CHARACTERISING POWER GRID EVENTS USING UNSUPERVISED LEARNING

By

ERIC BRYAN KLINGINSMITH

A thesis submitted in partial fulfillment of
the requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

WASHINGTON STATE UNIVERSITY
School of Engineering and Computer Science, Vancouver

MAY 2016

To the Faculty of Washington State University:

The members of the Committee appointed to examine the thesis of ERIC BRYAN KLINGINSMITH find it satisfactory and recommend that it be accepted.

Xinghui Zhao, Ph.D., Chair

Scott Wallace, Ph.D.

Sarah Mocas, Ph.D.

ACKNOWLEDGMENTS

I would like to thank the Department of Energy / Bonneville Power Administration for their generous support through the Technology Innovation Program (TIP# 319) and for providing the PMU data used in this thesis. I also would like to thank Dr. Xinghui Zhao, Dr. Scott Wallace, Richard Barella, Duc Nguyen, and Kuei-Ti Lu for supporting me in this research and providing insight into their own solutions to similar problems. Specifically, I would like to thank to Dr. Xinghui Zhao for guiding this research and helping me compose my thoughts. I would like to thank Dr. Scott Wallace for continuing to push this research into new and interesting directions while also focusing current endeavors towards presentable results. I would like to thank Richard Barella for not only helping me with difficult problems, but also providing the relative phase features which a great deal of my research is based on, as well as being a true friend. Finally, I would like to thank Solongo Munkhjargal for continuing to encourage me in all my efforts.

CHARACTERISING POWER GRID EVENTS USING UNSUPERVISED LEARNING

Abstract

by Eric Bryan Klinginsmith, M.S.
Washington State University
May 2016

Chair: Xinghui Zhao

Recent advances in modern sensing, communication, and information technologies have lead to the Smart Grid as an emerging field with its own challenges. One of the most critical challenges is a need for automatic solutions to data analysis, driven by the continuous, high-frequency data collection on the grid. Clustering is an unsupervised machine learning technique that can be applied to gain greater understanding of data characterization by categorizing data into groups based on their similarities. In this thesis, we present four main contributions in using unsupervised clustering for characterizing line faults and generator fault anomalies in a smart grid. First, we present features that when paired with filtering and the DBSCAN algorithm accomplish a homogeneity score of 0.939. Second, we show that instantaneous clustering under the same features is far superior to that of time series clustering. Third, we explore cluster-specific classifiers and show that with only a fraction of training data, they perform just as well as regular classifiers trained on the whole dataset, yet in some cases, clustering provides most of the classification. Finally, we present a novel set of features that can be used for generator fault discovery. We show that these novel features along with reasonable data filtering can provide a homogeneity score of at least 0.96. These

findings shed light on applying unsupervised learning techniques to data collected on the smart grid for characterizing known events and identifying unknown anomalies.

TABLE OF CONTENTS

List of Tables

List of Figures

Dedication

I dedicate this research to my family for their love, patience, and continued support.

Chapter 1

Introduction

As technology improves, so does the practical applications for that technology. Thus, new fields in applied research are continually opening up. The smart grid [1] is one of these new emerging fields, in which modern sensing communication, and information technologies are applied to the electrical grid to improve its reliability and intelligence. Smart grid technology comes with its own challenges. For instance, the big data challenge becomes more and more pronounced in this field. Specifically, a large amount of data is being collected by the gird every day by *Phasor Measurement Units (PMUs)*, which measure phase angles and magnitudes of the electrical waves in real time, at a high frequency, ranging from 30 measurements per second to hundreds of measurements per second. Because the amount of data is so large, it is critical to develop intelligent and automated approaches for analyzing and extracting useful information from the vast PMU data streams. Since the modern electric grid is reliable in general, fault events only occur a small number of times in a year, and the duration of these events is usually less than 1/10th of a second, resulting in imbalance of the data in terms of the ratio of anomalies and normal operation. Therefore, it is particularly important to develop methods that aid in the discovery and classification of both normal and anomalous events. Clustering is one such tool that can be leveraged to help accomplish this

task. In this thesis, we investigate different ways of applying clustering methods to PMU data for the purpose of event characterization and categorization.

1.1 Smart Grid Background

Our dataset was obtained from Bonneville Power Administration (BPA), one of the first transmission operators to comprehensively adopt synchrophasor technology in their wide-area monitoring system. Currently BPA has PMUs on the power grid at different locations across the Pacific Northwest region of the continental United States. Each of these PMUs record various measurements on the gird at 60Hz. In total, these PMUs collect approximately 40GB of data every day. A very small portion of this data is anomalous. Our research focuses on two types of anomalous events or faults: line faults, and generator faults. Specifically, we want to help classify, and discover, different classes of electrical faults. When a fault occurs at one location on the power grid, most other locations on the grid also experience similar readings. Therefore, a single event can be observed from most locations on the grid. This presents an interesting challenge as we must take this characteristic into account when applying machine learning techniques such as clustering.

1.2 Overview of Dataset

BPA has provided two separate datasets to use. The first contains 31 locations from across the Pacific Northwest. These PMUs measure line and/or bus voltage across all three phases (A, B, and C). They also record positive sequence voltage and current phasors, frequency and rate-of-change of frequency. At each PMU, the data is recorded at 60Hz. Measurements from this dataset are primarily from 500KV and 230KV buses, although a small number of lower voltage transmission lines are also covered. This dataset was collected from the

period October 17, 2012 to September 16, 2013 and contains 114 documented faults that occur at a bus or on a transmission line instrumented with at least one PMU. The faults include instances of Line to Ground (LG) faults, Line to Line (LL) faults, and Three Phase (TP) faults, in that order of relative frequency. The dataset also contains No Fault (NF) data. The second dataset contains 41 locations also within the Pacific Northwest. They record the same data as the first at the same 60Hz rate. Measurements are still primarily from 500KV and 230KV buses. Data was collected for the entire month of October 2014. Other than the date range in which it was collected the main difference between this dataset and the first one is that BPA included all electrical faults on the grid and not just the ones that occurred at a bus or on a transmission line instrumented with at least one PMU. This second dataset was only used for case studies 3 and 4, in this thesis. A comparison of the two datasets is shown in Table 1.1.

Dataset	# Locations	Period	# Faults	Size	Format	Used in
1	31	10-17-2012 to 09-16-2013	114	34 GB	h5	Case Studies 1, 2, 3, and 4
2	41	10-01-2014 to 10-31-2014	26	1,271 GB	pdat	Case Studies 3 and 4

Table 1.1: Datasets Breakdown

1.3 Clustering

Clustering is an unsupervised machine learning technique which categorizes data into clusters such that the intra cluster similarity is maximized and the inter cluster similarity is minimized [2]. Each data point in the dataset being clustered is an n-dimensional point, where n is the number of features each data point has. Features can either be raw data or a mathematical calculation on raw data. To evaluate the similarity of data points there must be a similarity distance metric, which is used to calculate how closely related two data points are across the n-dimensional feature space. Therefore, selection of features and the similarity metric plays an important role in clustering, as they, along with the clustering algorithm,

determines how the points are clustered. An advantage of the clustering method is that it does not require training data as it is an unsupervised learning technique. In addition, it can be used to discover new taxonomies for a given dataset. These new groupings of data can be non-trivial, but may provide important insight into the structure of one's data. We believe that clustering can be used to separate fault events into the different classes of line faults. As discussed in section 1.1 the same fault can be observed from several locations. Each location may measure the same fault event each with slightly differently readings than the others, yet they all represent the same class of event. Therefore, clustering should be a good fit for grouping similar events at all locations on the grid into the same cluster as it should be able to ignore these minor differences within a given cluster while still being able to separate the different classes of events due to the major differences between classes. For these reasons we have explored different approaches to determining good clustering techniques for fault categorization on the smart grid, as well as use clustering to explore unknown signatures in our dataset.

1.4 Contributions

We present four main contributions in using unsupervised clustering for characterizing line faults and generator fault anomalies in a smart grid. First, we present features that when paired with filtering and the DBSCAN algorithm accomplishes a homogeneity score of 0.939. Second, we show that instantaneous clustering under the same features is far superior to that of time series clustering. Third, we explore cluster-specific classifiers and show that with only a fraction of training data, they perform just as well as regular classifiers trained on the whole dataset, yet in some cases, clustering provides most of the classification. Finally, we present a novel set of features that can be used for generator fault discovery. We show that these novel features along with reasonable data filtering can provide a homogeneity score of at

least 0.96. Clustering is an important tool to characterize and categorize data into different groups, but good feature selection and filtering are just as, if not more, important than the clustering algorithm used.

1.5 Outline

The rest of this thesis is as follows. Chapter 2 explores related works of what others have accomplished in this field. The Chapters 3-6 relate to our four case studies that make up the bulk of this work. These case studies each explore one of our contributions. Case study 1 explores how clustering can be used to classify line faults by clustering a single moment in the event window. Case study 2 attempts to improve the results from case study 1 by clustering the entire event window instead of a single moment. Case study 3 uses the results from both case study 1 and 2 to train cluster-specific Support Vector Machines (SVM). These SVMs are then tested against a new set of data and the results are compared to that of a single SVM over the entire dataset. In addition, we also compare cluster-specific classifiers to a set of randomly trained and tested SVMs. The last case study attempts to discover a new class of events called generator faults. Finally, Chapter 7 summarizes the results discussed in the previous case studies, and gives some concluding thoughts.

Chapter 2

Related Work

PMUs are widely used in the power grid to monitor its operational status with the aim of enhancing the situation awareness for power system operators. A significant amount of work has been done to detect or monitor certain conditions of a power grid by leveraging information extracted from PMU data. Jiang *et al.* propose an online approach for fault detection and localization using SDFT (smart DFT) [3]. Liu *et al.* use Frequency Domain Decomposition for detecting oscillations [4]. Kazami *et al.* propose a multivariable regression model to track fault locations [5]. Besides voltage and current magnitudes, which are the most commonly used features in detecting faults, phase angle measurements can also be used in detecting outages [6].

Most recently, with the emergence of big data analytics, new technologies are introduced to PMU data storage and processing. Most importantly, a variety of machine learning techniques have been applied to analyze PMU data for the purpose of recognizing patterns or signatures of events. Two widely adopted techniques are classification and clustering.

Classification methods employ supervised learning and therefore, after training, can identify known signatures or patterns. Our work focuses on applying Support Vector Machines (SVMs) for classification of data belonging to a given cluster. Both Ruping *et al.* and Wu *et*

al. explore how different SVMs can be applied to analyze data [7, 8]. Ruping *et al.* present how different kernels can be used for time series data [7], whereas Wu *et al.*, investigate how the combinations of kernels can be used on a dataset as opposed to a single kernel [8]. Ray *et al.* build Support Vector Machines and decision tree classifiers based on a set of optimal features selected using a genetic algorithm [9]. Support Vector Machine-based classifiers can also be used to identify fault locations [10], and predict post-fault transient stability [11].

Other than using SVMs, additional work as been done in data classification. Nguyen *et al.* developed a decision tree based on the J48 algorithm, for the purpose of detecting line events on the power grid using PMU data streams [12]. Zhang *et al.* propose a classification method for finding fault locations based on pattern recognition [13]. Their key idea is to distinguish a class from irrelevant data elements using linear discriminant analysis. The classification is carried out based on two types of features: nodal voltage, and negative sequence voltage. Similar classification techniques are used to detect voltage collapse [14] and disturbances [9] in power systems. Specifically, Diao *et al.* develop and train a decision tree using PMU data to assess voltage security [14].

Although classification methods usually can provide high accuracy, they require a substantial amount of labeled data for training. Data characterization mehtods are different in that they do not require training and instead group datapoints based on their similarity. The majority of the work presented in this thesis falls in this category. Specifically, we focus on data clustering. Unlike classification, clustering methods do not require labeled training data. Density Base Spacial Clustering of Applications with Noise (DBSCAN) was developed by Ester *et al.* as a clustering algorithm that uses the data's density as its means of clustering [15]. Hierarchical clustering is also widely used. Murtagh surveyed various hierarchical clustering algorithms [16]. Wagstaff *et al.* have shown that in real world data k-means clustering is not sufficient, and background knowledge is necessary to obtain the best results [17]. In this thesis we use k-maens, DBSCAN, and hierarchical clustering. Rasanen

and Kolehmainen use a statistical approach to perform feature selection for clustering [18]. In addition to clustering in general, clustering has also been used to categorize time series data. A significant amount of real world data has a time component. Both Liao and Fu separately have conducted extensive surveys of different time series clustering methods [19, 20]. Fu *et al.* have also proposed a pattern recognition algorithm for time series clustering [21]. Evolutionary clustering is explored by Chakrabarti [22]. Evolutionary clustering is used to optimize two potentially conflicting criteria: first, the clustering at any point in time should remain faithful to the current data as much as possible; and second, the clustering should not shift dramatically from one timestep to the next [22].

Clustering has previously been shown as a method to categorize PMU data. Antoine *et al.* identify causes for inter-area oscillations by clustering a number of parameters provided by PMUs, including mode frequency, the voltage angle differences between areas and the mode shapes [23]. It has been shown that by clustering these parameters, changes in inter-area oscillations can be explained. Clustering methods have also been proven effective in identifying different types of disturbances [24].

For both classification and clustering, feature selection is a critical step. This step can become highly complex for data with a time component (time series data). Note that PMU data belongs to time series data as each record has a GPS-synced timestamp associated with it. Due to the high dimensionality of time series data, a significant amount of work has been done in an effort to reduce the dimensionality. Ye *et al.* maximizes classification by choosing a subsequence of the time series that best represents of a given class [25]. Keogh *et al.* investigate how piecewise aggregation approximation (PAA) can be used to reduce the dimensionality of time series data and then explores how dynamic time warping (DTW) can be used as a similarity metric for clustering [26]. Later on, Keogh *et al.* have also explored approximates curves by developing a locally adapted curve approximation method [27]. Keogh *et al.* are not the only ones to use DTW as a similarity metric. Wollmer *et al.*, and Holt *et*

al. both explore DTW further as they both study how it can be applied to multidimensional data [28, 29]. In addition to DTW, other similarity metrics have been explored. Singhal and Seborg use degrees of similarity between two similarity metrics as a means of multivariate time series clustering using a modified k-means algorithm on continuous exothermic chemical reactor data [30]. Finally, both Xing *et al.* and Wang *et al.* provide surveys of various approaches [31, 32]. Wang *et al.* analyze various similarity metrics over 38 different time series datasets [32]. In this thesis, we analyze PMU data in multiple ways, and explore the potential of applying clustering methods to both instantaneous PMU data and time series PMU data.

Chapter 3

Case Study 1: Clustering on Relative Phase Features

This case study explores features, filtering, two clustering algorithms and how they can be applied to categorize and characterize PMU data into groups based on a known taxonomy of events. Specifically, there are four categoris in our taxonomy: Line to Ground (LG), Line to Line (LL), Three Phase (TP), and No Fault (NF).

3.1 Dataset specifics

For this study we used the first dataset described in Section 1.2. From this dataset we used 100 of the 114 documented fault events and 19 randomly chosen no fault events (normal data), resulting in 119 events in total. Since not all events or locations on the electrical grid have all the required data for calculating the features required for our clustering, some events and locations were excluded. On average, we gathered data at 56 different locations on the electrical grid for each of the 119 events, resulting in 6676 data points in total. Of the 6676 data points, 4935 are Line to Ground faults, 331 are Line to Line, 196 are Three

Phase, and the remaining 1214 are marked as No Fault.

3.2 Feature Selection

In our dataset, the measurements recorded by the PMUs are voltage magnitude and phase angle (positive sequence and A, B, C phases), current magnitude and phase angle (positive sequence only), and frequency (60Hz). It has been shown in previous work that the per-phase voltage magnitude is an effective measurement to identify different types of line events [33]. To enable efficient clustering in a 2-dimensional space while retaining all the information contained in the three phase voltage magnitudes, we developed two new features, namely $RelativePhase2over1$, and $RelativePhase3over1$, by synthesizing the three per-phase voltage values. These two features are calculated in 4 steps. First, we calculate steady state voltages (ss_a, ss_b, ss_c). Algorithm 1 is the method we use for calculating steady state.

Algorithm 1: Steady Voltage (dataMinute, bus, phase, t)

```
steadyData = grab a sufficient window of data prior to the fault at time t;
end = t;
while steadyData contains a one second window ending at end do
    window = a one second window ending at end;
    m = average(window);
    mn = min(window);
    /* check the window minimum for smoothness */
    if mn <= 0.9m then
        move end one cycle earlier in time
        continue
    end
    /* check the window edges for smoothness */
    if window edges < 0.99m then
        move end one cycle earlier in time
        continue
    end
    return m;
end
```

Second, we calculate the relative phase deviations (RP) from steady state for three phases, as shown in Equations 3.1, 3.2, and 3.3.

$$RP_a = |1.0 - \frac{v_a}{ss_a}| \tag{3.1}$$

$$RP_b = |1.0 - \frac{v_b}{ss_b}| \tag{3.2}$$

$$RP_c = |1.0 - \frac{v_c}{ss_c}| \tag{3.3}$$

In the above equations, v values are per-phase voltage magnitudes, ss values are steady state voltages on the corresponding phases. Third, we then sort these relative phase deviations in ascending order, resulting in a tuple, as shown in equation 3.4.

$$[RP_3, RP_2, RP_1] = sort([RP_a, RP_b, RP_c]) \tag{3.4}$$

Note that sorting the relative phase deviations simplifies the complexity of our feature space. Although by doing that we lose the information about deviations on specific phases, the relative deviation magnitudes among three phases are captured, which still enable us to differentiate fault types. This is useful because the various classes of faults do not depend on which phase deviates from steady state, but rather how many phases deviate from steady state.

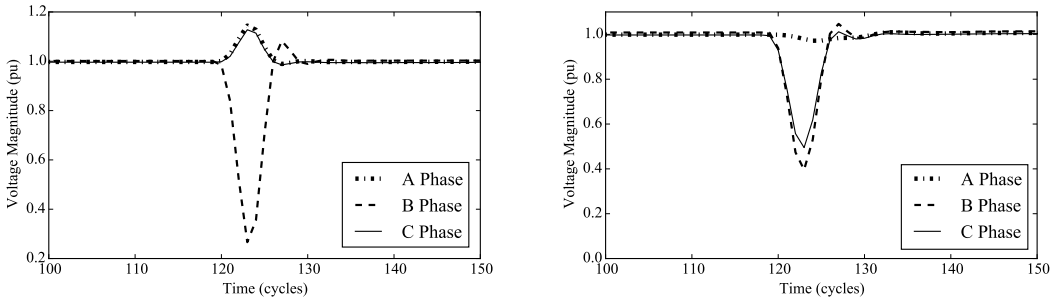
In the last step, we use the values in the relative phase deviation tuple to calculate the two new features, as shown in equation 3.5, and 3.6.

$$RelativePhase2over1 = \frac{RP_2}{RP_1} \tag{3.5}$$

$$RelativePhase3over1 = \frac{RP_3}{RP_1} \tag{3.6}$$

These two features retain the information contained in the three per-phase voltage values, yet provide a simplified 2-dimensional space for the clustering methods. In addition, by normalizing the magnitudes based on the steady state voltages, we eliminate the bias introduced by the absolute magnitude deviations, so that the characteristics, or patterns of different events can be better captured by the clustering methods.

3.3 Data Collection and Instantaneous Clustering



(a) A Typical Line to Ground Fault (b) A Typical Line to Line Fault

Figure 3.1: Typical Line Events on a Smart Grid

Some typical line events on the smart grid can be seen in Figure 3.1. This figure shows the per unit voltage for a typical LG fault and a LL fault. All three voltage phases are plotted. As can be seen in this Figure a typical fault lasts approximately 6 cycles (1/10th of a second). During a line fault, most sites on the grid experience voltage deviations at different levels. Specifically, sites that are closer to the fault location experience larger deviations than those further away. In addition, there is usually slight time delay from when the PMU closest to the fault location observes the deviation and when a PMU further away observes a similar pattern. While minor, potentially only 1/60th of a second, these time delays can be noticed by machine learning algorithms. Thus, data measured from different sites on the grid during the same event should be categorized by clustering algorithms within the same cluster along with different datapoints from different events of the same class. As discussed earlier in section 1.2, each PMU at every site-event takes more than one measurement. Calculating our features, $RelativePhase2over1$, and $RelativePhase3over1$, from these measurements results in a 2-dimensional datapoint per PMU for each site-event. In order to standardize this process such that slight shifts in time are considered, we select a single instantaneous moment in time across a given site for a given event. Therefore, each element of our dataset

is a 2-dimentional vector of scalar values, as opposed to a vector of vectors of scalar values. We call the data collected by this method instantaneous PMU data. The challenge in this method is to find a way to select the best moment in time for a given site-event. Another observation that makes this method challenging is that measurements are inconsistent in the amount of time they deviate from steady state, even at the same site. To address these challenges we developed a method to extract instantaneous PMU data from our dataset. For our feature set we need three measurements for each PMU per site-event: A Phase Voltage, B Phase Voltage, and C Phase Voltage. We process each of these measurements for each site-event as follows. First, each measurement has a window of time during the event in which it deviates from the steady state. For windows that are less than 8 cycles long, we add steady state data to both sides of the window until the duration is greater or equal to 8. Second, we perform a Piecewise Aggregation Approximation (PAA) [26] with a window size of 8 on each of these measurements, so that the data is divided into 8 slices, or time steps, such that each time step contains approximately the same amount of data. The mean value of the data within each time step is then used to represent that slice of the original window. Third, we add a steady state data point to each side of the new window, forming a time series data window with a fixed size of 10. This window starts and ends with steady state and contains the data recorded for that site-event. PAA is used to make this window have a consistent size as well as condensing larger faults into a smaller, more manageable window. After obtaining the 10 time steps for each measurement we then calculate the index of the 10 cycle window that has the greatest ΔV[1] and use this index as the moment in time for calculating our relative phase features for the given site-event.

[1]Section 3.4 and Equation 3.7.

3.4 Clustering Results

For instantaneous clustering, we choose two widely used methods, K-means [17] and DBSCAN (Density Based Spatial Clustering of Applications with Noise) [15] to perform the clustering on the instantaneous data points extracted from our dataset 1, as described in Section3.3. The two features presented in Section 3.2 are used in the clustering.

The results of the K-means clustering are shown in Figure 3.2. The data points are divided into 4 different groups, each of which is represented by a different color. The percentage breakdown for different event classes are shown in Table 3.1, with the dominant group for each type highlighted.

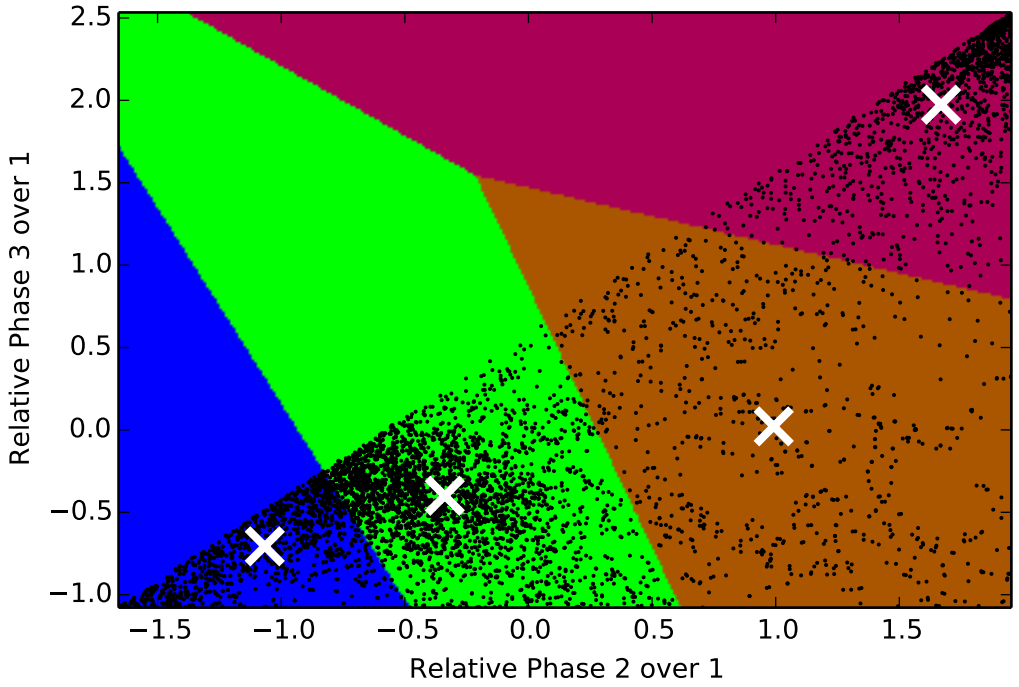


Figure 3.2: K-menas Clustering on Instantaneous Data (centroids are marked with white crosses

In general, K-means clustering performs well in separating different events into groups. Particularly, three-phase (TP) events are well isolated from the rest of the event types.

	LG	LL	TP	NF
Green	**64.74%**	7.85%	0.00%	4.53%
Purple	1.72%	0.60%	**98.98%**	**74.79%**
Brown	3.49%	**91.54%**	1.02%	20.02%
Blue	30.05%	0.00%	0.00%	0.66%

Table 3.1: K-menas Clustering % Breakdown

However, some Line to Ground (LG) events are mixed with Line to Line (LL) events, and the No Fault (NF) data points are separated into two main groups. This is reflected in an overall homogeneity score of 0.60. Note that the homogeneity score [34] is a metric indicating how well data points which belong to the same class are assigned to the same cluster. A perfectly homogeneous solution has a homogeneity score of 1.

We then applied the density-based DBSCAN clustering method on the same dataset, and the results are shown in Figure 3.3 and Table 3.2.

	LG	LL	TP	NF
1	**95.16%**	8.16%	0.00%	5.93%
2	0.16%	8.46%	0.00%	0.74%
3	1.62%	0.60%	**98.98%**	**69.44%**
4	0.04%	15.11%	0.00%	0.41%
5	0.04%	6.65%	0.00%	0.00%
6	0.63%	0.00%	0.00%	1.48%
7	0.55%	3.32%	0.00%	0.66%
8	0.00%	9.67%	0.00%	0.16%
9	0.06%	0.00%	0.00%	0.91%
10	0.00%	4.23%	0.00%	0.00%
11	0.10%	0.00%	0.00%	1.40%
12	0.04%	0.00%	0.00%	1.48%
13	0.00%	5.74%	0.00%	0.00%
14	0.00%	0.00%	0.00%	0.58%
15	0.02%	0.00%	0.00%	0.91%
16	0.10%	0.00%	0.00%	0.33%
Noise	1.48%	**38.07%**	1.02%	15.57%

Table 3.2: DBSCAN clustering % breakdown

Note that when using DBSCAN the user does not provide the number of clusters. Instead,

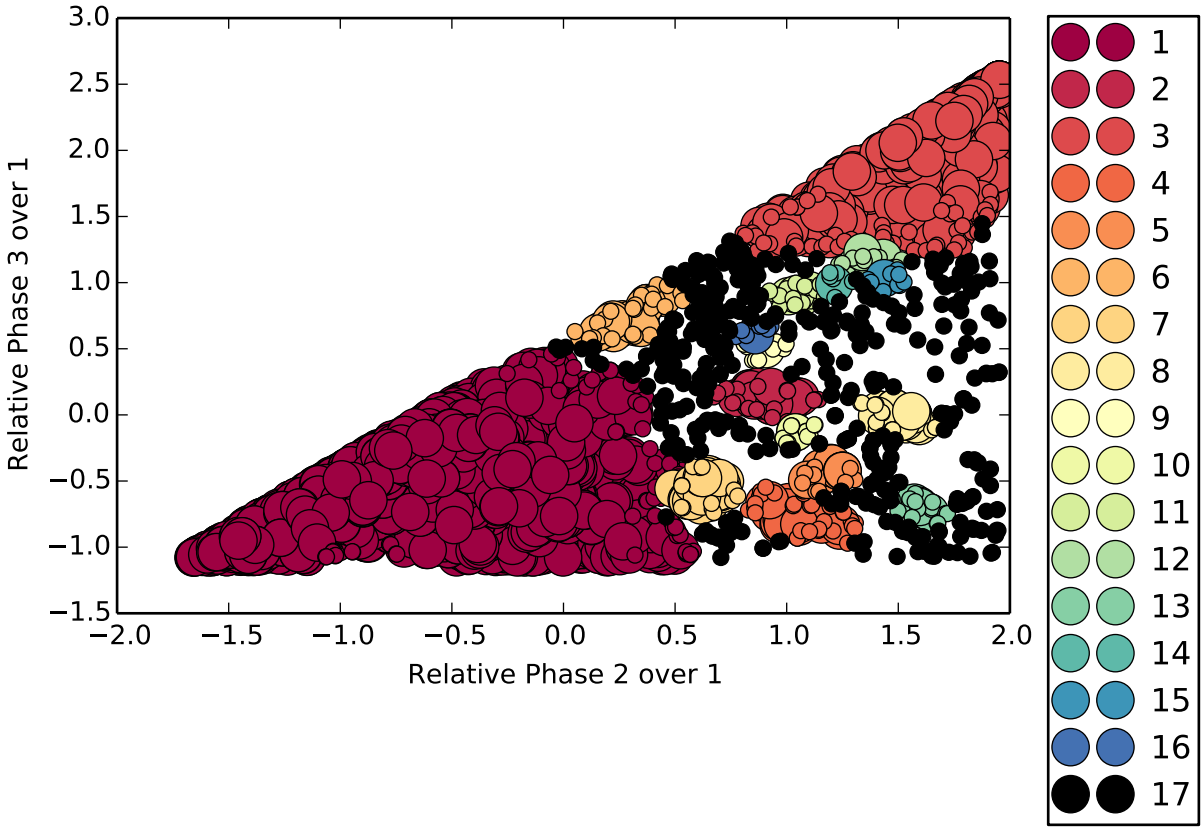


Figure 3.3: DBSCAN Clustering on Instantaneous Data

the algorithm breaks down the data into as many clusters as needed. Alternately, the user provides ϵ and MinSamples. These values determine how dense the data needs to be for a cluster to form and expand. In our case ϵ was empirically set to 0.105 and MinSamples was empirically set to 14. Along with the set of clusters, DBSCAN also adds an extra cluster for what it has determined to be noise in the data. In this case DBSCAN divided the data into 16 groups. Despite the large number of groups, DBSCAN performs well in dividing LG and TP from the other two type of events. However, instead of grouping LL into a single cluster, DBSCAN has divided it into 10 groups with 38% of LL data marked as noise. Most of the No Fault data points are grouped in the same cluster as TP, but a good portion of it is also marked as noise. This clustering has a homogeneity of 0.629.

We observed that there is a considerable amount of data being labeled as noise. If we could reduce the amount of noise, then that could help DBSCAN perform better. In addition, reducing the amount of noise could help separate NF data from TP. Since it has been shown that the combined voltage deviations, as illustrated in equation 3.7, is an effective metric for representing the impact of the event on the PMU site [33], this metric can be used as a filter to remove the noise data. A greater value of ΔV represents higher impact of an event on a PMU. In other words, we can use ΔV in order to remove the events which occur at locations that are far away from the PMUs. To this end, we filtered our dataset by removing all the data entries with ΔV values less than 0.018, a threshold suggested in [33]. After this filtering process, our dataset contains 2211 data points. That is to say, 4462 data points are filtered out, which, generally speaking are signals captured by PMUs which are far away from the location where the event occurred. The clustering results of K-means on this filtered data set are shown in Figure 3.4 and Table 3.3.

$$\Delta V = \frac{\sqrt{(v_a - ss_a)^2 + (v_b - ss_b)^2 + (v_c - ss_c)^2}}{3} \tag{3.7}$$

	LG	LL	TP	NF
Purple	0.05%	**99.19%**	0.00%	0.00%
Blue	37.13%	0.00%	0.00%	0.00%
Green	0.55%	0.81%	**100.00%**	**100.00%**
Brown	**62.26%**	0.00%	0.00%	0.00%

Table 3.3: Filtered K-means Clustering % Breakdown

The results here show some improvements over those on the non-filtered data, although LG is still split between two clusters. LL is predominately grouped in one cluster. Interestingly, all of TP events are clustered is the same group as all NF data points. After the ΔV filtering, only a small number of NF data points are left in the dataset. The homogeneity score of this method is 0.932, much higher than the results on the non-filtered dataset.

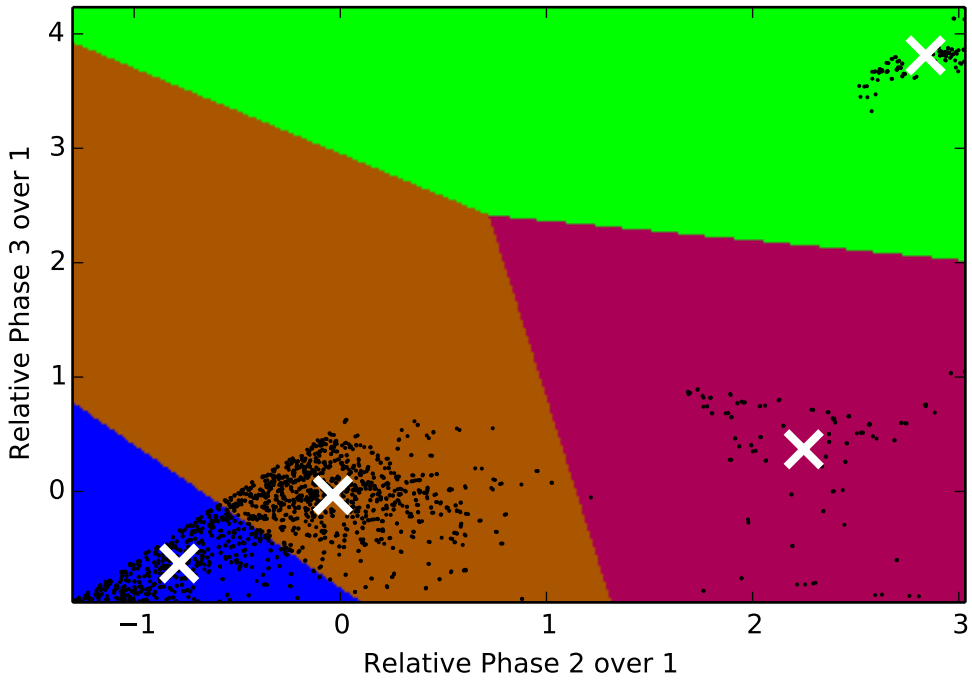


Figure 3.4: K-means Clustering on Filtered Instantaneous Data (centroids are marked with white crosses

On the same filtered dataset, we have also applied the DBSCAN clustering method. The results are shown in Figure 3.5 and Table 3.4. For this clustering ϵ was set to 0.7 and MinSamples was set to 3.

	LG	LL	TP	NF
1	0.00%	**99.19%**	0.00%	0.00%
2	**99.45%**	0.00%	0.00%	0.00%
3	0.55%	0.81%	**100.00%**	**100.00%**

Table 3.4: Filtered DBSCAN clustering % breakdown

Comparing to the results on the original dataset, this set of results shows significant improvement, because the filtering helped remove most of the noise. With the filtering process, DBSCAN performed excellently, illustrated by its homogeneity score of 0.939. It was able to cluster most of LG in cluster 2, LL in cluster 1, and all of TP in cluster 3. Similar to the K-means method, NF data are mixed with TP in cluster 3, because of the limited

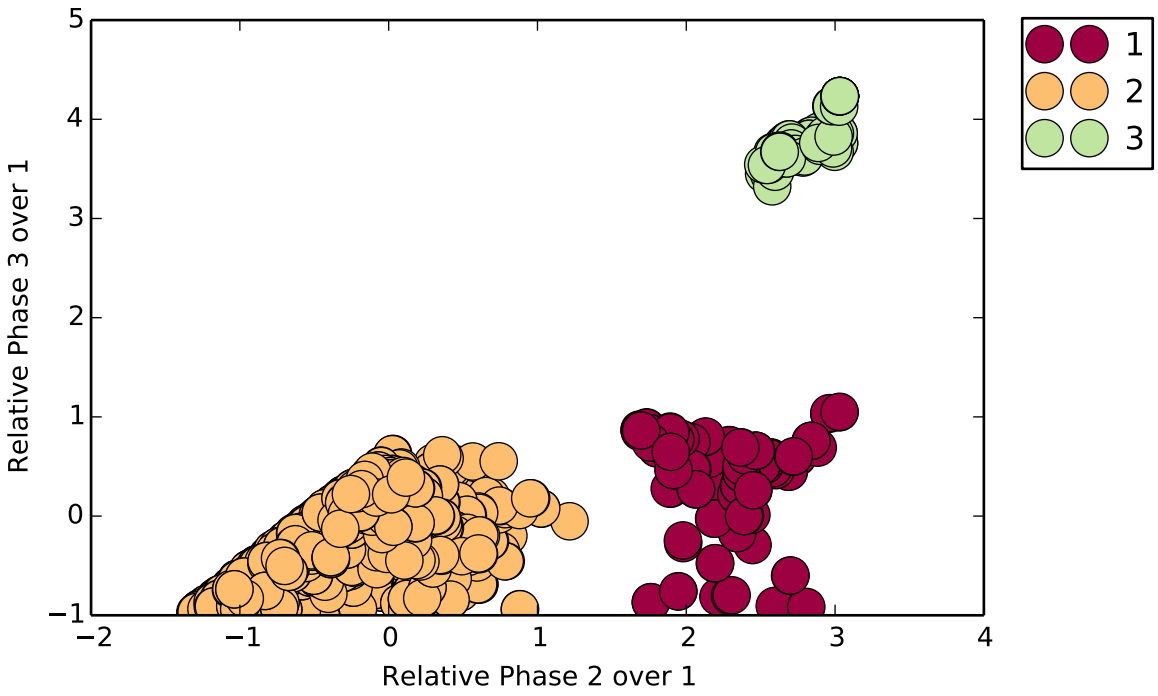


Figure 3.5: DBSCAN Clustering on Filtered Instantaneous Data

number of data samples after the filtering.

3.5 Concluding Thoughts on Case Study 1

What this case study shows is that under a good feature set and some data filtering we can achieve high homogeneity with both clustering methods we chose. For the most part we can separate our data into three of the four categories. However, we are not able to cluster NF data into its own cluster. In this case it ends up behaving most like TP data.

Chapter 4

Case Study 2: Hierarchical Clustering Over Time Series Data

Fault events usually last over a period of time. In Chapter 3 we used a single moment in time of this window on which we applied the clustering methods. The disadvantage of this approach is that it ignores the other moments in time that also experience the fault. In this chapter, we explore the potentials of using the entire event window for clustering instead of just using instantaneous data points. We call this method Time Series Clustering. The dataset used for this case study is the same as in Section 3.3 except for the minor change of using the entire 10-cycle window instead of the single time step. We also use the same features as described in Section 3.2.

4.1 Similarity Metric

Under the instantaneous clustering method we used euclidean distance as our similarity metric for computing the similarity, or distance, between two data points. However, this metric is not suitable for time series data. In Chapter 3, each data point is a 2-dimensional

point, one dimension per feature. In contrast, each time series datapoint is a 10×2 matrix, in which each row is a single time step in our 10-time-step window and each column is a feature. This paradigm shift requires a similarity metric that considers each time step as well as each feature in its calculations. The simplest approach would be to sum the distances between each row (time step) in the given matrices. However, this does not take into consideration possible offsets in the time series. Dynamic Time Warping (DTW) [26] is a better approach as it accounts for slight offsets between two rows and makes two time series data points better aligned. For this reason we have chosen DTW as our similarity metric for time series clustering. DTW is a dynamic programing method that works by computing the distances between all corresponding multidimensional time steps of one data point to another in addition to also computing the distance of the corresponding time step's neighbors. Of these distances it adds the smallest into a running sum as it moves down both time series. The resulting sum is the DTW distance between the two time series. Figure 4.1 is an example of what this process looks like. The dotted lines show which pairs of time steps were selected by the DTW algorithm for the distance betweeen them to be added to the total sum.

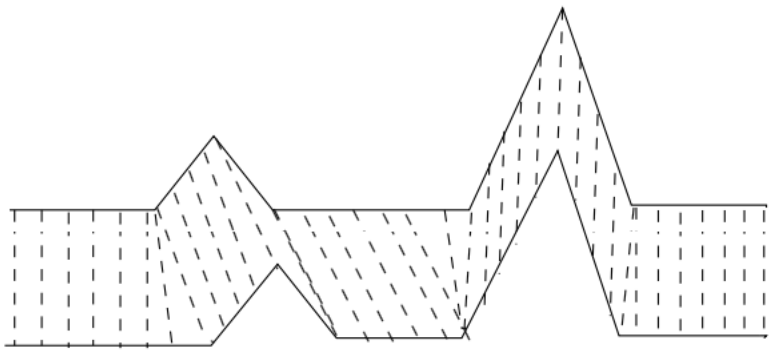


Figure 4.1: Dynamic Time Warping Example

4.2 Clustering Results

For clustering, we chose the hierarchical clustering algorithm [2], because it is the most suitable method for the time series model, and it allows customized distance metric, for which we used the DTW distance metric described above.

The clustering results for this method are shown in Figure 4.2. Different colors represent different groups of data entries. Since there are 10 time steps and 2 features the whole plot would be 20-dimensional, to accommodate for this high dimensionality, we have only plotted time step 5. Time step 5 was chosen because it is the center of the time series and therefore, it is the moment in time with the largest deviation from steady state. To associate the groups generated by the clustering method with the event types in our dataset, we have calculated the percentage breakdown of each event type in the 4 groups, and the results are shown in Table 4.1.

	LG	LL	TP	NF
0	14.08%	9.37%	**44.39%**	28.25%
1	14.23%	29.00%	11.23%	29.41%
2	18.70%	**38.37%**	35.71%	**40.94%**
3	**52.99%**	23.26%	8.67%	1.40%

Table 4.1: Time Series Hierarchical Clustering % Breakdown

As shown in the results, more than half of the LG events are clustered in group 3 while the rest are split among the other three groups. The majority of LL events are split among three groups 1, 2, and 3. The majority of the TP events are split among groups 0 and 2. Finally, most of the NF data entries are split between groups 0, 1, and 2. Overall, hierarchical clustering on PMU time series data does not work well, which is reflected by the low homogeneity score of 0.156.

For completeness, we have also carried out the hierarchical clustering on the filtered dataset, and the results are shown in Figure 4.3 and Table 4.2.

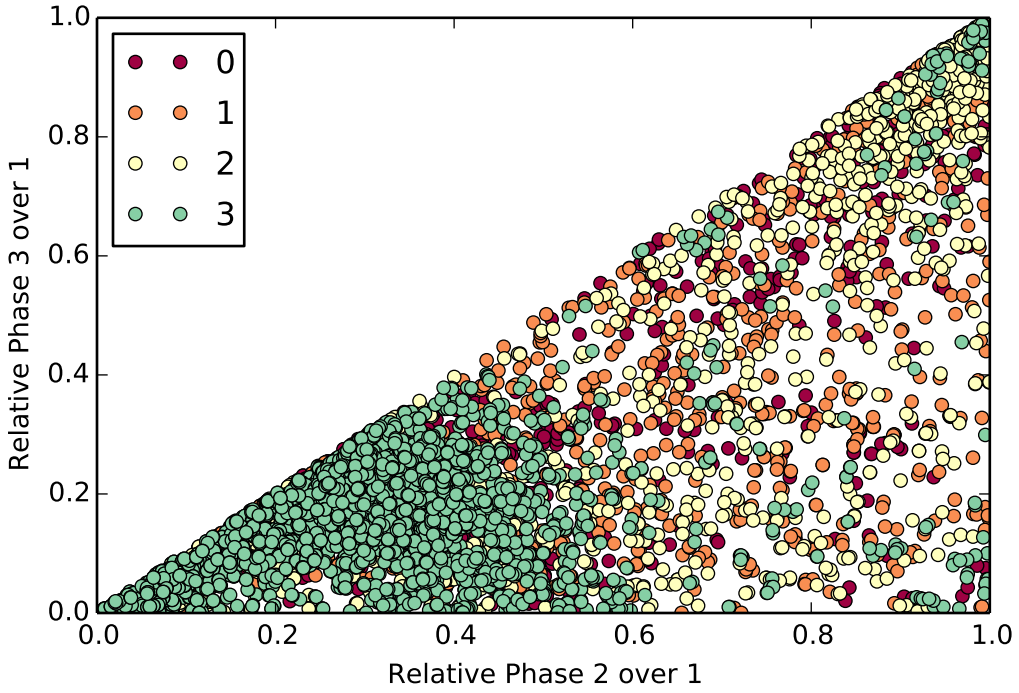


Figure 4.2: Hierachical Clustering at Time Step 5

The results of the hierarchical clustering on filtered dataset are similar to those on the original dataset, with an even lower homogeneity score of 0.027.

4.3 Concluding Thoughts on Case Study 2

Under the same conditions as instantaneous clustering, time series clustering found clusters that did not correspond to our desired taxonomy. This can be seen by its low homogeneity scores. This indicates that although PMU data are time series in nature, it is challenging to apply clustering methods on these time series, using the relative phase feature we developed. A potential future direction is to develop new features that are more representative of time series data, and we will explore this in our future work.

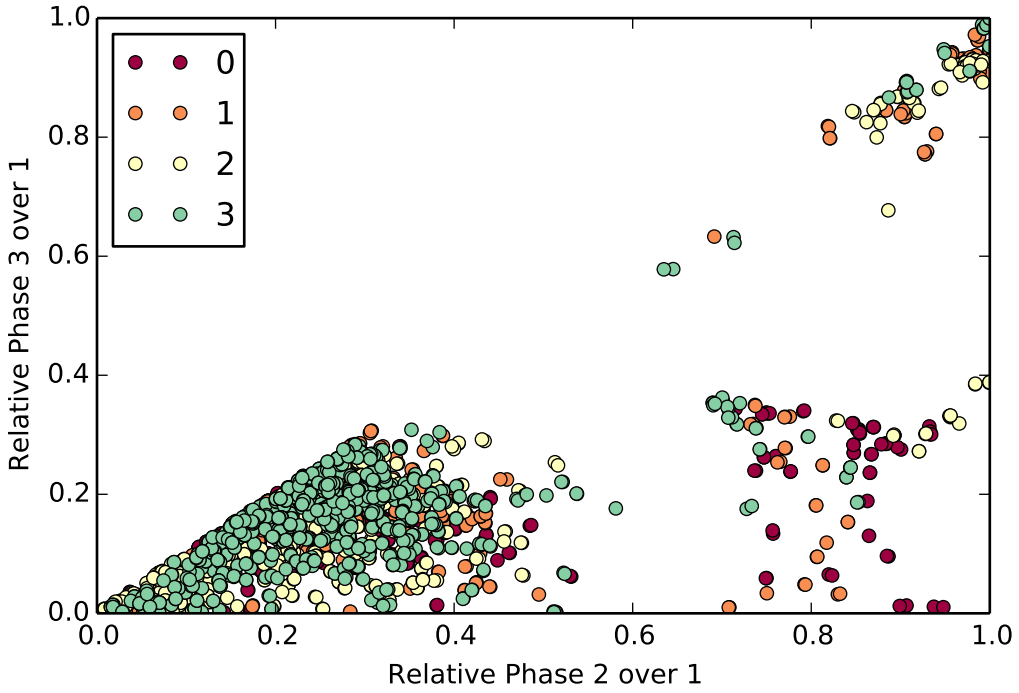


Figure 4.3: Hierarchical Clustering on Filtered Data at Time Step 5

	LG	LL	TP	NF
0	15.24%	**44.36%**	0.00%	0.00%
1	19.52%	21.77%	31.43%	0.00%
2	27.80%	15.32%	**58.10%**	33.33%
3	**37.44%**	18.55%	10.47%	**66.67%**

Table 4.2: Time Series with ΔV filter Hierarchical Clustering % Breakdown

Chapter 5

Case Study 3: Classification on Clustering Results

The previous 2 case studies demonstrate the effectiveness of clustering to categorize PMU data on the smart gird into groups based on a known taxonomy. The case study presented in this chapter uses those results to further investigate other machine learning techniques. Specifically, this case study focuses on using clustering as a prepossessing step to generate training data for a set of cluster-specific Support Vector Machines (SVMs). SVMs are a supervised machine learning model that learns a set of vectors within the feature space that separate different class of data.

5.1 Training and Testing Datasets

Within each of the previous 2 case studies we presented both non-filtered and filtered clustering results. Each of these four results has been used to train and test cluster-specific SVMs. Specifically, experiment 1 uses the non filtered results from case study 1, experiment 2 uses the filtered results from case study 1, experiment 3 uses the non filtered results from case

study 2, and experiment 4 uses the filtered results from case study 2. For each experiment we created a number of SVMs (one per cluster), and trained each SVM using the data residing in its corresponding cluster. Note that in order to obtain sufficient new testing data, in this case study, we used dataset 2 described in section 1.2 as the testing dataset. Within this testing set there are 26 line events. Some signals are not available for every event. On average each event is recorded at about 217 different PMUs on the power grid. In total this test set has 5631 data points, in which 3243 are LG faults, 1956 are LL faults, and 432 are TP faults. There are 0 NF data points in this dataset. For each experiment we collected data from this dataset using the same methods as the training set. For experiments 2 and 4 we also applied the same filter to the test dataset that was applied to the training set prior to clustering. This filtered testing dataset contains 888 data points. 762 are LG faults, 51 are LL, 75 are TP, and 0 are NF. In order for us to test the SVMs, we must split the test set into clusters, so that each SVM could be tested on data that belongs to the same cluster it was trained on. To this end, a nearest neighbor approach is used to determine which cluster a given point belongs to. This is a reasonable approach, because each datapoint in the testing set is very likely to be clustered with the datapoint in the training set that it is closest to. Note that in all four of these experiments a linear kernel was used for all SVMs. We chose not to use parameter fitting methods to avoid potential bias on our results.

5.2 Experiment 1 Results

This experiment trained 17 SVMs on the non-filtered dataset from case study 1 and then tested those SVMs on the second dataset detailed in Section 1.2 and further broken down in Section 5.1. The non-filtered dataset from case study 1 is made up of 17 clusters, however, 3 clusters could not be used for testing because the testing set does not contain data that belongs to those clusters. Specifically, we are not able to test clusters 12, 14, and 16. Each

SVM we tested has its own confusion matrix. These matrices are shown in Table 5.1. In addition, the accuracy of all 14 SVMs are given in Table 5.2. We also created a 15th SVM that was trained and tested on the entire training and testing sets respectively. This SVM's confusion matrix is the last one given in Table 5.1 and is listed as "All" in Table 5.2.

As shown in Table 5.2, most SVMs do not perform well in terms of accuracy. There are a few that do well, resulting in a total accuracy of 69%. The All SVM also gets an accuracy of 69%. However, the All SVM is trained on all the data, while the cluster-specific SVMs are only trained on a fraction of the dataset (one cluter). It should be noted that some of the most accurate SVMs, specifically 4, 5, and 10, where trained on a relatively small portion of the dataset. This shows that in some cases clustering does provide latent information that may be useful to training SVMs on smaller datasets.

For comparison purposes we also used a pseudo random number generator to randomly split our training and testing data into 17 clusters. We then trained and tested 17 new SVMs on these randomly clustered datapoints. Confusion matrices for these SVMs are shown in Table 5.3. The accuracy breakdown is also given in Table 5.4. Overall, the accuracy of these SVMs trained and tested on randomly clustered datapoints is 56%, 13% less than the clustered dataset. In addition, the randomly clustered data was much better distributed and some SVMs in this method were given more datapoints to train on, yet cluster-specific classifiers achieved similar accuracy. This shows that cluster-specific classifiers can perform better with less training data. However, there are still quite a few clusters that do poorly under similar, more in some cases, amounts of training data. We believe that this demonstrates that the clustering itself is doing most of the work and that the SVMs in most cases are just classifying test points as the same class as the majority class of the training data. This is further supported by the confusion matrices in which we see that all but SVM 17 and All, predicted the entire training set under a single class. This is interesting, because it informs us that the SVMs that do well demonstrate that data within these clusters in our

feature space are very good indicators of a particular class.

5.3 Experiment 2 Results

Experiment 2 uses filtered data from case study 1 to train 3 SVMs. Confusion matrices for this experiment are shown in Table 5.5, and the accuracy for all three SVMs is shown in Table 5.6. As in the previous experiment, an additional SVM was trained and tested on the entire filtered dataset. This result is also listed in Table 5.5 and 5.6 under "All".

This experiment is much more successfully at classification than experiment 1, however, this is due to the fact that SVM 2 was trained on a cluster containing a single classification. Therefore, all testing data that belongs to cluster 2 are classified as that class. The high accuracy shows that cluster 2 is very good in identifying that classes, however SVM 1 does very poorly with an accuracy of 40%. While this method gets a generally high accuracy of 91% it may not do well under different circumstances. Again the accuracy of the single SVM over all the data gets exactly the same accuracy of 91%, similar to experiment 1. In addition, SVM1 has a similar amount of training data to that of SVM 3, and yet there is a 53% difference in accuracy. Therefore, we can conclude that the clustering is able to identify good points for training and testing for SVM 3 while it is not optimal for SVM1.

Again, we have also trained and tested an additional 3 SVMs for this experiment on data that has been randomly clustered. Confusion matrices are given in Table 5.7 and a summary is shown in Table 5.8. These SVMs overall perform similarly to cluster-specific SVMs. Under this method SVM 1's accuracy is improved from 40% to 91% while the cluster-specific method has better accuracy for SVM's 2 and 3. This shows that cluster 1 requires more divers training data in order for an SVM to be an effective classifier while in the case of clusters 2 and 3 clustering can be used to increase classification accuracy, even under fewer training samples which is the case with SVM 3. However again, we can see that from the

confusion matrices that all three SVMs classify the entire testing set as a given class. This continues to support the idea that the SVMs are not doing any actual work and instead the clustering is achieving a high accuracy.

5.4 Experiment 3 Results

The previous two experiments, while vaguely promising, do not perform well. We speculate that the major issue is a lack of diversity in the training sets. DBSCAN does well to split the data into homogeneous clusters, but because the clusters are so homogeneous there is a lack of data from other classes for the SVMs to train on. Therefore, when testing data is introduced that contains data that a given SVM has trained very little on it is not surprising to see those data points being misclassified. In contrast, the time-series clusters are not very homogeneous, this implies that they may make better training sets because of the diversity of classes within the clusters. In addition, the clustering while not homogeneous may have discovered a relationship in the similarity of the data it clustered. Therefore, in the next two experiments we explore if training on time-series data clusters can improve classification. This third experiment specifically trains on the results from the non-filtered clustering of case study 2. These SVMs are then tested on the same testing dataset as the first experiment only this time the entire 10-cycle window is used to determine what cluster the data point is closest to. We used DTW as our similarity metric when categorizing which cluster each testing data point belongs to. SVMs expect data to be organized as a single vector of scalar values. To accommodate this trait we used cycle 5 of the 10-cycle window for training and testing because cycle 5 represents the middle of the window which is the point at which the deviations should be at their maximum. Confusion matrices for this experiment are given in Table 5.9, and the accuracy breakdown is shown in Table 5.10.

Although the clustering for this experiment has a low homogeneous score, SVMs ul-

timately perform much better with a total accuracy of 62%. This shows that while the clustering, from a homogeneous stand point, is not optimal, the performance can be greatly improved using a cluster-specific classifier. However, this result is very similar to the single SVM with an accuracy of 63%. Cluster 4 performs well with an accuracy of 72% even though it was trained on a fraction of the data.

In addition, four SVMs were trained and tested on randomly clustered data. Confusion matrices are given in Table 5.11 and the accuracy summaries are given in Table 5.12. Overall, cluster-specific classifiers are more accurate by 4%. The amount of training data is also very similar. This is due to time series clustering behaving more randomly and therefore performing similarly to that of random clustering. Unlike the previous two experiments the confusion matrices of this experiment do classify testing data as multiple classes. This shows that the SVMs are doing some additional work. This is expected as the training data provided to these SVMs are not dominated by a single class.

5.5 Experiment 4 Results

In this last experiment we used the filtered clustering results from case study 2 to train five SVMs. These SVMs were tested by filtering the test set from experiment 3. The confusion matrices are shown in Table 5.13, and the overview table is given in Table 5.14.

This result is actually the most promising one as it has a total accuracy of 88%. This is lower than that of experiment 2, however, overall it is superior because it does not rely on SVMs trained exclusively on data from one class, therefore, we believe that it can be applied to a broader range of data. The other promising result is that there is no single cluster with a low accuracy, unlike experiment 2 which had a SVM with 40% accuracy. The SVM trained over all the data has exactly the same accuracy of 88%.

Finally, just as in previous experiments, we explore training on random cluster SVMs.

The confusion matrices of these SVMs are shown in Table 5.15 and the summary breakdown is shown in Table 5.16. The randomized clustering method was very similar to the non-random approach. Overall it is only a 1% improvement making it practically the same. This result is very similar to experiment 2 in which the total accuracy of both the random clustering and non random clustering methods resulted in nearly the same accuracy. Again, time series clustering results in a similar amount of training data in comparison to random clustering. In addition, a majority of the SVMs do classify test data into multiple classes. Therefore, the SVMs do additional work in classifying the testing dataset, yet they perform no better than random clustering, especially when considering the amount of data used in training.

5.6 Concluding thoughts on Case Study 3

In all cases the single SVM performed just as well if not better than the cluster-specific classifiers. In experiments 1 and 2 clustering performs just as well as the total accuracy of all SVMs including the single SVM. However, most SVMs classify the testing data into a single class showing that the clustering itself is doing most of the work to classify the data. In some cases these SVMs achieve a very high accuracy. This demonstrates that these regions of the feature space are very good at identify a given classification. On the other hand, experiments 3 and 4 outperform time series clustering. In both experiment 1 and 3, cluster-specific classification outperformed random cluster classification, while experiments 2 and 4 performed similarly to random clustering. Instantaneous clustering is more adapt at discovering specific clusters that provide high accuracy based on the clustering alone while other SVMs score poorly due to a lack of diversity in the data. On the other hand, time series clustering does provide a more diverse dataset for training, but it does not perform much better overall than the random clustering method. This leads us to conclude that

cluster-specific classifiers are not an improvement over a single SVM trained on the entire dataset, in terms of accuracy, although some cluster-based SVMs require significantly less training data to achieve similar results and filtered data does not perform any better than random clustering. In addition, we conclude that, in the case of instantaneous clustering, SVMs trained on the clusters, in general, do not classify data into more than one class, leading us to believe that the clustering itself is achieving the accuracy of the classification.

Experiment 1 SVM 1

Actual Class		Predicted Class		
		LG	LL	TP
	LG	2577	0	0
	LL	348	0	0
	TP	3	0	0

Experiment 1 SVM 2

Actual Class		Predicted Class	
		LG	LL
	LG	0	12
	LL	0	24

Experiment 1 SVM 3

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	0	0	0	84
	LL	0	0	0	21
	TP	0	0	0	426
	NF	0	0	0	0

Experiment 1 SVM 4

Actual Class		Predicted Class	
		LG	LL
	LG	0	24
	LL	0	561

Experiment 1 SVM 5

Actual Class		Predicted Class
		LL
	LL	90

Experiment 1 SVM 6

Actual Class		Predicted Class	
		LG	LL
	LG	12	0
	LL	3	0

Experiment 1 SVM 7

Actual Class		Predicted Class	
		LG	LL
	LG	9	0
	LL	255	0

Experiment 1 SVM 8

Actual Class		Predicted Class	
		LG	LL
	LG	0	15
	LL	0	0

Experiment 1 SVM 9

Actual Class		Predicted Class	
		LG	NF
	LG	0	6
	LL	0	6
	NF	0	0

Experiment 1 SVM 10

Actual Class		Predicted Class
		LL
	LL	6

Experiment 1 SVM 11

Actual Class		Predicted Class	
		LL	NF
	LL	0	3
	NF	0	0

Experiment 1 SVM 13

Actual Class		Predicted Class	
		LG	LL
	LG	0	12
	LL	0	42

Experiment 1 SVM 15

Actual Class		Predicted Class	
		LG	NF
	LG	0	6
	NF	0	0

Experiment 1 SVM 17

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	0	111	0	375
	LL	3	543	0	51
	TP	0	0	0	3
	NF	0	0	0	0

Experiment 1 SVM All Data

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	2622	264	0	357
	LL	618	1275	0	63
	TP	6	0	0	426
	NF	0	0	0	0

Table 5.1: Experiment 1 Confusion Matrices

SVM Number	Accuracy	# Training Points	# Testing Points
1	0.88	4795	2928
2	0.67	45	36
3	0.00	1119	531
4	**0.96**	57	585
5	**1.00**	24	90
6	0.80	49	15
7	0.03	46	264
8	0.00	34	15
9	0.00	14	12
10	**1.00**	14	6
11	0.00	22	3
13	0.78	19	54
15	0.00	12	6
17	0.50	390	1086
Total	0.69	6640	5631
All	0.69	6676	5631

Table 5.2: Experiment 1 SVM Summary

Experiment 1 Random Cluster SVM 1

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	163	2	0	22
	LL	90	29	0	7
	TP	0	0	0	19
	NF	0	0	0	0

Experiment 1 Random Cluster SVM 2

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	160	7	0	29
	LL	79	28	0	4
	TP	0	0	0	25
	NF	0	0	0	0

Experiment 1 Random Cluster SVM 3

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	180	0	0	28
	LL	89	5	0	2
	TP	0	0	0	28
	NF	0	0	0	0

Experiment 1 Random Cluster SVM 4

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	161	1	0	32
	LL	93	8	0	11
	TP	0	0	0	26
	NF	0	0	0	0

Experiment 1 Random Cluster SVM 5

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	155	10	0	24
	LL	95	30	0	0
	TP	0	0	0	17
	NF	0	0	0	0

Experiment 1 Random Cluster SVM 6

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	166	6	0	30
	LL	74	27	0	4
	TP	0	0	0	24
	NF	0	0	0	0

Experiment 1 Random Cluster SVM 7

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	138	12	0	21
	LL	46	84	0	1
	TP	1	0	0	28
	NF	0	0	0	0

Experiment 1 Random Cluster SVM 8

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	145	9	0	30
	LL	57	61	0	6
	TP	0	0	0	23
	NF	0	0	0	0

Experiment 1 Random Cluster SVM 9

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	163	2	0	26
	LL	87	25	0	5
	TP	1	0	0	22
	NF	0	0	0	0

Experiment 1 Random Cluster SVM 10

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	162	0	0	33
	LL	89	5	0	17
	TP	0	0	0	30
	NF	0	0	0	0

Experiment 1 Random Cluster SVM 11

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	153	9	0	28
	LL	73	43	0	3
	TP	1	0	0	21
	NF	0	0	0	0

Experiment 1 Random Cluster SVM 12

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	162	1	0	35
	LL	92	18	0	2
	TP	1	0	0	20
	NF	0	0	0	0

Experiment 1 Random Cluster SVM 13

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	146	6	0	26
	LL	105	15	0	4
	TP	0	0	0	29
	NF	0	0	0	0

Experiment 1 Random Cluster SVM 14

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	175	3	0	18
	LL	83	20	0	4
	TP	0	0	0	28
	NF	0	0	0	0

Experiment 1 Random Cluster SVM 15

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	161	7	0	21
	LL	81	28	0	2
	TP	1	0	0	30
	NF	0	0	0	0

Experiment 1 Random Cluster SVM 16

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	163	2	0	26
	LL	91	17	0	4
	TP	1	0	0	27
	NF	0	0	0	0

Experiment 1 Random Cluster SVM 17

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	149	0	0	35
	LL	116	0	0	2
	TP	0	0	0	29
	NF	0	0	0	0

Table 5.3: Experiment 1 Random Cluster Confusion Matrices

SVM Number	Accuracy	# Training Points	# Testing Points
1	0.58	393	332
2	0.57	393	332
3	0.56	393	332
4	0.51	393	332
5	0.56	393	331
6	0.58	393	331
7	0.67	393	331
8	0.62	393	331
9	0.57	393	331
10	0.49	393	331
11	0.59	393	331
12	0.54	393	331
13	0.54	392	331
14	0.59	392	331
15	0.57	392	331
16	0.54	392	331
17	0.45	392	331
Total	0.56	6676	5631

Table 5.4: Experiment 1 Random Cluster SVM Summary

Experiment 2 SVM 1

Actual Class		**Predicted Class**	
		LG	LL
	LG	0	78
	LL	0	51

Experiment 2 SVM 2

Actual Class		**Predicted Class**
		LG
	LG	678

Experiment 2 SVM 3

Actual Class		**Predicted Class**	
		LG	TP
	LG	0	6
	TP	0	75

Experiment 2 SVM All Data

Actual Class		**Predicted Class**		
		LG	LL	TP
	LG	678	78	6
	LL	0	51	0
	TP	0	0	75

Table 5.5: Experiment 2 Confusion Matrices

SVM Number	Accuracy	# Training Points	# Testing Points
1	0.40	123	129
2	**1.00**	1971	678
3	**0.93**	120	81
Total	**0.91**	2214	888
All	**0.91**	2214	888

Table 5.6: Experiment 2 SVM Summary

Experiment 2 Random Cluster SVM 1

Actual Class		Predicted Class		
		LG	LL	TP
	LG	224	24	2
	LL	0	21	0
	TP	0	0	25

Experiment 2 Random Cluster SVM 2

Actual Class		Predicted Class		
		LG	LL	TP
	LG	226	29	2
	LL	0	13	0
	TP	0	0	26

Experiment 2 Random Cluster SVM 3

Actual Class		Predicted Class		
		LG	LL	TP
	LG	228	25	2
	LL	0	17	0
	TP	0	0	24

Table 5.7: Experiment 2 Random Cluster Confusion Matrices

SVM Number	Accuracy	# Training Points	# Testing Points
1	**0.91**	738	296
2	**0.90**	738	296
3	**0.91**	738	296
Total	**0.91**	2214	888

Table 5.8: Experiment 2 Random Cluster SVM Summary

Experiment 3 SVM 1

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	372	0	0	120
	LL	171	0	0	0
	TP	3	0	0	111
	NF	0	0	0	0

Experiment 3 SVM 2

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	450	15	0	171
	LL	255	276	0	33
	TP	0	0	0	93
	NF	0	0	0	0

Experiment 3 SVM 3

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	513	63	0	108
	LL	204	288	0	12
	TP	18	0	0	165
	NF	0	0	0	0

Experiment 3 SVM 4

Actual Class		Predicted Class		
		LG	LL	TP
	LG	1410	21	0
	LL	540	177	0
	TP	42	0	0

Experiment 3 SVM All Data

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	2673	111	0	459
	LL	1023	906	0	27
	TP	18	3	0	411
	NF	0	0	0	0

Table 5.9: Experiment 3 Confusion Matrices

SVM Number	Accuracy	# Training Points	# Testing Points
1	0.48	1156	777
2	0.56	1177	1293
3	0.58	1617	1371
4	0.72	2726	2190
Total	0.62	6676	5631
All	0.63	6676	5631

Table 5.10: Experiment 3 SVM Summary

Experiment 3 Random Cluster SVM 1

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	657	33	0	100
	LL	311	194	0	4
	TP	3	0	0	106
	NF	0	0	0	0

Experiment 3 Random Cluster SVM 2

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	710	4	0	101
	LL	402	87	0	1
	TP	6	0	0	97
	NF	0	0	0	0

Experiment 3 Random Cluster SVM 3

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	669	30	0	121
	LL	219	251	0	13
	TP	3	1	0	101
	NF	0	0	0	0

Experiment 3 Random Cluster SVM 4

Actual Class		Predicted Class			
		LG	LL	TP	NF
	LG	687	2	0	129
	LL	421	16	0	37
	TP	7	0	0	108
	NF	0	0	0	0

Table 5.11: Experiment 3 Random Cluster Confusion Matrices

SVM Number	Accuracy	# Training Points	# Testing Points
1	0.60	1669	1408
2	0.57	1669	1408
3	0.65	1669	1408
4	0.50	1669	1407
Total	0.58	6676	5631

Table 5.12: Experiment 3 Random Cluster SVM Summary

Experiment 4 SVM 1

Actual Class		Predicted Class	
		LG	LL
	LG	72	36
	LL	0	21

Experiment 4 SVM 2

Actual Class		Predicted Class		
		LG	LL	TP
	LG	135	30	6
	LL	0	30	0
	TP	0	0	36

Experiment 4 SVM 3

Actual Class		Predicted Class		
		LG	LL	TP
	LG	153	24	0
	LL	0	0	0
	TP	0	0	30

Experiment 4 SVM 4

Actual Class		Predicted Class	
		LG	TP
	LG	306	0
	TP	9	0

Experiment 4 SVM All Data

Actual Class		Predicted Class		
		LG	LL	TP
	LG	651	96	15
	LL	0	51	0
	TP	0	0	75

Table 5.13: Experiment 4 Confusion Matrices

SVM Number	Accuracy	# Training Points	# Testing Points
1	0.72	357	129
2	0.85	447	237
3	0.88	632	207
4	**0.97**	778	315
Total	0.88	2214	888
All	0.88	2214	888

Table 5.14: Experiment 4 SVM Summary

Experiment 4 Random Cluster SVM 1

Actual Class		Predicted Class		
		LG	LL	TP
	LG	156	26	5
	LL	0	13	0
	TP	0	0	22

Experiment 4 Random Cluster SVM 2

Actual Class		Predicted Class		
		LG	LL	TP
	LG	169	28	4
	LL	0	7	0
	TP	0	0	14

Experiment 4 Random Cluster SVM 3

Actual Class		Predicted Class		
		LG	LL	TP
	LG	164	24	1
	LL	0	12	0
	TP	0	0	21

Experiment 4 Random Cluster SVM 4

Actual Class		Predicted Class		
		LG	LL	TP
	LG	174	6	5
	LL	0	17	0
	TP	0	0	18

Table 5.15: Experiment 4 Random Cluster Confusion Matrices

SVM Number	Accuracy	# Training Points	# Testing Points
1	0.86	554	222
2	0.86	554	222
3	0.89	553	222
4	**0.94**	553	222
Total	0.89	2214	888

Table 5.16: Experiment 4 Random Cluster SVM Summary

Chapter 6

Case Study 4: Discovering Generator Faults

This last case study explores features and filtering techniques for clustering generator faults. These faults arise on the power grid under different circumstances than line faults. Using what we learned from the previous case studies we attempt to apply clustering methods to classify and characterize our dataset into three groups: line events, generator events, and no fault events. We contribute a novel set of features and filtering methods that when applied with the DBSCAN clustering algorithm produce a homogeneity score of 0.960.

6.1 Generator Dataset Specifics

In previous case studies we presented methods for categorizing and characterizing of line faults. We specifically left out generator faults from our previous datasets for the express purpose of focusing our efforts on clustering line events into the 4 main groups dictated by a known taxonomy of line faults. Therefore, a new dataset that includes generator faults was necessary for this case study. To be comprehensive, this dataset must contain the 3

line fualt types discused in previous sections. To accomplish this we used the line faults found in the first dataset described in Section 1.2. No fault events are randomly chosen, therefore, we could not reuse the previous dataset in its entirety, but instead chose new random no fault points for this dataset. We then identified, with the help of BPA, generator faults in the data and included them in this new dataset. These generator faults take place between June of 2014 and October of 2014. In total this new dataset contains 161 events and each event was recorded by approximately 90 PMUs on the smart grid (not all PMUs are present for each event). Among these events, 72 are line faults (LF), 35 are generator faults (GF), and the remaining 26 are no fault events (NF). In total this dataset contains 14548 data points such that each data point was measured at a single point on the grid during a particular event. Specifically, there are 5221 LF data points, 5463 GF data points, and 1774 NF data points. Previously, in order to accommodate time series clustering, as well as use a consistent window size, we used a piecewise aggregation approximation (PAA) on every data point to condense the data window into a consistent size of 8 and then calculated our features on that window. This was an appropriate approach for our two relative phase features as these features only rely on capturing the middle of this window and thus using a PAA did not modify the features significantly. However, generator faults require further study of additional features, some of which rely on the window size. For these reasons we chose not to use a PAA as it would limit us to only use features that have similar properties to that of RelativePhase2over1 and RelativePhase3over1.

6.2 Feature Selection

Our previous two features developed for line faults worked well under instantaneous DBSCAN clustering. However, it was unknown weather they would be suitable for generator faults. With the help of BPA we were able to explore features that can be potentially useful

for characterizing generator faults. In total, we synthesized 19 features from our dataset. Our method for selecting a reasonable subset of features from this set was empirical in nature. The selection process can be broken down into two steps. First, we constructed boxplots of each feature to determine how well individually a given feature separates our three categories of our data. Figure 6.1[1] shows two of these boxplots. Both of these boxplots are good examples of features that have potentials for characterizing our dataset, because there is a clear separation at around a RelativePhase2over1 of 0.6 at which the majority of line faults exist below and the majority of generator faults exist above. Similarly, all line faults and no fault entries have an event length less than 200 cycles, while generator faults usually last for more than 200 cycles. Secondly, we compared all features by plotting all combinations of features to identify which features when combined provide better data separation. Figure 6.2[1] is an example of one of these comparison plots, it further demonstrates that these two features, among many others, are good candidates for data categorizations of generator events, as most of the generator events are separate from the line faults. Using this method we were able to identify 10 features that best aid characterizing the differences between line faults, generator faults, and no fault data. Table 6.1 breaks down each of these features into further details. Relative Phase calculations are described in Section 3.2 and detailed in Equations 3.1 through 3.6. ΔV is described in Section 3.4 and detailed in Equation 3.7. In Table 6.1 v values are voltage magnitudes, ss values are steady state magnitudes, and f values are frequency magnitudes.

6.3 Filtering

To further assist characterization and categorization we developed filters as a preprocessing step to clustering. In Section 3.4 a ΔV filter was applied to achieve greater homogeneity. In

[1]Both Figure 6.1 and Figure 6.2 are plotted under the filter described in Section 6.3.

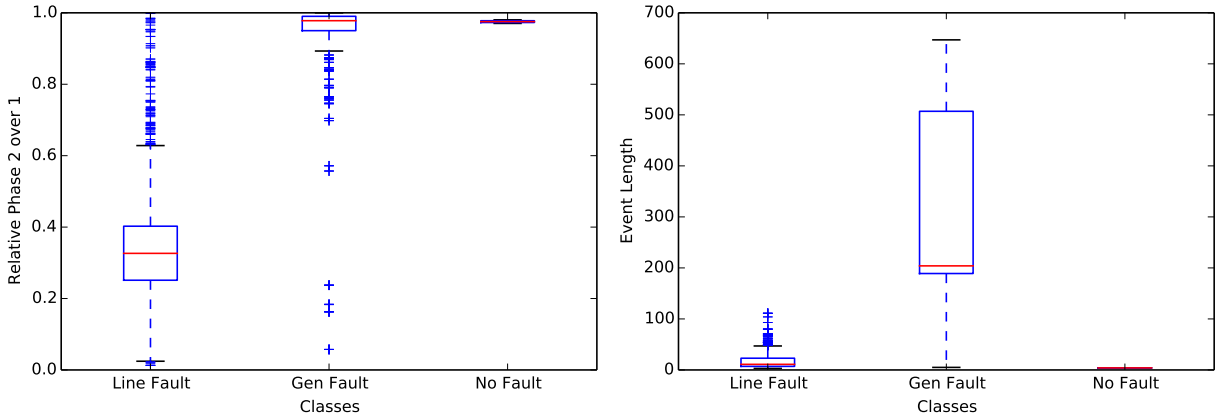


Figure 6.1: Boxplot of Features

Name	Calculation
Relative Phase 2 Over 1	$RP2/RP1$
Relative Phase 3 Over 1	$RP3/RP1$
Positive Sequence Deviation	$\|1.0 - v_{ps}/ss_{ps}\|$
Frequency Deviation	$1.0 - f/ss_f$
Event Length	$length(FrequencyWindow)$
3rd Phase Over Delta V	$RP3/\Delta V$
2nd Phase Over Delta V	$RP2/\Delta V$
1st Phase Over Delta V	$RP1/\Delta V$
Positive Sequence Over Delta V	$\|1.0 - v_{ps}/ss_{ps}\|/\Delta V$
Frequency Deviation Over Delta V	$(1.0 - f/ss_f)/\Delta V$

Table 6.1: Generator Fault Feature Breakdown

this case study, we applied a similar method with this dataset. Sites on the gird that are closer to an event have higher ΔV, but we have observed that on rare occasions sites that are very close to the location of the fault can measure low voltages for an extended amount of time after the rest of the grid has returned to normal. Therefore, our ΔV filter removes all data points greater than the 95th percentile, as well as all data less than the 30th percentile. We also observed that frequency is an important measurement in characterizing the difference between line faults and generator faults. Therefore, we removed data with a frequency deviation greater than the 95th percentile or lower than 60th percentile. Ultimately, we chose

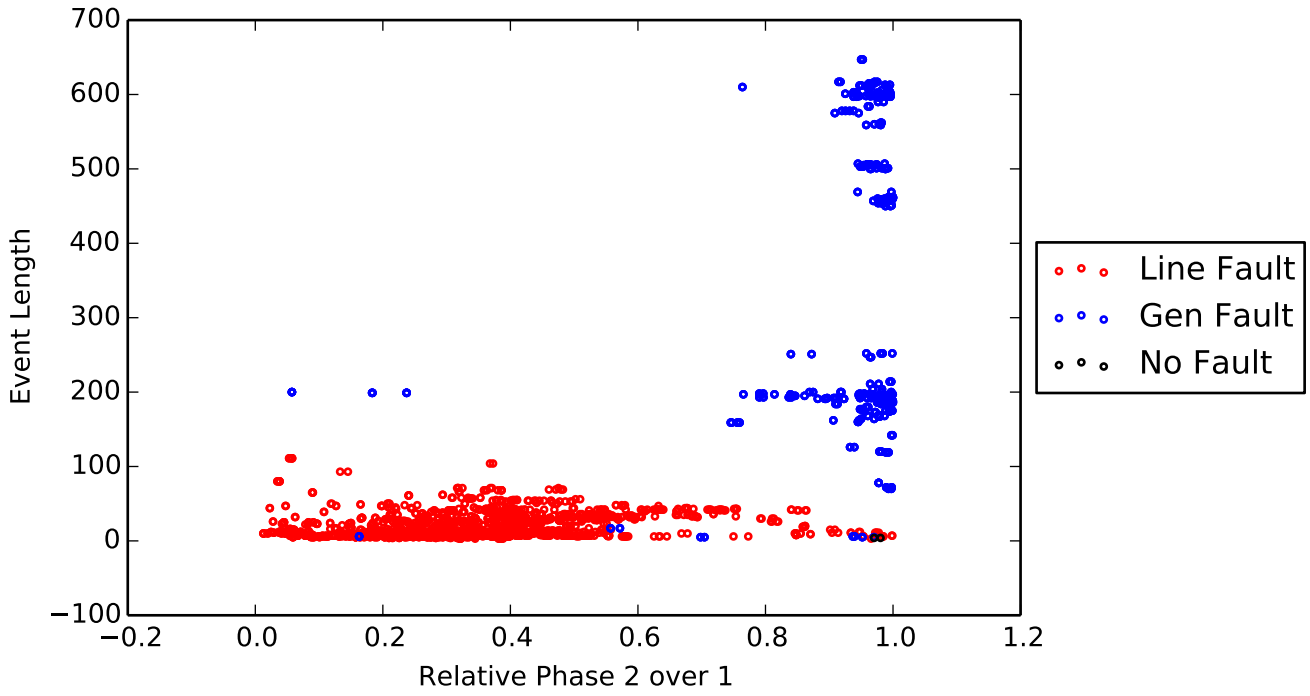


Figure 6.2: Comparison Plot of 2 Features

to filter our data by using the logical and of the ΔV filter and frequency deviation filters. After filtering this generator dataset was reduced to 3458 data points. Of the remaining data points points 2469 are line faults, 987 are generator faults and the remaining 2 are no fault data.

6.4 Clustering Results

For this experiment we chose DBSCAN as our clustering algorithm because of the success we have had with it in the past. Using all the features detailed in Table 6.1 and the filtering descried in section 6.3 we successfully clustered our data into 9 clusters. For completeness we have also included a clustering of the data without the filtering. This result is given in Figure 6.3 and Table 6.2.

As expected this unfiltered clustering performs poorly with an homogeneity score of 0.492.

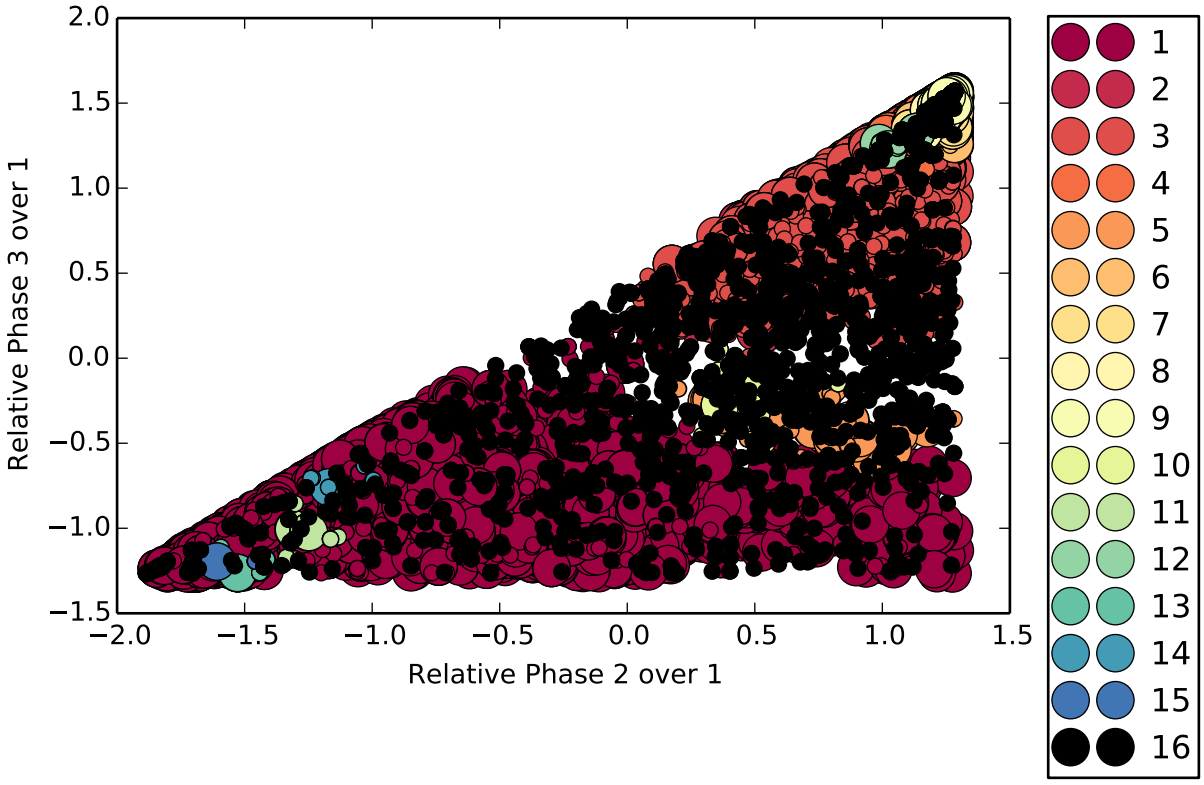


Figure 6.3: DBSCAN Clustering Generator Fault Dataset

Line faults predominately were clustered into a single cluster while both generator faults and no faults spread out among the various clusters. Since this clustering is 10-dimensional we are not able to show all 10 dimensions in the figures. Instead we only show the first 2 as they are the features used for our previous instantaneous clustering.

We then used DBSCAN to cluster on the filtered subset of this data. The results of the filtered data set are given in Figure 6.4 and a breakdown is given in Table 6.3. This filtered clustering performs well in separating all three classes of data from each other. LF are all contained in cluster 2 and NF are all contained within cluster 4. However, it has split GF into all 9 of the clusters. Nonetheless, this approach still results in a high homogeneity score of 0.96. This shows that the features and filters are helpful in improving homogeneity by characterizing the data in such a way that clustering is able separate GF data from the

	LF	GF	NF
1	**92.57%**	13.01%	5.75%
2	0.05%	**35.58%**	17.64%
3	3.04%	26.52%	28.13%
4	0.00%	4.39%	0.00%
5	1.70%	0.00%	0.00%
6	0.00%	3.29%	0.39%
7	0.00%	2.53%	0.00%
8	0.00%	0.66%	0.00%
9	0.00%	1.10%	0.00%
10	0.00%	0.82%	0.00%
11	0.00%	0.77%	0.00%
12	0.36%	0.00%	0.00%
13	0.00%	0.49%	0.00%
14	0.00%	0.38%	0.00%
15	0.03%	0.11%	0.00%
Noise	2.26%	10.32%	**48.08%**

Table 6.2: Non Filtered DBSCAN Generator Fault Clustering % Breakdown

rest, but these features place groups of GF data too far apart from each other within the 10-dimensional space for those groups to cluster into a single generator fault cluster.

6.5 Concluding Thoughts on Case Study 4

Characterizing and categorizations generator faults and line faults is nontrivial. Empirical feature selection and filtering greatly increase homogeneity, but generator faults in and of themselves are still a challenge as they are split among many groups. However, GF fault data can still be mostly separated from other classes as shown by a high homogeneity score of 0.96.

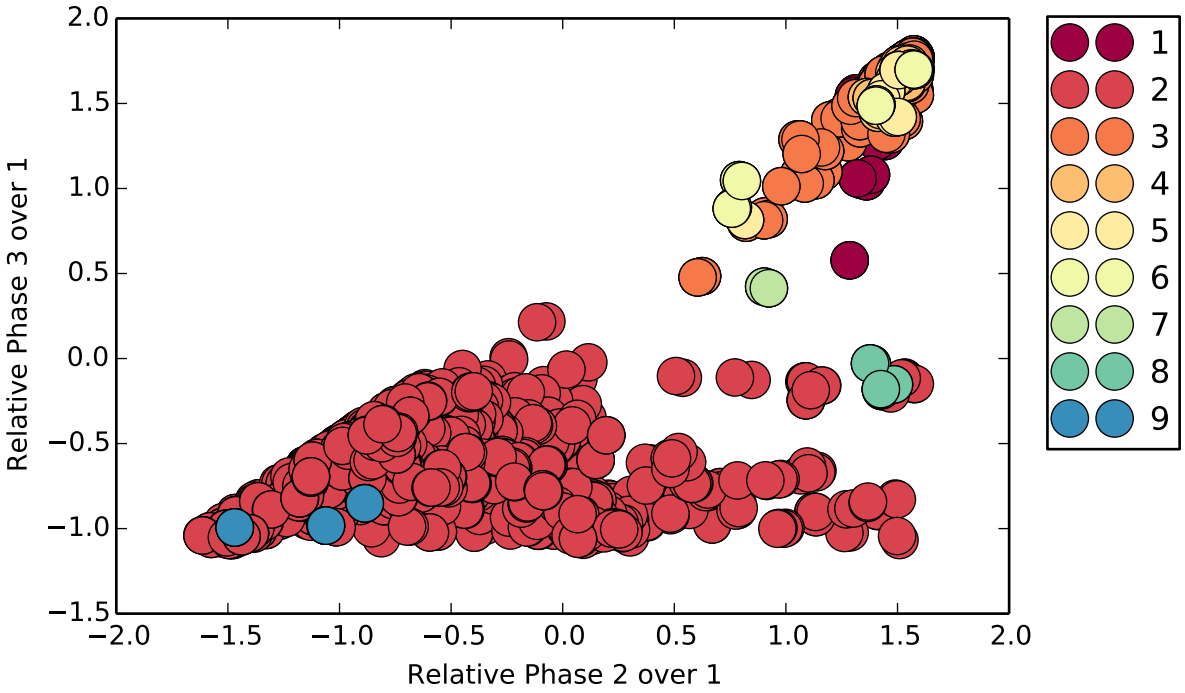


Figure 6.4: DBSCAN Clustering on Filtered Generator Fault Dataset

	LF	GF	NF
1	0.00%	29.18%	0.00%
2	**100.00%**	1.22%	0.00%
3	0.00%	**56.84%**	0.00%
4	0.00%	4.26%	**100.00%**
5	0.00%	2.43%	0.00%
6	0.00%	2.43%	0.00%
7	0.00%	0.61%	0.00%
8	0.00%	1.22%	0.00%
9	0.00%	1.82%	0.00%

Table 6.3: Filtered DBSCAN Generator Fault Clustering % Breakdown

Chapter 7

Conclusion

With recent advances in modern sensing communication, and information technologies, smart grid has become an emerging field with its own challenges. Namely, the increase in data collection and measurements have lead to the need for automatic solutions to data analysis. Clustering is an unsupervised machine learning technique that can be applied to gain greater understanding of data characterization and categorization. In the specific application of line fault detection we have shown that good feature selection and filtering when used with the DBSCAN clustering algorithm clusters our data into 3 groups that for the most part fit a known taxonomy. However, under the same features and filters time series clustering performs very poorly even though our dataset is time series in nature. We have shown a method for using clustering as an initial step to training SVM classifiers, and while this method does work well when using filtered time series clusters as training, it does no better than a single SVM trained on the entire dataset. In addition, both filtered results do not perform any better than random clustering. We have also shown that instantaneous clustering provides training sets for SVMs that are dominated by a single class. This leads to SVMs that do hardly any addition classification as they clasfy all points as the dominate training class. This leads to some SVMs performing very well, which show that these regions of the feature

space are very good indicators of a given class. Finally we introduced a set of 10 novel features and filtering thresholds that when used with the DBSCAN algorithm provide a clustering that is able to characterize and categorize generator faults in comparison to line faults and no fault data with a homogeneity score of 0.96. For future work, we look to further develop time series clustering. Further development of features that are modeled to fit the time series paradigm could lead to increased homogeneity. Semi-supervised clustering is another technique our research could explore in the future. Given our current knowledge of features and labels of our datasets a logical next step for this research would be to develop a model for semi-supervised clustering in an effort to improve clustering of unknown data points. In addition, semi-supervised clustering could be used as a next step for case study 3, in which we would use semi-supervised clustering instead of DBSCAN and hierarchical clustering. Throughout this thesis we have shown that clustering is an important tool to characterize and categorize data into different group, but good feature selection and filtering are just as if not more, important than the clustering algorithm used.

Bibliography

[1] C. Gungor Vehbi, S. Dilan, K. Taskin, E. Salih, B. Concettina, C. Carlo, and P. Hanchk Gerhard, "Smart grid technologies: Communication technologies and standards," *IEEE Transactions on Industrial Informatics*, vol. 7, no. 4, pp. 529–539, 2011.

[2] L. Rokach and O. Maimon, "Clustering methods," in *Data mining and knowledge discovery handbook*. Springer, 2005, pp. 321–352.

[3] J.-A. Jiang, J.-Z. Yang, Y.-H. Lin, C.-W. Liu, and J.-C. Ma, "An adaptive pmu based fault detection/location technique for transmission lines. i. theory and algorithms," *IEEE Transactions on Power Delivery*, vol. 15, no. 2, pp. 486–493, Apr 2000.

[4] G. Liu and V. Venkatasubramanian, "Oscillation monitoring from ambient pmu measurements by frequency domain decomposition," in *IEEE International Symposium on Circuits and Systems (ISCAS 2008)*, May 2008, pp. 2821–2824.

[5] G. Chang, J.-P. Chao, H.-M. Huang, C.-I. Chen, and S.-Y. Chu, "On tracking the source location of voltage sags and utility shunt capacitor switching transients," *IEEE Transactions on Power Delivery*, vol. 23, no. 4, pp. 2124–2131, Oct 2008.

[6] J. E. Tate and T. J. Overbye, "Line outage detection using phasor angle measurements," *IEEE Transactions on Power Systems*, vol. 23, no. 4, pp. 1644–1652, 2008.

[7] S. Rüping, "Svm kernels for time series analysis," Universität Dortmund, Dortmund, Technical Report, SFB 475: Komplexitätsreduktion in Multivariaten Datenstrukturen, Universität Dortmund 2001,43, 2001. [Online]. Available: http://hdl.handle.net/10419/77140

[8] Y. Wu and E. Y. Chang, "Distance-function design and fusion for sequence data," in *CIKM '04 Proceedings of the thirteenth ACM international conference on Information and knowledge management*, 2004, pp. 324–333.

[9] P. K. Ray, S. R. Mohanty, N. Kishor, and J. P. Catalão, "Optimal feature and decision tree-based classification of power quality disturbances in distributed generation systems," *IEEE Transactions on Sustainable Energy*, vol. 5, no. 1, pp. 200–208, 2014.

[10] R. Salat and S. Osowski, "Accurate fault location in the power transmission line using support vector machine approach," *IEEE Transactions on Power Systems*, vol. 19, no. 2, pp. 979–986, May 2004.

[11] F. R. Gomez, A. D. Rajapakse, U. D. Annakkage, and I. T. Fernando, "Support vector machine-based algorithm for post-fault transient stability status prediction using synchronized measurements," *IEEE Transactions on Power Systems*, vol. 26, no. 3, pp. 1474–1483, 2011.

[12] D. Nguyen, R. Barella, S. Wallace, X. Zhao, and X. Liang, "Smart grid line event classification using supervised learning over pmu data streams," in *Proccedings of the 6th IEEE International Green and Sustainable Computing Conference*, 2015.

[13] Y.-G. Zhang, Z.-P. Wang, J.-F. Zhang, and J. Ma, "Fault localization in electrical power systems: A pattern recognition approach," *International Journal of Electrical Power & Energy Systems*, vol. 33, no. 3, pp. 791–798, 2011.

[14] R. Diao, K. Sun, V. Vittal, R. O'Keefe, M. Richardson, N. Bhatt, D. Stradford, and S. Sarawgi, "Decision tree-based online voltage security assessment using pmu measurements," *IEEE Transactions on Power Systems*, vol. 24, no. 2, pp. 832–839, May 2009.

[15] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *KDD*, vol. 96, no. 34, 1996, pp. 226–231.

[16] F. Murtagh, "A survey of recent advances in hierarchical clustering algorithms," *The Computer Journal*, vol. 26, no. 4, pp. 354–359, 1983.

[17] K. Wagstaff, C. Cardie, S. Rogers, S. Schrödl *et al.*, "Constrained k-means clustering with background knowledge," in *ICML*, vol. 1, 2001, pp. 577–584.

[18] T. Räsänen and M. Kolehmainen, "Feature-base clustering for electricity use time series data," in *Adaptive and Natural Computing Algorithms*, vol. 5495. Springer Berlin Heidelberg, 2009, pp. 401–412.

[19] T. W. Liao, "Clustering of time series data–a survey," *Pattern Recognition*, vol. 38, no. 11, pp. 1857–1874, 2005.

[20] T. chung Fu, "A review on time series data mining," *Engineering Applications of Artificial Intelligence*, vol. 24, pp. 164–181, 2011.

[21] T.-c. Fu, F.-l. Chung, V. Ng, and R. Luk, "Pattern discovery from stock time series using self-organizing maps," in *Workshop Notes of KDD2001 Workshop on Temporal Data Mining*. Citeseer, 2001, pp. 26–29.

[22] D. Chakrabarti, R. Kumar, and A. Tomkins, "Evolutionary clustering," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2006, pp. 554–560.

[23] O. Antoine and J.-C. Maun, "Inter-area oscillations: Identifying causes of poor damping using phasor measurement units," in *Power and Energy Society General Meeting, 2012 IEEE*. IEEE, 2012, pp. 1–6.

[24] O. P. Dahal, S. M. Brahma, and H. Cao, "Comprehensive clustering of disturbance events recorded by phasor measurement units," *IEEE Transactions on Power Delivery*, vol. 29, no. 3, pp. 1390–1397, 2014.

[25] L. Ye and E. Keogh, "Time series shapelets: A new primitive for data mining," in *KDD '09 Poceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009, pp. 947–956.

[26] E. J. Keogh and M. J. Pazzani, "Scaling up dynamic time warping for datamining applications," in *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2000, pp. 285–289.

[27] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra, "Locally adaptive dimensionality reduction for indexing large time series databases," in *SIGMOD '01 Proceedings of the 2001 ACM SIGMOD international conference of Management of data*, 2001, pp. 151–162.

[28] M. Wollmer, M. Al-Hames, F. Eyben, B. Schuller, and G. Rigol, "A multidimensional dynamic time warping algorithm for efficient multimodal fusion of asynchronous data streams," *Neurocomputing*, vol. 73, no. 1–3, pp. 366–380, 2009.

[29] G. ten Holt, M. Reinders, and E. Hendriks, "Multi-dimensional dynamic time warping for gesture recognition," in *Thirteenth annual conference of the Advanced School for Computing and Imaging*, vol. 300, 2007.

[30] A. Singhal and D. E. Seborg, "Clustering multivariate time-series data," *Journal of Chemometrics*, vol. 19, no. 8, pp. 427–438, 2005.

[31] Z. Xing, J. Pei, and E. Keogh, "A brief survey on sequence classification," *ACM SIGKDD Explorations Newsletter*, vol. 12, no. 1, pp. 40–48, 2010.

[32] X. Wang, A. Mueen, H. Ding, G. Trajcevski, P. Scheuermann, and E. Keogh, "Experimental comparison of representation methods and distance measures for time series data," *Data Mining and Knowledge Discovery*, vol. 26, no. 2, pp. 275–309, 2013.

[33] X. Liang, S. Wallace, and X. Zhao, "A technique for detecting wide-area single-line-to-ground faults," in *Proceedings of the 2nd IEEE Conference on Technologies for Sustainability (SusTech 2014)*, ser. SusTech '14. IEEE, 2014, pp. 1–4.

[34] A. Rosenberg and J. Hirschberg, "V-measure: A conditional entropy-based external cluster evaluation measure," in *EMNLP-CoNLL*, vol. 7, 2007, pp. 410–420.