Brain Machine Interfaces Report

Manuela Giraud[1], Sergio Gosalvez[1], Cosima Graef[1], Leopold Hebert Stevens [1]
[1] Dept. of Bioengineering, Imperial College London, London, UK
e-mail: manuela.giraud18@ic.ac.uk, sergio.gosalvez18@ic.ac.uk, cosima.graef18@ic.ac.uk, leopold.hebert-stevens18@ic.ac.uk

***Abstract* –**

Neural decoding requires high computational power and specialised algorithms to efficiently translate thousands of neuronal spike trains into an external real-world variable real-world such as predicting movement. Whilst numerous algorithms have been developed to decode neural data, many still use traditional methods or the performance is not acceptable for clinical implementation. To analyse these different techniques, we designed and tested multiple data pre-processing approaches, angle detection classifiers, and trajectory decoding algorithms. The results were compared based on their accuracy, time response, and RMSE from the true trajectory. From a split ratio of 70% for training and 30% for testing, both peri-stimulus-time histogram (PSTH) and Gaussian Kernel were used for data pre-processing. Angle classification was determined using mean trajectory, k-Nearest-Neighbour (kNN), and support vector machines (SVM). Finally, the trajectory was decoded with Kalman and linear filters, mean and nearest trajectory, and a KNN algorithm. Overall, results show that combining a support vector machine classifier and using the mean trajectory was the best strategy, with a root mean squared error score of 12.1311 and a 97% classification accuracy.

***Keywords* – brain-machine interface, electrophysiology, goal-directed behaviour, monkey, neural decoding, neuroprosthetics.**

I. INTRODUCTION

Neural decoding, the act of accurately retrieving physiological information from the activity of neurons, represents a major challenge in brain-machine interfaces. Neuron spike trains, specifically of motor cortical neurons, have been shown to have the ability to encode both direction and velocity of arm movements [1][2]. The information gained provides a solid foundation for neural prosthetic devices, which can partially restore motor function for patients suffering diseases like spinal cord injury or cerebral palsy [3]. In addressing motor paralysis, these brain-machine interfaces are able to approximate the arm displacement performance of normal subjects [4], and in such, have the potential to drastically improve patient welfare and quality of life [3]. However, although there have been impressive technical advances in recent years in utilising the brain activity of patients to control neural prostheses [5], the performance is still not at a standard acceptable for clinical implementation [6].

Neural activity in the phase before the onset of movement has been extensively studied, and has shown that an initial movement plan, which comprises of a combination of cognitive processes and sensory information is created before movement onset [7]. In this initial phase, neuron activity is increased in the frontal and parietal cortical areas, particularly the pre-motor and primary motor cortices [7], and thus justifies most existing studies focusing on these brain regions for decoding reaching trajectories [8][9].

To investigate the significance of motor planning and execution in a reaching task and the tuning properties of neurons throughout, a commonly used experimental paradigm (adapted from [10]) featuring a monkey executing movement was implemented. The data is composed of spike activity of 98 neurons, which were recorded while the monkey moved its hand in 8 different directions at the onset of a cue. This was repeated for 100 trials for each direction. The corresponding hand positions in the x,y and z directions were also recorded. The main aim of this paradigm was to design a simple and efficient algorithm for using the recorded neuron activity to 1) classify which of the 8 target directions the monkey was reaching for; 2) predict the precise trajectory of the monkey's hand in terms of x and y coordinates as it reaches for the target and 3) evaluate the performance of the algorithm in decoding the neural spikes, using key metrics of RMSE and execution time, compared to existing decoding algorithms to determine the optimal one for the specific task at hand.

II. METHODS

A. Data Pre-Processing

Neuronal activity is highly variable. Even when experimental settings are maintained, the same neuron may produce diverse spike trains across trials. For this reason, neural decoding tasks can benefit from data pre-processing. The most common historical approach to find an accurate estimate of the firing rate has been to add spikes from multiple trials in a time-binned histogram known as a peri-stimulus-time histogram (PSTH). These histograms can be smoothed to get a continuous estimate of the firing rate, which is often of importance in single trial situations.

However, to get a continuous firing rate estimate researchers generally employ Kernel Smoothing. This is achieved through the convolution of the spike train with a kernel of a particular shape. An estimate is generated where the firing rate at any time is a weighted average of the nearby spikes. A Gaussian shaped kernel is most often used to smooth the spike data to a firing rate that is higher in regions of spikes and lower otherwise. Kernel smoothing's most evident benefit is its simplicity and speed.

For this study, both methods of pre-processing are considered. The Gaussian Kernel implemented with 100ms standard deviation and 500ms window size (`dataAV`) outperforms PSTH estimates, reducing the RMSE by 10% in trajectory decoding tasks. Nevertheless, for the task of angle classification a simple average over time and trials for each neuron (`dataAVAV`) gives the best results. (See Table I)

Finally, to avoid silent neurons affecting the performance of neural decoding algorithms, some neurons are removed based on their average firing rates (`badNeurons`). Only using neurons with activity on every angle results in a reduction of the RMSE by 3%.

B. Data Splitting Strategies

The selection of a data splitting strategy can have a significant impact on evaluating a model's performance and comparison to other models. The most commonly used method, is to split the data into training and testing sets [11], where the model is trained on the training set only, and performance evaluated on how well the model can generalise to the unseen testing set. By not using the entire dataset, overfitting can be prevented. The split ratio was chosen to be 70% and 30% for the training and test set respectively, with samples randomly selected across the entire dataset. This split was justified as according to [11], the data splitting strategy is highly dependent on the dataset, but a balance of 50-70% between training and testing is suitable for achieving a robust model.

C. Classification Algorithms for Angle Detection

In order to classify the correct reaching angle of the monkey, three classifiers were implemented, which include comparison to mean trajectory, k-Nearest-Neighbour (kNN) and support vector machines (SVM).

Mean Trajectory: For its low computational cost and simplicity, comparing the test vector to each of the mean vectors for each direction was tested. Each mean vector was created by transforming the training data into 8, 98 element vectors (one for each direction), by averaging the neuron spikes across time and trials for each angle. The test and mean vectors were compared using mean squared error (MSE), see equation 1, and mean absolute error (MAE), see equation 2, where n is the number of data points, y_i are the observed values and $\hat{y}_i$ the predicted values. The direction with lowest error was classified as the movement direction.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{1}$$

$$MAE = \frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|}{n} \tag{2}$$

K-Nearest-Neighbour: KNN based classification was implemented with MATLAB function `fitcknn` with 28 as the number of nearest neighbours. The direction chosen was based on the distance from the test data vector and its k-nearest-neighbours from the training set. To reduce the computation time, the data dimensionality was reduced by averaging across trials for all 8 directions. Since the directions are well separated, a small value of k is necessary to compute the right angle. The highest accuracy was obtained with k=1.

Support Vector Machines: The use of SVMs, a supervised learning method, in neural decoding has been demonstrated by [12], [13] and [14], and is a relatively effective classification approach for datasets with clear class separability. SVMs are used for binary classification, but can be extended to multi-class classification by combining binary classifiers [15]. The MATLAB function `fitcecoc` was used to build a multi-class classification model with $\frac{K(K-1)}{2} = 28$, binary SVM models, where K is the number of classes, using the one-versus-one coding design. After building the model with `fitcecoc` and the training data, MATLAB function `predict` was used to predict the angle for the test data.

D. Trajectory Decoding

The x and y coordinates of the monkey's hand position were estimated using one of the following methods: taking the mean trajectory of the the training data for each direction, Kalman and linear filters, KNN algorithm, or choosing the nearest trajectory from the training data's trials, within a chosen angle direction.

Mean Trajectories: To calculate the mean trajectories, the hand positions for all training trials were averaged for each of the 8 angles. The mean trajectory corresponding to the chosen angle from the classification algorithm was then used to predict the entire trajectory.

To improve this approach and account for position variability across trials, the initial x and y position of the test data was compared to every trial for the corresponding angle using MAE. The trial that matched the initial position the closest was then used to predict the trajectory across time. This comparison was only done for the initial x and y position, rather than iteratively over time, to ensure the position did not jump between trials.

Linear filtering: A linear filter, as the name indicates, assumes a linear relationship between the spikes' firing rate and the hand position. It takes the general form:

$$x_k = \mathbf{f}_x^T \mathbf{r}_k + b_x$$

$$y_k = \mathbf{f}_y^T \mathbf{r}_k + b_y$$

Where $\mathbf{r}_k$ is the vector containing the firing rates for all time steps k.

Such a model is widely used in the context of neural decoding as it has the potential to give accurate predictions [16]. The training data was used to find the vectors $\mathbf{f}$ and coefficients b, using the `MATLAB` function `fitlm`. To simplify computation, only the neuron with maximum activity for each predicted reaching direction was selected for model training and testing.

Kalman filtering: Kalman Filters have been successfully used as BMI decoding algorithms and claim to have some advantages over linear filtering, especially when treating with rapid motions. In this study, a Kalman filter is implemented based on the work from W. Wu [17].

The Kalman filter assumes that the arm state (x, y, v_x, v_y) evolves as a linear dynamical system: the arm state at discrete time t is a linear transformation of the arm state at time $t-1$, plus Gaussian noise. It also supposes that arm state and neural activity at time t have a linear relation. Based on these assumptions, the Kalman Filter infers the arm state only based on neural data observations. It iteratively progresses through time, updating its estimates of arm state and error covariance.

The physical relationship between neural firing and arm movement means that there is a delay between them. Hence, an optimum lag can be found experimentally (40ms), which improves the model and accuracy of decoding.

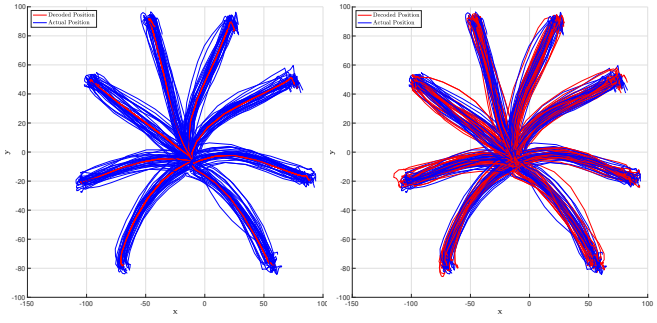
III. RESULTS

TABLE I
Model Performance: Direction

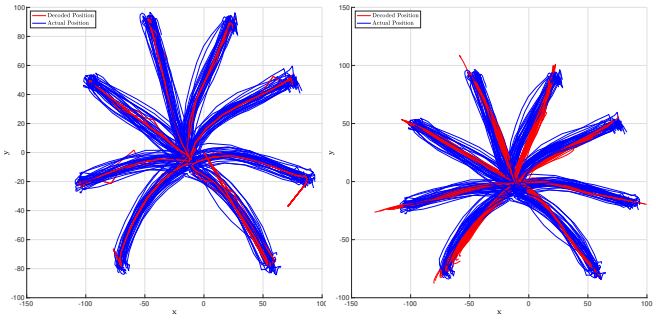
Direction classifiers	Accuracy [%]
MSE (`dataAVAV`)	94.58
MSE (`dataAV`)	14.58
MAE (`dataAVAV`)	96.25
MAE (`dataAV`)	17.50
SVM	97.08
kNN	94.58

TABLE II
Model Performance: Trajectory

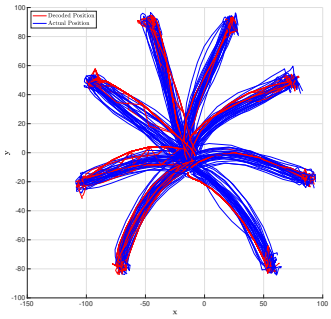
Trajectory decoders	RMSE	Time (s)
Mean Trajectory	12.1311	10.25
Nearest Trial	13.6193	13.48
kNN	13.1838	7.82
Linear Filter	12.6455	-
Kalman Filter	15. 5915	7.92



(a) Mean (b) Nearest



(c) Linear (d) Kalman



(e) kNN

IV. DISCUSSION

The accuracy in classifying the monkey's reaching angle for the supervised learning classifiers outlined in the methods section, can be seen in table I. Whilst MSE (dataAVAV) and kNN have the same accuracy of 94.58%, MAE (dataAVAV) and SVM outperform these classifiers with accuracies of 96.25% and 97.08% respectively. SVM providing the highest classification accuracy indicates that although it can only identify a few types of patterns compared to kNN, which does not predetermine the boundary between classes, it has more stable predictions. KNN has a tendency to overfit training data for small values of k and is more computationally demanding, whilst SVM has high accuracy when the classes have a distinct linear separation. The MAE approach having higher accuracy than MSE is explained by squaring the error giving more weight to larger errors and thus skewing the estimate of error towards outliers. Since SVM performed the best classification it was used for all trajectory decoders in table II.

From the trajectory prediction results in table II, the mean method gives the least error (12.1311) with a fast decoding time of 10.25s. For the proposed task, which was to estimate with least error the positions of the monkey's hand from the neural firing spikes, this method therefore provides the best performance. However, it does not provide any insight into the relationship between firing rate and movement direction or speed, and simply relies on statistical analysis. Taking the mean of trajectories only gives a robust result if there is little variance between trials, and is not adaptable to predictions where test trials differ greatly from training trials. In this context it performs well because variance across trials is low.

The nearest trial method has the advantage over the

previous method, by providing more adaptive predictions. The RMSE is higher (13.6193) than the mean method however, this method relies heavily on the quality of training data, similar to the mean method, cannot accurately predict trials that deviate from the training set.

The kNN method was also used for trajectory decoding and achieved an RMSE of 13.1838. Similar to the classification, the kNN trajectory decoding model uses the trajectory and position of the closest neighbour trajectory. However, a key weakness of this algorithm is that the trajectory shifts for different timesteps. As the training set size increases, the closest neighbours can vary, and the decoded hand trajectory can shift to a different path, creating an irregular looking curve.

Linear filtering proves to have the second best prediction with an RMSE of 12.6455. It can be observed in figure 1(c), that the predictions describe a trajectory very similar to the mean trajectory in figure 1. Even if the model gives different predictions at every trial, the variance between the predictions remains low compared to the actual variance between trials. We can conjecture that variance of hand position between trials is the result of non-deterministic processes that is not encompassed within a linear filter. To respond to this issue, Kalman modelling includes Gaussian noise to account for non-linear random processes. The resulting RMSE is slightly higher than a linear filter, 15.5915, due to overfitting of the noise.

Although the RMSE values obtained for the various methods are quite similar, how appropriate each method is depends on the intention of the decoding result. It can be seen that taking the mean of the training data can provide substantial information on the general direction to be decoded, and has the advantage of being fast. If however, the aim is to precisely differentiate between similar trajectories, another method that includes non-linear predictions will be necessary, such as KNN or Kalman.

V. CONCLUSIONS

By applying various decoding models to the neural data of a monkey conducting a reaching task in 8 distinct angles, it was found that the combination of a support vector machine classifier and using the mean trajectory for trajectory decoding produced the lowest RMSE. On the other hand, using a Kalman Filter for the trajectory decoding resulted in the fastest decoding time, although at the cost of highest RMSE. Important to note is that the performance of the decoders tested in this paper depends highly on the variance of the training data and so algorithms that are able to generalise better to new data might be beneficial. Future studies could include principal component analysis as an alternative dimensionality reduction whilst removing noise components. Random forests, neural networks and non-linear regression models might also offer more insight into the relationship between neural activity and arm movement.

REFERENCES

[1] Schwartz, A. B. (7 1993). Motor cortical activity during drawing movements: population representation during sinusoid tracing. Journal of Neurophysiology, 70, 28–36. doi:10.1152/jn.1993.70.1.28

[2] Moran, D. W., & Schwartz, A. B. (11 1999). Motor Cortical Representation of Speed and Direction During Reaching. Journal of Neurophysiology, 82, 2676–2692. doi:10.1152/jn.1999.82.5.2676

[3] Shih, J. J., Krusienski, D. J., & Wolpaw, J. R. (3 2012). Brain-Computer Interfaces in Medicine. Mayo Clinic Proceedings, 87, 268–279. doi:10.1016/j.mayocp.2011.12.008

[4] Wodlinger, B., Downey, J. E., Tyler-Kabara, E. C., Schwartz, A. B., Boninger, M. L., & Collinger, J. L. (2 2015). Ten-dimensional anthropomorphic arm control in a human brainmachine interface: difficulties, solutions, and limitations. Journal of Neural Engineering, 12, 016011. doi:10.1088/1741-2560/12/1/016011

[5] Brandman, D. M., Cash, S. S., & Hochberg, L. R. (10 2017). Review: Human Intracortical Recording and Neural Decoding for Brain–Computer Interfaces. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 25, 1687–1696. doi:10.1109/TNSRE.2017.2677443

[6] Cui, H. (10 2016). Forward Prediction in the Posterior Parietal Cortex and Dynamic Brain-Machine Interface. Frontiers in Integrative Neuroscience, 10. doi:10.3389/fnint.2016.00035

[7] Kalaska, J. F., & Crammond, D. J. (3 1992). Cerebral Cortical Mechanisms of Reaching Movements. Science, 255, 1517–1523. doi:10.1126/science.1549781

[8] Velliste, M., Perel, S., Spalding, M. C., Whitford, A. S., & Schwartz, A. B. (6 2008). Cortical control of a prosthetic arm for self-feeding. Nature, 453, 1098–1101. doi:10.1038/nature06996

[9] Wessberg, J., Stambaugh, C. R., Kralik, J. D., Beck, P. D., Laubach, M., Chapin, J. K., … Nicolelis, M. A. L. (11 2000). Real-time prediction of hand trajectory by ensembles of cortical neurons in primates. Nature, 408, 361–365. doi:10.1038/35042582

[10] Georgopoulos, A. P., Kalaska, J. F., Caminiti, R., & Massey, J. T. (11 1982). On the relations between the direction of two-dimensional arm movements and cell discharge in primate motor cortex. The Journal of Neuroscience, 2, 1527–1537. doi:10.1523/JNEUROSCI.02-11-01527.1982

[11] Xu, Y., & Goodacre, R. (7 2018). On Splitting Training and Validation Set: A Comparative Study of Cross-Validation, Bootstrap and Systematic Sampling for Estimating the Generalization Performance of Supervised Learning. Journal of Analysis and Testing, 2, 249–262. doi:10.1007/s41664-018-0068-2

[12] Wang, D., Zhang, Q., Li, Y., Wang, Y., Zhu, J., Zhang, S., & Zheng, X. (6 2014). Long-term decoding stability of local field potentials from silicon arrays in primate motor cortex during a 2D center out task. Journal of Neural Engineering, 11, 036009. doi:10.1088/1741-2560/11/3/036009

[13] Cao, Y., Hao, Y., Liao, Y., Xu, K., Wang, Y., Zhang, S., ... Zheng, X. (2013). Information Analysis on Neural Tuning in Dorsal Premotor Cortex for Reaching and Grasping. Computational and Mathematical Methods in Medicine, 2013, 1–9. doi:10.1155/2013/730374

[14] Xu, K., Wang, Y., Wang, Y., Wang, F., Hao, Y., Zhang, S., ... Zheng, X. (4 2013). Local-learning-based neuron selection for grasping gesture prediction in motor brain machine interfaces. Journal of Neural Engineering, 10, 026008. doi:10.1088/1741-2560/10/2/026008

[15] Clopath, C. (2022, February 15). Brain Machine Interfaces [Lecture recording]. Blackboard@Imperial College London. https://bb.imperial.ac.uk/

[16] Li, Z. (7 2014). Decoding methods for neural prostheses: where have we reached? Frontiers in Systems Neuroscience, 8. doi:10.3389/fnsys.2014.00129

[17] Wu, W., Black, M.J., Gao, Y., Bienenstock, L.E., Serruya, M.D., & Donoghue, J.P. (2002). Inferring Hand Motion from Multi-Cell Recordings in Motor Cortex using a Kalman Filter.