Academic Journal of Nawroz University (AJNU), Vol.13, No.1, 2024

https://doi.org/10.25007/ajnu.v13n1a1798

Improving the Accuracy of Neurons Spike Sorting by Using Supervised Machine Learning

Helat Ahmed Hussein [1] and Ahmed Khorsheed Mohammed [2]

1 Department of Computer Science, College of Science, University of Duhok, Duhok, KRG – Iraq
2 Department of Electrical and Computer Engineering, College of Engineering, University of Duhok, Duhok, KRG – Iraq

ABSTRACT: The brain is important for both the functioning and reasoning ability of the body. It plays a fundamental role in the coordination of body functioning as well as reasoning and thinking. To understand how the brain is working, we need to know how neurons communicate with each other by firing (Action potential) which is known as spike. To record these activities neurologists used the multi-electrode which record thousands of spikes at the same time. Therefore, neurologists used the Spike Sorting Algorithm (SSA) to know which spike belongs to which neuron. The accuracy of the spike sorting is the most important point. Accordingly, machine learning is used to improve the accuracy of the spike sorting. In this paper, the Principal Component Analysis (PCA) is implemented to extract features and for clustering step, the Supervised Machine Learning is applied by using the Support Vector Machine (SVM) and K-Nearest Neighbors (KNN) to compare the accuracy of the supervised clustering with the Template Matching method. A comparison between the results of Applied Machine Learning is achieved at different levels of noise to check the accuracy of each algorithm. The results showed that when the noise level was low, KNN accuracy reached 100% while SVM reached 95% and template 100 %. However, when the noise level increased to 0.5, the accuracy of KNN became 94 % and template 85.6 % and SVM 90 %.

Keywords: Machine learning, Neuron's spike sorting, spike classification, Spike clustering, Support vector machine, K-Nearest Neighbors

1. Introduction

Neurons in the brain communicate with each other through synapses, and action potentials are the electrical signals that propagate along the axons to transmit information from one neuron to another at the synapses. Therefore, a key to understanding brain function is the study of neuron spikes. Spike Sorting Algorithm (SSA) is very important to study the neuron spike (Yang, 2017) (Bod et al., 2022). The feature of the nerve cell near the recording site has demonstrated that not only the direction and distance, but also the intrinsic of the recording electrodes influence the shape of the action potential(Noce, Ciancio, & Zollol, 2018). This information helps us understand the inner mechanism and the nervous system implementation(Chen et al., 2019). The growth of microfabrication technology leads to large-scale electrode arrays to record the neuron signals in many channels simultaneously(Rey, 2015).

Despite the stated benefits, there are several limitations of using the microfabricated electrode arrays for the single data collected. The first limitation is the closed–spaced recording sites which lead to a severe problem of overlapping between various spikes. The second limitation is the detection of the millivolt signal level in living biological tissues which produces low signal to noise rate at each target neuron (Chaure et al., 2018), due to remote neuron noise interference. It is a crucial requirement for neuroscientists to design an efficient and comprehensive computational solution for multi-channel neuron classification by means of the Spike Sorting Algorithm to know which spike belongs to which neuron (Zamani et al., 2020). In the typical spike sorting frameworks, there are four main steps involved. The first step is spike detection, the second one spike alignment, the third one featuring extraction, and the final step is clustering (Quian Quiroga & Nadasdy, 2004). The accuracy of the Spike Sorting Algorithm has become the most important issue in neuroscience.There are many machine-learning algorithms that are used to generate an automatic clustering and classify each spike to which neuron it belongs with a high accuracy. Therefore, machine learning has been applied to spike sorting to solve the problem of manual curation that can be time consuming and less accurate. As we can see, the authors (Sukiban et al., 2019)applied support vector for clustering and PCS for features extraction by using the datasets with easy noise to get (96.3-98.4 %) accuracy, and with the difficult noise to get a good accuracy (98.6%). However, when they applied White Gaussian (**WG)** for the feature extraction and SVM, the accuracy becam higher for the easy and difficult noise (99%). While authors (Chen et al., 2019) applied accurate supervised spike sorting based on the wavelet by compering two methods wavelet package decomposition on (WPD) and PCA. The clustering accuracy for the first methods reached 99.7% while with PCA reached 22.35%. The authors (Noce, Ciancio, & Zollol, 2018) used SVM to optimize process for clustering.

This paper introduces the implementation of machine algorithms in the sorting stages and the use of PCA for the feature extraction. The advantage of the PCA is the dimensions reduction from 64 features to 3 features only. The supervised machine learning algorithms are used with the different background noise. The accuracy of each algorithm is measured. The comparison of the results is obtained with traditional method Template Matching for the clustering of the spikes. The remaining parts of this paper are organized as follow: section II deals with spike sorting fundamentals, section III tackles the Supervised Machine Learning, section IV describes the proposed model starting from applying the PCA for feature extraction and use of different supervised algorithms like SVM and KNN for clustering and applying different rates of noise to compare the results obtained, and the last section is allocated to conclusion

2. Spike Sorting Fundamentals

Spike Sorting is a fundamental process in neuroscience that identifies and classifies neural spikes (action potential) recorded from multiple neurons at the same time. As an electrode recording detects action potential (the raw data) from several cells as shown in the figure bellow, these signals must be detached from multiple cells and from the external noise before the behaviors of each neuron can be analyzeded. Many algorithms have been suggested to do spikes sorting efficiently (Bod et al., 2022). To understand the process of spike sorting, figure (1) illustrates the major steps of spike sorting starting from the raw of the multiple neurons that has been recorded simultaneously and final steps which is clustering the spikes with similar features together and each cluster represents the action potential of one neuron.

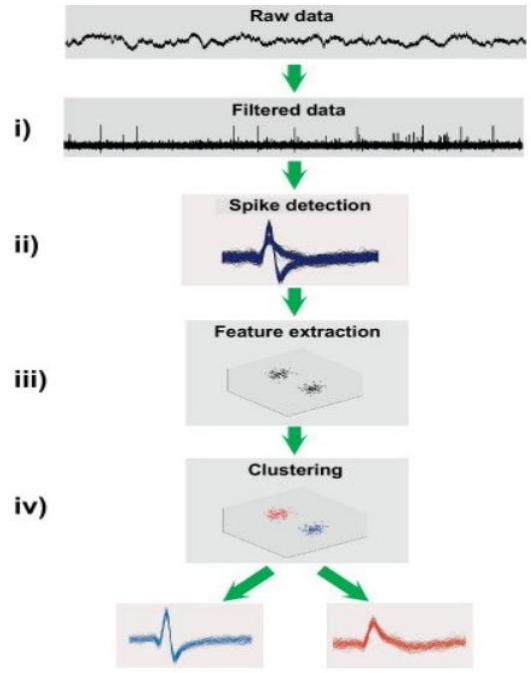


Figure 1:Spike Sorting Framework Overview *(Quian Quiroga & Nadasdy, 2004)*

Figure (1) shows the first step which is the continuous process of recoding the data and then the application of the thresholder in order to avoid low-frequency activity and in order to visualize the spikes.The second stage is to detect the spike by using thresholding. When the signal is above a certain threshold, it is considered a candidate spike. The next step is featuring extraction: relevant features are extracted from the candidate spikes depending on the characteristics of each spike like shape, amplitude, and duration. On of the common methods used in feature extraction is Principal Components for feature extraction and dimensions reduction. Finally, clustering is allocated to putative neurons by using a clustering technique based on conventions about the distribution of spikes of the same neuron in the feature space [Quian Quiroga, 2004]. There are three main steps in the spike sorting algorithm, and each of these steps will be carefully studied as its implementation will significantly affect the results.

Spikes are quickly visualized on top of the background noisy behavior after filtering and can be observed by using an amplitude threshold. For example, if the threshold value is too small, noise variations will lead to false-positive occurrences, and low amplitude spikes will be overlooked if the threshold value is too high (Rey et al., 2015). Just like the great majority of online spike detection systems, an acceptable threshold can be set manually. However, when dealing with a large number of channels, an automatic threshold is strongly advised. It can be set up so that it fluctuates by an amount that is a multiple of the signal's inherent variability (Williams et al., 2015). However, if you use the standard deviation of the signal, which accounts for spikes, you may get extremely high threshold values. This is conceivable whenever there is a high rate of firing and a sizable spike amplitude. There was a proposal to immediately set a threshold (*Thr*) in order to identify a workaround for this restriction:

$$Thr = 5\sigma_n \qquad \sigma_n = median\left\{\frac{|X|}{0.6745}\right\} \tag{1}$$

Where x is the value of the bandpass filter, and σ_n is the standard deviation of the background noise (Williams et al., 2015). They need to be processed for clustering until spikes are identified. There are two problems that require a concise explanation concerning spike storage. The first is how much data points will need to be stored. Of course, this depends on the frequency of sampling, and preferably you would like to store the whole spike form, i.e., around 2 *ms* of results. This equates to 60 data points with a sampling frequency of 30 KHz. Some function extraction approaches, such as wavelets (using a decomposition implementation of multiresolution), involve a power of 2 for the number of data points. In that case, 64 data points will be optimal for 30 *KHz* sampling (Williams et al., 2015). The second problem has to do with the spike shapes' orientation. It is necessary to match spikes to their maximum. However, the limit can be at various points in the spike form due to inadequate sampling (Mokri Y,Salazar, 2017). Then, the interposed forms can be aligned to the initial sampling rate and later decimated. The third and last step of the spike sorting procedure is to extract features of the various spike kinds. Since there are now fewer features in the space, the dimensionality has decreased from m (where m is the number of data points associated with each spike) to n. In a perfect world, one would be able to eliminate all of the noise-dominated data by extracting features that help to better separate the various clusters of spikes (Hilgen G, Sobaron, 2017). No more than two or three features should be visible on the same map, as this helps conserve computing resources and is required by certain clustering algorithms that are unable to process so many inputs in a reasonable amount of time or carry out a classification by manually choosing the cluster borders. (Noce, Ciancio, & Zollo, 2018). In comparison, removing noise-dominated inputs will undoubtedly maximize clustering results. More recently, it has been planned to use wavelets for feature extraction. The wavelet converts from a time to frequency domain (time-scale) i.e., the breakdown of the signal with the best resolution both in the time and frequency domains. However, there are many machine learning algorithms that are used for feature extraction, like the Principal Component Analysis (PCA) (Wu et al., 2017). To determine which machine learning algorithm to be used for feature extraction is a critical step because it is difficult to choose the features that better distinguish the different clusters of spike shapes. The fourth and final stage of spike sorting is to organize spikes into clusters of identical characteristics, matching the numerous neurons. Some commonly used methods are Principal Component Analysis (PCA) [Wu, 2017], and the methods that presume a Gaussian cluster distribution relying on the argument that the spike variability is calculated only by additive and Gaussian stationary background noise for a given cluster. While this assertion could be plausible in certain cases, it has been suggested that the background noise should not be portrayed as a stationary random Gaussian mechanism in general(Yang et al., 2017). The other commonly used method is clustering algorithms which focus on the nearest neighbor associations and this is another way to escape the concept of Gaussian distributions, which simply group contiguous points together, provided that the local density is greater than a certain value. Spike sorting is a rather complex mathematical problem that has drawn the interest of researchers from numerous fields. For researchers working on signal processing, especially those concerned with pattern recognition and machine learning techniques, this is indeed an important matter. Therefore, it is important to know which algorithm can be selected and also to know the accuracy of the selected methods. In the next section, we will give an overview of the machine learning algorithms (supervised and unsupervised).

3. Machine Learning

Machine Learning (ML) is often considered a subfield of Artificial Intelligence (AI) due to the fact that its algorithms can be seen as building blocks to teach computers to behave more intelligently through some manner of

generalizing, as opposed to merely storing and retrieving data items, as a database system and other applications would do. Although Machine Learning is a subset of AI. supervised, unsupervised, and reinforcement learning(Nath & Levinson, 2014) can all be used inside it as association analysis; for the sake of this inquiry, we will be focusing on supervised learning classification algorithms. The goal of supervised learning is to gain insight into data by developing a simple model of the frequency distribution of class labels in terms of predictive variables(Abdulqader et al., 2020). The generated classifier is then used to give predictions for classes when only the class label is unknown but the predictor features are known. In addition, machine learning methods are used in the clustering procedures. Whether supervised or unsupervised, different types of data call for different types of machine learning methods. Machine learning methods are depicted in figure 2(Bˆ & Mures, 2020). The image illustrates how different kinds of datasets can be used to classify different machine learning approaches. The figure also illustrates how different datasets call for different machine learning algorithms throughout clustering stages, but shows that K-means clustering is the most popular option. This study focuses mostly on the fundamentals of supervised machine learning(Muhammad & Yan, n.d.2018), with a special emphasis on KNN and SVM.

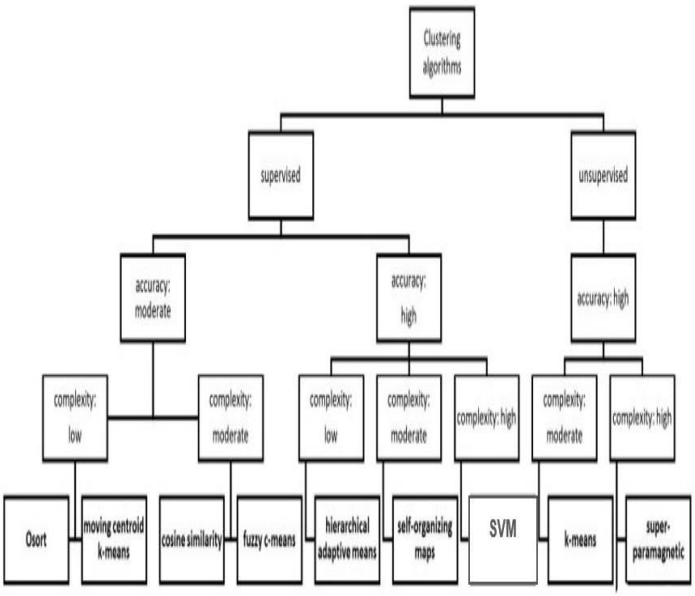


Figure 2: Types of Clustering Algorithms depending on datasets

A. Support Vector Machines (SVM)

Data classification, regression analysis, and the identification of outliers are only some of the applications of SVM, a family of supervised learning techniques. The following are some of the many potential advantages of using SVM: (I) It works well in high-dimensional spaces, and (2) it employs only a fraction of training points (support vectors), so it is memory- efficient, and can accommodate a wide variety of kernel functions, making it highly adaptable (Nath & Levinson, 2014). While predefined kernels are included, users are also given the option to define their own. No hyperplane exists that can reliably distinguish between the positive and negative examples in the training set, which is typical of most real-world scenarios. After projecting the data onto a space with more dimensions, a separating hyperplane can be defined there. There are other ways to tackle the inseparability issue, and this is one of them. This higher-dimensional region is known as the modified feature space (Takekawa et al., 2012), in contrast to the input space that the training examples inhabit, which is called the feature space. Using SVM to maximize earnings was a smart move. Improving results is crucially dependent on picking the right kernel function. This is due to the fact that the kernel function characterizes the new feature space in which the training set instances reside.

B. Nearest Neighbor's

The "nearest neighbor" method is one of the easiest and most natural approaches. The KNN classification algorithm is one of the simplest methods used in the field of data categorization technology. The K in the KNN algorithm stands for a constant chosen by the user. In order to classify the unlabeled test data, this method will assign each sample to the cluster to which the majority of the k nearest samples also belong (Kuang & Zhao, 2009).

This approach i.e., "nearest neighbor" is based on the premise that a query point *qi* in the feature space can be confidently classified as belonging to a certain category if the majority of the k samples that are most similar to *qi* in the feature space are classified as belonging to that category. This approach is known as the K Nearest Neighbor algorithm because similarity may be measured in terms of distance in the feature space. It works in the following way: To begin, you should have access to a trained dataset with well-defined labels for classes. The algorithm finds the distances between each point in the given dataset and a query dataset *qi*, where *qi* is a dataset, whose label is unknown and is presented by a vector in the feature space. This procedure is carried out on an unlabeled query dataset. The test point *qi*'s class label can be determined by looking at the labels of the k points in the trained dataset that are closest to it after the distance computation results have been sorted. The distance between two points in a multidimensional feature space can be defined in a variety of ways. The most popular approach is the Euclidean distance, which is defined as follows.

$$dist(p,q) = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2} \qquad (2)$$

In classification, the quality of the trained dataset has a direct impact on the reliability of the final results. Equally important is picking the right value for the parameter K, since doing so can cause a change in the names supplied to the various categories. Alternatively, the KNN technique can be used with high-dimensional datasets, and its computations are uncomplicated [19]. However, the computational complexity will be fairly high, and the operating time will be extremely protracted [19] if the test set, the train set, and the data dimension are all significantly larger than what was expected. figure 3 displays a flowchart of Ulsterite's KNN procedures, which may be used to learn about each individual step of the KNN algorithms (3)

Figure 3: KNN model steps flowcharts

4. System Model

The dataset used in this study depends on the same dataset which are employed by the researcher in [Quian Quiroga, 2004]. We used the semi-synthetic dataset. First, we need to identify the number of spikes. Second, the system is going to use a waveform of different spike shapes and spike amplitudes from the first firing rates of the spike. Third, the noise level will be applied. Then, the practical different level of noise (0.05, 0.15, 0.2, 0.25, 0.3, 0.35 …0.5) are used to check the accuracy of the clustering algorithm used. In the synthetic dataset, three spikes are identified with different shapes as shown in figure (4).

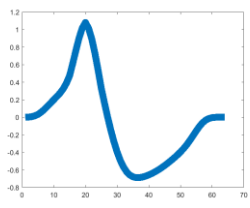
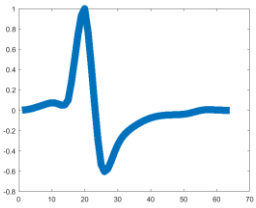
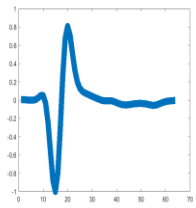
Spike 1: y1	Spike 2: y2	Spike 3: y3
		

Figure 4: The shape of the 3 spikes when sigma=1 and total number of these 3 spikes =300

The three distinct spikes were distributed by using a Poisson distribution of interspace intervals with a mean firing rate of 20 Hz. The A 2 *ms* refractory period between spikes of the same class was introduced. Therefore, the synthetic datasets are used to create 3 spikes shapes of 300 spikes and different noise levels. As can be seen, there are 3 classes of spikes and the spike time calculated in *ms*, while the spike sampling converts the time into sample number.

In this work, we used two different intelligent modeling methods. The flowchart of the system is illustrated in figure (5). First, the information was obtained from the dataset and the number of spikes to be used for the training were selected. The starting time of each spike needed to be calculated. The spikes were distributed by using Poisson. We used different level of noise (from 0.01 to 0.5). The datasets contained 64 features. We used PCA which reduced dimensions from 64 to 2 principal components (PC). Then, the spikes were classified by using three methods in order to compare the accuracy of each method using different level rates of noise. The comparison of the classification was made using (support victor machine, k nearest neighborhood (KNN) as supervised machine learning and template matching.

A. Template Matching Method

The Template Matching Method is used in this paper for the classification of the spikes depending on the shape of signal that matched a template. Each spike had its own shape as shown in figure (4). These three spikes shape needed to be applied to the 300 spikes where the noise ratio was modified from ***0.1 to 0.5 with 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45*** in between. In each time the alignment was performed to recognize the spikes from the background noise by applying the thresholding to filter the spikes from the noise as shown in figure (5).

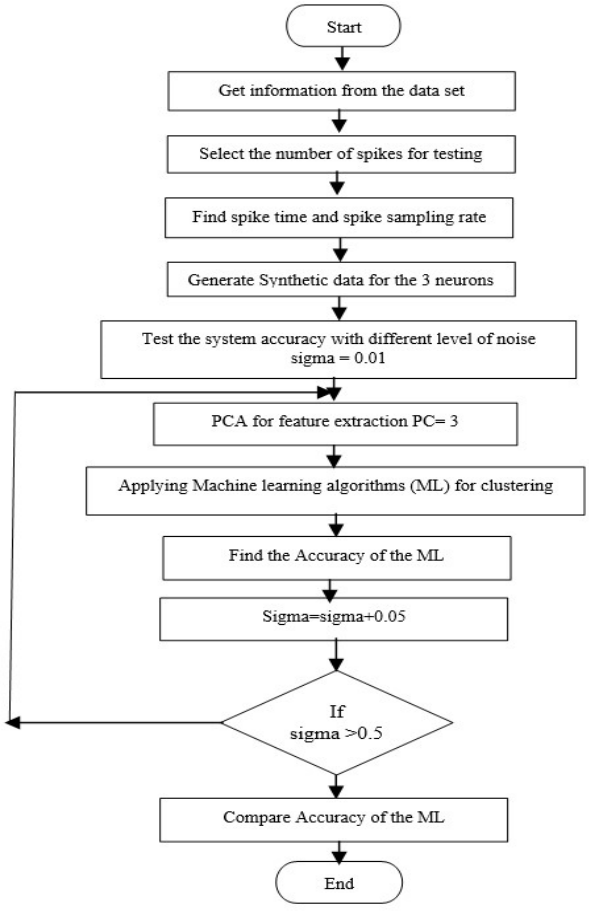


Figure 5: A flowchart for the proposed system

The process starts with generating information from the dataset and then the number of spikes is selected (here 300). The third step is to add background noise to the sequence where the sigma =0.1and sampling rate fs=30KHs. The fouth step starts by applying PCA for feature selection from the 64 spikes. Because this makes it difficult to visualized the spikes, we apply the PCA to plot the spikes into two dimensions. Thus, we select the 2 PCs instead of the 46 attributes from the datasets where the No of spike1=126, spike2=96, and spike 3=78. On the last step, we classify the results of clustering the spikes to 3 classes according to the shape of the spikes.

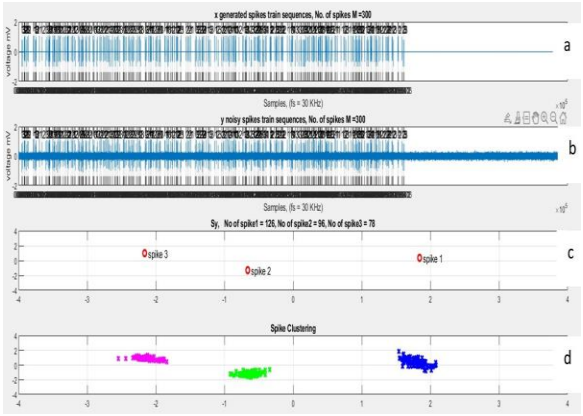


Figure 6: spike generated and classification using Template Matching Methods

When we applied different levels of noise sigma from (0.1to 0.5), the accuracy was 100 %, whereas for the sigma range from (0.1to 0.2) the accuracy dropped. For instance, when the noise level reached 0.5, the accuracy became 85.6%. As can be seen from table (2), the spike1 (sp1) and spike2 (sp2) were less affected when the noise levels became higher. On the other hand, the recall for the SP1=100% and SP2=84.6 % and SP3 decreased to reach 69.7%. The confusion matrix in figure (7) shows the application of PCA and use of two principal components (PC) at the noise level 0.35. To calculate the accuracy of the classification models, it was required to find the ratio of the correctly classified samples to the total number of the samples as shown in the equation below.

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP} \quad (3)$$

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP} \quad (4)$$

To find the Precision of the method used, the following equation was used where the TP is divided by the total number of the elements labeled as positive (the Sum of TP and FP). Once the precision is high it means that the model and the classification has more relevant results.

$$Precision = \frac{TP}{TP + FP} \quad (5)$$

The Recall (sensitivity) of the model was computed by dividing the total number of the elements that actually belonged to the positive class TP as in equation 6.

$$Recall\ (sensitivity) = \frac{TP}{TP + FN} \quad (6)$$

Figure (7) shows a confusion matrix plot, where the rows represent the output class (predicted class), and the columns represent the true class (Target Class). The diagonal cells represent the successfully categorized data points. Off-diagonal cells represent data that was improperly categorized. The total number of observations and the percentage of total observations are both shown in each cell. Number of observations is also shown.

Figure 7: confusion Matrix for Template methods where noise rate equal to 0.5 class1(spike1). Where 2 of spike1 are misclassified as spike2, 7 of spike2 also misclassified as class 3. for spike3 14 are misclassified as spike2

If you look at the right side of figure (7), you will see a column with the percentages of correct and incorrect classifications for the anticipated cases of each class. Accuracy, or positive predictive value, and reliability, or false discovery rate, are two typical names for these two metrics. Rows at the very bottom of the graph show the proportions of cases in each class that were correctly classified and those that were misclassified. Each of these quantities is frequently referred to by a different name, such as False Positive Rate (or recall) or False Negative Rate (or FNR). The overall precision is shown in the cell at the far right of the graph. The application of different noise levels for the performance of Template Matching for the three spikes is shown table (2) together with the prediction recall, precision and accuracy for each spike.

Table 1 Performance Template Matching on different noise level

Sigma	Class	Recall (%)	Precision (%)	Accuracy (%)
	Sp_1	100	100	
0.1	Sp_2	100	100	
	SP_3	100	100	100
	Sp_1	100	100	
	Sp_2	100	100	
0.15	SP_3	100	100	100
	Sp_1	100	100	
	Sp_2	100	100	
0.2	SP_3	100	100	100
	Sp_1	100	100	
	Sp_2	96.9	99	
0.25	SP_3	98.7	96	99.6
	Sp_1	100	99.2	
	Sp_2	94.1	100	
0.3	SP_3	100	93.6	98
	Sp_1	100	100	
0.35	Sp_2	93.1	97.9	97

	SP_3	97.3	91	
	Sp_1	98.3	92.1	
	Sp_2	82.7	89.6	
0.4	SP_3	84.6	84.6	89.3
	Sp_1	100	94.4	
	Sp_2	82.6	79.2	
0.45	SP_3	69.7	79.5	88.6
	Sp_1	100	94.4	
	Sp_2	84.6	79.2	
0.5	SP_3	69.7	79.5	85.6

It can be seen that accuracy, recall, and precision of each spike differ with the different levels of noise. There was small difference in the accuracy and recall of spike1. However, the recall and precision are higher compared to spike2 and spike3. While for spike3 recall and precision are the lowest. This means that the noise affects the classification of spike3 more than spike1 and spike2 with the application of different level of noise.

B. KNN

There was no specific metric to decide which algorithm to be used for a given spike dataset. Therefore, The KNN was applied. As shown in figure (3), the first step is loading the dataset 'times_C_Easy1_noise01_short'. The second step is defining k classes for $spike_1$, $spike_2$and, $spike_3$. The k points that are closest to a new point are chosen at random and studied in order to determine which category contains the highest number of points that are located in close proximity to the new point. This is done for each new point. In most cases, k is a relatively small number; hence, it is possible to assume that a hypercube may be formed by centering it on the new point and expanding it until k points are contained inside its boundaries. Next, the points are counted and it is determined from which cluster more points exist among the points of the hypercube. Thus, the new point is assigned to that cluster and its output is predicted using the method of cluster prediction. In most cases, the data are presented without providing any specifics regarding the testing sample. In practice, we create two subsamples by randomly dividing the data that we have available (the learning sample and the evaluation or test sample). Following the estimation of the model using the training set, the performance of the model is analyzed using the evaluation sample in order to eliminate any potential for bias caused by the arbitrary selection of the test population, as depicted in figure (8).

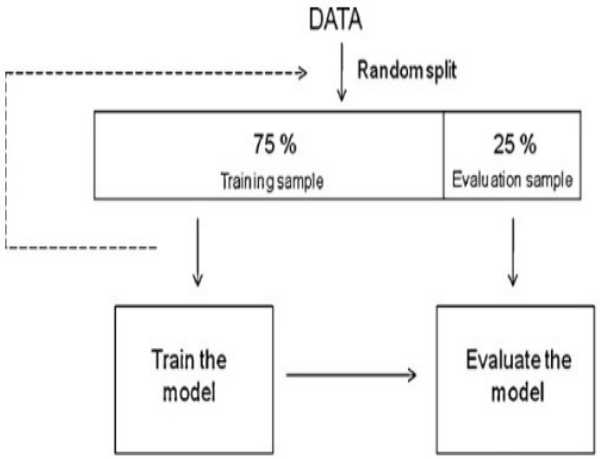


Figure 8: the dataset has been randomly splits to Training (75%) and testing data (25%)

The internal parameters of the used KNN of the current study are shown in Table 3. As obvious, the levels of noise are increased in each round of applying the method.

Table 2 :The parameters of the KNN model

Parameter	Value
Num Neighbors	5
Distance	Euclidean
Distance weight	Equal
Type	Classification
Sigma	[0.1-0.5]
Score Transform	Non
Num Observations	225

As for calculating the total accuracy from the confusion matrix to find the impact of the noise in each spike, one can see that when the noise level reached 0.5 the total accuracy reached 94 %. To spot which spike was affected more by this noise level, figure (9) is drawn which shows that spike3 correctly has been classified with accuracy of 90.1% where 9.9% of spike3 was incorrectly classified (6.2% as spike2 and 3.7% as spike1)

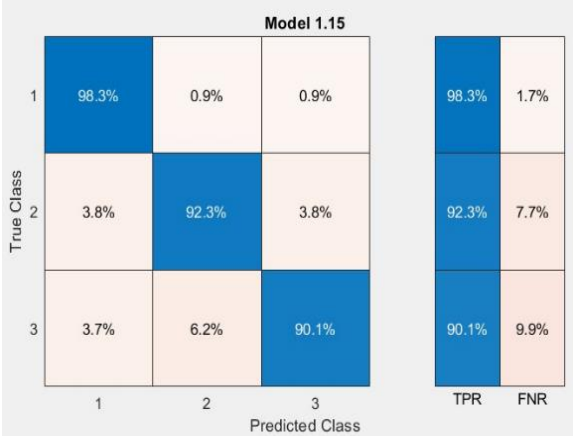


Figure 8: confusion Matrix for KNN model where noise rate 0.5. Where 1.7% of spike1 are misclassified as spike2 and spike3 0.9 for each. and 7.7 of spike2 also misclassified as spike1,3 3.8% for each class. for spike3 9.9 are misclassified as spike2 ,1

C. SVM

The SVM is a supervised machine learning method. Therefore, it depends on the dataset used and the training and testing dataset. For that reason, the distribution of the dataset should be balanced in the testing and training step. The objective of this study was to evaluate the effectiveness of SVM for spike sorting algorithm, as a modern computational intelligence method. A secondary objective was to evaluate the accuracy of SVM compared to a simpler and widely used classification technique in spike sorting such as the Nearest Neighbor. Figure (10) illustrate the block diagram of how this model works. The first step is the same as in Template Matching methods which is implemented for spike detection by applying threshold to separate the spike from the background noise and aligning them. The third step is building the model for the SVM. The same applies to KNN, where 75 % of the datasets are used for the training and 25% for testing .To select the kernel of the SVM, there are three types of the kernel ('gaussian' or 'rbf', 'linear', 'polynomial'). In this paper the linear kernel for SVM is used which has the highest accuracy rather than the other kernels.

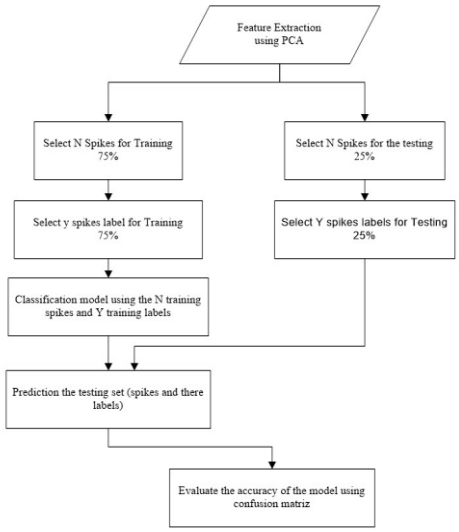


Figure 9 : SVM model block diagram

To spot the total accuracy from the confusion matrix to check the impact of the noise in each spike by using SVM, one can see that when the noise level reached 0.5, the total accuracy reached 90 %. Figure (11) shows the effect of the noise levels on the spikes. Spike3 has been classified correctly with accuracy 90.9% and 9.1% of spike3 was incorrectly classified as spike1. For spike2 93.3% was classified correctly while 6.7% was incorrectly classified as spike3. Lastly, for spike1 87% was correctly classified but 13% was classified as spike3.

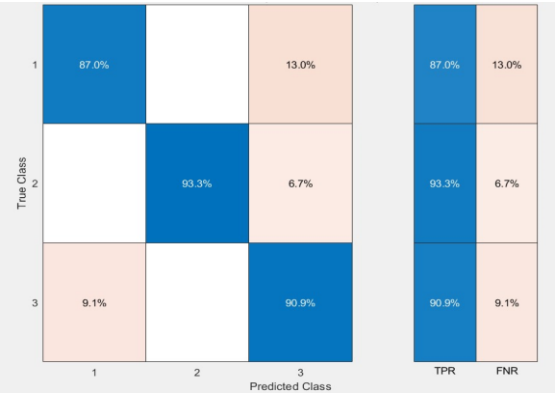


Figure 10 : confusion Matrix for SVM model where noise rate 0.5.

The total accuracy of the supervised machine learning like KNN and SVM with Template Matching using different levels of noise is described in table (4). When the noise level was low, KNN accuracy reached 100% while SVM 95% and template 100 % and when the noise level increased to 0.5, the accuracy of KNN was 94 % and template 85.6 % and SVM 90 %.

Table 3 : Comparison the Accuracy of the SVM, KNN and Template Matching with different levels of noise

Noise Rate	SVM	KNN	Template
0.05	95	100	100
0.15	95	100	100
0.2	93	100	100

0.25	93	100	99.6
0.3	93	99	98
0.35	92	98	97
0.4	92	96	89.3
0.45	91	96	88.6
0.5	90	94	85.6

As the data descriptions show, spike 3 is the mostly affected one by the level of noise with the application of the three models i.e., KNN, SVM and Template Matching. For example, in table (2), one can see that with the increase of noise, spike3 accurate classification decreases. The overall accuracy of the KNN is higher than that of the other two models as shown in table (4).

5. Conclusions

Spike Sorting Algorithm (SSA) is a very important technique to study the neuron communication in the brain. It starts with spike detection and then features extraction from the recorded neuron and then clustering of each spike that belongs to each neuron. One of the most important steps is the feature extraction and clustering. Machine learning algorithms improve accuracy with capability of automatic clustering and are better than traditional sorting methods. The supervised machine learning like SVM, KNN are applied in spike sorting in different fields. The accuracy of Templet methods and KNN are same at low levels of noise but when the noise level becomes higher, the KNN reaches a degree of accuracy higher than that of SVM and the Template Method. For low levels of noise, the Template Method and KNN accuracy reached 100 % whereas SVM reached 95% accuracy. However, when the noise level increases to 0.5, the accuracy of KNN scored 94 % and that of SVM and Template Method scored 90 % and 85.6 % respectively. For the future works there are other types of K-means that can be applied to enhance performance. There is no specific metric to decide which algorithm to be used for a given spike dataset. Therefore, it is essential to explore the data using EDA (Exploratory Data Analysis) and understand the purpose of using the dataset to come up with the best-fit algorithm.

References

[1] Abdulqader, D. M., Abdulazeez, A. M., & Zeebaree, D. Q. (2020). Machine learning supervised algorithms of gene selection: A review. *Technology Reports of Kansai University*, *62*(3), 233–244.

[2] Bˆ, H., & Mures, R. C. (2020). Machine Learning-Assisted Detection of Action Potentials in Extracellular Multi-Unit Recordings. *IEEE*, 16–25.

[3] Bod, R. B., Rokai, J., Meszéna, D., Fiáth, R., Ulbert, I., & Márton, G. (2022). From End to End: Gaining, Sorting, and Employing High-Density Neural Single Unit Recordings. *Frontiers in Neuroinformatics*, *16*(June). https://doi.org/10.3389/fninf.2022.851024

[4] Chaure, F. J., Rey, H. G., & Quian Quiroga, R. (2018). A novel and fully automatic spike-sorting implementation with variable number of features. *Journal of Neurophysiology*, *120*(4), 1859–1871. https://doi.org/10.1152/jn.00339.2018

[5] Chen, Y., Huang, L., He, J., Zhao, K., Cai, R., & Hao, Z. (2019). HASS: High Accuracy Spike Sorting with Wavelet Package Decomposition and Mutual Information. *Proceedings - 2018 IEEE International Conference on Bioinformatics and Biomedicine, BIBM 2018*, 831–838. https://doi.org/10.1109/BIBM.2018.8621401

[6] Hilgen, G., Sorbaro, M., Pirmoradian, S., Muthmann, J. O., Kepiro, I. E., Ullo, S., Ramirez, C. J., Puente Encinas, A., Maccione, A., Berdondini, L., Murino, V., Sona, D., Cella Zanacchi, F., Sernagor, E., & Hennig, M. H. (2017). Unsupervised Spike Sorting for Large-Scale, High-Density Multielectrode Arrays. *Cell Reports*, *18*(10), 2521–2532. https://doi.org/10.1016/j.celrep.2017.02.038

[7] Kuang, Q., & Zhao, L. (2009). A practical GPU based kNN algorithm. *International Symposium on Computer Science and Computational Technology (ISCSCT)*, *7*(3), 151–155.

[8] Mokri, Y., Salazar, R. F., Goodell, B., Baker, J., Gray, C. M., & Yen, S. C. (2017). Sorting overlapping spike waveforms from electrode and tetrode recordings. *Frontiers in Neuroinformatics*, *11*(August), 1–15. Https://doi.org/10.3389/fninf.2017.00053

[9] Muhammad, I., & Yan, Z. (n.d.). *SUPERVISED MACHINE LEARNING APPROACHES: A SURVEY*. https://doi.org/10.21917/ijsc.2015.0133

[10] Nath, V., & Levinson, S. E. (2014). Machine learning. In *SpringerBriefs in Computer Science* (Issue 9783319056050). https://doi.org/10.1007/978-3-319-05606-7_6

[11] Noce, E., Ciancio, A. L., & Zollo, L. (2018). Spike detection: The first step towards an ENG-based neuroprosheses. *Journal of Neuroscience Methods*, *308*, 294–308. https://doi.org/10.1016/j.jneumeth.2018.07.008

[12] Noce, E., Ciancio, A. L., & Zollol, L. (2018). Accuracy Optimization of the Spike Sorting Algorithm for Classification of Neural Signals. *Proceedings of the IEEE RAS and EMBS International Conference on Biomedical Robotics and Biomechatronics*, *2018-Augus*, 131–135. https://doi.org/10.1109/BIOROB.2018.8487665

[13] Quian Quiroga, R., & Nadasdy, Z. (2004). *Communicated by Maneesh Sahani Unsupervised Spike Detection and Sorting with Wavelets and Superparamagnetic Clustering*. *1687*, 1661–1687.

[14] Rey, H. G., Pedreira, C., & Quian Quiroga, R. (2015). Past, present and future of spike sorting techniques. *Brain Research Bulletin*, *119*, 106–117. https://doi.org/10.1016/j.brainresbull.2015.04.007

[15] Sukiban, J., Voges, N., Dembek, T. A., Pauli, R., Visser-Vandewalle, V., Denker, M., Weber, I., Timmermann, L., & Grün, S. (2019). Evaluation of Spike Sorting Algorithms: Application to Human Subthalamic Nucleus Recordings and Simulations. *Neuroscience*, *414*, 168–185. https://doi.org/10.1016/j.neuroscience.2019.07.005

[16] Takekawa, T., Isomura, Y., & Fukai, T. (2012). Spike sorting of heterogeneous neuron types by multimodality-weighted PCA and explicit robust variational bayes. *Frontiers in Neuroinformatics*, *6*(MARCH), 1–13. https://doi.org/10.3389/fninf.2012.00005

[17] Williams, I., Luan, S., Jackson, A., & Constandinou, T. G. (2015). A Scalable 32 Channel Neural Recording and Real-Time FPGA Based Spike Sorting System. *IEEE Biomedical Circuits and Systems Conference: Engineering for Healthy Minds and Able Bodies, BioCAS 2015 - Proceedings*, *16*(8), 0–4. https://doi.org/10.1109/BioCAS.2015.7348330

[18] Wu, T., Zhao, W., Guo, H., Lim, H. H., & Yang, Z. (2017). A Streaming PCA VLSI Chip for Neural Data Compression. *IEEE Transactions on Biomedical Circuits and Systems*, *11*(6), 1290–1302. https://doi.org/10.1109/TBCAS.2017.2717281

[19] Yang, K., Wu, H., & Zeng, Y. (2017). A Simple Deep Learning Method for Neuronal Spike Sorting. *Journal of Physics: Conference Series*, *910*(1).

[20] Zamani, M., Sokolic, J., Jiang, D., Renna, F., Rodrigues, M., & Demosthenous, A. (2020). Accurate, Very Low Computational Complexity Spike Sorting Using Unsupervised Matched Subspace Learning. *IEEE Transactions on Biomedical Circuits and Systems*, *14*(2), 221–231. https://doi.org/10.1109/TBCAS.2020.2969910