Discovering Similar Spike Patterns in High Dimensional Biomedical Signals

Sikder Tahsin Al-Amin
Department of Computer Science
University of Houston
USA

Robin Varghese
Department of Computer Science
University of Houston
USA

David Lloyd
Department of Biomedical Engineering
University of Houston
USA

Maria A. Gonzalez-Gonzalez
Department of Biomedical Engineering
University of Houston
USA

Mario I. Romero-Ortega
Department of Biomedical Engineering
University of Houston
USA

Carlos Ordonez
Department of Computer Science
University of Houston
USA

***Abstract*—We discuss our progress towards solving a challenging biomedical problem: identifying similar patterns among multiple physiological nerve signals hidden in high throughput data, collected from micro electrical sensors implanted in several animal organs. The problem is difficult because patterns come as spikes within millisecond time-windows, data sets have high dimensionality and there is background electrical noise. A previous analytic system discovers patterns combining PCA dimensionality reduction and K-means clustering, which is slow and misses important patterns hidden by noise. Moreover, it requires reading the data set several times and it requires multiple languages and tools. With such limitations in mind, we present an improved, integrated system that effectively allows the discovery of more accurate patterns, with automated algorithm parameter tuning, by learning model parameters incrementally exploiting summarization. Our integrated solution combines signal filtering, variable construction (feature engineering) and multidimensional data summarization, for a tighter and more effective integration of PCA and K-means clustering. We present preliminary experiments on signals collected from key nerves in a rat. We show our method discovers more patterns in larger time-windows, with better noise filtering, taking less time. In the future, we plan to link signal patterns to specific physiological functions, paving the way for innovative medical treatment via nerve stimulation.**

I. Introduction

Analyzing noisy time series data and finding patterns within them is a challenge in Big Data, especially when applied in Health and Medicine. Data sets can come from multiple sources (e.g. devices, sensors), and there is no explicit method to analyze these data sets together. Data also may not come at the same timestamp from each channel, creating unbalanced time series data across sources. Finding patterns from these data sets is also difficult as the signals include electrical and motion noise that overlap between channels. Furthermore these data sets can be huge (in the order of GB, TB) and current approaches require reading the data set several times. As a result, a major barrier in processing these data sets is RAM limitations. By utilizing many significantly smaller data summarizations, our incremental algorithm overcomes the limitation of RAM capacity. Filtering out noise in the data and detecting spikes from the filtered data requires 2 passes through the data set. Both steps are critical to achieve an accurate model and interpretation of the model. Filtering noise is necessary when working with animal neural signals as many dimensions result in being highly correlated as shown in Section IV. The noise can be best attributed to overlapping electronic noise in the signal across channels creating inaccurate correlations. Once noise is removed, spikes can be detected more accurately. In our improved algorithm because the data set is summarized incrementally, filtering noise and detecting spikes is done together in one pass and is further expanded on in Section III. Previous systems and their implemented algorithms for discovering patterns in biomedical signals, may require the use of different languages and proprietary software. One system for detecting patterns in neuron signals from animals [5] [17], is designed with many different tools and programming languages such as Plexon, MATLAB and Python.

We present a unified contemporary system completely designed in Python. Python is the preferred language in Big Data and Data Science because of it is general-purpose capabilities, maintains a high level of abstraction, and is not proprietary. Our system combines the improved algorithms for signal filtering and spike detection, variable selection through Singular Value Decomposition (SVD), and and data summarization, for an overall more effective implementation of PCA and K-means clustering, on patterns found in neural signals from Normotensive Wistar-Kyoto (WKY) rats. We perform singular value decomposition (SVD) on the correlation matrix to do variable selection as a variant calculation of PCA dimensionality reduction. Therefore in keeping the original variables, we can convey more accurate interpretations and meaningful analysis than the principal components. Then on the reduced dimensions, using a clustering algorithm such as K-means , k spikes are classified for each channel.

978-1-6654-8045-1/22/$31.00 ©2022 IEEE

The simultaneous activity of different organs is fundamental to understanding the complex physiology of an individual as a system and designing therapeutic strategies when detecting failures of any component, are both prominent biomedical challenges seen today. Finding signal patterns of activity from different organs in physiology highly suggest their functional connectivity. Conventional approaches read the data set several times, testing different parameters to discover patterns using a combination of dimensionality reduction and clustering algorithms, which is time-consuming, is subject of bias, and often do not provide with accurate results.

A. Long-term Benefits in Biomedicine

Understanding the inter-organ communication will benefit medical knowledge, biomedical technology, thereby improving health care. One of the main aims in Biomedicine is the design of bi-directional or closed-loop systems that allow the sensing in real time of signal changes associated to pathologies. By detecting and deciphering these signals, a modulatory action can be applied as a response to provide with a therapeutic effect. Our research will benefit the medical community in many ways. The implementation of new technologies with prophylactic benefits based on multi-organ state measurements and dysfunctional patterns detection, is a matter of improving the quality of new treatments and increase the life expectancy. On the other hand, it will benefit patients, the end user of biomedical technology. For instance, the pacemaker, invented in 1958, has saved millions of lives since then, and has extended the life expectancy for those patients suffering of arrhythmias. On these times it is imperative to move to the next step and implement integrative-multiorgan activity-based technologies, which is promising to design patient-specific therapies, instead of treating a disease that often presents different medical signs among patients.

B. Contribution

This work aims to efficiently analyze multiple continuous signals over time and detect signal patterns (spikes) across them with a common timestamp. We use two fundamental machine learning techniques: dimensionality reduction and clustering. Dimensionality reduction is needed to make clustering work and clustering finds similar patterns. Our contribution stems from combining clustering and PCA in a new way to detect new patterns and more efficient processing for big data, exploiting data summarization via sufficient statistics to filter data and compute ML models more efficiently.

II. Definitions

A. General

We define $\{D_1, D_2, ..., D_M\}$ as M spliced raw data sets, which can be intuitively understood as M streams with timestamped values. In our biomedical problem, they represent M signals collected from M channels. Each D_j is a large data set and N represents the maximum number of rows across channels. We assume $|D_j| \approx N$ rows. That is, $N_j = |D_j| = O(N)$. A signal measurement in each channel is represented by a pair, (t_{ji}, v_{ji}), where t_{ji} is the collected timestamp and v_{ji} is its measured signal value (for $j = 1, ..., M$ and $i = 1, .., N$). Intuitively, the data set has about N rows and $2M$ columns, where each row corresponds to one matched timestamp and M real values.

B. Biomedical Pre-Processing

Since the raw, unfiltered, signals $\{D_1, D_2, ..., D_M\}$ include noise, we need to filter the signals in each M channel and detect n peaks (spikes) per channel with a window of size d, resulting in a d-dimensional (transformed) data set. We define the filtered data sets as $X_1, X_2, ..., X_M$, represented as matrices. Thereafter, these M matrices are used as the input to machine learning algorithms. Each of these filtered data sets will have n_j rows (spikes), where $n_j = O(n)$ ($n << N$) and d dimensions (window size). Therefore, n is the data set size for ML analysis. These data sets $(X_1, X_2, ..., X_M)$ will be the input for our algorithm. The output will be M clustering models, corresponding to the M channels, where each model is a set of k clusters of the n_j input vectors. To simplify interpretation k is uniform across channels.

III. Algorithms and Systems

A. Background Of Algorithms

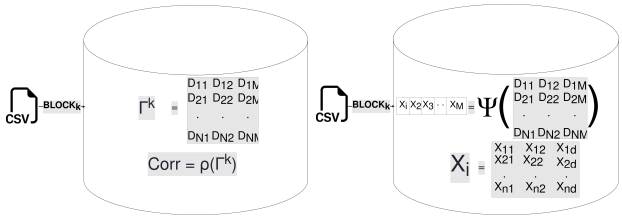
Real signal data sets, contain noise that will interfere with the learning of machine learning models. The model will falsely contribute this noise when generalizing to the data set, resulting in an inaccurate classification of a pattern. In the case of neural signals from WKY rats, overlapping patterns of noise across channels create inaccurate correlations and difficulty detecting spikes. This emphasizes the need of some dimensionality reduction method to filter noise and detect spikes. The most common and robust method being Principal Component Analysis (PCA). In PCA, dimensionality is reduced to principal components and computations are performed on the respective components, losing original interpretation of the data. We expand our improvement in dimensionality reduction in further detail below.

Although a plethora of clustering algorithms exist and are being improved upon daily in machine learning, K-means remains to be the predominant or workhorse clustering algorithm. K-means simplicity in algorithm, flexibility, and easy visualization and interpretation of clusters is preferred over possibly more accurate, outlier sensitive, slower algorithms. Other than the assumption of spherical like clusters, K-means is more robust than other clustering algorithms. Other algorithms like DBSCAN, may require assumptions of statistical properties of the data set, such as being dense or sparse.

B. Previous Algorithms and Systems

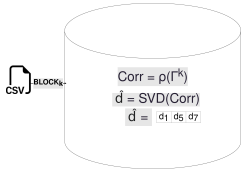
First, we discuss the previous algorithm and system [5], [17] to detect patterns from neuron signals in animals. The main steps of the algorithm are below:

1) Filtering Out Noise In Data. (Plexon)
2) Detecting Spikes From Filtered Data. (MATLAB)
3) Reduce Dimensions to k principal components. (Python)

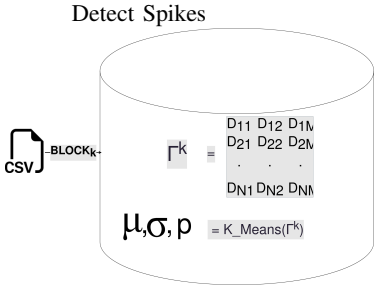


(a) Step 1: Compute Correlation Matrix

(b) Step 2: Filter Noise And Detect Spikes



(c) Step 3: Reduce Dimensions



(d) Cluster On Reduced Dimensions

Fig. 1: Steps of the new algorithm

4) Clustering Top k Principal Components. (Python)

The first step was to filter the noise in the raw data set obtained from API's accessible through Plexon, a propietary software. However, it was assumed that there was only one data set to hold the timestamp and their values for each channel. The next step was to detect spikes from the filtered data set and create M new data sets ($X_1 ... X_M$), corresponding to the M channels. Each M data sets are of size $n \times d$, where n corresponds to the number of spikes, and d is size of the window capturing the pattern. A spike is represented as $1 \times d$ row-vector of fixed window size. Then, the final steps were applying PCA to reduce the dimensions of the spikes (vectors) and running K-means clustering on the principal components to detect the spikes.

In Section III-C, we propose a new algorithm in identifying patterns from neuron signals in animals more efficiently, while preserving interpretation by keeping the original values. Our system is solely implemented in Python and is more contemporary, optimized and efficient in finding new patterns from raw signals while preserving the original signal values. In the next part, we will discuss our solution in detail.

C. Our Improved Algorithms and Systems

Here, we discuss our proposed solution. The main steps of the solution are given below.

1) Compute Correlation Matrix Amongst Channels. (Python)
2) Filter Noise And Detect Spikes. (Python)
3) Reduce Dimensions to $\hat{d}$ Dimensions. (Python)
4) Cluster $\hat{d}$ Dimensions. (Python)

1) Computing correlations between channels: To get initial exploratory analysis of the channels and how they are correlated, we compute the correlations based on the split raw data sets ($\{D_1, D_2, ..., D_M\}$). Raw data sets have a lot of noise and it is important to get an idea of how correlated they are, so we can eliminate the noise and analyze on the signal spikes. As the raw data set size is extremely large ($N_j = O(N)$), we can use our improved incremental algorithm [3] to compute the correlation matrix efficiently. The main idea is that we compute a summarization matrix based on the combined raw data sets by multiplying the combined raw data set (represented as a matrix) with its transpose for each channel (e.g. $D_1 D_1^T, ..., D_M D_M^T$). We compute this using sum of vector outer products while reading the data set by blocks. After our summarization matrix is computed, we compute the correlation matrix on it. This way, even if the size of the raw data is larger than the main memory, we can still compute it accurately and efficiently using smaller summarizations. Since our improved algorithm summarizes the data set incrementally, it is necessary to read the raw data set only once, a significant advantage over previous algorithms. Additionally, our system has the advantage of being a unified and contemporary solution, in contrast to previous systems constructed of different languages and proprietary software. The data pipeline from reading the input data set, to outputting a machine learning model, is programmed entirely in Python.

2) Filtering and detecting spikes: From the highly correlated channels from the previous step, we filter the noise and detect the spikes (peaks). In comparison to the previous approach, here filtering noise and detecting spikes are done together in one pass. As mentioned in Section II, each signal has a timestamp and a value (t_{ji}, v_{ji}) for each channel. We use a threshold (ψ) to detect the spikes, meaning points above the threshold are detected as spikes and the others are discarded ($v_{ji} >= \psi$). Typically, we set the threshold at three standard deviations ($\sigma = 3$) above the mean (μ) voltage. First, we find the peaks for each channel $j = 1..M$ and if it is too close to the beginning or the ending of the recording, we discard it as it can be mostly noise. Otherwise, we slice the peak based on our window size (d) where the window size consists of a continuous series of time points. That is, we are building the vector for each spike as we read through the data set. The output of the spike detection will be reduced data sets $X_1, X_2, ..., X_M$, each having a total of n_j spikes/peaks at d time points (window size). The number of rows ($n_j = O(n)$) will represent the number of spikes that are detected above the threshold and may vary across channels. However, the window size d is fixed across all the channels. As mentioned before, these detected spikes are stored in $X_1, ...X_d$ matrices and used as an input for further analysis. As the number of detected spikes (n) are much less than the raw data size (N), they easily fit in the main memory.

3) Reducing dimensionality and identifying dimensions: As mentioned earlier, d represents the window size of time points (vector). In the next step, we reduce the dimension of d from the detected spikes. As the values are continuous, we can use any standard dimensionality reduction algorithm like PCA for this task. However, PCA will return the principal components and it has already been explored before [5], [17]. Here, we perform singular value decomposition (SVD) on

the correlation matrix to do the variable selection. From d variables, we select $\hat{d}$ variables, where $\hat{d} \subset d$. Our idea is to select the original variables rather than the principal components and analyze further based on that. We believe original variables convey more useful information and we can get more meaningful analysis. From the SVD, we analyze the Eigenvectors, and select the variables having the maximal absolute variance. These variables are used for further analysis in the next steps.

4) Clustering time window vectors representing spikes: Based on the variables selected in the previous step ($\hat{d}$), we cluster the spikes. It is not a good idea to cluster based on 46 variables and it often leads to poor performance. However, to ensure we are not losing too much information by reducing the dimensions, we only select the dimensions having high variance. The clustering can be performed with any standard clustering algorithm like K-means. We need to run the clustering algorithm each time for all the selected channels and select k clusters. The algorithm will then cluster the k similar spikes for each channel.

D. System Designs

Prior systems may implement each step of their algorithm in different proprietary software or programming language, such as in the case of the system mentioned above utilizing Plexon, MATLAB, and Python. The current preference in Big Data and Data Science is to use Python. This is for its properties of being general-purpose, ability to manipulate data frames (pandas), efficient vectorized mathematical operations (NumPy), and while being a high-level abstract programming language. Our improved system appeases this demand by providing a contemporary solution, completely programmed in Python and avoids proprietary software like Plexon.

Rather than utilizing Plexon and MATLAB for noise filtering and spike detection, we use only Python. To compute the data summarizations and the correlation matrix NumPy is used. To filter and detect spikes from the raw signals, SciPy is used. As seen in Section IV, we noticed a significant speed up in the system with this transition, in comparison to the previous system.

E. Algorithmic and System Comparisons

A few differences in the overall algorithm between these two systems are in step 1) and step 2). In our proposed system, step 1), computes a correlation matrix using the raw input data set and is read as a .csv. Furthermore, step 2) of our proposed system, performs both step 1) and step 2) of the previous system, simultaneously. This emphasizes the advantage that our proposed system provides, in that only one pass of the data set is required to perform filtering and spike detection. By utilizing a threshold of $\sigma = 3$, we are able to accurately filter noise and capture spikes . In step 3), in contrast to the previous system, we retain the original d variable values rather than the principal components. This is to provide a more meaningful interpretation of the model and its results. As a result, step 4) in both systems compute a clustering algorithm, but in the new system it is computed on the original reduced dimensions. The raw data set contains high correlations amongst channels, as seen in the correlation matrix of the raw data set in Table II. These high correlations can be best attributed to the overlapping noise patterns amongst all channels, resulting in high correlations. However, this is not accurate. In the filtered data set as seen in Table II, the channels actually yield little correlations. Therefore as result, models reduce overfitting and learn parameters more accurately. Additionally our system can handle if raw data sets size are bigger than RAM capacity, through incremental learning.

F. Time Complexity and I/O Analysis

To compute our summarization matrix on the raw data (N), the time complexity will be $O(M^2N)$, and the space complexity is $O(M^2)$ [3], considering M as the number of channels. So, even if the raw data sets are bigger than the main memory, our summarization matrix can easily store the summary in the main memory. Analyzing raw signals would not have been possible with the previous approach even the data set size exceeded the RAM capacity. We compute correlation matrix from the summarization matrix. Then, we need one pass to the raw data sets to filter and detect the spikes together. However, we needed at least two passes (one to filter first, and one to detect spikes) to detect the spikes. After spike detection, the filtered data sets will have $\approx n$ rows, and we need one pass per channel to reduce the dimensions.

IV. Experimental Evaluation

TABLE I: Raw and Filtered Data sets. M = 4 channels. n = number of rows and in the case of channels n = number of spikes = number of rows

Ch.#	data set	n	size
Input Signal	Raw data set	31474451	8.2GB
Ch.1 (discard)	CH1 NAME	18539	9MB
Ch.2	SP1	60138	34MB
Ch.3	SP3	59044	33MB
Ch.4	RN	59891	34MB
Ch.5	cVN	19804	11MB
Ch.6 (discard)	CH6 NAME	28447	14MB

This section presents preliminary experimental results, showing a modest, but important qualitative improvement, over previous solutions (more patterns), but significantly more efficient processing (less time).

A. Experimental Setup

Biomedical aspects:: This paragraph discusses our experimental setup and data collection process from a biomedical perspective. Normotensive Wistar-Kyoto (WKY) rats were anesthetized with 2 % isoflurane vapor. The left femoral artery was exposed and implanted with a cannula pressure transducer connected to a wireless telemetry device (Data Sciences International) transmitting heart rate (HR), systolic, diastolic, and mean arterial pressure (MAP) every second. Via midline abdmoninal incision, the spleen and right kidney were exposed. Platinized graphene oxide electrodes (sutrodes) were

TABLE II: Correlation among channels on raw data vs filtered data (Ch2: SP1, Ch3: SP3, Ch4: RN, Ch5: cVN)

			Raw Data		
Ch.#	Ch.	SP1	SP3	RN	cVN
Ch.1	CH.1 NAME	discard	discard	discard	discard
Ch.2	SP1	1.00	0.98	0.97	0.73
Ch.3	SP3	0.98	1.00	0.97	0.70
Ch.4	RN	0.97	0.97	1.00	0.72
Ch.5	cVN	0.73	0.70	0.72	1.00
Ch.6	CH.6 NAME	discard	discard	discard	discard

			Filtered Data		
Ch.#	Ch.	SP1	SP3	RN	cVN
Ch.1	CH.1 NAME	discard	discard	discard	discard
Ch.2	SP1	1.00	0.01	0.01	0.01
Ch.3	SP3	0.01	1.00	0.02	0.01
Ch.4	RN	0.01	0.02	1.00	0.01
Ch.5	cVN	0.01	0.01	0.01	1.00
Ch.6	CH.6 NAME	discard	discard	discard	discard

implanted on the right renal nerve (RN) and the splenic nerve terminal branches 1 and 3 (SP1 and SP3) as described in [8]. A midline incision on the neck was used to access the left cervical vagus nerve (cVN), and a sutrode was implanted on the nerve. Neural signals were recorded with an OmniPlex recording system (Plexon, inc.) at 40 kHz sampling rate. Simultaneous recording of HR, MAP, and neural activity from the RN, SP1, SP3, and cVN at rest, with saline control injection, and with intravenous Phenyephrine (Phe, 10 µg/kg; vasopressor; P6126, Sigma Aldrich) injection to increase blood pressure. The connections were: Channel 2 - SP1, Channel 3 - SP3, Channel 4 - RN, and Channel 5 - cVN.

Signal Recording and Data Set: In our current lab setup, we used M=4 channels, but this number will grow as more electrodes can be inserted. Each channel has $N \approx 31M$ (Million) data points with their corrsponding timestamps and signal values. The data sets are stored in CSV files. The continuous data files contain both raw and filtered columns for each channel with 31M rows, resulting in an $31M$x8 matrix of voltage signals from the nerves of an animal (mouse). When exported to CSV format, this data set results in 6 to 8 Gb of data per experiment, with a single subject for this study undergoing 3-5 experiments. Thus the total amount of data expands rapidly for even a modestly sized experimental design.

Hardware and Software: As for hardware setup, we conducted our experiments on a server with Intel Pentium (R) Quad-core CPU running at 1.60 GHz with 8 GB RAM, 1 TB disk, and the Linux OS (Ubuntu).

We used Python as our choice of data science programming language because it provides a signal processing library, many math libraries (numpy,scipy) and computation time is fast. We must mention the previous algorithm was programmed in MATLAB, but both old and new algorithms were programmed in Python.

B. Detecting High Correlations among Channels

As mentioned in Section 3, we filter and detect spikes from raw signals. We do the filtering because the raw data captures noise from the system. To prove our hypothesis, we compute the correlation between the channels before and after detecting the spikes. Table II shows the correlation computed on raw data and on filtered data. We use our previous solution with the summarization matrix [3] to compute the correlations. However, compared to [3], we use only Python (NumPy library) instead of C++ to compute the summarization matrix, and for correlation matrix computation, we use Python as before. We can see from Table II that all the channels are extremely correlated on the raw data, but after removing noise, there are no correlations at all ($< 1\%$). Hence, it proves our assumption that the high correlation on the raw data sets is mostly based on overlapping noise and step 1 of our algorithm to detects this.

C. Pre-processing: Filtering Noise and Detecting Spikes

As mentioned in Section 3, we threshold and detect the spikes from each channel. After filtering and detecting we end up $n \approx 60K$ spikes (peaks) for channel SP1, SP3, and RN (channel 2,3,4 respectively), $n \approx 19K$ for channel cVN (channel 5). Each channel has a window size of $d = 46$ time points.

To filter and detect spikes (peaks) from the raw signals, we use Python SciPy library which can detect spikes from a 1D array and finds all local maximum by simple comparison of neighboring values. We set the threshold (ψ) to three standard deviations (σ = 3) above the mean voltage. For each channel, we repeat the same experiment to detect the spikes $(X_1, ..., X_M)$. After detecting the spikes, as mentioned in Section 3, we have spikes of a fixed window size with d time points. We set $d = 46$ as the total number of time points, meaning, each detected spike is a vector of size 46. However, the total number of spikes (row size) vary across channels. For our experiments, the number of detected spikes are roughly $n \approx 60K$ for channel 2,3,4 and $n \approx 20K$ for channel 5. Now, we reduce the dimension of the detected spikes from each channel. As mentioned in Section 3, we wanted to analyze the original signal voltages and we perform SVD on the correlation matrix to select the variables with high variance.

D. Quality Comparison: Clustering on Projections with Selected Dimensions from PCA

After detecting spikes and reducing dimensionality, we perform clustering on the reduced dimensions. We use K-means as our choice of the clustering algorithm. Figures 3 and 4 show clusters obtained by the previous algorithm. We can see that majority are just horizontal stripes. A key assumption in K-means, is that the clusters are spherical and may not work as efficient with clusters of other shapes. Especially in discovering patterns across channels, clusters not achieving this property will have much overlap and little interpretation. In contrast, Figure 5 and Figure 6 show clusters discovered by

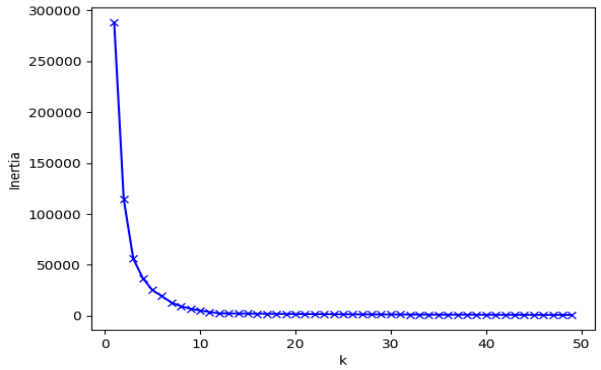
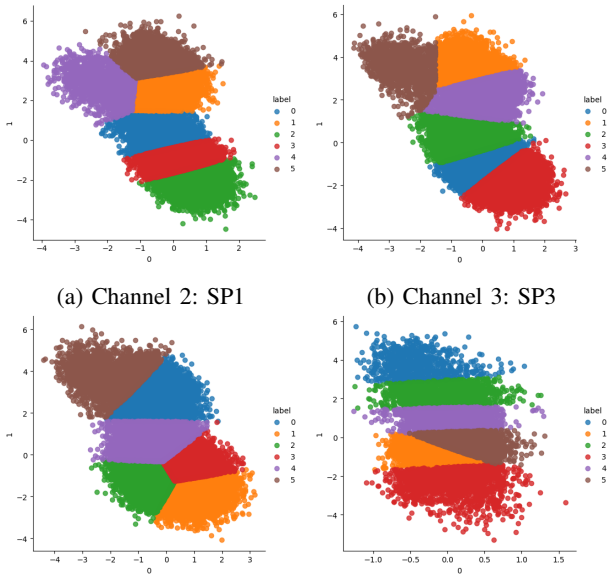


Fig. 2: Elbow method to select number of clusters (k) in K-means

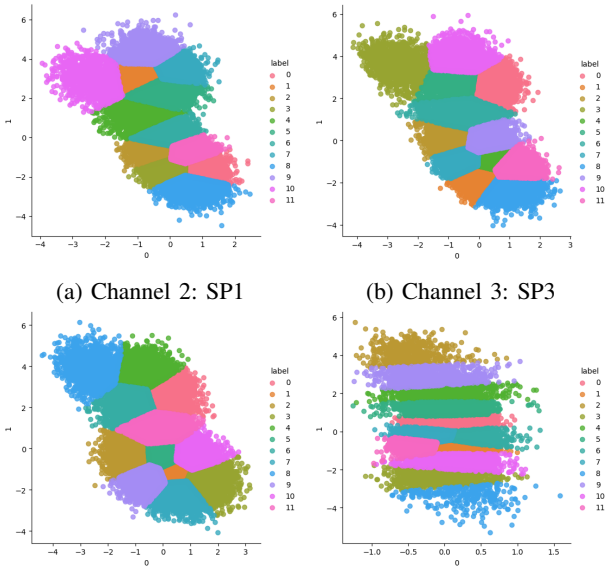


(a) Channel 2: SP1 (b) Channel 3: SP3

(c) Channel 4: RN (d) Channel 5: cVN

Fig. 3: Clustering spikes for each channel using the previous algorithm (Top 2 PCs , $k = 6$)

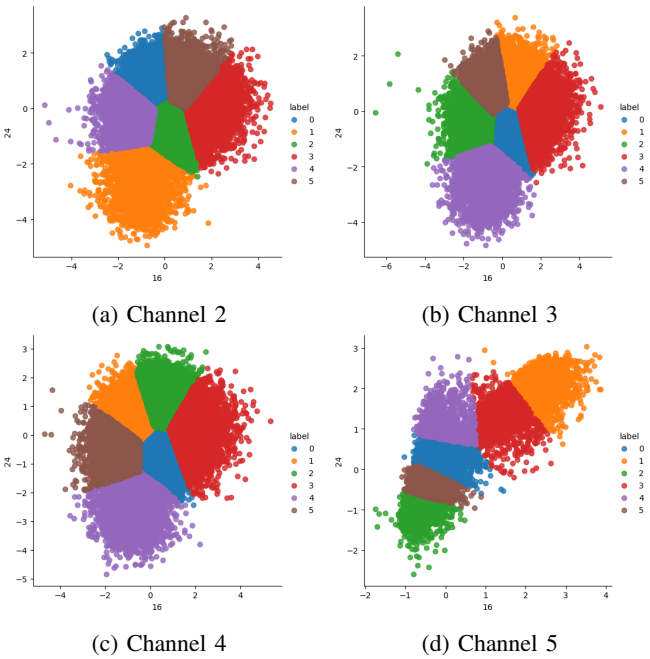
our improved algorithm. These new clusters resemble more closely to spherical clusters and as seen in Figure 6, when k increases to 12, the clusters shape continues to improve spherically. In our new algorithm we project onto the top dimensions, which exhibit the highest correlation values in the principal components. For our experiments, we select $\hat{d} = 2$ variables with the maximum variance for each channel. From the 46 time points, we chose the 16th and 24th time points. To determine k, the optimal number of clusters, we used the popular "elbow" visualization method and selected $k = 6, 12$. Figure 2 shows the plot of the elbow method for channel 2 (SP1). Even though not shown, the other channels show highly similar behavior in the elbow method as channel 2. The clusters in channels 2, 3, 4 behave the same way also. Our solution finds similar characteristics among these channels, which was not possible from the previous solutions clusters.



(a) Channel 2: SP1 (b) Channel 3: SP3

(c) Channel 4: RN (d) Channel 5: cVN

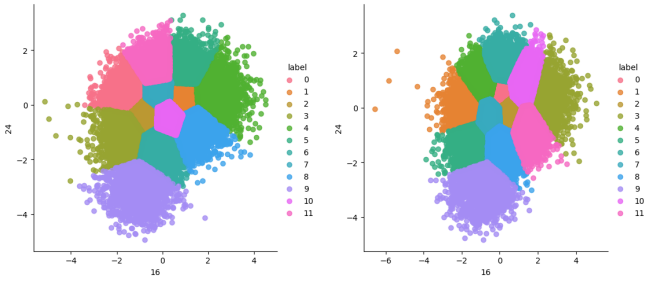
Fig. 4: Clustering spikes for each channel using the previous algorithm (Top 2 PCs , $k = 12$)



(a) Channel 2 (b) Channel 3

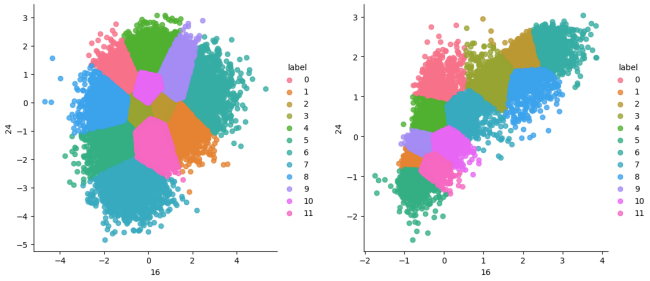
(c) Channel 4 (d) Channel 5

Fig. 5: Clustering spikes for each channel using our new algorithm ($d = 16, 24$, $k = 6$)



(a) Channel 2

(b) Channel 3



(c) Channel 4

(d) Channel 5

Fig. 6: Clustering spikes for each channel using our new algorithm ($d = 16, 24$, $k = 12$)

TABLE III: Computational time (in seconds) for different steps

Summarization	Correlation	Dim. reduction	Clustering
233	1	2	2

E. Time Performance Analysis

Here, we give an overview of the computational time using our solution. Table III shows the time needed by different algorithm steps. We omit time from the previous solution, which took more than 20 minutes and a more complicated setup with manual inspection in a GUI, MATLAB and Python. We can see that only the summarization matrix computation is computationally expensive as it is analyzing $31M$ records from each channel. However, we optimize the computation using NumPy vectorized dot product which is faster than performing the matrix multiplication using loops as shown in [3]. Correlation matrix is computed from the summarization matrix following the procedure in [2], which is done within a second. The other steps involving machine learning model computation is very fast and performed within seconds using Python standard libraries.

F. Biomedical Significance

Discovering the underlying connections among various neuronal hubs is extremely important to understand regulatory and processing circuitry in the central nervous system (CNS). In the cortex, electrically evoked potentials can define the connection type and order between cortical circuits to map function and structure [16]. By designing the experiment to probe various neuronal network in response to a single stimulus, machine learning techniques can be applied to identify common features and thus connections between these brain areas [10]. However, similar analysis has not been applied within the peripheral nervous system (PNS). This is especially relevant to autonomic regulatory circuits in the visceral organ, such as those that regulate blood pressure. We have previously explored the involvement of the spleen using advanced electrodes, but the full neural activity among the organs involved in BP regulation has not been fully explored [8]. By improving the accuracy and efficiency of extracting and clustering neuronal features, we can accelerate the development of neuronal circuitry mapping in the neuroscience field.

V. Related Work

Spike detection from a data stream has been used in many application areas like security systems, fraud detection, neural activity and so on. And a lot of research has been published [13], [1], [7] addressing this issue and many are still going on. However, most research focuses on improving efficiency of the spike detection algorithm and/or the data collection method from animals. Also, many of the analysis are done in proprietary software which makes it hard for other researchers to reproduce the analysis. In this paper, we have tried to tackle the problem from a big data and data mining perspective while maintaining the original pipeline. Our goal was to find the similar spikes in each channel with minimal computational complexity from the time series data. Our analysis on raw data is based on an efficient summarization matrix which can handle data sets bigger than the RAM both in a single machine [3] and in a parallel manner [2]. Summarizing a time series data has also been explored in [9] which finds top-k time series snippets from a large data set. However, our summarization matrix can be further used to compute basic statistics and accurate machine learning models compared to this approach.

Spike detection and clustering are the standard methods for analysis of neural signals recorded from both the central and peripheral nervous system (CNS and PNS). Currently, the gold standard is dimensionality reduction via PCA prior to clustering and analyzing firing rates of clustered features [5]. PCA and wavelet based decomposition before clustering in reduced dimensional space is still the gold standard [17]. By using firing rate analysis and manual alignment with electrical stimuli, underlying neural circuits can be probed, such as those in the cortex [16]. In the last two decades, the emerging of field of Bioelectronic Medicine (BM) as an alternative to drug-based therapies has gained attention in medical research [4], [6], [12]. BM is defined as the use of electrical signals to modulate the neural component in an organ, and obtain therapeutic outputs to treat medical conditions[11]. Developing effective BM treatments involves not only the implementation of new technologies on sensitive neural interfaces to deliver electrical current and sense the activity for closed-loop system, but the interpretation of the complex biological responses (i.e. neural activity). The ideal picture in BM would be deciphering electrophysiological patterns related to certain malfunctions and pathologies to apply the precise therapy and recover a health state. To achieve this, it is needed to identify biomarkers (patterned signals) and understand the output. This can

be achieved by data analysis, understanding the shape and underlying geometric structures arising from high-dimensional relations, and appears imminent the use of strong mathematical framework and manageable quantitative tools [8].

Paper [14] analyzes multimodal signals in medical data, which includes image, electrical and measurement data. We currently use electrical and measurement (BP) data and are likely to use images in future work. Furthermore, this paper shows the potential for a larger more integrated system. By utilizing many tools in Big Data such as natural language processing (NLP), sentiment analysis, and classification, much of a patients multi-modal health signals can be used to together for a far more meaningful doctor-patient encounter. Additionally, incorporating physiological nerve data and its applications, also can only aid in improving doctor-patient encounters. Research presented in [15] attempts to classify level of disease based on EKG signals, whereas we attempt to understand organ interactions via nerves (physiology). Our goals are different: disease classification vs physiological pattern identification. However, rather than using dimensionality reduction as in our experiments, here normalization and augmentation were applied to the raw Phonocardiogram signal data. It is evident that there is not a clear solution for the preprocessing of signal data and in particular biomedical signal data. As a long term goal, we hope our research will open new medical treatment.

VI. Conclusions

In this work we presented an improved and more integrated system, to detect similar spike patterns in high dimensional biomedical signals, received from micro electrical sensors in Normotensive Wistar-Kyot (WKY) rats, combining PCA and K-means clustering. Our system provides better interpretations of the model and data analysis. By identifying highly correlated dimensions within the time window using PCA and then computing clusters directly on the original dimensions, we maintain the original meaningful values of the data. On the contrary, instead of computing them on the principal components (PCs), this allows for a more clearer interpretation. We showed that the raw data sets from multiple channels are highly correlated because of overlapping noise and must be filtered out before detecting the patterns.

To efficiently filter noise, detect spikes, reduce dimensionality, and compute a clustering model, we used a summarization matrix. Biomedical data sets are often large and do not fit in main memory. To overcome main memory limitations of previous systems, we exploited the incremental property of the summarization matrix. Our system yields the following benefits: (1) We could discover new patterns, which were not found before with the previous algorithm. (2) Easier biomedical interpretation because identified dimensions point to the actual specific time points in the detected spike. Rather than the principal components which lose, interpretation. (3) Fully integrated in Python. (4) Memory efficient, Incremental Learning. (3) Faster processing.

A preliminary experimental evaluation on (WKY) rats, presented promising results on recently collected lab data. Our work opens new perspectives for future research on electrical signals in the biomedical domain. Scaling our proposed system to a growing number of channels, presents many solutions to Big Data and Biomedical challenges. More channels will introduce more overlapping noise and false correlations, hiding valuable information that can be gained from biomedical signals triggered from organ responses. We expect our system to allow the study of smaller time windows in analyzing spike shapes. We will develop evolving clustering models, to get clusters at different time points. Moreover, we plan to compute PCA and K-means at the same time, interleaving them, instead of two phases, with a more general summarization via sufficient statistics.

The previous system was built with a pipeline including Windows and MATLAB. To match modern industry standards in Big Data and Data Science, the entire system was reprogrammed completely in Python. Clusters formed in the previous system using the top PCs, did not achieve the spherical property required in k-means to perform well. In our new algorithm we project onto the top dimensions, exhibiting the highest correlation values in the PCs. For our experiments we select $\hat{d} = 2$ variables, with the maximum variance for each channel. Our system in comparison to the previous one, achieved far better experimental results. The new clusters formed from our system are more spherical and have less overlap, producing better analysis in spike patterns.

Studying optimal storage for multiple signals will enable more advanced correlation analysis. We anticipate that more modern electronic technology will allow collecting signals at a higher frequency. Therefore, producing signals with higher resolution. As our solution is deployed to healthcare, clusters can be linked to events with standard medical measurements including blood pressure, heart rate, sugar and oxygen levels, among others collected in real-time.

References

[1] Subutai Ahmad, Alexander Lavin, Scott Purdy, and Zuha Agha. Unsupervised real-time anomaly detection for streaming data. *Neurocomputing*, 262:134–147, 2017.

[2] Sikder Tahsin Al-Amin and Carlos Ordonez. Efficient machine learning on data science languages with parallel data summarization. *Data Knowl. Eng.*, 136:101930, 2021.

[3] Sikder Tahsin Al-Amin and Carlos Ordonez. Fast machine learning in data science with a comprehensive data summarization. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 2941–2948. IEEE, 2021.

[4] Karen Birmingham, Viviana Gradinaru, Polina Anikeeva, Warren M. Grill, Victor Pikov, Bryan McLaughlin, Pankaj Pasricha, Douglas Weber, Kip Ludwig, and Kristoffer Famm. Bioelectronic medicines: a research roadmap. 13(6):399–400, June 2014. Number: 6.

[5] David Carlson and Lawrence Carin. Continuing progress of spike sorting in the era of big data. *Current Opinion in Neurobiology*, 55:90–96, 2019.

[6] Patrick D. Ganzer and Gaurav Sharma. Opportunities and challenges for developing closed-loop bioelectronic medicines. *Neural Regeneration Research*, 14(1):46–50, January 2019.

[7] Yiyang Gong, Cheng Huang, Jin Zhong Li, Benjamin F. Grewe, Yanping Zhang, Stephan Eismann, and Mark J. Schnitzer. High-speed recording of neural spikes in awake mice and flies with a fluorescent voltage sensor. *Science*, 350(6266):1361–1366, 2015.

[8] Maria A. Gonzalez-Gonzalez, Geetanjali S. Bendale, Kezhong Wang, Gordon G. Wallace, and Mario Romero-Ortega. Platinized graphene fiber electrodes uncover direct spleen-vagus communication. *Communications Biology*, 4:1097, December 2021.

[9] Shima Imani, Frank Madrid, Wei Ding, Scott E. Crouter, and Eamonn J. Keogh. Introducing time series snippets: a new primitive for summarizing long time series. *Data Min. Knowl. Discov.*, 34(6):1713–1743, 2020.

[10] Kai J. Miller, Klaus-Robert Müller, and Dora Hermes. Basis profile curve identification to understand electrical stimulation effects in human brain networks. *PLOS Computational Biology*, 17(9):e1008710, 2021.

[11] Valentin A. Pavlov, Sangeeta S. Chavan, and Kevin J. Tracey. Bioelectronic Medicine: From Preclinical Studies on the Inflammatory Reflex to New Approaches in Disease Diagnosis and Treatment. *Cold Spring Harbor Perspectives in Medicine*, 10(3):a034140, March 2020.

[12] Valentin A. Pavlov and Kevin J. Tracey. Bioelectronic medicine: updates, challenges and paths forward. *Bioelectronic Medicine*, 5:1, 2019.

[13] Clifton Phua, Kate Smith-Miles, Vincent C. S. Lee, and Ross W. Gayler. Adaptive spike detection for resilient data stream mining. In *Data Mining and Analytics 2007, Proceedings of the Sixth Australasian Data Mining Conference (AusDM 2007)*, volume 70 of *CRPIT*, pages 181–188, 2007.

[14] Abrar Rahman, Ari Mitra, Fuad Rahman, and Marvin J. Slepian. Smart EHR - A big-data approach to automated collection and processing of multi-modal health signals in a doctor-patient encounter. In *IEEE International Conference on Big Data (IEEE BigData)*, pages 6198–6200. IEEE, 2019.

[15] Zeenat Tariq, Sayed Khushal Shah, and Yugyung Lee. Automatic multimodal heart disease classification using phonocardiogram signal. In *IEEE International Conference on Big Data (IEEE BigData 2020)*, pages 3514–3521. IEEE, 2020.

[16] Yukihiro Yamao, Riki Matsumoto, Takayuki Kikuchi, Kazumichi Yoshida, Takeharu Kunieda, and Susumu Miyamoto. Intraoperative Brain Mapping by Cortico-Cortical Evoked Potential. *Frontiers in Human Neuroscience*, 15:635453, 2021.

[17] James Zhang, Thanh Nguyen, Steven Cogill, Asim Bhatti, Lingkun Luo, Samuel Yang, and Saeid Nahavandi. A review on cluster estimation methods and their application to neural spike data. *Journal of Neural Engineering*, 15(3):031003, April 2018.