Comparison of machine learning architectures for neural decoding of self-location

Balazs Agoston Nyiro

Supervisor: Silvia Maggi

School of Psychology
University of Nottingham

September 2022

I would like to thank Anna Hoyle, my parents and my supervisor Silvia Maggi for their unwavering support without which this work would not have been possible.

Abstract

In the past decade with the rapid development of computing power and machine learning, many possible new tools emerged for decoding the neural code of the brain. However, the drawback of the now-classic neural network architectures is that they are only able to decode complex electrophysiological signals in time and space, after considerable pre-processing and a long learning time. In this project,I tested the capabilities of three machine learning-based technologies that have emerged and have not been used at all for this purpose or very superficially. The three methods are behavioural cloning, transfer learning and hybrid CNN-BiLSTM networks. Also, five of the most commonly used neural preprocessing methods were tested, finally, the three models with thoughtfully chosen parameters were compared on self-location encoding signals recorded from the CA1 region of rats. The work explored the drawbacks of each technique and setting and confirmed the suitability of behavioural cloning for accurately decoding animal velocity from neural code.

Contents

1. Introduction

Direct decoding of the signals recorded from neural processes has long been considered a Holy Grail of neuroscience, which could help us better understand how the human brain stores and transmits information in our everyday actions. This typically requires very detailed and noise-free data and huge computational power. The complexity of the task is compounded by the fact that the mammalian brain is typically multiplexed (Walker et al., 2011), meaning information is transmitted over countless parallel channels while performing a single task. The resulting robust - both spatially and temporally high dimensional - feature representations are difficult to deconstruct using more traditional mathematical tools. However, among the first successes, as early as the second half of the 1900s, several studies have demonstrated that, during orientation in familiar environments, pyramidal cells in the CA1 and CA3 of the mammalian hippocampus show site-specific firing patterns (O'Keefe and Nadel, 1978; Wilson and McNaughton, 1993). These pyramidal cells are called place cells. It is likely that these cells enable contextual processing of information and, through this, memory-based spatial orientation (Frank et al., 2000; Hölscher et al., 2004; O'Keefe and Nadel, 1978; Wood et al., 2000). The activity of place cells is typically constant and is not changed much by the removal of objects from slace (O'Keefe and Nadel, 1978; Quirk et al., 1990). Because upon changes in spatial exposure, the firing correlates of place cells rapidly reform and place cells change their firing rate, forming their own representation of the changed environment (Anderson and Jeffery, 2003; Bostock et al., 1991; Leutgeb et al., 2004).

Due to this stimulus-action direct coupling, the position of the individual can be decoded with high accuracy from signals recorded from Hippocampal areas (Frey et al., 2021; Wilson and McNaughton, 1993). However, the information stored as place code by place cells is not so static in all situations. Research published in 2012 by Zaneta Navratilova et.al. confirms that in situations where the animal stops or takes short breaks in its journey to obtain food, the place code is able to become independent of the current position of the rat (Wilson and McNaughton, 1993). The most likely reason for this is that the animal's brain retrospectively recalls from memory the route it has already taken (Davidson et al., 2009) or plans the route it wants to take in the future Pfeiffer and Foster (2013). A further observed phenomenon is that although the exposure to a new

place is relatively fast, the resulting place code can be observed for hours (Barry et al., 2012; Bostock et al., 1991; Karlsson and Frank, 2008; Wilson and McNaughton, 1993). All these circumstances make the place code more of a dynamically changing feature. Several recent studies have shown that the spatial position storage of hippocampal cells is closely linked to other features that describe the animal's locomotion. For example, it is now known that, in addition to the location of the animal, the position of the animal's head and the speed of its movement can be decoded from, for example, the firing patterns of pyramidal cells (Hafting et al., 2005; Kropff et al., 2015; Sargolini et al., 2006).

In the beginning, classical methods such as linear regression or linear-nonlinear-Poisson cascade models were used to decode the relationship between the fixed signal and the behaviour (Corrado et al., 2005; Kropff et al., 2015). Another popular approach is to compute the probability distribution from the fixed signal using Bayesian methods. The disadvantage of this approach is that its accuracy is highly dependent on the accuracy of the available data (Davidson et al., 2009; Frey et al., 2021; Towse et al., 2014). This can be greatly worsened by the noisy nature of brain recordings. Another serious drawback of the technologies mentioned so far is that they usually require a high degree of preprocessing and guessing about the neural response. The need for a model-free decoding policy has arisen. Over the last decade, neural networks have become increasingly efficient in training high-dimensional data into lower-dimensional labels. Within this area, training with so-called deep learning models is also prominent.

One of the most commonly used families of models is the so-called convolutional neural network (CNN), which uses kernels to extract features from one or two-dimensional data. In research published in 2021, Markus Frey et.al.'s team first converted signals recorded using tetrode electrodes implanted in the brains of a total of 5 rats into images using wavelet transforms (Frey et al., 2021). These images were then used to teach the convolution algorithm they designed - a total of 15 convolutions and 2 fully connected layered neural networks (Frey et al., 2021). The conversion of the signal to the frequency domain also provided an opportunity to demonstrate the previously suggested relationship between the frequency bands of the pyramidal cell signal patterns and self-localization (Frey et al., 2021). In their research, they found that the CNN not only achieved higher mean accuracy in predictions but also resulted in fewer large errors than the previously used Bayesian decoder (Bayesian decoder: 23.38 cm ± 4.35 cm mean error, CNN: 17.31 cm ± 4.40 cm mean error) (Frey et al., 2021). In terms of frequency coupling, the frequency bands between 150 and 250 Hz carry the most information about the animal's position among the input data filtered for different frequency ranges presented to the trained model (Epsztein et al., 2011). However, CNNs can only learn patterns in a given time window. A recurrent neural network is better suited to learning the relationships between temporal sequences of the signal and already has a memory

of previous time instants. This is proved by an experiment by Ardi Tampuu et.al. in which they converted signals recorded on rats into spike count vectors and trained their network of 2x512 long-short term memory (LSTM) layers (Tan and Le, 2020). Using this technique, they measured a mean error of 10.18±0.23 cm (Frey et al., 2021). Further experiments have shown that not only can the LSTM achieve the lowest possible mean average error (Xu et al., 2019), but also that this model can be used to predict the animal in situations where, for example, in a T-branch, the rat has to decide which direction to progress further (Jude and Hennig, 2021).

Finally, one of the latest and most promising decoding results was achieved with XGBoost(XGB). By predicting firing rates, the gradient boosted trees technology could decode the direction of the animal's head from the recorded signals with the lowest mean average error(MAE) of all major machine learning technologies. While the error value for the second lowest LSTM was 10.25 degrees, XGB generated an average error of 9.12 degrees (Xu et al., 2019). Further increasing XGB's advantage is the fact that LSTM required almost two hours of learning to achieve this result, while XGB's learning time was nearly six minutes (Xu et al., 2019).

In this work, I investigated the validity of three deep learning methods that have not been tried or are not commonly tested. First, we tested the behaviour cloning network used in self-driving cars, which is able to efficiently learn the correlations between presented patterns and the responses to them. Second, an extended transfer learning model was used to test whether a network pre-trained on a large set of RGB images can perform better on 124 channels than a CNN without this advantage. Finally, the advantages of the CNN and LSTM models were combined in a hybrid CNN-Bidirectional LSTM network in order to extract spatial and temporal features as efficiently as possible during training.

2. Methods

2.1 Dataset

For the comparison of the models, the tetrode record from the article published by Markus Frey at.al. was used. In this study, the aim was to record brain activity during food acquisition in rats. The team used five male rats with two single-screw microdrivers implanted in their heads. These targeted the left and right CA1 regions of the brain. Both devices contained 2x16 tetrodes that transmitted a total of 128 channels of local field potential signals. The recording frequency range was between 0 Hz and 15000 Hz and the sampling rate was 30 kHz. In parallel with the tetrode signals, the position of the animal's head was recorded by mounting two infrared LEDs of different intensities on the two drivers, which were fixed using a Raspberry Pi camera V2. The sampling rate of the camera was 30kHz and the resulting position data were synchronized with the data from the electrodes.

The animals were kept in 12-hour dark/light cycles for a week after surgery, during which time they were given training tasks related to the subsequent tasks. During the experiment, the rats were placed in a 1.75m x 1.25m area enclosed by a wall. During the 40-minute sessions, an automatic machine dropped chocolate-flavoured balls into the area to record a signal that could be used to detect the neural activity of spatial orientation that was assumed to occur during acquisition. The research team successfully identified putative interneuron signalling activity.

2.2 Preprocessing

Raw electrophysiological data is first passed through a cascade of filters and then converted to a frequency representation using a continuous wavelet transform (CWT). The individual elements of the filter pipeline can be set or disabled to compare the efficiency of different pre-processing versions. A large percentage of filters cause edge effects that can lead to the loss of important information. To avoid this problem, the signal was cut into 30-second time windows, with a few hundred milliseconds of extra time at each end, overlapping with the chunk before and after. These are cut off at the end of the process, thus removing the affected fragments. The raw signal is first passed through a band-pass filter (0.1Hz-250Hz), which helps to highlight the frequency range to be

tested, thus filtering out unwanted noise. Although a higher low cut threshold is often given in the literature (1Hz instead of 0.1Hz), this often allows artefacts to be included in the data (Tanner et al., 2015). The Chebyshev filter was chosen because it has the advantage over other band-pass filters (e.g. Butterworth filter) of much better performance in terms of frequency response (Pozar, 2011).

Line noise (50Hz) and its harmonics were filtered with a notch filter. In the next step, the signal is downsampled to 2kHz. Experience has shown that this is the range where the learning quality is not compromised, but the reduced amount of data speeds up the operation. The resampling process is Fourier and the starting point is the same as the ending point, thus reducing the edge effect. The rate of downsampling is given by

$$window = \left(\frac{l}{s_o}\right) * s_n \tag{2.1}$$

, where l is the input signal length, s_o is the old sampling rate and s_n is the target sampling rate.

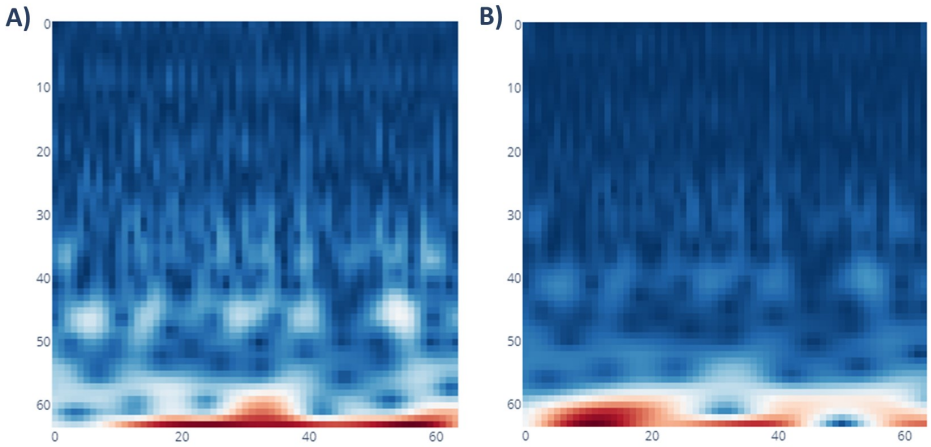

Figure 2.1 The wavelet spectrum for the raw signal(figure on the left), and for the downsampled data (figure on the right). In this case, the raw signal(with a 30,000Hz sampling rate) was wavelet transformed with a Morlet wavelet between 2 and 7,500Hz. After the downsampling, the data had a sampling rate of 20,000Hz.

Robust detrending has been applied to filter out any linear shifts in the recorded signal. This method allows to remove these trends by subtracting the linear least squares of the signal. The last step before extracting features is to compute the moving window mean of the signal, which can help in extracting slowly changing cortical processes, for example when predicting motion (Blankertz et al., 2001). Here, preliminary experiments with a window size of 20 milliseconds gave the best prediction results.

2.3 Feature extraction

In the case of convolutional neural networks, feature extraction is done using a complex wavelet transform. This is also necessary because the input layer of these types of neural networks takes a multidimensional matrix or an image. The wavelet transform allows the analysis of non-stationary power time series using the wavelet transform at different frequencies (Daubechies, 1990). After trying several waveforms, the Morlet wavelet gave the best frequency representation. In the case of Morlet wavelet, the waveform is a sine component multiplied by an element of a Gaussian curve, where the wavelet function(ψ) depends on a non-dimensional time parameter (η):

$$(\eta) = \psi_0(\eta) \;\; \psi_0(\eta) = \pi^{-1/4} * e^{i\omega_0\eta} e^{-\eta^2/2} \tag{2.2}$$

, where $\omega_0 = 6$ is a non-dimensional frequency component (Torrence and Compo, 1998).

The total frequency space of the signal was represented by 35 log-spaced Fourier frequencies between 2Hz and 15,000Hz. At the end of the preprocessing and feature extraction process, each chunk is cut into 2-second windows.

2.3.1 Calculate position, speed and head direction values

The position of the two LEDs on the head of the animals can be used to calculate their speed and head position. In the dataset, each of the two LEDs is described by an x and a y coordinate within the area the rats can walk. Since the sampling frequency of the tetrode and LED data is different, we use piecewise linear interpolation to fill in the missing data points. Then the velocity $v(t)$ is obtained as shown in the following equation:

$$v(t) = (\Delta x2 + \Delta y2)/lw \tag{2.3}$$

, where l_w is the length of each window. The angle of head position($\theta(t)$) is calculated in radians:

$$\theta(t) = atan2(\Delta x, \Delta y) \tag{2.4}$$

, where

$$atan2\left(x,y\right)=f\left(x\right)=\begin{cases}\arctan\left(\frac{y}{x}\right), & if\ x>0\\ \arctan\left(\frac{y}{x}\right)+\ \pi, & if\ x<0\ and\quad y\geq 0\\ \arctan\left(\frac{y}{x}\right)-\ \pi, & if\ x<0\ and\quad y<0\\ +\frac{\pi}{2}, & if\ x=0\ and\ y<0\\ -\frac{\pi}{2}, & if\ x=0\ and\ y>0\\ undifined, & if\ x=0\ and\ y=0\end{cases} \tag{2.5}$$

And the relative orientation of the head ($\theta_r\left(t\right)$) to the environment of the animal:

$$\theta_r\left(t\right)=\left[\left(\theta\left(t\right)+\ \delta+\ \pi\right)\ mod\ 2\pi\right]-\pi \tag{2.6}$$

, where the relative offset is $\delta=\frac{\pi}{2}$.

2.4 Decoder based on deep learning

Deep learning algorithms are a type of artificial neural networks(ANN), which, is inspired by the neural architecture of the brain. It can learn new parameter estimates by fine-tuning the weight values between nodes based on previous patterns. In deep learning networks, these nodes are organized in several layers, with a characteristic hierarchical structure consisting of an input layer that directly takes the size of the input to be learned. Then numerous hidden layers provide the "depth" of the network. Each layer can extract one more and more complex property from the patterns between the inputs. Finally, the output layer uses an activation function to translate these properties into the desired outputs. This hierarchical structure allows it to be able to efficiently solve regression or classification problems (Elman, 1990; Rumelhart et al., 1986). In this work, three types of deep learning solutions have been tested.

The first is behavioural cloning, which is part of the family of cellular neural networks(CNN). These models are designed to learn the rules followed during human cognitive functions (e.g. driving) and the subsequent correlations between actions (Sammut, 2010). Given that these end-to-end systems can efficiently reproduce or predict the complex behaviours of the operator (in this case, the rat brain), they may also be able to learn complex brain processes and subsequent behavioural patterns. The next technology tested is transfer learning. For this, a CNN-based model was tested, but the important difference is that the network has been pre-trained on a large dataset. It is possible to transfer the knowledge stored in the network weights to efficiently learn other problems related to the nature of the task. This ability can also help in the case of spec-

trograms, where hidden layers of a pre-learned network can more effectively extract the patterns encoded in them.

The disadvantage of the previous two networks, however, is that they only see the information for a short period before the action, and can only learn from patterns that occur in the same time window. This problem can be overcome by the so-called Long-Short Term Memory(LSTM), which already has memory, and thus has the advantage of being able to find correlations between patterns that are spatially and temporally distant (Pham, 2021). In terms of its structure, LSTM consists of cells, each of which contains a learn gate, a forget gate, a remember gate and a use gate. Learn gate(N_t) meets the recent predictions(RTM_i) from the short-term memory and the current event input(A_t) which are then merged and ignore those that are not relevant. The combination of the input weight matrix($W_{n,f,u}$) and the two other input parameters mentioned above is done by the hyperbolic tangent activation function($tanh$):

$$N_t = tanh\left(W_n\left[RTM_{t-1}, A_t\right] + b_n\right) * i_t \tag{2.7}$$

, where b_n is the bias. The forget gate(f_t) is used to decide which information to keep and which to forget:

$$f_t = \sigma(W_f\left[RTM_{t-1}, A_t\right] + b_f) \tag{2.8}$$

At the remember gate, the long-term memory from the forget gate and the short-term memory from the learn gate is used to form the new long-term memory(HTM_t):

$$HTM_t = HTM_{t-1} * f_t + N_t * i_t \tag{2.9}$$

Finally, the use gate forms the new short-term memory(RTM_t) from the long-term memory(U_t) received from the forget gate and the short-term memory(V_t) received from the learn gate:

$$U_t = tanh(W_u HTM_{t-1} * f_t + b_u) \tag{2.10}$$

$$V_t = \sigma(W_v\left[RTM_{t-1}, A_t\right] + b_v) \tag{2.11}$$

$$RTM_t = U_t * V_t \tag{2.12}$$

We used a hybrid CNN- bidirectional LSTM(CNN-BLSTM) network, where each hidden state of which uses a dedicated layer to extract contiguous information in both directions simultaneously in time. These networks have recently given very good accuracy in speech recognition problems (Graves et al., 2013) and, due to the similar nature of the problem, I decided to test the effectiveness of this technique for position prediction from tetrode signals from CA1 cortical layer.

2.5 Architecture

2.5.1 Behavioural cloning

Three-dimensional wavelets in 64x64 dimensions were applied as the input for behavioural cloning. Each of the five units in the architecture is composed of a convolution layer, a max pooling layer, and a dropout layer. The network can extract features from the input matrixes using these blocks. Three fully connected layers that allow mapping of the extracted information to the direction, location, and velocity of the head are then added after this. The batch normalization layer, a regularization technique to normalize the data and thus enable more accurate prediction, is implemented after each block in the final architecture. For the first convolution layer in each of the first five blocks, there were 16 filters, and there were 32 filters in total. Additionally, a 3x3 kernel size was applied. A 0.3 output value was chosen. Due to the problem's regression property, the model's final four output layers—one for velocity, one for head angle, and the final two for x, and y coordinates—all have linear activation. Due to this, Mean Square Error was also used to calculate the loss value (MSE). A total of 1 186 494 input parameters were used in the overall model. The output layer was the only layer not activated by a scaled exponential linear unit (SELU). This function returns exponential values for negative input values and linear values for positive values, as illustrated in the figure. This function has an advantage over the ReLu function in that it has the potential to generate negative results without killing neurons. Additionally, it enables quicker convergence. This model's output layer activation functions are sigmoid because they enable more subtle output value transitions than linear activation. To reduce the possibility of overfitting, every convolutional and dense layer was regularized by L1 and L2 functions.

2.5.2 EfficientNet B0

A popular convolutional neural network architecture called EfficientNet uses intricate coefficients to scale the depth and breadth parameters consistently. The EfficientNet scaling method, in contrast to conventional approaches, employs fixed scaling factors (Tan

and Le, 2020). It has the advantage that, despite being faster and smaller than ConvNet by more than six times and eight times, its prediction accuracy is not noticeably decreased (Tan and Le, 2020). This architecture was also tested alongside the NVIDIA model, with Imagenet weights pre-trained. The input is identical to the wavelet input used in the previous examples. These were frozen in order to prevent them from changing during the learning process. This strategy allowed us to take advantage of the model's previously obtained feature extraction skills by training it on multiple images from its own database, resulting in higher accuracy. We fed the EfficientNet B0 network's last layer before the output layers through a global average pooling and a batch normalization layer. The four output layers are built up by a dense layer with L1 and L2 regularization.

2.5.3 CNN-BiLSTM

Lastly, a hybrid CNN-BiLSTM architecture was tested. The input layer receives the signal normalized between -1 and 1 as a channel, only one time-window at a time. In this case, the preprocessed but not yet wavelet transformed data sequences are fed into a 1D convolutional layer at the CNN layers. CNN uses convolution and pooling techniques to extract implicit features from the input data. A fully connected layer is then provided with the combined extracted features. Finally, a RELU activation function is used to add nonlinearity to the output of the neurons. After a dropout layer, the highlighted features are put into the Bidirectional LSTM branch. Here it passes through a total of 100 layers. Lastly, before the last Flatten, Dense and Activation layers, a three-dimensional attention module is included. These modules can be used to determine which time step in bidirectional LSTM learning has the largest effect on the final result (Dubey et al., 2019). As with the meshes detailed above, predicted results are obtained through four linear activation layers.

2.6 Hyperparameters and training

The available data was split 70:20:10 ratio between the training, validation and test datasets. These data sets were randomly arranged. In order to avoid the end of a process already seen during training could leak into the test data set, a window between the training, validation and test data sets is not used. As the available server performance allowed, the training was performed over 200 epochs with a learning rate of 0.001. The learning rate during training is exponentially decremented every ten steps by a learning rate/epoch ratio. TensorFlow's built-in early stop function was used for regularization, allowing us to acquire the best loss version of our model while avoiding overfitting. When determining the learning rate, remember that while a low learning rate would significantly increase training time, a relative value would suggest too big gradient update steps, which would risk exceeding the recommended minimum value. Finally, the best

value was determined by guessing. The dropout probability is set between 0.3-0.5 for both models. For the optimizer, we chose the Adam optimizer, which is now an industry standard and allows us to handle sparse gradients, even for noisy problems.

Regarding batch size, the behaviour cloning model produced the best accuracy and lowest loss with a value of 30. However, EfficientNet produced excellent results with a value of roughly 20. Because there was no widely established approach for this, it was likewise decided via trial and error. The settings indicated above were optimized based on the findings of various experimental experiments. The best-performing trained neural network and its weight values were saved in.h5 format at the end of the learning process using TensorFlow's save function.

2.7 Model evaluation

Since the data published by Frey et.al.(2021) was used in training, the same four measures they defined were used in this study for the purpose of comparability. These measures are used to infer how effectively the different models capture the relationships in the data and how accurate predictions can be made later (Frey et al., 2021). For the x and y coordinates of the position, the difference between the predicted and ground truth values is given by Euclidean error($\mathscr{L}_{ED}$), while for the velocity, this is calculated by mean square error($\mathscr{L}_{MAE}$). The loss of the head direction is given by cyclical absolute error($\mathscr{L}_{CMAE}$)The different models were compared using R^2 score and cumulative distribution. These measures were defined as follows:

$$\mathscr{L}_{ED} : \quad \sqrt{\sum (\hat{y} - y)^2} \tag{2.13}$$

$$\mathscr{L}_{MAE} : \quad \frac{1}{M} \sum_{i=1}^{M} \left| \hat{y}_i - y_i \right| \tag{2.14}$$

$$\mathscr{L}_{CMAE} : \quad \min[\left| \hat{y}_i - y_i \right|, \left| \hat{y}_i - y_i \right| + \pi, \left| \hat{y}_i - y_i \right| - \pi] \tag{2.15}$$

$$R^2 : \quad 1 - \frac{\sum_{i=1}^{M} (y_i - \propto \hat{y}_i - \beta)^2}{\sum_{i=1}^{M} (y_i - \hat{y}_i)} \tag{2.16}$$

2.8 Code

The program codes for the project are available here: https://github.com/nyirobalazs/thesis/

3. Results

In the next section, I will present the steps, settings and results used to test the models. First, the impact of the filters and steps used during preprocessing on the training data is described. Second, the best results that can be obtained by each model and the settings required to obtain them are examined. Finally, the results obtained by the three models are compared.

3.1 Pre-processing steps

First, I investigated the effects of the various digital filters most commonly used in neural signal processing and developed them for this project. For this purpose, I used the signal from the CA1 region of the hippocampus, bilaterally implanted in rat electrodes (Frey et al., 2021). Of the 128 available channels recorded using 32 tetrodes, I used only the 124 good channels identified in the data. The raw signal was passed as 30,000-millisecond chunks through signal filters with preset parameters in each round. The processed signal was then cut into windows of 2,000 milliseconds.

Since the total dataset was enormous (almost 16GB) and due to the limited computing power (45GB RAM, Nvidia RTX A5000 GPU, 8 core CPU) it would have been very long to run the whole preprocessing process with all the options, so only a randomly selected 10% of the total dataset was used to be representative of all the data. Thus, using the window size mentioned above, there were 2820 windows of the reduced fixed signal. This was divided into training, validation and test parts in a 70:20:10 percentage ratio. The resulting data was trained as mentioned in the methods part using Behaviour Cloning(BC), Transfer Learning(TL) and hybrid CNN-BiLSTM supervised learning regression models. The ground truth velocity, head direction and position data for regression were calculated from the position of two LEDs attached to the animal's head. The tests were trained for 20 epochs, with batch sizes of 10 and a learning rate of 0.001. During training, the learning rate was exponentially decreased by 0.95 steps per 2 epochs. When testing filters, all settings other than the filter settings were fixed, as were the windows (randomly selected beforehand) used for training. Accuracy at each setting was calculated using the mean and standard distribution of the (288) results predicted from the

test data.

Due to the reduced data set, although the errors obtained were higher compared to the mesh trained on the full database, the aim of this step was to investigate the effect of the filters on the signal by generating comparable results in a standard environment.

3.1.1 Measuring baseline

First, I measured the decoding losses of the three models without filters. While the input of the CNN-BiLSTM is the raw/preprocessed signal, the Behaviour Cloning(BC) and Transfer Learning(TL) nets use the 64x64 reduced matrix after wavelet decomposition. Since only the two CNN models of the three models use two-dimensional inputs, this can only be compared for these two models. For the wavelet transform, we use a Morlet wavelet with a log-piecewise frequency scale as mentioned in the methods section. Since the frequency range of the signal in the dataset is between 2 and 15,000 Hz (Frey et al., 2021), I first ran the training without any filtering over this full frequency range to obtain the baseline loss values. Based on these results, the three models gave similar accuracy for the velocity data (CNN-BiLSTM mean: 7.74± 6.66 cm/s, BC: 6.47± 4.97 cm/s, TL: 8.53±7.22 cm/s), with a slight advantage for the behaviour cloning model. In terms of decoding the head direction, the TL and BC models produced similar results (BC: 0.93± 0.65 rad, TL: 9.15± 0.65 rad), while the CNN-BiLSTM model was nearly twice as inaccurate as the previous two (1.89± 0.95 rad). In terms of position, the X coordinate was decoded with an average error of 21% (CNN-BiLSTM mean: 24. 3±16.47 cm, BC: 28.31± 19.36 cm, TL: 25.26±41.325 cm), while the other one could be decoded with an average error of 16% (CNN-BiLSTM mean: 18.61±12.25cm, BC: 43.64± 20.91 cm, TL: 19.22±12.92cm). Here again, it can be seen that the coordinates were extracted from the wavelet transformed inputs with a much higher inaccuracy. In order to better understand this phenomenon and the signal, I first tested in the transfer learning and behaviour cloning networks how using a lower value for the number of voices per octave(nv=3) than the wavelet transform affects the results. This did not produce a coherent change in either case. In the next case, I first trained the two models with only high-frequency signals filtered between 2-150 Hz and then between 250-7,500 Hz. While narrowing the second frequency range resulted in a slight improvement for the BC network, a slight deterioration was observed for the transfer learning network. However, this is not significant. However, when the signal was wavelet transformed only in the range 2-150Hz, a strong degradation in the loss values was observed except for the speed (BC: 11.49± 1.17 cm/s, TL: 2.12±1.67 cm/s). More detailed results can be found in the appendix at the end of the publication(Figure 4.1).

3.1.2 Effect of filters on detection accuracy

In the analysis of neural signals, various digital filters are often used to remove residual noise in the signal. In our experiment, I developed the five most commonly used filters and investigated their effect on the accuracy of the learning process. The first of these filters was the Chebyshev-type band-pass filter. This filter is used to highlight the frequency to be tested. Here, in all cases, the incoming raw signal was filtered between 2 and 7,500Hz. The lower limit of 2Hz allows the filtering of low-frequency noise introduced as background noise (Tanner et al., 2015). The value of the high cut was determined on the basis that no significant changes were seen in this range in preliminary tests, and that adding or omitting it did not change the loss values in particular. Based on the three neural networks I tested, the use of a bandpass filter for the baseline values did not produce a clear improvement in any of the predicted values.

The next stage is the narrow band filter used to filter out line noise, which filters out the 50Hz from the mains (given that the signals were recorded in England) and its harmonics. When this is applied, it is generally said that all loss values are degraded. The degradation is perhaps most noticeable in the speed values, where the three neural networks achieved 33% worse accuracy than the baseline values on average (CNN-BiLSTM mean: 10.31± 8.47 cm/s, BC: 7.62± 4.73 cm/s, TL: 15.68±17.82 cm/s).

Downsampling is an established signal processing tool, allowing faster processing due to the smaller amount of data. Surprisingly, it also had a beneficial effect on the loss values, for example, the speed values were reduced by almost 40% after reducing the signal originally sampled at 30kH to 20kHz (CNN-BiLSTM mean: 5. 09± 4.33 cm/s, BC: 3.88± 3.57 cm/s, TL: 4.5±3.45 cm/s) and at the 10kHz sampling rate, this value reached 67% (CNN-BiLSTM mean: 2.69± 2.35 cm/s, BC: 2.14±1.63 cm/s, TL: 2.66±2.57 cm/s).

When observing the signal with the naked eye, it appeared that linear trends could be observed, which could have been due to the displacement of the sensor in the tissue. Contrary to expectations, the application of a detrend filter did not bring any significant improvement. In the prediction of head direction and animal velocity, I found that CNN-BiLSTM and transfer learning nets showed some improvement, but on the contrary, the behaviour cloning neural network caused a slight deterioration in loss values. However, for this model, the most spectacular degradation occurred for the coordinates (X mean loss: 47.63±25.95cm, Y mean loss: 52.78±23.94cm). The last filter tried was the moving window mean filter, which I hoped would highlight cortical processes (Blankertz et al., 2001, 2011). However interestingly, although it did not yield significant improvements in decoding accuracy, but it did show a positive effect on the validation and training loss rolls recorded during machine learning mesh learning. In the case of more general

models such as CNN-BiLSTM or Efficientnet for transfer learning, the phenomenon of so-called overfitting is often observed when training on a smaller database. This is when the model does not learn to generalize correctly from the data presented to it (Shalev-Shwartz and Ben-David, 2014). This is most often observed in terms of an improving training loss and a stagnating validation loss. I have also observed this phenomenon sometimes in my models, but the moving mean filter has proven effective in eliminating it.

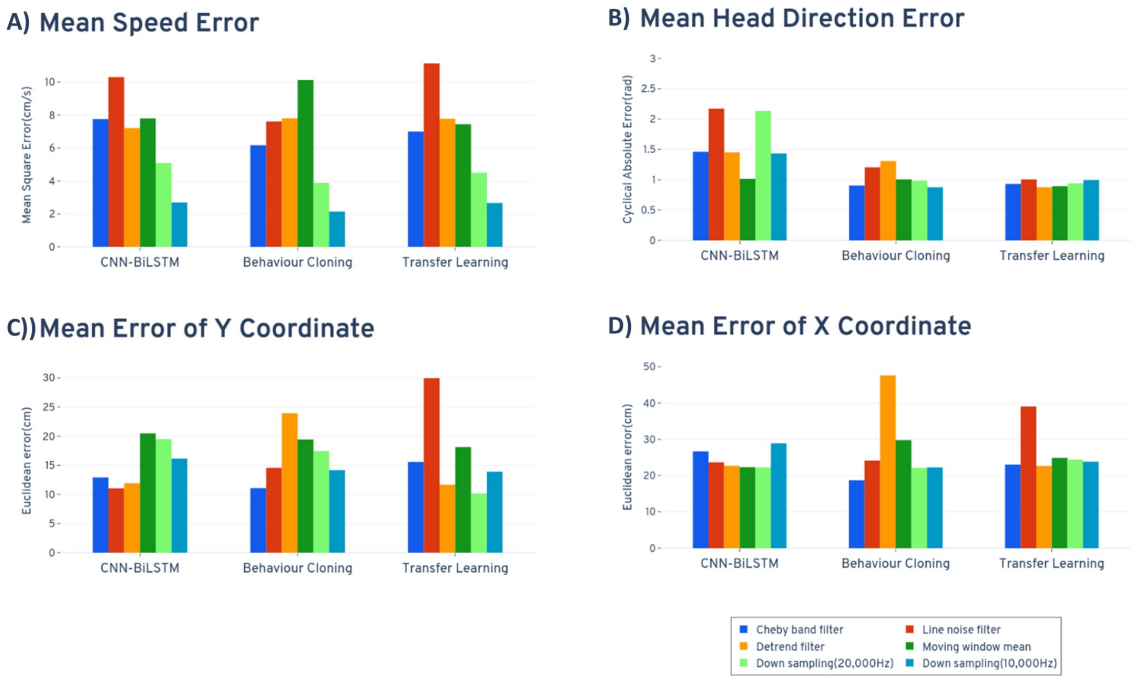

Figure 3.1 The effect of the filters used in preprocessing on the error value between the decoded output and the ground truth values. A) The mean square error loss for speed. B) The cyclical absolute error loss for head direction. C-D) Euclidean error loss values for coordinates

3.2 Comparing models through the accuracy of self-location

After adjusting the filters from the results obtained, downsampling the raw signal to 10,000Hz and avoiding overfitting, the moving window mean filter appeared to be the most effective. Therefore, I chose to use only these two filters. For wavelet decomposition, the frequency range was between 2-7,500Hz, where the number of voices was set to 5. To determine the best results obtained by the three models, I ran the training on the full database with batch sizes of 30 and validation batch sizes of 50. The learning could be run for a maximum of 100 epochs, however, to find the most optimal network I used early stopping which if the validation loss did not improve for 3 epochs stopped

the learning and saved the best network weights as the final result. The program first calculates the possible number of windows based on the length of the raw data. The order of window indices generated from these is shuffled, in order to minimize the RAM utilization of the neural nets To minimize the RAM utilization, the program preprocesses the next batch of window size(2,000s) sections with consecutive indices at each batch generation. The disadvantage of this is that preprocessing has to be repeated for each model. Using this method, it took 5-6 hours to train a model on a machine with the specifications mentioned earlier.

Tests on the full data set show that all three models decode the X coordinate from the recorded signals with a rather poor accuracy (CNN-BiLSTM mean: 36. 47±11.25 cm, BC: 23.56±12.73 cm, TL: 32.46±17.58 cm) and only slightly better results for the Y coordinate (CNN-BiLSTM mean: 34.91±23.53 cm, BC: 22.19±18.83 cm, TL: 25.4±14.87 cm).
Smaller losses have been achieved in decoding the direction of the head and the speed of movement. It are known that while the head direction is somewhat modulated by the spatial activity of place cells (Jercog et al., 2019; Yoganarasimha et al., 2006), animal velocity is also encoded by the amplitude and frequency of place cell firing rates during LFP activity between 7-10 (Jeewajee et al., 2008; McFarland et al., 1975). Head direction was also obtained by the behaviour cloning neural network with the lowest cyclic absolute error value (mean: 0.99±0.96 rad), while transfer learning (mean: 1.03±0.74 rad) and CNN-BiLSTM (mean: 1.46±0.61 rad) had a slightly worse loss value. In terms of velocity, the three models gave the best accuracy (CNN-BiLSTM mean: 1.91±1.31cm/s, BC: 1.03±0.83 cm/s, TL: 1.22±0.21cm/s). For the prediction of head direction and velocity values, it is also clear that while the behaviour cloning model can achieve better average accuracy, it underperforms the other two more complex models in terms of standard deviation values. Furthermore, the LSTM has a slight advantage in terms of variance, as it gives the smallest variance in the latter two categories.

To put the results in context, a model developed in 2021 by Frey et al., using the same data set and the same loss values calculated in the same way, was able to decode the animal's position with an Euclidean error of 17.7±4.96 cm (Frey et al., 2021). For the head direction angle, a cyclic absolute error of 0.80±0.18 rad was obtained, while for the velocity, a mean square error of 4.94±1.00 cm/s was obtained (Frey et al., 2021). This shows that while for position the three models we tested fall far short of their simpler CNN-based solution using wavelet decomposed data, they achieved head direction nearly as accurately and velocity nearly a fifth as well as theirs.

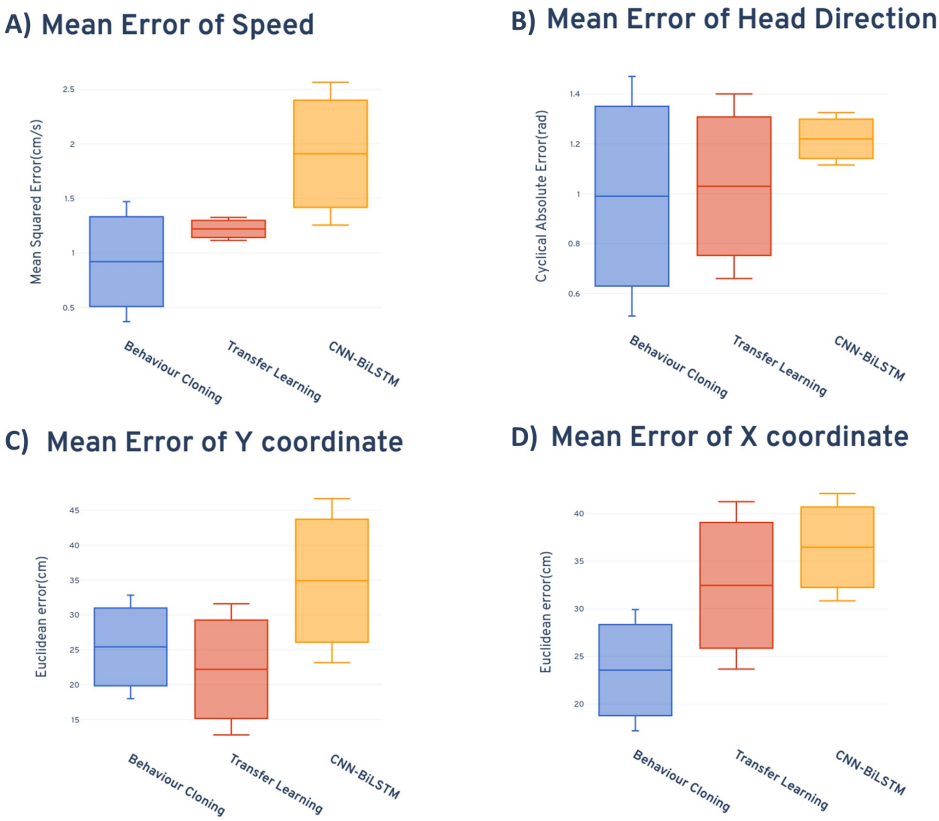

Figure 3.2 Loss values in the cases of neural network architectures, trained on the entire database. A) The mean square error loss for speed. For behaviour cloning, the loss value over the limited frequency band (2-250Hz) is shown. B) The cyclic absolute error loss for the head direction. C-D) Euclidean error loss values for coordinates.

3.2.1 More accurate speed prediction with narrower frequency range

Finally, since the previously detailed results show that speed is suspected to be encoded by lower frequency bands (Jeewajee et al., 2008; McFarland et al., 1975), I, therefore, tested whether it would improve the accuracy of decoding the velocity values if the upper limit of the wavelet decomposition was lowered to 250Hz, thus examining a specifically narrow band frequency range only. Here, I only changed the batch size from the previous settings to 15 to avoid over-generalization as much as possible. Due to the length of the test, it was not possible to try more variations, so I decided to run the test with the behaviour cloning model that had previously performed best in terms of speed data. During the experiment, learning reached loss minima very quickly (in 20 epochs). The average mean square error of the test run on 10% of the total data set was 0.92±1.1 cm/s. This was an improvement over the previous result of 1.03cm/s and yielded a markedly strong improvement in variance.

4. Discussion

The neural code provides a complex, non-linear representation of events, activities and cognitive states. One of the main goals of neuroscience is to read this code, which can shed light on the computations performed by brain networks. However, decoding it is a challenging task that requires a thorough prior knowledge of both the variables encoded and, perhaps more importantly, the way in which they are represented. One such well-documented piece of information, which is thought to be encoded by pyramidal cells in the CA1 region of the mammalian hippocampus, is the spatial location of the individual (Anderson and Jeffery, 2003; Barry et al., 2012; Leutgeb et al., 2004) and, linked to this, the speed of movement (Jeewajee et al., 2008; McFarland et al., 1975) and the direction of the head (Jercog et al., 2019; Yoganarasimha et al., 2006). In recent years, several researchers have attempted to decode this neural code, mainly using machine learning methods. In this research, three neural networks were used to attempt to achieve the same, which have not or rarely been used for this purpose before, and therefore there is not enough experience of the results they can achieve. These models were behaviour cloning, transfer learning and finally a hybrid CNN-BiLSTM. For comparability, the project used data from the CA1 area of the rat brain from the research published in Frey et al. (2021) and used error functions and a method to determine the error between predicted and ground truth results (Frey et al., 2021). A framework was also developed for the project that can handle electrophysiological data in multiple formats. The signals are processed by a preprocessing cascade system of filters and then feature extraction is performed using wavelet decomposition before being loaded into neural networks. Although our program was able to perform the preprocessing and learning process four times faster than the average available methods due to the high degree of optimization built-in, only a reduced number of experiments could be performed due to the size of the overall data set.

In the first half of the project, the impact of these filters on decoding performance was investigated on a reduced part of the original data set. Found that downsampling the raw signal significantly improved decoding accuracy and the moving window mean procedure helped to eliminate the phenomenon of overfitting. Meanwhile, the other three filters did not result in any significant change or even worsened the accuracy of the

predicted values. The reason for this in the case of the bandpass filter and the line noise filter may be that most of the devices used to record neurological signals have some degree of analogue signal filtering, which is precisely designed to eliminate noise from the environment. Finally, in the case of the detrend filter, a likely reason for the loss of efficiency is that the signal did not contain errors or shift trends caused by recording noise. Therefore, the filter was mostly used to change the relevant trends, which reduced the learning efficiency.

In the second half of the project, the models were trained on the full data set, thus testing how accurately the three networks can decode parameters related to the spatial self-location of the animal from the neural code. The best relative performance was achieved by the behaviour cloning model. This is a simpler neural network architecture compared to the other two models. However, the applied batch normalization and maxpooling layers can efficiently generalize from the data seen to scenarios not yet seen. Relatively poor prediction results compared to the hoped-for results were achieved with the hybrid CNN-BiLSTM network. This may be due to the fact that information extracted by a too shallow CNN layer was attempted to be generalized in time by an LSTM network that was too deep for the task. This was also evident from the fact that overfitting was regularly observed in this model. On the other hand, in the future, it would be worthwhile to apply some feature extraction on the input data, such as converting it into a spike count vector. In the case of transfer learning, no significant advantage could be observed compared to a non-trained network (such as the behaviour cloning model). This may be because these meshes were trained on RGB images of everyday scenes or objects, so the multichannel patterns found in the normalized wavelet transformed matrices may differ greatly and cannot be extracted as effectively. This problem could be solved by using a pre-trained network on a wavelet transformed database.

In terms of decoded values, the three neural networks tested were best at decoding the speed parameters from the electrophysiological signals. This accuracy was further enhanced when the behaviour cloning model was trained with a data set subjected to wavelet docking in the frequency range 2-250 Hz. All three models could only decode the head orientation and the spatial position of the animal with a rather high error rate. This may be due to the fact that the signal used for the delays did not sufficiently carry the patterns encoding the self-location. Another reason may be that the resolution of the wavelet transform used in the project masked or did not sufficiently represent the necessary patterns. It would be worth re-running the program on a server with a higher number of voices to find out. Another valuable line of research is the possibility of testing the scientific hypothesis that when animals move at near-zero speeds, the neural code actually encodes past (Davidson et al., 2009), or future (Pfeiffer and Foster, 2013) waypo-

ints with greater frequency. That is, whether the individual actually recalls or plans their path at these moments in time.

In summary, the project has been able to identify which pre-processing settings are most helpful for decoding the neural signals studied in the project. In addition, three novel neural network architectures were investigated, of which behaviour cloning was found to be particularly suitable for decoding animal velocity from neural code recorded from the CA1 region of the hippocampus. The code and data generated in this project, given the right hardware background and time, offer the potential for further research directions.

Bibliography

Anderson, M. I. and Jeffery, K. J. (2003). Heterogeneous Modulation of Place Cell Firing by Changes in Context. *J. Neurosci.*, 23(26):8827–8835. Publisher: Society for Neuroscience Section: Behavioral/Systems/Cognitive.

Barry, C., Ginzberg, L. L., O'Keefe, J., and Burgess, N. (2012). Grid cell firing patterns signal environmental novelty by expansion. *Proceedings of the National Academy of Sciences*, 109(43):17687–17692. Publisher: Proceedings of the National Academy of Sciences.

Blankertz, B., Curio, G., and Müller, K.-R. (2001). Classifying Single Trial EEG: Towards Brain Computer Interfacing. In *Advances in Neural Information Processing Systems*, volume 14. MIT Press.

Blankertz, B., Lemm, S., Treder, M., Haufe, S., and Müller, K.-R. (2011). Single-trial analysis and classification of ERP components — A tutorial. *NeuroImage*, 56(2):814–825.

Bostock, E., Muller, R. U., and Kubie, J. L. (1991). Experience-dependent modifications of hippocampal place cell firing. *Hippocampus*, 1(2):193–205. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/hipo.450010207.

Corrado, G. S., Sugrue, L. P., Seung, H. S., and Newsome, W. T. (2005). Linear-Nonlinear-Poisson Models of Primate Choice Dynamics. *Journal of the Experimental Analysis of Behavior*, 84(3):581–617. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1901/jeab.2005.23-05.

Daubechies, I. (1990). The wavelet transform, time-frequency localization and signal analysis. *IEEE Transactions on Information Theory*, 36:961–1005. ADS Bibcode: 1990ITIT...36..961D.

Davidson, T. J., Kloosterman, F., and Wilson, M. A. (2009). Hippocampal Replay of Extended Experience. *Neuron*, 63(4):497–507. Publisher: Elsevier.

Dubey, K., Agarwal, A., Lathe, A. S., Kumar, R., and Srivastava, V. (2019). Self-attention based BiLSTM-CNN classifier for the prediction of ischemic and non-ischemic cardiomyopathy.

Elman, J. L. (1990). Finding structure in time. *Cognitive Science*, 14(2):179–211.

Epsztein, J., Brecht, M., and Lee, A. K. (2011). Intracellular Determinants of Hippocampal CA1 Place and Silent Cell Activity in a Novel Environment. *Neuron*, 70(1):109–120. Publisher: Elsevier.

Frank, L. M., Brown, E. N., and Wilson, M. (2000). Trajectory encoding in the hippocampus and entorhinal cortex. *Neuron*, 27(1):169–178.

Frey, M., Tanni, S., Perrodin, C., O'Leary, A., Nau, M., Kelly, J., Banino, A., Bendor, D., Lefort, J., Doeller, C. F., and Barry, C. (2021). Interpreting wide-band neural activity using convolutional neural networks. *eLife*, 10:e66551. Publisher: eLife Sciences Publications, Ltd.

Graves, A., Jaitly, N., and Mohamed, A.-r. (2013). Hybrid speech recognition with Deep Bidirectional LSTM. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 273–278.

Hafting, T., Fyhn, M., Molden, S., Moser, M.-B., and Moser, E. I. (2005). Microstructure of a spatial map in the entorhinal cortex. *Nature*, 436(7052):801–806. Number: 7052 Publisher: Nature Publishing Group.

Hölscher, C., Jacob, W., and Mallot, H. A. (2004). Learned association of allocentric and egocentric information in the hippocampus. *Exp Brain Res*, 158(2):233–240.

Jeewajee, A., Barry, C., O'Keefe, J., and Burgess, N. (2008). Grid cells and theta as oscillatory interference: Electrophysiological data from freely moving rats. *Hippocampus*, 18(12):1175–1185. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/hipo.20510.

Jercog, P. E., Ahmadian, Y., Woodruff, C., Deb-Sen, R., Abbott, L. F., and Kandel, E. R. (2019). Heading direction with respect to a reference point modulates place-cell activity. *Nat Commun*, 10(1):2333. Number: 1 Publisher: Nature Publishing Group.

Jude, J. and Hennig, M. H. (2021). Hippocampal representations emerge when training recurrent neural networks on a memory dependent maze navigation task. arXiv:2012.01328 [q-bio].

Karlsson, M. P. and Frank, L. M. (2008). Network Dynamics Underlying the Formation of Sparse, Informative Representations in the Hippocampus. *J. Neurosci.*, 28(52):14271–14281. Publisher: Society for Neuroscience Section: Articles.

Kropff, E., Carmichael, J. E., Moser, M.-B., and Moser, E. I. (2015). Speed cells in the medial entorhinal cortex. *Nature*, 523(7561):419–424. Number: 7561 Publisher: Nature Publishing Group.

Leutgeb, S., Leutgeb, J. K., Treves, A., Moser, M.-B., and Moser, E. I. (2004). Distinct Ensemble Codes in Hippocampal Areas CA3 and CA1. *Science*, 305(5688):1295–1298. Publisher: American Association for the Advancement of Science.

McFarland, W. L., Teitelbaum, H., and Hedges, E. K. (1975). Relationship between hippocampal theta activity and running speed in the rat. *Journal of Comparative and Physiological Psychology*, 88:324–328. Place: US Publisher: American Psychological Association.

O'Keefe, J. and Nadel, L. (1978). *The Hippocampus as a Cognitive Map*. Oxford: Clarendon Press. Accepted: 2016-10-08T00:43:54Z.

Pfeiffer, B. E. and Foster, D. J. (2013). Hippocampal place-cell sequences depict future paths to remembered goals. *Nature*, 497(7447):74–79. Number: 7447 Publisher: Nature Publishing Group.

Pham, T. D. (2021). Time–frequency time–space LSTM for robust classification of physiological signals. *Sci Rep*, 11(1):6936. Number: 1 Publisher: Nature Publishing Group.

Pozar, D. M. (2011). *Microwave Engineering*. John Wiley & Sons. Google-Books-ID: _YEbGAXCcAMC.

Quirk, G. J., Muller, R. U., and Kubie, J. L. (1990). The firing of hippocampal place cells in the dark depends on the rat's recent experience. *J. Neurosci.*, 10(6):2008–2017. Publisher: Society for Neuroscience Section: Articles.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088):533–536. Number: 6088 Publisher: Nature Publishing Group.

Sammut, C. (2010). Behavioral Cloning. In Sammut, C. and Webb, G. I., editors, *Encyclopedia of Machine Learning*, pages 93–97. Springer US, Boston, MA.

Sargolini, F., Fyhn, M., Hafting, T., McNaughton, B. L., Witter, M. P., Moser, M.-B., and Moser, E. I. (2006). Conjunctive Representation of Position, Direction, and Velocity in Entorhinal Cortex. *Science*, 312(5774):758–762. Publisher: American Association for the Advancement of Science.

Shalev-Shwartz, S. and Ben-David, S. (2014). *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, Cambridge.

Tan, M. and Le, Q. V. (2020). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. Technical Report arXiv:1905.11946, arXiv. arXiv:1905.11946 [cs, stat] type: article.

Tanner, D., Morgan-Short, K., and Luck, S. J. (2015). How inappropriate high-pass filters can produce artifactual effects and incorrect conclusions in ERP studies of language and cognition. *Psychophysiology*, 52(8):997–1009. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/psyp.12437.

Torrence, C. and Compo, G. P. (1998). A Practical Guide to Wavelet Analysis. *Bull. Amer. Meteor. Soc.*, 79(1):61–78.

Towse, B. W., Barry, C., Bush, D., and Burgess, N. (2014). Optimal configurations of spatial scale for grid cell firing under noise and uncertainty. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 369(1635):20130290. Publisher: Royal Society.

Walker, K. M. M., Bizley, J. K., King, A. J., and Schnupp, J. W. H. (2011). Multiplexed and Robust Representations of Sound Features in Auditory Cortex. *J. Neurosci.*, 31(41):14565–14576. Publisher: Society for Neuroscience Section: Articles.

Wilson, M. A. and McNaughton, B. L. (1993). Dynamics of the Hippocampal Ensemble Code for Space. *Science*, 261(5124):1055–1058. Publisher: American Association for the Advancement of Science.

Wood, E. R., Dudchenko, P. A., Robitsek, R. J., and Eichenbaum, H. (2000). Hippocampal neurons encode information about different types of memory episodes occurring in the same location. *Neuron*, 27(3):623–633.

Xu, Z., Wu, W., Winter, S. S., Mehlman, M. L., Butler, W. N., Simmons, C. M., Harvey, R. E., Berkowitz, L. E., Chen, Y., Taube, J. S., Wilber, A. A., and Clark, B. J. (2019). A Comparison of Neural Decoding Methods and Population Coding Across Thalamo-Cortical Head Direction Cells. *Frontiers in Neural Circuits*, 13.

Yoganarasimha, D., Yu, X., and Knierim, J. J. (2006). Head Direction Cell Representations Maintain Internal Coherence during Conflicting Proximal and Distal Cue Rotations: Comparison with Hippocampal Place Cells. *J. Neurosci.*, 26(2):622–631. Publisher: Society for Neuroscience Section: Behavioral/Systems/Cognitive.

	Cheby band filter	Line noise filter	Down sampling	Detrend	Moving window mean	Feature extraction		Speed error			Head direction error			X coordinate error			Y coordinate error		
						Frequency band (Hz)	*Number of voices*	Median (cm/s)	Mean (cm/s)	Standard Deviation	Median (rad)	Mean (rad)	Standard Deviation	Median (cm)	Mean (cm)	Standard Deviation	Median (cm)	Mean (cm)	Standard Deviation
	Random guessing							*24.91*	*21.87*	*20.39*	*2.26*	*2.4*	*1.6*	*42.48*	*58.35*	*42*	*35.27*	*45.36*	*32.41*
CNN-BiLSTM			*Baseline*			**2-15,000**	**5**	**5.93**	**7.735**	**6.66**	**1.775**	**1.885**	**0.95**	**21.18**	**24.3**	**16.47**	**15.715**	**18.605**	**12.245**
	•					**2-7,500**	**5**	5.725	7.76	6.615	1.43	1.46	0.95	22.325	26.685	17.84	16.4	20.29	12.91
		•				**2-7,500**	**5**	9.385	10.315	8.47	2.025	2.175	0.97	22.075	23.68	14.735	16.49	17.61	11.045
			20,000Hz			**2-7,500**	**5**	3.99	5.09	4.33	2.265	2.135	0.975	21.79	22.29	14.2	45.19	48.88	19.485
			10,000Hz			**2-7,500**	**5**	1.91	2.685	2.35	1.32	1.435	0.995	22.71	28.89	21.555	17.445	22.41	16.15
				•		**2-7,500**	**5**	5.975	7.215	5.88	1.53	1.45	0.89	21.075	22.725	11.93	17.195	18.915	11.93
					•	**2-7,500**	**5**	6.645	7.8	5.955	0.775	1.015	0.775	20.545	22.33	15.62	17.745	18.37	20.49
Behaviour Cloning(BC)			*Baseline*			**2-15,000**	**5**	**5.81**	**6.47**	**4.975**	**0.75**	**0.93**	**0.65**	**24.27**	**28.305**	**19.36**	**47.735**	**43.635**	**20.91**
						2-7,500	**3**	4.395	5.61	5.09	0.805	0.86	0.6	15.75	20.715	14.6	49.25	48.23	21.12
						2-150	**5**	9.835	11.487	1.169	1.83	2.12	1.67	48.936	49.288	32.016	73.696	76.256	34.208
						250-7,500	**5**	6.335	7.215	5.14	0.97	1.3	0.95	15.21	18.42	18.06	33.66	33.66	15.715
	•					**2-7,500**	**5**	5.19	6.165	4.68	0.99	0.905	0.46	19.455	18.735	8.965	31.34	31.715	11.075
		•				**2-7,500**	**5**	7.21	7.615	4.725	1.02	1.205	0.945	23.905	24.135	15.315	36.195	35.065	14.565
			20,000Hz			**2-7,500**	**5**	2.86	3.88	3.585	0.84	0.985	0.705	20.8	22.14	17.565	33.055	34.095	17.45
			10,000Hz			**2-7,500**	**5**	1.845	2.135	1.625	0.77	0.875	0.595	21.795	22.275	13.525	36.505	34.505	14.18
				•		**2-7,500**	**5**	6.43	7.805	6.68	1.005	1.31	0.985	46.3	47.625	25.945	50.28	52.775	23.94
					•	**2-7,500**	**5**	8.86	10.14	6.85	0.805	1.005	0.835	22.575	29.8	20	49.21	45.34	19.455
Transfer Learning(TL)			*Baseline*			**2-15,000**	**5**	**6.895**	**8.525**	**7.22**	**0.85**	**0.915**	**0.65**	**21.685**	**25.26**	**41.325**	**18.57**	**19.215**	**12.915**
						2-7,500	**3**	7.04	9.67	14.235	1	0.99	0.5	23.2	28.025	42.07	18.32	21.18	28.89
						2-150	**5**	9.093	15.694	44.359	1.8	2.24	2.9	40.66	53.47	71.72	31.464	35.664	43.256
						250-7,500	**5**	6.73	7.77	5.705	0.99	0.925	0.495	19.905	23.495	15.76	18.735	18.045	9.55
	•					**2-7,500**	**5**	5.835	7.005	5.81	0.985	0.93	0.51	20.765	23.055	16.76	32.715	31.99	15.57
		•				**2-7,500**	**5**	10.575	15.68	17.835	0.99	1.005	0.77	30.04	39.09	34.785	22.365	27.18	29.95
			20,000Hz			**2-7,500**	**5**	3.745	4.495	3.45	0.975	0.94	0.465	22.985	24.425	15.79	16.985	17.545	10.17
			10,000Hz			**2-7,500**	**5**	2.035	2.66	2.565	1.04	0.995	0.59	21.635	23.85	21.305	18.59	20.505	13.905
				•		**2-7,500**	**5**	6.29	7.775	7.56	0.86	0.875	0.485	22.685	22.685	18.8	17.52	18.55	11.655
					•	**2-7,500**	**5**	6.505	7.45	5.935	0.875	0.895	0.525	22.6	24.885	19.635	17.245	20.485	18.135

Figure 4.1 The effects of individual filters in the form of differences between decoded data and ground truth data are shown in the results section. The filters were applied one by one to the raw signals until no other filter was active. The effects of different settings of feature extraction on the data were also investigated. The random guessing data shows the accuracy achieved by a perfectly random guess.