Neural Decoding of Arm Trajectories using LDA-based Classification and Regression

Nicolas Dehandschoewercker njd20@ic.ac.uk
Sonya Kalsi skk20@ic.ac.uk
Matthieu Pallud mrp220@ic.ac.uk
Pranathi Poojary psp20@ic.ac.uk
Department of Bioengineering (Imperial College London)

***Abstract*—This paper presents a causal neural decoding pipeline for brain-machine interfaces that predicts hand trajectories from primate neural spike data. Our two-stage model approach combines Linear Discriminant Analysis classification with direction-specific linear regression, achieving an RMSE of 9.85 centimetres and a classification accuracy of 98.39 percent. Dimensionality reduction through PCA and direction-specific regression significantly improved performance. The pipeline effectively captures directional tuning in neural activity, with rare misclassifications occurring only between adjacent movement angles. Based on our findings, we identify potential improvements such as stage-specific classifiers and velocity-based decoding. Our analysis demonstrates the effectiveness of traditional machine learning techniques for practical neural decoding applications.**

I. Introduction

Brain-Machine Interfaces (BMI) have emerged as a transformative technology enabling the brain to bypass conventional neuromuscular pathways and directly interact with external devices [1]. BMIs can be used to control devices such as computer cursors and prosthetic limbs. These systems use neural decoding which aims to translate brain signals to motor intent [2]. Decoding processes typically consist of a) signal processing such as Laplacian filtering and data normalisation, b) feature extraction through short-time Fourier Transforms, or Independent or Principal Component Analysis, and c) prediction [3]. The essence of neural decoding prediction lies in the classification/regression problem, for which there exist many machine learning tools. Although modern approaches such as neural networks and ensemble methods are gaining traction, traditional techniques such as linear regression remain widely used and are a key part of our approach [2].

This paper discusses the implementation and evaluation of a causal decoding pipeline using neural data supplied by Prof. Krishna Shenoy at Stanford University. All code was developed in MATLAB and is publicly available at https://github.com/pran872/BMI_MOM.

II. Data

The dataset contains spike trains recorded from a monkey performing reaching movements across 100 trials in each of the 8 directions: 30°, 70°, 110°, 150°, 190°, 230°, 310°, and 350°. Each trial includes simultaneous recordings from 98 neural units and corresponding hand positions in three-dimensional space.

III. Method

Our decoder follows a two-stage pipeline: first classifying the intended movement direction, then estimating hand position using a direction-specific linear regressor, one for each reaching angle.

A. Preprocessing

The raw spike trains were segmented into non-overlapping 20 ms bins. Firing rates were computed per bin, vectorised, and normalised to zero mean and unit variance. Features with variance below 10^{-6} were discarded as uninformative.

Principal Component Analysis (PCA) was performed as a dimensionality reduction step. For the classification stage, components preserving 95% of the variance across all trials were retained. For the regression stage, PCA was applied independently for each reaching angle with a 60% variance retention threshold to reduce overfitting. These values were chosen based on hyperparameter sweeps.

B. Classification Stage

We used Linear Discriminant Analysis (LDA) to assign trials to one of $K = 8$ reaching angles with each angle treated as a distinct class. Each trial's PCA-reduced feature vector $\mathbf{x} \in \mathbb{R}^d$ was projected into LDA space:

$$\mathbf{z} = \mathbf{x}^\top \mathbf{W}_c \tag{1}$$

where $\mathbf{W}_c \in \mathbb{R}^{d \times m}$ is the LDA projection matrix obtained during training. This matrix, $\mathbf{W}_c$ is computed from the training data by:

$$\mathbf{S}_B \mathbf{v} = \lambda \mathbf{S}_W \mathbf{v} \tag{2}$$

where $\mathbf{S}_B$ and $\mathbf{S}_W$ are the between-class and within-class scatter matrices, respectively, and λ and $\mathbf{v}$ are the corresponding eigenvalues and eigenvectors. The top eigenvectors form the columns of $\mathbf{W}_c$ to maximise the class separability in the projected space. During inference, the predicted class, $\hat{y}_c$ is:

$$\hat{y}_c = \arg\min_k \ \|\mathbf{z} - \boldsymbol{\mu}'_k\|_2^2 \tag{3}$$

where $\boldsymbol{\mu}'_k$ is the mean of class k in LDA space. This corresponds to assigning $\mathbf{x}$ to the class with the closest projected mean under the Mahalanobis distance.

C. Direction-Specific Regression Stage

For each of the 8 classified directions we train a linear regression model on windows of spike activity. Given a firing rate vector $\mathbf{x} \in \mathbb{R}^d$ and corresponding hand position $\mathbf{y} \in \mathbb{R}^2$, the weights $\mathbf{W}_r$ are learned via ordinary least squares:

$$\mathbf{W}_r = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{Y} \tag{4}$$

Each regressor includes a bias term; thus, input features at test time are extended as $\tilde{\mathbf{x}} = [\mathbf{x}; 1]$. The predicted position, $\hat{\mathbf{y}}$ is computed as:

$$\hat{\mathbf{y}} = \tilde{\mathbf{x}}^\top \mathbf{W}_r \tag{5}$$

To reduce overshooting at the end of trajectories, a post-processing step pushes predictions towards the average final hand position, i.e., a centroid (C_x^k, C_y^k) for each direction k. If a prediction is within radius, r of its target, we apply:

$$\hat{x} = C_x^k + \alpha \Delta x + \beta \cdot \text{sign}(\Delta x) \cdot \min(|\Delta x|, r), \quad \Delta x = \hat{x}_{\text{raw}} - C_x^k$$

and similarly for $\hat{y}$. Constants α, β, and r control convergence rate and damping strength.

D. Alternative Models

We tested other models such as K-Nearest Neighbors (KNN), Logistic Regression, Support Vector Regression (SVR), and Random Forests (RF) within our pipeline.

E. Training and Evaluation

The dataset was randomly partitioned into training, validation, and testing subsets (80:10:10). Hyperparameter optimisation was implemented using k-fold cross-validation and grid search (Tables I and II). During testing, spike data was fed causally from 320 ms onwards in 20 ms intervals to simulate real-time, causal decoding conditions. Performance was evaluated using total runtime for both stages, classification accuracy for the classifier, and RMSE for the regressor.

IV. Results

A. Best Model

Our best performing model achieved an RMSE of 9.85 cm on the test dataset, with a near-perfect classification accuracy of 98.39% and a runtime of 16.58 s. Fig. 1a shows the decoded hand trajectories closely following the actual movements across all directions, demonstrating the model's ability to generalise over time and across trials. Moreover, PCA significantly improved classification accuracy and RMSE for both the classifier and regressor (see Tables I and II).

B. Classifier Performance

The classifier achieved its best performance of 98.39% accuracy and runtime of 16.58 s using a 300 ms window, 20 ms bins, and a PCA threshold of 95% (Table I).

C. Regressor Performance

The optimal configuration—300 ms window, 20 ms bins, 60% PCA threshold, and centroid correction—provided the best trade-off between RMSE and runtime. Evaluation of individual regressors in Fig. 2a showed that the 8 regressors perform similarly with RMSE ranging from 6 to 9 cm, provided classification is correct.

Hyperparameter	Value	Classification Accuracy (%)	RMSE (cm)	Runtime (s)
Window size (ms)	100	74.92	33.33	31.74
	300	**98.39**	**9.85**	**16.58**
	400	97.47	10.76	34.08
Bin size (ms)	10	95.10	13.48	29.96
	20	**98.39**	**9.85**	**16.58**
	30	98.31	9.86	26.75
	40	95.26	13.33	25.72
PCA variance thresholds (%)	90	96.91	12.03	32.27
	95	**98.39**	**9.85**	**16.58**
	99	97.07	11.54	33.37
	No PCA	75.20	32.00	18.53

TABLE I: Effect of classifier hyperparameters on performance. Runtimes are averaged over 10 runs. A PCA threshold of 95% improved classification accuracy by 23.19% compared to using no PCA.

Hyperparameter	Value	RMSE (cm)	Runtime (s)
Window size (ms)	100	11.95	23.14
	300	**9.85**	**16.58**
	400	21.95	22.40
Bin size (ms)	10	**9.84**	254.86
	20	9.85	**16.58**
	30	11.37	19.89
	40	11.55	36.68
PCA variance thresholds (%)	50	9.88	28.45
	60	**9.85**	**16.58**
	95	10.18	29.47
	99	10.61	26.99
	No PCA	31.44	16.88
Centroid correction	Applied	**9.85**	**16.58**
	Not applied	10.46	17.54
Classifier accuracy (%)	100	**6.91**	**25.89**

TABLE II: Effect of regressor hyperparameters on performance. PCA and window size selection are crucial to improve RMSE.

D. Comparison with Other Models

As shown in Table III, LDA outperformed both KNN and logistic regression classifiers, with KNN having the lowest accuracy. Although RF regression achieved a slightly lower RMSE than linear regression, it had a higher runtime.

Classification Model	Accuracy	Regression Model	RMSE (cm)	Runtime (s)
LDA	98.39%	Linear Regression	9.85	**16.58**
		SVR	9.90	105.42
		KNN (k=5)	13.90	32.64
		Random Forest (n=50)	**9.84**	95.45
KNN (k=10)	83.08%	Linear Regression	**23.09**	**20.30**
		SVR	23.12	40.91
Logistic Regression	89.43%	Linear Regression	**18.85**	**17.78**
		SVR	18.88	42.08

TABLE III: Model comparison of two-stage pipeline combinations. LDA classification and linear regression performed best. Note that these results reflect our implementations; performance may differ with other implementations or further optimisation.

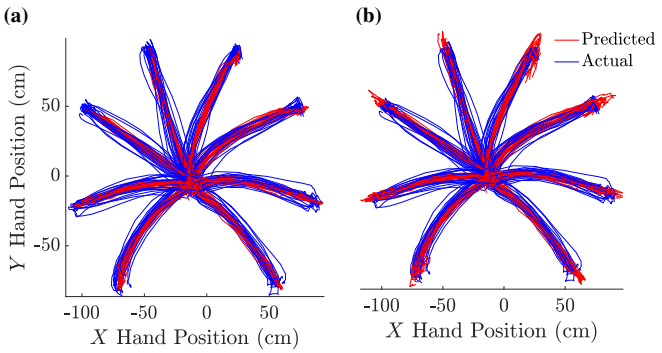

Fig. 1: Predicted and actual hand trajectories. **(a)** Best-performing model using our LDA classifier, linear regressor, and centroid correction. Predicted paths closely align with actual trajectories, achieving an RMSE of 9.85 cm. **(b)** Hand trajectories with no centroid correction overshoot with an RMSE of 10.46 cm.

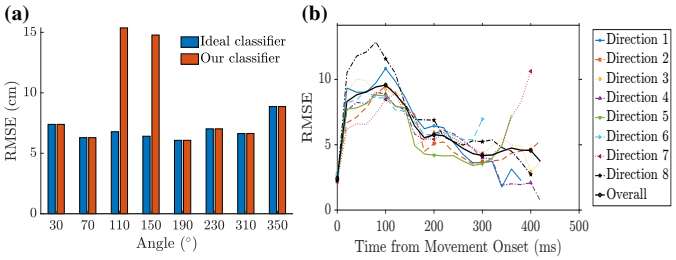

Fig. 2: Comparison of each direction-specific regressor. **(a)** Ideal classifier uses ground truth labels with 100% accuracy versus our classifier with a 98.39% accuracy. Regressors for angles 110° and 150° performed poorly with our classifier due to classification errors. **(b)** Mean RMSE across all correctly classified trials over time from movement onset. Prediction error peaks shortly after movement begins and then stabilises.

V. Discussion

In the following discussion, we examine the contributions and behaviours of each model component, critically evaluate their limitations, and propose avenues for future improvement.

A. Classifier Behaviour

The classifier's performance was critical in the decoding pipeline as classification errors propagate downstream. With perfect classification, RMSE significantly improves from 9.85 cm with our best-performing classifier to 6.91 cm with an ideal classifier. This suggests that the classifier is a bottleneck in our pipeline.

Ablation results in Table I show that dimensionality reduction through PCA significantly improved classification performance. This is likely because task-relevant neural activity lies in a low-dimensional space, and PCA helps extract these informative components by denoising irrelevant variability. LDA then strengthens class separation by maximising between-class variance, as seen in Fig. 3 [4].

The near-perfect accuracy of the classifier is shown by the confusion matrix in Fig. 4a. The few misclassifications that occurred were primarily between adjacent angles, near the boundaries of their respective clusters in the LDA latent space, suggesting strong directional sensitivity (Fig. 4b). In fact, only two trials showed any misclassification and in both those cases the classification error occurred early in the trajectory and persisted throughout, implying that once an incorrect direction is predicted, the classifier maintains that error for the remainder of the trial.

This behaviour likely stems from a training-testing mismatch: the classifier was trained on full trials, where each trial was treated as a single static feature vector. However, during inference, classification was performed causally, with features derived from shorter, incrementally accumulated bins. Further inspection of the LDA classifier's output scores over time showed that these scores remained largely unchanged as more spike data was added within a trial, reinforcing the idea that the classifier relies heavily on early neural activity. This early commitment may explain the persistence of classification errors once they occur and highlights the importance of representative training pipelines. Preliminary attempts to correct this mismatch by training causally led to unstable predictions, with the model frequently changing its predicted direction within a single trial.

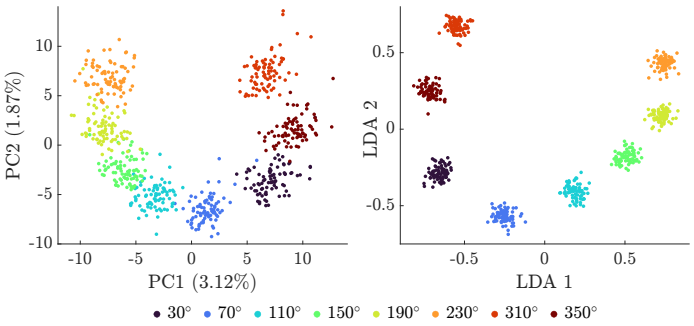

Fig. 3: PCA and LDA projections of classifier features on training data. Each dot represents one trial in one direction. *(Left)* PCA showing outlines of directional clusters in the neural activity, with some overlap. The first and second principal components capture approximately 5% of the explained variance. *(Right)* LDA on PCA-reduced features maximises inter-class variance.

B. Regressor Behaviour

The regressors demonstrated strong performance overall, with low and consistent RMSE across directions, provided the classifier correctly identified the movement direction. Fig. 2a shows that for angles 110° and 150°, the RMSE was significantly higher due to classification errors, emphasising the downstream impact of misclassifications (Fig. 4). These results reinforce the importance of classifier accuracy in maintaining robust trajectory decoding.

Within individual trials, the RMSE varies greatly over time, with a sharp increase at the onset of movement (Fig. 2b). This is true for all directions, even with correct classification, suggesting a flaw in the regressor due to the difficulty of decoding during the rapid transition from rest to movement.

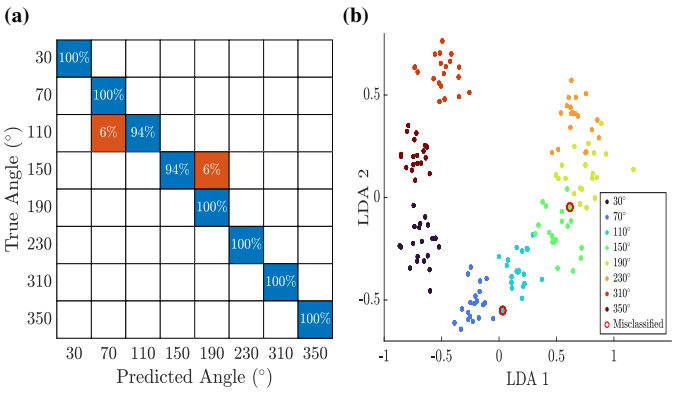

Fig. 4: Classifier performance. **(a)** Confusion matrix showing high classification accuracy (blue) with occasional misclassifications (orange) between adjacent angles. **(b)** LDA projection of test features shows distinct clusters with their true label. Each point represents a test trial in one direction. Misclassified points occurred only at the boundaries of clusters

Interestingly, without centroid correction, we observed that the decoded hand trajectories consistently overshot the true trajectory throughout the trial as shown in Fig. 1b. The predicted positions tended to move faster and further than the actual hand, likely due to the regressor implicitly modelling displacement without accounting for velocity or temporal constraints. Without a mechanism to adapt the trajectory speed or halt movement, the model continued to predict forward motion past the endpoint. Centroid correction reduced final-position overshooting and improved RMSE (Fig. 1a, Table II).

C. Comparison with other models

Among the models tested in Table III, LDA combined with linear regression offered the most effective trade-off between RMSE and runtime. Our KNN classifier had the lowest classification accuracy at 83.08%. However, prior work such as [5] has shown that KNN can perform well in neural decoding tasks, suggesting that its accuracy could be improved with more extensive hyperparameter tuning or improved feature selection. For regression, SVR and RF achieved comparable RMSEs but consistently had higher runtimes than linear regression.

D. Limitations and future directions

1) Stage-specific causal classification: To mitigate downstream errors from early misclassifications, future work could further explore causal training and stage-specific classifiers, with different models handling different time windows within a trial. This could enable the classifier to adapt to changes in neural activity, improving classification accuracy [6].

2) Velocity-based modelling: Although centroid correction mitigated endpoint overshooting, it did not prevent early overshoot (Fig. 2b). This form of regularisation also limits the model's applicability to the *general* hand position estimation task. Incorporating velocity-based modelling may mitigate overshoot more effectively, as prior work has shown that motor cortex activity reflects both movement direction and speed [7].

3) Stage-specific regression: Another potential avenue to reduce the early spike in RMSE and prevent early overshoot is a stage-specific regressor, analogous to the proposed stage-specific classifier. This idea is supported by evidence that some neural units show high variability before movement, indicating preparatory activity in the motor cortex [8]. These preparatory signals could be used to design an early regressor stage tailored to movement onset. This approach may enable the regressor to better handle the transition from rest to movement, though it could increase computational complexity.

4) Neural Networks: Due to data scarcity and short runtime constraints, we did not explore neural network architectures despite their growing traction in neural decoding [4]. However, recent work suggests that even shallow or transfer-learned networks can outperform classical methods in low-data regimes by capturing nonlinear patterns in neural activity, making them a promising direction for future exploration [9].

In conclusion, we developed a two-stage causal decoder using LDA and linear regression, showing that with careful tuning, classical methods can achieve strong performance in real-time neural decoding.

VI. Attributions

Data: SK. Model: ND, MP, PP. Report: ND, SK, MP, PP

References

[1] L. Xu, M. Xu, T.-P. Jung, and D. Ming, "Review of brain encoding and decoding mechanisms for EEG-based brain–computer interface," *Cognitive Neurodynamics*, vol. 15, no. 4, pp. 569–584, Aug. 2021. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8286913/

[2] J. I. Glaser, A. S. Benjamin, R. H. Chowdhury, M. G. Perich, L. E. Miller, and K. P. Kording, "Machine Learning for Neural Decoding," *eNeuro*, vol. 7, no. 4, pp. ENEURO.0506–19.2020, Aug. 2020. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7470933/

[3] O. Shevchenko, S. Yeremeieva, and B. Laschowski, "Comparative Analysis of Neural Decoding Algorithms for Brain-Machine Interfaces," Dec. 2024, pages: 2024.12.05.627080 Section: New Results. [Online]. Available: https://www.biorxiv.org/content/10.1101/2024.12.05.627080v1

[4] Z. Khademi, F. Ebrahimi, and H. M. Kordy, "A review of critical challenges in MI-BCI: From conventional to deep learning methods," *Journal of Neuroscience Methods*, vol. 383, p. 109736, Jan. 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016502702200262X

[5] P. Sahu, B. K. Singh, and N. Nirala, "Optimized k-nearest neighbors for classification of prosthetic hand movements using electromyography signal," *Engineering Applications of Artificial Intelligence*, vol. 133, p. 108390, Jul. 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0952197624005487

[6] C. S. Soon, M. Brass, H.-J. Heinze, and J.-D. Haynes, "Unconscious determinants of free decisions in the human brain," *Nature Neuroscience*, vol. 11, no. 5, pp. 543–545, May 2008, publisher: Nature Publishing Group. [Online]. Available: https://www.nature.com/articles/nn.2112

[7] D. W. Moran and A. B. Schwartz, "Motor cortical representation of speed and direction during reaching," *Journal of Neurophysiology*, vol. 82, no. 5, pp. 2676–2692, Nov. 1999.

[8] M. Churchland, J. Cunningham, M. Kaufman, S. Ryu, and K. Shenoy, "Cortical preparatory activity: representation of movement or first cog in a dynamical machine?" *Neuron*, vol. 68, no. 3, pp. 387–400, Nov. 2010. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2991102/

[9] X. Zhu, P. Li, C. Li, D. Yao, R. Zhang, and P. Xu, "Separated channel convolutional neural network to realize the training free motor imagery BCI systems," *Biomedical Signal Processing and Control*, vol. 49, pp. 396–403, Mar. 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1746809418303264