Predicting Primate Upper Limb Movements Via Neuron Firing Rates is No Monkey Business

Cindy Li, Jason Manuel, Put Dam

Introduction:

Spinal cord injury damages the neural connection between the brain and muscles but otherwise leaves the motor cortex and muscles intact. One potential treatment to paralysis is thus thought to be routing the connections from the brain to the limb artificially. It has already been shown that monkeys can control a robotic arm using cortical activity (Moritz, Perlmutter, Fetz. 2008).

Moving an upper limb may engage many cortical areas, among which the motor cortex plays an important role. The neurons in the motor cortex may then hold enough information to reconstruct upper limb positions and thus limb movement. Previous work with decoding this neuronal information has shown the potential of decoding this neuronal information to use in the reconstruction of natural upper limb movement. With the use of a linear state model, which showed a 20% improvement over least-squares linear filters, Vargas-Irwin et al. were able to conclude that a small subset of primary cortex neurons is sufficient for reconstruction of upper-limb positions (Vargas-Irwin, et al. 2010).

Our goal was to see if deep learning methods could be used to decode the neuron information and translate them to limb movement and whether that would offer any improvement over previous models.

Methodology:

The data we used was already collected and given to us by Dr. Vargas-Irwin, et al. They collected cortical readings for 260 neurons from 96-microelectrode arrays implanted in the arm and hand regions of the motor cortex in macaque monkeys and used 29 reflective markers to record the monkey's arm movement. Each position is represented by a single number rather than an x, y, z coordinate system. The monkey then engaged in reaching and grasping tasks, where it grabbed different items in different trials, such as cubes, cylinders, and rings.

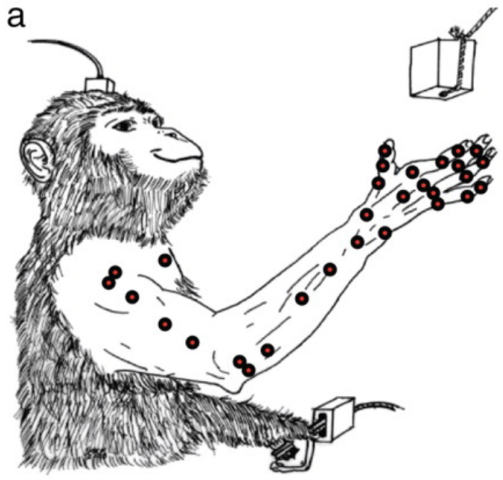

Figure 1. Example of the setup used to collect data. Monkey is tasked with grabbing something. Meanwhile, motor cortex activity and upper limb motion is recorded (Vargas-Irwin, et al. 2010).

We then processed the raw data into bins of time, where each bin represented a 10 ms window. We then iterated through the neurons to determine the spike counts for each neuron for each bin and interpolated the bin start and edge times to get the kinematic position at that time. We then grouped the bins into sequences of length 10 (~100 ms) and used these sequences as inputs to our model. The kinematic position at the end of each sequence was used as labels. After binning and grouping into sequences, we had 862 sequences across 6 different grasping tasks.

For our model, we played around with a model using an LSTM followed by dense layers and later with dropout layers as well. Model 1 used an LSTM layer followed by a flatten layer to get it down to a 2-dimensional matrix instead of a 3-dimensional matrix, followed by a dense layer with output size 1000 with a leaky relu activation followed by another dense layer with output size 29. Model 2 used an LSTM layer followed by a flatten layer followed by a dense layer of output size 2000, a dropout layer with rate 0.1, a dense layer of output size 1000, a dropout layer with rate 0.3, and a final dense layer with output size 29. The final output shape for both models was a 1x29 vector where each number represented the position for one marker. We used a batch size of 128, and a learning rate of 0.001. We shuffled the set of sequences and used 80% for the training set and 20% for the testing set. We trained over 200 epochs, and shuffled the training set between each epoch to prevent overfitting.

Given the sequential nature of the data, we wanted to use an RNN of some form. We initially decided on an LSTM due to its ability to better remember things for a longer span of time than a traditional RNN. We had also considered other models besides using an LSTM. This included other recurrent neural networks, such as a GRU model. Our implementation of a GRU model had similar runtimes to the LSTM, but noticeably poorer results. We also considered if other approaches presented in class-- such as the Transformers model and reinforcement learning-- would be viable. Although these models have merit, we ultimately settled on using an LSTM. The data gathered represents a continuous stream of neuronal activity mapped to continuous physical movement. Consequently, an LSTM seemed most appropriate for this translation from neuronal activity to movement.

Challenges

Challenges included preprocessing the data as the raw data was unbinned and the format, after importing from a matlab file, could be a little weird. For example, some of the tables were a single array of arrays, which affected the shape and thus some of the steps that had to be taken to make it work with the numpy operations we wanted to use. Additionally, in the preprocessing we had to decide what data we would use in our model, as this would affect how accurate it was. It was also a challenge to determine how we can quantify the success of our model. We did settle on using a correlation coefficient between the true and predicted positions, and comparing our results with that of the original paper.

Results:

The loss of our LSTM model was calculated by taking the mean square error of the predicted kinematic positions and the true kinematic positions. For Model 1, the loss between the predicted and actual kinematic positions starts at ~3000 and ends at ~100. During the first twenty five epochs, the loss quickly dips to about 200, but then fluctuates for another 50 epochs before it begins to level off at around 100. The remaining epochs showed little to no improvement as the point of diminishing returns seems to be reached consistently within the first 100 epochs. The

leveling off of the loss can most likely be attributed to the lack of data. Model 2 followed the same pattern but with a better final MSE of ~80.

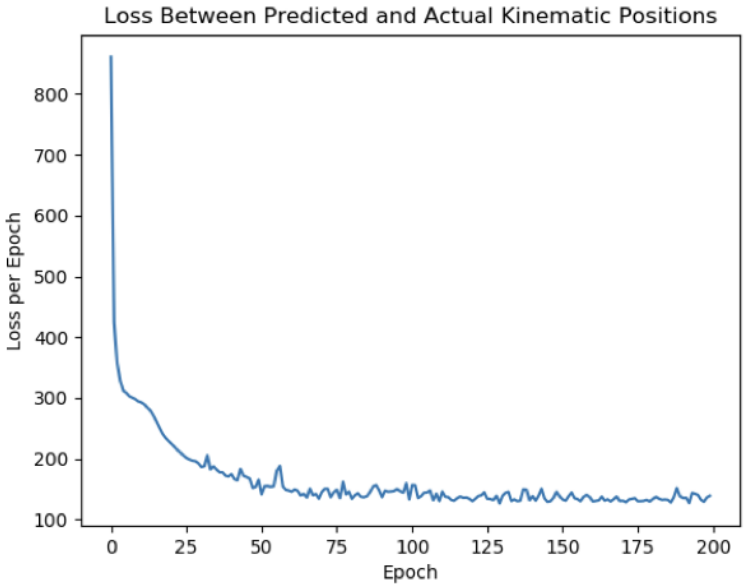

Figure 2. Graph of the loss (MSE) on the training set at each epoch for Model 1. The loss drastically declines in the first 25 epochs but quickly begins to level off after.

Our quantifiable goal for the model is to produce a correlation coefficient of at least 0.6. The results of the R coefficient graph have a close correspondence to the graph of the kinematic position loss. For both models, over the first twenty five epochs, the correlation coefficient fluctuates frequently between 0.3 and 0.4 as the loss is steeply decreasing. Then over the next 50 epochs, as the loss stabilizes, the coefficient spikes to just over 0.5 before gradually decreasing and then leveling off at around 0.45 for the remaining epochs, for our first model. Model 2 did slightly worse, with the correlation coefficient reaching its peak around 0.45. Despite the stabilization of the loss in the final 100 epochs, the R coefficient graph continues to decrease, albeit gradually.

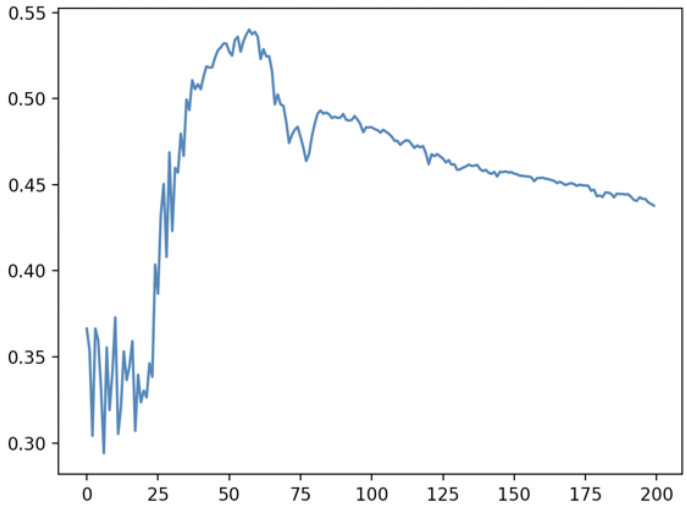

Figure 2. Graph of the correlation coefficient at the end of each epoch for Model 1. The correlation coefficient goes up initially but then begins a downward trend after about 50 epochs.

During training, we noticed that while the training loss continued to decrease, eventually reaching the single digits, the testing loss held steady at ~100. This could be an indication that the model was overfitting to the training set, which was further exacerbated by the limited data. Conversely, it could also be reflective of underlying issues in our model that we have yet to pinpoint.

As previously mentioned in the original paper by Vargas-Irwin, et al, only a small subset of neurons is required to reconstruct the upper limb positions. However, our model utilizes the entire set of 260 neurons, and so our model is also being trained with neurons which may have little to no impact on the upper limb motion. Consequently, the decreasing R coefficient in the last 100 epochs could potentially be linked to the inclusion of these extraneous neurons in our training data.

Reflection

This was a very interesting project for us to work on. It really emphasized the useful applications of deep learning in making a potential positive, impactful difference on people's lives. Rather than building a model for a homework grade, we were building a model to potentially be used in the real world. Though our model did not yield the best results, knowing that our work had the potential to help patients with paralysis still made the whole experience very cool for us.

It was also a great learning experience for us in that there really were no instructions for us to follow. We had to determine which layers to use and what the hyperparameters should be. There were no guidelines to go off of and we just had to experiment by ourselves.

We think it would be very interesting to continue with this work. As previously mentioned, our model did not yield great results, so it would be interesting to see if with some more layers or more fiddling with hyperparameters or even a different model altogether, could significantly improve the results. We would also like to experiment with taking subsets of neurons to see if that improves the performance of the model. Because of the vast number of possible permutations of neuron subsets, there is much extensive progress to be made here.

Additionally, we would like to possibly consider different loss functions and how they may impact our results. For example, in the paper, Dr. Vargas-Irwin utilized the mean absolute error to quantify the difference between predicted and true kinematic positions.

We think the work that Vargas-Irwin, et al. is doing is super important and we are grateful to have been able to contribute in any small way that we could. We would like to thank Dr. Carlos Vargas-Irwin and Maria Daigle of the Donoghue Lab for granting us access to their data and for lending their expertise throughout the course of this project. We are also grateful to have been able to learn about and how to use all these different kinds of deep learning models from Daniel Ritchie and the wonderful TAs. Thank you for the great semester!

References

Moritz CT, Permutter SI, Fetz EE. Direct control of paralysed muscles by cortical neurons. Nature. 2008;456(7222):639-42.

Vargas-Irwin CE, Shakhnarovich G, Yadollahpour P, et al. Decoding Complete Reach and Grasp Actions from Local Primary Motor Cortex Populations. J Neurosci. 2010;30(29): 9659-69.

Github: https://github.com/cli1903/monkey-business