

Neural Spike Sorting

1 Introduction

Extracellular action potential recordings are commonly used to study the activity of the brain [1]. These extracellular action potentials, often referred to as spikes, are recorded by inserting bipolar electrodes into the extracellular tissue within the cortical region of the brain [1]. In recent years, neuroscience research has looked into the activity of individual neurons (single-unit activity) for brain-machine interfaces and the treatment of memory loss and paralysis [2]. As extracellular action potential recordings often contain several spikes from multiple neurons, a procedure is needed to separate the recordings into single-unit activity.

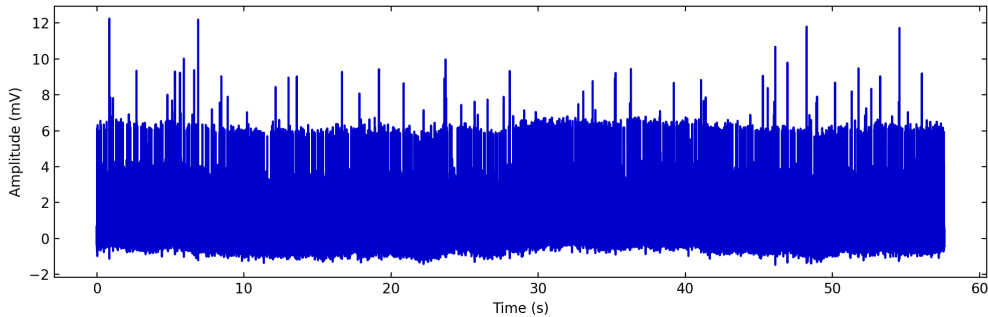
As implied by its name, spike sorting is a technique used in the analysis of spike recordings. The spike sorting procedure comprises three main processes: spike detection, spike extraction, and spike classification. Spike detection involves filtering out noise from the recordings and subsequently identifying the positions of spike occurrences [3]. As neurons often fire concurrently, many of these spikes may be overlapping. Based on the aforementioned series of positions, individual spike waveforms (single-unit activity) are extracted by taking a small segment of the signal around each spike position. Lastly, the extracted spikes are classified according to their morphology.

2 Data Processing

This section describes how spike detection and extraction were carried out. The training and submission recordings were inspected to determine the initial steps that had to be carried out before these processes could be implemented.

2.1 Normalization

After plotting the raw time domain recordings found within the training and submission datasets, presented in Figures 1a and 1b respectively, it was evident that low frequency noise, which resulted in a fluctuating signal, was present within the submission recordings. To ensure that the training and submission recordings had similar shapes, the low frequency noise was filtered out from the submission recording using a high-pass filter with a cut-off frequency of 25 Hz. The resulting normalized signal is displayed in Figure 1c.



(a) Training recording

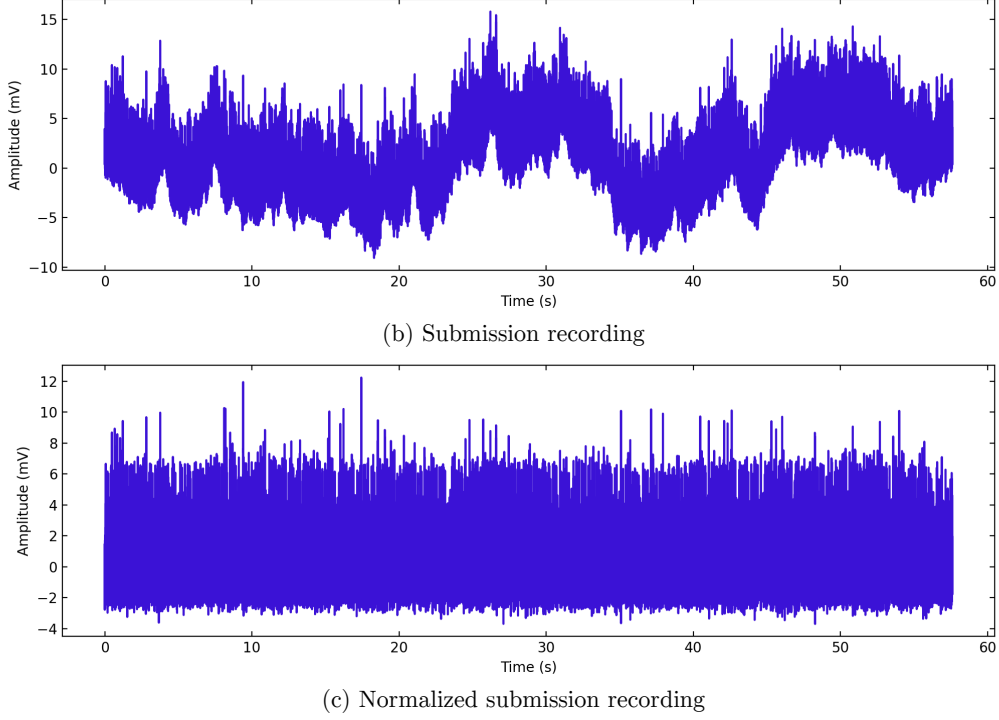


Fig. 1. Training and submission recordings

2.2 Spike Detection

Generally, spikes in an extracellular recording can be identified by their relatively large amplitudes. Thus, it can be inferred that the position of spike occurrences correspond to peaks in the raw time domain recordings. These peaks are detected by setting a threshold based on the median absolute deviation (Eq. 1) of the data, a robust measure of the spread of data that is often used to detect outliers [4], where all local maxima above said threshold are identified as peaks. Figure 4 displays peaks detected on the unfiltered training data.

$$\text{MAD} = \text{median} \left\{ \frac{|x|}{0.6745} \right\} \quad (1) \quad [4]$$

Although a threshold equal to $5 \times \text{MAD}$ was used, the peak detector identified multiple peaks for each observable spike, of which there are a total of 3704 in the training dataset. This is primarily due to the high frequency noise present in the signal, which results in multiple local maxima on each individual spike. A low-pass Butterworth filter is therefore used to attenuate the high frequency signals and ensure that any detected peaks correspond to individual spikes marked by the index values. In order to select a cut-off frequency, the number of detected peaks for a range of cut-off frequencies was determined. These values are presented in Table 1.

Table 1. Number of detected peaks for cut-off frequencies of 1000 Hz to 3500 Hz

Frequency (Hz)	1000	1500	2000	2500	3000	3500
# of Peaks	3525	3556	3581	3657	3788	4054

Evidently, a cut-off frequency of 2500 resulted in a total number of peaks closest to the ground truth spike total of 3704. The discrepancy between the two values is likely a result of overlapping spikes. As evidenced by Figure 5, increasing the frequency beyond 2500 Hz, which resulted in a greater total number of peaks detected, did not identify multiple peaks for these overlapping spikes. Thus, it can be inferred that the additional detected peaks are a result of the high frequency noise and do not correspond to additional spikes.

With regards to the submission recordings, the filtering and spike detection processes were carried out in a similar manner. However, as the subject was moving when these recordings were being made, the high frequency noise is more apparent when compared to the training recordings; thus, a lower cut-off frequency is needed. As the total number of spikes within the submission recording is unknown, the power spectrum (Fig. 2) of the signal was inspected to determine the ideal cut-off frequency. As the spectrum flattens beyond around 1700-2100 Hz, the cut-off frequency was selected as 1900 Hz.

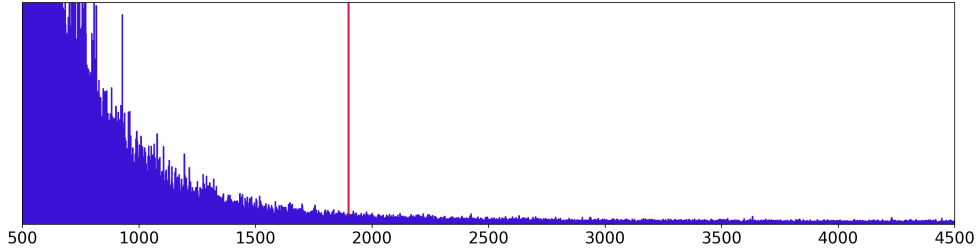
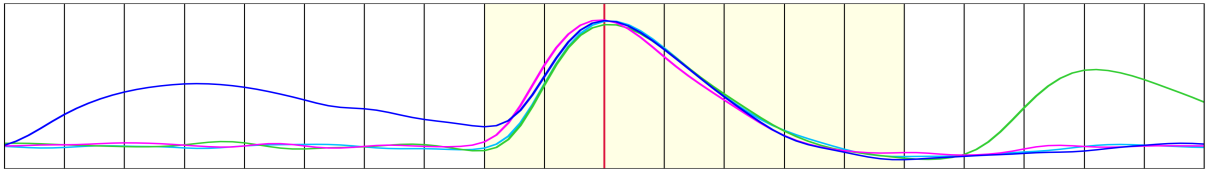


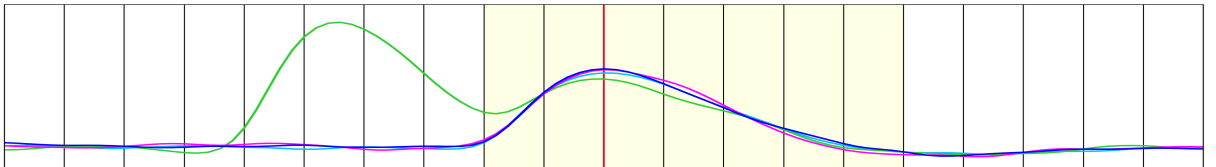
Fig. 2. Power spectrum of submission recording

2.3 Spike Extraction

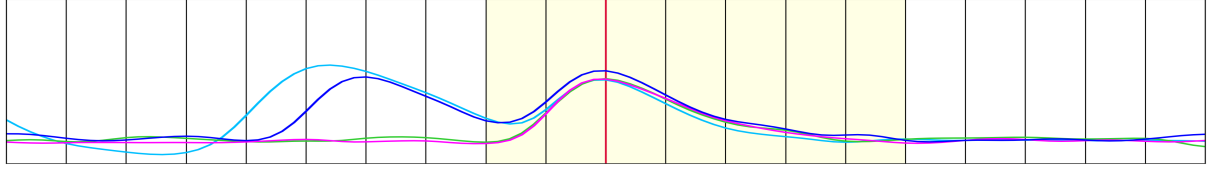
To determine the number of data points (i.e. window) to extract for model fitting and classification, the distributions of four random spikes, around their peaks, for each type of neuron were plotted in Figure 3. Through manual inspection of these plots, it was determined that 41 data points (15 points to the left, 25 to the right) would sufficiently capture the main features of each type of spike without including neighboring peaks.



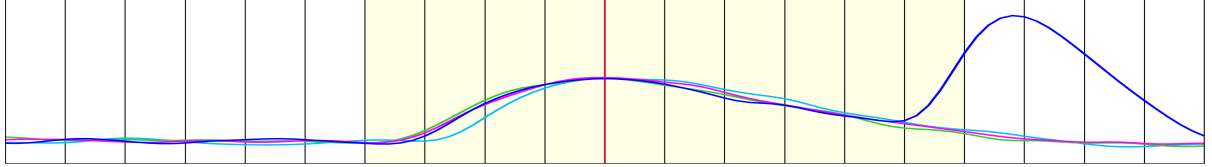
(a) Spikes produced by type 1 neurons



(b) Spikes produced by type 2 neurons



(c) Spikes produced by type 3 neurons



(d) Spikes produced by type 4 neurons

Fig. 3. Distribution of spikes around the peaks for the four spike types. The vertical lines are spaced 5 data points apart from one another.

For each extracted spike in the training recording, the corresponding neuron type was found by first identifying the index value within the Index vector closest to the peak of the spike. The neuron type was then found by finding the value in the Class vector in the same position as the identified index value. A total of 69 duplicates (i.e. spikes corresponding to the same index value) were discovered and subsequently deleted, resulting in a final training dataset of 3588 spikes. The distribution of neuron classes is displayed in Table 2.

Table 2. Distribution of neuron classes in the dataset

Neuron Class	1	2	3	4
# of Spikes	859	925	925	879
Proportion	23.9%	25.8%	25.8%	24.5%

3 Performance Metrics

Various metrics exist for evaluating the performance of classification techniques. Generally, these metrics are extracted from a table known as a confusion matrix. Confusion matrices provide a visual interpretation of the quality of the output of a classifier [5]. Figure 4 illustrates a confusion matrix for a multi-class classification problem with respect to Class 1. Each row of the matrix represents the instances in an actual class, whereas each column represents the instances in a predicted class [6]. The data within the matrix is indicative of four outcomes: true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). For a given class, a true positive is an outcome where the model correctly predicts said class, whereas a true negative is an outcome where the model correctly predicts false classes [6]. On the other hand, a false positive is an outcome where the model incorrectly predicts the given class, whereas a false negative is an outcome where the model

incorrectly predicts other classes [6]. From these outcomes, a number of metrics can be derived. The most notable of these metrics are accuracy, precision, and recall, given by Equations 2, 3, and 4 [6] respectively.





		TP 	TN 	FP 	FP 
	Class 1	149	0	3	0
	Class 2	4	164	2	1
	Class 3	1	2	210	1
	Class 4	0	1	1	130
Actual label		Class 1	Class 2	Class 3	Class 4
		Predicted label			

Fig. 4. Multi-class confusion matrix

$$A = \frac{TP_1 + TP_2 + TP_3 + TP_4}{\text{Total Predictions}} \quad (2) \quad P_n = \frac{TP_n}{TP_n + FP_n} \quad (3) \quad R_n = \frac{TP_n}{TP_n + FN_n} \quad (4)$$

Accuracy, often considered the most intuitive performance metric as it simply measures the fraction of correct predictions, is only appropriate when the distribution of classes within the dataset is nearly balanced [6]. Therefore, as the neuron classes are distributed evenly (Table 2), accuracy was selected as the primary performance metric due to its simplicity. By increasing accuracy, the total number of correctly labeled classes would increase. However, as an increase in accuracy may not result in a subsequent decrease in either false positives or negatives, a second metric that takes both precision and recall into consideration would be advantageous. Rather than considering each of these metrics individually, the F_1 score (Eq. 5), which is the harmonic mean of the average precision and recall of a classifier [7], can be used. This metric can therefore be targeted to reduce the overall number of misclassified classes caused by either poor precision or poor recall. As this metric is generally used for unbalanced datasets, it will only be used as a secondary performance metric in case splitting the dataset results in an unbalanced train or test subset.

$$F_1 = \frac{2 \times P_{\text{avg}} \times R_{\text{avg}}}{P_{\text{avg}} + R_{\text{avg}}} \quad (5) \quad [8]$$

4 Classification

The two computational intelligence techniques that were implemented and tested for the purpose of classification were (1) a feed-forward artificial neural network, and (2) the k-nearest neighbors (KNN) algorithm. In both cases, the training dataset was shuffled in a reproducible manner to ensure consistency and then split into train and test subsets with a standard 80-20 split [6]. Furthermore, parameters were either set to default values (KNN) or drawn from literature (ANN).

4.1 Artificial Neural Network

Neural networks are universal functional approximations and can therefore approximate to any function, given enough data and computational power [8]. As classification problems involve finding an underlying functional relationship between a class membership and the features of an object [8], it can be inferred that neural networks are appropriate for the classification of spikes according to the neuron that produced them. The main characteristics of the network are discussed below.

Structure — It has been widely regarded that a multilayer perceptron (MLP) with a single hidden layer is sufficient for most classification problems [9]. Additionally, it has often been cited that the number of nodes within said layer should be as low as possible to ensure that the network can perform well on unseen data [9], and is generally set as half the number of input features for initial models [10]. Thus, as the network inputs have 41 data points, the resulting network structure consists of 41 input nodes, 20 hidden nodes, and 4 output nodes corresponding to the four neuron classes.

Activation & Loss — One-hot encoding was used to transform the categorical class labels into one-hot vectors (i.e. $2 \rightarrow 0, 1, 0, 0$) to improve the overall performance of the classifier [11]. For multi-class classification problems with mutually exclusive output classes encoded as one-hot vectors, the general convention is to use the softmax function as the activation function in the output layer of the neural network, and categorical cross-entropy as the loss function [11]. As each spike can only belong to a single neuron class, the aforementioned functions were used.

Optimizer — With batch gradient descent methods, a lot of pruning and optimization is required to determine the ideal learning rate for a classification problem. Adam optimization, a stochastic gradient descent method, was therefore used in place of batch gradient descent as it is fairly robust to the choice of learning rate [9]. The learning rate was therefore set as the default Adam optimizer value of 0.01.

Keras was used to build this model due to its focus on quick deployment and fast experimentation. The model was then trained for 50 epochs and subsequently tested a total of 5 times. As seen on Table 3, due to the random initialization of weights, the accuracy and F_1 score values differed, albeit slightly, from one run to the next. The proposed network structure resulted in an average accuracy of 0.9849 and an F_1 score of 0.9847. The lower F_1 score indicates the presence of several false positives or false negatives due to misclassified data. The misclassified spikes for one of the five trials, plotted according to their predicted classes, are displayed in Figure 5.

Table 3. Performance results for the artificial neural network

Trial #	1	2	3	4	5	Avg.
Accuracy	0.9863	0.9849	0.9877	0.9822	0.9836	0.9849
F_1 score	0.9860	0.9848	0.9875	0.9819	0.9831	0.9847

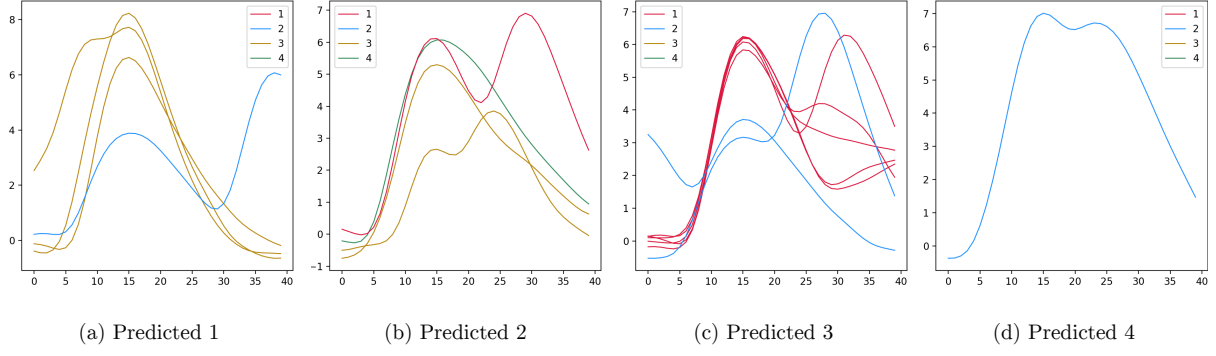


Fig. 5. Spikes misclassified by the artificial neural network

As overlapping spikes, which are often characterized by high amplitudes (> 6.5 mV) or the presence of multiple neighboring peaks and large widths, contain features associated with multiple neuron classes, the network faced the greatest difficulty in classifying them. As the network may also associate the features of these overlapping spikes with multiple neuron classes, their presence within the dataset would impair the ability of the network to perform well on unseen noisy data, such as that of the submission recording.

4.2 K-Nearest Neighbors

The k-nearest neighbors algorithm is widely used for classification problems due to its simplicity and ease-of-implementation. This algorithm was selected as the second technique due to its dissimilarity to neural networks, whereby it simply measures the distance between a test input and all training inputs to make predictions [12]. It is essentially a measure of similarity and predicts the class of an unseen input based on the classes of the k-nearest training inputs. Hence, this technique is less likely to be affected by the presence of overlapping spikes within the dataset, as they form a minority of the extracted spikes within the dataset.

As the spikes within both the training and submission recordings can be subtly differentiated based on their morphologies, KNN can be used to determine the neuron type corresponding to a spike based on its distance to spikes with similar morphologies. The KNN classifier has two main parameters: the number of nearest neighbors (k) and the distance metric (p). Assuming constant parameters, the predictions made by a KNN classifier remain consistent across trials. For evaluation purposes, the default parameter values of 5 neighbors and a Euclidian distance metric ($p = 2$) were used. This resulted in an identical accuracy and F_1 score of 0.9822.

From both Figures 6 and 7, it is evident that KNN primarily faced difficulties with spikes produced by type 2 and type 3 neurons, and often misclassified them as being produced by type 1 neurons. This could be due to the similarities between the shapes of type 1, type 2, and type 3 spikes. Optimizing the parameters of this classifier, particularly the number of neighbors, could ensure that these spikes are not placed in the wrong categories. On the other hand, as expected, the classifier did not have much trouble with overlapping spikes.

Actual label	Type 1	171	2	0	0
	Type 2	2	192	2	1
	Type 3	3	1	167	1
	Type 4	0	1	0	186
		Type 1	Type 2	Type 3	Type 4
		Predicted label			

Fig. 6. Confusion matrix produced by the k-nearest neighbors algorithm

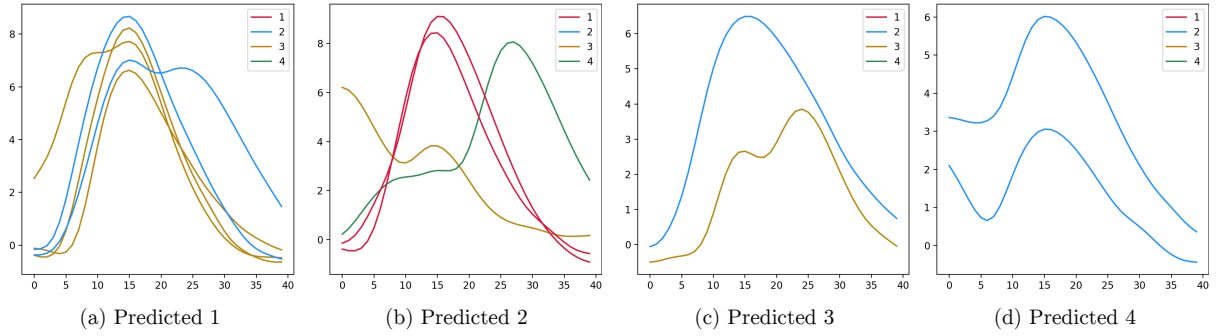


Fig. 7. Spikes misclassified by the k-nearest neighbors algorithm

5 Optimization

Although the artificial neural network had a slightly better performance with regards to the selected metrics, k-nearest neighbors was selected for further optimization for several reasons. As the primary flaw of the compiled neural network was its misclassification of overlapping peaks, a consequence of limited data, further hyper-parameter tuning would not significantly diminish this shortcoming. On the other hand, as KNN primarily misclassified individual spikes, there is greater potential for improvement through parameter optimization. Additionally, KNN has the benefits of maintaining consistency between trials and having two integer-valued parameters.

As the parameters can only take discrete values, simulated annealing, a technique used to approximate to the global optimum of a given function, was used to optimize the classifier due to its ability to extensively examine a discrete search space for an optimal solution in relatively low computational times [13]. As selecting optimal parameters based on the performance of the classifier on the test subset would likely result in overfitting, cross-validation was used for parameter tuning. This was achieved by splitting the training subset into the actual train set and a validation set. Due to the apparent uniformity between the two performance metrics, the cost function for the simulated annealing algorithm was derived based on the accuracy of the validation set predictions.

Prior to optimization, principle component analysis, a dimensionality reduction technique, was utilized to speed up computation. To ensure that the significant features of the spikes were preserved, the number of principle components was set to capture 99% of the explained

variance of the train and validation sets. After 100 iterations of simulated annealing, one neighbor ($k = 1$) and the Euclidean distance metric ($p = 2$) were identified as the optimal parameters. Utilizing the reproducible shuffled dataset from Section 4, these parameters resulted in an improved accuracy of 0.9891. Additionally, by comparing Figures 7 and 8, it is evident that optimizing the number of neighbors improved the overall ability of the classifier to distinguish between type 1, type 2, and type 3 spikes.

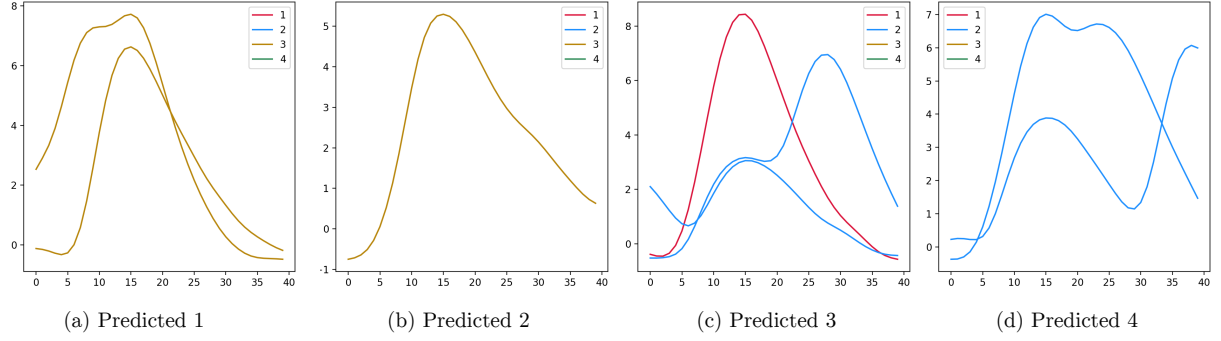


Fig. 8 — Spikes misclassified by the optimized k-nearest neighbors algorithm

However, to both minimize the impact of anomalies such as the overlapping spikes on the classification of spikes within the submission recording and retain the benefits of the optimized KNN classifier, a number of neighbors between the default value of five and the optimal value of one is preferred [14]. Thus, the parameter was set to a value of three when generating the Class and Index vectors from the submission dataset.

6 Concluding Remarks

Due to its wide-ranging applications, neuronal spike sorting has become a trending research topic in modern neuroscience. This report explored the design of a system that automatically analyzes a set of extracellular recording to detect, extract, and classify individual spikes based on their morphologies. Two computational intelligence techniques, namely an artificial neural network and the k-nearest neighbors algorithm, were evaluated for spike classification purposes after the spikes were detected and extracted. Although the former performed slightly better with regards to the selected metrics, k-nearest neighbors was further optimized due to its consistency, simplicity, and potential for improvement.

After the parameters were tuned through simulated annealing, the performance of the k-nearest neighbors classifier on the submission dataset was qualitatively appraised by plotting 50 randomly selected spikes from each of the predicted classes with a reference spike (thicker black spike) of the same class from the training dataset (Fig. 14). The morphologies of the majority of the predicted spikes and the reference spikes were very similar. Thus, one can be very confident in the performance of the classifier on the submission dataset. The spikes that do not match the observed morphology within each of the four plots appear to have high amplitudes, multiple neighboring peaks, or large widths, characteristics of overlapping spikes.

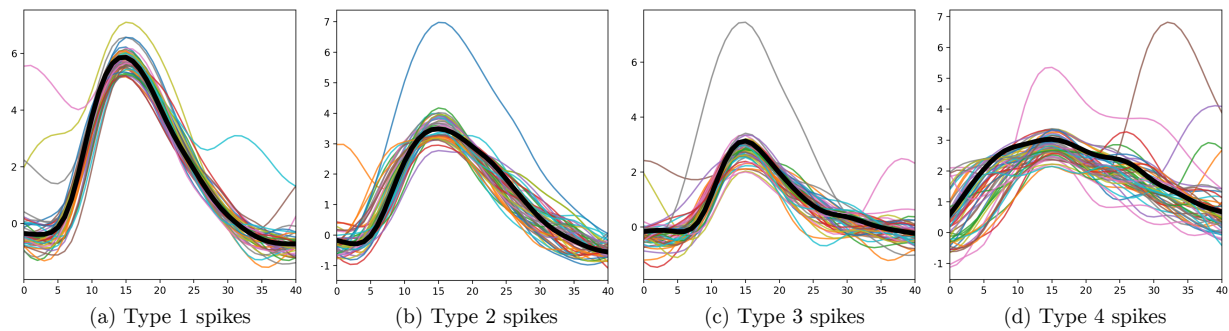


Fig. 14 — Classification of spikes within the submission dataset (colored lines) with reference to known spikes from the training dataset (black line)

References

- [1] C. Gold, D. Henze, C. Koch and G. Buzsáki, "On the Origin of the Extracellular Action Potential Waveform: A Modeling Study", *Journal of Neurophysiology*, vol. 95, no. 5, pp. 3113-3128, 2006.
- [2] R. Mukamel and I. Fried, "Human Intracranial Recordings and Cognitive Neuroscience", *Annual Review of Psychology*, vol. 63, no. 1, pp. 511-537, 2012.
- [3] I. Park et al., "Deep Learning-Based Template Matching Spike Classification for Extracellular Recordings", *Applied Sciences*, vol. 10, no. 1, p. 301, 2019.
- [4] S. Kim and J. McNames, "Automatic spike detection based on adaptive template matching for extracellular neural recordings", *Journal of Neuroscience Methods*, vol. 165, no. 2, pp. 165-174, 2007.
- [5] S. Stehman, "Selecting and interpreting measures of thematic classification accuracy", *Remote Sensing of Environment*, vol. 62, no. 1, pp. 77-89, 1997.
- [6] "Machine Learning Crash Course", Google Developers. [Online]. Available: <https://developers.google.com/machine-learning/crash-course/>. [Accessed: 05- Jan- 2021].
- [7] "F-score", *En.wikipedia.org*. [Online]. Available: <https://en.wikipedia.org/wiki/F-score>. [Accessed: 05- Jan- 2021].
- [8] G. Zhang, "Neural networks for classification: a survey", *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, vol. 30, no. 4, pp. 451-462, 2000.
- [9] I. Goodfellow, Y. Bengio and A. Courville, *Deep Learning*. 2015.
- [10] C. Ranjan, "Rules-of-thumb for building a Neural Network", *Medium*, 2019. [Online]. Available: <https://towardsdatascience.com/17-rules-of-thumb-for-building-a-neural-network-93356f9930af>. [Accessed: 06- Jan- 2021].
- [11] V. Martinek, "Cross-entropy for classification", *Medium*, 2020. [Online]. Available: <https://towardsdatascience.com/cross-entropy-for-classification-d98e7f974451>. [Accessed: 06- Jan- 2021].
- [12] N. Altman, "An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression", *The American Statistician*, vol. 46, no. 3, pp. 175-185, 1992.
- [13] "Simulated annealing", *Wikipedia*. [Online]. Available: https://en.wikipedia.org/wiki/Simulated_annealing. [Accessed: 05- Jan- 2021].

- [14] "Does the presence of the outliers affect the 1NN algorithm?", CrossValidated. [Online]. Available: <https://stats.stackexchange.com/questions/369148/does-the-presence-of-the-outliers-affect-the-1nn-algorithm>. [Accessed: 05- Jan- 2021].