A Simple Deep Learning Method for Neuronal Spike Sorting

To cite this article: Kai Yang et al 2017 J. Phys.: Conf. Ser. 910 012062

View the article online for updates and enhancements.

Related content
- Some Critical Questions in Biological Physics: Spike trains and the senses—the mouse’s whiskers
  T A Waigh
- Towards online spike sorting for high-density neural probes using discriminative template matching with suppression of interfering spikes
  Jasper Wouters, Fabian Kloosterman and Alexander Bertrand
- BEACH 2016: an Overview (of sorts)
  P D Rubin
A Simple Deep Learning Method for Neuronal Spike Sorting

Kai Yang\textsuperscript{1,1a}, Haifeng Wu\textsuperscript{1,2,b} and Yu Zeng\textsuperscript{1,2,c}

\textsuperscript{1}School of Electrical and Information Technology, Yunnan Minzu University
\textsuperscript{2}Key Laboratory of Wireless Sensor Networks of Yunnan Province, Kunming, China
Email: \textsuperscript{a}250145249@qq.com, \textsuperscript{b}whf5469@gmail.com, \textsuperscript{c}yv.zeng@gmail.com

Abstract. Spike sorting is one of the key techniques to understand brain activity. With the development of modern electrophysiology technology, some recent multi-electrode technologies have been able to record the activity of thousands of neuronal spikes simultaneously. The spike sorting in this case will increase the computational complexity of conventional sorting algorithms. In this paper, we will focus spike sorting on how to reduce the complexity, and introduce a deep learning algorithm, principal component analysis network (PCANet) to spike sorting. The introduced method starts from a conventional model and establishes a Toeplitz matrix. Through the column vectors in the matrix, we train a PCANet, where some eigenvalue vectors of spikes could be extracted. Finally, support vector machine (SVM) is used to sort spikes. In experiments, we choose two groups of simulated data from public databases available and compare this introduced method with conventional methods. The results indicate that the introduced method indeed has lower complexity with the same sorting errors as the conventional methods.

1. Introduction
Neuronal spikes are basic units of a neural communication system. Neurons in a brain communicate with each other through the spikes. Hence, the study of neuron spikes is a key to understand brain activity. Spike sorting is very important for the study of spikes since different spikes sorted successfully mean their neurons identified successfully. For example, neurons about memory process could be identified successfully through spike sorting [1].

Generally, neuronal spikes are collected from electrodes [2]. The aim of spike sorting is to find when the spikes occur and which neuron the spikes belong to. The earliest spike sorting method is matched filtering [3], which can identify a neuron spike from signals collected by a single electrode. The idea of matched filtering [3] is to match the observed signal to a neuron spike template. If the matched errors are not beyond a threshold, the matched spike would be considered as the neuron’s spike. Today, the matched filtering is still widely applied in single-cell electrophysiology [3]. When the observed signals are from multi-neuron signals collected by multi-electrodes, however, the method does not work well [4] because the observed signals may be from other neurons. In this case, the matched errors will change and the threshold needs to be manual adjustment for better performance.

A cluster method [2, 5-12] is a popular spike sorting method, which works better in the case of multi-neuron spikes collected by multiple electrodes. The cluster method usually involves three steps [5]. First, segment a raw signal into some chips containing spikes via threshold parameters, such as absolute values [13], square values [6] and Teager energy [14], or some nonlinear operators [15]. Second, extract the eigenvalues of the segmented spikes [2, 6, 7]. Third, cluster the extracted eigenvalues via some clustering method, such as k-means [5], superparamag-netic clustering (SPC) [2] and other hybrid cluster method [8].
With the development of modern electrophysiology technology, however, the number of spikes simultaneously collected by electrodes increases dramatically. Now, the multi-electrode technology has been able to record the activity of thousands of neuronal spikes simultaneously [16, 17]. For such large numbers of spikes sorted, the computational complexity has to be considered. Some conventional methods even spend several days or weeks to complete the data sorted [18-20].

At present, there are several new proposed cluster-based methods to solve the spike sorting [10, 11, 21]. Sparse coding algorithm is an unsupervised learning method, which is used to find a set of "overcomplete" base vectors to represent the sample data more efficiently. In recent years, sparse coding has also been applied to spike sorting [21, 22], and has shown better performance. However, as the number of neurons increases, their complexity would not be overlooked.

This paper introduces a principal component analysis network (PCANet) for spike sorting. PCANet is a deep learning framework proposed in recent years [23]. The learning algorithm could be trained and used to classify different types of data and background. The advantage of PCANet is lower computational complexity and simple parameter modulation. This paper will start from a conventional model and train a PCANet, where some eigenvalue vectors are extracted. Then, support vector machine (SVM) is used to sort spikes. From experiment results, the introduced method has lower complexity than traditional sorting methods.

2. System Model
Before sorting spikes, we need to preprocess the raw voltage trace recorded by electrodes. The preprocessing steps are high-pass filtering, whitening and peaks detecting [10]. After detecting the peaks, we find the time when the neuron fires an action potential. Then, the filtered and whiten voltage trace is segmented into some chips containing the peaks. The segmented spikes are just what we need to sort. If \( v(m) \) is denoted as the potential amplitude of the \( m \)th sampling point in a segmented spike, it could be expressed as [10]

\[
v(m) = \sum_{j=1}^{J} w_j(m) \ast \delta(m - n_j) + \varepsilon(m) \tag{1}
\]

where
- \( m \in \{1, 2, \ldots , M\} \),
- \( M \) denotes the length of the spike segment,
- \( w_j(m) \) denotes the potential amplitude of the \( j \)th neural spike template at the \( m \)th point,
- \( \delta(\cdot) \) denotes a unit impulse function,
- \( \ast \) represents a convolution,
- \( n_j \in \{-M + 1, -M + 2, \ldots , M\} \) denotes the time shifted,
- \( \varepsilon(m) \) is an additive noise.

![Figure 1 Schematic diagram of spikes signal system model](image)

**Figure 1** Schematic diagram of spikes signal system model
Eq. (1) means that observed voltage trace can be expressed as the sum of convolutions between several spike template signals and their respective time-shifted impulse functions, shown in Fig. 1. Note that the spike template signal \( w_j(m) \) in (1) could be from the center of clusters via methods, e.g. \( k \)-means. For spikes collected from multiple electrodes, besides, we could constitute them one-by-one into a sequence signal. Here, we only give the case of a single electrode for simplified expression.

3. Algorithm

3.1 Training Data for PCANet

As shown in last section, an observed voltage trace is in fact the superposition of several time-shifted spike templates. We could obtain training data through the time-shifted spike templates and the training data will contain the recorded voltage trace. Then, the parameter of PCANet could be obtained via the training data. Next, we describe how to establish the training data. Firstly, we change (1) into a matrix form as

\[
V = W \Lambda + \varepsilon
\]

where

\[
V = [v(1), v(2), \ldots, v(M)]^T
\]

and \( \varepsilon = [\varepsilon(1), \varepsilon(2), \ldots, \varepsilon(M)]^T \). Since (1) represents convolutions between spike templates and impulse functions, the matrix \( W \) in (2) should be a Toeplitz matrix. Moreover, \( \Lambda \) is a vector composed of the unit impulse function \( \delta(t) \). And, the matrix \( W \) could provide training data that PCANet requires. Here, we will establish the Toeplitz matrix \( W \) as

\[
W = [\overline{W}_j, \overline{W}_j, \ldots, \overline{W}_j]
\]

where,

\[
\overline{W}_j = [w_j(-r_j), w_j(-r_j+1), \ldots, w_j(s_j-1)]
\]

\[
w_j(m) = \begin{cases} \theta_{m-w_j}(1), w_j(2), \ldots, w_j(M-m) \end{cases}^T & \text{if } 0 \leq m \leq s_j-1 \\ \{ w_j(-m+1), w_j(-m+2), \ldots, w_j(M-1) \} \theta_{m-w_j}^T & \text{if } -r_j \leq m \leq -1 \end{cases}
\]

From (4-6), the matrix \( \overline{W}_j \) is composed of several column vectors \( w_j(m) \), \( m = -r_j, -r_j+1, \ldots, s_j-1 \), which is from the \( j \)th neural spike template \( w_j(m) \), \( m = 1, 2, \ldots, M \) through left or right time-shifted. Therefore, the Toeplitz matrix \( W \) could be obtained from \( j \) time-shifted neural spike templates. Besides, \( r_j \) and \( s_j \) represent the maximum left and right shifts, respectively. The choice for \( r_j \) and \( s_j \) is decided by the waveform amplitude of the spike templates and ensure \( 0 < r_j < M \), \( 0 < s_j < M \).

Finally, the dimension of \( W \) is \( M \times N \) if \( u_j = r_j + s_j \), \( N = \Sigma_{j=1}^{J} u_j \), and the column vectors \( w_j(m) \), \( m = -r_j, -r_j+1, \ldots, s_j-1 \) would be the training data for PCANet. Finally, the dimension of \( W \) is \( M \times N \) if \( u_j = r_j + s_j \), \( N = \Sigma_{j=1}^{J} u_j \).

When the training data \( W \) is constructed, it is necessary to transform the \( i \)th training sample \( w_i \) into a matrix, and Figure 2 gives an example of its construction.

After the training matrix is constructed, PCANet algorithm is used to obtain the required model parameters. Table I gives the calculation procedure.
When PCANet completes training, the eigenvalues of each segmented spike signals are extracted. Thus, SVM could sort the spikes via the eigenvalues. The details of PCANet applied to spike sorting are given in Table II.

![Figure 2 An example of transforming training data into matrices](image)

### 3.2 Computational Complexity

In this subsection, we will analyze the computational complexity of this sorting method. From (4), the $i$th column vector $\mathbf{w}_i$ is converted into a $p \times q$ matrix $\mathbf{w}_i$, where $M = p \times q$. In our method, the patch size, i.e. 2D filter size is denoted as $k_i \times k_i$ at all stages. If we let the number of the $i$th layer filter in PCANet be $L_i$, the computational complexity of a two-staged PCANet applied to our spike sorting could be shown as [23]

$$O(pqk_i(L_i + L_i) + pq(k_i)^3)$$

(7)

### 4. Experiment

In experiments, we use two groups of simulated data to verify our PCANet method. And, our methods are compared with the traditional $k$-means, sparse coding [21] and SPC algorithm [2]. All of algorithm codes for this experiments are performed on MATLAB 2012b and the CPU of PC computer is Core i3 3.70GHz.

#### 4.1 Simulated data1

In this sub-section, a simulated data are used to verify the introduced methods and the traditional algorithms. The data denoted by C_Easy1_noise015 are from literature [2], where their waveforms are from real environment and noises are chosen to be able to simulate realistic background activity. Some parameters in this experiment and the algorithms are shown in Table III. More parameters for SPC could be found from [2] and a software, WaveClus version 2.0 downloaded from https://vis.caltech.edu/~rodri/Wave_clus/Wave_clus_home.htm.

The first line of Table IV gives the results for sorting C_Easy1_noise015 data through the four methods, PCANet, SPC, $k$-means, and sparse coding (denoted by SC), respectively. Except the parameters listed in Table III, we specify some parameters for PCANet as follows. A maximum left shift and right shift $r_j$, $s_j$ of the Toeplitz matrix in (4-6) are set to $r_j = 20$, $s_j = 19$, $j = 1, 2, 3$. We specify that, besides, if the difference between spike time in experiment and ground truth is within 4ms, it will be concluded that the both match. When the true spike time is not found to match the experimental one, we regard it as “miss”. When the experimental one does not match the true one, on the other hand, we regard it as “false positives”. We can see from this group of results that, the misses by SPC is the highest, arriving at 486. The number of misses for PCANet is 246, nearly the same as $k$-means, 244 and both of them are higher than sparse coding. The reason why the misses by $k$-means are higher is that, clustering methods will make the eigenvalue point for an overlapped spike be far away from all centers of clusters. For SPC, the highest misses have the following two reasons. One is peak detection because some spikes are not detected. The other is the same as $k$-means. After all, SPC is also based on clustering methods. In addition, the reason for higher misses of PCANet is overlapping spikes. Finally,
it is seen that the false negatives of PCANet is 4, slightly higher than 2 of k-means, lower than 18 of SPC and 10 of sparse coding.

Table 1 PCANet Algorithm

| Training steps |
|-----------------|
| **Input:** training data \( W \) | computing \( \tilde{w}_i \) \( \tilde{w} = [\tilde{w}_{i1}, \tilde{w}_{i2}, \ldots, \tilde{w}_{iM}] \) \( \tilde{w}_{i} = \text{mat}_{i} \left( q_i (\tilde{w} \tilde{w}^t) \right) \in \mathbb{R}^{\lambda_i}, \lambda_i = 1, 2, \ldots, L_i \) \( \tilde{w} = \sum_{i=1}^{L_i} \tilde{w}_{i} \) \( \sum_{i=1}^{L_i} \lambda_i = \lambda = 1, 2, \ldots, L \) \( y' = \tilde{y} = \tilde{y} \) \( \tilde{y} = \text{mat}_{i} \left( q_i (\tilde{y} \tilde{y}^t) \right) \in \mathbb{R}^{L_i}, \lambda_i = 1, 2, \ldots, L \) \( o_{i}' = \left( \tilde{w} \cdot \tilde{w} \right)_{i,x} \) \( \bar{T}' = \sum_{i=1}^{B} \text{hist} (\bar{T}_{i}, \ldots, \text{hist} (\bar{T}_{i}))' \in \mathbb{R}^{(B \times L_i)} \) |
| **Output:** filters \( \tilde{w} \) and \( \tilde{w}' \), feature vector \( f \) |

where

- \( w_i \) denotes the \( i \)-th training sample of the training data \( W \), which is obtained by mean-removed, block sampling.
- \( M = p \times q \), \( \lambda_i \), and \( \lambda_i \) are the patch sizes.
- \( \lambda_i \) denotes the number of filters in layer \( i \).
- \( q_i (\tilde{w} \tilde{w}^t) \) denotes the feature vector corresponding to the \( l \)-th principal component of \( \tilde{w} \tilde{w}^t \).
- \( \text{mat}_{i} (v) \) is a function that maps \( v \) to matrix \( \tilde{w}_i \).
- \( \text{hist} (\bar{T}_{i})' \) means that after \( \bar{T}_{i} \) is divided into \( B \) blocks, all \( B \) histograms are connected into one vector.

**Steps:**
1. The matrix \( \tilde{w}_i \) is obtained from the training data \( W \) via (T-1) and (T-2).
2. The filter \( w_i \) of the first layer network is obtained by the formula (T-3).
3. The output \( w_i \) of the \( l \)-th filter of the first layer network is obtained by (T-4).
4. The mean-removed and block sampled matrix \( v' \) is obtained by (T-5) and (T-6).
5. The filter \( w_i \) of the second layer network is obtained by the formula (T-7).
6. The output \( o_i \) of the second layer network is obtained by the formula (T-8).
7. The final feature vector \( f \) is obtained by (T-9) and (T-10).

Table V gives running time of PCANet for the data, C_Easy1_noise015. From the table, PCANet spends about 441.1765 seconds, which is lower than 3905 of sparse coding.
Table 2 PCANet Applied to Spike Sorting

| Algorithm steps |
|-----------------|
| 1. A raw voltage trace is preprocessed, i.e. whitened, filtered, peaks detected and segmented into a number of signal chips containing the peaks. |
| 2. The eigenvalues of the signal segments are extracted through PCA and clustered into $J$ clusters through $k$-means. Then, the centers of the clusters are used as $J$ spike templates $w_j^{(0)}$, $j = 1, 2, \ldots, J$. |
| 3. The templates $w_j^{(0)}$, $j = 1, 2, \ldots, J$ time-shifted constitute a Toeplitz matrix $W$ in (4), and its column vectors $w_j^{(0)}$ are used as training data for PCANet. |
| 4. The required model parameters are obtained by Table I, and then SVM is used to classify the signal segments. |

Table 3 Some Parameters for the First Group of Data

| Parameter | Value |
|-----------|-------|
| Number of neurons | $J=3$ |
| Number of channel | 1 |
| Filtered | Cut-off frequency at 250 Hz with a Butterworth high pass filter of order 50. |
| Whitened | Whitened in time |
| Peaks detected | A threshold is 6 |
| Length of signal segment | $M=81$ |
| $k$-means: |
| Number of principal component | 4 |
| Contribution rate of principal components | 90% |
| Distance measure | Euclidean distance |
| Clustering repeated | 25 |
| Selection of initial centroid | 3 centroid randomly selected |
| Sparse coding: |
| Sparse coding method | Non-negative least squares |
| Prediction method | k-nearest neighbor |
| A maximum left shift and right shift | $s_r = s_l = 42$, $j = 1, 2, 3$ |
| $I$-th largest | $I=25$ |
| Kernel function | Linear |
| Sparsity threshold | $10^{-4}$ |
| PCANet algorithm: |
| Number of Stages | 2 |
| Patch Size | $k_1 = k_2 = 7$ |
| The size of matrix $\hat{W}$ | $p=q=9$ |
| Number of Filters | $L_1 = L_2 = 8$ |
| Size of each block | 7,7 |
| overlapped block region ratio | 0.92 |
Next, Table VI shows the impact of \( r_j \) and \( s_j \) on the performance of PCANet. From C_Easy1_noise015, the values of \( r_j \) and \( s_j \) have a slight impact on the performance of the algorithms. In practice, we choose the values having fewest sorting errors, \( r_j =20 \), \( s_j =19 \), \( j =1,2,3 \).

Table 4 Results of Sorting Spike in Two Groups of Data by Four Methods

| Experiment data | Number of spikes | PCANet Misses | PCANet FP | SPC Misses | SPC FP | k-means Misses | k-means FP | SC Misses | SC FP |
|----------------|----------------|--------------|----------|-----------|--------|---------------|-----------|-----------|-------|
| C_Easy1_noise015 | 3477 | 246 | 4 | 486 | 18 | 244 | 7 | 10 |
| Simulation 2 | 1058 | 0 | 11 | 0 | 0 | 0 | 0 | 0 |
| Average | 2268 | 123 | 8 | 243 | 9 | 122 | 1 | 4 | 5 |

Note: SC and FP refer to sparse coding, false positives, respectively. These abbreviations indicate the same meaning in the following tables.

Table 5 Running time (second) of PCANet algorithm in different experimental data

| Experiment data | PCANet | SPC | k-means | Sparse coding |
|----------------|--------|-----|---------|---------------|
| C_Easy1_noise015 | 441.1765 | 16.8100 | 6.1590 | 3905 |
| Simulation 2 | 148.5905 | 12.6100 | 8.7420 | 316.70 |

Table 6 Impact of Maximum Right Shift and Left Shift on Performance of PCANet

| \( r_j \) | \( s_j \) | \( r_j \) | \( s_j \) | C_Easy1_noise015 Misses | C_Easy1_noise015 FP | Simulation 2 Misses | Simulation 2 FP |
|----------|----------|----------|----------|----------------|----------------|----------------|----------------|
| 50 49 | 47 49 | 48 47 | 47 260 | 18 3 | 17 |
| 46 44 | 44 44 | 44 44 | 44 260 | 18 8 | 12 |
| 44 43 | 44 43 | 43 43 | 43 260 | 18 15 | 18 |
| 42 42 | 42 42 | 42 42 | 42 252 | 10 5 | 17 |
| 40 41 | 41 41 | 41 41 | 41 254 | 12 7 | 17 |
| 38 38 | 40 40 | 40 40 | 40 257 | 15 10 | 17 |
| 34 33 | 32 32 | 32 32 | 32 266 | 24 1 | 13 |
| 31 30 | 30 30 | 30 30 | 30 259 | 17 0 | 11 |
| 28 28 | 27 28 | 28 26 | 26 261 | 19 1 | 12 |
| 24 23 | 24 23 | 23 22 | 21 248 | 6 4 | 18 |
| 22 21 | 21 21 | 21 21 | 21 250 | 8 2 | 13 |
| 20 19 | 20 19 | 20 19 | 19 246 | 4 4 | 15 |
| 19 18 | 18 18 | 18 18 | 18 249 | 7 1 | 15 |
| 18 17 | 17 17 | 17 17 | 17 248 | 6 1 | 16 |

4.2 Simulated Data

In this subsection, we choose another simulated spike data to verify our method and other conventional methods. The simulated data are from literature [24] and availables downloaded from http://www2.le.ac.uk/centres/csn/software. Most of the parameters are the same as Table III, except some of them in Table VII.

The second line in Table IV gives the sorting results of four methods above, where the values of \( r_j \) and \( s_j \) are chosen as \( r_j =31 \), \( s_j =30 \), \( r_j =30 \), \( s_j =30 \), \( r_j =31 \), \( s_j =30 \). From the table, the misses for all of the four methods are 0 and the false negatives of PCANet are 11, slightly higher than the other three methods.

Table V gives the running time for PCANet. From the table, PCANet spends about 148.5905 seconds, which is also lower than sparse conding. Next, Table VI shows the impact of \( r_j \) and \( s_j \) on the performance of PCANet. From simulation 2, the values of \( r_j \) and \( s_j \) have a slight impact on the
performance of the algorithms. In practice, we choose the values having fewest sorting errors, \( r_1 = 31, s_1 = 30 \), \( r_2 = 30, s_2 = 30 \), \( r_3 = 31, s_3 = 30 \).

**Table 7** Some Parameters for the Second Group of Data

| Parameter                      | Value |
|--------------------------------|-------|
| \( k \)-means:                | 2.    |
| Number of principal            |       |
| compon-ent                     |       |
| Sparse coding:                 |       |
| A maximum left shift           | \( r_1 = 18 \), \( r_2 = 19 \) |
| and rig ht shift               | \( s_1 = s_2 = s_3 = 17 \) |
| \( I \)-th largest             | \( I = 18 \) |

5. Conclusion

Our method is based on a conventional model. We firstly adopt \( k \)-means algorithm to obtain several centers of clusters, through which a Toeplitz matrix is established. Then, the column vectors of the Toeplitz matrix are considered as input training signals of PCANet. In the next step, the output eigenvalue vectors of PCANet would be the input of SVM. The final output of the SVM is sorting result. From experiment results, of course, our method does not show better sorting performance on overlapping spikes. However, the method has less running time of sorting spikes. Therefore, it is probable that a deep learning algorithm, PCANet would have lower computational complexity when sorting multi-electrode spikes.

6. Acknowledgment

This work was supported by the National Natural Science Foundation of China under Grant No. 61262091, The 17th batches of Young and Middle-aged Leaders in Academic and Technical Reserved Talents Project of Yunnan Province under Grant No. 2014HB019, Supported by Program for Innovative Research Team (in Science and Technology) in University of Yunnan Province. Corresponding author is Haifeng Wu.

7. References

[1] Hernan Gonzalo Rey, Maryam Ahmadi, and Rodrigo Quian Quiroga, “Single trial analysis of field potentials in perception, learning and memory”, Current Opinion in Neurobiology, vol. 31, pp. 148-155, 2015.

[2] Quian Quiroga R, Nadasdy Z, Ben-Shaul Y, “Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering”, Neural Computation, vol. 16, no. 8, pp. 1661-1687, 2004.

[3] Turin G, “An introduction to matched filters. Information Theory”, IRE Transactions on Information Theory, vol. 6, no. 3, pp. 311–329, 1960.

[4] Bar-Gad I, Ritov Y, Vaadia E, Bergman H, “Failure in identification of overlapping spikes from multiple neuron activity causes artificial correlations”, Journal of neuroscience methods, vol. 107, no. 1-2, pp. 1–13, 2001.

[5] Lewicki MS, “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. 53–78, 1998.

[6] Rutishauser U, Schuman EM, Mamelak AN, “Online detection and sorting of extracellularly recorded action potentials in human medial temporal lobe recordings, in vivo”, Journal of Neuroscience Methods, vol. 154, no. 1-2, pp. 204–224, 2006.

[7] Harris K, Henze D, Csicsvari J, Hirase H, Buzsaki G, “Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements”, Journal of Neurophysiology, vol. 84, no. 1, pp. 401–414, 2000.
[8] Shoham S, Fellows MR, Normann RA, “Robust, automatic spike sorting using mixtures of multivariate t-distributions”, Journal of Neuroscience Methods, vol. 127, no. 2, pp. 111–122, 2003.

[9] Pouzet C, Mazor O, Laurent G, “Using noise signature to optimize spike sorting and to assess neuronal classification quality”, Journal of Neuroscience Methods, vol. 122, no. 1, pp. 43–57, 2002.

[10] Pillow J, Shlens J, Chichilnisky E, Simoncelli EP, “A model-based spike sorting algorithm for removing correlation artifacts in multi-neuron recordings”, PLoS ONE, vol. 8, no. 5, article e622123, 2013.

[11] Chen B, Carlson D, Carin L, “On the analysis of multi-channel neural spike data”, Advances in Neural Information Processing Systems 24, pp. 936-944, 2011.

[12] Wehr M, Pezaris JS, Sahani M, “Simultaneous paired intracellular and tetrode recordings for evaluating the performance of spike sorting algorithms”, Neurocomputing, vol. 26–27, pp. 1061–1068, 1999.

[13] Obeid I, Wolf P, “Evaluation of spike-detection algorithms for a brain–machine interface application”, IEEE Transactions on Biomedical Engineering, vol. 51, no. 6, pp. 905–911, 2004.

[14] Choi JH, Jung HK, Kim T, “A new action potential detector using the MTEO and its effects on spike sorting systems at low signal-to-noise ratios”, IEEE Transactions on Biomedical Engineering, vol. 53, no. 4, pp. 738–46, 2006.

[15] Rebrik SP, Wright BD, Emondi AA, Miller KD, “Cross-channel correlations in tetrode recordings: implications for spike-sorting”, Neurocomputing, vol. 26–27, pp. 1033–1038, 1999.

[16] Dion Khodagholy, Jennifer N Gelinas, Thomas Thesen, et al, “NeuroGrid: recording action potentials from the surface of the brain”, Nature Neuroscience, vol. 18, no. 2, pp. 310-315, 2014.

[17] Hernan Gonzalo Rey, Carlos Pedreira and Rodrigo Quián Quiroga, “Past, present and future of spike sorting techniques”, Brain Research Bulletin, vol. 119, no. Part B, pp. 106-117, 2015.

[18] Rossant C, Kadir S N, Goodman D F M, et al, “Spike sorting for large, dense electrode arrays”, Nature Neuroscience, vol. 19, no. 4, pp. 634-641, 2016.

[19] Hilgen G, Sorbaro M, Pirmoradian S, et al, “Unsupervised spike sorting for large-scale, high-density multielectrode arrays”, Cell Reports, vol. 18, no. 10, pp. 2521-2532, 2017.

[20] Pachitariu M, Steinmetz N, Kadir S, et al, “Kilosort: realtime spike-sorting for extracellular electrophysiology with hundreds of channels”, bioRxiv, 2016: 061481.

[21] Kai Yang, Haifeng Wu, Yu Zeng, “Sparse coding approaches for neuron spike sorting”, Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), pp. 600-604. IEEE Press, Xi An, 2016.

[22] Li Y, Ngom A, “Sparse representation approaches for the classification of high-dimensional biological data”, BMC systems biology, vol. 7, no. 4, pp. S6, 2013.

[23] Chan T H, Jia K, Gao S, et al, “PCanet: A simple deep learning baseline for image classification?”, IEEE Transactions on Image Processing, vol. 24, no. 12, pp. 5017-5032, 2015.

[24] Martinez J, Pedreira C, Ison M J, et al, “Realistic simulation of extracellular recordings”, Journal of neuroscience methods, vol. 184, no. 2, pp. 285-293, 2009.