A NEURAL NETWORK-BASED SPIKE SORTING FEATURE MAP THAT RESOLVES SPIKE OVERLAP IN THE FEATURE SPACE

Jasper Wouters[1] *Fabian Kloosterman*[2,3,4] *Alexander Bertrand*[1]

[1] KU Leuven, Electrical Engineering Dept. (ESAT),
Stadius Center for Dynamical Systems, Signal Processing, and Data Analytics, Belgium
[2] Neuro-Electronics Research Flanders (NERF), Leuven, Belgium
[3] KU Leuven, Brain & Cognition Research Unit, Belgium [4] VIB, Leuven, Belgium

ABSTRACT

When inserting an electrode array in the brain, its electrodes will record so-called 'spikes' which are generated by the neurons in the neighbourhood of the array. Spike sorting is the process of detecting and assigning these recorded spikes to their putative neurons. Many spike sorting pipelines rely on a clustering algorithm that groups the spikes coming from the same neuron in a pre-defined feature space. However, classical spike sorting algorithms fail when spike overlap, i.e., the near-simultaneous occurrence of two or more spikes from different neurons, is present in the recording. In such cases, the overlapping spikes segment ends up in a seemingly random position in the feature space and is not assigned to the correct cluster. This problem has been addressed before by extending the sorting algorithm with a template matching post-processor. In this work, a novel approach is presented to resolve spike overlap directly in the feature space. To this end, a neural network feature map is presented, that generates spike embeddings (feature vectors) that behave as a linear superposition in the feature space in the case of spike overlap. Its performance is quantified on semi-synthetic data obtained through a data augmentation procedure applied to real neural recordings.

Index Terms— spike sorting, spike overlap, neural network, feature extraction

1. INTRODUCTION

When studying the brain, an often used method to explore brain activity is the acquisition and processing of extracellular recordings. Extracellular recordings can be obtained from measurement electrodes that are lowered into the brain. If the implanted electrodes are in close proximity to some neurons, i.e., a certain type of brain cells that can communicate with each other through electrical-chemical signalling, the electrodes can pick up voltage patterns that occur when a neuron is actively signalling other neurons. This voltage pattern co-occurs with the so-called intracellular action potential and is referred to as a spike, due to its spiky waveform. To get a better understanding of micro-scale brain network dynamics, extracellular recordings are often processed to extract the spike times of individual neurons from which the spikes were recorded. This process of detecting and assigning spikes to their putative neurons is known as *spike sorting* [1].

Spike sorting is an unsupervised machine learning problem that is based on the assumption that all spikes from the same neuron exhibit the same spatio-temporal waveform, and that every neuron has a unique spatio-temporal spike pattern when compared to other neurons. The spike sorting problem is typically solved through the use of a classical clustering pipeline [2]. This pipeline roughly consists of a preprocessing, spike detection, feature extraction, and clustering phase. The preprocessing block minimally consists of a high-pass filter, that filters out the low-frequency energy that is unrelated to the spiking activity. The spike detection block then selects only recording segments that have spiking activity. Next, a feature vector is calculated for every spike segment, such that finally the clustering algorithm can isolate dense clusters of spikes that are believed to belong to the same neuron.

Such a clustering approach typically suffers from the overlapping spikes problem [3]. This problem arises when a detected spike segment consists of the activity of multiple neurons. Typically, such segments containing overlapping spikes will lead to points in the feature space, that are unrelated to the clusters that represent the well isolated spikes of the neurons that contribute to the overlapping spikes segment. They appear at seemingly random positions in the feature space, depending on the amount of overlap and the relative spike times of both neurons. It is believed that spike overlap can not be resolved in the feature space [4]. As such, the classical pipeline has been augmented with a template matching post-processing step in many spike sorting implementations [5] [6] [7] [8] [9]. Alternatively, iterative model-based algorithms [3] [10] [11] have been developed to cope with spike overlap. Typically, in such extended pipelines, first the clustering approach is taken, after which heuristics are used to identify clusters that contain the activity of individual neurons. Next, spike templates are estimated from those single-neuron clusters. Those templates are then used to design matched or discriminative filters that have the capability to resolve overlapping spikes. Augmenting the classical pipeline in such a way, leads to a considerable increase in complexity.

This work was carried out at the ESAT Laboratory of KU Leuven, in the frame of KU Leuven Special Research Fund projects C14/16/057, and the Research Foundation Flanders (FWO) project FWO G0D7516N. This project has received funding from the European Research Council (ERC) under the European Unions Horizon 2020 research and innovation programme (grant agreement No 802895). The scientific responsibility is assumed by its authors.

In this work we present a neural network-based feature map that, contrary to popular believe, is capable of resolving spike overlap directly in the feature space, hence, resulting in a pipeline with overall reduced computational complexity. In Section 2 the overlapping spikes problem is formalized. Section 3 presents the neural network feature map capable of resolving overlapping spikes. In Section 4 a data augmentation procedure that is used for the generation of training and testing data is presented. Section 5 summarizes the experimental results. Finally, Section 6 concludes this work by discussing the proposed method.

2. PROBLEM STATEMENT

Consider the high-pass filtered extracellular multi-channel brain recording $\mathbf{x}[k] \in \mathbb{R}^N$ at sample time k with N the number of recording sites of interest. Typically, only compact subsets of recording sites are processed simultaneously in spike sorting because the neuronal spike is a spatially compact pattern. For modern probes $N \ll N_{\text{tot}}$, where N_{tot} is the total number of recording sites of the recording device. A spatio-temporal recording snippet of duration L containing an entire neuronal spike waveform can be represented as an NL-dimensional vector $\mathbf{y}[k] = \left[\mathbf{x}[k]^T \ \mathbf{x}[k-1]^T \ \dots \ \mathbf{x}[k-L+1]^T\right]^T$, which captures a spike waveform between sample time $k-L+1$ and k. After spike detection, we store the values of $\mathbf{y}[k]$ at those sample indices k that correspond to a spiking event, where k is selected such that the largest peak of each spiking event is aligned across the extracted snippets $\mathbf{y}[k]$. This results in a set of extracted snippets represented as NL-dimensional vectors $\{\mathbf{y}_1 \dots \mathbf{y}_M\}$, each containing an aligned spatio-temporal representation of a spike waveform. Since each neuron has a signature spatio-temporal waveform, the snippets $\mathbf{y}$ that capture spikes from the same neuron i will be close to each other in the $\mathbb{R}^{NL}$ space and their average can be viewed as a spike template for neuron i. In the remaining of this paper we will mostly drop the time index k in $\mathbf{y}[k]$, except when the sample time is providing clarity.

During spike sorting, the aligned snippets are mapped to a lower-dimensional feature space to accommodate the clustering stage. Consider the feature extraction $f : \mathbb{R}^{NL} \rightarrow \mathbb{R}^P$, with P the dimensionality of the feature space. In many spike sorting pipelines f is a linear function, e.g., based on principal component analysis on all aligned spike snippets. The implication of such a linear feature map is that $f(\mathbf{s}_i + \mathbf{s}_j) = f(\mathbf{s}_i) + f(\mathbf{s}_j)$ for any $\mathbf{s}_i, \mathbf{s}_j \in \mathbb{R}^{NL}$. Let $\mathbf{s}_i$ and $\mathbf{s}_j$ denote the spike waveform of neuron i and neuron j, respectively. When using a linear feature map, it is expected that overlapping spike snippets of the form $\mathbf{y} = \mathbf{s}_i + \mathbf{s}_j$ would form a new cluster of which the centroid is equal to the sum of the centroids of the clusters associated to neuron i and j (cfr. the red, blue and purple cluster in the right panel of Fig. 1). If this were to be the case, it would be straightforward to resolve spike overlap. Unfortunately, this is not the case in practice as this requires the main peaks of the spikes $\mathbf{s}_i$ and $\mathbf{s}_j$ to be perfectly aligned in time (remember that all snippets are aligned through their maximum peak value).

In reality, spike overlap can be better modelled as $\mathbf{y}[k] = \mathbf{s}_i[k-m] + \mathbf{s}_j[k-n]$ with $m, n \in \mathbb{Z}$ being probabilistic sample offsets, and where $\mathbf{s}_i[k]$ and $\mathbf{s}_j[k]$ correspond to the *mutually aligned* isolated spike snippets of neuron i and j, respectively. Next we introduce the operator $\oplus$ to denote a sum with probabilistic sample offsets, which abstracts away the actual values for m and n. The shifted sum can now be rewritten as

$$\mathbf{s}_i \oplus \mathbf{s}_j = \mathbf{s}_i[k-m] + \mathbf{s}_j[k-n]. \tag{1}$$

Because of the introduction of the time shift, $f(\mathbf{s}_i \oplus \mathbf{s}_j) \neq f(\mathbf{s}_i) + f(\mathbf{s}_j)$ for the linear feature map f, if at least one of the two sample offset is non-zero, thereby rendering the proposed approach for the identification of overlapping spikes clusters in the feature space useless as shown in the left panel of Fig. 1. In this work we aim to design a non-linear feature map g for which $g(\mathbf{s}_i \oplus \mathbf{s}_j) \approx g(\mathbf{s}_i) + g(\mathbf{s}_j)$, i.e., the linearity property is preserved despite the random sample offsets in the overlapping spike. Such a feature map g would be capable of resolving spike overlap, because the relation between clusters can be derived by simply adding pairs of cluster centroids, as is shown for the cluster centroids of neuron one and neuron two in the right panel of Fig. 1, where those cluster centroids sum up to the cluster containing randomly shifted overlapping spikes of neuron one and neuron two.

3. METHOD

To obtain the desired feature behaviour, we train a neural network using the following custom cost function:

$$\sum_{\substack{i,j \in \mathcal{T} \\ i \neq j}} \left[\|g(\mathbf{s}_i) + g(\mathbf{s}_j) - g(\mathbf{s}_i \oplus \mathbf{s}_j)\|_2^2 + \frac{\kappa}{\|g(\mathbf{s}_i) - g(\mathbf{s}_j)\|_2^2} \right], \tag{2}$$

where the sum is taken over all pairs of spikes (both corresponding to different neurons) that are in the training set $\mathcal{T}$. When minimizing the cost function over g, the first term in the cost function will decrease when the feature acts as if it is a linear operator applied to a linear combination, i.e., the behaviour that is desired to resolve overlap in the feature space. The second term is added to try to increase the distance between spikes from different neurons, and hence their separability, in the feature space and this will as well prevent the network from converging to the trivial solution during training. The hyperparameter κ associated with the second term can be altered to trade off the desired linearity with the cluster separability.

Notice that the cost function in (2) does not explicitly control for the intra-class variability, which is an important characteristic for a feature map intended for clustering. In this work the intra-class variability is controlled through on the one hand limiting the degrees of freedom by choosing an appropriate neural network architecture and on the other hand by applying regularization during the training phase. In this work early stopping regularization is used [12], where thirty percent of the training data is used as a validation set for this purpose. If the validation cost does not sufficiently decrease over a certain number of epochs, the training process is stopped to prevent the network from overfitting on the training data.

In Fig. 2 the neural network architecture that is used in the scope of this work is shown. The architecture consists of a multi-layer perceptron with two hidden layers. The

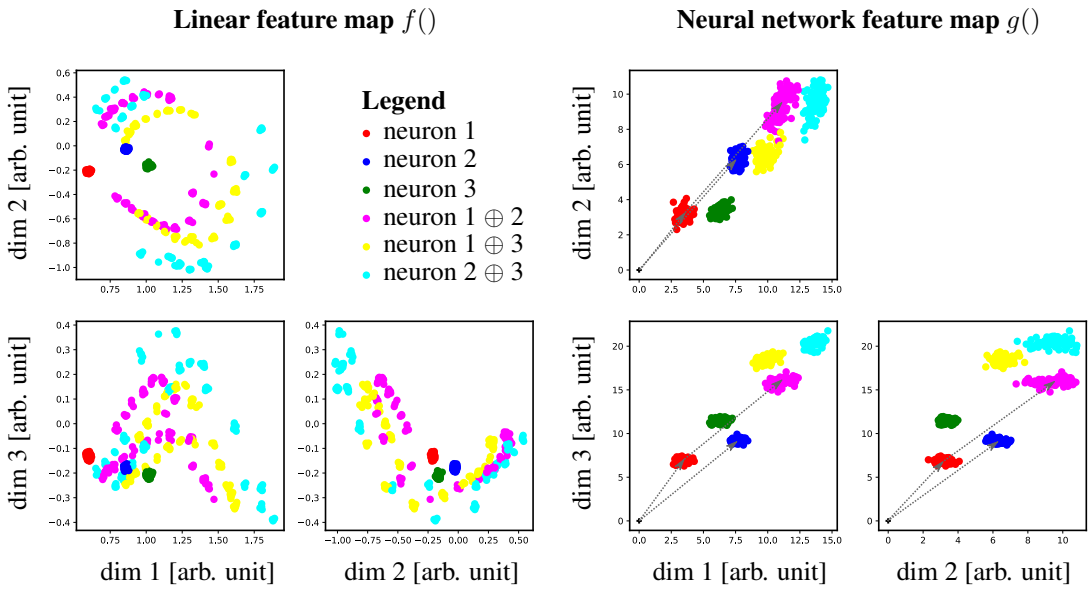

Fig. 1. 3 orthogonal viewpoints on a 3-dimensional feature space with 3 neurons. **Left**: Overlapping spikes in a (linear) principal component feature map do not form separable clusters and do not have a clear relationship with the single-neuron clusters. **Right**: The proposed neural network feature map is capable of resolving overlapping spikes.

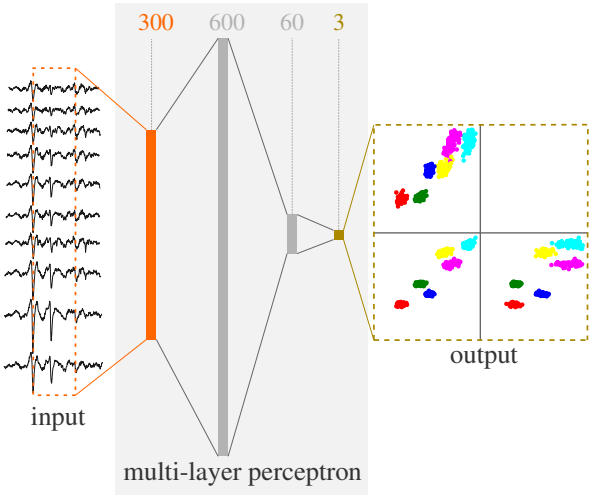

Fig. 2. The proposed neural network feature map is implemented as a multi-layer perceptron with two hidden layers.

network takes an NL-dimensional[1] snippet $\mathbf{y}$ as input, with $N = 10$ and $L = 30$. The first and second hidden layer consist of 600 and 60 hidden perceptrons, respectively. The network outputs a three dimensional feature vector for every data chunk at its input. All layers use a rectifier activation function.

[1]The choice of N and L is related to the spatial resolution of the electrode array and the sampling frequency, respectively. The dimensions of the proposed network were designed for a neural probe with an electrode pitch of 20 µm and a sampling frequency of 30 kHz.

4. DATA AUGMENTATION

The training and testing data used in this work are obtained through a data augmentation procedure similar to [13]. First, 131 single-neuron spike templates are calculated from a publicly available in-vivo Neuropixels [14] brain recording data set that has been spike sorted [15]. For each of the 131 neurons, a spatio-temporal spike template is calculated by averaging over the spike snippets of that neuron where a neuron-specific spatial region of ten neighbouring channels is selected. Next, the single-neuron templates are randomly split into two sets, one containing 88 templates used for generating the training data and 43 used for generating the testing data. For both the set of training and testing templates separately, principal components are derived that account for 99 percent of the template energy. Both sets are then projected onto their respective principal component subspaces. Along each principal component an independent skew normal distribution [16] is fitted. Finally, synthetic training and testing spike templates are constructed by sampling from the estimated distributions in each principal component. The templates obtained through data augmentation have a duration of 82 samples.

5. EXPERIMENTS

From the training distributions obtained in the previous Section, 50000 different pairs of neuronal spikes are generated to form the training set $\mathcal{T}$, i.e, the summation in (2) runs over 50000 training samples. For each pair (i, j), 3 snippets are created: $\mathbf{s}_i$, $\mathbf{s}_j$ and $\mathbf{s}_i \oplus \mathbf{s}_j$. $\mathbf{s}_i$ and $\mathbf{s}_j$ are created by adding random white noise to the two synthetic spikes, aligning both of them according to their largest peak, and cropping them in the time dimension to the desired window size of $L = 30$ samples. $\mathbf{s}_i \oplus \mathbf{s}_j$ is obtained by randomly time-shifting the template of neuron i with respect to the template of neuron j,

after which white noise is added and the summed waveforms are aligned based on the largest peak of the mixed waveform and cropped to $L = 30$ samples. The relative shift is sampled from a discrete uniform distribution where the minimum and maximum shift are -5 and 5, respectively. All synthetic snippets have a signal-to-noise ratio of 20 dB. The network presented in Section 3 is trained using the Adam optimization algorithm [17]. The hyperparameter $\kappa = 10$ is chosen through a grid-search, but the performance on the training data is not very sensitive to the exact choice of κ, even when considering differences of several orders of magnitude.

After training, the performance on the test set of the neural network feature is quantified in terms of two metrics: 1. the adjusted Rand index [18], which is a cluster performance metric that is equal to one for perfect clustering and zero when the clustering performs equal to chance, and 2. the center prediction error

$$e_{i,j} = \frac{\|\mathbf{c}_{i\oplus j} - (\mathbf{c}_i + \mathbf{c}_j)\|_2}{\frac{1}{3}(\sigma_{i\oplus j} + \sigma_i + \sigma_j)}, \tag{3}$$

i.e., the distance in the feature space between the true overlapping spikes cluster centroid $\mathbf{c}_{i\oplus j} \in \mathbb{R}^P$ and the predicted centroid by taking the sum of the isolated cluster centroids $\mathbf{c}_i + \mathbf{c}_j$, normalized with respect to the intra-class variabilities $\sigma_{i\oplus j}$, σ_i and σ_j, which are the average distances between points in the feature space and their corresponding centroid.

The test data contains 500 pairs of synthetic test templates. For every pair of test templates two sets of 100 isolated spikes are derived by adding white noise to the templates and 100 snippets containing spike overlap with a random offset similar to the offset used in the training set are generated. The synthetic training snippets also have a signal-to-noise ratio of 20 dB. The neural network feature map is then applied to every group of 300 snippets and a k-means clustering [19] with k = 3 is applied for all 500 test pairs. The performance metrics for all test cases are summarized in Fig. 3. Because 483 pairs have an adjusted Rand index equal to one, random jitter is added to the horizontal axis to make the graph more readable (hence, explaining test pairs with a depicted Rand index slightly bigger than one). As one can see in Fig. 3 the majority of the test pairs is situated in the bottom right corner. For those test pairs, unseen during training, the neural network feature map generalizes very well. The feature allows for near perfect clustering for those pairs, even using a very simple clustering method such as k-means. Secondly, only a small center prediction error is made, indicating the feature's capability to resolve spike overlap in the feature space.

In Fig. 3 we also highlighted three groups of pairs for which the feature map performs suboptimal. For the pairs in *group 1* the single-neuron clusters are very close to each other in the feature space, combined with the fact that the cluster containing all overlapping spikes is elongated. Because of this, the single-neuron clusters are grouped together and the overlapping spikes cluster is split in two by the clustering algorithm, hence explaining both the poor Rand index and center prediction error for those pairs. This issue might be resolved by using a more advanced clustering approach, e.g., a density based clustering algorithm [20]. For the pairs in *group 2*, one of the single-neuron clusters is close to the overlapping spikes cluster, as such, spikes belonging to the overlapping spikes cluster are assigned to that single-neuron cluster by the clustering algorithm. Because only a minor fraction of pairs is wrongly assigned, the center prediction error is still within acceptable bounds. This issue might also be resolved by using an alternative clustering algorithm. The pairs in *group 3* show perfect clustering, but a big center prediction error. For this small fraction of pairs, the spikes of one of the neurons are not capable of fully exciting the neural network. This behaviour is characterized in the feature space by having one of the single-neuron clusters with at least one dimension equal to zero. For the dimensions in the feature space that are not properly excited, the desired linear behaviour is not obtained. This issue might be alleviated by extending the training set with more examples.

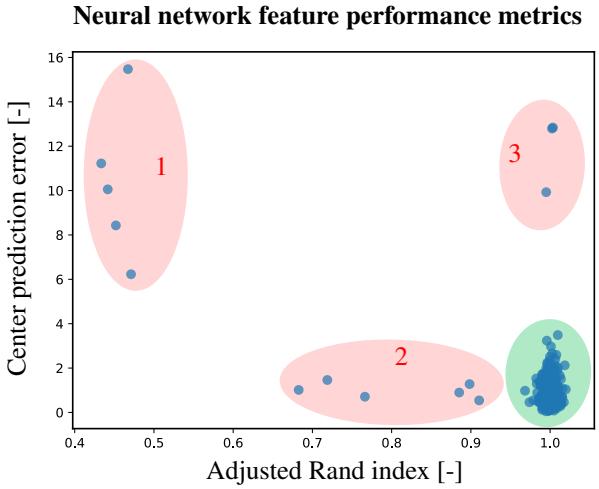

Fig. 3. On the horizontal axis the adjusted Rand index (with random jitter for readability) is depicted and on the vertical axis the center prediction error is shown. Every blue dot represents the feature map performance for a pair of test neurons.

6. CONCLUSION AND DISCUSSION

In this work we have presented a novel approach to resolving spike overlap in the feature space through the design of a neural network feature map. We have shown that the desired neural network behaviour for resolving spike overlap generalizes well on most of the testing data. On a small fraction of the testing pairs, the feature map performed suboptimal. For those pairs we have studied their failure mechanisms, and proposed solutions if possible. Since the feature map is trained using a data augmentation procedure that uses real spikes from a specific type of probe, this feature map is believed to be probe dependent. Further research is necessary to assess how well this feature map works on real recordings using the same probe, and whether it can be used with different recording probes as well. When compared to template matching based approaches, our method is limited in terms of resolving the exact spike times of the individual neurons that fire near-simultaneously. For experiments where this limitation is acceptable, a much simpler spike sorting pipeline, capable of resolving spike overlap, is given in return. Combining the proposed feature map with an adaptive clustering algorithm is a promising and potentially powerful approach to real-time spike sorting.

7. REFERENCES

[1] Michael S Lewicki, "A review of methods for spike sorting: the detection and classification of neural action potentials," *Network: Computation in Neural Systems*, vol. 9, no. 4, pp. R53–R78, 1998.

[2] Sarah Gibson, Jack W Judy, and Dejan Marković, "Spike sorting: The first step in decoding the brain," *IEEE Signal processing magazine*, vol. 29, no. 1, pp. 124–143, 2011.

[3] Jonathan W Pillow, Jonathon Shlens, EJ Chichilnisky, and Eero P Simoncelli, "A model-based spike sorting algorithm for removing correlation artifacts in multi-neuron recordings," *PloS one*, vol. 8, no. 5, pp. e62123, 2013.

[4] David Carlson and Lawrence Carin, "Continuing progress of spike sorting in the era of big data," *Current opinion in neurobiology*, vol. 55, pp. 90–96, 2019.

[5] Felix Franke, Michal Natora, Clemens Boucsein, Matthias HJ Munk, and Klaus Obermayer, "An online spike detection and spike classification algorithm capable of instantaneous resolution of overlapping spikes," *Journal of computational neuroscience*, vol. 29, no. 1-2, pp. 127–148, 2010.

[6] Felix Franke, Rodrigo Quian Quiroga, Andreas Hierlemann, and Klaus Obermayer, "Bayes optimal template matching for spike sorting–combining fisher discriminant analysis with optimal filtering," *Journal of computational neuroscience*, vol. 38, no. 3, pp. 439–459, 2015.

[7] Pierre Yger, Giulia LB Spampinato, Elric Esposito, Baptiste Lefebvre, Stéphane Deny, Christophe Gardella, Marcel Stimberg, Florian Jetter, Guenther Zeck, Serge Picaud, et al., "A spike sorting toolbox for up to thousands of electrodes validated with ground truth recordings in vitro and in vivo," *Elife*, vol. 7, pp. e34518, 2018.

[8] Jasper Wouters, Fabian Kloosterman, and Alexander Bertrand, "Towards online spike sorting for high-density neural probes using discriminative template matching with suppression of interfering spikes," *Journal of neural engineering*, vol. 15, no. 5, pp. 056005, 2018.

[9] Jasper Wouters, Fabian Kloosterman, and Alexander Bertrand, "Signal-to-peak-interference ratio maximization with automatic interference weighting for threshold-based spike sorting of high-density neural probe data," in *2019 9th International IEEE/EMBS Conference on Neural Engineering (NER)*. IEEE, 2019, pp. 247–250.

[10] Chaitanya Ekanadham, Daniel Tranchina, and Eero P Simoncelli, "A unified framework and method for automatic neural spike identification," *Journal of neuroscience methods*, vol. 222, pp. 47–55, 2014.

[11] Marius Pachitariu, Nicholas Steinmetz, Shabnam Kadir, Matteo Carandini, and Kenneth D Harris, "Kilosort: realtime spike-sorting for extracellular electrophysiology with hundreds of channels," *BioRxiv*, p. 061481, 2016.

[12] Rich Caruana, Steve Lawrence, and C Lee Giles, "Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping," in *Advances in neural information processing systems*, 2001, pp. 402–408.

[13] Jin Hyung Lee, David E Carlson, Hooshmand Shokri Razaghi, Weichi Yao, Georges A Goetz, Espen Hagen, Eleanor Batty, EJ Chichilnisky, Gaute T Einevoll, and Liam Paninski, "Yass: Yet another spike sorter," in *Advances in neural information processing systems*, 2017, pp. 4002–4012.

[14] James J Jun, Nicholas A Steinmetz, Joshua H Siegle, Daniel J Denman, Marius Bauza, Brian Barbarits, Albert K Lee, Costas A Anastassiou, Alexandru Andrei, Çağatay Aydın, et al., "Fully integrated silicon probes for high-density recording of neural activity," *Nature*, vol. 551, no. 7679, pp. 232, 2017.

[15] Nick Steinmetz, Matteo Carandini, and Kenneth D. Harris, ""Single Phase3" and "Dual Phase3" Neuropixels Datasets," 3 2019.

[16] Adelchi Azzalini, *The skew-normal and related families*, vol. 3, Cambridge University Press, 2013.

[17] Diederik P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[18] William M Rand, "Objective criteria for the evaluation of clustering methods," *Journal of the American Statistical association*, vol. 66, no. 336, pp. 846–850, 1971.

[19] David Arthur and Sergei Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.

[20] Alex Rodriguez and Alessandro Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, 2014.