Clusterless Decoding of Motor Activity for Dense Multielectrode Probes

Uttam Singh(us2193@nyu.edu), Idan Lau (yl9727@nyu.edu), Tse-Heng (Tyler) Hsueh (th3448@nyu.edu), Ge(James) Jin (gj2148@nyu.edu)

1 Introduction and background

Neural decoding plays a crucial role in understanding the relationship between neural activity and behavior, particularly in brain-computer interface (BCI) applications. Traditional neural decoding methods rely on spike sorting, which assigns detected spikes to individual neurons. However, current spike sorting algorithms often introduce inaccuracies and fail to account for the uncertainty of spike assignments, leading to information loss. The recent emergence of high-density multi-electrode array (HD-MEA) devices has enabled the extraction of rich spike features without the need for explicit spike sorting.

Relying on sorted single-units can bias rate code estimates, motivating the development of alternative spike-sorting-free approaches (Ventura 2008). Building upon this idea, a density-based decoding method was proposed that directly models the distribution of spike features using a mixture of Gaussians (MoG), aiming to bypass spike sorting entirely while maintaining high decoding performance. A Mixture of Gaussians (MoG) model that encodes spike feature uncertainty and incorporates behavior-dependent firing rate modulation, allowing for more flexible and robust decoding compared to traditional methods was introduced (Zhang et al. 2023). This approach outperformed both multi-unit thresholding and spike sorting-based decoding, demonstrating its potential as a state-of-the-art clusterless decoding baseline.

2 Datasets

The datasets used in this study are obtained from large-scale electrophysiological recordings conducted by the International Brain Laboratory (IBL) (IBL et al. 2022). The recordings were collected using Neuropixels (NP) probes implanted in mice performing a decision-making task. Each dataset consists of multiple trials, where neural activity and behavioral variables such as choice, face motion energy, and wheel speed were recorded.

Each trial lasts 1.5 seconds and is divided into 30 time bins of 50 milliseconds each. The NP probes provide high-density neural recordings across multiple brain regions, enabling the capture of rich spatiotemporal spike features (Steinmetz et al. 2021). However, the high density of recorded signals introduces challenges such as spike collisions and noise contamination, which complicate traditional spike sorting approaches (Buccino et al. 2022).

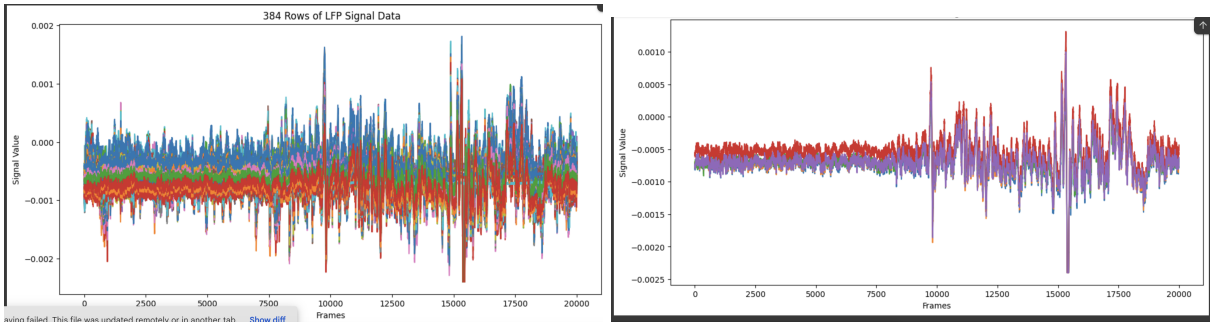
2.1 Data Visualization

2.1.1 Raw Data from Channels

IBL dataset has raw data in alf folder. It consists of 385 channels, 384 being neuropixel channels and 1 sync channel. A visualization of raw data from 384 channels for 2000 frames and a small subset of 5 channels have been represented in Fig:1. In IBL dataset not all 384 channels are useful. There are 10 bad channels are removed using raw indexes made available along with the dataset. A very important observation drawn from the second figure is that although 5 channels are being plotted, effectively only two channels can be distinguished. This observation can be critical for data reductions The neural activity which is being decoded as part of this exercise is the wheel movement. The correctness of wheel movement can be inferred with the timestamps of rewards which is available as licks in the data. An overlay of wheel velocity and water rewards is shown in the fig.2. Since the overlay is dense, a conclusion can be made that the subject's response to stimulus was mostly correct.

2.2 Signal Densities

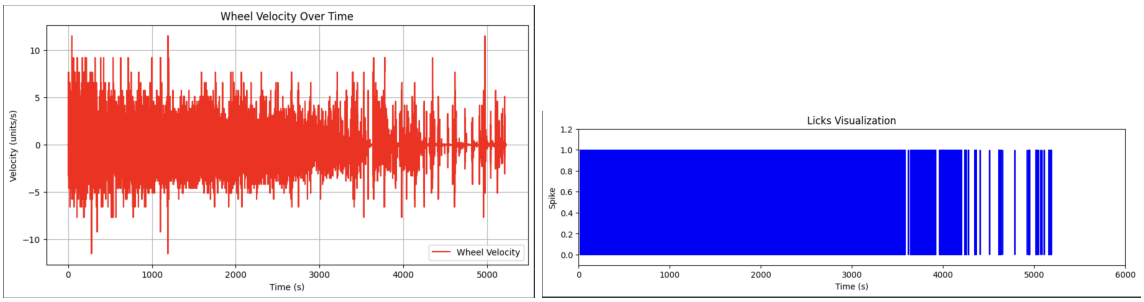
To further understand the subject's neuronal behaviour when correctly responding to a stimulus, we selected a particular stimulus event within the dataset with a quick response time (~0.09 seconds) and correct choice. From this window, action potential signals from all of the probe's channels were sampled.



(a) Raw Data from 384 channels for 20000 frames

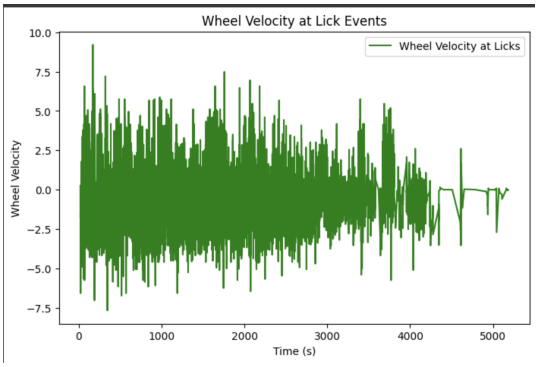
(b) Raw Data from 5 channels for 20000 frames

Figure 1: Raw Data Visualization



(a) Wheel velocity over Time

(b) Water Licks for the subject over time



(c) Wheel Velocity whenever the subject received rewards

Figure 2: Activity Plots

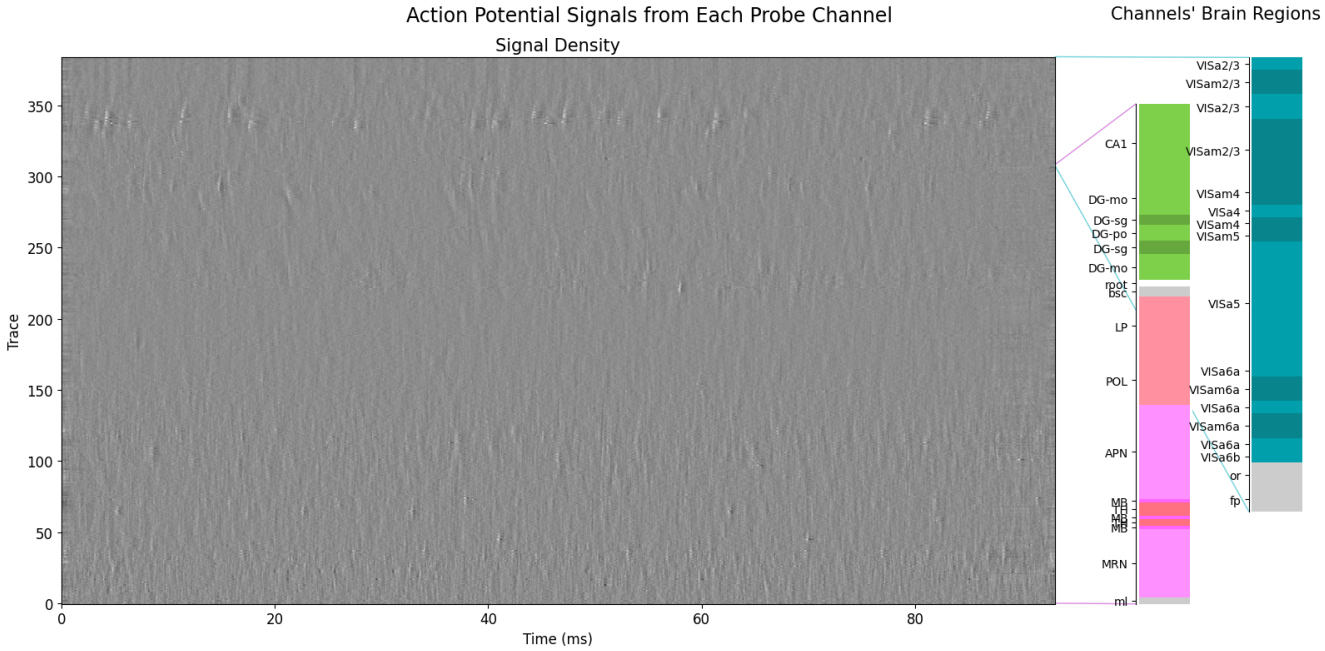


Figure 3: 0.09 Seconds of AP Signal From an Experimental Event of Interest

Since local field potential signals are slower acting, it was sampled from a larger (~ 1.1 seconds) window between the stimulus being turned on and off. Based on these raw data, we visualized the signal densities along all channels of the probe (fig. 3 and fig. 4). The Allen Atlas acronyms of the brain regions that each channel is located in are also shown in accompanying bar plots.

Spectrograms (fig. 5) for both action potential signals and local field potentials are also provided for the same time windows. However, due to the large number of channels along the probe, only the ~5 channels corresponding to the Medial lemniscus (ml) region was chosen and averaged. Both types of plots demonstrate that sparse low frequency activities are found in certain channels.

2.2.1 Spectral density

2.2.2 LFP Power Spectrum

This plot is critical for understanding how neural activity varies across different depths and frequency ranges. By visualizing the power spectrum along the probe (fig. 6), we can identify the specific frequencies that dominate at certain depths.

2.2.3 LFP PSD / RMS

We also explore alternative representations based on Local Field Potential (LFP) features. Fig. 7 shows the LFP Power Spectral Density (PSD) across frequency bands and channels, capturing the frequency characteristics of neural signals. Meanwhile, Fig. 8 presents the LFP Root Mean Square (RMS) of the LFP signals across channels, which provides a compact measure of neural activity amplitude. These representations serve as potential input features for downstream tokenization strategies, offering an alternative approach for motor decoding.

2.2.4 Probe Visualisation

This visualization(Fig:9)contains the probe's locations on the brain.

2.3 Spike Trains Visualization

Another component we utilize from the IBL dataset is the sorted spikes provided through Kilosort Pachitariu et al. 2016. This representation presents the firing timestamp and intensity of all spikes from all identified neurons across the trial duration (fig. 10)

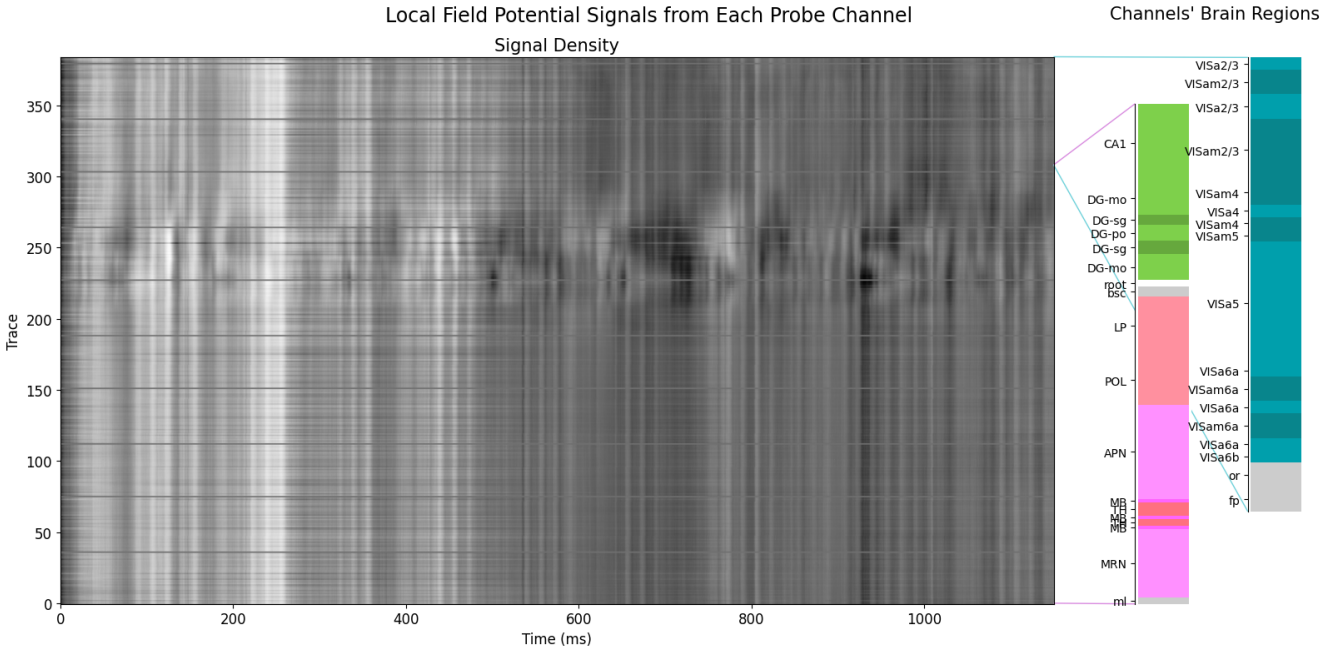


Figure 4: 1.1 Seconds of LFP Signal From an Experimental Event of Interest

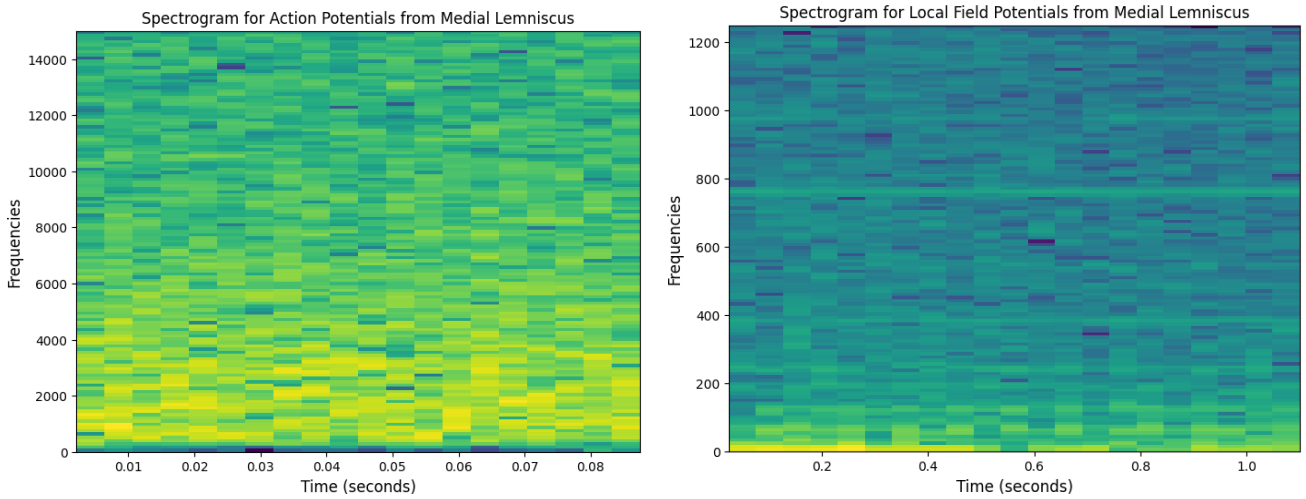


Figure 5: Spectrogram of AP and LFP Signals From the Same Experimental Event of Interest

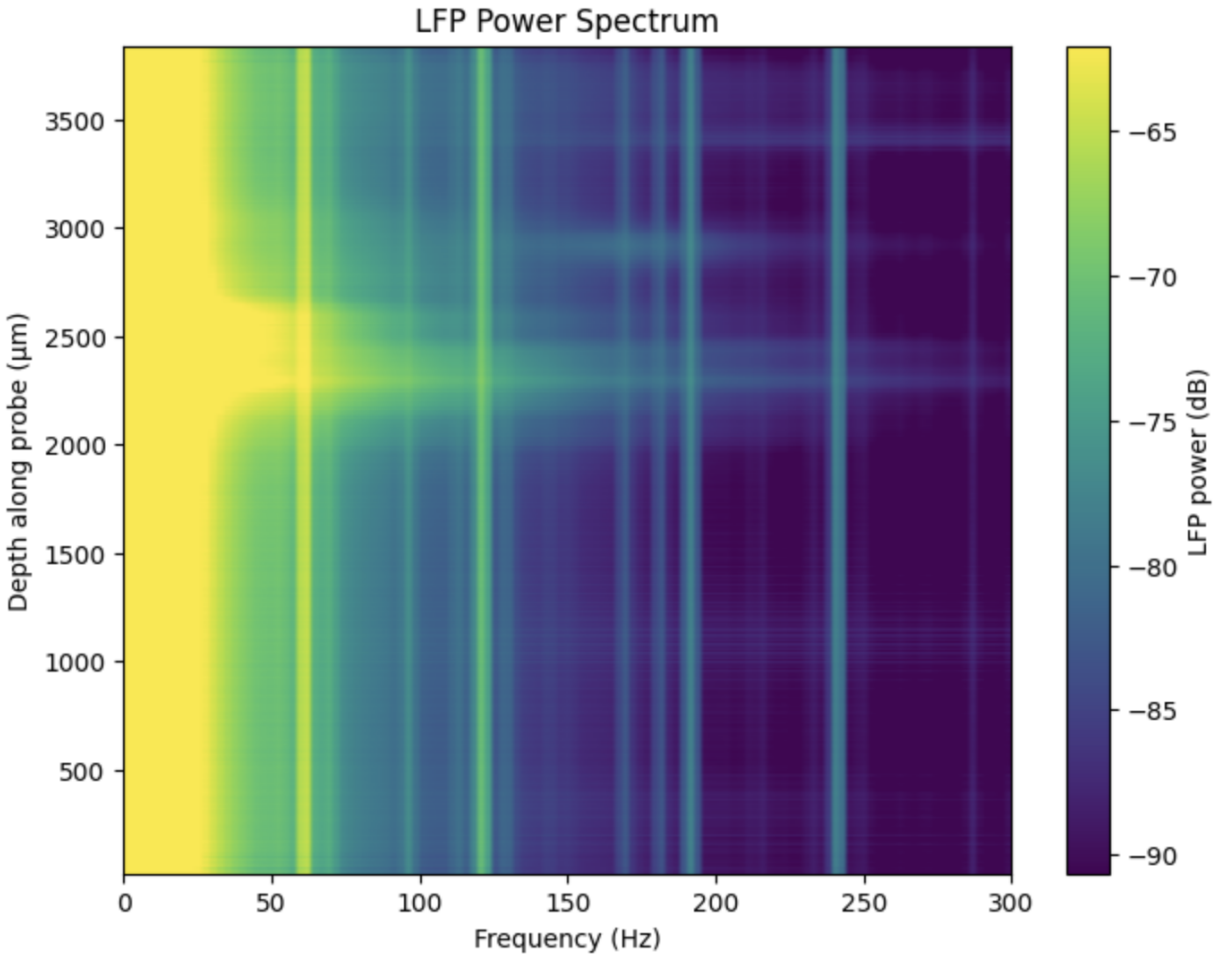


Figure 6: LFP Power Spectrum Across Probe Depth

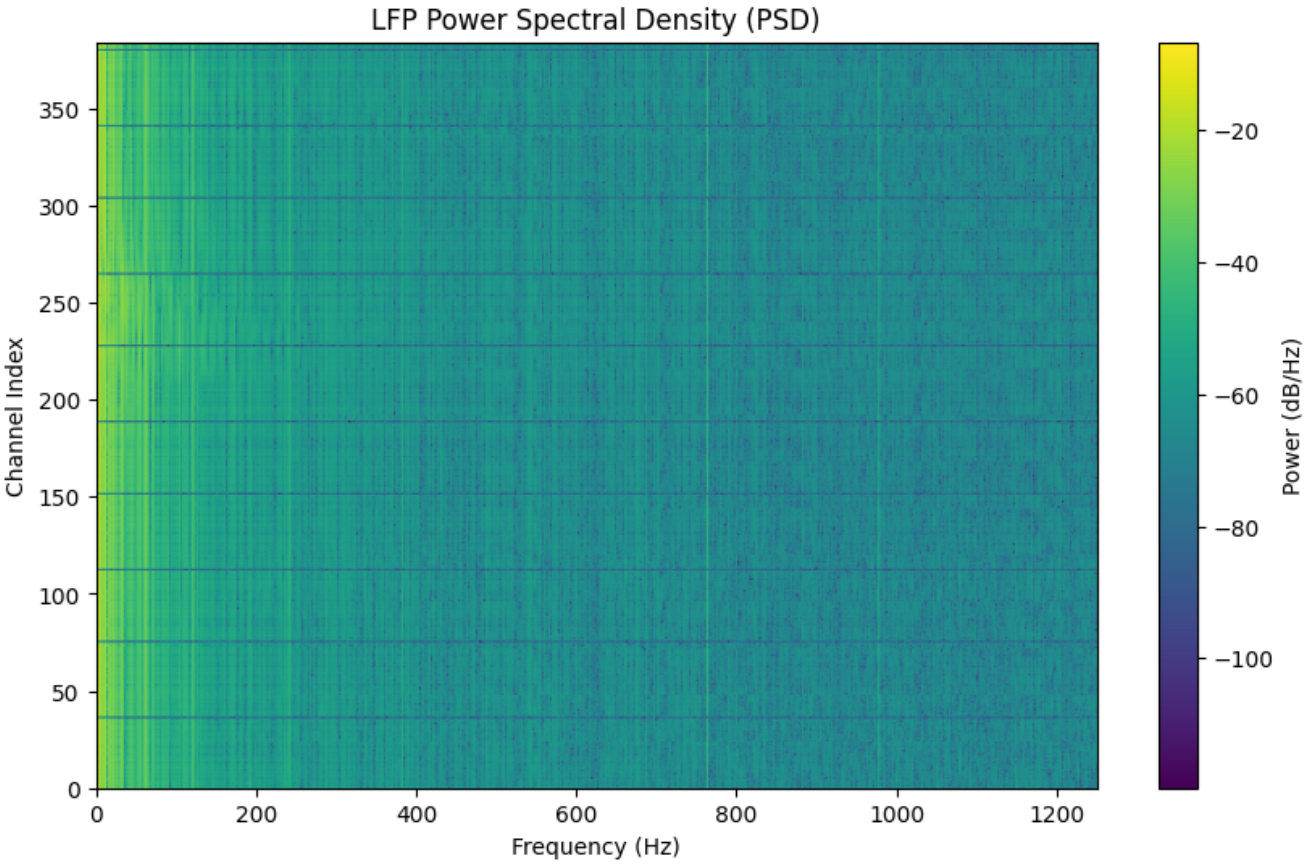


Figure 7: LFP Power Spectral Density (PSD)

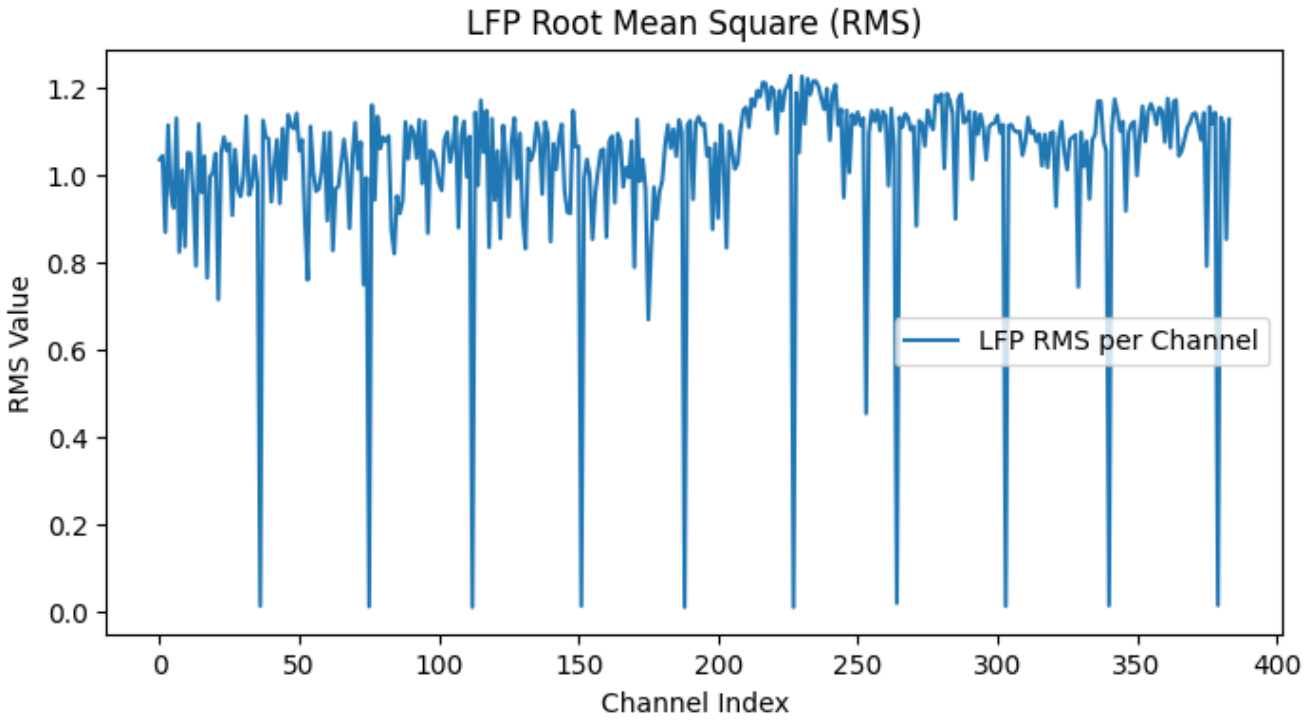


Figure 8: LFP Root Mean Square (RMS)

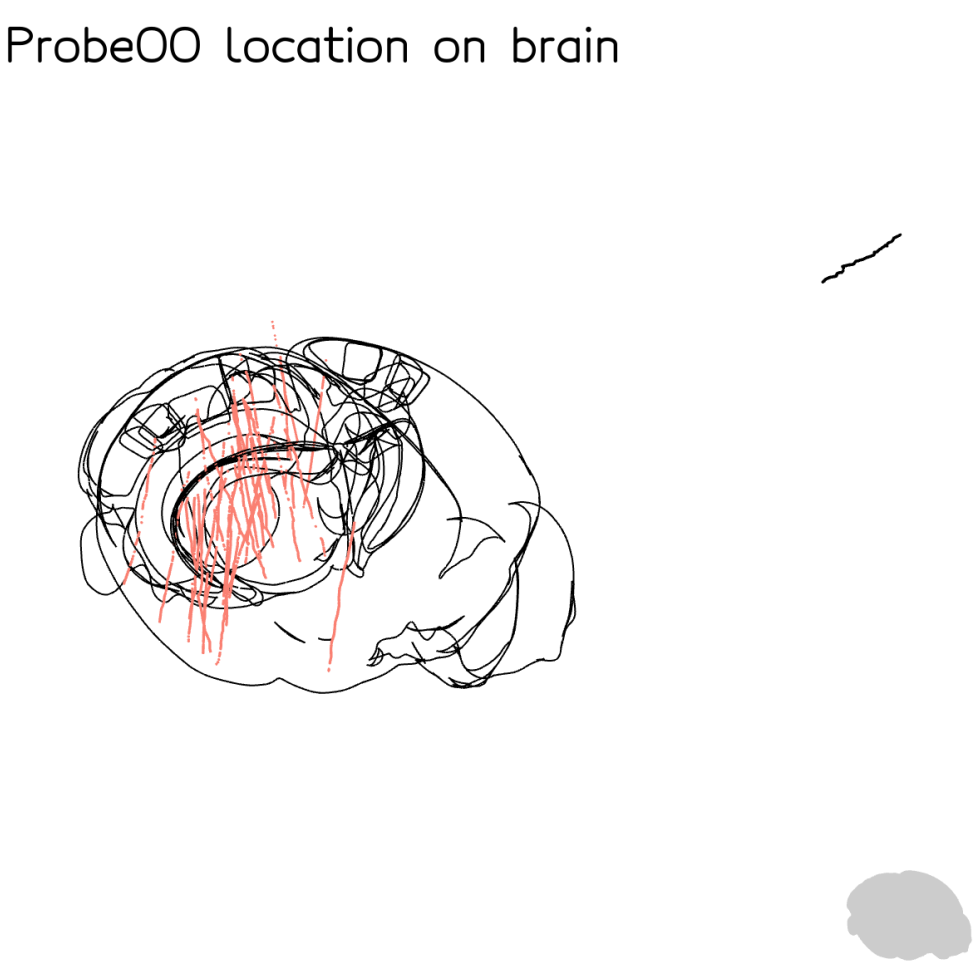


Figure 9: Probe00 location on mouse brain

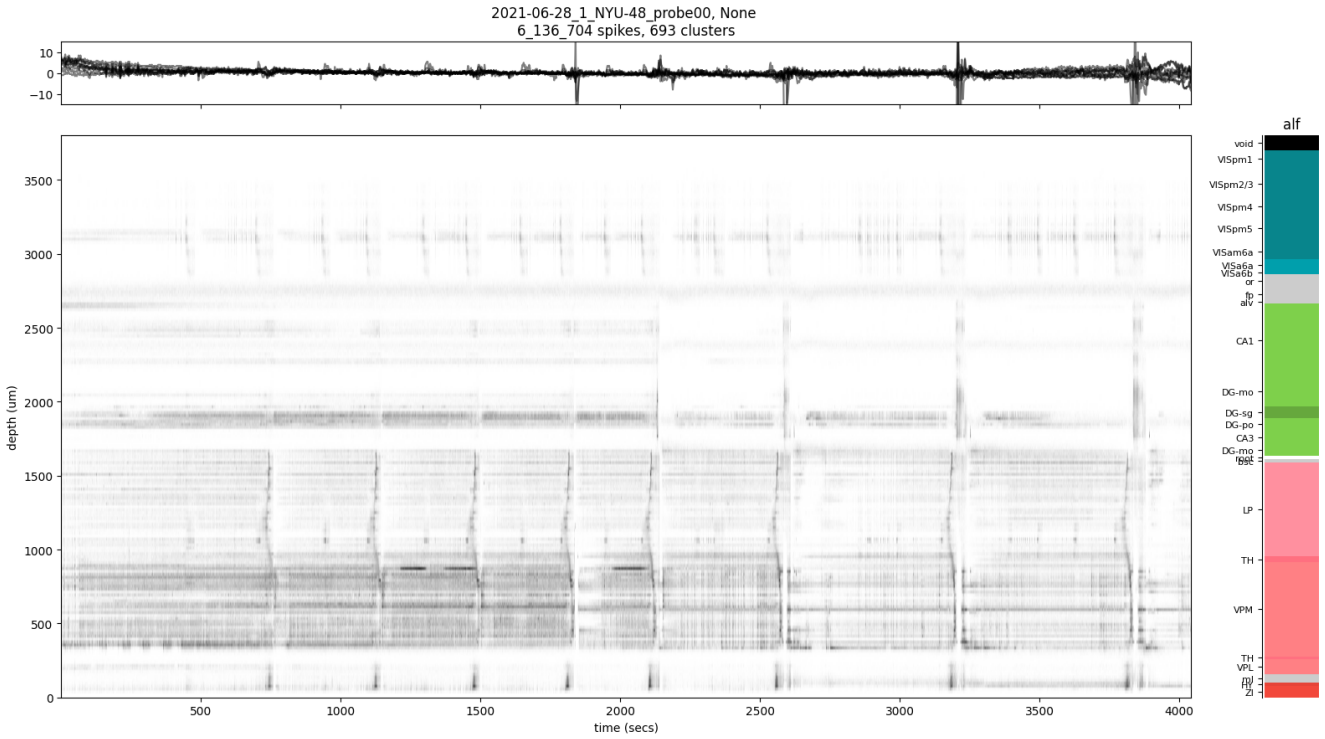


Figure 10: Spike Raster of Training Dataset

2.4 Data Preprocessing

To ensure data quality, the following preprocessing steps are applied:

- **Destriping and artifact removal**: Line noise and voltage leakage artifacts are removed using a de-striping procedure (IBL 2022).
- **Spike detection and denoising**: A subtraction-based spike detection method is applied to improve signal clarity and mitigate overlapping spike artifacts (Boussard et al. 2023).
- **Spike localization**: The spatial position of detected spikes is estimated along the x- and z-axis of the probe, which helps in feature extraction for decoding (Boussard et al. 2021).
- **Motion correction**: To account for motion drift in the recordings, a drift correction algorithm is applied to align spike positions across trials (Windolf et al. 2023).

The final spike features used for decoding include spike locations and maximum peak-to-peak amplitudes, which have been shown to contain rich information about neural activity (Hilgen et al. 2017).

3 Methods

The goal of this study is to evaluate the performance of a clusterless decoding algorithm that bypasses traditional spike sorting and directly utilizes spike feature distributions for neural decoding.

3.1 Baseline Methods

To compare the performance of the clusterless decoding algorithm, the following baseline methods are proposed based on the literature will be used.

3.1.1 Clusterless Decoding using Mixture of Gaussains

In the paper by Zhang et al. 2023, the authors employ a probabilistic framework based on a Mixture of Gaussians (MoG) to model spike features dynamically in response to behavioral variables.

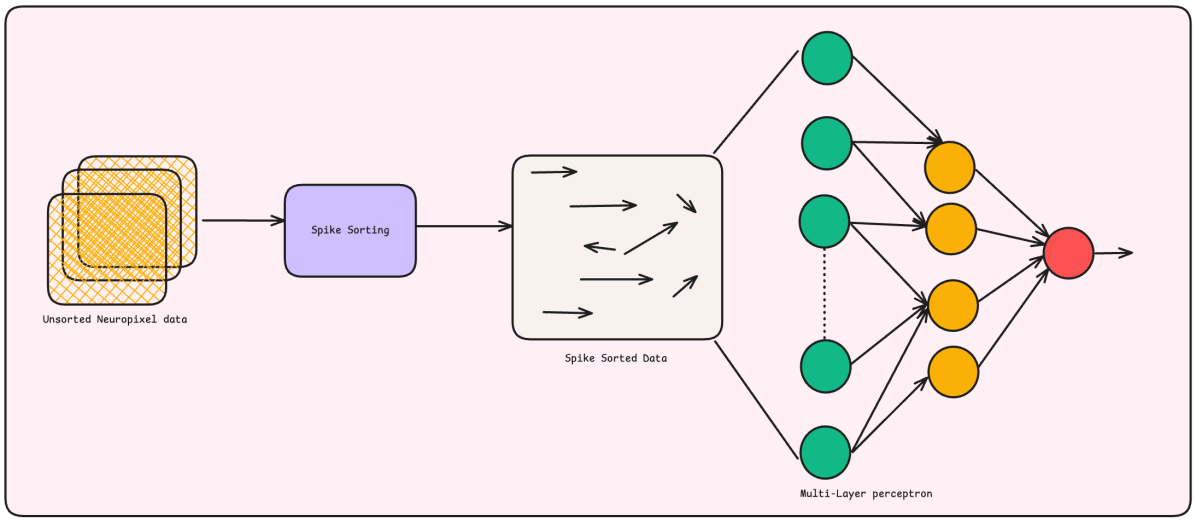


Figure 11: Pipeline for Spike Sorted Decoding Activity

Given an electrophysiological recording composed of K trials, each trial is divided into T time bins. The spike feature distribution in each time bin is represented as:

$$s_{itk} = (x_{itk}, z_{itk}, a_{itk}) \in \mathbb{R}^3$$

where x_{itk} and z_{itk} denote the spatial coordinates of the spike on the probe, and a_{itk} is the spike's peak-to-peak amplitude.

The MoG model assumes that the spike features are generated from a set of latent Gaussian components:

$$s_{itk} \sim \sum_{c=1}^{C} \pi_{ctk} \mathcal{N}(\eta_c)$$

where π_{ctk} is the behavior-dependent mixing proportion, and $\eta_c = (\mu_c, \Sigma_c)$ represents the mean and covariance of the c-th Gaussian component.

The firing rates of the components are modeled as:

$$\lambda_{ctk} = \exp(b_c + \beta_{ct} \cdot y_{tk})$$

where b_c is the baseline firing rate, β_{ct} represents the behavior modulation term, and y_{tk} is the observed behavioral variable.

To estimate the latent parameters, authors apply variational inference techniques, including Automatic Differentiation Variational Inference (ADVI) (Kucukelbir et al. 2017) and Coordinate Ascent Variational Inference (CAVI) (Bishop et al. 2006). The objective is to maximize the evidence lower bound (ELBO) defined as:

$$\mathcal{L} = \mathbb{E}_q \left[\log p(s, b, \beta \mid y) - \log q(b, \beta) \right]$$

where $q(b, \beta)$ is a variational distribution that approximates the true posterior.

3.1.2 Spike Sorting with MLP

Another method to prove the effectiveness of clusterless decoding is to compare its performance with spike-sorted clustering methods. For this approach, the data will be passed to a spike sorting algorithm which will cluster the data. we will use this to train a multi-layer perception(MLP) which would be used to decode the output behaviours. A representation of this methodology is shown in Fig.11

3.1.3 Evaluation Metrics

Authors evaluate decoding performance using 5-fold cross-validation. For continuous behavioural variables (e.g., motion energy and wheel speed), they compute the coefficient of determination (R^2), while classification accuracy is used for discrete behavioural tasks (e.g., choice). Additionally, they analyze the robustness of each method under varying levels of spike sorting quality, different brain regions, and multiple probe geometries (Steinmetz et al. 2021).

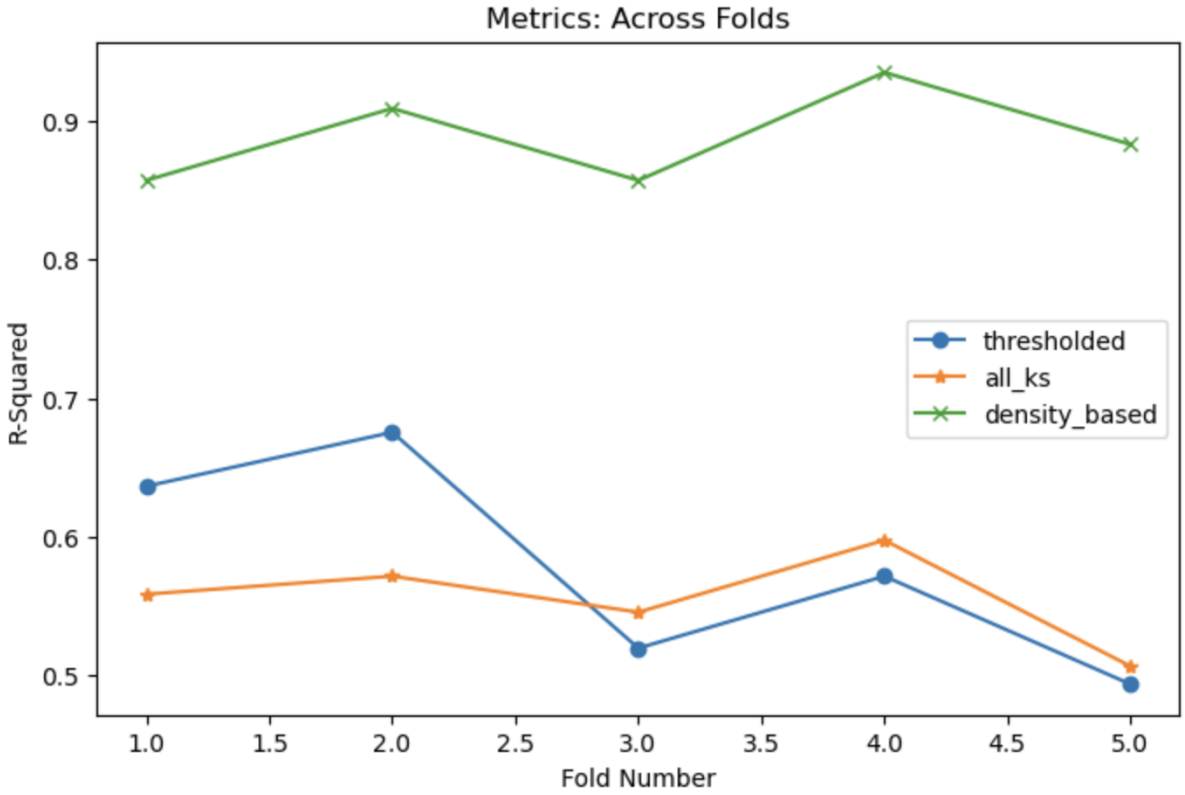


Figure 12: Wheel velocity R^2 of MoG and its two linear decoders across the 5 folds

3.1.4 Baseline Results

We present the baseline results in this sections for both the proposed baseline methods. As the MoG supports both discrete and continuous prediction, we predicted the wheel velocities of the moues per trial as well as its choice of turning the wheel left or right. On the other hand, the MLP baseline is solely focused on decoding wheel velocities.

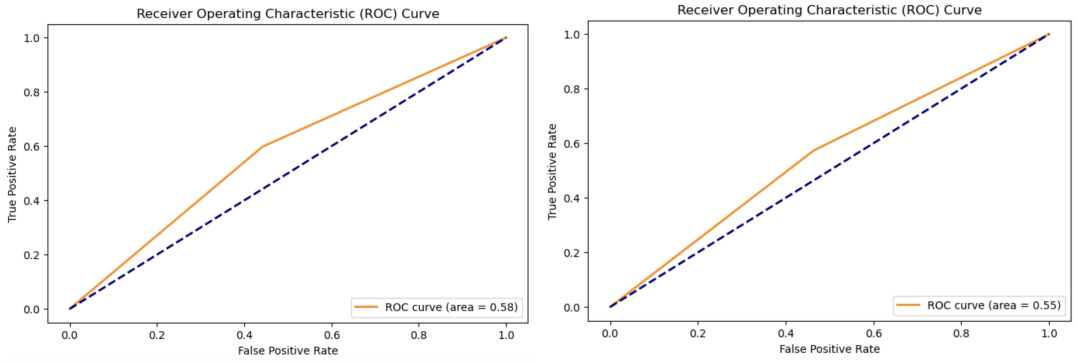
The MoG method itself also internally compares against two linear decoders with spike sorted data as input, namely one that decodes using all available units ("all_ks") and one that uses units filtered by Kilosort Pachitariu et al. 2016 based on spike sorting quality ("thresholded"). In figure 12, the wheel velocity prediction performance of the MoG method ("density_based") is presented.

Table 1: Wheel Choice Prediction Results from MoG Methods

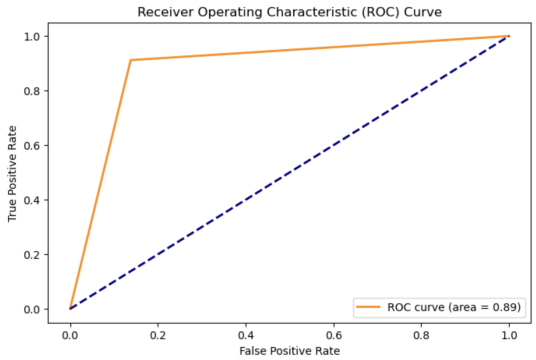
	Accuracy	Precision	Recall
density_based	0.8883	0.8619	0.8966
thresholded	0.5792	0.558	0.551
all_ks	0.5558	0.5359	0.5272

For MoG's classification performance on deducing choice of wheel direction, we observed Receiver Operating Characteristic Curves (ROC) (fig.13) and confusion matrices (fig. 14). These results are also represented quantitatively in 1

As the MLP method does not participate in the classifcation task, we just visualized its predicted wheel velocity ("predicted") against the ground truth wheel velocity ("gt") in fig. 15. However, though the input data was formatted according to Glaser et al. 2020 for this type of MLP, the model visibly failed to learn much underlying pattern, despite the overall decrease in loss throughout the entire training duration (fig. **fig:mlp_pred**). It's plausible that insufficient data preprocessing is a main cause, since the MoG method achieves reasonable decoding performance from the linear decoders it runs alongside the MoG algorithm.



(a) Thresholded (b) all-ks



(c) density based

Figure 13: ROC Curves for MoG's wheel direction prediction

4 Final project plan

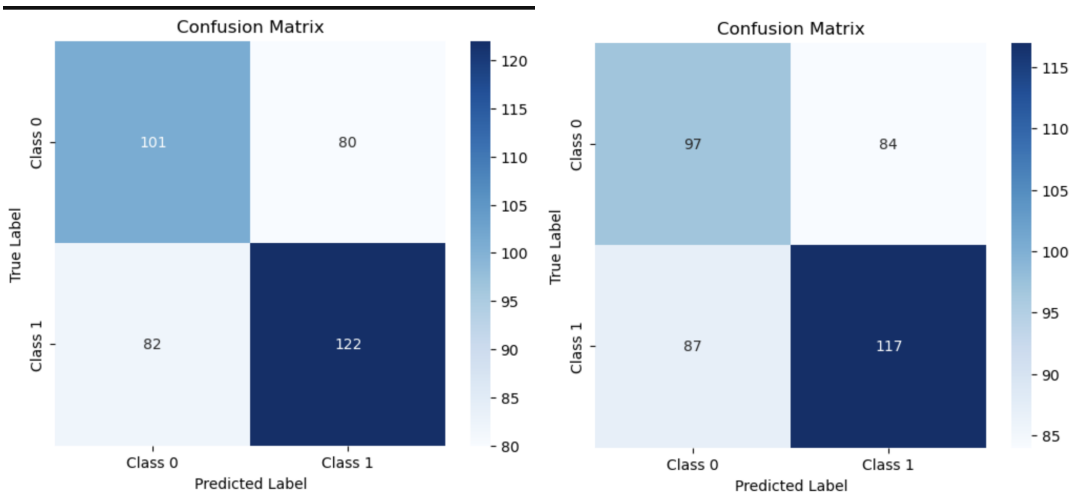
From our preliminary experiments using MoG-based density decoding and MLP-based decoding, we have gathered several key insights that will guide the final model development:

1. Effectiveness of Clusterless Decoding: The MoG-based approach effectively captures uncertainty in spike assignments and provides a robust representation for decoding behavior. Meanwhile, the Multi-Layer Perceptron (MLP) enhances the model, facilitating a more effective comparison with other approaches.

2. Limitations and Areas for Improvement: The MLP model, although straightforward, does not fully leverage the temporal dependencies in the data, which suggests that a more sequential architecture (e.g., Transformer-based models) may improve performance.

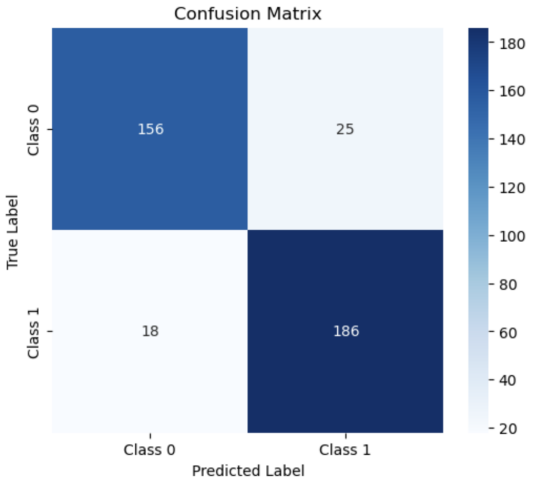
Our project aims to advance spike sorting-free neural decoding by developing a Transformer-based clusterless decoding model that builds on the strengths of the MoG approach while enhancing efficiency and scalability. To further improve performance, we propose making the entire process self-supervised by replacing explicit spike trains with latent variables representing neural activity. This allows the model to infer behaviour directly from unsorted spike features, potentially uncovering hidden neural dynamics, and improving both accuracy and scalability. In the next phase, we will implement and refine our Transformer-based model, comparing it against MoG and MLP baselines while validating its generalization across different datasets and behavioural tasks. Through this approach, we aim to create a more robust and adaptable method for neural decoding.

To address the concerns about poor predictions due to data quality, we will first verify that our data loader correctly matches inputs with their labels. We also plan to implement full data preprocessing—destriping, subtraction-based spike detection and denoising, spike localization, and drift registration—following the procedure presented in our MoG baseline Zhang et al. 2023. Furthermore, since both neural activity and wheel velocity are temporal, we will explore using a recurrent-based decoder instead of an MLP to better capture these dependencies.



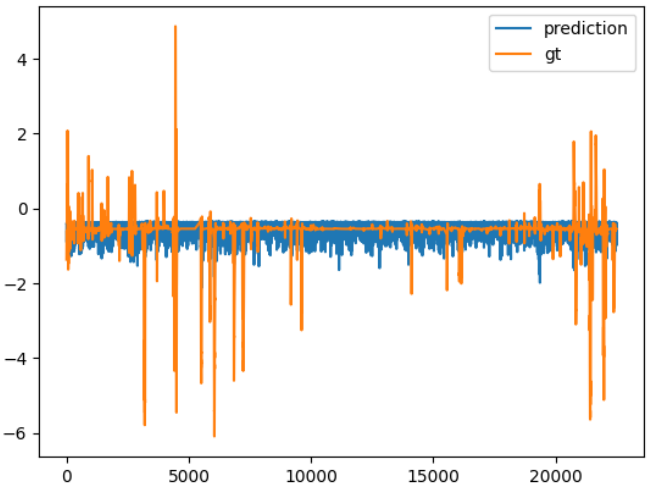
(a) Thresholded

(b) all-ks

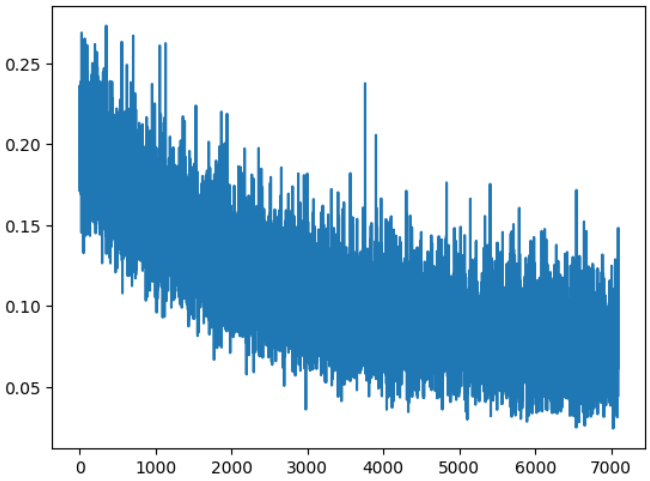


(c) density based

Figure 14: Confusion matrices for MoG's wheel direction prediction



(a) MLP wheel velocity prediction



(b) MLP training loss

Figure 15: Results from MLP

5 Author contributions

Special acknowledgments goes to TianXiao for helping us with the project's initial ideation and workflow and Malhar and Rishab for sharing the visualization tools and clearing a lot doubts in office hours.

Table 2: Author Contributions

	Idan	James	Tyler	Uttam
Introduction & Background	25%	20%	30%	25%
Datasets (Data Visualization)	15%	30%	30%	25%
Methods (Baseline Results)	30%	30%	10%	30%
Final Project Plan	30%	20%	30%	20%

6 Initial and fine-tuned results:

Our group is currently exploring different directions to see which yields the most interesting results, which we will focus on later.

6.1 Utilising transformers to model temporal neuron cluster behaviour:

In this setup, the GLM models the changes in the MOG mixing proportions over time in response to behavior. The parameters learned from the GLM are then used as priors for the ADVI layer. (The input data is spike features instead of raw data)

We propose replacing the GLM with a transformer model. Our hypothesis is that the transformer can encode more complex information, thereby enabling a better estimation of the MOG mixing proportions. In turn, this should improve downstream decoding performance. Initial results are promising. For example, on the behavioral choice (classification) task, slight gains were observed in folds 1, 3, and 5 compared to the baseline. On the wheel velocity dataset, however, gains are less consistent—folds 1 and 5 show improvement with the new setup, whereas fold 2 performs better using the baseline.

Below is the mathematical formulation for both the GLM baseline and our transformer branch setup:

GLM Baseline:

$$\begin{aligned} &\mathbf{U} \in \mathbb{R}^{n_c \times n_r}, \quad \mathbf{V} \in \mathbb{R}^{n_r \times n_t}, \quad \mathbf{b} \in \mathbb{R}^{1 \times n_c \times 1}, \\ &\boldsymbol{\beta} = \mathbf{U} \cdot \mathbf{V} && \in \mathbb{R}^{n_c \times n_t}, \\ &x_{\text{pred}} = \boldsymbol{\beta}[\texttt{None}, :, :] \odot y + \mathbf{b}, \end{aligned}$$

Our Transformer Branch Setup:

$$\begin{aligned} \mathbf{X} &= \text{TransformerEncoder}(\mathbf{X}) && \in \mathbb{R}^{T \times B \times d}, \\ \hat{\mathbf{X}} &= \text{Decoder}(\mathbf{X}) && \in \mathbb{R}^{B \times T \times C}, \\ \boldsymbol{\beta} &= \mathbf{U} \cdot \mathbf{V} && \in \mathbb{R}^{C \times T}, \\ \mathbf{G} &= \boldsymbol{\beta}[\texttt{None}, :, :] + \mathbf{b} && \in \mathbb{R}^{1 \times C \times T}, \\ \mathbf{O} &= \alpha \cdot \hat{\mathbf{X}} + (1 - \alpha) \cdot \mathbf{G} && \in \mathbb{R}^{B \times C \times T}. \end{aligned}$$

In the equations above, both for the baseline and our transformer setup, the matrices $\mathbf{U}$ and $\mathbf{V}$ are the parameters optimized during training. These parameters are subsequently used as learned priors for the ADVI step.

In the current transformer setup, the $\mathbf{U}$ and $\mathbf{V}$ matrices are used in conjunction with the bias term $\mathbf{b}$ in an additive manner. Our next step will be to incorporate these matrices directly as a projection

matrix applied to the input to see if it improves results. We might also experiment with fancier embedding methods. Most importantly, we will also be analyzing to see where our method and baseline method fails/succeed, giving insight to the model's underlying behavior.

Modeling Results on Choice Behaviour Dataset
(EID: c4f6665f-8be5-476b-a6e8-d81eeae9279d)

Table 3: Transformer Modeling (Our model)

Fold	Accuracy	AUC
1	**0.795**	**0.775**
2	**0.855**	**0.847**
3	0.880	0.868
4	**0.831**	**0.799**
5	0.843	0.820

Table 4: Baseline GLM Modeling

Fold	Accuracy	AUC
1	0.783	0.766
2	0.819	0.802
3	0.880	0.868
4	0.819	0.789
5	0.843	0.820

Modeling Results on Wheel Velocity Dataset
(EID: c4f6665f-8be5-476b-a6e8-d81eeae9279d)

Table 5: Transformer Modeling (Our model)

Fold	R2	Corr
1	**0.117**	**0.421**
2	0.161	0.418
3	0.191	0.462
4	**0.260**	0.512
5	**0.219**	**0.473**

Table 6: Baseline GLM Modeling

Fold	R2	Corr
1	0.066	0.371
2	**0.184**	**0.436**
3	0.198	0.467
4	0.259	**0.513**
5	0.207	0.464

Updates from previous initial result:

Now, the beta matrix of the priors U and V is directly used as matrix multipliers for the transformer-refined feature of the features. Below are the results. Unfortunately, this setup did not improve our model, especially in the regression task, where our setup is still not convincingly better. As a next potential step, we plan on introducing additional learnable transformations before applying the priors, or exploring alternative ways to integrate U and V are obtained through attention-based modulation instead of direct multiplication.

$$
\begin{aligned}
\mathbf{X} &= \text{TransformerEncoder}(\mathbf{X}) && \in \mathbb{R}^{T\times B\times d},\\
\hat{\mathbf{X}} &= \text{Decoder}(\mathbf{X}) && \in \mathbb{R}^{B\times T\times C},\\
\boldsymbol{\beta} &= \mathbf{U}\cdot\mathbf{V} && \in \mathbb{R}^{C\times T},\\
\mathbf{O} &= \boldsymbol{\beta}[\texttt{None},:,:]\times\hat{\mathbf{X}}+\mathbf{b} && \in \mathbb{R}^{B\times C\times T}.
\end{aligned}
$$

Modeling Results on Choice Behaviour Dataset
(EID: c4f6665f-8be5-476b-a6e8-d81eeae9279d)

Table 7: Transformer Modeling (Our model)

Fold	Accuracy	AUC
1	0.795	0.775
2	0.819	0.802
3	0.880	0.868
4	0.831	0.799
5	0.819	0.795

Table 8: Baseline GLM Modeling

Fold	Accuracy	AUC
1	0.783	0.766
2	0.819	0.802
3	0.880	0.868
4	0.819	0.789
5	0.843	0.820

Modeling Results on Wheel Velocity Dataset
(EID: c4f6665f-8be5-476b-a6e8-d81eeae9279d)

Table 9: Transformer Modeling (Our model)

Fold	R2	Corr
1	0.143	0.438
2	0.176	0.437
3	0.235	0.494
4	0.182	0.452
5	0.219	0.473

Table 10: Baseline GLM Modeling

Fold	R2	Corr
1	0.066	0.371
2	0.184	0.436
3	0.198	0.467
4	0.259	0.513
5	0.207	0.464

6.2 Transformer-Based Decoding with Variable Labeling Thresholds

To explore alternative transformer-based decoding strategies for motor behavior prediction, we first constructed a classification pipeline using local field potential (LFP) data and movement labels. This pipeline aims to classify whether the subject is moving or still based on LFP data aligned to stimulus onset times. First, we use the IBL dataset to extract LFP segments around `stimOn_times` within a 750ms symmetric window. Raw LFP signals from all channels are high-pass filtered at 2Hz to remove slow drifts, then z-scored per channel to normalize for baseline fluctuations. We reject flat segments with extremely low variance to mitigate noise from non-informative trials.

Movement labels are generated by computing the subject's wheel velocity from the gradient of position data. If the absolute average velocity within the window surrounding a stimulus exceeds a predefined threshold, the corresponding LFP segment is labeled as "moving" (1); otherwise, it is labeled as "still" (0).

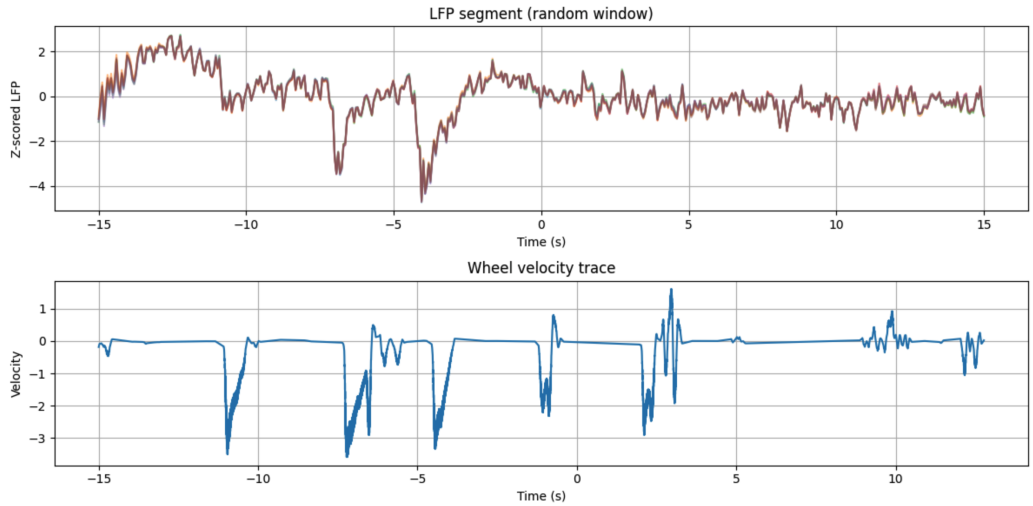


Figure 16: Z-scored LFP signals (top) and wheel velocity trace (bottom) from a random time window, showing temporal correspondence between neural activity and movement.

To better understand the temporal relationship between local field potential (LFP) activity and motor output, we visualized a randomly selected LFP segment alongside the subject's wheel velocity trace over the same time window. The top panel shows multi-channel LFP signals after high-pass filtering at 2 Hz and z-scoring per channel, which removes baseline drift and scales all channels to a comparable range. The bottom panel plots the wheel velocity trace, computed as the temporal gradient of the recorded wheel position.

These visualizations provide key insights for both label construction and model development. We observe that movement onset (e.g., large negative or positive velocity spikes) often coincides with transient broadband fluctuations in the LFP signal, suggesting that motor-related information is embedded in these non-spike, field-based neural recordings.

This observation supports our approach of using z-scored, raw LFP segments as model input for decoding motor behavior. Furthermore, by aligning LFP and velocity signals in a shared time window,

we ensure that downstream labels (e.g., binary movement or continuous velocity) reflect physiologically relevant neural dynamics rather than noise or artifacts.

To classify movement, we introduce an **Improved Tiny Transformer** model that combines temporal modeling capacity with efficiency. The architecture consists of a 1D convolutional projection layer to map the 384 LFP channels to an embedding space, followed by positional encodings and a two-layer Transformer encoder. The temporal features are aggregated via mean pooling and passed through a small regression head that outputs a binary classification via sigmoid activation.

We train this model using the binary cross-entropy loss and Adam optimizer, with early stopping based on validation performance. Evaluation on a held-out test set includes standard metrics such as accuracy, precision, recall, F1 score, and AUC-ROC.

We conducted experiments using four configurations of the movement labeling window size (0.75s vs 1.5s) and wheel velocity threshold (2.0 vs 1.75) on LFP data from (PID: 5246af08-0730-40f7-83de-29b5d62b9b6d).

Our best-performing setting used a 1.5-second window with a movement threshold of 1.75, achieving an accuracy of **73.6%**, F1 score of **0.776**, and AUC-ROC of **0.733**. This indicates a significant improvement over chance level (50%) and supports the hypothesis that movement-related information is embedded in broadband LFP structure.

The second-best configuration used a shorter 0.75s window but the same threshold (1.75), resulting in slightly lower but comparable metrics. The longer window (1.5s) helped smooth out noise, but an overly strict threshold (2.0) likely led to label imbalance and reduced classification accuracy. These findings suggest that both temporal context and careful label engineering are critical in LFP-based behavior decoding.

Table 11: Improved Transformer Model on LFP Data
(EID: c51f34d8-42f6-4c9c-bb5b-669fd9c42cd9)
(PID: 5246af08-0730-40f7-83de-29b5d62b9b6d)

Window (s)	Threshold	Accuracy	F1 Score	AUC
0.75	2.0	0.694	0.667	0.703
0.75	1.75	**0.708**	**0.747**	**0.767**
1.5	2.0	0.667	0.625	0.715
1.5	1.75	**0.736**	**0.776**	**0.733**

```
Accuracy : 0.7361111111111112
Precision: 0.7674418604651163
Recall   : 0.7857142857142857
F1 Score : 0.7764705882352941
AUC-ROC  : 0.7325396825396826
```

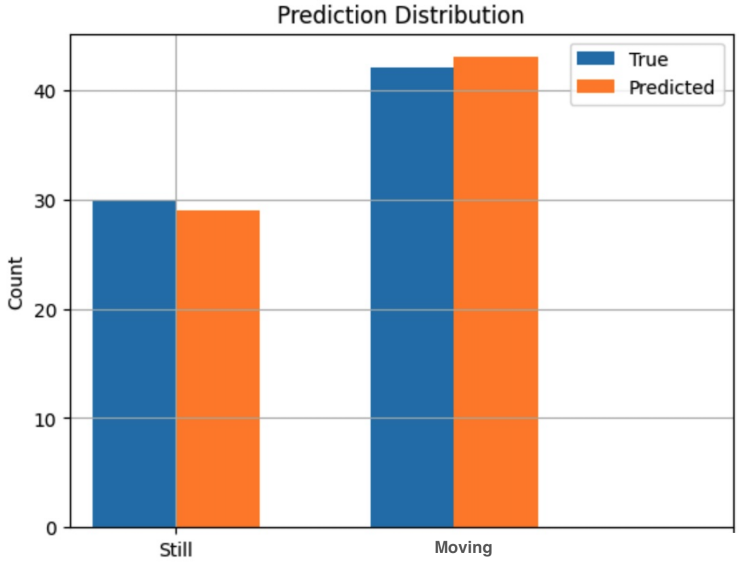


Figure 17: Prediction distribution for movement classification Window = 1.5 and Threshold = 1.75

While the current binary classification framework provides encouraging results in identifying motor activity from LFP signals, there remain several directions for improvement and expansion. To ensure robustness and adaptability, we have outlined the following backup plans and next steps:

1. **Alternative Architectures – Transformer Variants:** To improve sequence modeling and better capture temporal dependencies in LFP signals, we intend to experiment with other Transformer-based models, such as **TimesNet**, which is designed specifically for time series forecasting and has shown promise in high-dimensional temporal domains.

2. **Limitations and Addressing Them:** The current model is limited by manually chosen thresholds and potential class imbalance. Future iterations will include **data-driven threshold selection** to balance classes and reduce labeling noise.

These strategies ensure that our pipeline remains flexible, interpretable, and aligned with the broader goal of unsupervised, spike-free decoding of neural signals for behavioral prediction.

As a first step in exploring alternative architectures, we implemented a binary classification pipeline using **TimesNet** to replace the original Transformer-based model. TimesNet is designed for high-dimensional time series data and incorporates stacked convolutional blocks with residual connections to model temporal dependencies.

Using the same preprocessed LFP segments and binary movement labels (threshold = 1.75, window = 0.75s), we trained the TimesNet model under identical training and validation procedures. The evaluation metrics on the test set were as follows:

Table 12: TimesNet Transformer Model on LFP Data
(EID: c51f34d8-42f6-4c9c-bb5b-669fd9c42cd9)
(PID: 5246af08-0730-40f7-83de-29b5d62b9b6d)

Window (s)	Threshold	Accuracy	F1 Score	AUC	Recall
0.75	2.0	0.528	0.614	0.551	1.000

While the model achieved perfect recall, its low precision and modest AUC suggest that it strongly over-predicted the positive class, leading to high false-positive rates. This underperformance compared to our original Transformer model indicates that further refinement is needed in terms of model capacity, regularization, and label balancing.

As a second step to address limitations due to manually selected labeling thresholds and potential class imbalance, we conducted a systematic grid search across multiple combinations of wheel velocity thresholds and temporal window lengths. The goal is to identify the optimal setting for binary movement classification based on model performance.

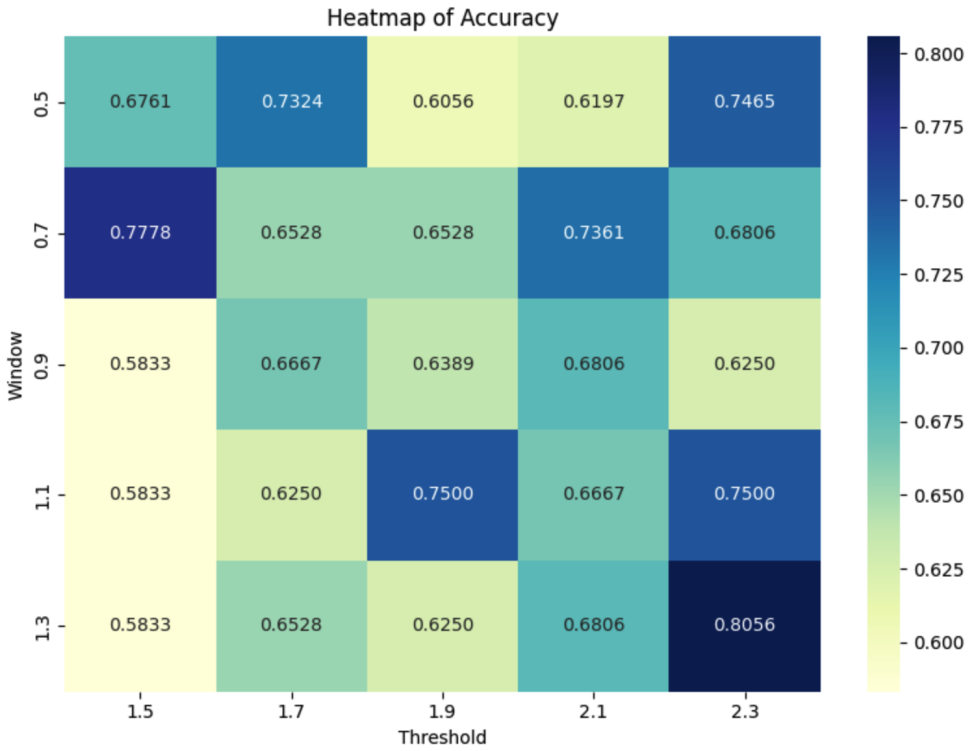


Figure 18: Heatmap of Accuracy across different movement thresholds and time windows

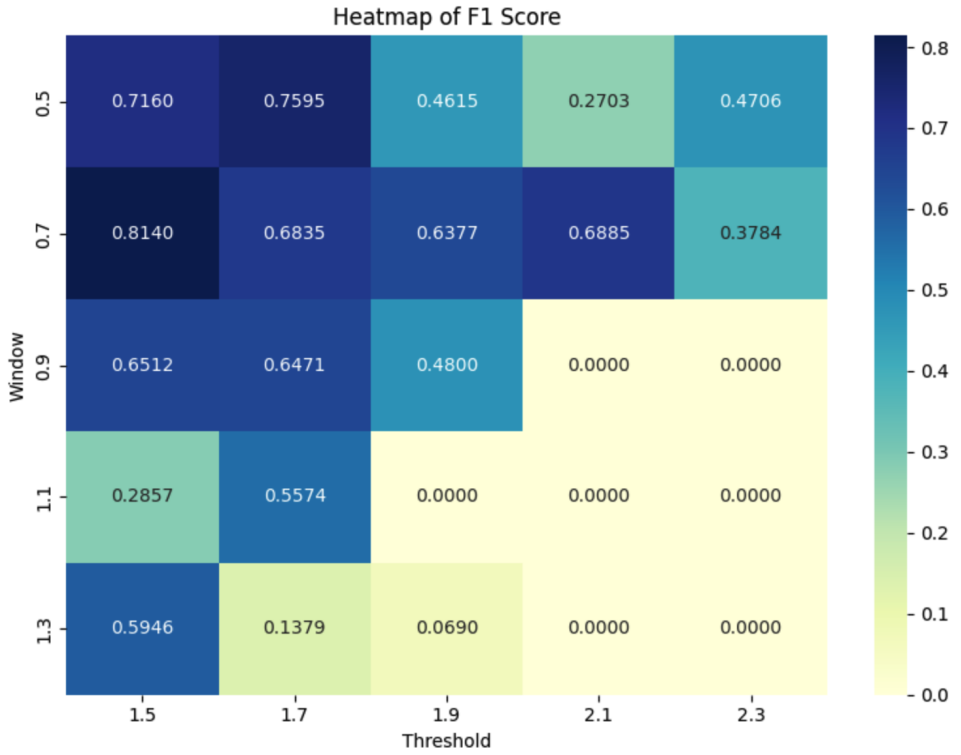


Figure 19: Heatmap of F1 Score indicating impact of class imbalance

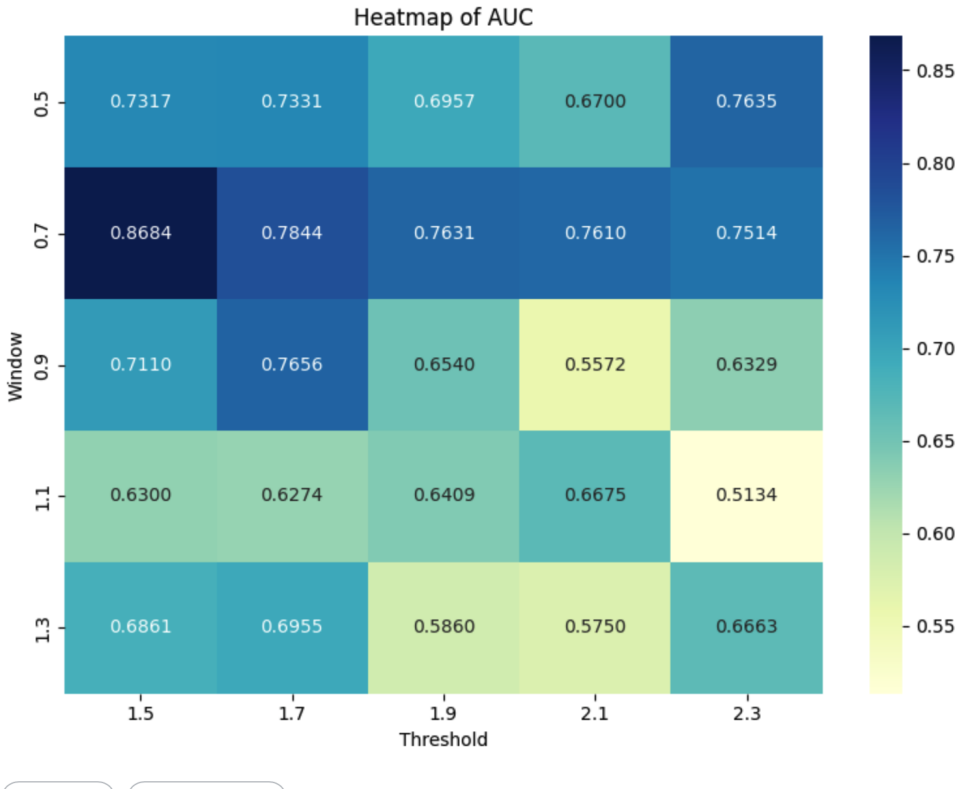


Figure 20: Heatmap of AUC showing overall classification robustness

The results suggest that performance is highly sensitive to both hyperparameters. An overly short window (e.g., 0.5s) can miss relevant motor-related activity, while a too-large velocity threshold (e.g., 2.3) can exclude moderate movement trials, causing severe class imbalance. These effects are especially visible in the F1 score heatmap (Figure 19), where many combinations yield zero F1 due to collapsed predictions.

Best Performing Configuration

The optimal setting was found to be:

- **Window size: 1.5 seconds**
- **Movement threshold: 0.7**

This configuration yields the best trade-off across all metrics:

- Accuracy: **0.736**
- F1 Score: **0.776**
- AUC-ROC: **0.733**

We believe this success is driven by:

1. **Longer window context:** The 1.5s window allows the Transformer to capture extended motor planning and initiation activity in the LFP signal, improving temporal modeling.
2. **Balanced threshold:** A threshold of 0.7 includes more moderate movement examples, avoiding extreme label imbalance. This yields more generalizable learning and reduces overfitting to the dominant class.

Next Steps

Future work will focus on:

- **Automated, data-driven threshold selection** using distribution-aware heuristics or clustering on wheel velocity traces to define dynamic movement cutoffs.
- **Soft labeling or label smoothing** to reflect motion as a spectrum rather than a hard binary label.
- **Advanced loss functions** such as focal loss to mitigate class imbalance during training.

These adjustments aim to improve robustness across subjects and trials while keeping the decoding framework interpretable and computationally efficient.

6.3 Point Cloud Transformer on Spike Features

Since the project's scope also includes decoding from unclustered spikes, an alternative approach is to treat the feature vectors of spikes as points within a point cloud. Thus, the neural decoding problem can be transformed into a point cloud classification / regression problem (Pan 2023). Although spike features have higher dimensionality than 3D point clouds, we explore the potential of applying deep learning point cloud networks to nevertheless learn the underlying patterns of spike features.

Specifically, we chose the Point-MAE network (Pang et al. 2022), a transformer-based architecture trained as a masked autoencoder (fig. 21). In our use case, the network functions as an ordinary transformer encoder, where the embedding and downstream decoding layers are implemented as MLPs. Network structure-wise, a convenient parallel between neural decoding and transformer-based point cloud networks is that although the latter requires a fixed number of input embeddings, the former can be binned into a fixed number of time windows, facilitating a sensible treatment electrophysiology data.

Nevertheless, a naive approach is to simply feed all spikes from an entire trial (a fixed window around when the mouse made a move, as defined by Zhang et al. 2023) into Point-MAE at once. Although simple, this approach ignores the temporal nature of spikes and does not obtain satisfactory performance in the classification task of the choice made by the mouse (table 13).

Table 13: Point-MAE Classification Results Naive Approach

	Accuracy	F1	AUC
From Scratch	0.643	0.73	0.51
Pretrained	0.653	0.772	0.51

Table 14: Point-MAE's Regression and Classification Performance Binned Data

	Regression		Classifiation
R2	0.2738	accuracy	0.7942
Correlation	0.5313	F1	0.8505
[rgb] .91, .91, .91		AUC-ROC	0.7469

6.4 Behavioral decoding via stochastic modeling and statistical inference:

Most existing approaches to behavioral decoding predominantly rely on spike-based features, extracted through spike sorting algorithms. However, spike sorting is an inherently non-deterministic process, often prone to misclassification and loss of information. In contrast, the raw electrophysiological recordings represent the only verifiable source of ground-truth neural activity, albeit contaminated by environmental and experimental noise.

In this study, we propose a decoding framework Fig:(22) that operates directly on raw local field potential (LFP) signals, bypassing the uncertainties associated with spike extraction. Although the

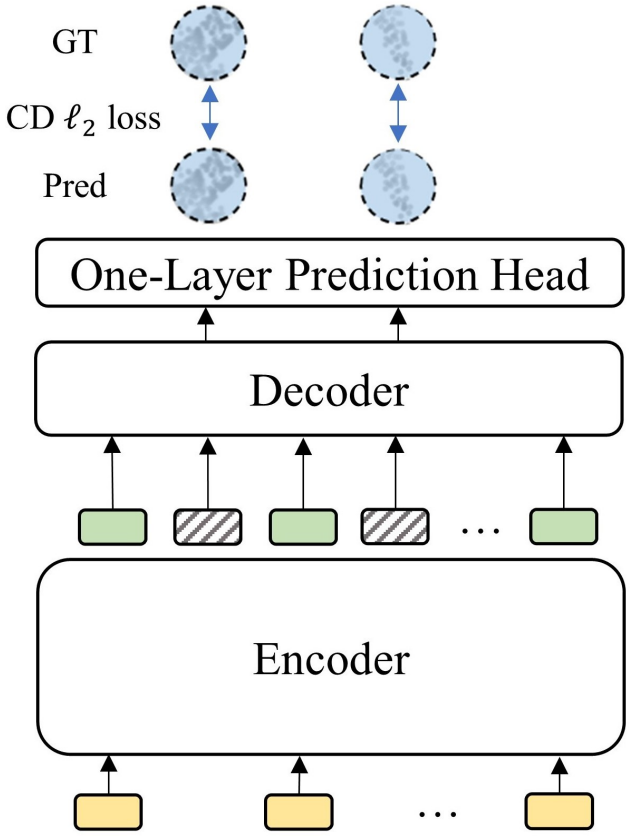


Figure 21: The Point-MAE Network Architecture (Pang et al. 2022)

International Brain Laboratory (IBL) dataset offers a high-fidelity recording environment, it remains subject to non-neural noise sources. Due to the lack of external benchmarks for noise characterization in our setting, we make the simplifying assumption that environmental noise is stationary over the course of the experiment, except for transient disturbances due to motion artifacts. A significant challenge arises from the mismatch in magnitude and dynamic range between the raw LFP signals and the behavioral outputs, complicating direct regression modeling. To mitigate this, we preprocess the LFP signals by applying a 2 Hz high-pass filter to remove low-frequency drifts, followed by z-scoring each channel independently to normalize for baseline fluctuations and channel-specific variability.

Given the high-dimensional nature of the recordings (384 channels per session), and the fact that many channels are only weakly informative for behavioral decoding, we employ an intra-channel covariance analysis to identify the most predictive subsets. Specifically, we select the top-64 channels (a tunable hyperparameter) ranked by covariance metrics, thereby improving signal quality and reducing computational complexity for downstream decoding models.

Following dimensionality reduction, the LFP data are segmented into temporal bins for feature extraction. Initially, fixed-size time bins were employed; however, this approach yielded low inter-channel variance within bins, likely due to the transient and bounded nature of the neural signals. Motivated by this observation in Fig:23, we introduced stochasticity into the binning process to better capture dynamic variations.

Specifically, we leveraged the behavioral event timestamps provided by the IBL dataset to define variable-length bin boundaries. To mitigate potential spectral leakage at bin edges, we appended a short window (100 ms) of preceding signal to each bin. This design introduces controlled randomness while maintaining temporal alignment with behaviorally relevant events.

For each constructed time bin, we extracted a comprehensive set of summary statistics from the LFP signals across all selected channels. Concretely, for each channel within a bin, we computed the mean, standard deviation, minimum amplitude, maximum amplitude, signal energy, skewness, kurtosis, zero-crossing rate, and entropy. These channel-wise features were subsequently concatenated to form a single feature vector per time bin, serving as the input for downstream decoding models.

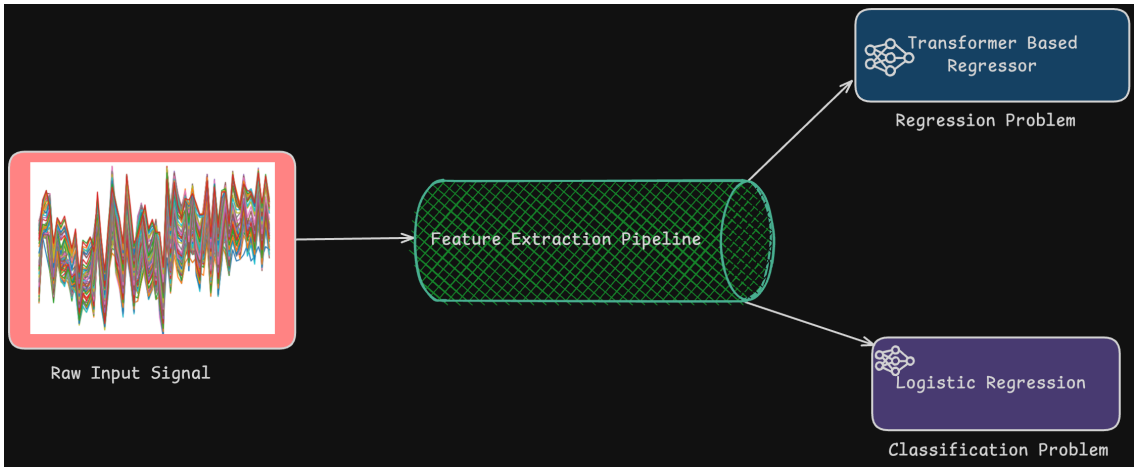


Figure 22: The Neural Behaviour Decoding Methodology

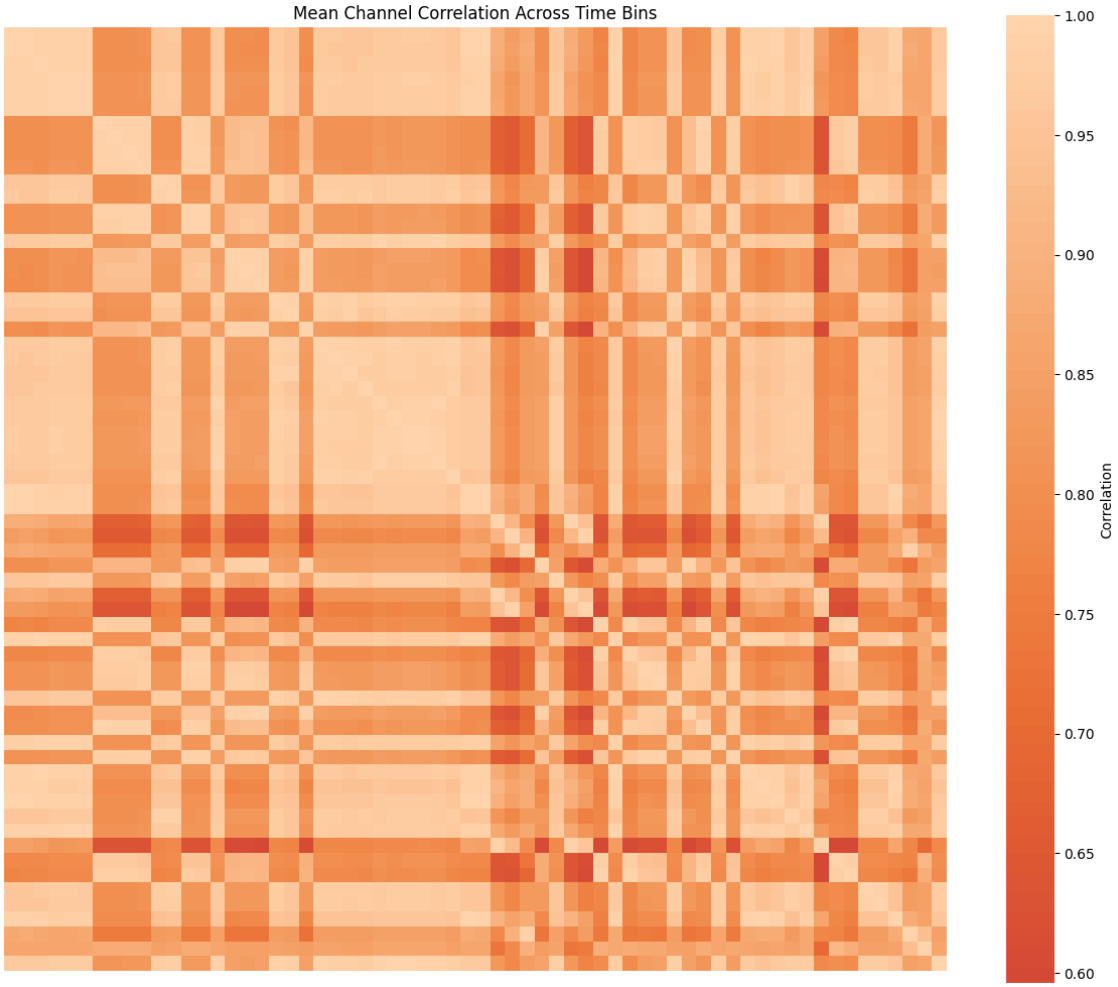


Figure 23: Mean Correlation between channels for fixed time bins

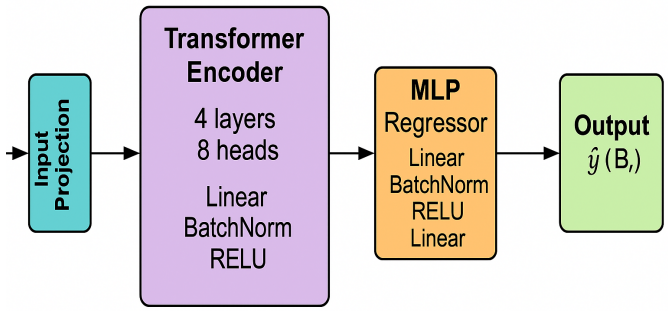


Figure 24: Transformer + MLP wheel velocity regressor

Now we model the problem statement, Given a signal can we model a representation to decode behaviour where the representation is z_t

$$z_t = f(X_t) \text{ where is the input feature}$$

$$y_t = \mathcal{P}(\frac{b_t}{z_t})$$

where b_t is the behaviour value which can be a $0, 1$ value for classification or a real number $\mathcal{R}$ for regression setting

6.4.1 Regression Setting

We propose a regression model that leverages Transformer encoders to capture complex dependencies within high-dimensional LFP-derived feature vectors. The architecture maps temporally binned neural signals to continuous behavioral outputs using an end-to-end trainable framework. The model consists of three principal stages: input projection, Transformer-based representation learning, and an MLP-based regression head.

Input Projection: Given an input feature vector $\mathbf{x}_t \in \mathbb{R}^F$ at time bin t, where F is the concatenated set of statistical features from selected LFP channels, we first apply a linear transformation to a latent embedding space of dimension $d = 256$:

$$\mathbf{z}_t = \text{ReLU}(\text{BN}(\mathbf{W}_1\mathbf{x}_t + \mathbf{b}_1))$$

This is followed by dropout ($p = 0.2$) to improve generalization. The transformed representation $\mathbf{z}_t$ is reshaped as a single-token sequence for input to the Transformer.

Transformer Encoder: To model interactions among neural features, we use a multi-layer Transformer encoder. Although Transformers are conventionally used for sequential data, we treat the single token $\mathbf{z}_t$ (reshaped to $(B, 1, d)$) as a compressed representation of time-local neural activity. The encoder consists of $L = 4$ stacked Transformer layers, each with multi-head self-attention ($H = 8$ heads) and a position-wise feedforward network:

$$\mathbf{z}_t^{(\text{enc})} = \text{Transformer}(\mathbf{z}_t)$$

The encoder outputs a contextualized embedding for each time bin, capturing latent structure in the input features.

MLP Regressor Head: The output from the Transformer encoder is passed through a multi-layer perceptron (MLP) to yield a scalar behavioral prediction $\hat{y}_t$:

$$\hat{y}_t = f_{\text{MLP}}(\mathbf{z}_t^{(\text{enc})})$$

The MLP consists of three hidden layers of sizes 256, 128, and 64, each followed by batch normalization, ReLU activation, and dropout. The final layer is a fully connected linear layer:

$$\hat{y}_t \in \mathbb{R}$$

Training Objective: We train the model using mean squared error (MSE) loss:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{N}\sum_{t=1}^{N}(y_t - \hat{y}_t)^2$$

where y_t denotes the ground-truth behavioral variable at time bin t. We optimize this loss using the Adam optimizer with weight decay for regularization.

Evaluation Metrics: To evaluate the generalization capability of the trained model, we perform 5-fold cross-validation on the held-out test set. This approach provides a robust estimate of prediction accuracy and reduces the variance associated with a single train-test split, especially in low-sample settings.

Let $\mathcal{D}_{\text{test}} = \{(\mathbf{z}_i, y_i)\}_{i=1}^{N}$ denote the test set of N samples, where $\mathbf{z}_i$ is the latent input vector and y_i is the ground-truth behavioral variable. We partition $\mathcal{D}_{\text{test}}$ into 5 equally sized folds using KFold with shuffling and fixed random seed for reproducibility. For each fold:

1. The model, pretrained on the train/validation split, is evaluated on the fold-specific test subset.
2. Predictions $\hat{y}_i$ are obtained using the frozen model.
3. The following regression metrics are computed:

$$\text{R}^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_i (y_i - \hat{y}_i)^2}$$

$$\text{MAE} = \frac{1}{N}\sum_i |y_i - \hat{y}_i|$$

The metrics are averaged across folds, and we report the mean and standard deviation of the R^2 score, along with mean RMSE and MAE.

We report the following:

- **Coefficient of Determination (R^2):** Quantifies the proportion of variance explained by the model.
- **Root Mean Squared Error (RMSE):** Captures the average magnitude of squared prediction errors.
- **Mean Absolute Error (MAE):** Measures average absolute deviation from ground truth.

Results: Table 15 reports the mean performance metrics across 5-fold cross-validation on the test set for two different recording durations. We observe that decoding accuracy, as reflected by the mean R^2 score, is higher for the 10-minute dataset compared to the 30-minute dataset. Longer recordings introduce additional variability, likely due to non-stationarities and behavioral drift, leading to increased prediction errors (higher RMSE and MAE). A scatter plot of true(in blue) and predicted(in orange) for regression values on wheel velocity are showed in Fig.25

Table 15: Test Set 5-Fold Cross-Validation Results for Different Recording Durations

Duration	Mean R^2 (± Std)	Mean RMSE	Mean MAE
10 minutes	0.1447 ± 0.0051	2.2395	1.5851
30 minutes	0.0912 ± 0.0028	3.0570	2.2074

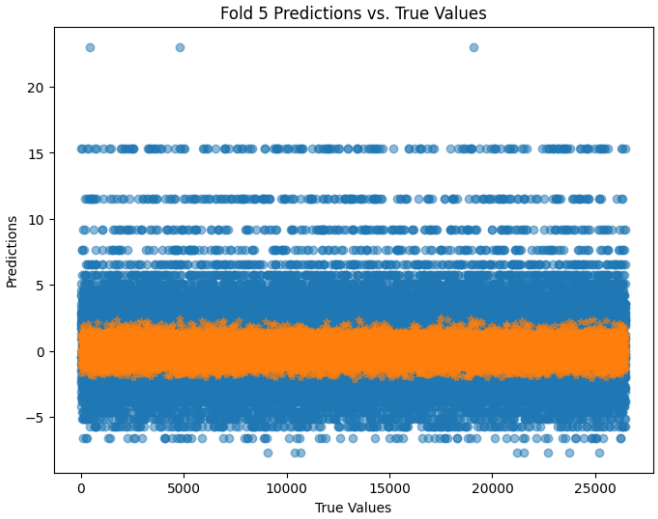


Figure 25: Regression Results for 5*th* fold

6.4.2 Classfication Problem

To decode binary choice behavior, specifically, whether the mouse moved the wheel to the left or right, we employ a logistic regression classifier. Given the limited size of the dataset (472 labeled time bins), we favor logistic regression over more complex models such as Transformers, which are prone to overfitting and instability when trained on small datasets.

We implement a logistic regression classifier to decode binary behavioral choices from LFP-derived features. The model is trained in an end-to-end fashion, mapping statistical representations of neural activity to binary action labels.

Model Architecture: Given an input feature vector $\mathbf{x}_t \in \mathbb{R}^F$ at time bin t, where F denotes the number of extracted LFP features, the model computes a two-dimensional logit vector as:

$$\mathbf{z}_t = \mathbf{W}\mathbf{x}_t + \mathbf{b}$$

where $\mathbf{W} \in \mathbb{R}^{2\times F}$ and $\mathbf{b} \in \mathbb{R}^2$ are trainable parameters.

Predicted class probabilities are obtained via the softmax function:

$$\hat{\mathbf{y}}_t = \text{softmax}(\mathbf{z}_t)$$

Training Procedure: The model is optimized using the cross-entropy loss between predicted probabilities and true labels:

$$\mathcal{L}_{\text{CE}} = -\frac{1}{N}\sum_{t=1}^{N}\sum_{c=0}^{1} y_{t,c}\log \hat{y}_{t,c}$$

where $y_{t,c}$ is the one-hot encoded ground-truth label for class c.

We train the model using the Adam optimizer with a learning rate of 10^{-3} and a weight decay regularization of 10^{-5}. The dataset is partitioned into training (70%), validation (15%), and test (15%) sets. Training is performed over 20 epochs with a batch size of 32.

Evaluation Metrics: We assess model performance using:

- Validation accuracy during training.
- 5-fold cross-validation on the held-out test set.

The following metrics are computed:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$$

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

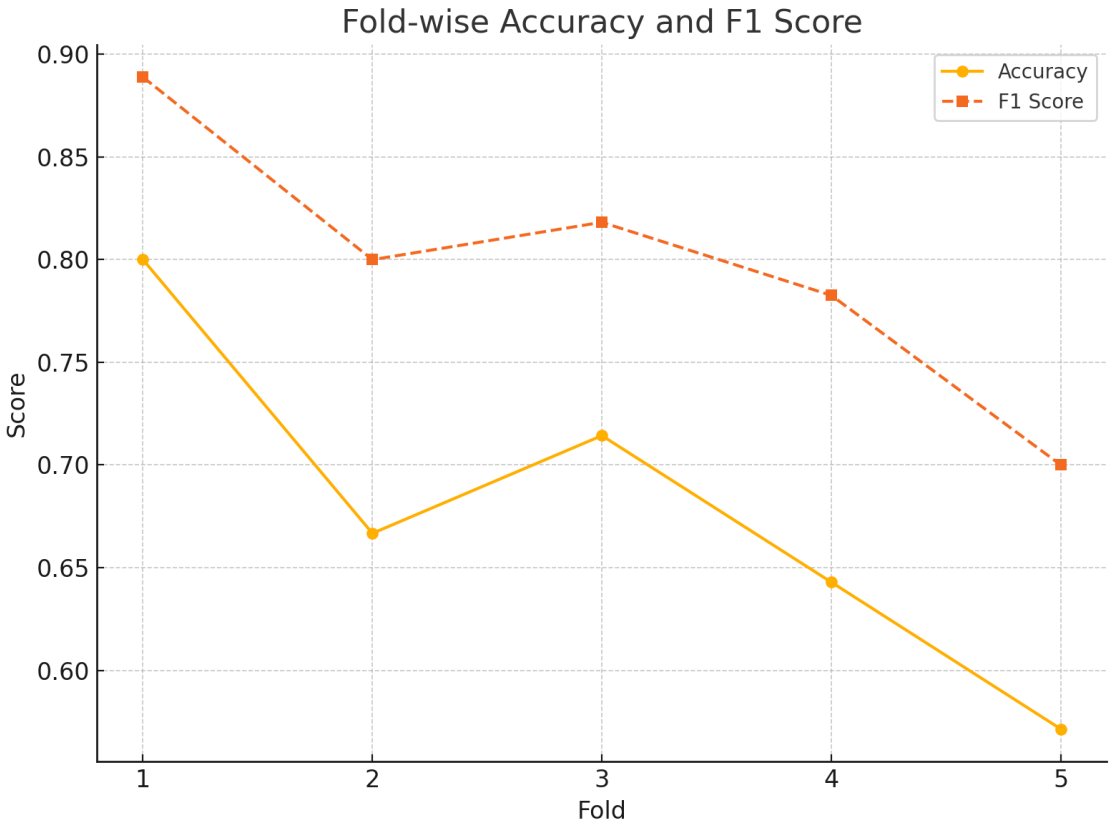


Figure 26: Plot with $F1$ and accuracy scores acroos 5 folds for classification problem

where precision and recall are calculated for the positive class (label 1).

Results: Table 16 reports the fold-wise performance of the logistic regression classifier on the held-out test set. The model achieves a mean accuracy of 67.9% and a mean F1 score of 0.7979 across 5 folds. While moderate variability in accuracy is observed across folds (standard deviation of 7.6%), the consistently high F1 scores suggest that the model reliably captures the underlying structure of the binary choice behavior. A plot is shown in Fig:26

Table 16: 5-Fold Cross-Validation Results for Binary Choice Classification on Test Set

Fold	**Accuracy**	**F1 Score**
Fold 1	0.8000	0.8889
Fold 2	0.6667	0.8000
Fold 3	0.7143	0.8182
Fold 4	0.6429	0.7826
Fold 5	0.5714	0.7000
Mean (± Std)	0.6790 ± 0.0760	0.7979

7 References

Bishop, C. M. and N. M. Nasrabadi (2006). *Pattern recognition and machine learning*. Vol. 4. 4. Springer.

Boussard, J. et al. (2021). "Three-dimensional spike localization and improved motion correction for Neuropixels recordings". In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato et al. Vol. 34. Curran Associates, Inc., pp. 22095–22105. URL: https://proceedings.neurips.cc/paper_files/paper/2021/file/b950ea26ca12daae142bd74dba4427c8-Paper.pdf.

Boussard, J. et al. (2023). “DARTsort: A modular drift tracking spike sorter for high-density multi-electrode probes”. In: *bioRxiv*. DOI: 10.1101/2023.08.11.553023. eprint: https://www.biorxiv.org/content/early/2023/08/14/2023.08.11.553023.full.pdf. URL: https://www.biorxiv.org/content/early/2023/08/14/2023.08.11.553023.

Buccino, A. P., S. Garcia, and P. Yger (2022). “Spike sorting: new trends and challenges of the era of high-density probes”. In: *Progress in Biomedical Engineering* 4.2, p. 022005. DOI: 10.1088/2516-1091/ac6b96. URL: https://dx.doi.org/10.1088/2516-1091/ac6b96.

Glaser, J. I. et al. (2020). “Machine learning for neural decoding”. In: *Eneuro* 7.4.

Hilgen, G. et al. (2017). “Unsupervised Spike Sorting for Large-Scale, High-Density Multielectrode Arrays”. In: *Cell Reports* 18.10, pp. 2521–2532. ISSN: 2211-1247. DOI: https://doi.org/10.1016/j.celrep.2017.02.038. URL: https://www.sciencedirect.com/science/article/pii/S221112471730236X.

IBL (May 2022). “Spike sorting pipeline for the International Brain Laboratory”. In: DOI: 10.6084/m9.figshare.19705522.v4. URL: https://figshare.com/articles/online_resource/Spike_sorting_pipeline_for_the_International_Brain_Laboratory/19705522.

IBL Banga, K. et al. (2022). “Reproducibility of in-vivo electrophysiological measurements in mice”. In: *bioRxiv*, pp. 2022–05.

Kucukelbir, A. et al. (2017). “Automatic Differentiation Variational Inference”. In: *Journal of Machine Learning Research* 18.14, pp. 1–45. URL: http://jmlr.org/papers/v18/16-107.html.

Pachitariu, M. et al. (2016). “Kilosort: realtime spike-sorting for extracellular electrophysiology with hundreds of channels”. In: *BioRxiv*, p. 061481.

Pan, B. (2023). “Clusterless Decoding of Mouse Choices Based on Point-wise Neural Network”. In: URL: https://james-bole-pan.github.io/file/clusterless.decoding.pdf.

Pang, Y. et al. (2022). “Masked autoencoders for point cloud self-supervised learning”. In: *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part II*. Springer, pp. 604–621.

Steinmetz, N. A. et al. (2021). “Neuropixels 2.0: A miniaturized high-density probe for stable, long-term brain recordings”. In: *Science* 372.6539, eabf4588. DOI: 10.1126/science.abf4588. URL: https://www.science.org/doi/abs/10.1126/science.abf4588.

Ventura, V. (Apr. 2008). “Spike Train Decoding Without Spike Sorting”. In: *Neural Computation* 20.4, pp. 923–963. ISSN: 0899-7667. DOI: 10.1162/neco.2008.02-07-478. eprint: https://direct.mit.edu/neco/article-pdf/20/4/923/817267/neco.2008.02-07-478.pdf. URL: https://doi.org/10.1162/neco.2008.02-07-478.

Windolf, C. et al. (2023). “Robust Online Multiband Drift Estimation in Electrophysiology Data”. In: *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5. DOI: 10.1109/ICASSP49357.2023.10095487.

Zhang, Y. et al. (2023). “Bypassing spike sorting: Density-based decoding using spike localization from dense multielectrode probes”. In: *Thirty-seventh Conference on Neural Information Processing Systems*. URL: https://openreview.net/forum?id=tgQRMrsxht.