Master Thesis:
Unsupervised Detection of Patterns from Time Series Data

Rakesh Lagare
Matriculation number:374183

Advisors:

Mr. Shreekanth Devasya
Mr. Alexander Graß

Examiners:

Prof. Dr. Matthias Jarke
Prof. Dr. Christian Beecks

Rheinisch-Westfälische Technische Hochschule Aachen

February 27, 2020

Zentrales Prüfungsamt/Central Examination Office

RWTH AACHEN UNIVERSITY

Eidesstattliche Versicherung
Statutory Declaration in Lieu of an Oath

_____ _____

Name, Vorname/Last Name, First Name

Matrikelnummer (freiwillige Angabe)
Matriculation No. (optional)

Ich versichere hiermit an Eides Statt, dass ich die vorliegende Arbeit/Bachelorarbeit/ Masterarbeit* mit dem Titel
I hereby declare in lieu of an oath that I have completed the present paper/Bachelor thesis/Master thesis* entitled

_____

_____

_____

selbstständig und ohne unzulässige fremde Hilfe (insbes. akademisches Ghostwriting) erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt. Für den Fall, dass die Arbeit zusätzlich auf einem Datenträger eingereicht wird, erkläre ich, dass die schriftliche und die elektronische Form vollständig übereinstimmen. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.
independently and without illegitimate assistance from third parties (such as academic ghostwriters). I have used no other than the specified sources and aids. In case that the thesis is additionally submitted in an electronic format, I declare that the written and electronic versions are fully identical. The thesis has not been submitted to any examination body in this, or similar, form.

_____ _____

Ort, Datum/City, Date

Unterschrift/Signature

*Nichtzutreffendes bitte streichen

*Please delete as appropriate

Belehrung:
Official Notification:

§ 156 StGB: Falsche Versicherung an Eides Statt
Wer vor einer zur Abnahme einer Versicherung an Eides Statt zuständigen Behörde eine solche Versicherung falsch abgibt oder unter Berufung auf eine solche Versicherung falsch aussagt, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.
Para. 156 StGB (German Criminal Code): False Statutory Declarations
Whoever before a public authority competent to administer statutory declarations falsely makes such a declaration or falsely testifies while referring to such a declaration shall be liable to imprisonment not exceeding three years or a fine.
§ 161 StGB: Fahrlässiger Falscheid; fahrlässige falsche Versicherung an Eides Statt
(1) Wenn eine der in den §§ 154 bis 156 bezeichneten Handlungen aus Fahrlässigkeit begangen worden ist, so tritt Freiheitsstrafe bis zu einem Jahr oder Geldstrafe ein.
(2) Straflosigkeit tritt ein, wenn der Täter die falsche Angabe rechtzeitig berichtigt. Die Vorschriften des § 158 Abs. 2 und 3 gelten entsprechend.
Para. 161 StGB (German Criminal Code): False Statutory Declarations Due to Negligence
(1) If a person commits one of the offences listed in sections 154 through 156 negligently the penalty shall be imprisonment not exceeding one year or a fine.
(2) The offender shall be exempt from liability if he or she corrects their false testimony in time. The provisions of section 158 (2) and (3) shall apply accordingly.

Die vorstehende Belehrung habe ich zur Kenntnis genommen:
I have read and understood the above official notification:

_____ _____

Ort, Datum/City, Date

Unterschrift/Signature

Contents

Abstract

Similarity search in time series data has gained considerable success in building modern large-scale data analytics and forecasting applications. Focus of the existing Similarity search work is usually either on subsequence matching or full comparison where complete time series are compared. Subsequences are compared against each other to find patterns in time series. In the context of this thesis In last few decades, lots of similarity measures and classification methods were proposed due to the huge growth in collection of data. Anyways, nearest neighbor algorithms are considered as so far one of the best time series classifiers. They can be easily adjusted such that any similarity search problem can easily be solved by using distance functions. Many research papers have noticed that, distance functions produce more accurate results and has dependency on very less parameters. Despite the effectiveness of distance functions, they have some major drawbacks, where the main disadvantages are high time and space complexity. Another shortcoming is they don't provide any significant understanding of the classification results. Final reason is how to obtain essential hidden patterns from huge amounts of time series data effectively for any further time series tasks. Hence, there is a need for an approach that can process large dataset and provides efficient as well as comprehensive insight into the time series data.

This document proposes a master thesis work that aims to provide an approach that identifies motifs as well as discords in time series data and ranks them based on frequency of occurrences. Similarity search is performed using multi-step approach where symbolic representation and approximation is adopted as a fundamental strategy. Patterns are further narrowed down to produce more accurate results using distance functions like DTW,Euclidean Distance. We shall reviewed and compared the state-of-the-art similarity search architectures and visualize the results. Proposed architecture is empirically compared with the state-of-the-art architecture which demonstrate its performance improvement, provide fast and precise similarity search result.

Keywords: Time Series, Similarity Search, Motif Discovery, Discords, Nearest Neighbor

Chapter 1

Introduction

According to Brockwell et el.[7], a time series is a set of observations x_i, each being recorded at a specific time t. It can be defined scientifically as a set of vectors x(t), t = 0,1,2, 3... where t is elapsed time [22] [7]. It is interesting to note that variable x(t) is treated as a random variable. There are many ways to categorize a time series. A time series is a univariate if it contains single variable records and it is called as multivariate if it contains more than one variable records. A time series can also be categorized as continuous or discrete. It can be continuous if the observations are estimated at every time instance. For example, sensor readings, water flow, chemical process readings etc. A time series is discrete where observations are measured at discrete time intervals. Usually these observations are made at equal time intervals like hourly, weekly, monthly or yearly. Company production, stock exchange, population of different countries represent discrete time series data.

Over the years, time series analysis has gained the considerable amount of attentions of variety of researchers where they invested vast amount of time in analyzing the many aspects of the time series components. The main motivation behind the analysis is to collect, continuously study the previous time series data and make observations on certain aspects of the time series. Which provides better understanding and overview of the huge data without missing out the minute details. Further, with the help of these observations appropriate model can be developed for the prediction or forecasting the time series.

In the past couple of decades, Motif and discord discovery has been one of the main aspects of time series analysis. To define motif and discord we need the context in which they are being used because they can be defined in different ways according to the context. In the context of this thesis work, Motifs can be defined as frequently occurring repeated patterns of a longer time series and Discords are rarely occurring patterns of a longer time series. Finding out those frequently occurring repeated patterns in a time series is known as motif discovery. Motif and discord discovery have been used in different aspects, such as telecommunications, medicine, web and sensor networks for many significant tasks such as classification [16] [33], time series visualization [32], data clustering [31], anomaly detection [11] [23] [10].

Time series data is an area of vast study in similarity search and visualization research. There are various methods for similarity search. Aigner [2] examined the various systems for pattern search. Clustering method is used by many systems for similarity search and visualizing time series data, where the main goal is to find interesting groupings of the data considered as collection.

Most of the time, domain expert wants to find not only the most frequently occurring motif, but rather the patterns in the data which are questionable or interesting yet rare patterns in certain domains. One important point to consider is domain expertise is essential for such pattern discovery. Finding these most occurring motifs as well as rare patterns with or without domain expertise is why classic similarity search approaches fail to produce the expected or desired results. There are very few approaches for such similarity search and scaling up motif search, which explicitly addresses the fast and precise similarity search.

Consider an example of the portion of data collected from sensor measuring process of a plastic Injection molding machine. In this the process expert must find out anomalies. But while searching for anomalies, the expert might encounter the measurements such as machine shutdown or start up cases, where calibration has similar pattern. But these patterns are known/uninteresting which needs to be ignored. One could argue that this issue only affects the beginning of signals and could be easily solved by domain expertise. But there is a need of domain knowledge. And it is also important to consider that this problem is not restricted to particular data.These known/uninteresting patterns may mask a much unexpected and rarer repeated pattern. Sometimes, even the automatic methods fail in the above cases without the user intervention. Hence a more general and semi-automatic approach is proposed in this thesis.

1.1 Research Question

A lot of research has been carried out on the various Time Series analysis. Similarity Search has been the main aspect of these analysis. Between different patterns in Time series, we focus on Motif and Discord discovery.

The research questions we would like to answer can be summarized as :

- What is the efficient way of performing semi automatic similarity search and rank patterns based on the frequency of occurrences ?

The remainder of this paper is sorted out as follows. Chapter 2 gives brief idea about different methods for Time series representation as it is main component of proposed approach. Chapter 3 provides background of the existing algorithms and research work as well as discusses relevant work. Chapter 4 discusses about different underlying components proposed model and their alternatives. Chapter 5 explains proposed architecture and in depth explanation about the implementation.

Section 6 contains an experimental evaluation of the proposed approach. In chapter 7, conclusion of the thesis work is discussed. Finally, future work of thesis explicated in section 6.

Chapter 2

Theory

2.1 Time Series Representation

Time series can be represented in various ways. Most of them tend to depict the time series with combination of basic functions. The most well known as mentioned in [26] are Discrete Fourier Transformation (DFT), Discrete Wavelet Transformation (DWT), Symbolic Aggregate Approximation with Piecewise Aggregate Approximation(PAA). Each one them have benefits and the drawbacks of each such representation. Few of the time series representation technique mentioned in [20] are summarized below.

2.1.1 Discrete Fourier Transform

The first suggested technique for reducing the size of massive data in the time series domain was Discrete Fourier Transform (DFT). DFT converts a time series domain sequence into another frequency domain sequence. The original sequence is represented and replicated roughly by only the first few frequency components. Controls the compression ratio and data representation precision in the number of frequencies chosen (DFT coeffecents). Even if the computational complexity of the standard DFT approximation for long-range time series n is $O(n^2)$, the Fast Fourier Transform (FFT) will reduce it to $O(n \log n)$. For the similarity analysis of a financial time-series database, DFT was used in [1] as a feature extraction method. It controls the compression ratio and data representation precision in the number of frequencies chosen (DFT coeffecents). Even if the computational complexity [30] of the standard DFT approximation for long-range time series n is $O(n^2)$, the Fast Fourier Transform (FFT) will reduce it to $O(n \log n)$. For the similarity analysis of a financial time-series database, DFT was used in [1] [17] as a feature extraction method.

The use of the first few frequency components avoids the problem of high dimensionality and improves reasonable search time performance. Also influences the accuracy of representing the original time series, the number of frequency components affects the efficiency of further time series mining activities.

For analyzing periodical time series Fourier transform [43] is optimal. The DFT performs well positioned in the frequency domain, not in the time domain. For example, it is not possible to detect a Spike in its DFT-transformed series in the electrocardiogram (ECG) series. Then Discrete wavelet transform(DWT)is used to solve DFT limitations.

2.1.2 Discrete Wavelet Transform

Discrete Wavelet Transform gives a multi-resolution time series representation and has a time-frequency position. The Harr Wavelet Transfrom (HWT) complexity is $O(n)$ whereas $O(n \log n)$ is the DFT's. Chan et al.[9] suggested a technique based on the HWT time series matching. The time series, based on HWT, were effective in terms of pruning power, scalability and complexity as compared to the DFT-based F-Index according to the stock market assessment and different sizes of synthetic time series datasets[17]. Nevertheless, the DWT has certain disadvantage, for example, when HWT is used in the ECG time series, the wavelet coefficients have to be chosen according to their importance. Because different time series have large coefficients in different locations, it becomes much more difficult to index a time series with complex DWT coefficients [26].

2.1.3 Symbolic Aggregate Approximation with Piecewise Aggregate Approximation(PAA)

Keogh et al. and Faloutos [29] have proposed a data dimensionality reduction method, Piecewise Aggregate Approximation (PAA), based on the estimation sophistication and the tedious indexing scheme. PAA splits the time series of length n into w segments of the same size and uses each segment's integer mean value as coefficient of PAA. It is fast and easy to incorporate the transformation.

Even though above discussed dimensionality reduction methods have different way of time series representation, at the same time have some vulnerabilities. Since the break through in time series analysis causes to an excessive number of algorithms, it is highly recommended that time series needs to be transformed into symbolic representations. Symbolic aggregate approximation(SAX) is one of the earliest and effective dimensionality reduction technique that provides symbolic way of representing data. SAX basically converts the data into a PAA representation and then transforms the PAA coefficients into a symbolic representation.

As PAA is integral part of the SAX, initially the time series data is represented as PAA coefficients. Then using a look up table which contains PAA coefficients to letter mapping, these coefficients are transformed into a string. This is where time series dimensionality reduction occurs. So the complete time series which is represented which was given as a numerical dataset now its transformed into series of letters or a string. As compared to DFT and DWT, data is processed in less time using PAA.

Chapter 3

Related Work

This chapter discusses some of the work in the field of motif and discord discovery in time series, distance functions, time series representation and dimensionality reduction .

Because of the increasing amount of time series data gathered and the boost in the complexities involved in its practical understanding, processing and analyzing such data has become more substantial procedures to understand the characteristics of the data and gain constructive insights and understanding from it. Different methods were developed to extract valuable information, including data mining, from raw time series data. Nonetheless, automated approaches are not producing satisfactory results in many cases, so experts focus on visual analytics tools to carry out their tasks. Visual Analytics combines the characteristics of computer technologies with human capabilities to promote discovery, study, interpretation and perspective. The digital analytics approach seeks to closely integrate automated methods of research with dynamic simulation in order to gain information from raw data and to present an opportunity for researchers to evaluate, interpret and understand data via interaction activities.

Recently lot of research work is going on for similarity search in time series data. Usually similarity search is known as an important subroutine task for classification and clustering in time series. Focus of the existing work is usually on either subsequence matching or full comparison where the complete time series are compared. Subsequences of equal length are compared against each other to find patterns in time series. In the context of the thesis, similarity search referred as sequence comparison in regards of their value or shape.

Definitive distance functions such as Dynamic Time Warping, Euclidean Distance or cosine distance also known as angular separation are used to measure the similarity or dissimilarity between two time series or two Subsequences.

3.1 Distance Functions

Ideally, we could easily calculate the similarity of two time series or two different segments of the same time series by just calculating the Euclidean distance (shortest path between two points) between them that occur at the same time. This is an effective similarity measurement if both time series are in a sync or aligned and move at the very same time and speed (for example every single event in both time series happen at the very same time). When the time series is out of sync, this indicates that similar points on the both time series are extended farther apart by time in simple terms, Euclidean distances are getting larger and the time series are becoming less similar.

Truong and Anh [39] proposed a Dynamic Time Warping (DTW), an alternate distance function for Euclidean Distance to tackle with similarity search in two time series which are not in sync. As pointed out by Ding et al. [25] that Dynamic Time Warping is an algorithm used to measure similarity between two sequences of time series data which may vary in time or speed. Recently, Rakthanmanon et al. [36] proposed a series of techniques for accelerating the DTW similarity search and a clustering-based method to find out motifs on large time series datasets. In any case, the complexity of similarity search on N time series sub-sequences is about $O(N)$, but the complexity of motif discovery on N time series sub-sequences cost about $O(N^2)$.

In time series analysis, the all-around characterized and approximated representation for the data is the most significant. In the last couple of decades lots of approaches have been proposed to address similarity search issue is time series data.

One of the earliest methods was proposed by Keogh et al. [26] called Piecewise Aggregate Approximation (PAA) to represent each segment of time series data into k fragments with equal length and average value of each segment, which is used as a coordinate of a k-dimensional feature vector. However, the PAA approach has various advantages over different methods. It is very fast and easy to implement and also index can be built in linear time. However, it additionally has a few disadvantages. As mentioned, PAA approach limits dimensionality by the mean estimations of equivalent measured frames. This may cause a probability to miss some significant patterns in some time series data analysis.

3.2 Similarity Search

Another similarity search algorithm, Matrix Profile was introduced by [13], which takes an arbitrarily long, unlabeled time series and produces intuitive and useful patterns. One of the important observations made in this algorithm is that it generally does not attempt to explain all the data in the time series, rather only considers salient sub-sequences. They used Minimum Description Length (MDL) to extract such kind of sub-sequences. Since it uses nearest neighbours method, chances of missing all other important patterns are very high.

One of the significant yet non-inconsequential issue for a large amount of time series data set is efficient and accurate similarity searching. To overcome this problem there are many dimensionality reduction techniques [40] [14] [8] have been proposed. Few of the vastly used methods are Symbolic Aggregate Approximation (SAX) [29] and Discrete Fourier transform. Numerous researchers have likewise thought about transforming real valued time series into symbolic portrayals, such represen-

tations would conceivably enable algorithms for content preparing, as well as permitting segments to be handled by the streaming network. While the SAX approach permits a generally excellent dimensionality reduction and distance measures by the mean values of equal sized frames. This value-based pattern search causes a high probability to miss some significant patterns in some time series data such as financial time series data.

Lin et al. [29] is proposed method a classical symbolic approach for time series called Symbolic Aggregate Approximation (SAX). The main idea behind this approach is to transform numerical time series into a sequence of symbols using mapping rules. Dimensionality or numerosity reduction can be achieved by SAX using PAA. PAA is core part of the SAX. It extends the PAA based approach which provides low computational complexity and simplicity as well as satisfactory sensitivity and selectivity in range query processing. In SAX, initial step is transformation of time series data into PAA representation. Transformed PAA representation is symbolized into a sequence of strings. SAX has lower bound to the Euclidean Distance. The distance in symbolic representation and the Euclidean distance has the error value that falls within the lower bound. Thus PAA helps SAX to speed up the similarity search task of time series data in the mean time maintaining the better accuracy results.

Pavel Senin proposed a proposed with the principles of SAX is SAX-VSM. SAX-VSM [37] is pattern discovery method for time series which is able to consequently find and rank time arrangement designs by their significance to the class, which not just makes well performing classifiers and encourages clustering, yet in addition gives an interpretable class speculation.

Chapter 4

Proposed Model

In chapter 2 Theory, we discussed about different methods for Time series representation as it is main component of proposed approach. In this chapter we discuss different underlying components proposed approach and their alternatives. Each component have many alternatives so we mention the purpose behind selecting particular method for each component. First we brief through importance data preprocessing and ways of handling faulty data. Then we elaborate necessity of symbolic representation in the context of our work. Next we discuss ways of normalizing data and its significance to the proposed work. Later we discuss different methods of segmentation. Then we go through how PAA works as core for symbolic representation. Finally different distance functions that can be adapted in proposed approach.

4.1 Data Preprocessing

According to A.Famili [18] data preprocessing is absolutely critical as it helps raw experimental data to be increased in quality. The main objective of data preprocessing is to minimize or greatly reduce those small data contributions involved with the experimental error, regardless of whether they are systematic, such as baseline drifts or random, such as the noise contributions associated with the instrumental measures. However, encoding of the experimentally produced data can also be done at this pretreatment level to retain using the lowest data storage specifications. Therefore, to enable a more accurate measurement and comparison of the examined samples, various chemical and mathematical normalization and scaling techniques can also be implemented.

As explained above raw time series data contains different types of unwanted and unprocessed data. Before reprocessing the raw data, it is significant to understand the flaws in the raw data as summarised in [6] :

- Imperfect data: lack of attribute values, lack of interest in certain attributes, or include only aggregate data.
- Noisy: contains mistakes or outliers.

- Inconsistent: the symbols or titles contain inconsistencies.

Data preprocessing is a one of the initial and major strategy in time series analysis. Machine or algorithm can not process raw data as they contains unreliable data format or human errors in storing the real world . Raw data needs to be transformed into comprehensible format so that data is fed into algorithms in expected format.

In the context of this thesis work, we mainly concentrate on the two aspects of faulty real world data. They are noisy and missing values. As explained above, data needs to be given in a suitable format. So we check for many noisy and missing values in raw data. Even before handling the data effectively, it is crucial to understand the idea of missing values and noisy data. If the concept of missing values is not dealt correctly , the results produced by the algorithms might be incorrect and can draw unexpected conclusions or even produce unexpected results. Noisy data is produced due to human error during data entry or while collecting the data from the different sources.

Data cleaning is one of the important part of data analysis. This section handles above mentioned faulty data effectively as explained in [35]. There are two ways to handle missing and noisy data Values. Either to fill the missing values manually or by attribute mean or the most probable value.

- First approach to deal with missing and noisy data is to treat them as null values. The main idea behind this is once the faulty is detected, replace that data with null values. If any row contains many null values for specific column then delete the entire row. In the same way if the specific column has higher than 75% of missing values then delete the entire column. This method is only recommended when there are enough samples in the data set. But one thing to remember is such a removal of huge data may result in a loss of information and that will not deliver the expected results while predicting output.

- Second approach is to deal with faulty data in statistical way. The mean, median or mode is determined from the non faulty data. This value is substituted for all the missing values in a particular column or a row. Actually this approach produces better results than the first approach where entire rows and columns are eliminated and thus will mitigate data loss. Second approach is to bring it closer to the disparity between adjacent values.

So in our work we use second approach as a initial step which is data reprocessing as it handles the data in better way without losing any important information.

4.2 Symbolic Representation

As explained in previous section handling the faulty data is initial as well as important step of our approach. In the same way data needs to be represented in suitable way to make pattern discovery task easy. As justified by Alexis et el. [4], the choice of a suitable representation for time series analysis continues to be important, especially to achieve a good compromise between the reduction in size and information stored. One way to achieve this by transforming original time series into symbol sequence and thus reduction in dimensionality. Symbolic time series event-driven

representation that encodes standard data points distributions. This technique was first described as a heuristic algorithm focused on a formalized approach to coclustering.

Apart from time series representation in symbolic way, many time series representations models have been released in recent decades, all of them have one or the other major flaws as explained in Chapter 2. Given the time series data, a model selection problem occurs for time series representation. Comparative experiments show the advantage of the new codification, which leads to interpretations that increase data compression significantly and preserve valuable information for classification tasks.

In addition to allowing previously only "batch-only" problems to be solved by the streaming data, other researchers proposed transforming real valued time series into symbolic representations. Such representations may help researchers to benefit from huge of data and algorithms in the text processing and the time series analysis.

The symbolic representation's dimensionality is the same as its original data and nearly all algorithms for data mining are of poor dimensionality. Furthermore, even if symbolic solutions can be described as distance measures, those remote measurements are not very much related to the distance measures defined in the original time series. Eventually, many of these symbolic methods require access to all details, before the symbolic representation is produced. This last function directly opposes attempts with streaming algorithms to use the symbolic representations.

In this thesis we introduce a symbolic time series representation method based on SAX [29]. One of the main requirement of our approach for symbolic representation is to reduce time complexity. This can be achieved by dimensionality reduction. As number of comparisons are consumes most of the algorithmic run time, if we reduces the comparison size then it automatically results in reduction in time complexity. This function is especially significant as it helps to perform such pattern search with the easily modified symbolic representation and delivers the same and some times better results as the original approach SAX. Lastly, our representation enables the similarity search for time series large data, with only a minimum of time and space.

4.3 Normalization

Once we decided that symbolic representation is a suitable method for our time series data, before proceeding further one more data preprocessing task called Normalization needs to carried out. Even though it is an intermediate step, but it plays important role in the final output. Normally normalization is used for many purposes such as to remove the different measurement units of the data or to convert the data to be of a same scale, which increases the efficiency and for reliability of the results. In our approach, normalization is used for latter cause.

There are many ways to achieve normalization such as Maximum Scaling , Min-Max scaling, Z-Normalization, Normalizing vectors etc. Each on of them have different purposes and as per our requirement we choose normalizing techniques. Z-Normalization and Min-Max scaling are two closely suitable for our approach.

4.3.1 Z-Normalization

The outcome of Z-score normalization [29] [37] is that the data will be rescaled in such a way that they have the characteristics of a normal distribution. $\mu = 0$ $\sigma = 1$ where μ is the mean and σ is the standard deviation from the mean. Z-Normalization also know as Z-score can be calculated as follow:

$$z = \frac{x - \mu}{\sigma}$$

4.3.2 Min-Max Scaling

Min-Max scaling [24] also known as Min-Max normalization is widely used normalization technique. The outcome of this method is that data is rescaled between the range of 0 to 1. By comparison to Z-normalization, the cost of maintaining a range is that may result with smaller standard deviations that can minimize the effect of outliers.

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Z-Normalization is used in our approach as data needs to fit under Gaussian Curve.

4.4 Segmentation

As explained in the previous section data needs to be normalized in order to scale all the data points to the similar level. But collected or given input data can be sometimes in millions or even in billions. At a time whole data series is hard to process. Transforming time series into symbols itself is a tedious work and if we choose to transform millions of data points into symbolic representation in one stretch, its not only close to impossible also impact the run time of the process and also on machine performance as it consumes lots of resources. Evaluation of such data becomes impossible. In last decades lots of research work is going on how huge data can be processed without any hampering the algorithm performance. One such possible solution is to divide the data into smaller segments hence the process is called segmentation.

Even though lots of research work is going on in the aspect of segmentation, but it is not discussed by the researchers on what basis the data is segmented. The simplistic solution to this issue is to use fixed length segments, so it is better to follow a dynamic and highly controlled approach so that the time series can be flexibly and easily segmented in compliance with user needs and implementations.

In order to solve such a segmentation problem, time series segments need to be calculated in the at different intervals of time series. This method allows a significant set of patterns for analysis or querying to be generated. By identifying significance interval directly from the time domain, time series segments of different lengths can be compared and intuitive pattern matching can be effectively and efficiently carried out. Many important segmentation methods exist as referenced

in [38] are explained below.

4.4.1 Partitioning algorithms

Partitioning algorithms were designed to build and evaluate various partitions according to certain parameters. The moderate idea is to find a k-cluster partition to refine the chosen partitioning requirements of a k. k-means and k-medoids [15] are the two principal heuristic processes. Over the first one, the center of the cluster is depicted for each cluster. During the second cluster, each of the items in the cluster is represented. When storing a time series, the partitioning algorithms have their benefits and limitations. K-means can for example deal with large amounts of data, and only with Euclidean distance as the mean is determined by the Euclidean distance.

4.4.2 Hierarchical algorithms

Hierarchical clustering is a cluster analysis method which seeks to create a cluster hierarchy, also referred to as hierarchical cluster analysis. Hierarchical clustering techniques are commonly classified into two groups [5]. For the first, every finding ends in its own cluster and couples of clusters combine as the hierarchy is shifted. It's a strategy "front up". The second is to initiate all experiments of one cluster and divisions are carried out recursively when the hierarchy is moved down. It's a strategy that's "top down". The hierarchical clustering is available over a range of distances compared with partitioning algorithms, but it can not accommodate a large data set. This allows us to pick an algorithm for segmentation that satisfies demand for a particular case.

4.4.3 Sliding Window

Sliding window is one of the most widely used segmentation algorithm because of it is not only easy to implement also can segment any large data set. Sliding window is an brute force algorithm works to segment data.

The basic sliding window algorithm as explicated by Keogh et el. [27] performs by anchoring the left point of a possible segment at the first data point in a time series and then trying to approximate the data to the right with increasing with longer segments. At some stage i, the error is larger than that provided by an user threshold , thereby transforming the anchor to i-1 sequence into a segment. The anchor is shifted to position i and the cycle repeats itself until the entire time series becomes a piecewise linear approximation. Sliding window pseudocode is as shown Figure [4.1]

Additional optimizations may be possible depending on the error measurement used. Vullings et al [41] observed that because of the uniformity of the residual error when adding additional data points, not all i values from 2 to the final selected value must be checked. They initially recommend that set i to s, where s is the average length of the previous segments. The algorithm continues to increase, as with the classic algorithm, if it is negative (the error calculated is still less than the max error). Otherwise the error will start to decrease until it is less than max error. This

Figure 4.1: Sliding window algorithm, referred from [27]

```
Algorithm Seg_TS = Sliding_Window(T , max_error)
anchor = 1;
while not finished segmenting time series
i = 2;
 while  calculate_error(T[anchor: anchor + i ]) < max_error
   i = i + 1;
 end;
 Seg_TS = concat(Seg_TS, create_segment(T[anchor: anchor + (i-1)]);
 anchor = anchor + i;
end;
```

optimization will greatly improve the algorithm if the average length of segments is relative to the standard length deviation. The non-decreasing residual error monotonous property enables binary search for the section range.

To explain in simple terms, sliding window algorithm is performed on the complete time series. Window starting from first(from left) data will keeps on shifting to right by user mentioned threshold. This process repeats for all the through out the time series.

Since our proposed approach should perform better on larger dataset with less time complexity, sliding window technique is used as segmentation method for segmenting large dataset into small sub-sequences as per user requirement.

4.5 Piecewise Aggregate Approximation

We discussed in the Symbolic Representation section that the time series representation method has become an important problem. And now a days increase in that dataset size making it even more difficult. Large datasets are harder to manage and analyze effectively. To solve that problem several high-level time series representations approaches have been proposed. For our approach, symbolic representation is suitable to reprsent large dataset.

In a similarity search, in order to minimize dimensionality PAA is used as core part. PAA calculates mean values of the equal sized segments. PAA focuses on mean value of the segments rather than all the data points in the segment. Even though it helps in transforming time series into symbols, sometimes it may lead to the missing of some significant pattern for some time serial data sets. And also the mean values of segments fails to capture other important information and produce often inaccurate results in time series analysis. During implementation we explain how to solve this problem. Even with its few drawbacks, PAA is straight forward option for standard form of real value representation which is simplified to obtain a series of mean segment values. A series is first divided into segments using a sliding window in order to symbolise them. Then the PAA assigns symbol to each segments.

Assume a time series of length (n) $Y = y_1, y_2, \ldots, y_n$ is to be separated or condensed into a series $X = x_1, x_2, \ldots, x_M$ where $M \leq n$, the overall equation defining the elements in the reduced sequence can be summarized as follows [28]:

$$\bar{X}_i = \frac{M}{n} \cdot \sum_{j=n/M(i-1)+1}^{(n/M)\cdot i} x_j$$

This equation gives the mean of the elements in the equally large frame that form the matrix of the reduced dimensional sequence.

- 1. M = n: Is a direct representation of the original sequence for the reduced series .
- 2. M = 1: The mean of the original sequence for the reduced series.

The second is a special case in which the product consists of a piecewise constant approximation method. This is the origin of the name piecewise aggregate approximate, since the different methods involves mean values in a frame. That is, the initial input vector is separated into frames and the mean values are determined in the frame. How these equally big frames are generated is the most interesting aspect of the algorithm. It should be noted here that the input vector is Z-Normalized before the actual mean of the windows is calculated. When it's done, it can be calculated on a piecewise basis. When $M < n$ and $M \bmod n > 0$, python code is given as follow Figure [4.2].

Figure 4.2: Snippet code of PAA, taken from [28]

```
import numpy as np

Y = np.array  # input vector
paa = int     # number of outputs in the output vector

if Y.shape[0] % paa == 0:
    sectionedArr = np.array_split(Y, paa)
    res = np.array([item.mean() for item in sectionedArr])
```

If the remainder is empty, the vector can be separated into the same sizes in principle. For these equal size frames, the mean can be determined. The same equation can not be applied when $M < n$ and $M \bmod n > 0$. The frames can no longer be easy to equi-sized when the balance is greater than zero. The way the frames can be made is unbalanced. It is now necessary to resize a frame so that the mean of an even-sized frame of the input vector is the each element of the output series. The is a sample code is given in Figure [4.3]:

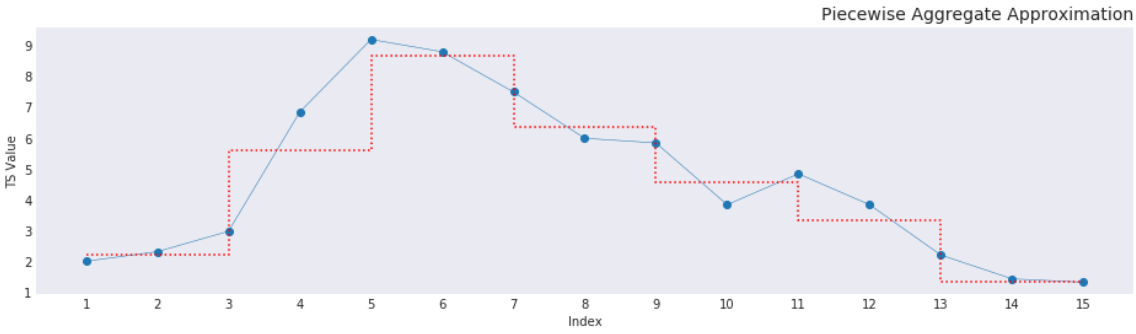
In figure [4.3] , *output_index*, creates the number of elements in the input index vector that make up one single frame for the index in the output vector equal to the unique values. The array below named the *input_index*, provides the indices of the input vector Y that should be summed up from

```python
import numpy as np

Y = np.array  # input vector
paa = int     # number of outputs in the output vector

if Y.shape[0] % paa != 0:
    valuespace= np.arange(0, Y.shape[0] * paa)
    output_index = value_space // Y.shape[0]
    input_index = value_space // paa
    uniques, nUniques = np.unique(output_index, return_counts=True)
    res = [input[indices].sum() / input.shape[0] for indices in
           np.split(input_index, nUniques.cumsum())[:-1]]
```

Figure 4.3: Snippet code of PAA,taken from [28]

the dataset. The first step is to construct the total space through the multiplication of the size of the input array Y with the necessary output frame X. Once the PAA is completely applied on whole dataset, time series is converted into equi-sized frames as shown in Figure [4.4].

Figure 4.4: Dimensionality Reduction by PAA



4.6 Distance Functions

PAA transforms each time series segment into a string. Proposed approach stores each string along with index and the data points belong to that segment in a matrix. Now our goal is to further narrow down these strings (theoretically patterns) to get more precise results. To achieve this we need a distance function is required to measure the distance between pair of segment data points that belong to same string (can be assumed as pattern).

The goal of time series analysis processes is to build a distance measurement between two time series data. In the context of this thesis work, distance measurement is calculated between two sub-sequences (segments). Usually, the similarity or dissimilarity of two sub-sequences is determined by transforming the data into vectors and measuring distance between the vector space points. There are two main state-of-the-art distance functions.

4.6.1 Dynamic time warping (DTW

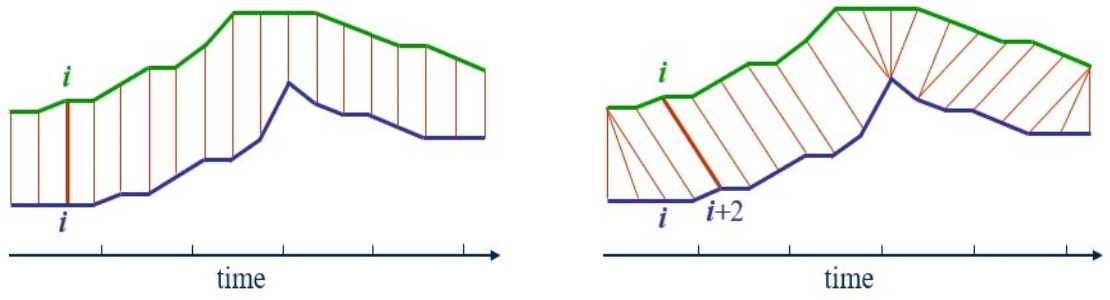
Dynamic time warping (DTW) is an algorithm [34] that compares the similarities between two time sub-sequences, those two sub-sequences may vary in length or speed . Originally, the algorithm was designed to be used for speech recognition, but it is also a good solution to time series problems. It is a good way to find the optimal path between two sequences. Assume two time sub-sequences X and Y. DTW compares one point in X data point in one sub-sequence with any points in Y sub-sequence. Even if X and Y are of different lengths DTW distance can be calculated. But in our approach each sub-sequence is of equal lengths. Finally, every warp route fits the sub-sequences X and Y, so that all points at least one point from a different time series matches. One main point remember is that DTW is a distance quantity and can not be used as a metric.It is an effective distance measurement function used in time series similarity search.

4.6.2 Euclidean Distance

Euclidean Distance is an algorithm that compares the similarities between two time sub-sequences, but those two sequences have to be in sync i.e they must not vary in length or speed.

4.6.3 Why DTW is better than Euclidean Distance Function?

For comparison Figure [4.5a], it is possible to calculate Euclidean distances function from the two sequences. Here the i-th point of the one-time series have to be associated with the i-th point of the second. If both sequences are not in sync, Euclidean distance fails. Patterns with similar shapes but of different lengths and shapes will typically prevent a system from correctly recognizing patterns using Euclidean distance.



(a) Compare Time Series using Euclidean (b) Compare Time Series using DTW

Figure 4.5: Distance Function Comparison, referred from [19]

DTW provides a non-linear (elastic) Figure 4.5b, coordination between two time sub-sequences. The desired balance between the two time sub-sequences is specifically pursued. It results in a more straightforward calculation of similarity measure that allows similar shapes to align although even though they are out of sync in time. In this way DTW explores all possible paths,

Drawbacks of DTW

Even though DTW is better than Euclidean distance, it is not a perfect solution for dimensionality reduction as explained by [42]. DTW offers an excellent framework for measuring similarity. With regard to the statistical complicity, $O(M^2)$ is the best match for two patterns of M-length to the DTW set. However, in the case of a query pattern the squared time complexity would become a large computational burden, if DTW is used to search for similar patterns in a large data set. For examples, DTW would take the time $O(M^2.N)$ if N were to identify the best fit of a given pattern of length M in a large data set of N objects.

This is the major reason that DTW is not suitable for large dataset. DTW produces large number of comparisons. Each segment is compared with all other remaining segments to calculate distance and produce distance matrix. This results in high time complexity.

To overcome this problem PAA is used in our approach to reduce number comparisons. Once there are same or similar patterns grouped together by PAA, comparison size will be less. Now DTW is applied to each group of patterns to produce rank table based on the lowest DTW value.

Chapter 5

Implementation of the Prototype

In this chapter the implementation of proof-of-concept discussed in the previous chapter is explained. A working prototype of similarity search using dimensionality reduction technique is developed. However, the architecture is implemented in a modular way such that the high-performance modules can replace the existing ones. This prototype is completely implemented using python.

Proposed Multi-step approach consists of six sub steps as shown in Figure[5.1]. In this chapter each steps are explained along with code snippets.

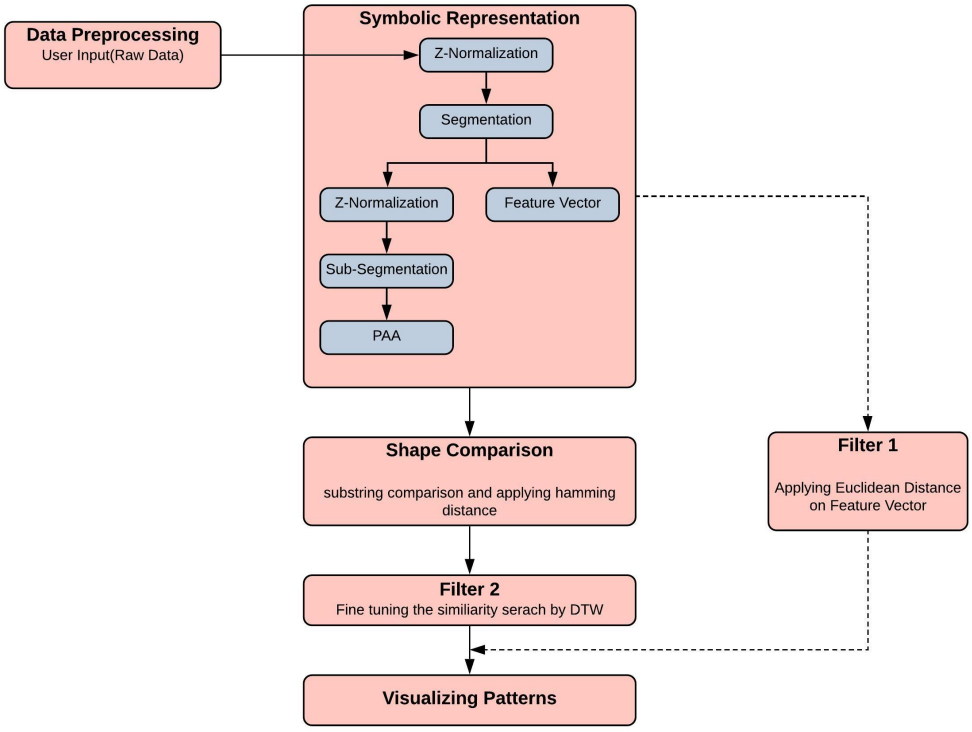
5.1 Data Pre-processing

User will provide the raw time series data as input file to the system. This data is pre-processed to remove unwanted values such as NULL or NaN values. Mean or Median is calculated from the non-missing data to impute the missing data values.

5.2 Symbolic Representation

As explained in initial section handling the missing data is important step of our approach. In the same way data needs to be represented in suitable way to make pattern discovery task easy. Symbolic representation is a widely used data representation method for time series, which is based on transforming the numerical form of a time series into a sequence of discrete symbols under the prescribed mapping guidelines. It minimizes data dimensionality using PAA. The SAX representation thus accelerates the process of similarity search for time series data while maintaining the accuracy and performance.

Figure 5.1: Overall Architecture of proposed approach



5.2.1 Notations

Before proceeding to the implementation of Symbolic representation architecture, few notation needs to be defined [29] to understand the model.

Definition 1. ***Time Series:*** *A time series T of length n, T = t1,.....,tn, is an ordered set of n real-values.*

The bulk of the data of the time series with real time stamps is very long. We focus on local features of time series data, which means dividing sequence by sequence instead of the whole series. It is more interesting to find out sub-sequence matches than the whole match, as hidden information interval by interval can be extracted. We define the term sliding window and time series sub-sequence to find out the relationship of the sub-sequence, which is derived using the sliding window.

Definition 2. ***Sub-sequence:*** *Given a time series T of length n, a sub-sequence S of length m of T, in which $m \leq n$ starts at p, that is $S = t_p, ..., t_(p + m - 1)$ for $1 \leq p \leq -m + 1$, a contiguous location starts at p.*

Definition 3. ***Word:*** *Every sliding window sub-sequence is separated into segments (w) with each*

being an alphabetic symbol. Then a number of alphabetic symbols, which arrange a sliding window sequence, are called a word. The number of segments in a sub sequence is therefore proportional to word size, i.e. alphabet numbers in a word.

There are two more additional parameters that required for symbolic representation.They are alphabet size (a) and window size (w). During segmentation process using sliding window time series is divided into equal-sized segments. This size is indicated by window size (w). PAA segments that are converted from the original time sequence. The alphabet size (a) is the number of alphabets that map the PAA values to the alphabet symbols depending on the break point lookup table obtained from the equal-sized areas under the Gaussian curve.

Before explaining proposed symbolic representation technique, it is significant to understand how original SAX works and how our approach is different from original SAX.

The original SAX transforms Figure[5.2] a time series X of length n into the string of arbitrary length ω, where $\omega \leq$ n typically, using an alphabet A of size a ≥ 2. Initially, it Z-normalizes time series and then divides time series into smaller sub-sequences. Later it transforms these sub-sequences into the PAA representation and finally, converts the PAA data into a string.

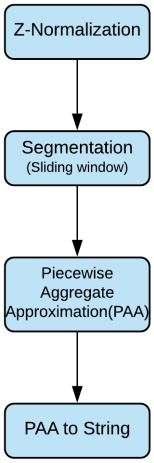


Figure 5.2: Architecture of original SAX

Our Symbolic Representation approach contains following steps:

5.2.2 Z-Normalization

System collects the user input file and now data is z-normalized. It is proved by the researchers [21] [29] that values of z-normalized time series pursue the normal distribution. As briefed in [37], Z-normalization is the process of normalizing the data to zero mean and zero unit of energy which is to say that the mean is 0 and the standard deviation is approximately 1.

First, the mean time series is subtracted from the true values, and second, the standard deviation value divides the difference. Most of the recent work on structural design mining in time series shows that Z-normalization is a crucial preparatory step that enables the mining algorithm not to concentrate on amplitude-driven but on structural similarities. Z-score normalization is a data standardization technique that avoids this issue. The formula for Z-score normalization is below:

$$z = \frac{x - \mu}{\sigma}$$

In this sense μ is the mean feature value and $\sigma$is the standard feature deviation. If a value is precisely the same as the mean of all the feature values, it will be normalized to 0. It's going to be a negative number if it's below the mean, and if it's above the mean it's going to be a positive number. The size of such negative and positive numbers is calculated by the standard deviation. If there is a large standard deviation in the non normalized numbers, the normalized values will be closer to 0. Hence, it's possible to pick an equivalent region under the Normal curve for cutting co-ordinates (y co-ordinates),and cutting the area under the Gaussian curve.

5.2.3 Segmentation

Normalized data is segmented into smaller sub-sequences each of length n using sliding window technique. While applying sliding window technique for sub-sequence extraction, we need to store all extracted sub-sequences for further manipulation.

```
def sliding_window(timeSeries, stepSize, windowSize):
    # slide a window across the time series
    for i in range(0, timeSeriesLength):
        segments =  timeSeries[curr_count:(curr_count+window_size)]  # the current window
        curr_count = curr_count + stepSize
```

Figure 5.3: Snippet code of Sliding Window

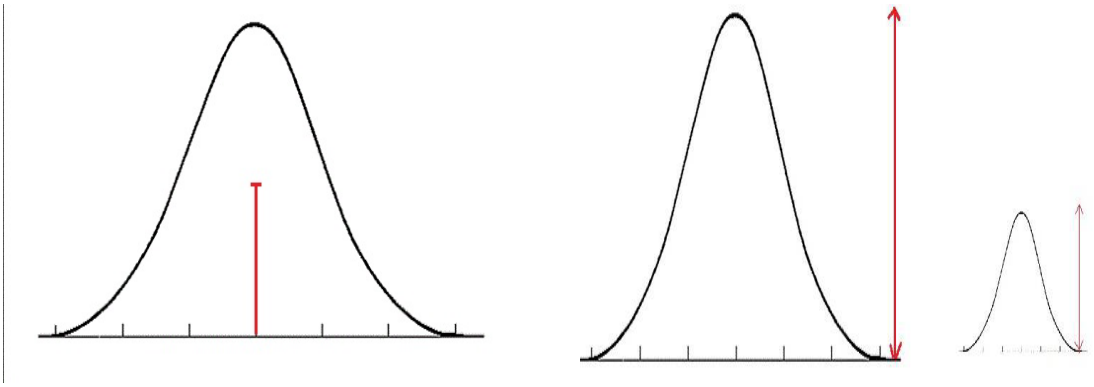
The *sliding_window* method takes three arguments. The first is the time series that we are going to loop over. The second argument is the *stepSize*. The *stepSize* indicates how many data points we are going to "skip". Normally, we would not want to loop over each and every data point of the time series (i.e. $stepSize = 1$) as this would be computationally costly. Generally *stepSize* is set to 10% of the actual length of the time series.

5.2.4 Feature Vector

The data collected is after the normalization, sometimes may not provide the right estimators. It is important to think about data and understand whether the applied normalization match the results we are looking for and provide the expected results. So in that sense it is important preserve the features we might loose in further steps. It is essential to keep features such as ***Scale*** or ***offset*** between the data keep it in the data set.

Since the sub-sequences are again normalized [at step 5.3.4] hence we lose some important features of these sequences in later step. It is important to store features Figure[5.4] of these sub-sequences. Hence the feature Vector is generated for that reason.

For each sub-sequence, mean value of all the data points in that sub-sequence is calculated and stored as ***offset*** as shown in Figure [5.4a] and subtracted value of highest and lowest data points of each sub-sequence is stored as ***scale*** as shown in Figure [5.4b] in the Feature vector. This feature vector optionally acts as filter. Using these position and/or size, patterns can be optionally filtered out to provide more precise pattern matching.



(a) Offset (b) Scale

Figure 5.4: Features

5.2.5 Z-Normalization

Now sub segments are again Z-normalized. The reason behind this Z-normalizing is to scale down all the patterns to the same level. Normalization makes the conditioning less vulnerable to the scale of the characteristics, so that coefficients can be better solved.

Normalization approach will enhance overall multi-step model approach. In addition, if we don't apply this normalization it would be difficult (probably not possible) to match the patterns due to the problems of scaling. Z-Normalization increases the speed of pattern matching. Normalizing guarantees that there is no huge variation in a pattern matching problem, which allows optimization.

5.2.6 Sub-Segmentation

The commonly encountered issue in many classic similarity search approaches is that finding meaningful motifs. In order to avoid missing out any patterns, segmented data is once more segmented into smaller sub segments of x co-ordinate string length s. While applying sliding window technique for sub-sequence extraction, we need to store all extracted sub-sequences for further manipulation.

5.2.7 Piecewise Aggregate Approximation (PAA)

Once the data is again segmented into smaller sub-sequences of size s, each sub-sequence is transformed into a symbol using PAA. PAA is symbolic representation technique. As explained in

	3	**4**	**5**	**6**	**7**	**8**	**9**	**10**
$\beta 1$	-0.43	-0.67	-0.84	-0.97	-1.07	-1.15	-1.22	-1.28
$\beta 2$	0.43	0	-0.25	-0.43	-0.57	-0.67	-0.76	-0.84
$\beta 3$	-	0.67	0.25	0	-0.18	-0.32	-0.43	-0.52
$\beta 4$	-	-	0.84	0.43	0.18	0	-0.14	-0.25
$\beta 5$	-	-	-	0.97	0.57	0.32	0.14	0
$\beta 6$	-	-	-	-	1.07	0.67	0.43	0.25
$\beta 7$	-	-	-	-	-	1.15	0.76	0.52
$\beta 8$	-	-	-	-	-	-	1.22	0.84
$\beta 9$	-	-	-	-	-	-	-	1.28

Table 5.1: The look-up table[29] for alphabet size 3 that contains the breakpoints that partitions a Gaussian distribution in an arbitrary number

Piecewise Aggregate Approximation section, PAA helps in dimensionality reduction by reducing comparison size. For each sub-sequence mean values is calculated after the application of the PAA transformation. Numerous symbols are called alphabets of alphabet size a. The next step is to distinguish these mean values. The first ten letters of the classical Latin alphabet are used in this thesis.

It is important to discretize the time series in such a way that each symbol has the same probability of occurring and also has a Gaussian distribution after normalization of time series which allows the right intervals to be established for discretion. Such intervals are characterized by breakpoints [29] $B = \beta_1, \beta_2, \beta_3, ..., \beta_{a-1}$. The region under a Gaussian curve between β_i and β_{i+1} must be $\frac{1}{a}$. The first interval is obviously from $-\infty$ to β_1 and the last one from β_{a-1} to $+\infty$. In a statistical table, the precise values of alphabet size a are shown. The values for all letter sizes between 3 and 10 are shown in Table [5.1].

The PAA-transformation now converts all the mean values into the symbol that fits the interval. Thus the first symbol in the alphabet ,(here it is a)now represents all points of the original time series whose mean value of the windows is under $\beta 1$. All points represented by mean $\beta 1$ to $\beta 2$ values are now represented by the second symbol b, and so on. The resulting symbol sequence is called word which is of length w, obviously.

5.3 Shape Comparison Algorithm

At the end of PAA, PAA transforms each time series segment into a string. All these segments are stored along with their index (position of the respective segment in the time series). Each string is compared against other remaining strings. Matched strings are grouped and stored in matrix along with their indices. Theoretically, all the members in each group will be having the same shape.

Optionally, hamming distance is used to avoid the loss of true positive patterns.

As clearly stated in the code snippet [5.8] , two string arguments of the same length would have been given as input to *hamming_distance* function. The strings obtained must be of the same

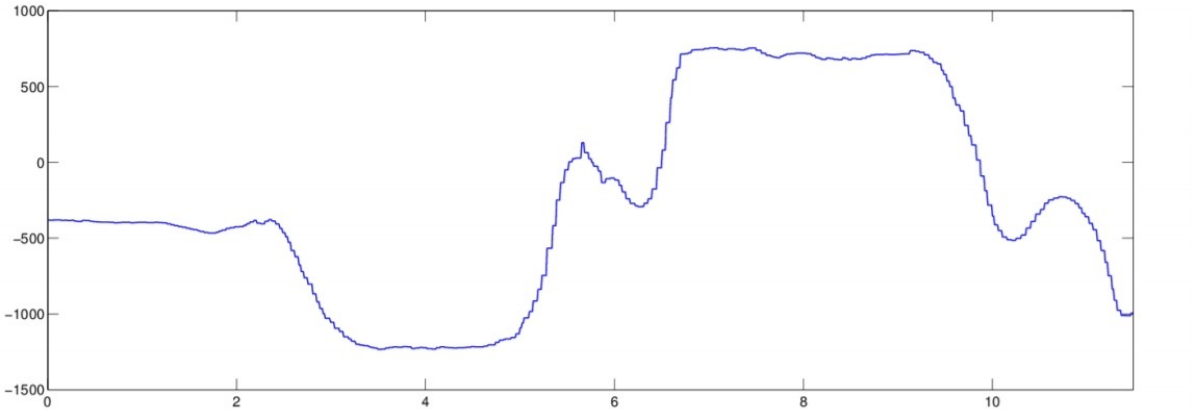


Figure 5.5: The originally time series. referred from [20]

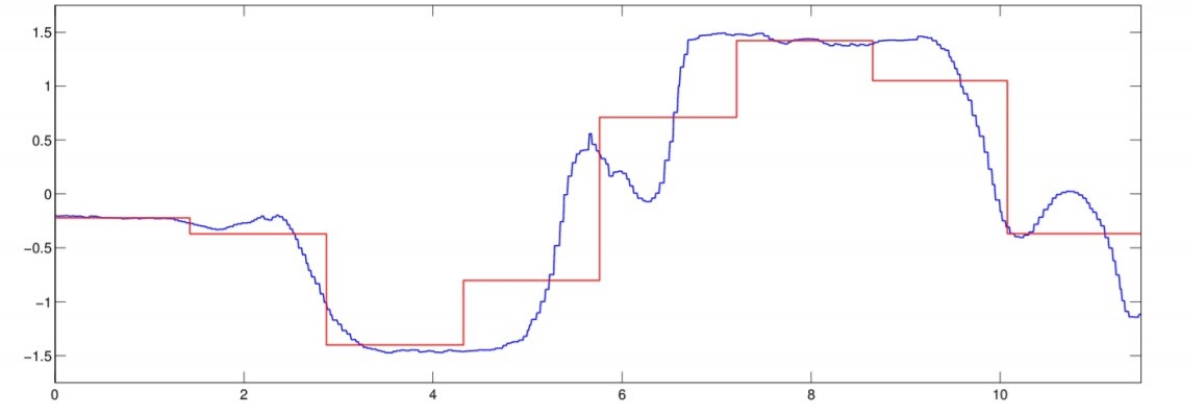


Figure 5.6: The time series of Figure [5.5] normalised and represented by PAA with w = 8 windows.

size. So before comparing each strings, it is important make sure that both strings are of same length. If they are not of same length this function fails. Next, need to compare every character in the first string with the character holding the corresponding position in the second string. This helps to identify the points where the differences exist, because the total number of these instances gives the hamming distance between two strings.

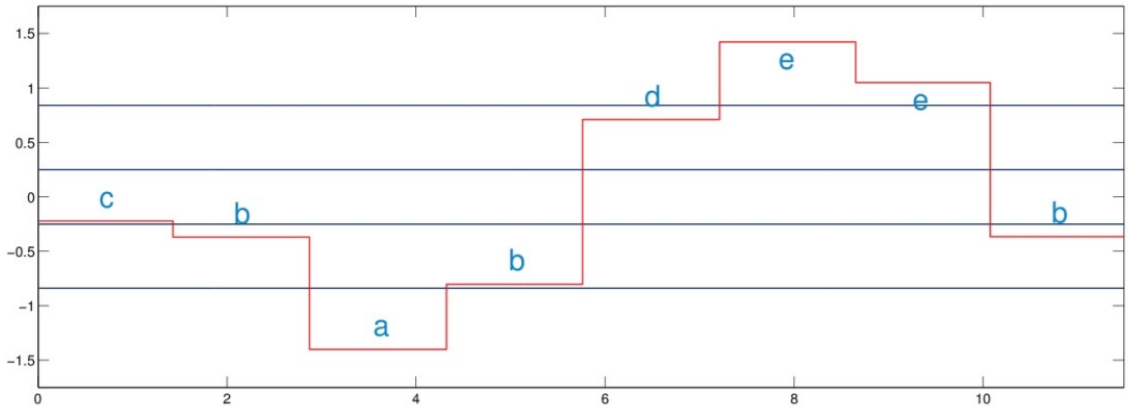


Figure 5.7: This plot shows the breakpoints and the discretisation by SAX of the time series in Figure [5.6], referred from [20]. As a result the time series is represented by the word 'cbabdeeb' using an alphabet size of a = 5.

```
def hamming_distance(str1, str2):
    distance = 0
    for ch1, ch2 in zip(str1, str2):
        if ch1 != ch2:
            distance += 1
    return distance
```

Figure 5.8: Snippet code of Hamming Distance

5.4 Filter 1: Feature Comparison by Euclidean Distance

Optionally using scale or offset from feature vector to provide more accurate results such as sub-class categorisation.

After the initial normalization and segmentation we might miss some vital information regarding the shape of the patterns. So to avoid that, we generated feature vector to store ***offset*** and ***scale*** as features related to the pattern. Feature vector is proposed to achieve both accuracy and efficiency. Our idea is for the pattern matching along with the reduced time complexity in the discovery of variable length patterns, so that the original time series data is initially transformed into symbols. *Shape comparison algorithm* produces a matrix with each string along with its index. This results in a positional index with a feature filter algorithm. The algorithm first locates the patterns along the transformed data based on the strings. Then it is used to filter out the patterns based on features like Scale and Offset values for each pattern.

The feature vector is used to sub-classify the time series data based on features. Our goal is to achieve higher classification accuracy with shorter computation time, we adopt DTW with the

occurrence frequency of candidate patterns to rank the patterns with the help of features.

5.5 Filter 2: Fine Tuning the Similarity Search using Dynamic Time Warping

As we described disadvantages of DTW in **??** section, DTW would take the time $O(M^2.N)$ if N were to identify the best fit of a given pattern of length M in a large data set of N objects. In this thesis we resolve the drawback by using a PAA. PAA is seen as a down sample method that can deliver as symbolic representative for a long time series. In comparison to DTW, PAA produces lesser comparisons. Hence, computations save time with the use of their equivalent PAA representation.

Finally, Dynamic time warping is applied on each pattern groups obtained from compare shape algorithm to further narrow down the results and getting only precise sub-sequences. After the Filter 1 step, same length patterns have already collected. As our challenge is to find out the significant patterns and frequently occurring patterns between them, the well-known distance function called Dynamic Time Warping (DTW) [3] is used to calculate similarity between patterns of same lengths. Here, we describe significant patterns as regularly occurring time series patterns whose frequency is greater and the similarity between a set of time series in the same pattern is strong.

Each pair of patterns belonging to the same string i.e same class of patterns compared against are assigned with a DTW-distance value. In the context of this thesis work, this DTW-distance value is considered as rank for those patterns. Patterns having lower DTW-distance value will have the higher ranks in the rank table. At the end of this step, a rank table will be generated for all the patterns ordered based on the DTW value.

5.6 Visualization

All the patterns obtained from the previous step are visualized with respect to the rank table. Optionally user can select patterns based on features such as offset or scale to get more precise patterns.

Chapter 6

Evaluation

In the evaluation work,

- Synthetic data generation plan is discussed for accuracy evaluation in comparison among proposed approach and Exhaustive Pattern Search by DTW .
- Computation time or performance evaluation in comparison among two different existing methods with different UCR datasets.
- Class and Sub-class categorization are used as accuracy metrics.
- Using brute force algorithm, Sub-sequence are captured.Sub-sequences in the data sets are compared against Exhaustive Search using Dynamic Time Warping and the proposed approach.
- Compare the results with our proposed method with feature vector proposed method without feature vector and previously proposed state-of-the-art method in classification and efficiency.

6.1 Datasets

In comparison with the Exhaustive Pattern Search Using DTW method, we use three different data sets for evaluating the accuracy and efficiency of our proposed method. Table 5.1 displays their features, as well as their preferred window sizes, as shown in Table 6.1. For symbolic representation, our approach required two additional parameters: word size (w) and symbol size(a). Then the randomly generated synthetic data is used for further evaluation.

6.1.1 Synthetic Data

For evaluation purpose synthetic data is generated. For each instances of the evaluation two types of synthetic datasets are generated randomly. One for the pattern classification analysis and an-

No	Dataset	Length	WindowSize	Description
1	UCR [12]	15-6000	Varies	UCR Time Series Data
2	ECG	120	20	Electrocardiogram data
3	Synthetic Data	1000 - 100,000	1000	Randomly generated synthetic data

Table 6.1: Selected data sets attributes.

other pattern sub-classification analysis as explained below.

6.1.1.1 Normalized Dataset

Normalized random dataset is generated [6.1] which includes only noise without any features Scale and Offset.

Classes /Patterns: For predefined patterns each of size n data is generated and added to the dataset. These patterns are added randomly.

Noise: For each pattern above may or may not be added noise depending on the random number.

```
def data_generate(rand_class,rand_noise):
    rand_class = randrange(8)
    rand_noise = randrange(2)
```

Figure 6.1: Snippet code of Normalized dataset generation

Depending on the random number generated for the rand_class , that pattern/class data is generated. If the generated random noise (rand_noise) is 1 , noise is added to the above else noise not added i.e above data is kept same.

The complete dataset of size 'x' is generated, each pattern and their respective class is noted/stored.

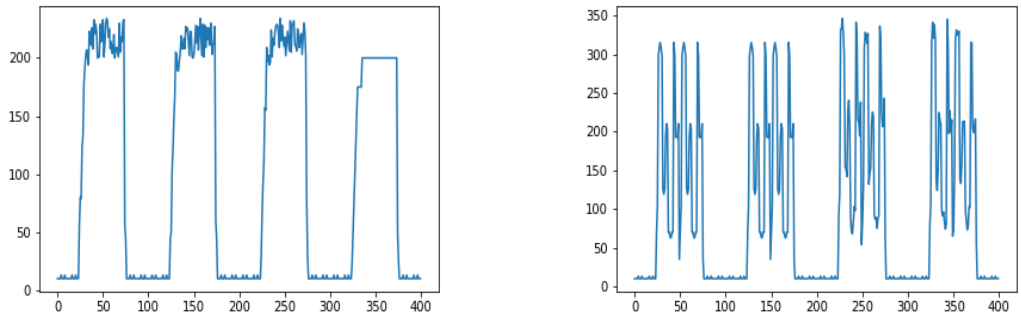
6.1.1.2 Non-Normalized Dataset

Non-Normalized random dataset is generated [6.3] which includes only aswell as features, Scale and Offset.

Classes /Patterns: For predefined patterns each of size "n" data is generated and added to the dataset. These patterns are added randomly.

Noise: For each pattern above may or may not be added noise depending on the random number.

Scale: Each above pattern may or may not be upscaled by 3 times depending on the random number.



(a) Class 1 (b) Class 2

Figure 6.2: Normalized synthetic data of different classes

Offset: Offset of each above pattern may or may not be increased depending on the random number.

```
def data_generate(rand_class,rand_noise):
    rand_class = randrange(8)
    rand_noise = randrange(2)
    random_offset = randrange(2)
    random_scale = randrange(2)
```

Figure 6.3: Snippet code of Non-Normalized dataset generation

Depending on the random number generated for the rand_class , that pattern/class data is generated .

- if the generated random noise(rand_noise) is 1 , noise is added to the above else noise not added i.e above data is kept same.
- if the generated random scale(rand_scale) is 1 , above class data is upscaled (multiplied by 3 times) else class above data is kept same.
- if the generated random scale(rand_scale) is 1 , above class data position(mean) increased else above class data is kept same.

The complete dataset of size 'x' is generated, each pattern and their respective class and sub-class is noted/stored.

6.2 Exhaustive Pattern Search by DTW (EPS-DTW)

Dynamic time warping (DTW) is widely used for sub-sequence matching. Initially the large dataset is divided into sub-sequences using same sliding window technique used in proposed approach. Then

DTW distance measured between first sub-sequence and remaining all sub-sequences. For each pair (first and other sub-sequence) of these sub-sequences DTW distance assigned and stored in a matrix. This process continues for second sub-sequence and remaining all other sub-sequences. And so on. Theoretically, pair having lower DTW distance are most likely to be similar pattern. This matrix is sorted on ascending order of the DTW value. Top-n sequences from these matrix are most occurring patterns.

6.3 Accuracy Evaluation

In this segment, we provide the task of classifying time series accuracy by using the patterns discovery as a metric. Nonetheless, We do not comply with the frequency of occurrence of the patterns for classification in time series.

6.3.1 Similarity Search on Synthetic Data

Normalized and Non-normalized data set is generated to evaluate the accuracy among the methods. Initial step is to find top-k patterns from the large set of patterns based on the pattern/segment data collected by the evaluator. As explained in the beginning of the evaluation process, Evaluator has the better understanding of the patterns and class in which each pattern belong. Top-K is calculated as :

$$^{n}C_{2}$$

Where,n is total number of patterns/segments belong to particular class, C is the class for which patterns belong. Different number of k-motifs are selected on different datasets in the evaluation.

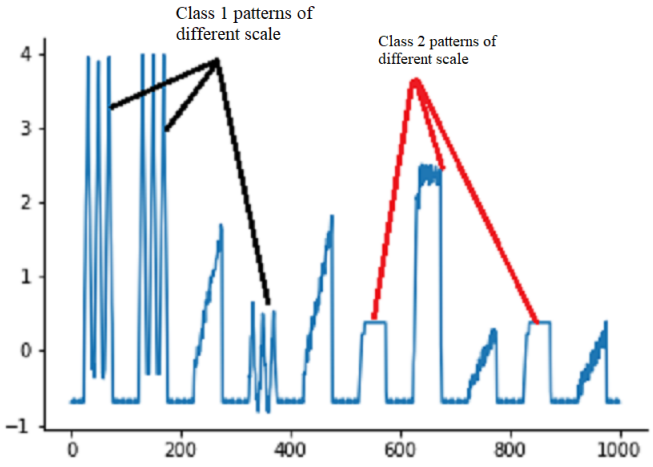


Figure 6.4: Non-Normalized synthetic data.

The similarity distance between the patterns is then determined and order them in ascending. As the distance value is smaller, the more similar, the top one distances must be taken into account and the relevant patterns should be identified as significant patterns between top-k similar patterns.

6.3.2 Carried Out Evaluation Plan

The Synthetic data is discretized using word size 4 and symbol size 5 with the sliding window length 100 for different length of data sets.

6.3.2.1 Class Accuracy on Normalized data :

For Exhaustive Pattern Search Using DTW:

Top 'k' pair sub-sequences with lowest values obtained from the EPS-DTW matrix method are matched against the synthetic data(segment-class) table. If the top 'k' DTW pair sub-sequences which are matched with synthetic segments data, they are true positives, else patterns false positives. Remaining all the sub-sequences found outside the top-k from EPS-DTW matrix (i.e. excluding top 'k' patterns) are true negatives.

For Proposed approach without feature parameters (Scale and Offset):

Proposed approach classify each sub-sequences to their respective classwise (patternwise) and prepares a matrix for all pair of sub-sequences. If the proposed segment-class combination matched with top-k synthetic pair (sub-sequences-class), then its true positives ,else false positives. Remaining all the sub-sequences from proposed approach matrix are true negatives.

Since the normalized data doesn't contain any sub-classes (patterns without either Scale or offset), only class accuracy measurement is carried out for normalized data.

6.3.2.2 Sub-Class Accuracy on Normalized data

For Proposed approach with feature parameters (Scale and Offset):

In this section, proposed approach classify sub-sequences based on the scale/offset value obtained from feature vector. This method has scale and offset values form feature vector to compare against the synthetic data. Proposed approach classify each sub-sequences to their respective classwise(patternwise) as well as sub-classwise and prepares a matrix for all the pair of sub-sequences. If the proposed pair (sub-sequences-SubClass) combination matched with top-k synthetic meta data

pair (sub-sequences-subClass) combination then its true positives, else flase positives. Remaining all the sub-sequences outside the top-k from proposed approach matrix are true negatives.

For EPS-DTW:

Since to tackle with the sub-class pattern categorization problem, in proposed approach extra filter or refinement step is added for this particular reason. Exhaustive pattern discovery method doesn't have that filter, it can't do the sub-class pattern categorization.

We classify 6 synthetic time series datasets with baseline methods and report the results in Table 6.2. We highlighted in the highest true positive (pattern classification) rate for each distance measure in that table. We can see that true positive (pattern classification) rate for proposed approach is same as for EPS-DTW. The lowest True positives mostly occurs due to the small size dataset. As the dataset size increased, there is hardly any true positive gap between proposed and EPS-DTW. Even though both approach captures similar amount of Top-K patterns, EPS-DTW has the high time complexity compared to proposed approach. It is discussed in detail in Performance Evaluation section.

Dataset Size	**Top K Value**	**True Positives**	
		DTW	**Proposed**
1,000	18	18	18
2,000	43	43	43
5,000	301	299	**299**
10,000	1,226	1,217	**1,218**
25,000	7,773	7,757	**7,758**
100,000	62,693	62,663	**62,665**

Table 6.2: True positives for Synthetic data.

Since the true positives rate is same in both approach ,it directly impacts on the false positives. As highlighted in Table 6.3 , false positives are significantly lower for proposed approach. Since proposed approach uses dimensionality reduction technique, it has the lower true negatives as compared to DTW. That means DTW produces more patterns than proposed approach which may not be useful or overlapping patterns. It results in high time complexity.

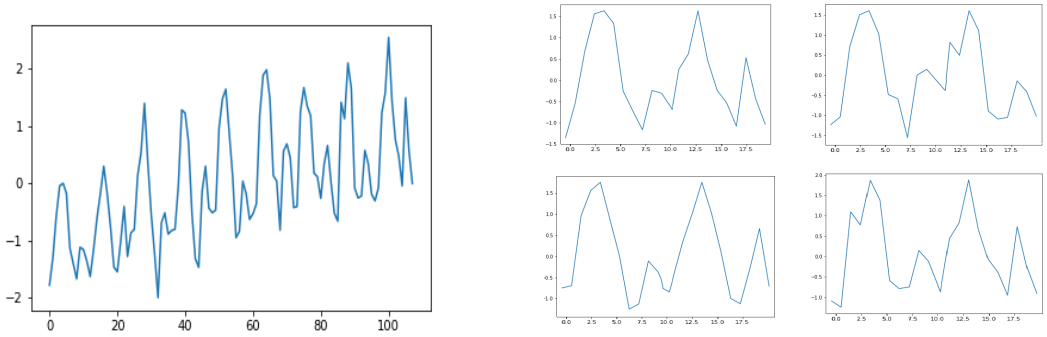
Dataset Size	**False Positives DTW**	**False Positives Proposed**	**True Negatives DTW**	**True Negatives Proposed**
1,000	0	0	87	**9**
2,000	0	0	190	**21**
5,000	6	**2**	4950	**88**
10,000	19	**8**	4950	**195**
25,000	61	**15**	31125	**656**
100,000	150	**46**	461594	**2367**

Table 6.3: False positives and True Negatives for Synthetic data.

6.3.3 Similarity Search on ECG data

ECG data Figure [6.5a] with window of length 20 was discretized using word size of 3 and symbol size of 4. Our proposed method successfully categorize top motifs of different length which is illustrated in Figure [6.5]. There would be a question why the locations of discovered motifs from our method and existing methods are different and which one is significant for the domain users. Our method found out the different lengths of motifs when we have selected top-2 motifs ordered by their similarity calculated by DTW distance.

The execution time of our motif discovery method is 0.122 sec, while EPS-DTW required 1.2 sec. As the domain knowledge referenced from [12], the beginning part of the ECG dataset are calibration signals which are the initial settings of ECG apparatus and not detecting a biological signal of a human. Domain experts would like to find out the actual patient data not like noisy data. Fixed length of motif discovery method is difficult to deal with that situation without providing domain knowledge.



(a) Original ECG Data

(b) Classified Patterns

Figure 6.5: Pattern Search on ECG data

6.4 Performance Evaluation:

For the computational efficiency comparison, we compared our similarity search approach measurement time with the Exhaustive Search using DTW with the following environment and settings:

- The experiment is carried out on a machine running on the 64 bit operating system Windows, using a 2.20GHz core i7 processor and an 8 GB RAM system.
- The run time is measured from the receiving of the dataset to the output of the search result. In this case, the time of execution consists of both the time of transformation of the dataset and the time of classification based on the distance measure.
- The segment size (w) and skipOffset is kept same on its time series .

Analysis of the number of comparisons produced by the methods as shown in 6.6 of the synthetic datasets of different sizes ranging from 10k to 100k with same (w) segment sizes. Proposed approach produces less comparisons as opposed to Exhaustive Search using Dynamic Time Warping. Comparisons are directly proportional to the execution time i.e as comparison size increases execution time increases which affects the performance of the system. This is beacuse our proposed method achieves the dimension reduction ratio. It can achieve reduced dimension about 40% of original data.

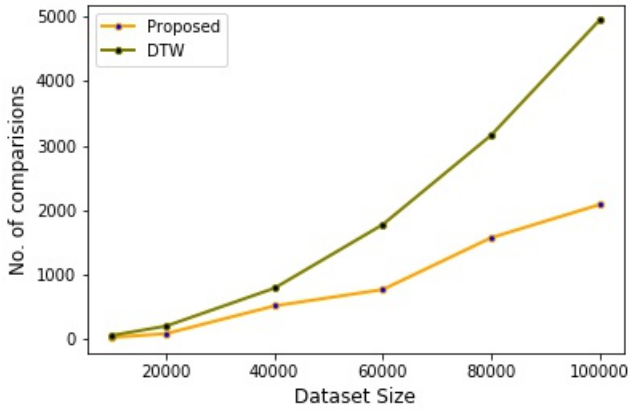


Figure 6.6: Analysis of the methods in no. of comparisons produced on time series of different length.

We compare the execution time shown in Figure 6.7 of the datasets with different length and same segment size(w) and a fixed symbol size (a=5). As we noticed in figure, when the dataset size becomes larger, the execution time takes longer on each distance measure. Besides, among the methods, proposed approach has the lowest computation time because of dimensionality reduction. Exhaustive Pattern discovery by DTW method has the higher execution time for its representation preciseness. Even though for smaller datset, execution time between the methods was not that noticeable, but when the datset size increased proposed approach takes 4 time less execution time.

We carried out performance evaluation on proposed approach for 1 million dataset with *window size* = 2000. Proposed approach completed the pattern discovery task and rank patterns based on the frequency of occurrences about 4 times less than that of EPS-DTW.

Figure 6.8 With the help of feature vector, similarity search based approach worked on symbolic representation. We analyzed the computational speed on 7-UCR benchmark data sets [12]. The proposed approach with feature vector and rank table on frequent patterns increases the classification speed. On the other hand pattern matching with DTW seemingly takes more computational time.

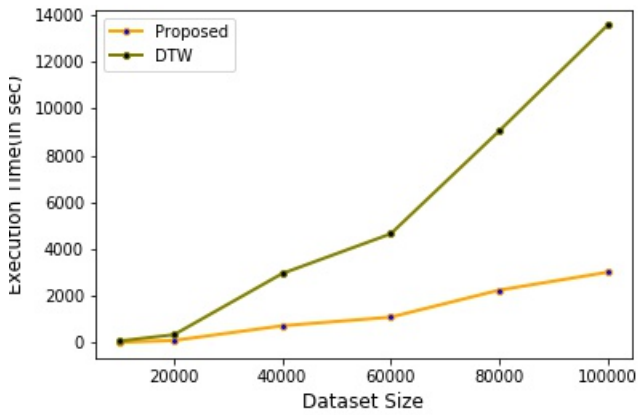


Figure 6.7: Comparison of methods in execution time taken for Similarity search on random synthetic time series of different size.

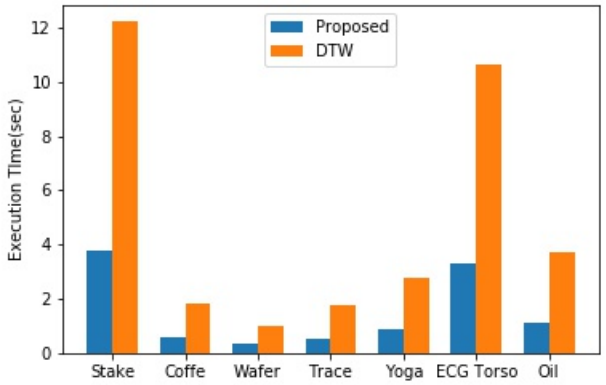


Figure 6.8: Similarity search speed in (sec) for 7-UCR data sets [12] for proposed approach and DTW.

6.5 Evaluation Summary

We presented the approach for pattern discovery with a feature vector and evaluated its classification speed and accuracy with and without features. The proposed discovery method works well for most data sets when the experimental assessment is performed, reducing the number of patterns up to 40% and improving the classification speed. In the mean time, classification accuracy of patterns with feature vector is statistically significant for sub-class categorization.

There is no trade-off between classification accuracy and dimensionality reduction. Time-series

data compression doesn't affect the classification accuracy and it does not drop. Even if sometimes accuracy is dropped, because of the significant features of original data tend to disappear, feature Vector helps in capturing such features. We preserve the original data along with classification accuracy is also preserved well. The proposed approach attains its classification accuracy at certain level with reasonable dimension reduction ratio.

Chapter 7

Conclusion

The secret to further time series mining tasks like clustering, classification, anomaly detection, etc. is an effective technique in the time series data representation. A wide variety of applications in financial assessment, weather forecasting, medical data evaluation, multimedia systems, monitoring and management of resources, network traffic analysis, marketing and social research are subject to the knowledge of motif patterns.

There are two types of the main challenges associated with time series results. Firstly, efficient data representation technology should be developed that reduces the high dimensionality of time series data. The second is how useful hidden patterns can be extracted efficiently from huge amounts of time series data for further mining time series. In addition, data representation require to maintains important data features for additional time series analysis tasks.

We developed an algorithm to discover motifs of variable lengths and to prune non-overlapping patterns in the time series using multi-step approach. Our goal was to find patterns with different lengths through low computational time, so the original real-time time series was transformed into strings. Patterns of different length are discovered by symbolic representation using PAA and patterns are ranked with DTW distance on original real valued time series. After the patterns discovery , the filtering of overlapping of of performed for the original time series, to meet the user expectations and evaluate relevant patterns. In addition, we analyzed motifs as feature vectors for the time series classification task.

The experimental findings on different data sets demonstrate that in contrast to state-of-the-art approach, our proposed approach finds out significant motifs as well as anomalies and rank them efficiently.

Chapter 8

Future Work

For the pattern detection with the help of feature vector parameters , the followings are future work directions.

- Further evaluation on even large numbers of different data sets on the proposed approach.
- To use our proposed methods for applying pattern discovery and anomaly-detection in multidimensional time series.
- Further analysis on motifs and their characteristics in various time series to evaluate their relevance for further time series analysis tasks with other discriminative measures.
- To encourage different domain users, a visualisation tool is provided for pattern detection.

Bibliography

[1] Rakesh Agrawal, Christos Faloutsos, and Arun Swami. Efficient similarity search in sequence databases. In *International conference on foundations of data organization and algorithms*, pages 69–84. Springer, 1993.

[2] Wolfgang Aigner, Silvia Miksch, Wolfgang Müller, Heidrun Schumann, and Christian Tominski. Visual methods for analyzing time-oriented data. *IEEE transactions on visualization and computer graphics*, 14(1):47–60, 2007.

[3] Donald J Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA, 1994.

[4] Alexis Bondu, Marc Boullé, and Antoine Cornuéjols. Symbolic representation of time series: A hierarchical coclustering formalization. In *International Workshop on Advanced Analysis and Learning on Temporal Data*, pages 3–16. Springer, 2015.

[5] Athman Bouguettaya, Qi Yu, Xumin Liu, Xiangmin Zhou, and Andy Song. Efficient agglomerative hierarchical clustering. *Expert Systems with Applications*, 42(5):2785–2797, 2015.

[6] Max Bramer and Vladan Devedic. *Artificial Intelligence Applications and Innovations*. Springer, 2004.

[7] Peter J Brockwell and Richard A Davis. *Introduction to time series and forecasting*. springer, 2016.

[8] Carmelo Cassisi, Placido Montalto, Marco Aliotta, Andrea Cannata, Alfredo Pulvirenti, et al. Similarity measures and dimensionality reduction techniques for time series data mining. *Advances in data mining knowledge discovery and applications*, pages 71–96, 2012.

[9] Kin-Pong Chan and Ada Wai-Chee Fu. Efficient time series matching by wavelets. In *Proceedings 15th International Conference on Data Engineering (Cat. No. 99CB36337)*, pages 126–133. IEEE, 1999.

[10] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Outlier detection: A survey. *ACM Computing Surveys*, 14:15, 2007.

[11] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15, 2009.

[12] Hoang Anh Dau, Anthony Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn Keogh. The ucr time series archive. *arXiv preprint arXiv:1810.07758*, 2018.

[13] Hoang Anh Dau and Eamonn Keogh. Matrix profile v: A generic technique to incorporate domain knowledge into motif discovery. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 125–134. ACM, 2017.

[14] Steve De Backer, Antoine Naud, and Paul Scheunders. Non-linear dimensionality reduction techniques for unsupervised feature extraction. *Pattern Recognition Letters*, 19(8):711–720, 1998.

[15] A Dharmarajan and T Velmurugan. Applications of partition based clustering algorithms: A survey. In *2013 IEEE International Conference on Computational Intelligence and Computing Research*, pages 1–5. IEEE, 2013.

[16] K SRIVASTAVA Durgesh and B Lekha. Data classification using support vector machine. *Journal of theoretical and applied information technology*, 12(1):1–7, 2010.

[17] Christos Faloutsos, Mudumbai Ranganathan, and Yannis Manolopoulos. *Fast subsequence matching in time-series databases*, volume 23. ACM, 1994.

[18] A Famili, Wei-Min Shen, Richard Weber, and Evangelos Simoudis. Data preprocessing and intelligent data analysis. *Intelligent data analysis*, 1(1):3–23, 1997.

[19] Duarte Folgado, Marília Barandas, Ricardo Matias, Rodrigo Martins, Miguel Carvalho, and Hugo Gamboa. Time alignment measurement for time series. *Pattern Recognition*, 81:268–279, 2018.

[20] Yifeng Gao and Jessica Lin. Efficient discovery of variable-length time series motifs with large length range in million scale time series. *arXiv preprint arXiv:1802.04883*, 2018.

[21] Dina Q Goldin and Paris C Kanellakis. On similarity queries for time-series data: constraint specification and implementation. In *International Conference on Principles and Practice of Constraint Programming*, pages 137–153. Springer, 1995.

[22] James Douglas Hamilton. *Time series analysis*, volume 2. Princeton university press Princeton, NJ, 1994.

[23] Victoria Hodge and Jim Austin. A survey of outlier detection methodologies. *Artificial intelligence review*, 22(2):85–126, 2004.

[24] Y Kumar Jain and Santosh Kumar Bhandare. Min max normalization based data perturbation method for privacy protection. *International Journal of Computer & Communication Technology*, 2(8):45–50, 2011.

[25] Rohit J Kate. Using dynamic time warping distances as features for improved time series classification. *Data Mining and Knowledge Discovery*, 30(2):283–312, 2016.

[26] Eamonn Keogh, Kaushik Chakrabarti, Michael Pazzani, and Sharad Mehrotra. Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and information Systems*, 3(3):263–286, 2001.

[27] Eamonn Keogh, Selina Chu, David Hart, and Michael Pazzani. Segmenting time series: A survey and novel approach. In *Data mining in time series databases*, pages 1–21. World Scientific, 2004.

[28] Vignesh Krish. Overview: Piecewise Aggregate Approximation.

[29] Jessica Lin, Eamonn Keogh, Stefano Lonardi, and Bill Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, pages 2–11. ACM, 2003.

[30] Mathias Lohne. The computational complexity of the fast fourier transform. 2017.

[31] Andrew McCallum, Kamal Nigam, and Lyle H Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 169–178. Citeseer, 2000.

[32] Steven A Morris, G Yen, Zheng Wu, and Benyam Asnake. Time line visualization of research fronts. *Journal of the American society for information science and technology*, 54(5):413–422, 2003.

[33] Yonghong Peng. A novel ensemble machine learning for robust microarray data classification. *Computers in Biology and Medicine*, 36(6):553–573, 2006.

[34] Joseph Picone. Fundamentals of speech recognition: A short course. *Institute for Signal and Information Processing, Mississippi State University*, 1996.

[35] Erhard Rahm and Hong Hai Do. Data cleaning: Problems and current approaches. *IEEE Data Eng. Bull.*, 23(4):3–13, 2000.

[36] Thanawin Rakthanmanon and Eamonn Keogh. Data mining a trillion time series subsequences under dynamic time warping. In *Twenty-Third International Joint Conference on Artificial Intelligence*, 2013.

[37] Pavel Senin and Sergey Malinchik. Sax-vsm: Interpretable time series classification using sax and vector space model. In *2013 IEEE 13th international conference on data mining*, pages 1175–1180. IEEE, 2013.

[38] Runfeng Tian. *An Enhanced Approach using Time Series Segmentation for Fault Detection of Semiconductor Manufacturing Process*. PhD thesis, University of Cincinnati, 2019.

[39] Cao Duy Truong and Duong Tuan Anh. A novel clustering-based method for time series motif discovery under time warping measure. *International Journal of Data Science and Analytics*, 4(2):113–126, 2017.

[40] Michail Vlachos, Carlotta Domeniconi, Dimitrios Gunopulos, George Kollios, and Nick Koudas. Non-linear dimensionality reduction techniques for classification and visualization. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 645–651. ACM, 2002.

[41] Rik Vullings, Bert De Vries, and Jan WM Bergmans. An adaptive kalman filter for ecg signal enhancement. *IEEE transactions on biomedical engineering*, 58(4):1094–1103, 2010.

[42] Yaodong Zhang and James Glass. A piecewise aggregate approximation lower-bound estimate for posteriorgram-based dynamic time warping. In *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

[43] Yunyue Zhu. *High performance data mining in time series: techniques and case studies*. PhD thesis, New York University, Graduate School of Arts and Science, 2004.

Glossary

discord In the context of this thesis work, Motifs can be defined as rarely occurring patterns of a longer time series.. 7

k-means Partitioning n observations into k clusters where each observation belongs to the nearest mean cluster.. 25

k-medoids k-medoids is selects data points as centers (medoids) in contrast to K-means.. 25

motif In the context of this thesis work, Motifs can be defined as frequently occurring repeated patterns of a longer time series.. 7