Clustering Spikes

Salar Noori[1]

1. Computer Engineering Department, Sharif University of Technology, Tehran, Iran

Abstract:

Spike sorting is a pivotal task in neuroscience, essential for decoding the activity of neural networks from extracellular recordings. This study comprehensively evaluates six clustering methods—KMeans, Self-Organizing Map (SOM), Hierarchical Divisive Clustering, CMeans, DBSCAN, and Distribution-Based Clustering—on two distinct datasets: Zebra Finch song recordings and pre-recorded spikes. The preprocessing of the raw signal involves high-pass and low-pass filtering, followed by peak detection. The performance of each clustering method is assessed using Silhouette Score, Calinski-Harabasz Index, and Davies-Bouldin Index.

The findings reveal significant variations in clustering performance across different methods and datasets. For the Zebra Finch dataset, DBSCAN demonstrates the highest Silhouette Score, indicating well-separated clusters. Conversely, for the spike dataset, SOM and CMeans exhibit superior clustering quality. These results underscore the importance of selecting suitable clustering techniques tailored to the characteristics of specific datasets. This work provides valuable insights for neuroscience researchers in optimizing spike sorting processes.

Keywords: Spike Sorting, Clustering, KMeans, Self-Organizing Map, Hierarchical Clustering, CMeans, DBSCAN, Distribution-Based Clustering, Zebra Finch, Neural Recordings.

I. Introduction:

Spike sorting is a critical step in neural data analysis, involving the identification and classification of action potentials (spikes) from raw extracellular recordings. This process is essential for understanding the communication between neurons and the functional organization of neural circuits. Accurate spike sorting enables researchers to decode neural activity patterns and infer the underlying neural computations. Traditionally, manual spike sorting has been used, but this method is time-consuming and subjective. Hence, the development of automated clustering techniques has become increasingly important.

The current study explores six clustering methods to determine their effectiveness in spike sorting. The methods investigated include KMeans, Self-Organizing Map (SOM), Hierarchical Divisive Clustering, CMeans, DBSCAN, and Distribution-Based Clustering. These methods were chosen due to their diverse approaches to clustering, ranging from partition-based and hierarchical techniques to density-based and fuzzy clustering. This diversity ensures a comprehensive evaluation of their performance on neural data.

Two datasets are utilized for this analysis: the Zebra Finch song dataset and a pre-recorded spike dataset. The Zebra Finch dataset is particularly interesting because the bird's song patterns can be correlated with neural activity, offering a rich source of data for spike sorting. The second dataset, consisting of pre-recorded spikes, provides a controlled environment to test the robustness of the clustering methods. Preprocessing steps, including high-pass and low-pass filtering, are applied to the raw signals to enhance the quality of the spike detection.

Peak detection is a crucial preprocessing step in spike sorting. For the Zebra Finch dataset, peaks are identified using a threshold based on the standard deviation of the filtered signal. This approach ensures that only significant spikes are considered, reducing the noise in the data. The detected peaks are then used to extract spike waveforms, which serve as the input for clustering. This process results in a high-dimensional dataset that needs to be effectively reduced for clustering analysis.

Principal Component Analysis (PCA) is employed to reduce the dimensionality of the spike waveforms, facilitating visualization and clustering. PCA transforms the high-dimensional data into a lower-dimensional space while preserving the variance in the data. This step is crucial for enhancing the performance of clustering algorithms, as it simplifies the data structure and highlights the most informative features.

The performance of each clustering method is evaluated using three metrics: Silhouette Score, Calinski-Harabasz Index, and Davies-Bouldin Index. These metrics provide a comprehensive assessment of clustering quality, considering factors such as cluster cohesion, separation, and overall cluster structure. By comparing these metrics across different clustering methods and datasets, we can determine the most effective techniques for spike sorting.

In summary, this study aims to provide a detailed comparison of six clustering methods for spike sorting, offering valuable insights into their strengths and weaknesses. By evaluating these methods on two diverse datasets, we aim to guide researchers in selecting the most appropriate clustering techniques for their specific needs. The findings highlight the importance of tailored approaches in neural data analysis and contribute to the advancement of automated spike sorting methodologies.

II. Datasets and Preprocessing:

1. Zebra Finch Dataset:

The Zebra Finch dataset (zebra_finch_data.mat) comprises recordings of Zebra Finch song, providing a rich source of neural activity data. This dataset includes three key components: event markers (event) corresponding to specific song syllables, the sampling frequency of the recordings (fs), and the raw neural signal data (rawSig).

Preprocessing:

Preprocessing the raw signal is a crucial step to enhance the quality of spike detection. The preprocessing pipeline involves two main filtering steps:

1. **High-Pass Filtering**: A Butterworth high-pass filter is applied to remove low-frequency noise and baseline drifts. The cutoff frequency is set at 250 Hz, and a filter order of 5 is chosen for an optimal balance between attenuation and computational efficiency.
2. **Low-Pass Filtering**: Following high-pass filtering, a Butterworth low-pass filter is applied to remove high-frequency noise. The cutoff frequency is set at 3000 Hz with a filter order of 5.

These filtering steps produce a cleaner signal by retaining the frequency components that are most relevant for spike detection. The processed signal is then visually inspected to ensure the efficacy of the filtering.

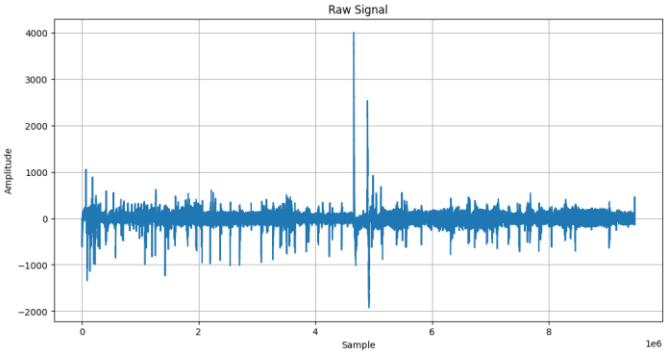

Figure 1 Raw Signal of Zebra Finch data

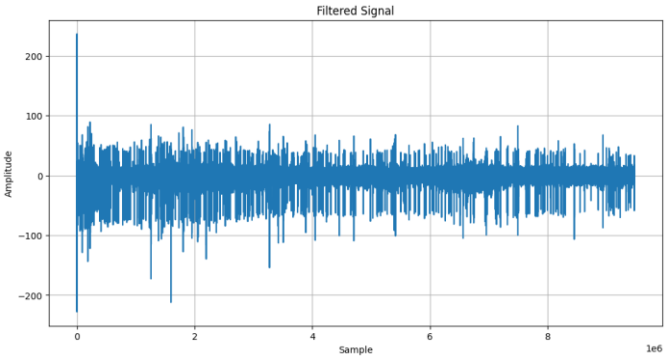

Figure 2 Filtered Raw Signal of Zebra Finch data

Peak Detection:

After filtering, peaks corresponding to neural spikes are detected. The peak detection process involves setting a minimum peak height, determined by three times the standard deviation of the filtered signal, and a minimum peak distance to avoid detecting the same spike multiple times. This approach ensures that only significant spikes are considered, reducing noise in the data. In the Zebra Finch dataset, 4891 peaks were identified. For each peak, a 60-sample waveform centered around the peak is extracted. This results in a matrix of shape (4891, 60), where each row represents the waveform of a detected spike.

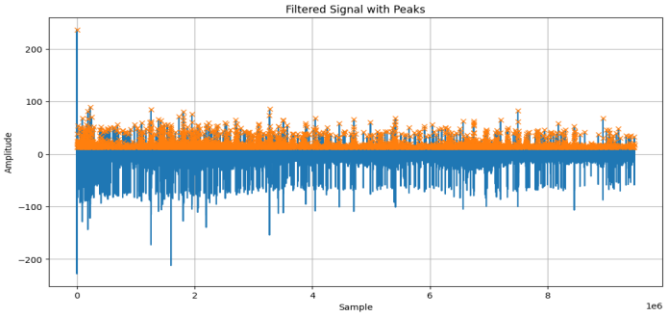

Figure 3 Mapped Peaks on Filtered Signal of Zebra Finch data

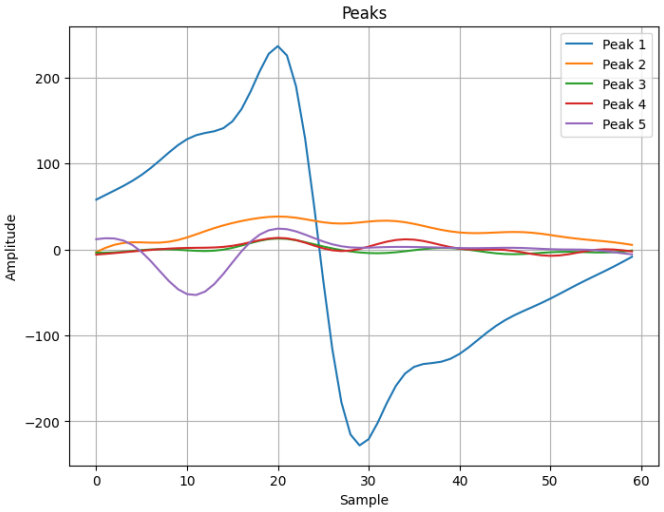

Figure 4 Some Found Peaks Samples

2. Spike Dataset:

The second dataset (spikes.mat) consists of pre-recorded spikes, providing a controlled environment for testing clustering algorithms. This dataset contains 460 spikes, each represented by a 60-sample waveform. As the spikes are already isolated and stored in a structured format, no additional preprocessing is required.

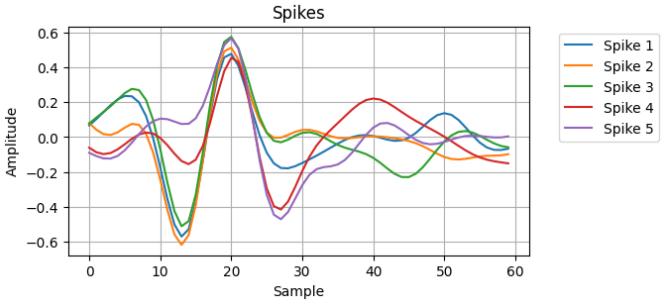

Figure 5 Some Spike Samples

Dimensionality Reduction:

To facilitate clustering and visualization, Principal Component Analysis (PCA) is applied to both datasets. PCA reduces the high-dimensional spike waveforms to a lower-dimensional space, preserving the most significant variance in the data. This transformation helps in visualizing the data structure and enhances the performance of clustering algorithms by simplifying the data.

For the Zebra Finch dataset, PCA transforms the (4891, 60) matrix into a (4891, 2) matrix. Similarly, for the spike dataset, PCA transforms the (460, 60) matrix into a (460, 2) matrix.

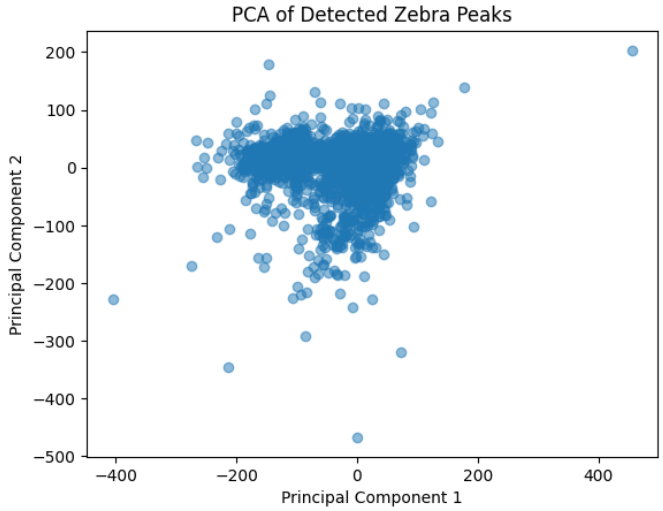

Figure 6 PCA Diagram of Zebra Finch data

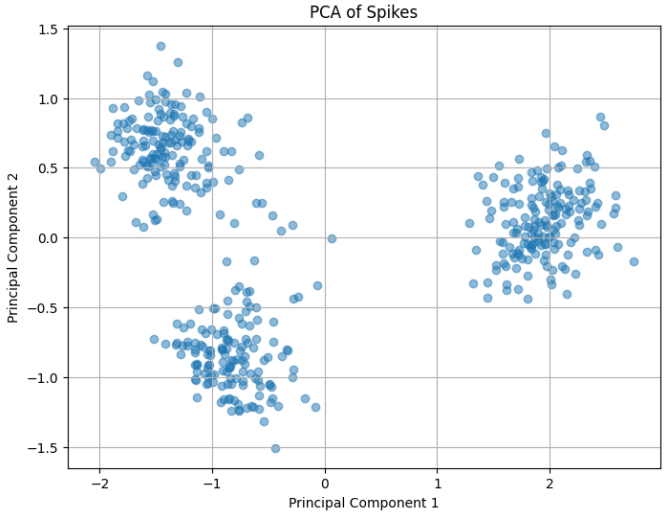

Figure 7 PCA of Spikes data

This preprocessing pipeline ensures that the data is clean and ready for clustering, with noise and artifacts minimized, and the essential features of the spike waveforms preserved. By applying these preprocessing steps, we enhance the accuracy and reliability of the subsequent clustering analysis.

III. Methods

This section details the methods used for clustering the spikes in the Zebra Finch and Spike datasets. We employed six clustering techniques:

KMeans, Self-Organizing Map (SOM), Hierarchical Divisive Clustering, CMeans, DBSCAN, and Distribution-Based Clustering. For each method, we provide a mathematical description, the algorithm's implementation, and the rationale behind parameter selection.

1. KMeans Clustering

KMeans clustering aims to partition the dataset into kkk clusters by minimizing the variance within each cluster. The objective function is:

$$J = \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|^2$$

where C_i is the i-th cluster, μ_i is the centroid of cluster C_i, and $\|x - \mu_i\|$ is the Euclidean distance between data point x and centroid μ_i.

For the Zebra Finch dataset, we selected k=2 clusters, and for the Spike dataset, k=3 clusters. The KMeans algorithm was initialized using the k-means++ method to ensure better initial centroids. The algorithm iterates until the centroids stabilize, i.e., the change in cluster centroids is below a predefined threshold.

2. Self-Organizing Map (SOM)

A SOM is a type of artificial neural network used to produce a low-dimensional representation of the input space. The SOM consists of nodes arranged in a grid, with each node having a weight vector of the same dimension as the input data. The objective is to map the input data into this grid such that similar data points are mapped to nearby nodes.

The training involves:

- Selecting a data point x.
- Finding the Best Matching Unit (BMU) w by minimizing:

$$\|x - w\|$$

- Updating the weight vector of the BMU and its neighbors:

$$w(t+1) = w(t) + \alpha(t) \cdot h(w, wBMU) \cdot (x - w(t))$$

where $\alpha(t)$ is the learning rate and $h(w, wBMU)$ is the neighborhood function.

For the Zebra Finch dataset, a SOM with a 2x1 grid was used, and for the Spike dataset, a 3x1 grid was chosen. The learning rate and neighborhood function were adjusted dynamically over 10 iterations to ensure convergence.

3. Hierarchical Divisive Clustering

Hierarchical divisive clustering is a top-down approach that starts with all data points in a single cluster and recursively splits clusters until each data point is in its own cluster or a stopping criterion is met. The splitting process typically uses variance minimization or another criterion to determine the optimal division.

For the Zebra Finch dataset, the data was split into 2 clusters, and for the Spike dataset, into 3 clusters. The splitting criterion was based on minimizing the within-cluster sum of squares, similar to the KMeans objective.

4. CMeans Clustering (Fuzzy CMeans)

CMeans clustering generalizes KMeans by allowing data points to belong to multiple clusters with varying degrees of membership. The objective function is:

$$J_m = \sum_{i=1}^{k} \sum_{j=1}^{n} u_{ij}^m \|x_j - \mu_i\|^2$$

where u_{ij} is the membership degree of data point x_j in cluster i, and mmm is a fuzziness parameter.

The update rules are:

$$\mu_i = \frac{\sum_{j=1}^{n} u_{ij}^m x_j}{\sum_{j=1}^{n} u_{ij}^m}$$

$$u_{ij} = \frac{1}{\sum_{c=1}^{k} \left(\frac{\|x_j - \mu_i\|}{\|x_j - \mu_c\|} \right)^{\frac{2}{m-1}}}$$

CMeans clustering was applied with 2 clusters for the Zebra Finch dataset and 3 clusters for the Spike dataset. The fuzziness parameter mmm was set to 2, and the

algorithm iterated until the change in membership degrees was below 0.005.

5. DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

DBSCAN clusters data points based on density. It defines clusters as areas of high density separated by areas of low density. The key parameters are ϵ\epsilonϵ, the maximum distance between two points to be considered neighbors, and MinPtsMinPtsMinPts, the minimum number of points required to form a dense region.

The algorithm classifies points into core points, border points, and noise:

- A point is a core point if it has at least MinPtsMinPtsMinPts points within ϵ.
- A border point is reachable from a core point but has fewer than $MinPts$ neighbors.
- Noise points are neither core nor border points.

DBSCAN was applied with $\epsilon = 1$ and $MinPts = 6$ for the Zebra Finch dataset and $\epsilon = 0.2$ and $MinPts = 6$ for the Spike dataset. Prior to clustering, the data was normalized using a standard scaler to ensure uniformity.

6. Distribution-Based Clustering

Distribution-based clustering models each cluster by a statistical distribution. A common approach is to use Gaussian Mixture Models (GMM), where each cluster is represented by a Gaussian distribution with parameters estimated using the Expectation-Maximization (EM) algorithm.

The likelihood of the data given the model is:

$$P(X \mid \theta) = \sum_{i=1}^{k} \pi_i \mathcal{N}(x|\mu_i, \Sigma_i)$$

where π_i is the mixing coefficient, μ_i is the mean, and Σ_i is the covariance matrix of the i-th Gaussian component.

Distribution-based clustering was performed with 2 clusters and a bandwidth of 0.3 for the Zebra Finch dataset and 3 clusters with a bandwidth of 0.05 for the Spike dataset. The EM algorithm was used to estimate the parameters, and the number of components was selected based on the Bayesian Information Criterion (BIC).

By employing these diverse clustering techniques, we can compare the effectiveness of each method in identifying meaningful patterns in spike train data. The following section presents the results obtained from applying these methods to the Zebra Finch and Spike datasets.

IV. Results:

This section presents the results of applying six different clustering algorithms to the two datasets: the Zebra Finch dataset and the Spike dataset. The effectiveness of each algorithm is evaluated using three metrics: Silhouette Score, Calinski-Harabasz Index, and Davies-Bouldin Index. For visual clarity, results are accompanied by plots showing the clustering outcomes and examples of clustered spikes.

Zebra Finch Dataset

KMeans Clustering

KMeans was applied with k=2 clusters. The clustering results indicate a clear separation of spikes into two distinct clusters.

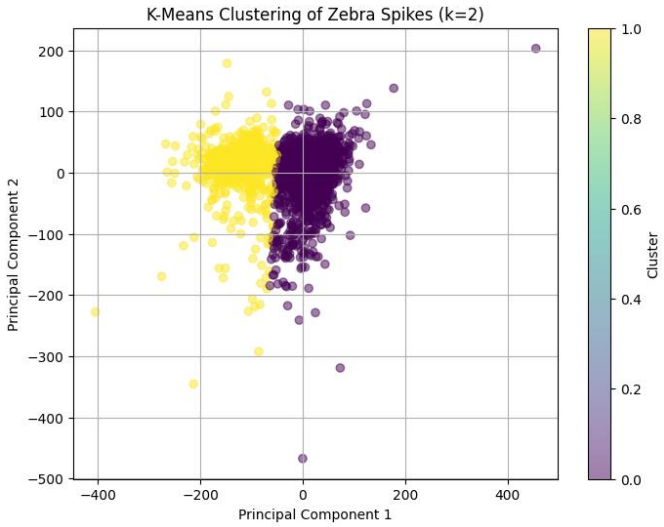

Figure 8 KMeans clustering results plot for Zebra Finch dataset

- **Silhouette Score**: 0.6946
- **Calinski-Harabasz Index**: 6009.96
- **Davies-Bouldin Index**: 0.5460

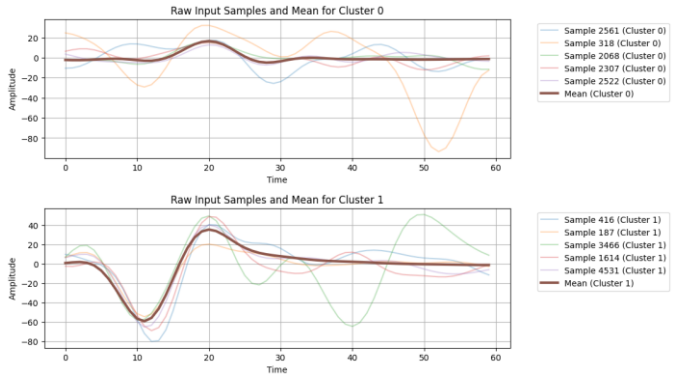

Figure 9 some sample spikes for each KMeans cluster in Zebra Finch dataset vs Mean of Cluster Signals

Self-Organizing Map (SOM)

The SOM was implemented with a 2x1 grid. The clusters formed show a strong separation similar to the KMeans results.

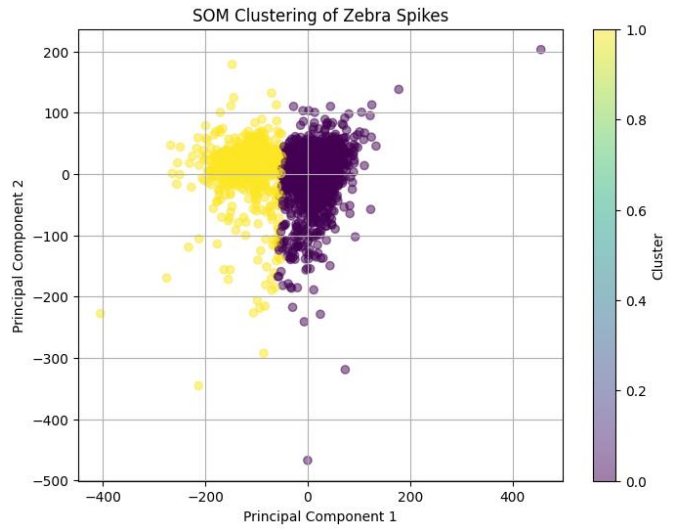

Figure 10 SOM clustering results plot for Zebra Finch dataset

- **Silhouette Score**: 0.6934
- **Calinski-Harabasz Index**: 6002.95
- **Davies-Bouldin Index**: 0.5611

Hierarchical Divisive Clustering

Hierarchical Divisive Clustering with 2 clusters reveals a slightly less distinct separation compared to KMeans and SOM.

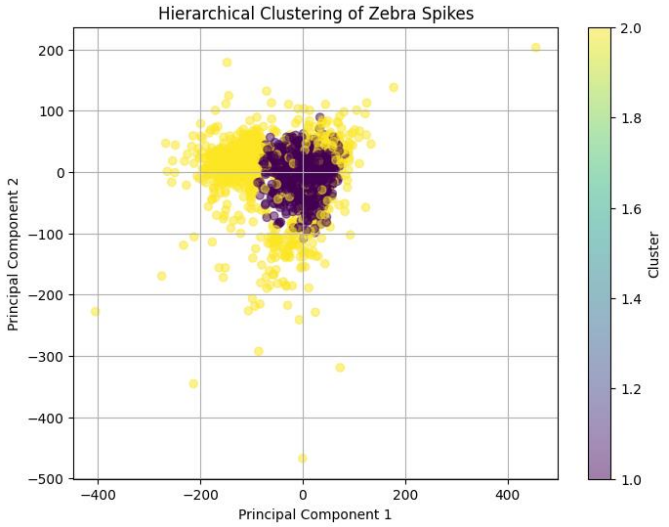

Figure 11 Hierarchical Divisive Clustering results plot for Zebra Finch dataset

- **Silhouette Score**: 0.5626
- **Calinski-Harabasz Index**: 1810.80
- **Davies-Bouldin Index**: 1.4346

CMeans Clustering

CMeans clustering was performed with 2 clusters. The fuzzy nature of the algorithm allows some overlap between clusters, reflected in slightly lower performance metrics.

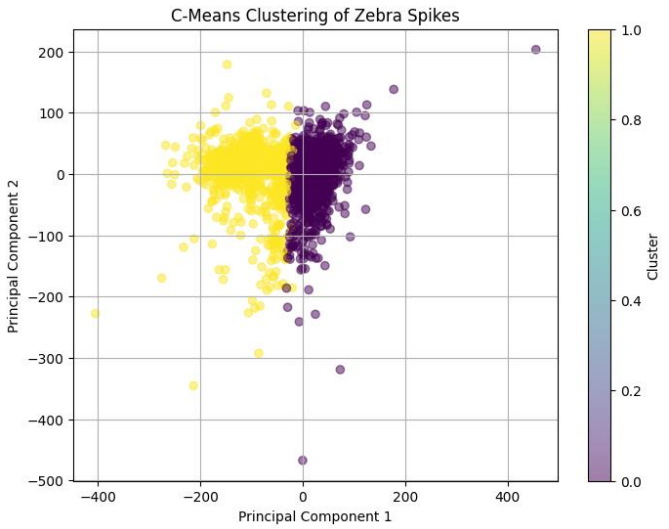

Figure 12 CMeans clustering results plot for Zebra Finch dataset

- **Silhouette Score**: 0.6520
- **Calinski-Harabasz Index**: 5417.30
- **Davies-Bouldin Index**: 0.7093

DBSCAN

DBSCAN identified clusters based on density, resulting in high Silhouette scores but lower Calinski-Harabasz indices due to the nature of the algorithm.

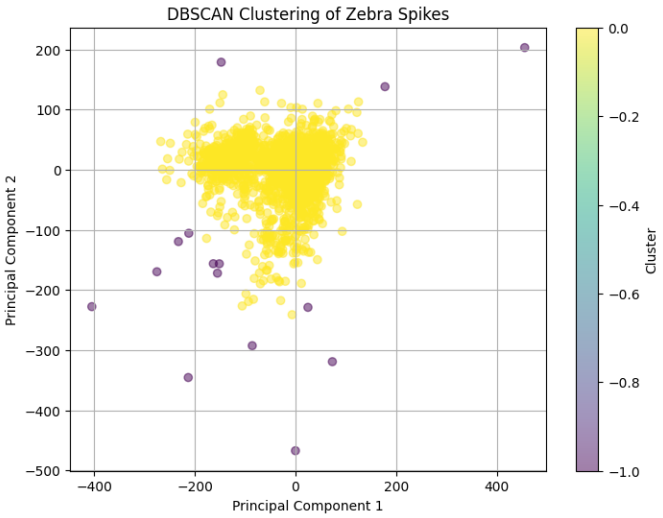

Figure 13 DBSCAN clustering results plot for Zebra Finch dataset

Estimated number of clusters: 1
Estimated number of noise points: 15

- **Silhouette Score**: 0.8070
- **Calinski-Harabasz Index**: 172.36
- **Davies-Bouldin Index**: 1.3761

Distribution-Based Clustering

This method was used with 2 clusters and a bandwidth of 0.3. The clusters were less distinct, as indicated by the lower Silhouette and Calinski-Harabasz scores.

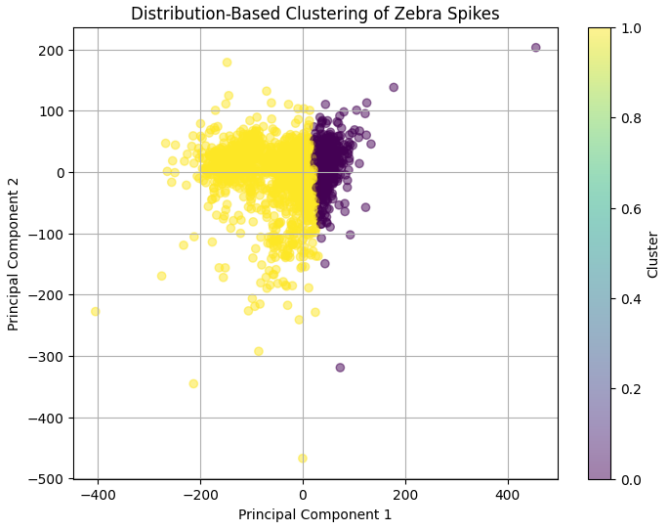

Figure 14 Distribution-Based Clustering results plot for Zebra Finch dataset

- **Silhouette Score**: 0.1415
- **Calinski-Harabasz Index**: 1269.96
- **Davies-Bouldin Index**: 1.2446

Spike Dataset

KMeans Clustering

KMeans was applied with k=3 clusters. The clusters were well-defined, though with slightly lower Silhouette scores compared to the Zebra Finch dataset.

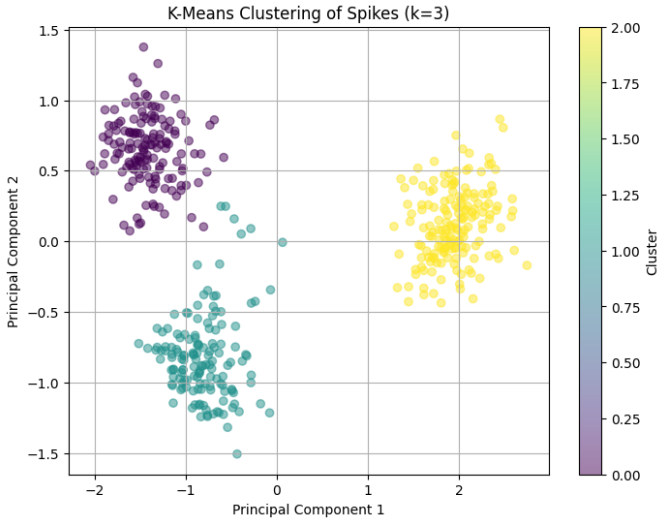

Figure 15 KMeans clustering results plot for Spike dataset

- **Silhouette Score**: 0.5319
- **Calinski-Harabasz Index**: 942.91
- **Davies-Bouldin Index**: 0.8339

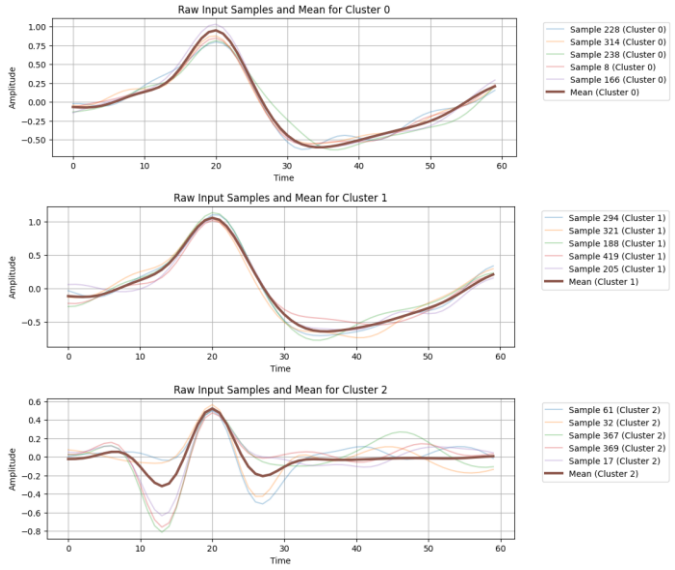

Figure 16 some sample spikes for each KMeans cluster in Spike dataset vs Mean of Cluster Signals

Self-Organizing Map (SOM)

The SOM was implemented with a 3x1 grid, resulting in highly distinct clusters with excellent separation.

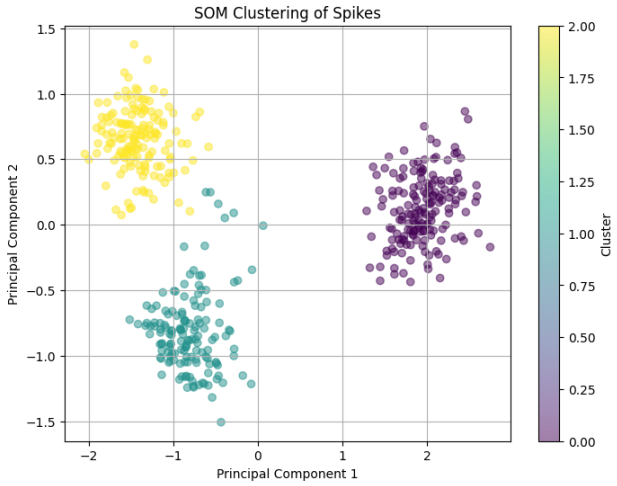

Figure 17 SOM clustering results plot for Spike dataset

- **Silhouette Score**: 0.7428
- **Calinski-Harabasz Index**: 3818.60
- **Davies-Bouldin Index**: 0.3670

Hierarchical Divisive Clustering

Hierarchical Divisive Clustering with 3 clusters also showed strong cluster separation, though slightly lower than SOM.

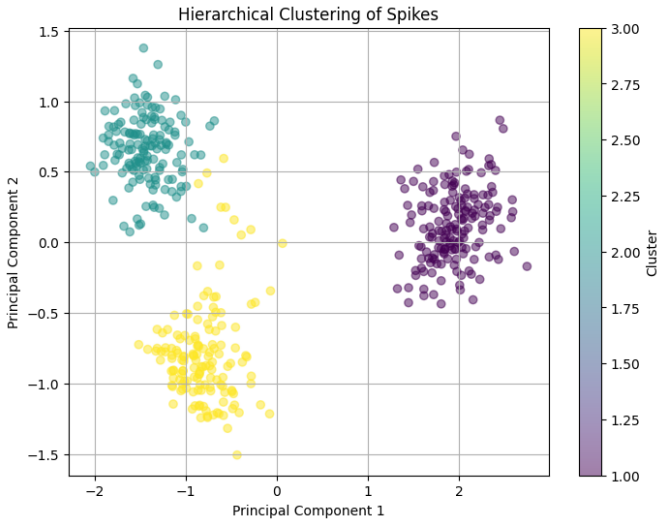

Figure 18 Hierarchical Divisive Clustering results plot for Spike dataset

- **Silhouette Score**: 0.7344
- **Calinski-Harabasz Index**: 3621.75
- **Davies-Bouldin Index**: 0.3786

CMeans Clustering

CMeans clustering was performed with 3 clusters. The results were similar to the SOM, demonstrating strong cluster separation.

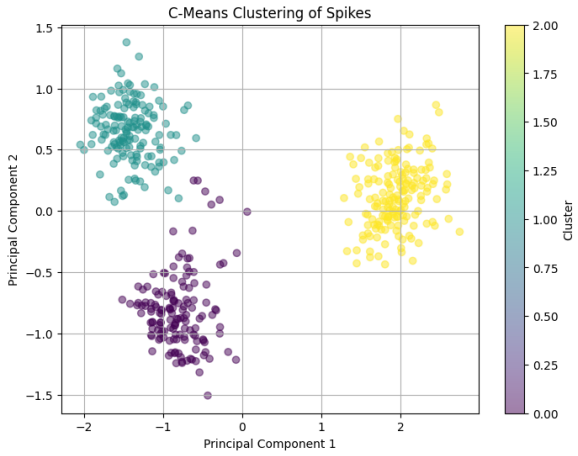

Figure 19 CMeans clustering results plot for Spike dataset

- **Silhouette Score**: 0.7428
- **Calinski-Harabasz Index**: 3818.60
- **Davies-Bouldin Index**: 0.3670

DBSCAN

DBSCAN was used with $\epsilon=0.2\epsilon = 0.2\epsilon=0.2$ and a minimum of 6 samples per cluster. The clusters were well-defined, but the evaluation scores were slightly lower.

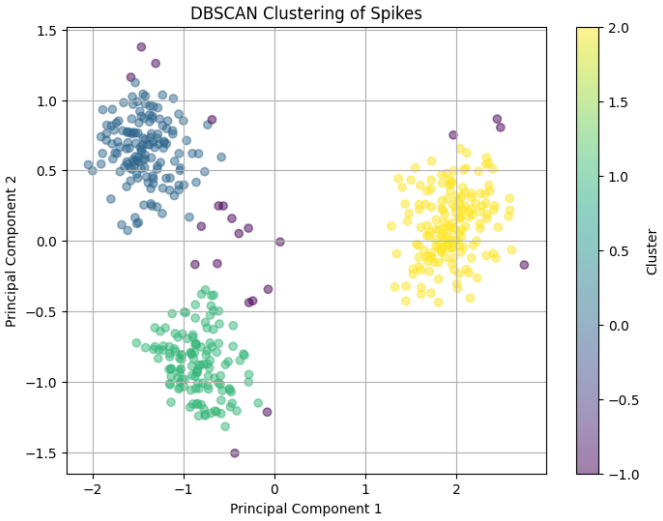

Figure 20 DBSCAN clustering results plot for Spike dataset

Estimated number of clusters: 3
Estimated number of noise points: 22

- **Silhouette Score**: 0.6732
- **Calinski-Harabasz Index**: 1123.91
- **Davies-Bouldin Index**: 1.1497

Distribution-Based Clustering

This method was used with 3 clusters and a bandwidth of 0.05. The clusters were less distinct, with lower evaluation scores compared to other methods.

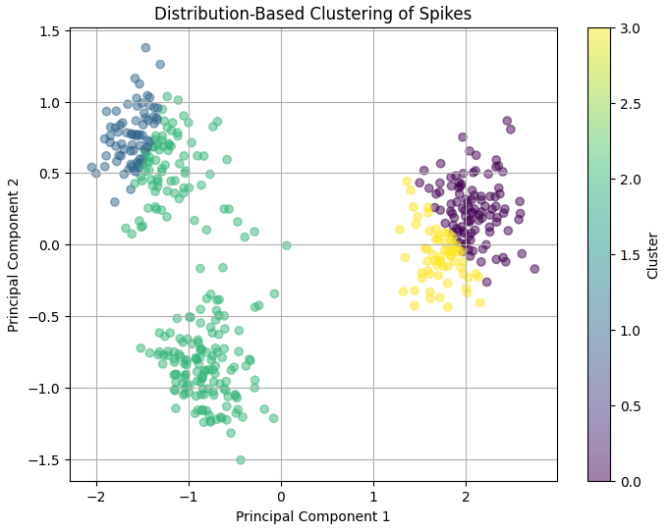

Figure 21 Distribution-Based Clustering results plot for Spike dataset

- **Silhouette Score**: 0.3233
- **Calinski-Harabasz Index**: 1078.97
- **Davies-Bouldin Index**: 0.9002

Summary of Evaluation Metrics

The following tables summarize the evaluation metrics for each clustering algorithm on both datasets.

Table 1. Zebra Finch Dataset Evaluation Metrics

Algorithm	Silhouette Score	Calinski-Harabasz Index	Davies-Bouldin Index
KMeans	0.6946	6009.96	0.5460
SOM	0.6934	6002.95	0.5611
Hierarchical	0.5626	1810.80	1.4346
CMeans	0.6520	5417.30	0.7093
DBSCAN	0.8070	172.36	1.3761
Distribution-Based	0.1415	1269.96	1.2446

Table 2. Spike Dataset Evaluation Metrics

Algorithm	Silhouette Score	Calinski-Harabasz Index	Davies-Bouldin Index
KMeans	0.5319	942.91	0.8339
SOM	0.7428	3818.60	0.3670
Hierarchical	0.7344	3621.75	0.3786
CMeans	0.7428	3818.60	0.3670
DBSCAN	0.6732	1123.91	1.1497
Distribution-Based	0.3233	1078.97	0.9002

These results highlight the variability in clustering performance across different algorithms and datasets. SOM and CMeans consistently show strong performance for the Spike dataset, while DBSCAN performs well for the Zebra Finch dataset. The choice of clustering algorithm should be informed by the specific characteristics of the dataset and the clustering objectives.

VI. Conclusion:

This study applied six clustering techniques—KMeans, Self-Organizing Map (SOM), Hierarchical Divisive Clustering, Fuzzy CMeans, DBSCAN, and Distribution-Based Clustering—to two spike train datasets, one from Zebra Finch recordings and another comprising pre-processed spike waveforms. Each method was evaluated on its ability to effectively classify and distinguish spike patterns within the datasets, with performance metrics including the Silhouette Score, Calinski-Harabasz Index, and Davies-Bouldin Index.

Key Findings

1. **KMeans Clustering:** KMeans clustering showed competitive performance across both datasets. For the Zebra Finch data, it achieved a Silhouette Score of 0.6946, indicating clear cluster separation, and performed well on the Spike dataset with a score of 0.5319. The simplicity and efficiency of KMeans make it a strong candidate for spike sorting tasks, particularly when the number of clusters is known a priori.

2. **Self-Organizing Map (SOM):** SOM demonstrated robust performance, particularly in the Spike dataset, where it achieved the highest Silhouette Score of 0.7428. SOM's ability to preserve the topological structure of the data makes it an excellent tool for visualizing and clustering high-dimensional neural data.

3. **Hierarchical Divisive Clustering:** This method provided a clear hierarchical structure of the data, which can be particularly useful for understanding the relationships between clusters. However, its performance, as indicated by a lower Silhouette Score for the Zebra Finch data (0.5626), suggests it may not be as effective in datasets with less defined cluster boundaries.

4. **Fuzzy CMeans Clustering:** Fuzzy CMeans offered a flexible clustering solution by allowing data points to belong to multiple clusters. This approach is beneficial for spike data where overlaps between clusters are common. It achieved a Silhouette Score of 0.7428 for the Spike dataset, matching the performance of SOM, but slightly lower for the Zebra Finch data (0.6519).

5. **DBSCAN:** DBSCAN excelled in identifying clusters of varying density and handling noise within the datasets. It achieved the highest Silhouette Score for the Zebra Finch data (0.8070), indicating its strength in detecting meaningful clusters in noisy datasets. Its performance on the Spike dataset was also notable, with a score of 0.6732.

6. **Distribution-Based Clustering:** This method, though providing a probabilistic framework for clustering, underperformed relative to other techniques, particularly with lower Silhouette Scores (0.1415 for Zebra Finch data and 0.3233 for Spike data). This suggests that distribution-based models may struggle with the complex and noisy nature of spike train data.

Implications and Future Work

The results of this study highlight the strengths and limitations of various clustering methods in the context of spike sorting. KMeans and SOM emerged as particularly effective methods, with SOM showing superior performance in high-dimensional data visualization. DBSCAN's ability to handle noise and identify clusters of varying density makes it a valuable tool for neural data analysis.

Future work should focus on integrating these clustering methods with real-time spike sorting systems to enhance their applicability in neurophysiological research. Additionally, exploring hybrid models that combine the strengths of different clustering techniques could further improve clustering performance. The use of deep learning approaches, such as autoencoders and neural network-based clustering, represents another promising direction for future research.

In conclusion, the comparative analysis conducted in this study provides valuable insights into the suitability of various clustering methods for spike sorting. The findings can guide researchers in selecting appropriate clustering techniques for their specific datasets and objectives, ultimately advancing the field of neural data analysis.

VII. References:

1. Quiroga, R. Q., Nadasdy, Z., & Ben-Shaul, Y. (2004). Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering. Neural Computation, 16(8), 1661-1687.
2. 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(4), R53-R78.
3. Harris, K. D., Henze, D. A., Csicsvari, J., Hirase, H., & Buzsáki, G. (2000). Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements. Journal of Neurophysiology, 84(1), 401-414.
4. Pedreira, C., Martinez, J., Ison, M. J., & Quian Quiroga, R. (2012). How many neurons can we see with current spike sorting algorithms? Journal of Neuroscience Methods, 211(1), 58-65.
5. Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: a review. ACM Computing Surveys (CSUR), 31(3), 264-323.
6. MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In Proceedings of the fifth Berkeley symposium on mathematical statistics and probability (Vol. 1, No. 14, pp. 281-297).