Neural Networks 134 (2021) 131–142

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Deep-learned spike representations and sorting via an ensemble of auto-encoders

Junsik Eom [a,1], In Yong Park [a,1], Sewon Kim [a,1], Hanbyol Jang [a,1], Sanggeon Park [b,c,d,e], Yeowool Huh [b,c], Dosik Hwang [a,*]

[a] School of Electrical and Electronic Engineering, Yonsei University, Seoul 03722, Republic of Korea
[b] Department of Medical Science, College of Medicine, Catholic Kwandong University, Gangneung 25601, Republic of Korea
[c] Translational Brain Research Center, Catholic Kwandong University, International St. Mary's Hospital, Incheon 22711, Republic of Korea
[d] Department of Neuroscience, University of Science and Technology, Daejeon, 34113, Republic of Korea
[e] Center for Neuroscience, Korea Institute of Science and Technology, Seoul 02792, Republic of Korea

ARTICLE INFO

Article history:
Received 8 June 2020
Received in revised form 1 October 2020
Accepted 16 November 2020
Available online 27 November 2020

Keywords:
Unsupervised spike sorting
Deep learning-based auto-encoder
Feature extraction
Clustering

ABSTRACT

Spike sorting refers to the technique of detecting signals generated by single neurons from multi-neuron recordings and is a valuable tool for analyzing the relationships between individual neuronal activity patterns and specific behaviors. Since the precision of spike sorting affects all subsequent analyses, sorting accuracy is critical. Many semi-automatic to fully-automatic spike sorting algorithms have been developed. However, due to unsatisfactory classification accuracy, manual sorting is preferred by investigators despite the intensive time and labor costs. Thus, there still is a strong need for fully automatic spike sorting methods with high accuracy. Various machine learning algorithms have been developed for feature extraction but have yet to show sufficient accuracy for spike sorting. Here we describe a deep learning-based method for extracting features from spike signals using an ensemble of auto-encoders, each with a distinct architecture for distinguishing signals at different levels of resolution. By utilizing ensemble of auto-encoder ensemble, where shallow networks better represent overall signal structure and deep networks better represent signal details, extraction of high-dimensional representative features for improved spike sorting performance is achieved. The model was evaluated on publicly available simulated datasets and single-channel and 4-channel tetrode *in vivo* datasets. Our model not only classified single-channel spikes with varying degrees of feature similarities and signal to noise levels with higher accuracy, but also more precisely determined the number of source neurons compared to other machine learning methods. The model also demonstrated greater overall accuracy for spike sorting 4-channel tetrode recordings compared to single-channel recordings.

© 2020 Elsevier Ltd. All rights reserved.

1. Introduction

Extracellular single-unit recording enables investigation of how the activity patterns of individual neurons contribute to specific behaviors. Extracellular recordings, however, often contain signals (action potentials or spikes) from multiple neurons that first must be attributed to individual neurons (referred to as spike sorting). Spike sorting is based primarily on spike shapes, which differ according to distance from the recording electrode, by the position relative to other neurons, and among different source neurons. For manual spike sorting, spike shape features such as peak amplitude, width, and after-hyperpolarization are most frequently considered (Guenther et al., 2009; Lewicki, 1998). However, such manual spike sorting requires much time and effort, and in cases where spike shapes are similar, classification may be difficult even for well-trained neuroscientists (Harris, Henze, Csicsvari, Hirase, & Buzsaki, 2000). Therefore, many automatic spike sorting methods have been developed to increase performance. In general, automatic spike sorting is performed in three steps. (1) spike detection, (2) feature extraction, and (3) clustering to assigned labels (i.e., specific individual neurons). For spike detection, analogue neural signals are amplified, band-pass filtered (usually at 300–6000 Hz), and digitized using an analog-to-digital converter (ADC). Spikes at different time events are then selected by setting a threshold and aligned by the spike occurrence times (Nenadic & Burdick, 2004). Although it is possible to directly use raw data features, using feature extraction to

* Corresponding author.
E-mail addresses: junsik424@yonsei.ac.kr (J. Eom), inyong@yonsei.ac.kr (I.Y. Park), sewon.kim@yonsei.ac.kr (S. Kim), hanstar4@yonsei.ac.kr (H. Jang), chalspark.korea@gmail.com (S. Park), huh06@cku.ac.kr (Y. Huh), dosik.hwang@yonsei.ac.kr (D. Hwang).
[1] Equally contributing authors.

https://doi.org/10.1016/j.neunet.2020.11.009
0893-6080/© 2020 Elsevier Ltd. All rights reserved.

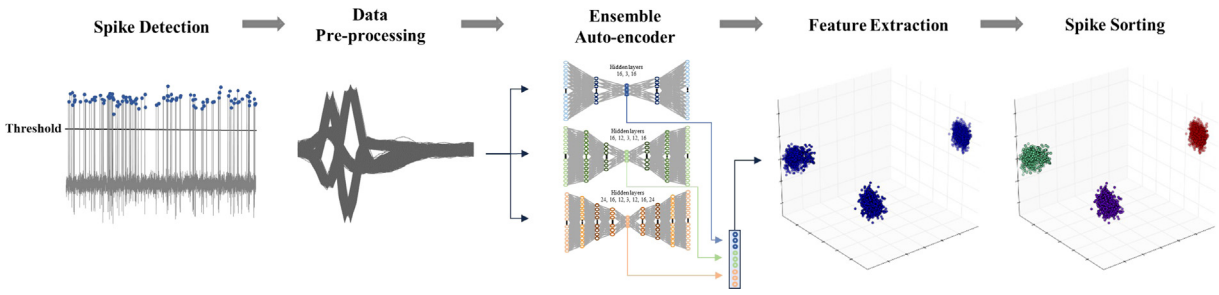

Fig. 1. Overall flow of the ensemble auto-encoder spike sorting model. First, spikes are filtered from raw neuronal action potential signals using a voltage threshold. The spike data are then pre-processed so that voltage difference values are used instead of the raw values as input. An ensemble auto-encoder then performs feature extraction, yielding latent space values as distinguishing features for sorting. Machine learning-based clustering algorithms are then applied to the extracted features for spike sorting.

reduce the data into lower feature dimensions greatly improves spike sorting performance (Gibson, Judy, & Marković, 2011; Wild, Prekopcsak, Sieger, Novak, & Jech, 2012). Feature extraction algorithms such as principal component analysis (PCA), diffusion map (DM), self-organizing map (SOM), wavelet transform (WT), and linear discriminant analysis (LDA) are used to improve spike sorting performance (Adamos, Kosmidis, & Theophilidis, 2008; Keshtkaran & Yang, 2017; Nguyen et al., 2015; Quiroga, Nadasdy, & Ben-Shaul, 2004; Vesanto & Alhoniemi, 2000). However, these methods are highly sensitive to noise and often fail to extract meaningful features in cases of high spike shape similarity. Therefore, a novel method applying an appropriate feature extraction technique that is both robust to noise and signal similarity is needed for improved spike sorting. Deep learning-based methods are an intensive area of investigation for classification of various physiological signals and image features due to their outstanding performance. In the current study, we describe an unsupervised feature extraction model based on deep learning for extracting discriminative features from spikes. The critical aspect of our deep learning-based method is that feature extraction is performed by 3 different auto-encoders (AEs). Using this method, we were able to successfully extract meaningful features even from signals with high structural similarity and noise (low signal-to-noise). When these extracted features are processed using various clustering algorithms such as K-means, Gaussian mixture models (GMMs), superparamagnetic clustering (SPC), and density-based spatial clustering of applications with noise (DBSCAN) (Blatt, Wiseman, & Domany, 1996; Ester, Kriegel, Sander, Xu, et al., 1996; Likas, Vlassis, & Verbeek, 2003; Reynolds, 2009), it yielded high classification and cluster number prediction accuracy. To test model performance, we compared classification and cluster number prediction accuracy to other machine learning (ML) algorithms on both simulated and *in vivo* data with various noise levels and degrees of spike shape similarities.

2. Methods

The spike sorting algorithm using deep learning-based ensemble AE is illustrated in Fig. 1. The first step is to detect spikes from recorded signals by setting an amplitude threshold value. Each spike data is then pre-processed into gradient values and normalized. To obtain the key features from the pre-processed data, ensemble AE is then utilized for feature extraction. Finally, a clustering algorithm is applied to the extracted features. To evaluate the performance of the proposed method, two simulated datasets and *in vivo* datasets were used. Our model takes approximately 68.81 s on average for the automatic sorting of 3526 spikes in these datasets.

2.1. Dataset

We used two simulated datasets to evaluate the clustering performance of these models. Simulated data is advantageous for evaluation of spike sorting algorithms as the final output can be compared to known labels. The first simulated dataset we used for our study was provided by the University of Leicester and published by Quiroga et al. (2004). It is a wildly used dataset for spike sorting performance evaluation. The dataset is composed of 20 subset data with varying spike similarities and noise levels. Depending on the spike similarity, the subset data is categorized into 4 difficulty types, named Easy1, Easy2, Difficult1, and Difficult2. Each difficulty type data contains 4 levels of noise (0.05, 0.10, 0.15, and 0.20) except for Easy1 which additionally contained cases with more extreme levels of noise (0.25, 0.30, 0.35, and 0.40). The noise levels were determined by taking the standard deviation of the voltage trace relative to spike amplitude. The simulated data were generated at a sampling rate of 96 kHz and down-sampled to 24 kHz. Each of the 20 data subsets is composed of 1,440,000 sampling points, and specific spike times are given to each spike based on ground truth. The three different spikes have a mean firing rate of 20 Hz with Poisson distribution of inter-spike intervals, and the refractory time between action potentials from the same neuron is 2 ms. We will refer to these records as Dataset1. The second simulated dataset (Pedreira, Martinez, Ison, & Quiroga, 2012) is composed of 95 simulated extracellular recordings sampled at 24 kHz. The data contains various numbers of source neurons ranging from 2 to 10. To further simulate various neuronal numbers, the signals from the second simulated dataset were recomposed to create 100 subsets with the number of 2, 3, 4, or 5 source neurons. The resulting 400 subsets are referred to as Dataset2. Both of the simulated datasets are generated based on recordings from basal ganglia and neocortex. The *in vivo* dataset was obtained from the mouse cortex using a tetrode. Neural activities were band-pass filtered at 600 Hz to 6 kHz, amplified by 10,000, and digitized at 30.3 kHz using Digital Lynx (Neuralynx, Bozeman, MT, USA). Single units were manually sorted using SpikeSort 3D (Neuralynx) by a trained expert, and cluster quality was inspected using isolation distance, L-ratio, inter-spike interval in the ISI histogram (ISI >1 ms), and cross-correlation. For our study, both single-channel and 4-channel tetrode recordings were used for evaluation of the proposed spike sorting model (Huh, Bhatt, Jung, Shin, & Cho, 2012; Huh & Cho, 2013, 2016). The noise level of *in vivo* dataset, obtained by the identical method used for simulated Dataset1, was 0.03.

2.2. Environment

The proposed unsupervised spike sorting algorithm was written in Python and implement with Keras using TensorFlow backend. Spike sorting is performed using Intel Xeon E5-1620 CPU with a Nvidia Titan X GPU and 96 GB RAM.

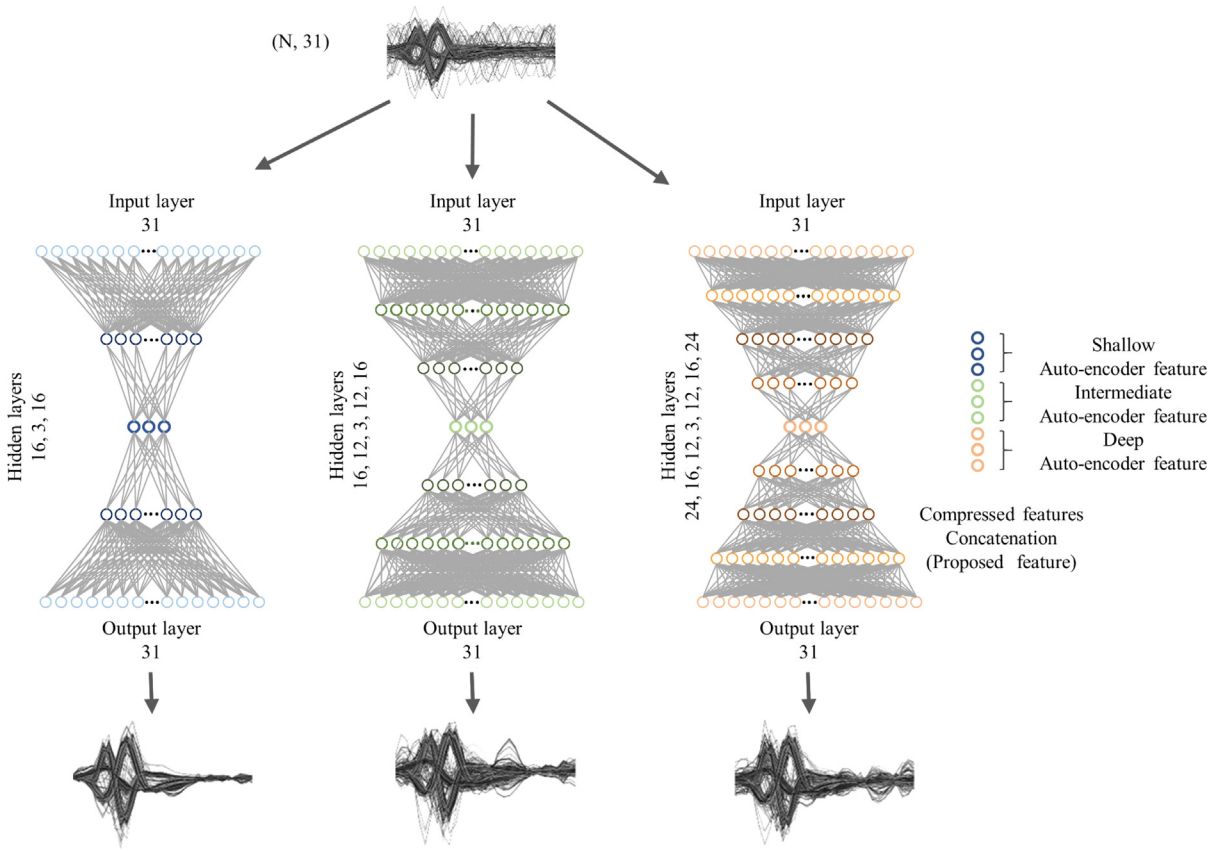

Fig. 2. The deep learning-based ensemble auto-encoder. Input spikes are applied to different sizes of auto-encoder models, and the reconstruction results are shown at the bottom of each model. The extracted features are shown on the right side, which are concatenated from latent spaces of each models.

2.3. Data processing

The raw extracellular neural signal was sampled at 24 kHz using an ADC and band-pass filtered at 300–6000 Hz for noise exclusion and DC offset removal. Spikes can be detected using a variety of methods (Nenadic & Burdick, 2004; Nguyen, Khosravi, Creighton, & Nahavandi, 2014; Obeid & Wolf, 2004; Wilson & Emerson, 2002). In our paper, spike detection was performed by setting a threshold $4\sigma_n$ (Quiroga et al., 2004) where

$$\sigma_n = median\left\{\frac{|x|}{0.6746}\right\} \tag{1}$$

and saved as 32 sampling points (41.6 μs intervals between sampling points). To avoid misalignment, all spikes were aligned using their spike times (Quiroga et al., 2004). Input data for the deep learning model was min–max normalized to a scale of 0 to 1 so that the model was less sensitive to the absolute scale of features. Furthermore, rather than using the raw action potential values as input, differential geometry, the gradient in our case, was used since it is more amenable to signal processing (Manton, Applebaum, Ikeda, & Le Bihan, 2013). The gradient is obtained by calculating the normalized voltage difference every 41.6 μs $*$ $v\,(v = 1, 2, \ldots, 10)$. The pre-processed data for the deep learning model can be explained by the equations

$$\begin{aligned} X_v\,(N, f) &= \frac{P_N\,(f + v) - P_N\,(f)}{v} \\ &(v = 1, 2, \ldots, 10) \\ &(k = 32 - v) \\ &(f = 1, 2, \ldots, k) \end{aligned} \tag{2}$$

where N represents the number of spikes in dataset X, f is the sampling time of each sampling point, and v is the point interval length. An interval value of 41.6 μs $*$ $v\,(v = 1, 2, \ldots, 3)$ showed higher performance for determining the correct number of source neurons compared to other sampling point values and pre-processed data with 41.6 μs $*$ $v\,(v = 4, 5, \ldots, 10)$. Using $v = 1$ resulted in highest performance. Fig. 6a shows that the performance increased using the gradients values rather than the point values as described in Results section.

2.4. Feature extraction through auto-encoder

A deep learning-based AE is an unsupervised tool for learning data representations utilizing the bottleneck structure of the model. It is composed of three major parts: the encoder (θ), where the input data (spike signals) are reduced to lower dimensions; the latent space (h), i.e., the dimensionally reduced input data from the encoder, which its values become the extracted features; and a decoder (ϕ), where the latent space values are reconstructed back into the original signal. The weights of the network are trained so that the difference between input values and output values are minimized for all data. In other words, the network learns to extract key features from input so that the model can best reconstruct the input from the extracted features. Unlike PCA, a linear transformation method that represents data only in a lower dimensional hyperplane, our deep learning-based AE method can represent high-dimensional relationships within the data by expanding dimensions using non-linear activation functions before extracting features. The operations of AE are defined by the equation

$$\theta, \phi = argmin\|X - \hat{X}\|^2 \tag{3}$$

where X is the input data and $\hat{X}$ is the output data. The encoder is responsible for reducing the data into lower dimensions while

preserving as much information as possible (Kingma & Welling, 2013). The encoding input data X is composed of n samples, each with m features. When processed, m features are reduced to the number of nodes in the latent space (h) layer. The encoding process is described by the equation

$$h = \sigma(WX + b) \tag{4}$$

where σ is a non-linear activation function $ReLU$, W is the weight trained by the neural network, and b is the bias vector. The decoder then takes the dimensionally reduced data (the latent space values) from the encoder to best reconstruct the original input data (Kingma & Welling, 2013). The output of the decoder $\hat{X}$ is obtained by

$$\hat{X} = \sigma'\left(W' + b'\right) \tag{5}$$

The training process of the AE algorithm is performed through learning to minimize the reconstruction loss between input and output. The commonly utilized loss function mean square error (MSE, below) is utilized in our model.

$$\begin{aligned} \mathcal{L}\left(X, \hat{X}\right) &= \|X - \hat{X}\|^2 \\ &= \|X - \sigma'\left(W'\left(\sigma\left(WX + b\right)\right) + b'\right)\|^2 \end{aligned} \tag{6}$$

2.5. Ensemble auto-encoder

There are many distinct types of auto-encoders, such as variational auto-encoders, spare auto-encoders, and denoising auto-encoders, that can achieve optimal feature extraction for specific tasks (Meng, Catchpoole, Skillicom, & Kennedy, 2017; Vincent, Larochelle, Bengio, & Manzagol, 2008; Zhang, Cheng, Liu, He, & Liu, 2018). We found that an ensemble AE was able to improve feature extraction by utilizing different hidden representations of the input signals. The features are generated by concatenating the reduced features (the latent space) from AE models of varying architectures. Models with different numbers of layers and nodes yield different loss values and reconstruction qualities, so an ensemble of AE models with distinct structures can detect multiple underlying qualities of the signal. Moreover, if there are critical qualities shared among the different models, the assembly can adjust the feature weighting according to importance. The first AE model shown in Fig. 2 is composed of 3 hidden layers with 16, 3, and 16 nodes, respectively, where the layer with 3 nodes corresponds to the latent space. The second model contains 5 hidden layers of 16, 12, 3, 12, and 16 nodes, respectively, and the third contains 7 hidden layers of 24, 16, 12, 3, 12, 16, and 24 nodes, respectively. The shallow model with 3 hidden layers only reconstructs the most general features of the signal input such as the spike peak and valley but loses more detailed features. On the other hand, the deeper AE models reconstruct the signal with greater preservation of detail. We then used the concatenated features obtained from the bottleneck structure of the model as input for the clustering algorithms. Each latent space (compressed feature) of the three AEs can be expressed by the following equations

$$\begin{aligned} h_I &= \sigma_I\left(W_I X + b_I\right) \\ h_{II} &= \sigma_{II}\left(W_{II} X + b_{II}\right) \\ h_{III} &= \sigma_{III}\left(W_{III} X + b_{III}\right) \end{aligned} \tag{7}$$

where h_I, h_{II}, and h_{III} are the latent spaces obtained from each AE model. Each contains a total of n samples with 3 nodes of latent space. Our proposed feature H is obtained by concatenating the compressed features from each model, resulting in n samples each with 9 features.

$$H = [h_I, h_{II}, h_{III}] \tag{8}$$

Concatenating the 3 different features from the 3 different AE models compensates for the limitations of each model, yielding best feature extraction performance when the latent spaces are ensembled. And the extracted features are applied as the input of clustering algorithms such as K-means, GMM, and DBSCAN for spike sorting.

2.6. Feature extraction methods

2.6.1. Principal component analysis

Principal component analysis (PCA) is a technique for dimensional reduction which is widely used on numerous types of data, including spike recordings. It reduces the dimensionality of the dataset by obtaining the principal components (PCs) of the data and using them as the new dimensions. Each PC is orthogonal to all other PCs and the data projected on the PCs have maximum variance, i.e., PC2 is a vector that is orthogonal to PC1 and has maximum variance when data is projected on it while being orthogonal to PC1. The PCs are obtained by taking the eigenvectors from the covariance matrix of the data.

2.6.2. Wavelet transform

Wavelet transform is often used as a feature extraction method. After wavelet transform is applied to each spike with specific sets of mother wavelets of different scales and times, wavelet coefficients that separate the spike clusters best are obtained and chosen as the new dimensions. Unlike Fourier transform, which represents frequency components under the assumption that the signal does not change in time, the wavelet transform represents the time and frequency components of signals with frequency components that change in time. The function $\omega(t)$, called the mother wavelet, is defined by $\omega(t) \in L^2(R)$ and is characterized by a mean of zero and normalized distribution.

$$\begin{aligned} \int_{-\infty}^{\infty} \omega(t) &= 0 \\ \|\omega(t)\|^2 = \int_{-\infty}^{\infty} \omega(t)\,\omega^*(t)\,dt &= 1 \end{aligned} \tag{9}$$

Wavelets are obtained from the mother wavelet through property of the dilation and translation states, and defined as

$$\omega_{s,\tau}(t) = \frac{1}{\sqrt{s}}\omega\left(\frac{t-\tau}{s}\right) \tag{10}$$

where s and τ are the scaling parameter and translation parameter of the wavelet function, respectively. Through the equation below, one dimensional signal $f(t)$ can be converted to two dimensional coefficients $c(s, \tau)$ as

$$\begin{aligned} c(s,\tau) = \int_{-\infty}^{\infty} f(t)\,\frac{1}{\sqrt{s}}\omega^*\left(\frac{t-\tau}{s}\right)dt, \\ s \in \mathbf{R}^+, \tau \in \mathbf{R} \end{aligned} \tag{11}$$

Wavelet coefficients decomposed by the wavelet function represent local or subsections of the entire data. Therefore, wavelet transformation can be used as a dimensionality reduction method that extracts important features from the entire dataset.

2.6.3. Diffusion map

Like PCA, diffusion map can be used as a dimensionality reduction technique. However, diffusion map differs in that it takes into consideration the underlying manifold of the data by calculating the "diffusion distance" between points. This is achieved through the following steps. First, given a set of points $X = \{x_1, x_2, \ldots, x_n\} \in \mathbf{R}$, we can represent the data as a weighted graph in the form of an adjacency matrix, where each data

point is a node and the connectivities between k-nearest points are weights assigned to each undirected edge derived using the Gaussian kernel

$$W_{ij} = e^{-\frac{(x_i - x_j)^2}{\varepsilon}} \tag{12}$$

In this equation, ε is a data-dependent parameter. Too small of a value will result in entries of W close to 0, while too large of a value will result in entries of W close to 1, so intermediate values where the slope of $\sum_k \sum_k W_{ij}$ was largest were chosen (Bah, 2008). We then created a Markov matrix P that represents the transition probability between data points, i.e. a random walk on the data as follows:

$$P_{ij} = \frac{W_{ij}}{\Sigma_k W_{ik}} \tag{13}$$

Diffusion distance between two points is defined as the weighted L^2 distance where L^1 is the norm $\|P(x,.) - P(z,.)\|$, so the equation for diffusion distance becomes

$$D(x,z) = \sqrt{\sum_k \frac{(P(x,k) - P(z,k))^2}{\phi(k)}} \tag{14}$$

where $\phi(k) = 1(P_e(k))$ (Nadler, Lafon, Coifman, & Kevrekidis, 2006). This applied with spectral theory on random walk results in an eigenvalue problem where the non-trivial principal eigenvectors weighted by the corresponding eigenvalues become the new coordinates of the data in which the Euclidean distances approximate the diffusion distances between points (the first eigenvector is omitted since it is a constant vector) (Coifman & Lafon, 2006)

$$P\nu = \lambda\nu \tag{15}$$

Finally, the dimensionality reduction can be expressed as

$$Y = \{\lambda_2\nu_2, \lambda_3\nu_3, \ldots, \lambda_{d+1}\nu_{d+1}\} \tag{16}$$

2.6.4. Linear discriminant analysis

Linear discrimination analysis is a method for extracting discriminative and low-dimensional features by iterative subspace selection. Generating a linear discriminative projection matrix $W_{(f\times l)}$ for dimensional reduction ($l < f$) from $X_{(N\times f)}$ is important for maximum separability of l-dimensional features. Let S_B and S_W be a "between classes scatter matrix" and a "within classes scatter matrix", respectively. The scatter matrices could be defined as

$$\begin{aligned} S_W &= \sum_{k=1}^{K} \sum_{x_i \in C_k} (x_i - \mu_k)(x_i - \mu_k)^T \\ S_B &= \sum_{k=1}^{K} n_k \mu_k \mu_k^T \end{aligned} \tag{17}$$

where K is the number of clusters and μ_k represents the center of C_k among each data point x_i belonging to the cluster C_k. To achieve the best cluster separation, the distance between classes needs to be large and the scatter distance within classes needs to be small. By selecting proper eigenvectors and eigenvalues, it is possible to obtain maximum separability.

$$max \frac{W^T S_B W}{W^T S_W W} \tag{18}$$

2.6.5. Self-organizing map

A self-organizing map (SOM) is an artificial neural network trained for generating a low-dimensional representation from the original data while preserving topological characteristics. SOM trains datasets through competitive learning unlike error-correction learning algorithms such as backpropagation, then high-dimensional data are consecutively mapped into SOM nodes (10×10 in this case). The best matching unit (BMU) node for each observation is selected by the equation

$$\|x - s_c\| = argmin_i \|x - s_i\| \tag{19}$$

where x is the input vector, s_c is the weight vector closest to x, and s_i is the weight vector of the SOM nodes. And both weight vectors for the BMU and the neighborhood nodes are updated during the training based on the Euclidean distance between the weight vectors and the input vectors. The update moves the SOM nodes to denser regions of the data. The weight vectors are updated by the equation

$$s_i(t+1) = s_i(t) + h_{ci}(t)\,\mu(t)\,[x - s_i(t)] \tag{20}$$

where h_{ci} is the update weighting for the BMU and the neighborhood nodes, μ is the learning rate, and t is the time. By updating the weight vectors, high-dimensional dataset can be mapped into 2 or 3 dimensions, making it easier to cluster the dataset.

2.7. Clustering methods

2.7.1. K-means

K-means clustering is widely used ML algorithm for dividing data sets into k groups (Kojima, 1969). It is an iterative algorithm started by selecting the number of k for the randomly positioned centers and minimizing the clustering error by changing the center position. The most widely used version for clustering is the Hartigan–Wong algorithm (1979), which is based on the sum of the squared Euclidean distances between each observation x_i and the corresponding centroid m_k. Given dataset $X = \{x_1, x_2, \ldots, x_n\}, X_n \in R^d$, the total distance of each data point from the cluster center $E(C_k)$ is given by

$$\sum_{k=1}^{k} E(C_k) = \sum_{k=1}^{k} \sum_{x_i \in C_k} \|x_i - m_k\|^2 \tag{21}$$

As the K-means algorithm can be sensitive to the initial position of the cluster center, it should be iterated multiple times with different initial positions. If the total distance is minimized properly, it shows good compactness of clustering.

2.7.2. Gaussian Mixture model

The Gaussian mixture model clustering method is a probabilistic approach using weights, means, and covariance of Gaussians. The multinomial Gaussian distribution with K components for the number of observations (n) is defined as

$$p(x_i) = \sum_{k=1}^{K} \pi_k N(x_i|\mu_k, \Sigma_k) \tag{22}$$

where π_k is the probability of the observed data which belongs to the cluster k, μ is the average value of each cluster, and Σ is the covariance of each cluster. The log likelihood function of the GMM algorithm for finding the optimal parameters (π, μ, Σ) that maximize the probability is defined by

$$p(x_i|\pi, \mu, \Sigma) = \sum_{i}^{n} \ln \sum_{k}^{K} \pi_k N(x_i|\mu_k, \Sigma_k) \tag{23}$$

The expectation–maximization (EM) algorithm is used to obtain the optimal values for π, μ, Σ. The EM method is performed in 4 steps.

Initial step: Randomly initialize π, μ, Σ

E-step: calculate responsibility

$$\gamma(z_{ik}) = \frac{\pi_k N(x_i|\mu_k, \Sigma_k)}{\Sigma_j \pi_j N(x_i|\mu_j, \Sigma_j)} \tag{24}$$

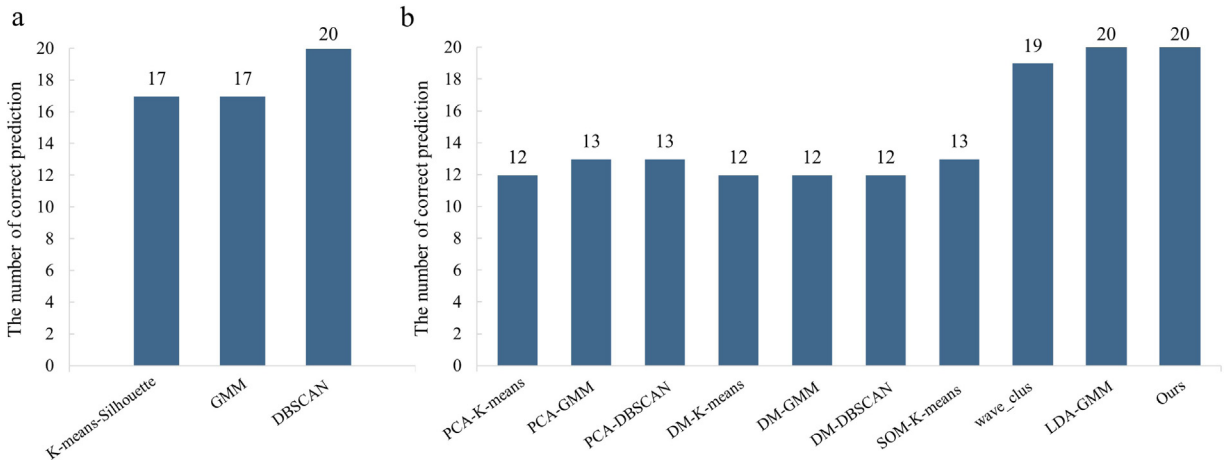

Fig. 3. Performance of the model in determining the number of source neurons for simulated Dataset1 compared to other feature extraction methods. (a) Performance ratings of different clustering algorithms for correctly determining the number of source neurons in simulated Dataset1 based on extracted features using the proposed model. (b) Performance ratings of spike sorting methods using different feature extraction and clustering algorithms for correctly determining the number of source neurons in Dataset1.

M-step: parameter estimation with responsibility.

$$\mu_k^{new} = \frac{1}{N_k}\sum_i^n \gamma\left(z_{ik}\right) x_i$$
$$\Sigma_k^{new} = \frac{1}{N_k}\sum_i^n \gamma\left(z_{ik}\right)\left(x_i-\mu_k\right)\left(x_i-\mu_k\right)^T \quad (25)$$
$$\pi_k^{new} = \frac{N_k}{N}$$

Evaluation-step: repeat E-step and M-step until responsibility or parameters converge.

2.7.3. DBSCAN

The main purpose of DBSCAN is to partition dense regions and measure the optimal number of clusters within a given dataset. The radius ε of the neighborhood around a point and the minimum number of points n within radius ε are the two main parameters required to arrange points into 3 different categories: the core points, the border points, and the outlier points. Points with neighboring points within their radius ε equal to or more than n are defined as core points. The minimum number of points n must satisfy the condition $n \geq$ number of data dimensions $+ 1$. Points that have at least one core point within their radius ε and are not core points are defined as border points. Finally, outlier points are points that are neither core points nor border points. If two or more core points share a border point within their radius ε, the core points and all their corresponding border points are considered in the same group.

2.7.4. Silhouette statistics

Evaluation of each clustering by silhouette scores is based on comparison of compactness with separability (Rousseeuw, 1987). The silhouette score measures the quality of clustering by estimating the average distance between clusters. Let x_i be an observation of the dataset. We can calculate the average dissimilarity a_i by deriving the average distance of x_i from all other points within the cluster. Similarly, the average dissimilarity b_i can be calculated by deriving the lowest average distance of x_i from all points in other clusters. In other words, b_i can be described as the dissimilarity between x_i and its neighboring clusters. Finally, the silhouette score for x_i th observation is defined as

$$S_i = \frac{b_i - a_i}{\max\left(a_i, b_i\right)} \quad (26)$$

The range of silhouette score is $-1 \leq S_i \leq 1$ according to the equation above. A silhouette score that is close to 1 indicates good compactness within clusters and high separation among clusters. A silhouette score that is close to -1 on the other hand indicates clustering with low compactness and separability, making the classification task more challenging.

3. Result

3.1. Performance of ensemble auto-encoder

Extracted features from ensemble AE models were applied as input to the clustering algorithms for spike sorting. The results show that the features extracted by the proposed method are reliable, robust under different noise levels, and show good clustering performance with high compactness and separability. There are two different types of accuracy to consider in spike sorting. The first is the accuracy in determining the number of source neurons, and the second is the classification accuracy of all spikes contained in the dataset. For automatic spike sorting, it is critical to accurately determine the number of source neurons. The performances of our proposed feature extraction methods using K-means, GMM, and DBSCAN for clustering are shown in Fig. 3a. Both K-means and GMM clustering methods require a prior knowledge of the number of clusters, while DBSCAN does not. Therefore, clustering performance by K-means and GMM was rated using silhouette scores for each pre-determined cluster number from 1 to 20, with the optimal number of clusters receiving the largest silhouette score. Alternatively, the optimal number can be predicted using DBSCAN by iterating the algorithm. The performances of clustering algorithms on simulated Dataset1 are shown in Fig. 3a. Both the K-means and GMM clustering methods were able to correctly predict the number of source neurons in 17 of 20 cases, but DBSCAN correctly determined the number of source neurons for all 20 cases. Therefore, all subsequent clustering procedures for the ensemble AE-extracted features were performed using DBSCAN.

3.2. Comparison with other spike sorting algorithms

Fig. 3b compares the performance of other spike sorting algorithms to our proposed method for estimating the number of source neurons in simulated Dataset1. All PCA, DM, and SOM-K-means (Pacella, Grieco, & Blaco, 2016) exhibited lower performance, yielding the correct number in only 13, 12, and 13 of

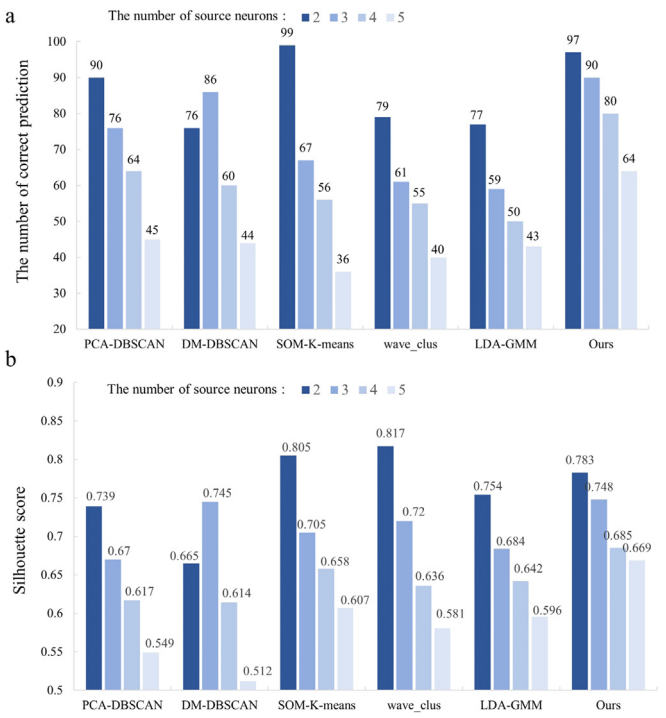

Fig. 4. Performance comparison among models on feature extraction for simulated Dataset2. (a) Performance of various feature extraction and clustering algorithms for correctly determining the number of source neurons for Dataset2. (b) Silhouette scores of extracted features for the different feature extraction methods (True labels were given to all data when calculating the silhouette scores.).

20 cases, respectively. On the other hand, our model as well as LDA-GMM were able to accurately predict the number of source neurons in all 20 cases. Fig. 4a compares the performances of the feature extraction and clustering methods on simulated Dataset2. This dataset contains 100 subsets with signals from 2, 3, 4, and 5 source neurons. We also compared the conventional feature extraction methods with the DBSCAN clustering algorithm and three previously published methods: SOM-K-means, wave_clus, and LDA-GMM. Our proposed model showed superior performance over the other spike sorting algorithms in most cases with different numbers of source neurons. When the cluster count was 2 and 3, our model scored 4%–31% higher than other spike sorting algorithms except the SOM-K-means. The latter scored 2% higher than our model when the cluster count was 2. And when cluster count was more than 3, our model outperformed the other algorithms by at least 16%. We then investigated the spike classification accuracy using both Datasets1 and 2, where spike sorting accuracy was defined as correctly classified spikes/total number of spikes. Table 1 compares the classification accuracy of the proposed feature extraction model clustered using DBSCAN with the other spike sorting algorithms on simulated Dataset1. While the other spike sorting algorithms demonstrated classification accuracies below 90% when the noise level was above 0.3 with Easy1, the proposed model demonstrated over 99% accuracy. In addition, our model yielded greatest performance on Difficult2, the most challenging subsets from Dataset1, with average spike sorting accuracy of 99.81%. Classification accuracy was also tested on simulated Dataset2 (Table 2). The values are presented as averaged spike sorting accuracy of 100 subsets for each number of source neurons. Again, our feature extraction model with DBSCAN shows better performance for predicting the number of source neurons and higher spike sorting accuracy than compared spike sorting algorithms.

3.3. Evaluation of extracted features

Fig. 5a shows the extracted features from a few selected subsets of simulated Dataset2. To visually evaluate the quality of clusters obtained by the different feature extraction methods, the extracted features were reduced from 9 to 3 dimension using PCA. As indicated by Fig. 5a, our proposed method better separates each cluster visually. The extracted features were also evaluated quantitatively using the silhouette statistic (Fig. 4b), where greater clustering quality is indicated by a score closer to 1. Our feature extraction method yielded the highest scores among other feature extraction methods tested for all numbers of source neurons except for two cases, but our classification performance was still 97%. As the number of source neurons increased and spike sorting task became more difficult, the decrease in silhouette score was smallest when using our model. The average silhouette scores on Dataset2 using our model with different source neuron numbers was 0.0871 higher than DM-DBCSAN, which yielded with the lowest score (0.6344), and still 0.0276 higher than SOM-K-means, which yielded the second best score (0.6937). Fig. 5b presents the projections of extracted features from the 4 *in vivo* datasets obtained by Huh et al. Similar to simulated data, our model accurately clustered all 4 *in vivo* datasets, with spike sorting accuracies of 97.65%, 98.8%, 93.92%, and 95.98%, respectively, and corresponding silhouette scores of 0.6291, 0.6333, 0.5334, and 0.5071 (Fig. 5c). Extracellular recordings using tetrodes or multi-electrode arrays have been widely used to study associations between neuronal activity patterns and specific responses, and our adjusted model also demonstrated outstanding performance on these data without extensive optimization. Indeed, the only adjustment made to the model structure was that the tetrode *in vivo* dataset were concatenated to form 124-D data as input instead of using the 31-D single electrode action potentials as input. Using tetrode *in vivo* dataset, spike sorting accuracy increased from 96.58% to 98.27% (+1.9%) and silhouette score increased from 0.5757 to 0.7426 (+0.1669). The results for single and multi-channel recordings are compared in Fig. 5c. This comparison also demonstrates performance increases when more information is added to the deep learning model. Therefore, it is highly likely that performance will increase further with proper optimization.

3.4. Selecting parameters for ensemble auto-encoder

Fig. 6 shows the optimal interval value for the gradient, number of latent space nodes, and number of AEs for our ensemble model. Our proposed model achieved the highest performance with an interval value of 1, while performance decreased with increasing interval values (Fig. 6a). Performance was also optimal when using 3 nodes for the latent space (Fig. 6b) and the performance decreased with increasing number of latent space nodes, reaching a plateau at 9. Therefore, each of the ensemble AE models were constructed with 3 latent space nodes based on the assumption that such architecture extract best distinguishing features. Finally, the number of AEs in the ensemble was also tested, and the results revealed that 3 AEs yielded highest accuracy for predicting the number of source neurons correctly, with 20 correct out of 20 data subsets from simulated Dataset1 (Fig. 6c). We also demonstrated the superiority of the ensemble AE by visualizing the extracted features and using silhouette scores to evaluate the cluster separability (Fig. 7). Fig. 7a shows that the ensemble AE yields the best feature extraction performance as it most distinctively clusters Difficult2 data compared to the 3 single-AE models with different architectures. Additionally, the ensemble model demonstrated better feature extraction performance according to the silhouette scores than the single-AE models on Difficult1 and 2 subsets from Dataset1, scoring the

Table 1
The results of spike sorting accuracy (%) for Dataset1 of our method and other spike sorting methods.

Dataset	Noise level	No.spikes	PCA DBSCAN	DM DBSCAN	SOM K-means	wave_clus	LDA-GMM	Ours
Easy1	0.05	2729	99.87	100	100	99.96	100	**100**
	0.1	2753	96.86	98.65	100	99.81	100	**100**
	0.15	2693	93.48	97.06	99.95	99.81	99.81	**100**
	0.2	2678	86.71	94.51	99.15	99.55	98.17	**99.96**
	0.25	2586	69.94	90.56	97.63	97.52	95.66	**100**
	0.3	2629	45.72	85.77	86.99	89.5	90.79	**99.44**
	0.35	2702	37.55	56.43	76.76	82.12	89.3	**99.63**
	0.4	2645	35.22	56.48	62.17	71.98	88.43	**99.63**
Easy2	0.05	2619	97.24	99.23	99.95	99.88	100	99.70
	0.1	2694	75.98	66.33	95.9	99.62	100	**100**
	0.15	2648	41.18	60.19	87.35	98.3	99.81	**100**
	0.2	2715	39.46	55.46	79.5	88.72	98.6	**100**
Difficult1	0.05	2535	93.87	98.26	100	100	100	**100**
	0.1	2742	71.66	89.67	98.32	98.44	100	**100**
	0.15	2631	41.65	53.85	92.87	96.95	100	**100**
	0.2	2716	37.70	48.67	79.63	75.19	99.88	**99.92**
Difficult2	0.05	2616	84.28	98.01	98.61	100	100	**100**
	0.1	2638	34.38	52.69	44.56	99.7	100	**100**
	0.15	2660	34.40	36.61	40.97	83.16	99.92	99.36
	0.2	2624	34.98	37.76	40.18	46.17	98.3	**99.28**
Average			62.60	73.80	84.02	91.32	97.93	**99.81**

Table 2
The result of spike sorting accuracy (%) for Dataset2 of our method and other spike sorting methods.

No. source neurons	No. spikes	PCA DBSCAN	DM DBSCAN	SOM K-means	wave_clus	LDA-GMM	Ours
2	800	95.31	87.78	99.92	92.46	92.74	97.65
3	1200	89.36	92.52	85.82	93.03	86.91	**94.62**
4	1600	80.60	85.14	63.61	93.67	85.09	**94.25**
5	2000	74.42	72.04	54.77	90.21	85.49	**90.49**

highest on most of the subsets (Fig. 7b). Moreover, the single-AE models showed steeper decreases in silhouette score with increasing noise levels for Difficult2 data, characterized by high spike similarity, compared to the ensemble model. Furthermore, only the ensemble model yielded an averaged silhouette score over 0.5.

3.5. Deep learning-based spike attention mapping

Deep learning demonstrates high classification performance for a variety of tasks (Eo et al., 2018; Park et al., 2020). However, there are "black box" problems where the computational processes of the deep learning model are opaque (Burrell, 2016; Gunning, 2017; Ribeiro, Singh, & Guestrin, 2016). Many recent artificial intelligence (AI) studies have attempted to address this problem through feature identification, heat maps, and diagnostic classification (Bau et al., 2018; Hupkes, Veldhoen, & Zuidema, 2018; Zhou, Khosla, Lapedriza, Oliva, & Torralba, 2016). The conventional method for attention mapping is to change the input data and check the difference in output. Likewise, we applied a similar idea to 1-D spike data to explain the workings of our deep learning method. Our deep learning model performs feature extraction using the 3 AEs, each composed of different number of layers. Each AE consists of an encoder section that reduces the dimensions of the data and a decoder section that uses the dimensionally reduced data as input (the latent space) and attempts to best reconstruct it to match the encoder input. For our model, the number of nodes for the latent space is set to 3, and these 3 nodes can be considered the most compressive representations of the spikes. By altering the node values of the latent space and analyzing the changes induced in decoder output, we can understand the computation process of the model. Fig. 8, referred to as an attention mapping, is an example of such an analysis as it shows the changes in decoder output in the form of a heat map. The details of the procedure are as follows. First, given a fully trained model, a single specific case is provided as input and 3 latent space node values are obtained. Second, one of the 3 latent space node values is replaced by the minimum node value from the dataset within the same cluster, excluding the other two node values. Third, the decoded output is obtained with the adjusted node value. This process is repeated nineteen times with the replaced node value uniformly increasing so that the last replacing value is the maximum node value within the same cluster. This results in twenty waveform outputs that can be used to create a 1D attention mapping of the variance for each sampling point of time frame. We speculate that these variance values help reveal the spike waveform reconstruction qualities induced by the latent space nodes. Moreover, due to the structure of AE, where data X is compressed to the latent space h and then reconstructed back to X, the reconstruction quality reveals the compression quality as well under the assumption that a perfectly trained AE yields no loss of information. This allows us to estimate the key regions of the waveform to which the AE model is attending, and hence the name "attention mapping" is given to the variance heat mapping. This process can be applied to each of the nine latent space nodes, and the nine attention mappings can help explain how each node affects the model. A few example attention mappings are shown in Fig. 8a. Here, the geometric features of the spike considered highly informative for spike sorting, such as the peak, valley, and steepness (Quiroga, 2012; Rey, Pedreira, & Quiroga, 2015), are heavily weighted by the attention mapping. This demonstrates that our ensemble AE model attends to the key features considered by most spike classification studies. For a more quantitative analysis, we also calculated the correlations between the sampling point of the highest variance value in the attention mapping with the sampling points of the peak and valley, where a sampling point of the maximum variance value within ± 2 of the peak or valley is considered a correlation. For Dataset1, Dataset2, and the *in vivo* data, 76.39% of peak variance values were correlated with the

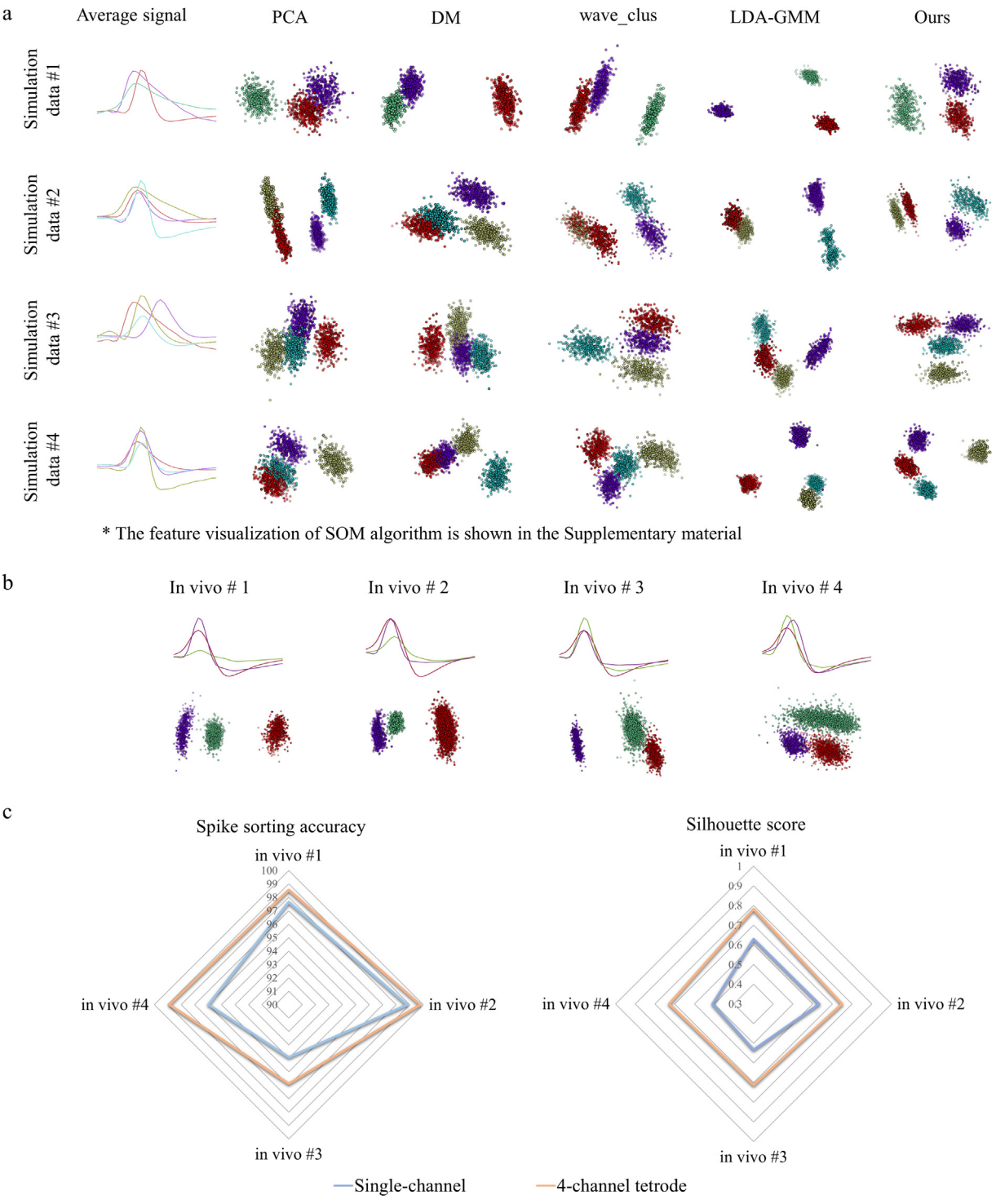

Fig. 5. Projections of extracted features of Dataset2 and *in vivo* dataset. The performance comparison of single-channel and 4-channel tetrode recordings. (a) Projection of extracted features data from all the compared spike sorting algorithms on Dataset2. (b) Projection of extracted features data from our proposed method on *in vivo* dataset. (c) The spike sorting accuracy and silhouette score results from 2 cases, a single-channel and 4-channel tetrode recordings, using the proposed method are shown. The blue line and orange line represent results on the single-channel recordings and 4-channel tetrode recordings, respectively. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

peak and 20.23% with the valley. The probability of either one of these values considered was 84.67%. Thus, in over 84% of cases, our deep learning model attends to the peak or valley of the spike. However, this does not mean that attention is restricted to peaks and valleys. As shown in Fig. 8b, heavy attention was also paid to the epoch prior to the spike and to the tail portions of the spike, such as the after-hyperpolarization, which may also be useful for spike analysis and sorting (Melonakos, White, & Fernandez, 2016). Overall, the attention mapping helps reveal the logic behind deep learning, shows that the model focuses on spike characteristics used in previous spike signal analyses, and automatically determines and considers the discriminative properties of the spike for clustering. However, it is also important to note that even for a single node from an identical model, the

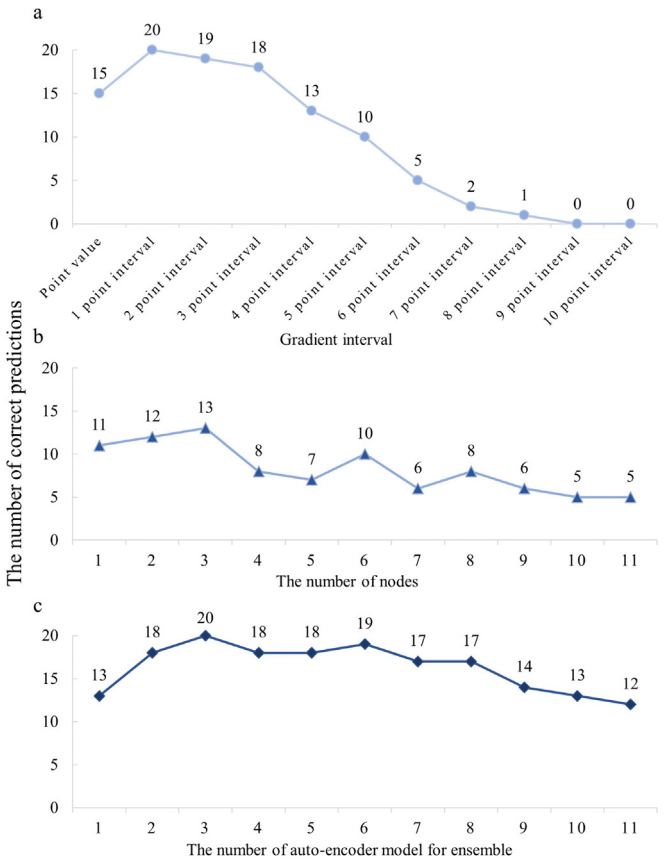

Fig. 6. Selecting the optimal parameters for gradient interval value, number of nodes in the latent space, and number of auto-encoder models for predicting the number of source neurons in the dataset. (a) Accuracy for determining the number of source neurons in the dataset using different interval values for gradient. (b) Accuracy using different numbers of nodes in the latent space. (c) Accuracy using different numbers of auto-encoder models (Ensembles).

attention mapping may change depending on the dataset or the cluster to which it belongs. The performance of our proposed deep learning-based method was also compared to a number of ML algorithms, mainly on single-channel data. The attention mapping of the ensemble AE for 4-channel tetrode recordings resembles that of the single-channel recordings (Fig. 8b) and also demonstrates strong correlations with spike peak and valley. Our explainable AI is one of many possible approaches available.

4. Conclusions and future work

Our AE-based spike sorting algorithm allows for accurate automatic classification of neuronal spikes without advanced knowledge of optimal distinguishing features. Compared to other ML-driven methods, such as diffusion map, SOM, wave_clus, and LDA, our model requires almost no parameter tuning and still achieves superior classification performance on datasets with different degrees of spike shape similarities, noise levels, and numbers of source neurons. Moreover, the most critical aspect of our AE model, feature extraction using latent space in AE, is a fully automatic procedure requiring no parameter tuning for any dataset. This allows for wide application in other spike sorting tasks with potentially greater accuracy due superior data clustering. By visualizing the reconstructive qualities of the AE latent space using AE attention mappings, we were able to identify the specific spike regions (components) considered by the model to project the data into lower dimensions. Not only does this help address the black box problem of deep learning, it also allows users to inspect the credibility of the spike sorting results through manual analysis of the attention mappings. One downside of deep learning-based model is the need for proper training data. However, because our model is an unsupervised model where the training data need not be manually labeled, the cost of data labeling does not exist. As a matter of fact, the test data is the training data, which allows the model to be trained and optimized for each dataset. Therefore, unlike other supervised deep learning classification models, our model is free of performance drop from lack of training data, and it is expected to show robustness for all types of spike data. We assume that this data specificity contributes to the superior spike sorting performance compared to other ML methods. While all ML spike sorting methods depend on specific predetermined features, our deep learning model can discover and select new optimal features for extraction from each dataset. However, our model does require training for each test dataset, resulting in additional computational and time costs. Due to the simple structure of our model, however, the estimated time cost for training 3526 spikes was only 68.81 s. Our model is currently designed and optimized for classifying different types of spikes from single-channel recordings as it is assumed that less data makes this task more challenging. However, increasing the dimensions of the input data to utilize the tetrode *in vivo* dataset resulted in a 1.9% increase in spike sorting accuracy and a 0.1669 increase in silhouette score without additional optimization. It is assumed that with proper adjustment of the model structure and increased input data dimensions, for example adding

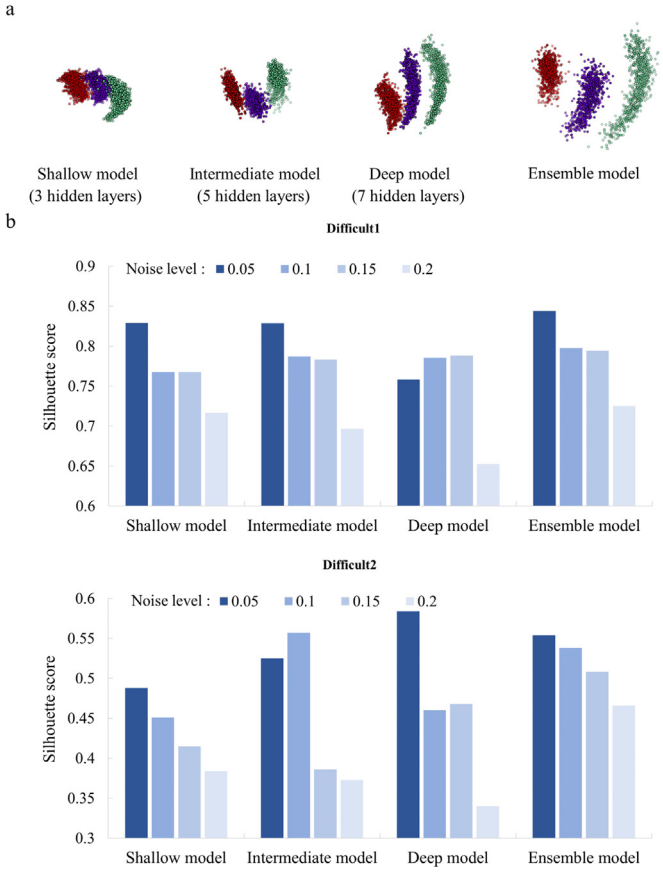

Fig. 7. Projections and silhouette scores of the extracted features from single auto-encoders and ensemble auto-encoder. (a) Projections of extracted features from each auto-encoder with Difficult2 noise level 0.15. (b) Silhouette scores of extracted features from each auto-encoder with Difficult1 and 2.

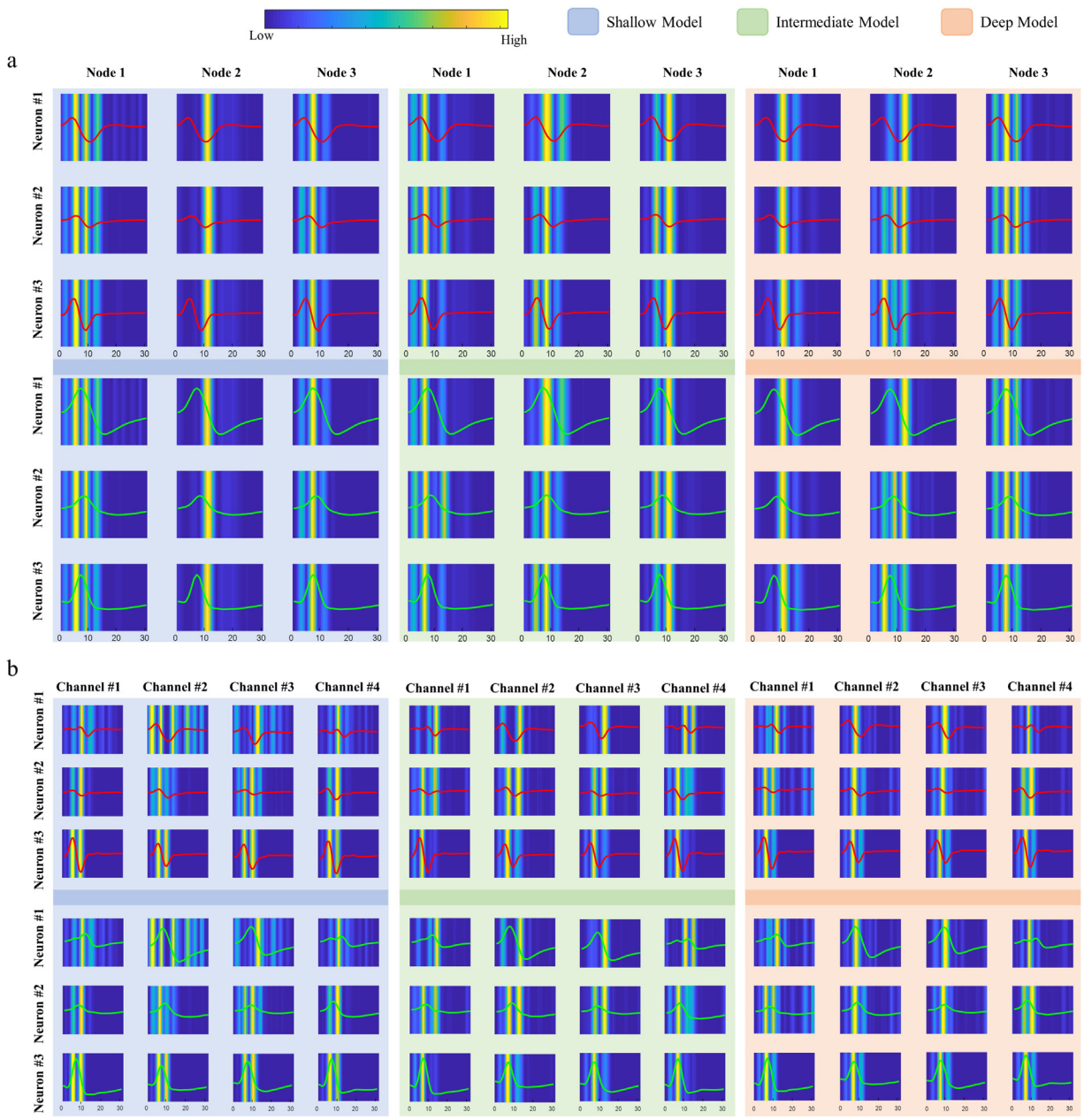

Fig. 8. Attention mapping on spikes using auto-encoders. (a) Attention mapping of all latent space nodes in a deep learning-based ensemble auto-encoder for 3 spikes each from different source neurons of the single-channel recordings. Spikes are projected onto the attention mapping, with the red lines displaying the gradient data used as auto-encoder input and the green lines displaying the original spikes recorded using a single-channel tetrode. (b) Attention mapping of 3 latent space nodes, Node 1 from each different auto-encoder model, in a deep learning-based ensemble auto-encoder for 3 spikes each from different source neurons of the 4-channel tetrode recordings. Spikes are projected onto the attention mapping, with the red lines displaying the gradient data used as auto-encoder input and the green lines displaying the original spikes recorded using a 4-channel tetrode. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

a convolutional layer, the spike sorting performance can be further improved. For our future work, new model structures for multi-electrode array spike data will be compared to other multi-electrode array spike sorting algorithms. Decreased performance as the number of source neurons increases is a common problem among all spike sorting methods. Although our model still showed highest spike sorting performance among tested models as the number of source neurons increased, there was still a noticeable drop in accuracy. We speculate that this performance decrease is caused mainly by limitations of the sorting algorithm and not the feature extraction process. As shown in Figs. 4a and 4b, the clustering performance drop is noticeably greater than the silhouette isolation score drops for dimensionally reduced data. However, as it is difficult to specify an exact correlation between the silhouette score and clustering performance, further study is needed on different clustering techniques for varying numbers of source neurons. In future work, we plan to incorporate deep learning into the clustering process for improved spike sorting performance and increased robustness to number of source neurons. Further, including deep learning will eliminate all dependence on user parameters. We describe a deep learning-based AE algorithm for accurate and efficient automatic spike sorting, whose results on the tetrode recordings demonstrate its capability on multi-channel recording as well. The analyzability of

this method by manual inspection may allow deeper understanding of data and also broaden its applicability to other datasets and tasks as well.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was partially supported by Graduate School of YONSEI University Research Scholarship Grants in 2019 and the Brain Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning, South Korea (2018M3C7A1024734, 2018M3C7A1024736, 2015M3C7A1028392).

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.neunet.2020.11.009. More details are described regarding parameters of our deep learning-based model and feature visualization of SOM algorithm.

References

Adamos, D. A., Kosmidis, E. K., & Theophilidis, G. (2008). Performance evaluation of PCA-based spike sorting algorithms. *Computer Methods and Programs in Biomedicine*, *91*, 232–244.

Bah, B. (2008). *Diffusion maps: Analysis and applications*.

Bau, D., Zhu, J. -Y., Strobelt, H., Zhou, B., Tenenbaum, J. B., Freeman, W. T., et al. (2018). GAN dissection: Visualizing and understanding generative adversarial networks. arXiv preprint arXiv:1811.10597.

Blatt, M., Wiseman, S., & Domany, E. (1996). Superparamagnetic clustering of data. *Physical Review Letters*, *76*, 3251.

Burrell, J. (2016). How the machine 'thinks': Understanding opacity in machine learning algorithms. *Big Data & Society*, *3*, Article 2053951715622512.

Coifman, R. R., & Lafon, S. (2006). Diffusion maps. *Applied and Computational Harmonic Analysis*, *21*, 5–30.

Eo, T., Jun, Y., Kim, T., Jang, J., Lee, H. -J., & Hwang, D. (2018). KIKI-net: Cross-domain convolutional neural networks for reconstructing undersampled magnetic resonance images. *Magnetic Resonance in Medicine*, *80*, 2188–2201.

Ester, M., Kriegel, H. -P., Sander, J., Xu, X., et al. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *Vol. 96*, In *KDD* (pp. 226–231).

Gibson, S., Judy, J. W., & Marković, D. (2011). Spike sorting: The first step in decoding the brain: The first step in decoding the brain. *IEEE Signal Processing Magazine*, *29*, 124–143.

Guenther, F. H., Brumberg, J. S., Wright, E. J., Nieto-Castanon, A., Tourville, J. A., Panko, M., et al. (2009). A wireless brain-machine interface for real-time speech synthesis. *PLoS One*, *4*.

Gunning, D. (2017). *Explainable artificial intelligence (XAI). 2*. DARPA.

Harris, K. D., Henze, D. A., Csicsvari, J., Hirase, H., & Buzsaki, G. (2000). Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements. *Journal of Neurophysiology*, *84*, 401–414.

Huh, Y., Bhatt, R., Jung, D., Shin, H.-s., & Cho, J. (2012). Interactive responses of a thalamic neuron to formalin induced lasting pain in behaving mice. *PLoS One*, *7*.

Huh, Y., & Cho, J. (2013). Discrete pattern of burst stimulation in the ventrobasal thalamus for anti-nociception. *PLoS One*, *8*.

Huh, Y., & Cho, J. (2016). Differential responses of thalamic reticular neurons to nociception in freely behaving mice. *Frontiers in Neuroscience*, *10*, 223.

Hupkes, D., Veldhoen, S., & Zuidema, W. (2018). Visualisation and'diagnostic classifiers' reveal how recurrent and recursive neural networks process hierarchical structure. *Journal of Artificial Intelligence Research*, *61*, 907–926.

Keshtkaran, M. R., & Yang, Z. (2017). Noise-robust unsupervised spike sorting based on discriminative subspace learning with outlier handling. *Journal of Neural Engineering*, *14*, Article 036003.

Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114.

Kojima, K.-i. (1969). Proceedings of the fifth Berkeley symposium on mathematical statistics and probability. *American Journal of Human Genetics*, *21*, 407.

Lewicki, M. S. (1998). A review of methods for spike sorting: The detection and classification of neural action potentials. *Network: Computation in Neural Systems*, *9*, R53–R78.

Likas, A., Vlassis, N., & Verbeek, J. J. (2003). The global k-means clustering algorithm. *Pattern Recognition*, *36*, 451–461.

Manton, J. H., Applebaum, D., Ikeda, S., & Le Bihan, N. (2013). Introduction to the issue on differential geometry in signal processing. *IEEE Journal of Selected Topics in Signal Processing*, *7*, 573–575.

Melonakos, E. D., White, J. A., & Fernandez, F. R. (2016). Gain modulation of cholinergic neurons in the medial septum-diagonal band of Broca through hyperpolarization. *Hippocampus*, *26*, 1525–1541.

Meng, Q., Catchpoole, D., Skillicom, D., & Kennedy, P. J. (2017). Relational autoencoder for feature extraction. In *2017 international joint conference on neural networks* (pp. 364–371). IEEE.

Nadler, B., Lafon, S., Coifman, R. R., & Kevrekidis, I. G. (2006). Diffusion maps, spectral clustering and reaction coordinates of dynamical systems. *Applied and Computational Harmonic Analysis*, *21*, 113–127.

Nenadic, Z., & Burdick, J. W. (2004). Spike detection using the continuous wavelet transform. *IEEE Transactions on Biomedical Engineering*, *52*, 74–87.

Nguyen, T., Bhatti, A., Khosravi, A., Haggag, S., Creighton, D., & Nahavandi, S. (2015). Automatic spike sorting by unsupervised clustering with diffusion maps and silhouettes. *Neural Computation*, *153*, 199–210.

Nguyen, T., Khosravi, A., Creighton, D., & Nahavandi, S. (2014). Spike sorting using locality preserving projection with gap statistics and landmark-based spectral clustering. *Journal of Neuroscience Methods*, *238*, 43–53.

Obeid, I., & Wolf, P. D. (2004). Evaluation of spike-detection algorithms fora brain-machine interface application. *IEEE Transactions on Biomedical Engineering*, *51*, 905–911.

Pacella, M., Grieco, A., & Blaco, M. (2016). On the use of self-organizing map for text clustering in engineering change process analysis: A case study. *Computational Intelligence and Neuroscience*, *2016*.

Park, I. Y., Eom, J., Jang, H., Kim, S., Park, S., Huh, Y., et al. (2020). Deep learning-based template matching spike classification for extracellular recordings. *Applied Sciences*, *10*, 301.

Pedreira, C., Martinez, J., Ison, M. J., & Quiroga, R. Q. (2012). How many neurons can we see with current spike sorting algorithms? *Journal of Neuroscience Methods*, *211*, 58–65.

Quiroga, R. Q. (2012). Spike sorting. *Current Biology*, *22*, R45–R46.

Quiroga, R., Nadasdy, Z., & Ben-Shaul, Y. (2004). Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering. *Neural Computation*, *16*, 1661–1687.

Rey, H. G., Pedreira, C., & Quiroga, R. Q. (2015). Past, present and future of spike sorting techniques. *Brain Research Bulletin*, *119*, 106–117.

Reynolds, D. A. (2009). Gaussian mixture models. In *Encyclopedia of biometrics*: *Vol. 741*.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). " Why should i trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1135–1144).

Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, *20*, 53–65.

Vesanto, J., & Alhoniemi, E. (2000). Clustering of the self-organizing map. *IEEE Transactions on Neural Networks*, *11*, 586–600.

Vincent, P., Larochelle, H., Bengio, Y., & Manzagol, P. -A. (2008). Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on machine learning* (pp. 1096–1103).

Wild, J., Prekopcsak, Z., Sieger, T., Novak, D., & Jech, R. (2012). Performance comparison of extracellular spike sorting algorithms for single-channel recordings. *Journal of Neuroscience Methods*, *203*, 369–376.

Wilson, S. B., & Emerson, R. (2002). Spike detection: A review and comparison of algorithms. *Clinical Neurophysiology*, *113*, 1873–1881.

Zhang, C., Cheng, X., Liu, J., He, J., & Liu, G. (2018). Deep sparse autoencoder for feature extraction and diagnosis of locomotive adhesion status. *Journal of Control Science and Engineering*, *2018*.

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., & Torralba, A. (2016). Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2921–2929).