A Recurrent Neural Net for Neurons:
Continuous Decoding of Intracortical Brain Signals for BMI Applications

Jonathan Zwiebel CS ‘21, Robert Ross CS ‘21, Samuel Lurye EE ‘20
[jzwiebel, rross, slurye]@stanford.edu

Stanford | ENGINEERING

Motivation

Brain Machine Interfaces (BMIs) are becoming feasible **clinical treatments for paralyzed patients**. Using just thought, patients are able to move robotic arms and computer cursors with increasing **precision and dexterity**. Basic neuroscience shows that neuronal dynamics are **not effectively** modeled with **linear models**. This project builds on existing linear methods to continuously **decode 192-dimensional binary neural signals into 2-dimensional X-Y positional coordinates**. The goal of this project is to create a model that can give a **live prediction of position** given a **live stream of neural signals**.

Data

- The data consisted of 11,136 trials [1] of a monkey performing radial reach tasks to one of 48 targets (Figure 1).
- Each trial consists of 192x800 matrix representing the presence or absence of a spike for any of the 192 neurons for an 800ms sample duration. (Figure 2)
- Additionally, each trial contained a 2x800 matrix containing the true X-Y position of the monkey’s hand.
- We extracted the hand velocities and target vectors for each ms (Figure 3). Furthermore, we binned this data into 25ms bins to reduce variability.

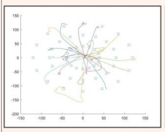


Figure 1- 48 possible targets with 20 randomly selected reaches overlayed

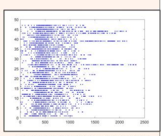


Figure 2- Example of spike data for 50 electrodes during one trial

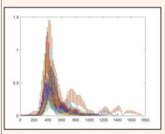


Figure 3- Plot of hand velocity magnitude for 25 trials

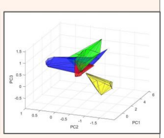


Figure 4- PCA for up, down left and right reaches.

Model

Our Model
Given the sequential structure of the data and the necessity to provide live predictions, we decided to use a forward-facing many-to-many RNN. We initially planned on testing basic RNNs, GRUs, and LSTMs, but opted exclusively for LSTMs after running into exploding gradient issues. Each cell took in a 192 element vector as an input and had its output fed into a series of fully connected ReLU layers which reduced its size to a 2 element vector. Note that the final FC layer does not include any activation function, as we are attempting to predict real number values. We experimented with both RMSE Loss and Huber Loss after determining that the distribution of our loss across training examples contained many outliers. The sequence length of our model was a crucial hyperparameter and we sampled values ranging from 50 to 800.

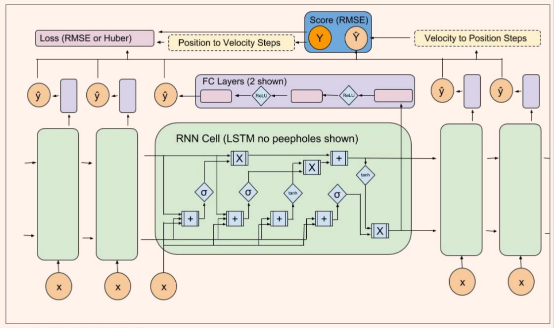


Figure 5 - Block diagram showing model, inputs, outputs, and cost

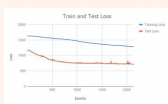
Relative Position vs Velocity
One of the major distinctions between our models was the choice of position or velocity as our target prediction. In models with position as a target we compared our predictions directly to the raw truth values for both our loss and score function. In models with velocity as a target we calculated velocity from our raw dataset and compared it to predicted velocity for our loss function. To score velocity models on the test set we calculated a stepped predicted position based on our predicted velocity and compared it to the raw dataset values for position.

Binning
For one of our models we calculated velocity values over time intervals of 25 measurements, using binned sums of neuron fire counts over an interval.

Clipping
To combat exploding gradients we implemented gradient clipping on all weights, which we found drastically decreased initial training loss.

Results

Loss Calculations



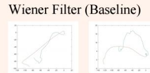
Final Model Scores (RMSE)

No Prediction (0,0)	465.76
Wiener Filter	32.316
Kalman Filter	4.281
Best Position RNN	**17.535**
Best Velocity RNN	118.56
Best Velocity Bin RNN	3437.8

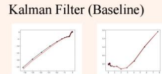
On the Best Position RNN, we trained the model with slightly over 2,000 epochs. Utilizing a Mean Squared Error(MSE) loss to penalize the predictions against the actual positions, resulted in a generally decreasing loss that correlated with increase in position predictions.

The Best Position RNN was able to to surpass the Wiener Filter on the subjective appearance of the position values as well as through the objective scoring function of Root Mean Squared Error(RMSE). However, the models with velocities - especially velocity with binning - performed poorly due to overfitting caused by the sequence lengths becoming too small. Furthermore, the secondary baseline of the Kalman Filter still outperformed the best model that was constructed.

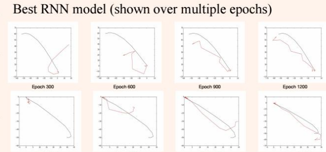
Wiener Filter (Baseline)



Kalman Filter (Baseline)



Best RNN model (shown over multiple epochs)



Discussion

- **Fewer hidden units** in the LSTM cells result in **comparable** performance at significantly **shorter** training periods.
- **100ms initial** skip time with a **500-cell RNN** maximized performance. This is consistent with that the data distribution shown in Figure 2.
- **Velocity-trained RNNs** preformed substantially **worse** because X-Y features were abstracted away.
- Multiple fully-connected layers in between LSTM cells improved performance.
- Sequence lengths **below 200** ms led to **severe overfitting**, even with dropout.
- Binned velocity models always overfit for our data as they resulted in sequence lengths of 25 - 30.
- Our model was able to **outperform** the naive **Wiener filter** but was not able to match the performance of the Kalman filter.

Future Work

- Gather **significantly more data**. The amount of trials we had is small for modern deep learning algorithms.
- Update loss function. More sophisticated loss functions should **penalize smoothness and directionality**.
- Develop test sequences of significant length (10 - 20s) and test extended model performance over them.
- Feed previous position predictions as inputs to future states.
- Train with **lower learning rate** and higher hidden layer size on more powerful computers.

References

1. All data was graciously provided by Saurabh Vyas collected from Professor Krishna Shenoy’s Neural Prosthetics Systems Laboratory at Stanford.
2. Kao, J., Stavisky, S., Sussillo, D., Nuyujukian, P., & Shenoy, K. (2014). Information Systems Opportunities in Brain–Machine Interface Decoders. *IEEE.*
3. Olah, C. (2015, August 27). Understanding LSTM Networks. Retrieved from http://colah.github.io/posts/2015-08-Understanding-LSTMs/