UNSUPERVISED DETECTION OF CELL ASSEMBLIES WITH GRAPH NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Cell assemblies, putative units of neural computation, manifest themselves as repeating and temporally coordinated activity of neurons. However, understanding of their role in brain function is hampered by a lack of scalable methods for their unsupervised detection. We propose using a graph neural network for embedding spike data into a sequence of fixed size vectors and clustering them based on their self-similarity across time. We validate our method on synthetic data and real neural recordings.

1 INTRODUCTION

Cell assemblies (Hebb, 1949) are observed as patterns of neural activity that repeat themselves with a high degree of similarity across time. However, their intrinsic variability (inexact relative timing, deletions or additions of spikes) makes it challenging to detect them without a behavioral reference (such as the position of the animal in space). While popular methods like PCA and ICA are certainly useful, they only capture temporally correlated activity of large groups of neurons but disregard the relative order of spikes. Attempting to overcome this limitation, a variety of methods have been proposed ranging from edit similarity-based template matching (Watanabe et al., 2019) and transport-based clustering (Grossberger et al., 2018) to convolutional non-negative matrix factorization (Mackevicius et al., 2019; Williams et al., 2020; Peter et al., 2016). Unique from previous approaches, here we treat neural data as a sequence of directed weighted graphs, each of which is transformed with a graph neural network into fixed-size embeddings and clustered using a standard clustering algorithm. By representing data as graphs, our method leverages the natural sparsity of neural activity, and should scale well to very large datasets. We demonstrate the performance of the method on synthetic and real data.

2 METHOD

Data preparation. We start with a recording of N neurons for T time steps, $\mathbf{X} \in \{0,1\}^{N \times T}$, such that $X_{n,t} = 1$ if there is a spike at time t on the n-th neuron and $X_{n,t} = 0$ otherwise. This matrix is then segmented into M overlapping windows $G^{(t)} = \mathbf{X}_{:,t:t+w}, t \in \{0, k, 2k, \ldots, M\}, M = T//k$, $w = 200$ is the window length in samples and $k = 4$ is the number of time steps between two adjacent windows. Each of the windows is converted into a weighted directed graph $\mathcal{G}^{(t)}$, in which the nodes represent spikes, and the edge weights from the i-th to the j-th neuron are a function of time difference Δt between two consecutive spikes on those neurons. In this paper we calculated edge weights as $e^{-\tau \Delta t}$, with $\tau = 0.04$, to emphasize the contribution of spikes that occur in close temporal proximity. For any two spikes on neurons i and j, the edge weight is non-zero if $t_i < t_j$. If more than two spikes occur between the same pair of neurons, the weight of the edge between them is the sum of the contributions of all the spikes within that window. The nodes (neuron indices) are encoded as integers $\{0, \ldots, N-1\}$.

Model. We use a stack of 3 graph convolution layers (Kipf & Welling, 2016) with an output dimensions of 10, 10 and 6, respectively, each followed by an ELU non-linearity. Neuron indices are converted to 10-D vectors using an embedding layer. Since each window of the original data contains a varying number of spikes (hence a different number of nodes in the corresponding graph), we use a global average pooling layer on top of the GCN stack to obtain a fixed size embedding for each graph.

Training. We fit the model using the AdamW optimizer with default parameters to minimize the following loss function: $\mathcal{L}(\theta) = XE(\mathbf{Z}, \mathbf{y}) + \beta TV(\mathbf{Z})$, where $XE(\mathbf{Z}, \mathbf{y})$ is the cross-entropy loss between the embeddings and their cluster assignments (recalculated at the beginning of each epoch using K-means ($K = 6$) as in (Caron et al., 2018; Hsu et al., 2021)) and $TV(\mathbf{Z}) = \sum_{i=1}^{6} \sum_{t=1}^{M-1} (Z_{t,i} - Z_{t+1,i})^2$ is the total variation of the embeddings over time, which encourages temporal consistency of the class labels. We set β to 0.1, update the weights at the end of each epoch, and continue training until the cross-entropy loss plateaus.

3 Results

Synthetic data. To facilitate fair comparison of the model's performance on real and synthetic data, the synthetic dataset was made to match the properties of the real CA1 recording we consider in the next subsection. We embedded 3 artificial spike patterns into a binary matrix of background activity $\mathbf{X} \in \{0, 1\}^{N \times T}$ (N=452, T=18137) obtained by permuting inter-spike intervals of the CA1 dataset (ensuring that the synthetic and real datasets had approximately the same spike statistics). Each of the artificial patterns was a temporally jittered (std=10 time steps) sequence of 100 spikes: one spike per neuron per time step. The patterns were pruned by dropping spikes with a probability of 0.2.

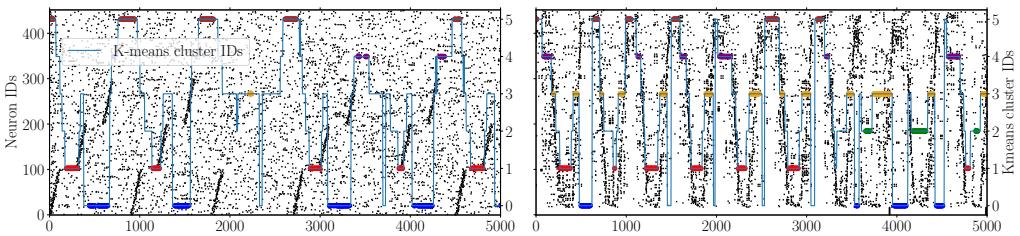

Figure 1: The model successfully disentangles repeating patterns in both the synthetic (left) and real (right) datasets. Colored dots indicate significant detections, i.e. where the graph embeddings are significantly similar across time (see Appendix A). We sorted the neurons to reveal the patterns, but this is not required for the method to work.

Real data. We used data from (Rubin et al., 2019) recorded from the area CA1 of a mouse running on a linear track to collect water rewards. With the same hyperparameters as for the synthetic data, the model was able to distinguish sequential activity of place cells (Fig 1, right) with similar spatial tuning (although with a lower accuracy, which is likely due to the animal's non-constant speed on the track).

4 Conclusions and Future Directions

We proposed a method for unsupervised detection of patterned neural activity, which represents spike data as a sequence of directed weighted graphs. Given the natural sparsity of cortical neurons, this approach might offer superior performance and enhanced sensitivity compared to other methods. However, establishing whether that is the case requires rigorous benchmarking, which we leave to future work.

URM Statement

The authors acknowledge that at least one of the authors of this work meets the URM criteria of ICLR 2023 Tiny Papers Track.

References

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 132–149, 2018.

Lukas Grossberger, Francesco P Battaglia, and Martin Vinck. Unsupervised clustering of temporal patterns in high-dimensional neuronal ensembles using a novel dissimilarity measure. *PLoS computational biology*, 14(7):e1006283, 2018.

Donald O Hebb. The first stage of perception: growth of the assembly. *The Organization of Behavior*, 4(60):78–60, 1949.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460, 2021.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Emily L Mackevicius, Andrew H Bahle, Alex H Williams, Shijie Gu, Natalia I Denisenko, Mark S Goldman, and Michale S Fee. Unsupervised discovery of temporal sequences in high-dimensional datasets, with applications to neuroscience. *Elife*, 8:e38471, 2019.

Sven Peter, Daniel Durstewitz, Ferran Diego, and Fred A Hamprecht. Sparse convolutional coding for neuronal ensemble identification. *arXiv preprint arXiv:1606.07029*, 2016.

Alon Rubin, Liron Sheintuch, Noa Brande-Eilat, Or Pinchasof, Yoav Rechavi, Nitzan Geva, and Yaniv Ziv. Revealing neural correlates of behavior without behavioral measurements. *Nature communications*, 10(1):4745, 2019.

Keita Watanabe, Tatsuya Haga, Masami Tatsuno, David R Euston, and Tomoki Fukai. Unsupervised detection of cell-assembly sequences by similarity-based clustering. *Frontiers in Neuroinformatics*, pp. 39, 2019.

Alex Williams, Anthony Degleris, Yixin Wang, and Scott Linderman. Point process models for sequence detection in high-dimensional neural spike trains. *Advances in neural information processing systems*, 33:14350–14361, 2020.

A APPENDIX

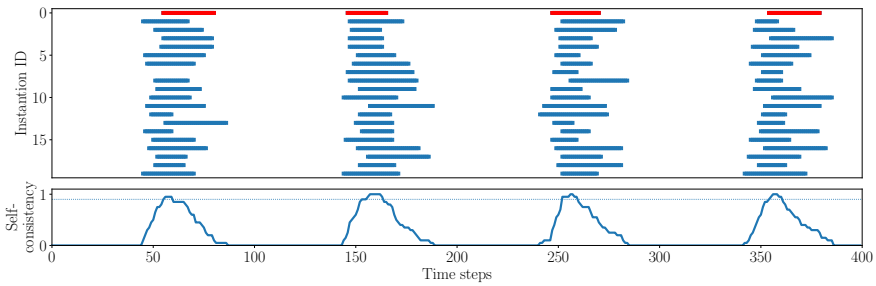

Figure 2: Illustration of the procedure to test pattern significance. Top: binary matrix each row of which indicates the time steps at which the model assigned the corresponding graph to the cluster that happened to have the highest similarity with the reference (top row, red). The average over the rows of that matrix (bottom) provides a measure of self-consistency of the reference pattern. Dotted line marks the significance criterion.

Statistical testing. We consider a pattern statistically significant if it is self-consistent across 90 % of model instantiations, each of which is initialized with the same hyperparameters, but different weights and cluster centroids, and trained for a fixed number of epochs on the same data (Figure 2). Conveniently, the model's relatively small size (4842 parameters) enables training dozens of its instantiations in parallel. The self-consistency criterion we enforce in this paper (90 %) can be relaxed depending on the nature of data and the level of confidence desired. Since for different

instantiations cluster indices might be different, we first select one "reference" cluster (e. g. with index 0 in the zeroth instantiation) and record the time steps at which the model predicted the zeroth cluster. From the predictions of the other instantiations we select one cluster that has the highest similarity with the reference cluster and record its corresponding time steps. This process produces a binary matrix, whose average over the rows reflects the self-consistency of the reference pattern. If at least four peaks of the self-consistency curve exceed the criterion, we consider the reference pattern to be significant.