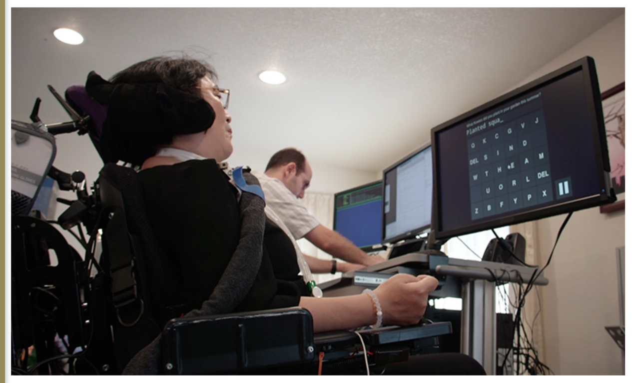
Fast Direction Decoder for Brain Computer Interfaces

Sean Metzger and Jinhie Skarda

Project Goal and Motivation

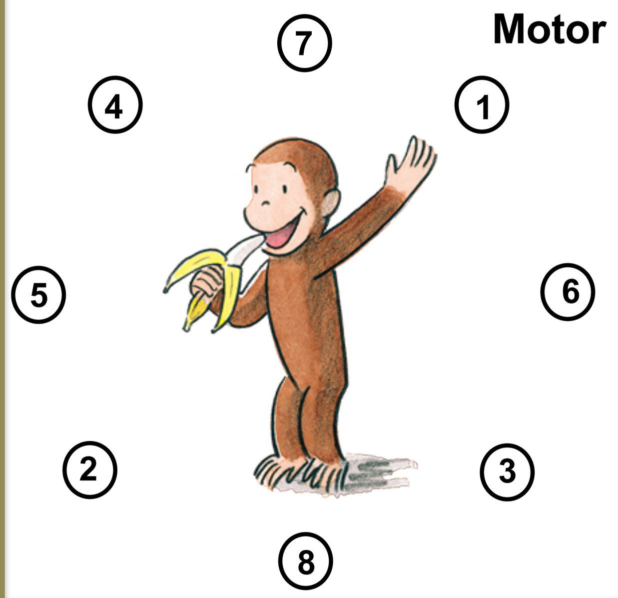


1 in 50 Americans is paralyzed. Brain computer interfaces use neural signals to control devices like prosthetics or cursors on computer screens, thus offering enhanced mobility and quality of life for this significant proportion of the population that is unable to move.

The goal of this project is to create a fast classification algorithm for brain computer interfaces[1] that quickly predicts the direction someone is reaching based on real-time electrode data from their primary motor cortex using a machine learning approach. In the context of a brain computer interface that uses neural signals to control a computer screen cursor, we would want our algorithm to quickly predict the direction the user wants to move the cursor and move the cursor to that point instantly for faster, easier brain computer interface use. This technology can be translated to help paralyzed people since when they think about reaching somewhere, even if spinal cord lesions block signals from reaching their arm, the motor cortex signals are the same as if they were actually reaching.

Dataset and Methods

Motor Cortex Electrode Data

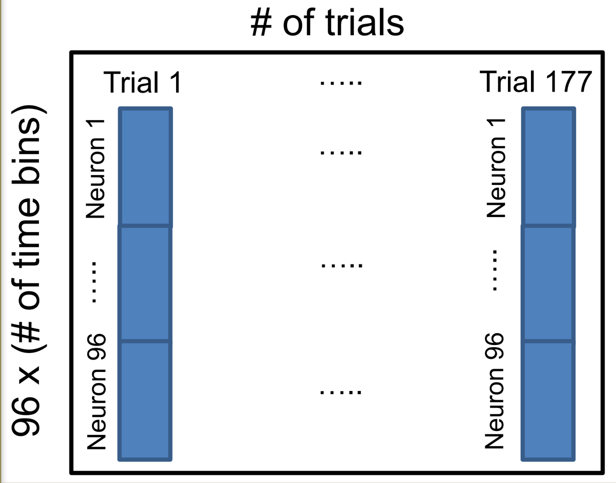


- Real-time electrode data from the primary motor cortex of a monkey from BrainGate.
- Monkey performs the task of reaching in one of eight possible directions
- 253 reaching trials
  - 96 electrode probes measuring the signal from one neuron each
  - Data sampled at 1 ms resolution

Representation of Data and Application of Softmax

- Binned the time-history data for each probe by summing the data in each bin width so the data is counts of how many spikes a given probe registered
- Time series lengths are not the same across trials so chopped the end of longer trials to make them all the same length
- Considered all of the data from the 96 probes from each trial as a separate feature and stacked the binned data from each probe to create a long vector of length 96*(# of bins) for each trial.
- Used 1-hot representation for the true labels for each of the trials
- Used 70% of the trials as the training set and 30% as the test set

Training set

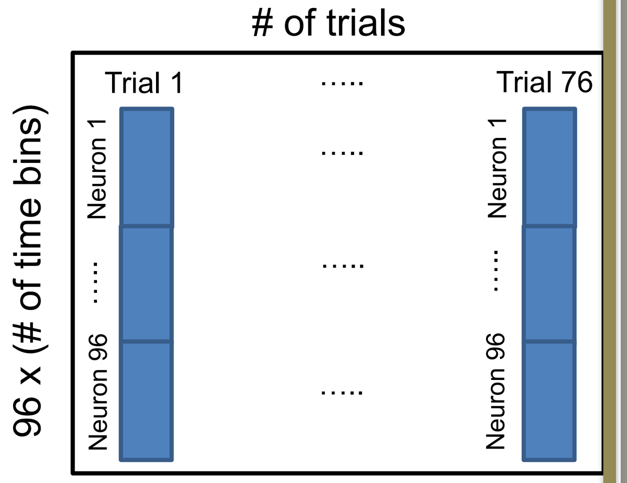


Softmax Regression

$$p(y=i|x;\theta) = \frac{e^{\theta_i^T x}}{\sum_{j=1}^{k} e^{\theta_j^T x}}$$

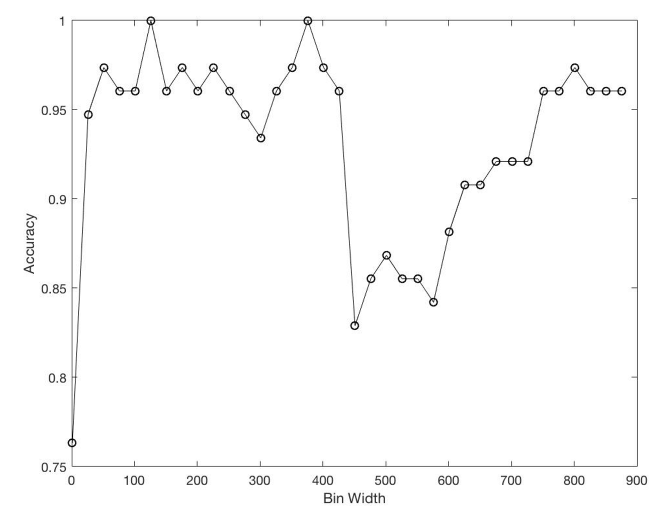
Implemented softmax using a 1-layer Neural Network in Matlab's Deep Learning Toolbox

Test set



Softmax Classifier Results

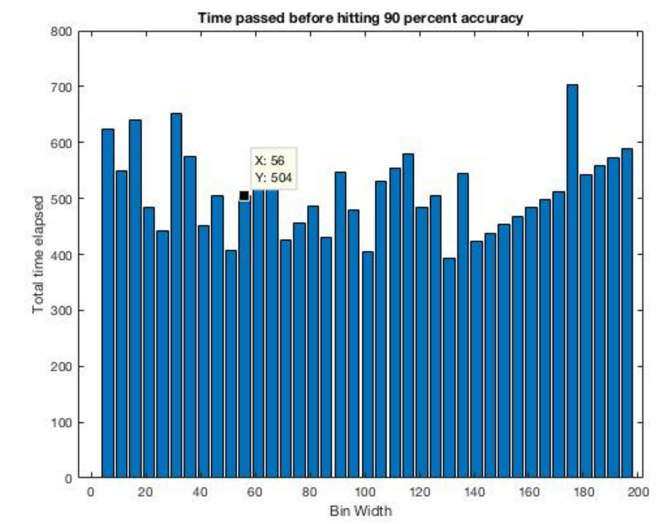
How does bin width affect accuracy of the classifier?



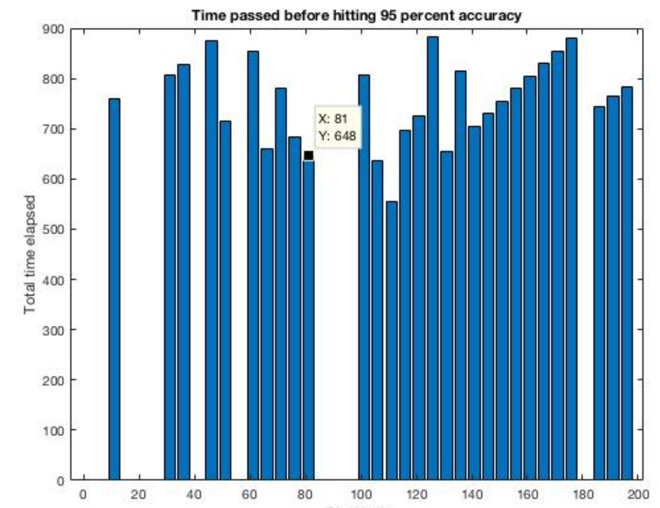
Accuracy vs. Binwidth for our Softmax Decoder.

- Very good accuracies for most bin widths – some bin widths like 125 ms give 100% test set accuracy
- Accuracy drops after 400ms because then we ignore half the data since there isn't enough to fill a second bin

How fast can we guess reach direction?



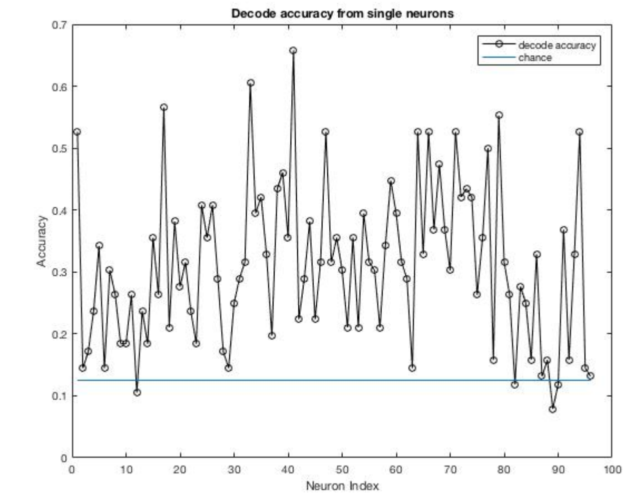
First time we can guess reach direction with 90% accuracy vs. binwidth



First time we can guess reach direction with 95% accuracy vs. binwidth

- For 90% accuracy, only takes 500-600ms
  - 400ms before the end of most trials
  - Hack for better accuracy – don't guess when model outputs 1 and 6.
  - 95% accuracy takes longer ~700ms

Decoding with a Single Neuron

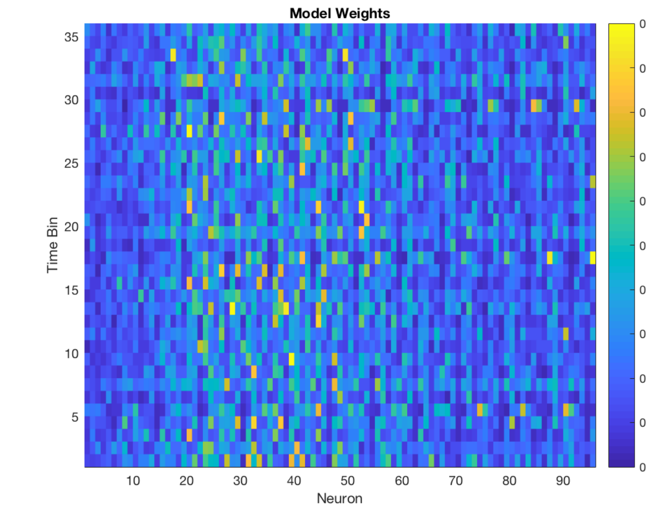


Decode Accuracy from a single neuron. Most do better than chance.

- One neuron (41), perfectly predicted reach direction 8
  - No false positives or negatives, 100% true negative rate.

Neuron #	17	21	33	38	41	73
Reach Direction	8	4	5	1	8	6
True Positive Rate	0.9	1	1	1	1	1
True Negative Rate	0.984	0.905	0.943	0.904	1	0.929

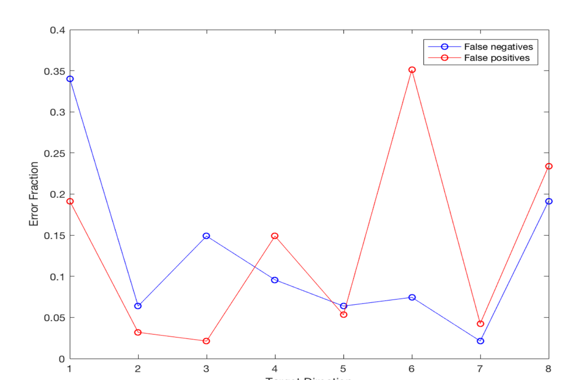
Classifier Weights



- Weights vary more with the neuron we're decoding on than the time at which we're decoding it.
- Suggests that for future work, Increasing number of neurons is important

Analyzing Model Performance

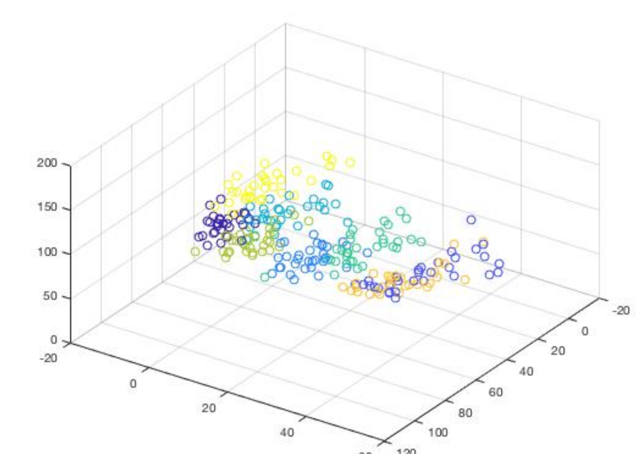
Misclassified Directions



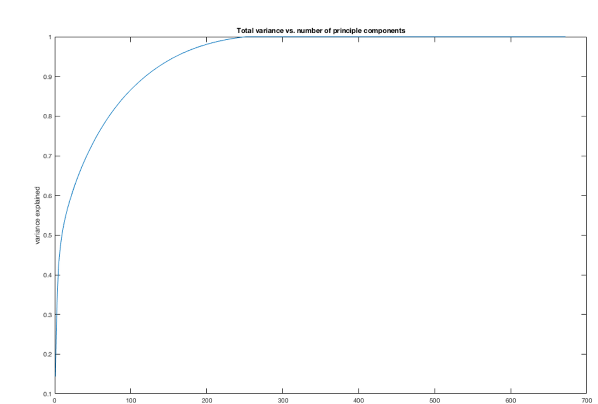
Fraction of errors by reach direction over all binwidths

- Directions 1 and 6 are the most frequently misclassified directions
  - This could make sense because they are less involved than reaches across the body

Understanding Our Results Through PCA



Projection of data onto first three principle components. Legend: Purple = reach direction 1 , blue = 2, light blue = 3, aqua = 4, green = 5, brown-green = 6, orange = 7, yellow = 8



Most variance is contained within 200 PC's. Not surprising given so much of our data is correlated [e.g. neuron_1(t1) -> neuron_1(t2)].

- Sought to understand great performance using PCA.
- Differs from previous work[2] by ignoring trajectories and focusing solely on discerning the final target.
- Directions differentiable by projecting time & activity vectors on first three components.

Future Work

- Develop a potentially faster brain computer interface keyboard by extending the 8 reach directions studied here to 26. The keyboard would then be represented as a circle with letters distributed evenly around it. Our algorithm would quickly determine which letter the user intended to type by using the time-history electrode signal to predict the direction they were picturing reaching and snap to it quickly, potentially improving speed over current methods.[3]
- Extend analysis of which neurons contribute most to decoding which directions in order to create a detailed map of which parts of the motor cortex are responsible for which reach directions.
- Use a larger set of electrodes in order to study more sets of neurons – can we find more 'perfect neurons' like #41?
- Further study neural reasons why our softmax classifier has a harder time classifying reaches on the same side of the body than reaches across the body
- Use cursor position as a feature

References

[1] Gilja V, Nuyujukian P, Chestek CA, Cunningham JP, Yu BM, Fan JM, Churchland MM, Kaufman MT, Kao JC, Ryu SI, Shenoy KV (2012) A high-performance neural prosthesis enabled by control algorithm design. *Nature Neuroscience.* 15:1752-1757.
[2] Churchland MM, Cunningham JP, Kaufman MT, Foster JD, Nuyujukian P, Ryu Si, Shenoy KV (2012) Neural population dynamics during reaching. *Nature.* 487:51-56.
[3] Nuyujukian P, Fan JM, Kao JC, Ryu SI, Shenoy KV (2015)A high performance keyboard neural prosthesis enabled by task optimization. *IEEE Transactions on Biomedical Engineering*. 62:21-29.

seanmetz@stanford.edu and jskarda@stanford.edu