Decoding Position from Neural Activity in the Hippocampus

Kathryn McClain

May 12, 2018

1 Introduction

Unrivaled by any other computational system, the brain has incredible abilities to process information and command a vast set of precise responses. We know the brain possesses the information required for these computations, but we still have little understanding of how the information is stored and the form that it takes within the brain. One region, called the hippocampus, is thought to contain spatial information relating to an animal's environment. Individual neurons in this region change their pattern of activity as a function of the animal's position. Together, a large population of these cells is thought to encode a spatial map that an animal may use to guide its navigation. However, we still do not know how the information of multiple neurons is assembled to create a functional map of an environment. To begin addressing this issue, I explore methods for extracting spatial information from populations of hippocampal neurons.

Decoders are used in neuroscience to predict some external variable from neural data. In this project I aimed to predict the position of a rat on a linear track based on the neural activity in the rat's hippocampus. I evaluate the accuracy of 3 different decoding algorithms: a linear decoder, a Bayesian decoder, and a neural network, and determine that a neural network performs best on this task. Identifying an effective decoder gives us access to the spatial information in the hippocampus. With this information, we can ask new questions about how the spatial navigation system works and in the future develop brain machine interfaces that could navigate with human neural activity.

2 Data

2.1 Set Up

In a standard spatial navigation task, a rat ran around a circular track (diameter=1m) to collect a reward. To receive the reward, he had to alternate between clockwise and counter-clockwise runs for each trial. He completed 207 trials, totaling 48 minutes of running. Throughout this time, the rat's position was captured by a 3D tracking system and his neural activity was recorded by electrodes that were chronically implanted in his hippocampus.

2.2 Preprocessing

The position data was recorded as vectors in a 3D space. Since the rat ran around an effectively 1D circular track, I linearized the position data by projecting it onto a circle. I then binned the data into 360 spatial bins, conveniently corresponding to 360 degrees around the circle. The circular topology of this space requires the distance metric be angular rather than euclidean, *i.e.* the distance between bin 350 and bin 5 is 15, not 335.

The neural data captured the activity of 109 hippocampal neurons. The unit of activity in neural data is a spike, wherein the voltage across a neuron's cell membrane quickly changes then returns to baseline. The frequency of spikes that a neuron elicits (termed "firing rate") encodes the signal that that neuron transmits. To compute firing rates, I binned the number of spikes occurring within each frame of the camera. The frame rate of the camera was 30fps, so I computed a firing rate for each neuron at each 1/30th of a second. This choice of bin size is convenient given

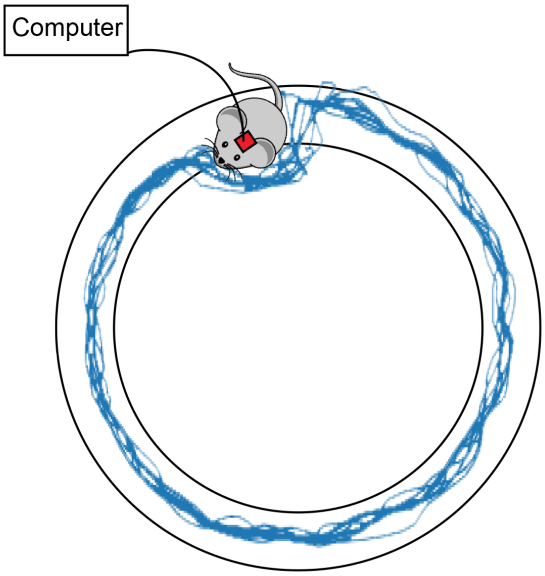

Figure 1: Set up: a rat ran around a circular maze while neural activity in his hippocampus was recorded. The blue lines represent 5 example trials showing the stereotyped behavior.

the position data, but it is arbitrary with respect to the neural data. To mitigate any possible artifacts of this choice, I performed Gaussian smoothing over time.

From here, I selected trials where the rat ran clockwise. Hippocampal neurons are known to change their activity dramatically based on the direction the animal travels, so for consistency I used only one direction. Next, I discarded neurons that did not have responses in this direction since as features in a predictive model, they provide no useful information.

The final neural data was a matrix (X) where each row is one 33ms observation and each column is the time series of one neuron's firing rate. The position data was a vector ($\mathbf{p}$) of the animal's position at 33ms point in time.

2.3 Neural Data

The "spatial tuning" of neurons in hippocampus is established by averaging their firing rate over space across trials. This produces curves like figure 2 where a each neuron has a preferred spatial location. Some neurons show this ideal guassian-like tuning, but others show more varied patterns, as in figure 4. These still contain spatial information, but do not have a standardized curve. Moreover, hippocampal neurons have notorious variability, so on any given trial their firing rate may deviate substantially from the average firing rate for that position (figure 5). This heterogeneity of tuning and variability combine to make the problem of spatial decoding relatively complex.

Different neurons also demonstrate different baseline levels of activity. Figure 6 shows the distribution of average firing rates for neurons recorded. The overall average is 7.6 spikes/second, but the neurons can be separated into two groups: fast firing, and slow firing. The cutoff between these can be seen around 5 spikes/second, where the distributions separate. The average firing rate for the fast firing neurons is 18.6 spikes/second, and for slow firing neurons it is 0.57 spikes/second. Figure 6 also shows that the firing rates are not normally distributed, making normalizing across neurons (features) difficult. For this reason I avoided normalizing features and instead only consider algorithms that are insensitive to the scale of each feature.

3 Decoding

The basic structure of any decoder is $f(x) = \hat{p}$, where x is a vector of instantaneous firing rates across the neurons, $\hat{p}_i$ is the predicted position, and f is the prediction function. The prediction function can be evaluated with some distance metric, $d(p, \hat{p})$, which in this case is the angular distance (d_θ) between the true position, p and $\hat{p}$. The median is used to evaluate across all data points, rather than a mean, or mean squared error, because it is expected that momentary neural

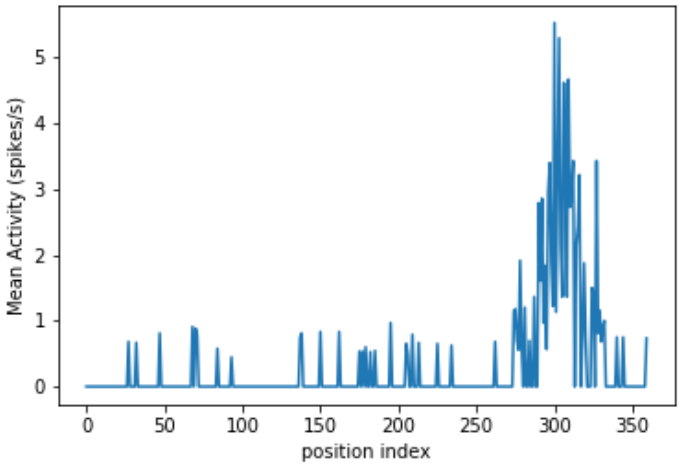

Figure 2: Average firing rate of one neuron over the length of the track showing canonical Gaussian spatial mapping

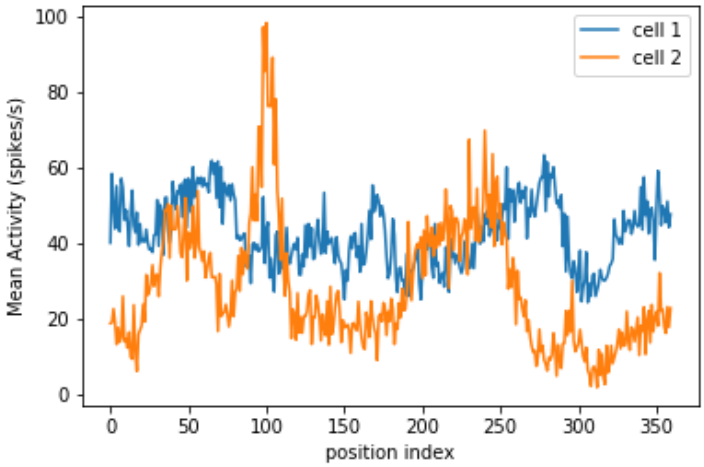

Figure 3: Average firing rate of two neurons over the length of the track showing non-Gaussian spatial mapping.

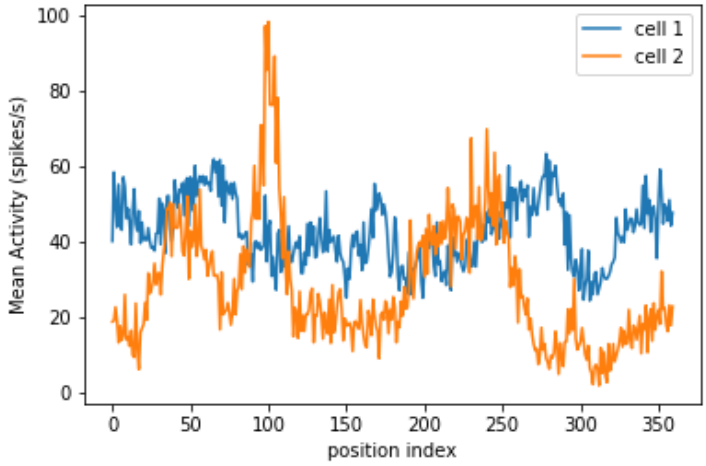

Figure 4: Average firing rate of two neurons over the length of the track showing non-Gaussian spatial mapping.

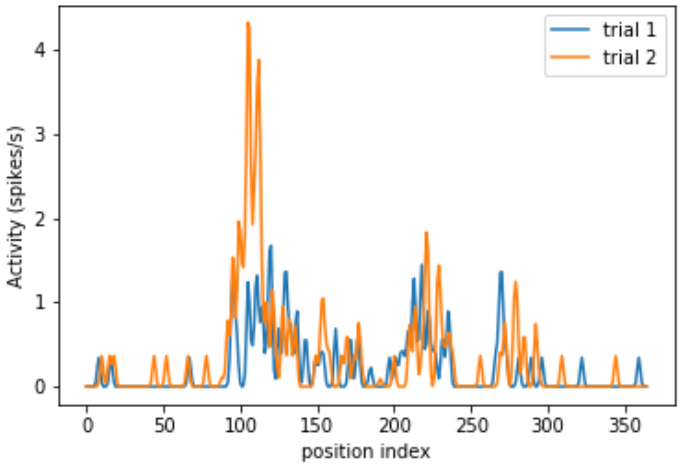

Figure 5: Firing rate of a single neuron on 2 different trials, showing trial-to-trial variability

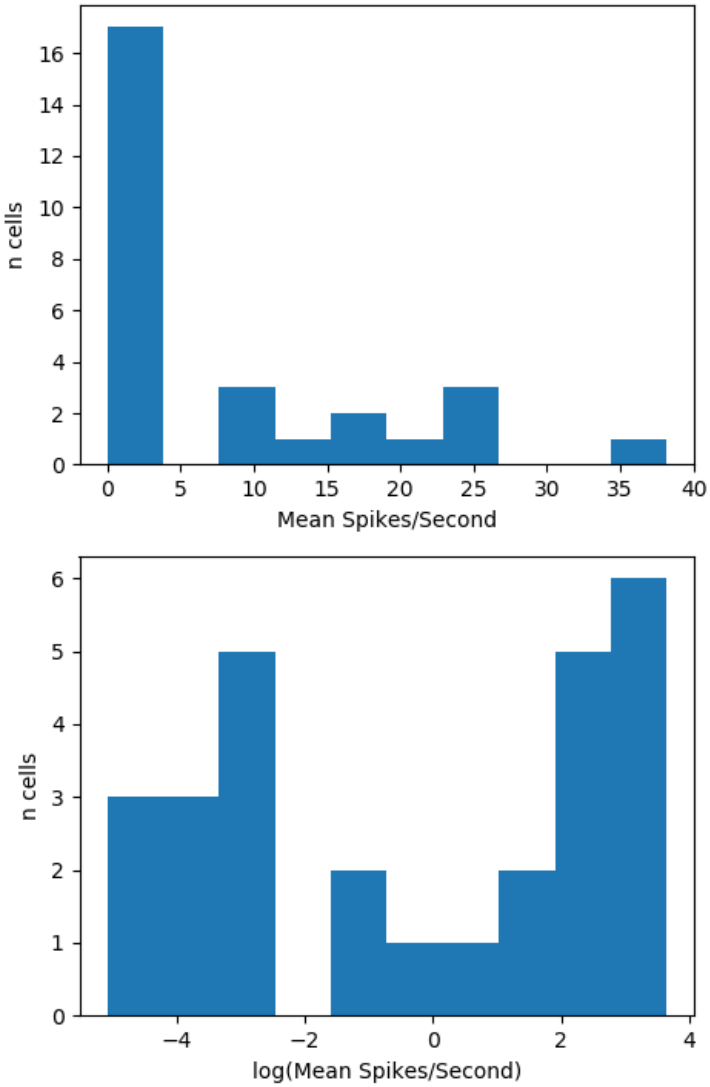

Figure 6: Distribution of average firing rates across all recorded neurons on regular axis (upper) and natural log axis (lower).

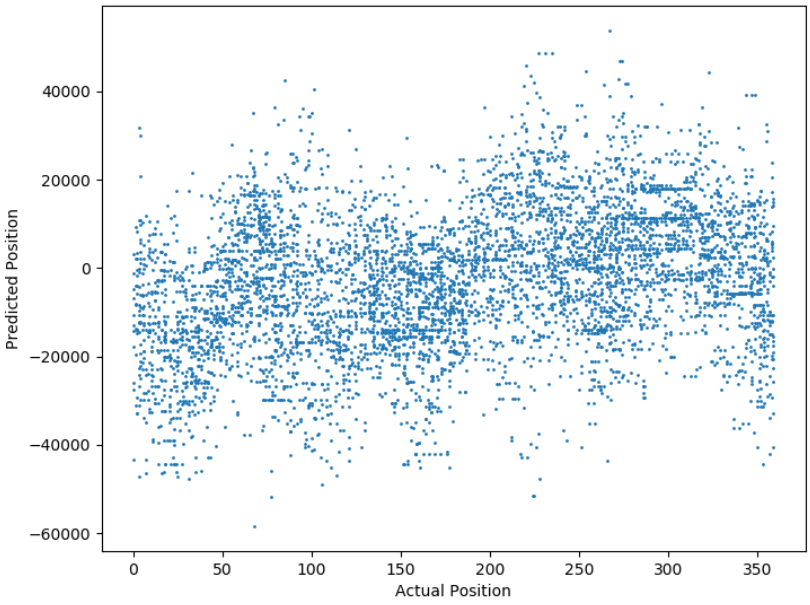

Figure 7: Regression Decoder: Position predicted by decoder vs. actual position for all test data points.

fluctuations will disrupt the spatial code, so there may be a few data points with very large error, but these are less important than reducing the error for the majority of data points. The median is more robust to outliers than the mean or mean squared, so for this problem it is the correct choice. The evaluation function is

$$E = \text{median } d_\theta(f(X), \mathbf{p})$$

Each of the decoders was trained on a randomly selected 3/5ths of the data and tested on the other 2/5ths.

3.1 Regression Decoder

The Regression Decoder is the most simple framework that I tested. It assumes that the position is simply a weighted sum of neural activity, *i.e.* $f(x) = Wx + b$. As figures 7 and 8 show, unsurprisingly this decoder does not perform well. The median testing and training error are 90 degrees, and the distributions of testing errors is essentially flat. However based on figure 7, this may even be overly optimistic. The some of the predicted values are orders of magnitude off, which is likely a function of the inflexibility of this framework: if an input point is an outlier, the output will also be. The errors here are only within the 0-180 range because the distance computed is an angular distance, and these ridiculously large values are "wrapping around" the track many times. This means the algorithm is operating near chance and no spatial information is being extracted. Clearly, a more nuanced decoder is needed to capture the complexity of this system.

3.2 Bayesian Decoder

The Bayesian Decoder is the most commonly used decoder within neuroscience[1] [2]. It is appealing because it allows us to encode our assumptions about variability in firing and provides some measure of uncertainty, since the probability of each possible output is returned. In the Bayesian model, $f(x) = \underset{p}{\text{argmax}}\ P(p|x)$ where $P(p|x)$ is the probability that the position p corresponds to neural activity x. With Bayes theorem, we see

$$P(p|x) \propto P(x|p)P(p)$$

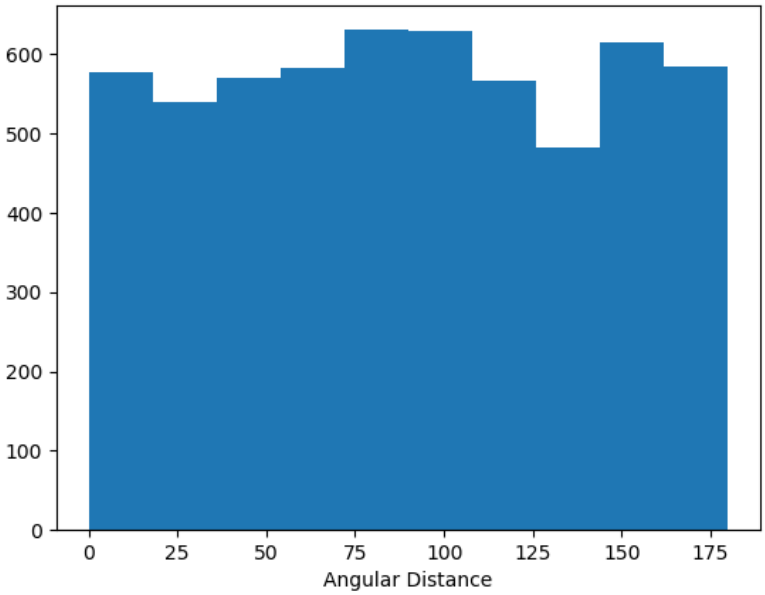

Figure 8: Regression Decoder: Normalized histogram of angular distances between predicted and true positions.

$P(p)$ is the prior on the rat's location and can be computed by normalizing the occupancy of all positions, or the proportion of time spent at each location. The prior is approximately flat across the map, however some minor heterogeneity may influence results.

$P(x|p)$ is the likelihood of getting neural response x when the rat is at location p. Assuming independent neurons, this can be computed as a product over n neurons, *i.e.* the probabilities that the ith neuron has response x_i at location p:

$$P(x|p) = \prod_{i=1}^{n} neuralnetwork P(x_i|p)$$

Neural spiking is well modeled by a Poisson process, which means the variability in firing rate can be modeled by a Poisson distribution. The expected value of this distribution should be the expected firing rate, which we assume changes with position. If $\bar{x_i}(p)$ is the expected firing rate of neuron i at position p,

$$P(x_i|p) = e^{-\bar{x}_i(p)} \frac{\bar{x}_i(p)^{x_i}}{x_i!}$$

All together, this provides the information needed to compute $P(p|x)$ and $f(x)$.

As figures 9 and 10 show, this decoder performs much better than the Regression Decoder, likely because the Poisson firing distribution elegantly handles the variability in firing rate that likely impeded the Regression Decoder. The median training and testing errors for the Bayesian Decoder were 15 degrees which corresponds to 13cm on the track. The body of a rat is about 10 cm, so this estimate of position is well within functional limits. However, figure 9 shows some "streaks" of data points horizontally, that indicate that some position are being predicted for neural activity that corresponds to a wide range of true positions. I suspect that these "preferred" positions may be a result of a heterogenous prior. If the likelihood for some activity patter is low across all positions, the prior will dominate the prediction function and the positions where the animal spent the most time will be predicted. This is one of the shortcomings of the Bayesian method that I intended to eliminate with the Neural Network Decoder.

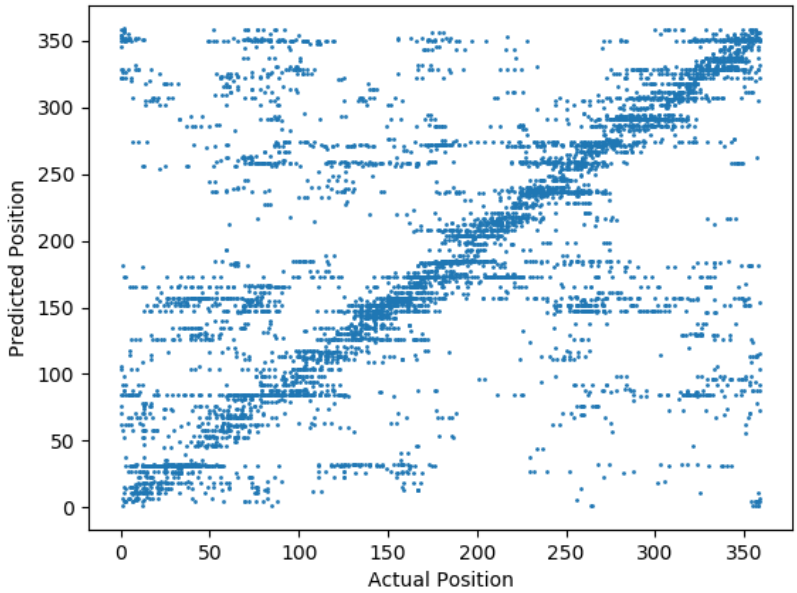

Figure 9: Bayesian Decoder: Position predicted by decoder vs. actual position for all test data points.

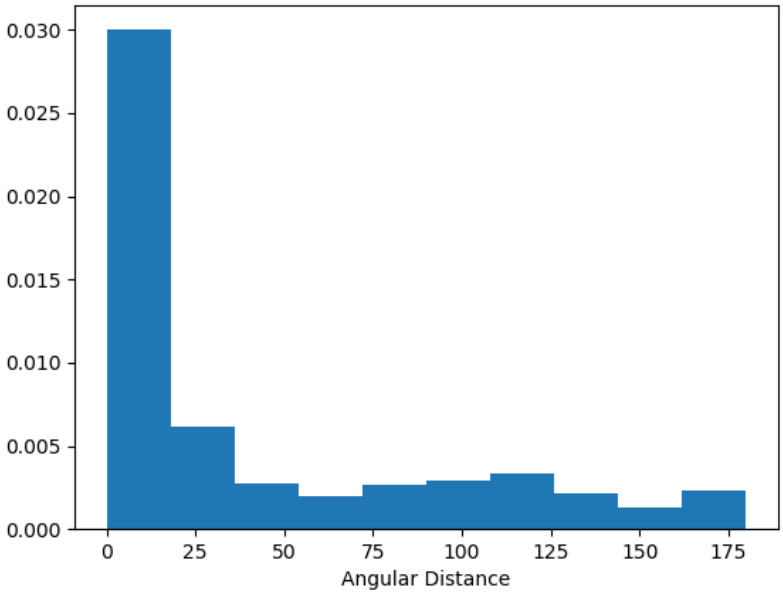

Figure 10: Bayesian Decoder: Normalized histogram of angular distances between predicted and true positions.

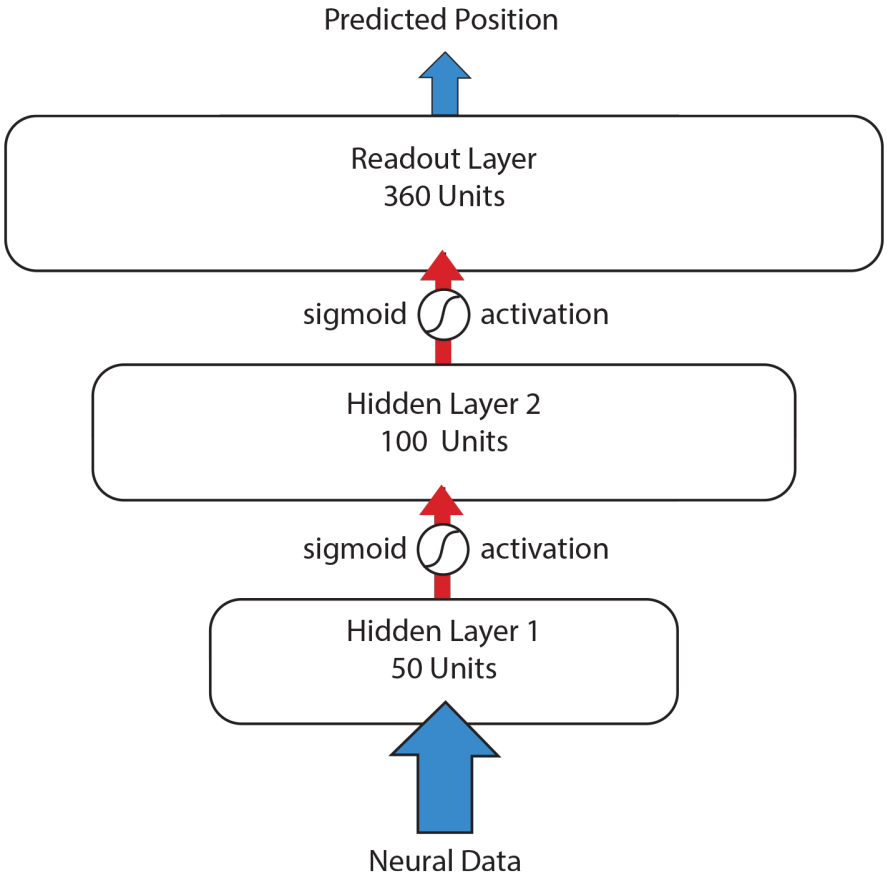

Figure 11: Neural Network Decoder: Shematic diagram of neural network architecture used in the Neural Network Decoder.

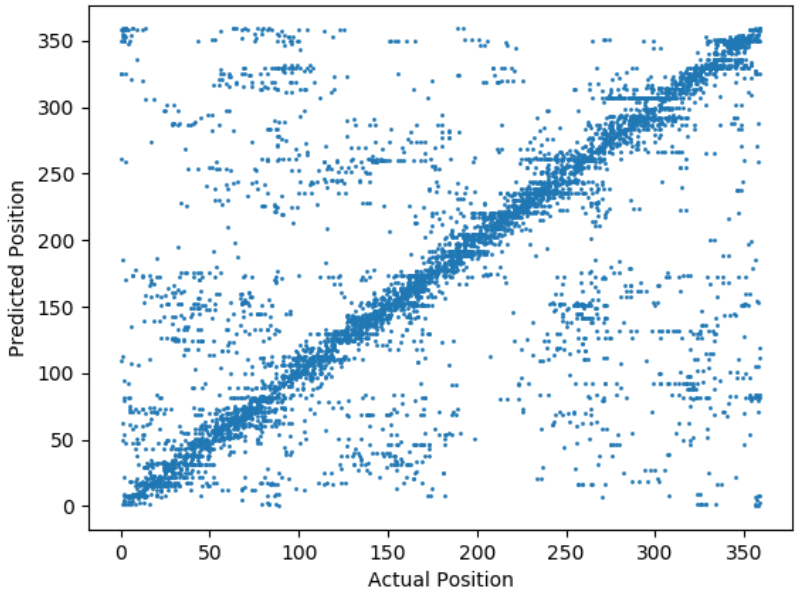

Figure 12: Neural Network Decoder: Position predicted by decoder vs. actual position for all test data points.

3.3 Neural Network Decoder

I designed the Neural Network Decoder as a relatively simple classification network. As figure 11 shows, the network has 3 layers, with 50, 100 and 360 units in each– the largest layer corresponding to the 360 spatial bins. Each layer takes a weighted sum of its inputs then computes a nonlinear activation function of the result. A sigmoid activation function was used in each layer. The weights in the weighted sum are the component of the network that are updated using back-propagation. The network was trained using minibatch gradient descent and a softmax cross entropy loss function. This function measures the difference between the predicted probability distribution for each position and the true probability distribution for a position (which is a delta function centered on the true position).

The Neural Network Decoder outperformed the Bayesian Decoder by nearly a factor of 2. The median training error for this decoder was 7 degrees and the median test error was 8 degrees. This corresponds to 7cm on the track. Neural networks are very effective for extracting patterns from noisy data, and it appears that when applied to neural data they perform better than the other decoders.

The improvement in accuracy that came with this decoder also came with a cost. The training time for this decoder was orders of magnitude larger than the others. While the Regression Decoder took a few seconds to compute and the Bayesian Decoder a few minutes, the Neural Network Decoder took over three hours to train. Figure 14 shows that over 5 million iterations this network still had not converged.

4 Conclusion

In this investigation I showed that the problem of accurately decoding information from a population of neurons requires more sophisticated techniques than a simple linear regression. The commonly used Bayesian Decoder can decode a rat's position with reasonable accuracy, but may at times rely too heavily on the prior. A new Neural Network Decoder was able to decode position with substantially improved accuracy, but training this decoder was computationally costly. The choice between a Bayesian Decoder and a Neural Network Decoder relies on the priorities of the user. If computation time is a priority, the Bayesian Decoder is a viable option. If accuracy is a priority, and independence from a prior, then a Neural Network Decoder is the best choice.

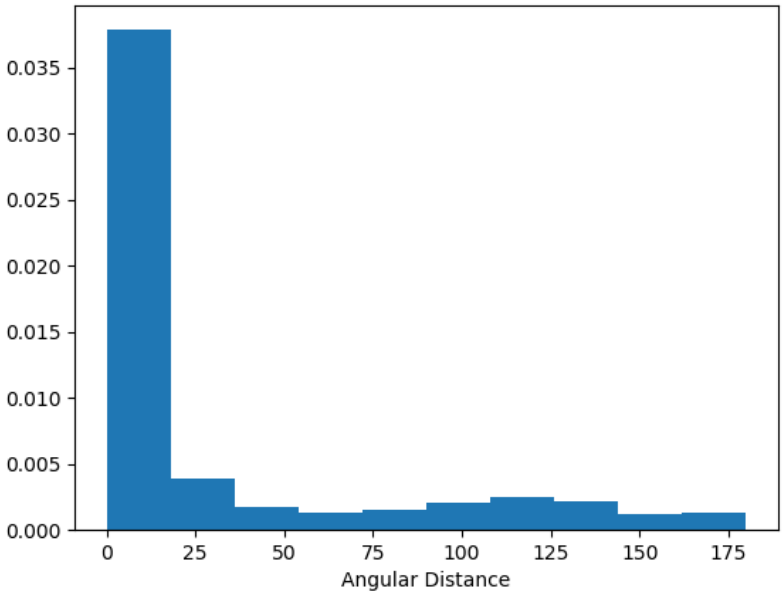

Figure 13: Neural Network Decoder: Normalized histogram of angular distances between predicted and true positions.

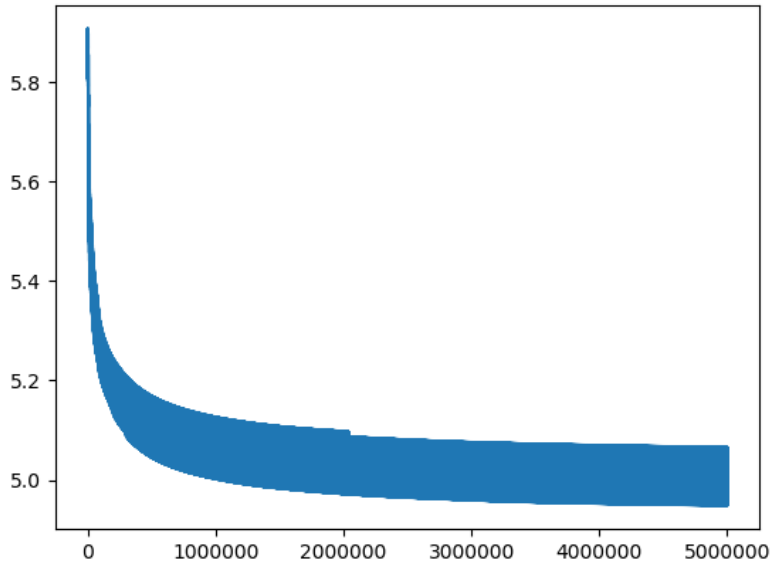

Figure 14: Neural Network Decoder: Training error for neural network throughout 5 million iterations of minibatch gradient descent.

5 Future Directions

Next steps in this work can fall into two categories: 1) Refining the features of the decoders to maximize performance and 2) Using the decoder to answer scientific questions about the brain.

In the first category there are a few obvious avenues for exploration. The intuition that the Bayesian Decoder is at times being dominated by a prior could be tested by substituting alternative priors. Perhaps a flat prior across the entire track would improve the accuracy of the Bayesian Decoder. The assumption of Poisson firing rate distributions for individual neurons could also be tested by substituting different models for the observed variability (there is substantial debate in the field on whether neurons obey Poisson firing dynamics). The Neural Network Decoder provides the most room for exploration. The network architecture that I chose was designed to gradually increase the number of units per layer, following a vague notion of continuity promoting stability. It is possible that different numbers of units, or different ordering of layers could improve performance. I would also like to know if a two layer network could achieve the same accuracy. If so, this would be faster to train and could ameliorate the primary drawback of this decoder. The activation function was also chosen based on previously observed improvements in classification problems. However, a different nonlinearity could have better performance. Exploring the parameter space of this network would likely lead to improvements in performance.

In the second category of exploration there are quite a few interesting questions to be asked. The hippocampus is known to *usually* represent the current position of an animal. However, there is some debate about weather the hippocampus may also represent locations that the animal is thinking about. For example, if the animal is running towards a reward, does the neural representation of the reward location intermingle with the current location of the animal? Using the decoder I could measure whether the erroneous data points are structured in some way that reflects the behavior of the animal. Along these same lines, it is known during sleep the hippocampus will replay events that occurred while the animal was awake in a process of memory consolidation. With an accurate decoder, I could train the decoder on data from an animal performing a task, then examine what the decoder predicts from the neural activity during sleep. Perhaps the process of retracing events could be observed in the predicted positions.

Overall an accurate decoder provides an exciting opportunity to understand the information that the brain uses and stores. With this knowledge we can develop exciting ways of interacting with the brain. One example that is particularly relevant for the hippocampus is in developing a neurally guided wheel chair. Many prosthetics are controlled by activations of muscles or neurons in motor cortex that directly control muscles. The hippocampus is further removed from these movement centers but as shown above, contains substantial spatial information. If a wheel chair could be controlled in a manner that bypasses movements and navigates directly to a location that a user wants to go, this would be an exciting development that is only possible with a precise spatial decoder.

References

[1] Zhang, Kechen, Ginzburg, Iris, McNaughton, Bruce, Sejnowski, Terrence (1998) Interpreting Neuronal Population Activity by Reconstruction: Unified Framework with Application to Hippocampal Place Cells. J. Neuro.

[2] Van Der Meer, Matthijs A. A., Carey, Alyssa A., Tanaka, Youki (2017). Optimizing for generalization in the decoding of internally generated activity in the hippocampus. Hippocampus.