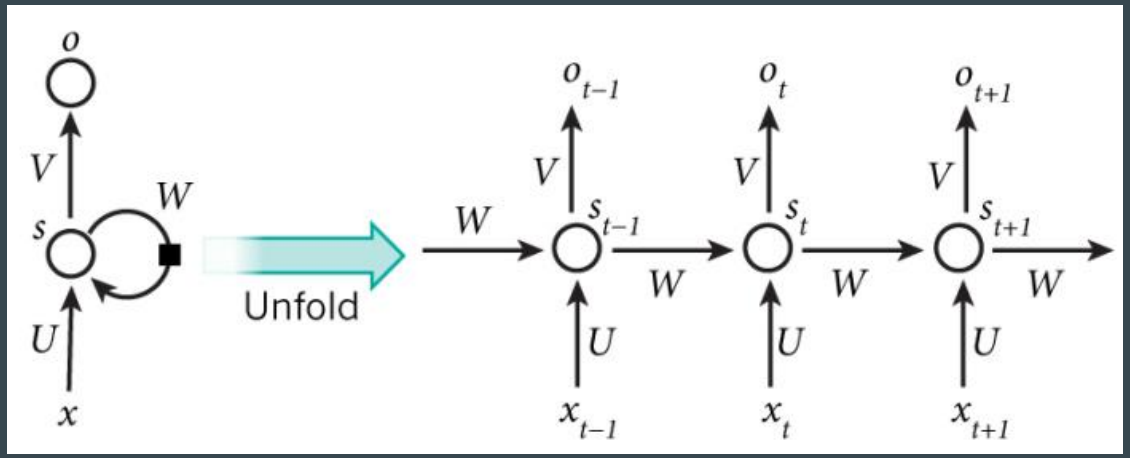
Neural Networks

- A neural network is a computational system comprised of interconnected processing elements (nodes) which use state activation to process input and produce output.
- From an artificial intelligence and machine learning POV, there is a huge scope of applications (host of decision making problems)
- Provides an abstraction of neural circuitry



Recurrent Neural Networks, Deep Learning, and Hessian Free Optimization

- Recurrent neural networks create many problems training complex tasks
- Backpropagation and Vanishing Gradient
- Hessian Free Learning
- Model created using Daniel Rasmussen's Hessian Free Learning Python module

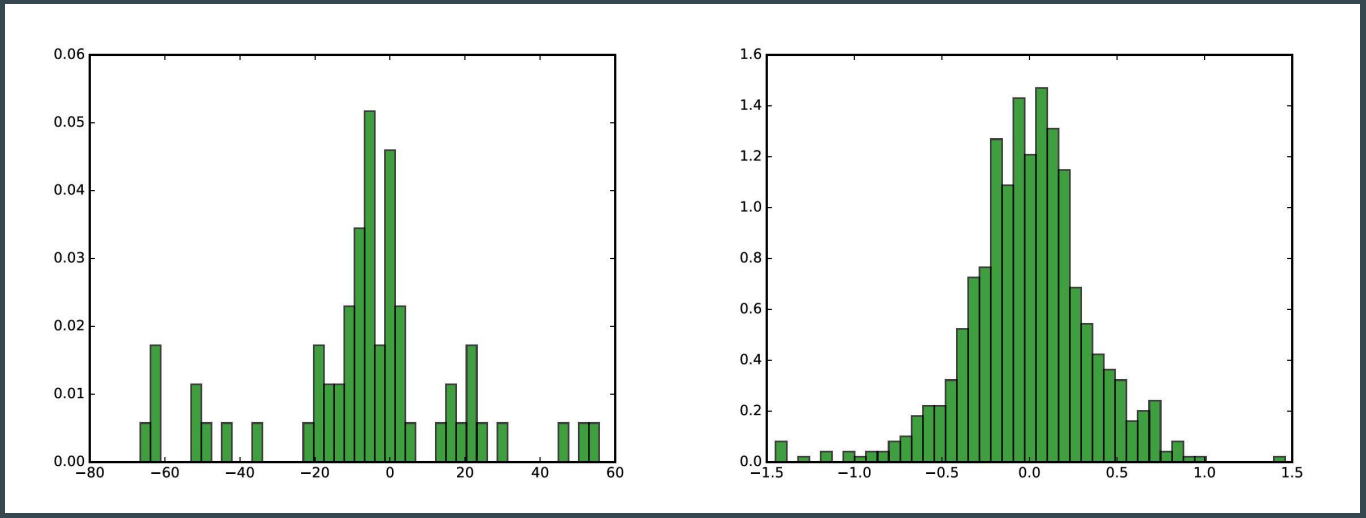


Approach and Previous Work

- Train a neural network capable of simulating a working memory task
  - Many models of working memory have been created, but none have done so in a general case. That is, without forcing arbitrary optimizations and structural decisions
- I attempt as little intentional manipulation of training as possible, aka “hacking”
- Understand the “strategy” used to solve the problem - analyze the connectivity of the model, the dimensionality, activation patterns, using tools such as t-SNE, PCA, Linear Regressions, RSA, and more.

Generalization of Model

- As with any highly unconstrained model, generalization and overfitting were early hurdles.
- Confirming previous literature, excessively large hidden layers lead to overfitting.
- By reducing the hidden layer and increasing the size of the input space, we were able to find a balanced, generalized network.

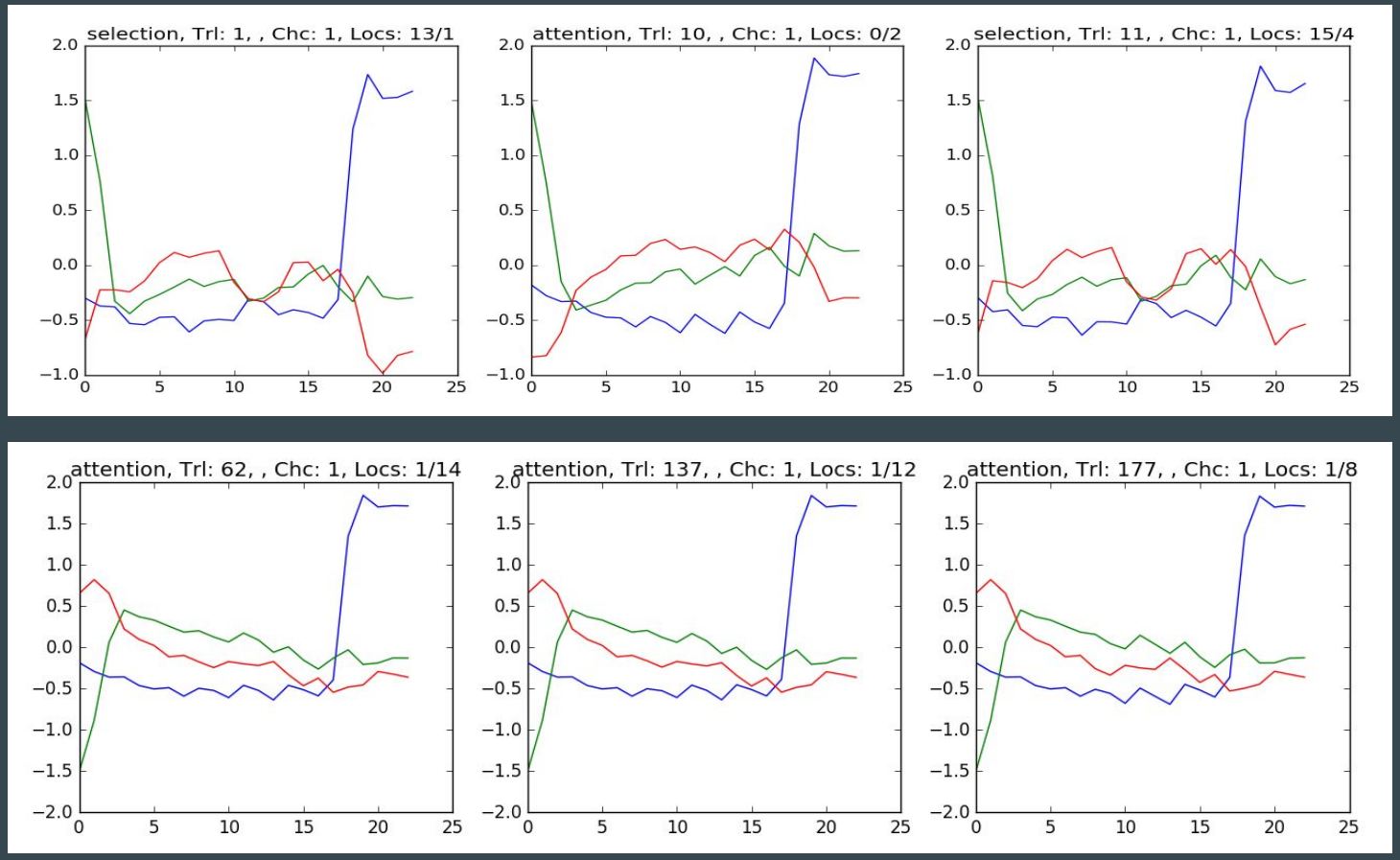


Linear Regressions

- Linear regressions with feature vectors as inputs and activation states as outputs can be used to infer relative importance of given features in underlying activation structure.
- Magnitude of coefficients correspond to relative importance in the representation.
- Tried various forms of regression (OLS, Lasso, Ridge, Elastic Net). Regularization and mitigation of highly multicollinearity of regressors was essential. LassoCV was ideal.
- Found that, in general, only locations and choice parameter significantly predicted activation states. Conjunctive components were insignificant.

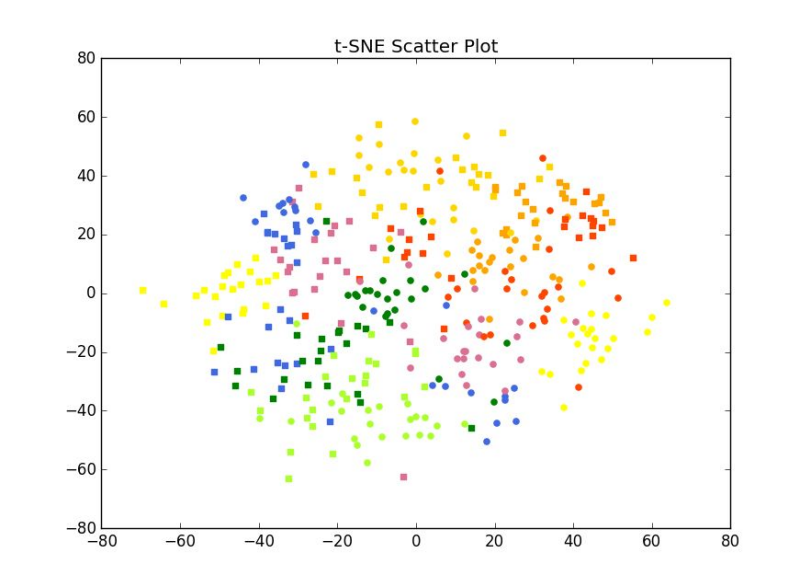
Principal Component Analysis

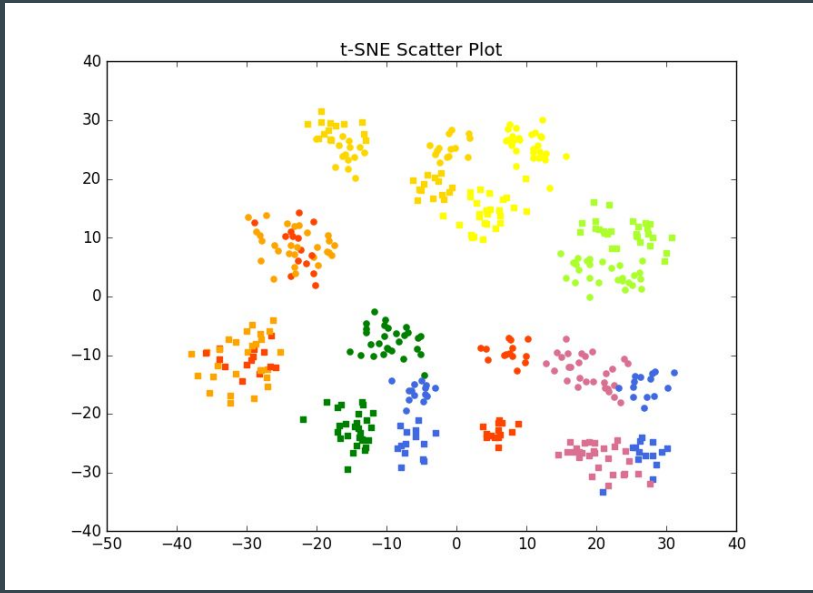
- Most common way of reducing dimensionality of network activations.



t-Distributed Stochastic Neighbor Embedding (t-SNE)

- Nonlinear dimensionality reduction, similar to PCA
- Chooses dimensions which create most separation between dissimilar samples





Representational Similarity Analysis

- Uses a second-order isomorphism approach by relating the similarity structure of objects (i.e. inputs or representations) to the similarity structure of themselves (i.e. RSM/RDM)
- Advantage over first-order correlation techniques in that it finds correlations between similarities rather than absolute representations (matches patterns of representation)
- Can create a pseudo-random distribution of correlation coefficients by shuffling trial by trial feature set, allowing for significance tests on any set of features tested
- For small hidden layers and generalized networks, features were almost always spread out across all hidden units. Solutions utilize population encoding, never individual units

Conclusions and Questions

- Complex working memory tasks are clearly solvable in a shared representation space using simple, single layer recurrent neural networks.
- Generalized solutions necessarily represent the underlying structure of the task as separate features within the population.
- Single units are rarely selective for only a subset of task features. Rather, solutions are formed from population encodings rather than individual unit encodings (grandmother cells are a myth!)