Automated Spike sorting using Wavelet and DBSCAN

Mounika Gorantla
mgorantl@asu.edu
1208042195
Cidse-ASU

Vignesh Soundararajan
vsounda2@asu.edu
1207207293
Cidse-ASU

Shivanshu Shukla
sshukla5@asu.edu
1207888353
Cidse-ASU

Abstract: *Spike sorting algorithms uses the shapes of waveforms collected using electrodes to distinguish the activity of one neuron from another. The algorithm includes three stages spike detection, feature extraction and clustering. The main focus of this project is feature extraction and clustering as the provided data is already preprocessed. Numerous research papers have been published studying the above stages. Most of the currently available papers use either PCA or wavelet for feature extraction along with other statistical information like zero crossing. The clustering have been studies extensively and many people have formulated multiple algorithms. Ex. k-means, Super-paramagnetic Clustering (SPC), Gaussian Mixture Model (GMM), DBSCAN, OPTICS, Hierarchical Clustering, etc. After doing the survey and experimentation, we found the combination of wavelet and DBSCAN gave a better performance and automated clustering than the other available method. With the above approach, we were able to achieve more than 95% accuracy for all the four test data set (Sampledata_test(1-4).mat).*

Keywords: Artificial Neural Computation, Spike sorting clustering, PCA, Wavelet transform, k-mean, DBSCAN.

1. Introduction

This Project's aim is to read through some of spike sorting algorithm and decide a viable approach for attempting to sort the dataset provided to us using unsupervised learning methods to develop nearly automatic algorithm for sorting Neural spikes.

From [1] we understood basic steps we need to sort spike by processing raw data generated from EEG. The entire process consists of 4 steps:

1) Filtering
2) Spike detection
3) Feature extraction
4) Clustering

Filtering: The data that we collect from EEG is not clean spikes data, there is always some noise. The noise can be due to nearby neurons or environment or sometimes due to displacement of electrode. To make sure that we don't pick up noise as a spike we need to first get read of this noise. Filtering can be done by non-causal band pass filter generally the band used for spike filtering is between 300 to 3000Hz. This is an important step as having a good spike with least amount of noise possible will help is reducing misclassification.

Spike Detection: In This step we basically separate each spike from one another. Raw data can have overlapping spikes and we need to separate those spike to have a better classification. Generally data is collected over multiple channels and we used the data from there channels to separate out the spikes. The most basic techniques used for separating out spikes (assuming that the data used doesn't have overlapping spikes) is by thresholding. We have implemented a simple thresholding techniques as a part of this project.

Feature Extraction: In this step we extract features from individual spikes and use those features for classification. Generally Feature extraction and clustering makeup a spike sorting Algorithm. And detection in itself is a challenging problem, so for our Final project it's logical to only concentrate on just sorting rather than trying to get the entire process right.

Clustering: This is where the spikes are classified. We can use different techniques for clustering like K-means, Bayesian Classification, super-paramagnetic clustering, DBSCAN etc.

The classification efficiency is dependent on entire process but we will assume that raw data was converted into spikes in the best possible way so that we can extract features and use them for clustering.

There are different techniques that are used for feature extraction of which we have analysed Principal Component Analysis (PCA), wavelet transform, and template matching. For clustering we have analyzed K-means, DBSCAN.

We have implemented a technique which uses PCA for extracting features and k-means for clustering. This technique is not very effective but is still workable. Then we performed derivative on the given data and used PCA and Wavelet transform for feature extraction and clustered using DBSCAN. The noise is clustered using template matching, knn and correlation.

2. Algorithms Used

We have not done any preprocessing before extracting the features in the algorithm we have implemented but because the results for sample dataset_1 and sampledata_2 seem separable we did not perform any preprocessing.

The techniques we have looked into are:

A. PCA+k-means

This techniques is a very basic techniques where we get Principal components as shown in [2]. As most of the information lies in first 3 components we will used only first 3 components and will lose very little in after running the Algorithm we will then plot the scores. When we looked at the score graph we could clearly see that there are 3 clusters, so to label those clusters we apply k-means clustering and see how well are the points clustered. When we look on the graph in figure1 above we can see that there are effectively 3 clusters. To counter we also tried other techniques as discussed in B, C and D.

B. KPCA+kmeans

This techniques is an extension to fairly common and basic feature extraction technique called PCA. This is kernel PCA [3] where we basically use a kernel method to plot the data given to us in a high dimension space and then try to use the separation to cluster. We used Gaussian, and polynomial of 2 to create the kernel for projecting our data to higher dimensional space. We were able to project the data onto high dimensional space. We notice that after projecting the dataset to higher dimension we did not see 3 clusters but we saw 4 clusters when we plotted those different cluster we were able to separate out the outlying noisy neural components into cluster number 4.

C. Wavelet Transform + K-means

The wavelet transform (WT) is a time-frequency representation of the signal that has two main advantages over conventional methods: it provides an optimal resolution in both the time and the frequency domains, and it eliminates the requirement of signal stationarity. It is defined as the convolution between the signal x(t) and the wavelet functions Ψ(a,b) (t),

$$W_\Psi X(a,b) = \langle x(t) | \Psi_{a,b}(t) \rangle, \qquad 1$$

Where Ψ(a,b) (t) are dilated (contracted), and shifted versions of a unique wavelet function Ψ(t),

$$\Psi_{a,b}(t) = |a|^{\frac{-1}{2}} \Psi\left(\frac{t-b}{a}\right) \qquad 2$$

where a and b are the scale and translation parameters, respectively. Equation 1 can be inverted, thus providing the reconstruction of x(t). [4]

The WT maps the signal that is represented by one independent variable t onto a function of two independent variables a, b. This procedure is redundant and inefficient for algorithmic implementations; therefore, the WT is usually defined at discrete scales a and discrete times b by choosing the set of parameters $\{a_j = 2^{-j}; b_{j,k} = 2^{-j}k\}$, with integers j and k. Contracted versions of the wavelet function match the high-frequency components, while dilated versions match the low-frequency components. Then, by correlating the original signal with wavelet functions of different sizes, we can obtain details of the signal at several scales. These correlations with the different wavelet functions can be arranged in a hierarchical scheme called multiresolution decomposition (Mallat, 1989). The multiresolution decomposition algorithm separates the signal into details at different scales and a coarser representation of the signal named "approximation" (for details, see Mallat, 1989; Chui, 1992; Samar, Swartz, & Raghveer, 1995; Quian Quiroga, Sakowicz, Basar, & Sch ¨urmann, 2001; Quian Quiroga & Garcia, 2003). [4]

In this project we used a selection of wavelet coefficients chosen with a Kolmogorov Smirnov test of Normality. Given a data set x, the test compares the cumulative distribution function of the data (F(x)) with that of a Gaussian distribution with the same mean and variance (G(x)). Deviation from normality is then quantified by

$$\max(|F(x) - G(x)|)$$

In our implementation, the first 10 coefficients with the largest deviation from normality were used. The selected set of wavelet coefficients provides a compressed representation of the spike features that serves as the input to the clustering algorithm.

D. Derivative+Wavelet+DBSCAN

This is the final approach we followed. We performed derivative on the given spikes, extracted the features using wavelet and clustered using DBSCAN.

Derivative: The recorded waveforms from neurons with similar ion channel populations may have localized variations (high frequency) making it difficult to examine in time domain. A filter is needed to boost these high frequency spike features for clustering these similar spikes. Derivative operation amplifies these localized variations making it easier to cluster these spikes. Figure 1 shows two spikes from different cluster. Analyzing these spikes in time domain is difficult as they both look similar visually. After doing derivative operation, the wavelet coefficients of these spikes showed 30% increase in the euclidean distance between each other.

Wavelet: Use wavelet transform to calculate the coefficients for each spike. Select the wavelet coefficients using Kolmogorov Smirnov test for Normality. Our implementation uses the first 10 coefficients with the largest deviation from normality.

Wavelet coefficients provides a compressed representation of the spike features that serves as the input to the clustering algorithm.

DBSCAN: It is a density based clustering algorithm. A cluster satisfies two points.

- All points within the cluster are mutually density-connected.
- If a point is density-reachable from any of the cluster, it is a part of the cluster as well

The border points which do not follow above condition are treated as noise by DBSCAN are sorted separately using knn and correlation methods.

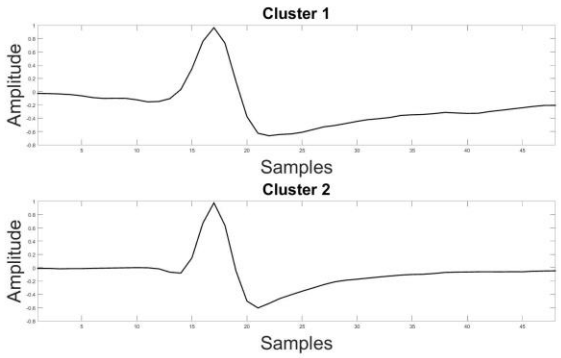

Figure 1: similar looking spikes from different clusters

DBSCAN needs 2 parameters to cluster the dataset they are epsilon- the maximum distance of neighborhood points. Minpoint i.e the number of points needed by a point in its epsilon neighborhood to get clustered as one cluster. These inputs can be considered user intervention. To make the algorithm automated we need to get a good estimate of both epsilon and minpoints. There are certain techniques proposed like k-neighbor graph, elbow method and more. We are using mean of the distances at minpoints neighborhood. Using mean distance of k-th neighboring points we get a rough estimate of epsilon and using that epsilon and minpoints we cluster using DBSCAN. We have used DBSCAN implementation of [6].

Knn: Noisy spike is assigned to the nearest cluster.

Correlation: Found a spike that shows high correlation with other spikes in the cluster. Compare the correlation coefficient of each noisy spike with the above spike.

Among the four approaches we found that wavelet of derivative clustered using DBSCAN is more efficient. The drawback with kmeans is that it is a centroid based algorithm. Some of the clusters might have the centroid outside the cluster. For the given dataset in our project, sample dataset 1 has centroid inside the cluster this doesn't have any problem so it works good with kmeans but for sample dataset 3 the centroid lies outside the cluster, as shown in figure 2 which needed to be resolved using a different approach. These type of clusters are classified using DBSCAN which gives good results as density will differentiate the cluster able and spaces between the clusters.

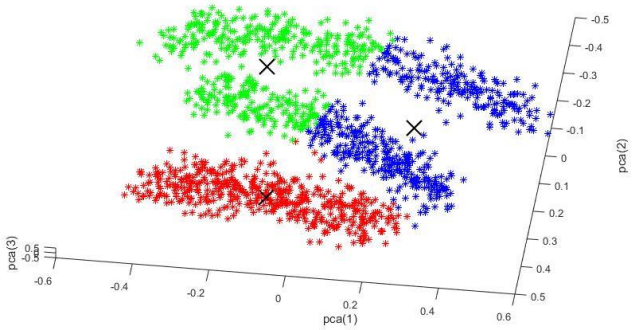

Figure 2: failure of k-means for feature extracted using derivative+PCA.

3. Results

Sample data set 1

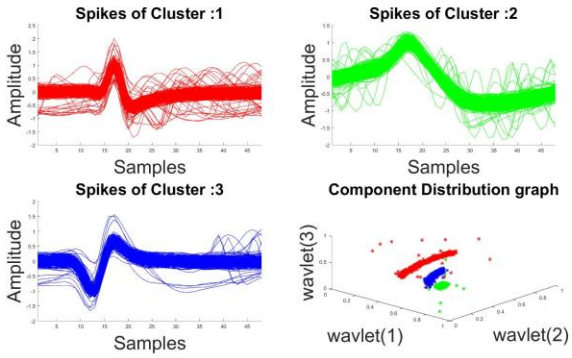

Figure 3: sample dataset distribution, and sorted spikes for dataset 1

Sample data set 2

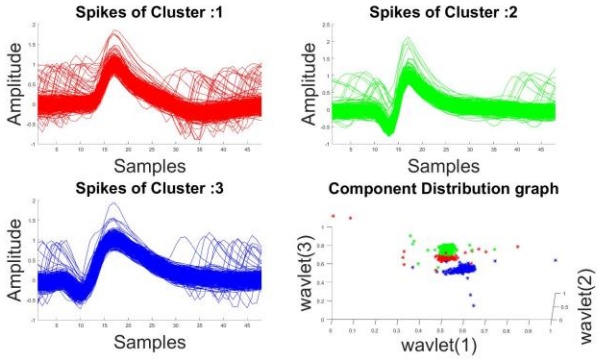

Figure 4: sample dataset distribution, and sorted spikes for dataset 2

Sample data set 3

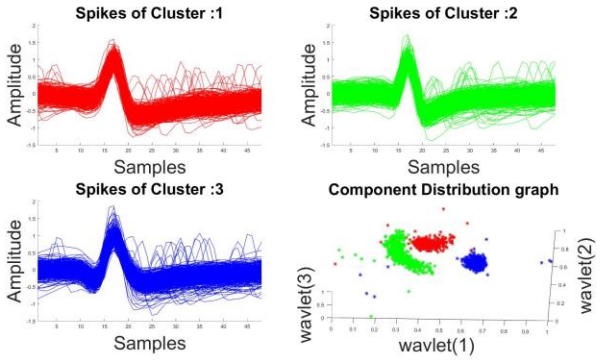

Figure 5: sample dataset distribution, and sorted spikes for dataset 3

Sample data set 4

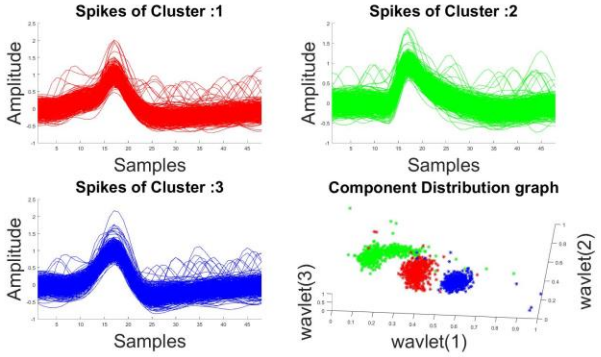

Figure 6: sample dataset distribution, and sorted spikes for dataset 4

Accuracy matrix:

Dataset name	Accuracy
Sampledataset_1	95.93%
Sampledataset_2	94.4%
Sampledataset_3	94.53%
Sampledataset_4	95.13%
Sampledataset_test_1	96.5%
Sampledataset_test_2	96.7%
Sampledataset_test_3	96.8%
Sampledataset_test_4	95.2%

Entire algorithm is automated and will cluster all the data set. Code explanation will be given in a readme file. Total execution time for one dataset of 1500 samples averages around 4-5 secs, which included extracting features, clustering, plotting graphs, and writing the data to an excel file.

Final accuracy as shown in Table 1 looks good for all the datasets. While testing we tried manually adding more clusters to see if it clusters all the clusters properly and we were able to see more clusters.

4. Conclusion

We have tried different approaches for sorting the given spike their advantages and drawbacks. The final approach of performing derivative on spikes, extracting features using wavelet and clustering using DBSCAN has given the best results. Using derivative and wavelet seems to work very well for spikes and once we have good feature extractor we can thing of working with a good clustering algorithm. For the kind of data distribution that we have we found that density based clustering works very well and it can be used if we have clusters with similar densities. We have not tried with variable density clusters which can be handled by different algorithms like one mentioned in [5].

5. Future work

We have considered the situation where the density of all the clusters is assumed almost similar but that not very accurate in the dataset that we were given the density of different clusters is different and we may need to take into consideration to get a very good estimate for epsilon for DBSCAN. There can be significant improvements made in the direction of clustering multi-density clustering.

6. Acknowledgement

We are very thankful to Professor Jennie Si who helped and guided us throughout the project and provided us the opportunity to work with real-life datasets to get a good feel of clustering. This project has helped in many ways. By working on this project we were able to expand our knowledge of clustering and model a system for a clustering application.

7. References

[1] A Tutorial on Principal Component Analysis

[2] *Zhang, T.; Ramakrishnan, R.; Livny, M. (1996). "BIRCH: an efficient data clustering method for very large databases". Proceedings of the 1996 ACM SIGMOD international conference on Management of data - SIGMOD '96. pp. 103—114*

[3] Kernel principal component analysis
B Schölkopf, A Smola, KR Müller - Artificial Neural Networks—ICANN'97, 1997 - Springer

[4] Quiroga, R. Quian, Zoltan Nadasdy, and Yoram Ben-Shaul. "Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering."Neural computation 16.8 (2004): 1661-168

[5] Brecheisen S, Kriegel H-P, Pfeifle M. Multi-step densitybased clustering. *Knowledge and Information Systems* 2006; **9**: 284–308

[6] Thanh N. Tran*, Klaudia Drab, Michal Daszykowski, "Revised DBSCAN algorithm to cluster data with dense adjacent clusters", Chemometrics and Intelligent Laboratory Syste