AUTOMATED RULE GENERATION FOR COMPLEX EVENT PROCESSING

DILUM CHANDIMA VITHANA

(158254P)

Degree of Master of Science

Department of Computer Science and Engineering

University of Moratuwa
Sri Lanka

February 2019

AUTOMATED RULE GENERATION FOR COMPLEX EVENT PROCESSING

DILUM CHANDIMA VITHANA

(158254P)

Thesis/Dissertation submitted in partial fulfillment of the requirements for the degree
Master of Science

Department of Computer Science and Engineering

University of Moratuwa
Sri Lanka

February 2019

DECLARATION

I declare that this is my own work and this thesis/dissertation does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also, I hereby grant the University of Moratuwa the non-exclusive right to reproduce and distribute my thesis/dissertation, in whole or in part in print, electronic or any other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Signature:……………………...   **Date:** ………………………

Name: D.C Vithana

The above candidate has carried out research for the Masters/MPhil/Ph.D. thesis/dissertation under my supervision.

Signature of the supervisor:…………………...   **Date:** ………………………

Name: Dr. Surangika Ranathunga

Abstract

The key concept of Complex Event Processing (CEP) is setting up accurate queries to pick up the important events. Since all CEP engines support SQL like queries, this rule setup requires some technical skills plus domain knowledge. The best solution to address this issue is to automate the query generation of CEP. Existing automated query generation methodologies are computationally expensive and are not fully automated processes. This study addresses the above two issues by proposing a shapelet based approach. This new approach is not computationally expensive, and it is a fully automated process with zero manual user intervention required. The proposed method uses the computationally efficient algorithm called Fast Shapelet Selection (FSS) algorithm. This FSS algorithm is used to extract the shapelets from data set. Then extracted shaplets are used to generate CEP queries. This proposed method can be used analyze to multivariant time series and this is more efficient than previously proposed shapelet based approaches.

Acknowledgements

First and foremost, I would like to thank my supervisor, Dr. Surangika Ranathunga for the valuable guidance given throughout to successfully complete this research. I would also like to thank Department of Computer Science and Engineering, Faculty of Engineering, University of Moratuwa Sri Lanka for their support.

Finally, I must express my very profound gratitude to my family for providing continuous support and encouragement.

TABLE OF CONTENTS

Contents

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AEP	Admissible Entropy Pruning
BF	Brute Force Algorithm
CEP	Complex Event Processing
DBMS	Database Management Systems
EDL	Event Description Language
FS	Fast Shapelet Discovery
FSS	Fast Shapelet Selection Algorithm
GRSF	Generalized Random Shapelet Forest
HMM	Hidden Markov Model (HMM)
IDP	Important Data Point
LS	Learning Shapelet Algorithm
nHMM	noise Hidden Markov Model
PLR	Piecewise Linear Representation (PLR)
PLR_IDP	Piecewise Linear Representation based on Important Data Points
RS	Random Shapelet (RS)
ST	Shapelet Transformation Algorithm (ST)
SDEA	Subsequence Distance Early Abandon (SDEA)
LFDP	Local Farthest Deviation Points (LFDP)

CHAPTER 1: INTRODUCTION

1.1 Background

With fast growing use of the electronics devices day by day, the requirement for the processing power of the computers is growing exponentially. Now smart devices have become very common and they share large volume of data all over the world. Since these smart devices came in to play, data flowing rates has rapidly increase and capturing, handling, preprocessing this data has become a huge challenge for humans. People are using different kind of technologies to mine such data and gather useful information.
Data processing is one of the key concepts in current computer industry and processing streaming data is a challenging task in this context. Identifying and extracting valuable information from streaming data by processing efficiently is a much-needed requirement for every field. Complex Event Processing (CEP) is a technology used to process streaming data [1].

1.2 Problem and Motivation

Complex Event Processing (CEP) is based on user pre-defined rules and those rules are being used to identify the relationship between events and facts that are expected to be identified. In current CEP systems, those static rule sets need to be defined manually by domain experts. Defining such a rule set is very time-consuming task and it needs domain specific knowledge to define those rules. Thus, this is becoming a limiting factor to growth of the CEP systems. Accuracy and efficiency of each CEP system heavily depend on user defined rule set of the CEP system. Therefore, defining a correctly analyzed rule set is highly effective for the performance of the system .

Different type of approaches has been proposed to reduce the complexity of the CEP rule setup. Some of the proposed approaches are semi-automated and mainly focused to reduce the technical complexity when setting up the rules. Approaches proposed by Turchin et al [3], Mutschler et al [4] and Sen et al [5] can be considered as semi-automated approaches. Margara et al [1,2], Mousheimish at al [6] and Navagamuwa et al [7] are proposed fully automated approaches for query generation. But those approaches are not computationally efficient and not scalable for large data set.

Automating a rule defining process of the CEP is a novel research area and so far, has received very low attention. Instead of using static pre-defined rule set, If CEP system can adjust rule set dynamically based on past events and relationships between events and facts, users can save their time and effort that they need to put, in defining a rule set.

1.3 Problem Statement

The problem we are intending to address in this research can be stated as follows :
How to identify a methodology to automate the CEP query generation for annotated multivariant time series without having domain related knowledge.

1.4 Research Objective

The main objective of this research is to implement a mechanism that has the capability to learn from past events, generate a CEP rule set dynamically and to provide implementation of such framework. This framework will define automated CEP rule generation process and identify the key sections of the process. Once the implementation is completed system will be evaluated with data sets used in previous research.
Outcome of the research will generate a query like below while having correct filtering parameters.

SELECT {*} **WHERE** { *attribute1 < a and attribute2 ≤ b* }
WITHIN { *time ≤ t1 and time ≤ t2 }*

1.5 Scope of the Research

Primary scope of this research is implementing an automated query generation system for CEP. The proposed system is based on the shaplet extraction methodology to generate CEP queries. In this study, unlabeled data is not considered for the scope and system doesn’t support unlabeled data sets. Accuracy of the proposed system is measured by using the ESPER complex event processing engine. The proposed system doesn’t evaluate against with other CEP engines.

1.6 Research Contribution

This study proposed a methodology to automate query generation for multivariate time series. The proposed system addresses the several short comes of the existing system have. The new system is a fully automated system manual user intervention is not required. Also, the proposed system is computationally efficient and scalable with large data set. Evaluation has been done with two data sets and obtained results proved that the accuracy, computationally efficiency and scalability of the system.

1.7 Outline

The remaining chapters of the report organized as follows. Chapter 2 discuss previous works done for the automate query generation of the CEP. This section also contains the discussion about the shapelet and different shapelet extraction algorithms. Chapter 3 discuss the methodology and implementation information about the proposed system. Chapter 4 contains information about the evaluation results. Final chapter contains conclusion and future work of the study.

CHAPTER 2: LITERATURE REVIEW

This chapter provides details of what is complex event processing is, the history of Complex Event Processing (CEP), the terminology of complex event processing systems, their use cases and how important it is to automate rules.

2.1 Complex Event Processing

Complex Event Processing is a system which has the capability to trigger events based on predefined rule sets [1]. CEP is a system which has the capability to collect information from various sources, then process the collected information and produce the action based on the processed data. CEP is doing the opposite action what Database Management Systems (DBMS) are doing. DBMS is the system which keeps the data and runs the queries top of the data. CEP is a system which keeps the query and runs the data stream against queries.

2.2 Automated Complex Event Generation

CEP is a system which holds the queries and runs through the data stream as mentioned earlier. Thus, queries are playing a major role in CEP systems. A Query is building the relationship between attributes of the data stream and situation that expect to be identified. In order to write the queries for the given set of events, the user should have strong domain knowledge on the context which those events occur. Performance of the CEP system is totally depending on the accuracy of the queries. To build accurate queries for the CEP system, domain knowledge plus technical skills are required. This constraint becomes a restriction for the CEP system to be used in many domains since it is difficult to find people with domain and technical knowledge. Automating the query generation is the possible solution to overcome this problem. Section 2.2.1 discuss the several approaches that have been taken to automate query generation. In order to automate this process, different types of data mining techniques and pattern recognition methodologies have been proposed. This section discusses the different complex event mining methods and relevant studies.

2.2.1 Machine Learning Approaches

Machine learning is studying and developing algorithms which have the capability to learn from data. In machine learning algorithm takes label data set as input and produce a set of rules as output. Depend on the nature of inputs machine learning tasks can be categorized into three sections.

01 – Supervised Learning – Sample inputs and their desired output are provided to the algorithm. The algorithm will be learning rules which can be used to map given inputs to the output.

02 – Unsupervised Learning – No labels are given for provided data set as inputs, the algorithm itself needs to learn rule set.

03 – Reinforcement Learning - Algorithm is dynamically interacting with an environment without providing a label data set.

Mostly used approach is using Machine Learning algorithms to extract patterns of the event stream. iCEP [2] , prediction correction algorithm [3], learning event detection are examples for such systems.

2.2.1.1 iCEP Framework

iCEP [2] is an automated rule generation framework which is developed by using machine learning techniques to identify hidden causalities from the event data stream. iCEP is designed as a modularized system which contains several modules and each module is responsible to identify a specific characteristic of the events. Every module doesn't need to be deployed to every environment. By analyzing the situation users can decide what the required modules that need to be deployed are.

iCEP framework contains five modules and below describes the usage of each module.

The methodology used for this research is to decompose the main problem into subproblems. To identify the event from the stream of events, they have decomposed events to subproblems and each subproblem is linked to one of each module.

01 - Identify the relevant time frame to consider. [Window learner]

02 - Identify the relevant attributes and event type. [Event and Attribute learner]

03- Identify the predicate which selects the event. [Predicate learner]

04- Identify the order of the events within the time frame. [Sequence learner]

05- Identify the event which should not appear in the composite event. [Negation learner]

iCEP is based on supervised machine learning technique and the user must provide the label data set to train the system. Inputs of the system are a series of event instances and each event instance is represented as a vector of attributes. As an output of the system, it produces a set of rules.

Evaluation of the iCEP framework has been done by focusing on two major points.

1. Quantitatively measure the accuracy of the generated rules.
2. Evaluate how valuable the generated rules are.

To evaluate the system, training and test data set for historical events have been synthetically generated. As the first step of the evaluation process, they have defined rule R and used rule R to identify composite events of the training data set. Then they split the training data set to positive and negative events. These events are considered as the input of the iCEP. Using these events iCEP infers the rule R*. To evaluate the performance of the iCEP quantitatively, the test data set is evaluated against rule R and rule R* is inferred. With this output Recall and Precision is calculated.

After evaluating iCEP quantitatively to measure accuracy, the system is evaluated subjectively as to how good the system is to generate the rule correctly. To evaluate the system subjectively, five matrices have been used.

1. *Type Matrix:* Measure the number of primitive types available in rule R but not infer rule R* and vice versa.
2. *Window Matrix :* Calculate the window size difference between in rule R and infer rule R*.
3. *Predicate Matrix:* Calculate the interval size of each predicate in rule R and R*.
4. *Sequence Matrix:* Measure the ordering constraint of each rule.
5. *Negation Matrix:* Measure the number of negations appear in one of the rules without having other rules.

With the small number of positive traces recall and precision of their algorithm is very low. With 700 positive traces, their algorithm was able to meet above 0.9 recall and precision. Accuracy of the result is decreased when the noise events are existing in the data set. Also result accuracy is decreased when negative events are selected by randomly. To overcome this issue, domain expert input is required when selecting negative events. But one year later they were able to come up with Associate Rule Mining based approach and that algorithm meet above 0.95 recall and precision with 100 positive traces.

2.2.1.2 iCEP – Association Rule Mining

In order to address the accuracy issues of the iCEP [2], new implementation of iCEP [1] has been proposed with ruled based approach. General idea behind that new implementation is that, for each positive trace defines all the possible constraints and take the intersection of the constraints to define the rules. As and example, if we have two positive traces X@0Y@3Z@1 and X@0Y@3W@1. Following constrains can be defined for each positive trace.

Constrains of the X@0Y@3Z@1 -

X – event X must occur , Y – event Y must occur , Z – event Z must occur

X -> Y – event X should occur before the event Y

Y -> Z – event Y should occur before the event Z

X -> Z – event X should occur before the event Z

Constrains of the X@0Y@3W@1-

X – event X must occur , Y – event Y must occur , W – event W must occur

X -> Y – event X should occur before the event Y

Y -> W – event Y should occur before the event W

X -> W – event X should occur before the event W

When defining the rule, iCEP only consider the intersection of the above two constraints. Which means only following constraints are considered to infer rule.

X – event X must occur

Y – event Y must occur

X -> Y – event X should occur before the event Y

This is the simple concept of the iCEP [1] implementation. Architecture of the iCEP [1] is more similar to the architecture of the iCEP[2].

In this approach also, if the positive trace count is less than 40, accuracy of the system is drop. When increasing the positive trace count, accuracy of the system also increases.

2.2.1.3 Prediction Correction Algorithm

In currently available CEP systems, rules are defined by the domain experts. For them, it is hard to identify to all the applicable rules, their combinations and to estimate values of the parameters. In order to address this problem, there should be an intelligent framework which continuously monitors the events and corrects the rule parameters. Turchin et al [3] proposed a solution to address this problem by combining knowledge of domain experts and machine learning techniques. They have developed an algorithm to identify and tune the parameters using Discrete Kalman Filter.

The difference between the Magara et al [2] approach and this one is that in this paper proposed the mechanism which consists of two repetitive stages namely “Rule Parameter Prediction” and “Rule Parameter Correction”. In the first stage, the rule parameters are updated using available expert knowledge of the future event changes. In the second stage, the rule parameters are corrected using the available expertise feedback of the past event occurrence.

When comparing this approach with iCEP [2] solution, there are two main drawbacks. This solution requires human intervention during the tuning phase of the algorithm. Also, this does not support negative queries as in iCEP [2].

2.2.2 Learning Event Detection rule with Hidden Markov Model

Mutschler et al [4] proposed a Hidden Markov Model (HMM) based approach for automating the query generation. Each CEP rule engine has its own Event Description Language (EDL) and it is varying from natural language. So, it is error prone to setup EDL rules by a person who has less programming knowledge. In this research, they have tried to overcome this issue, which means to allow the domain expert to identify the correct triggering event and facilitate him to set up the identified event as an EDL rule without prior knowledge about EDL. In other words, they have tried to eliminate the Software Engineer role but keeping the domain expert role. Thus, this research is different from other automated CEP rule generation researches because other researches had tried to eliminate both Software Engineer and domain expert role.

In this study, they have used an extended version of the HMM, which is called noise

Hidden Markov Model (nHMM). With this approach, the domain expert marks the event when a significant event occurred. The proposed system is capable to infer rules, based on nHMM model and then use them to identify similar events later.
In this study, they have defined five types of events with a notation. Below described are those five events and this will be used in the next section when describing the event detection rule learning.
Event: "*Event is a significant incident that is signaled to the user.*" [4]
Target Event: "*Target event is the event that is going to be modeled and it is denoted by \$.*" [4]
Model Event: "*An event that is part of the target event.*" [4]
Noise Event: "*An event that is not part of the target event.*" [4]

$$O: \underbrace{\mathtt{KUFOQNWSUKDAWHOBYRJGWC}}_{O_1}\mathtt{\$}\underbrace{\mathtt{DYJAHCHNHQCNWB}}_{\cdots}$$
$$\underbrace{\mathtt{IEDTEPYC}}_{O_2}\mathtt{\$}\underbrace{\mathtt{PLVBPGMIDDWAJMYPINBDRIKC}}_{O_3}\mathtt{\$}\cdots$$

Figure 2.1 Sample Event Sequence [4]

As shown in Figure 2.1 the sample event sequence is 'O_1' and we assume that domain expert signals the correct point when the event occurs. Target event can be denoted as '*A*B*C', where symbol '*' represents the noise event and A, B, C represent the model events. If we need to find the event which has length k (k=3), we need to model an HMM that can learn the pattern from event data. HMM model is depicted in Figure 2.2. The states which are in the gray color are the noise states. According to Figure 2.2 start state can be transit noise state or state 'A'. In the next step again, it can be transit to noise state or emission state. As a final step after emitting the event 'C' and \$, it can move to the starting state. Instead of having one large event to train the model, we can have small event instances, such as O_1, O_2 and O_3. Each training instance adjust the production probability of the model. With the explained method, they were able to train the model with the training data set.

As a summary, with this study, they have proposed an HMM based approach to automate query generation. But the proposed method is not a fully automated process as mentioned earlier. Domain expert inputs need when training the model.

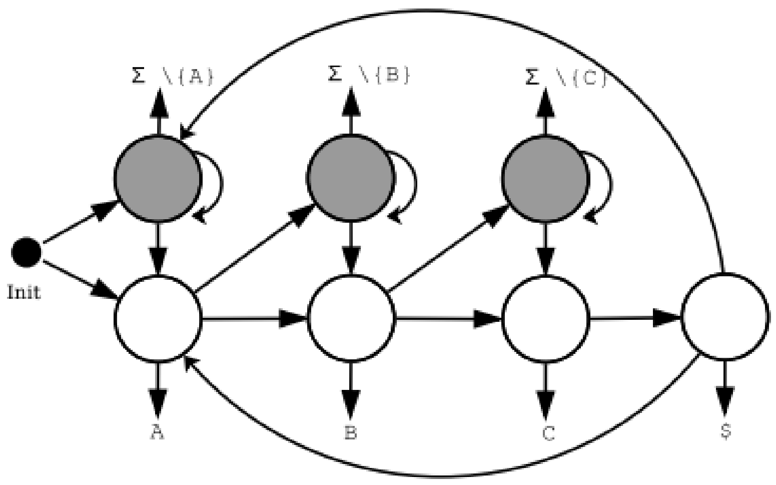


Figure 2.2 HMM for pattern ABC [4]

2.2.3 Recommendation Based Pattern Recognition

Recommendation based pattern recognition is another way to automate the query generation of the CEP. But again, this approach is not a fully automated process, and this can be considered as a semi-automated process. In the CEP system, defining new rules and updating the existing rules is a challenging task. Domain expert can provide a guideline when setting up the rules but cannot expect to provide all the required details. This recommender-based approach provides suggestions to the domain expert when setting the rule.

2.2.3.1 Iterative Event Pattern Recommendation

Sen et al [5] proposed a recommender-based pattern recommendation to semi-automate the query generation. In this study, suggested methodology supports the domain expert to design new rules and identify the missing patterns. Based on the architecture of the system, there are two ways to support the domain expert when creating the queries.

01- Total or partial reuse of pattern

02- Identify missing patterns

If there is a total or partially matching pattern already exist in the pattern repository, then that can be reused when defining the new pattern. This pattern reuse consists of three steps.

Step 01 – The domain expert defines a pattern artifact (PA_n) which expects to find rule repository (R).

Step 02 – Then system evaluate the pattern repository R and try to find the pattern artifact $PA_{r,}$ which is similar to $PA_{n.}$ All the findings ranked based on the distance and presented to the user.

Step 03 – User can either select the entire pattern or part of the pattern.

The second way to define the query is to identify the missing patterns. This also involves a three steps process.

Step 01- – The domain experts defines a pattern artifact (PA_m) which expects to find rule repository (R).

Step 02 – Based on the user input, the system identify the pattern which contains the pattern artifact PA_{rs} that is similar to $PA_{m.}$ The system finds the patterns which are conceptually equal. Conceptual equal means that pattern structure is similar, but property values are different.

Step 03 – The domain expert can select part of the system identified pattern and define a new one.

As a summary of this study, they have proposed a semi-automated system to support the domain expert to setup CEP rules. This is not a fully automated system and user intervention is required.

The proposed system is only evaluated against one domain, this needs to be further evaluated with different domains.

2.2.4 Shapelet Based Approach

Shapelet is a subsequence of the time series. Shapelet is originally proposed for time series classification in 2009 [8]. This has become more popular in time series classification domain and many types of research were published relevant to shapelet. Since there is a direct link between events and time series, this time series primitive has been considered as an approach of CEP query generation. Section 2.2.4.1 and 2.2.4.2 discuss the studies relevant to the shapelet based automate query generation. Section 2.3 discuss the shapelet and different shapelet extraction algorithm.

2.2.4.1 autoCEPs

autoCEP [6] is another automated query generation system which is based on shapelet extraction. This research can be considered as the first automated query generation approach which has been done based on shapelet extraction approach. autoCEP is designed as a two-phase system by using data mining techniques. The first phase of the system is responsible to learn historical trends and patterns and the second phase of the system is responsible to generate CEP queries by using identified historical trends and patterns.

The first phase of the system generates shapelets for given classified time series. Classified time series needs to be given as input of the system. Maximum and minimum lengths of shapelets are optional parameters of the first phase. If prior knowledge available about the shapelet length, that can be used to optimize the performance of the system. If this information is not available default values are used for minimum and maximum lengths.

Shapelet is a subsequence of the time series which is defined using the following three parameters. The distance between the time series and the shapelet is calculated by using Euclidean Distance. All the possible shapelets are generated for a given length.

$$S = (s, \delta, c_s)$$

$S - Shapelet$

$s - Numeric\ Time\ Series$

$\delta - Distance\ threshold$

$c_s - Class\ of\ the\ shapelet$

$$S \cong T = \begin{cases} true & if\ |S,T|_{Euclidean} \leq \delta \\ false & otherwise \end{cases}$$

In the second phase of the autoCEP system, the identified shapelets are used to generate CEP queries. Generated CEP queries contain three main sections. The first section is there to identify the time frame, the second component contain the relevant events and final block contain the condition which is required to be satisfied in identifying the event.

Within[*window]* {*relevant events*} **where**[*conditions*]

When evaluating the performance of the system two main parameters have been considered. The first evaluation parameter is the ‘Exactitude' of the system and the second parameter of the system is that ‘Earliness of the predictions'. To calculate the correction of the algorithm f score has been used.

$$\frac{1}{length(C)} \sum_{cl \in C} \frac{2 \times precision(cl) \times recall(cl)}{precision(cl) + recall(cl)}$$

To calculate the early prediction of the system, use the average percentage of time points needed to make the prediction. This is calculated by using the following formula.

$$\frac{1}{length(D)} \sum_{T \in D} \frac{EMT(\hat{s}, T)}{length(T)}$$

D – Data set

T – Time Series

s - Shapelet matched with T time series

EMT – Earliest matching time

The system is evaluated with the transportation process of Louver Museum [6]. During the transportation process of the artworks, they have measured the temperature variation and predicted the temperature values using embedded sensors. The user interaction of the autoCEP is optional. But results showed that domain expert interaction during shapelet learning period could speed up the process. During the shapelet learning process, maximum and minimum shapelet lengths need to be given as an additional two parameters. This will speed up the learning process of autoCEP.

autoCEP introduced Predictive Analysis to CEP rule generation domain and Early Classification has been used as a data mining technique. In this research, they have only implemented a univariant time series. Also, they have done the theoretical level analysis for multivariate time series and how that can be used to extend autoCEP. In the real-world scenario univariant time series analysis is not enough for complex CEP rules. Performance of the system relies on user input of the shapelet length.

2.2.4.2 Parallel Coordinates Based Automated Query Generation Approach

The main drawback of autoCEP is that proposed shapelet based approach only works for univariant time series. In the real-world scenarios, most of the time series are multivariant time series. Since the proposed approach cannot be used to automate query generation practically. Navagamuwa et al [7] proposed a technique that combines parallel coordinate and shapelet based approach to automate query generation for multi-variant time series. In addition to this, they have provided a clustering approach if the data set is unlabeled.

The proposed system mainly consists of four components. Data Processor converts the input data (XML, CSV) set to into common format. This module has the capability to label the data set if it is not pre-annotated. If the labeled data set is available this process is not required and can be skipped by the user. If the data set is not annotated the Euclidean distances between each data point are calculated. Then calculated distances are passed to the OPTICS algorithm and data is annotated based on them.

Shapelet generator is an important component of the system which is responsible to extract shapelets from the given data set. The first step of the shapelet extraction is to map multi-variant data set to parallel coordinates. Once the data is mapped to parallel coordinates all possible shapelets are extracted by varying window length (*l*). Extracted shapelet length is bounded to $1 < l <$ m. where ‘m’ is the number of attributes of the data set. With this approach, there is no need to pass user input to identify the minimum and maximum shapelet length to the system as in [6].

After extracting the all possible shapelets from the data set, the subset of shapelets need to be identified which represents the parallel coordinates. To identify the subset of shapelets Information Gain is calculated. Once the information gain is calculated, the shapelets are ranked on the descending order of the information gain. Then sorted shapelets are divided into subgroups and the shapelets which reside within the same subgroups are merged. Once the shapelets are merged, important shapelets have been extracted.

They have implemented a visualizing module which visualize the extracted important shapelets. This is an optional module and the user has the capability to select the shapelets which need to be used for query generation.

Query Generator is the final module of the proposed solution. Extracted shapelets need to be passed as an input parameter and module is responsible to generate the single query for each class.

The system was evaluated using the two data sets. The recall and the precision of the system are more promising when compared to the previous solutions. But again, the shapelet extraction time of the system depends on the length of the shapelets. The same drawback exists in autoCEP, which is not addressed in this study.

Summary :

During the literature review of the study, different approaches for automated query generation is evaluated. Most of the proposed solution is not fully automated and manual user intervention is required. Approaches discussed in Section 2.2.2 and 2.2.3 are semi-automated approaches. In these studies, the objective was to reduce the complexity of the query setup by giving some insight to the domain expert. With those approaches, domain expert doesn't need to be aware of the technical side of the query language. But his insight about the domain needs to be used to generate the queries. Approaches suggested by Margara et al [1,2], atoCEP [6] and parallel coordinates-based approach [7] are fully automated solutions for query generation. Among these, the results of shapelet based approach are more promising than others. Only two studies have been done in the shapelet based domain. Considering the above reasons this study mainly focusses on shapelet based approach to find a more optimized solution for automated query generation. Starting from Section 2.3 study is mainly focused on Shapelets and different shapelet extraction algorithms to identify a suitable solution for implementation.

2.3 Shapelet and Shapelet Extraction Algorithms

Definition of the shapelet and mathematical representation of the shapelet is discussed in Section 2.3.1. During the last two decades, several shapelet extraction algorithms were proposed to extract best matching shapelets from the time series data set. Under Section 2.3 the different type of shapelets algorithms are discussed.

2.3.1 Definition of the Shapelet

Shapelet is a time series primitive which has been introduced to overcome the limitation of the time series classification [8]. Until shapelet has been introduced as a new time series classification approach, the nearest neighbor algorithm was considered as the most accurate and robust approach for time series classification. To overcome the time and space complexity of the nearest neighbor algorithm, new time series primitive called shapelet has been introduced in [8]. According to the simple definition given in [8], shapelet is a subsequence of the time series.

Definition: " Given a time series T of length m, a subsequence S of T is a sampling of length $l \leq m$ of contiguous positions from T, that is, S = t_p , ,t_{p+l-1} , for $1 \leq p \leq m - l + 1$. " [8]

2.3.2 Brute Force Algorithm

Ye et al [8] proposed a brute force algorithm to extract shapelet from time series. Brute Force algorithm is the simplest algorithm to extract shapelets from Time Series. Along with data set D maximum and minimum length of the shapelets needs to be passed as input parameters. First of all, the algorithm generates all possible shapelets for given minimum and maximum lengths. Then the information gain for each candidate in the generated candidate list is calculated. Finally, the algorithm returns the candidate which is having the highest information gain.

In this algorithm, it is required store all the possible candidates of the data set. Therefore, it is extremely space inefficient. If the data set size is 'k' and the average length of the time series is 'm', the size of the candidate set is $O(m^2k)$. Also, this algorithm needs to calculate information gain for each candidate. The time complexity of the information gain calculation is O(mk). The overall complexity of the algorithm is $O(m^3k^2)$. Therefore, this brute force algorithm is very inefficient and cannot use for real-world scenarios.

2.3.3 Shapelet Discovery Algorithms (FS)

Mueen et al [10] proposed a new shapelet discovery algorithm which discovers the shapelet efficiently than brute force algorithm. This Fast Shapelet Discovery (FS) algorithm used two concepts to improve the performance of the shapelet extraction.

In this algorithm, they have proposed to use a caching structure for distance calculation for future use. The second concept was to prune low information gain candidates to reduce the number of calculation iteration of the shapelet discovery algorithm. Under Section 2.3.3.1 and 2.3.3.2 discuss the two approaches that have been used to reduce the calculation time complexity.

2.3.3.1 Efficient Distance Calculation

As depicted in Figure 2.3, if there are two subsequences D_j and D_k in data set D, in order to calculate the distance between two subsequences, we need to calculate the squared distance of each parallelogram. When considering the distance calculation of all the subsequence, this squared distance calculation needs to be repeated redundantly.

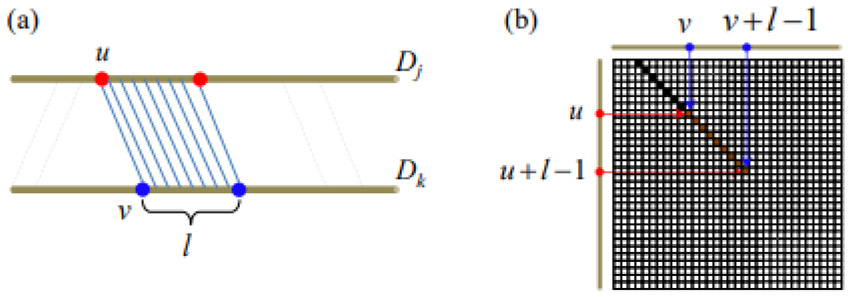


Figure 2.3 Distance computation required between a pair of subsequences [6]

In [6] they have proposed an approach to reduce this redundancy calculation. For every pair of points (D_j, D_k) they have calculated the five arrays.

S_x, S_y – The cumulative sum of the individual time series X and Y

S_{x^2}, S_{y^2} – The cumulative sum of the squared values.

$M_{[u,v]}$ – This is a 2-D array. Which contains the sum of products of the different subsequence of x , y.

$$\mathbb{S}_x[u] = \sum_{i=0}^{u} x_i \,, \qquad \mathbb{S}_y[v] = \sum_{i=0}^{v} y_i,$$
$$\mathbb{S}_{x^2}[u] = \sum_{i=0}^{u} x_i^2 \,, \qquad \mathbb{S}_{y^2}[v] = \sum_{i=0}^{v} y_i^2$$
$$\mathbb{M}[u,v] = \begin{cases} \sum_{i=0}^{v} x_{i+t} y_i & \text{if } u > v, \\ \sum_{i=0}^{u} x_i y_{i+t} & \text{if } u \leq v \end{cases}$$

where t = abs(u − v)

Mean, Variance and Euclidean distance can be calculated in constant time by using the above five arrays. Since values of the five arrays are calculated prior to the distance calculation, the redundant calculation can be omitted.

2.3.3.2 Candidate Pruning

Using this candidate pruning technique the number of iterations that need to be run to discover candidates are reduced. The basic concept behind this is that when it is known that the information gain of the given candidate is low; information gain of the candidate which is having a similar subsequence should be low. Therefore, expensive distance calculation can be avoided by identifying that type of candidates early.

As a summary, this algorithm was proposed to address the inefficiencies of the Brute Force algorithm. As an optimization step, this algorithm precomputes enough statistics that are needed to calculate the distance in constant time. With this approach space complexity of the algorithm is increased. They have sacrificed some space complexity to reduce the time complexity considerably. This Fast Shapelet Selection algorithm improves the computational performance magnitude when compared to the Brute Force algorithm. Similar kind of Shapelet Discover Algorithms are proposed in [11] and [12].

The accuracy of algorithm is measured with different time series classification algorithms. According to the evaluation results, the highest accuracy is given by 1-

NN Dynamic Time Warping algorithm. The proposed new algorithm is the second-best algorithm when considering the classification accuracy.

2.3.4 Shapelet Transformation Algorithm (ST)

Hills et al [13] proposed another shapelet extraction algorithm. This is an extended version of the algorithm that is proposed in [8]. As a first step, for each time series in data set all the possible shapelets are extracted from min length to max length. Then all the subsequences are assessed, and the distances are calculated. All the subsequences are stored in an array with their calculated quality values. Then the shapelets are sorted based on their quality value. Then the self-similar shapelets are removed from the list. Then non-self-similar shapelets are merged with current best shapelets and the top k shapelets are retained. This algorithm does not retain all possible shapelets. Instead, it only retains the best k shapelets. With this approach, it reduces a large amount of space overhead.

All algorithms proposed in [8, 10, 11, 12, 13] required minimum and maximum lengths of the shapelets as two parameters to extract all possible shapelets from time series data set. Length of the smallest subsequence and length of the longest subsequence are the two values that have been used as values of those two parameters. Hills et al [13] proposed an algorithm to get an estimated value for those two parameters. According to the proposed algorithm, ten number of time series are selected from the data set randomly. Then extract the best 10 candidates from given time series. In order to extract shapelets candidates from time series minimum length is passed as 3 and the maximum length is set to n. This algorithm is invoked 10 times to retrieve 100 best shapelets. Then shapelets are sorted by length. Length of the 25th shapelet is selected as the minimum length and length of the 75th shapelet is selected as the maximum length.

This is the extended version of the Brute Force algorithm. But this doesn't address the time complexity issue of BF algorithm. As same as the BF algorithm this is generating all the possible candidates for given shapelet length. But the space complexity issue is addressed in this algorithm. This algorithm removes the self-similar shapelets, then merge the shapelets and keep the best k shapelets based on the information gain. Because of the time complexity issue in this algorithm, it is difficult to use this algorithm in real-world scenarios.

2.3.5 Learning Shapelet Algorithm (LS)

All the shapelet finding algorithms discussed above are trying to find all possible shapelets from a given time series data set and then reduce the number of shapelets by calculating the distance and information gain. Grabock et al [14] propose a novel approach to extract shapelets from time series data set without evaluating all possible shapelets. Based on this new approach, the algorithm is capable to learn optimal shapelets without evaluating all possible shapelets. This new approach can be divided into two phases.

1. Start with initial guesses for shapelets
2. Iteratively evaluate and learn the shapelets by minimizing the classification loss function

In this paper, they have used the logistic regression classification since it provides the capability to predict binary targets as probabilistic confidence. The evaluation results show that this algorithm is much faster than the Shapelet Transformation algorithm.

2.3.6 Fast Shapelet Selection Algorithm (FSS)

Ji et al [15] proposed an algorithm to optimize the shapelet extraction time without sacrificing the accuracy. Algorithms proposed in [13, 10, 16] nearly pick all possible candidates. Then try to reduce the distance calculation complexity using different methodology. But in the algorithm proposed in [15], they have reduced the number of shapelets picked in the first stage. Subclass splitting method has been used to reduce the number of shapelets. Then the algorithm identifies the Local Farthest

Deviation Points (LFDP) for sampled time series and selects the subsequence time series between two non-adjacent LFDP points as a shapelet. With this approach the number of shapelets extracted from the data set is reduced, thereby reducing the time complexity of the shapelet extraction process. Section 2.3.6.1 discuss the subclass splitting methodology and Section 2.3.6.2 discuss the Local Farthest Deviation Points (LFDP) identification methodology.

2.3.6.1 Subclass Splitting

Sequence in a given time series can only belong to a single class, but it can have different characteristics. This is called sub classing. Figure 2.4 depict sequence which contains two clearly identifiable subclasses. First step of the subclass splitting is to identify the pivot sequence. Picking a random sequence from the data set is the first option to identify the pivot sequence. But using a mathematically calculated pivot sequence can improve the accuracy of the classification.

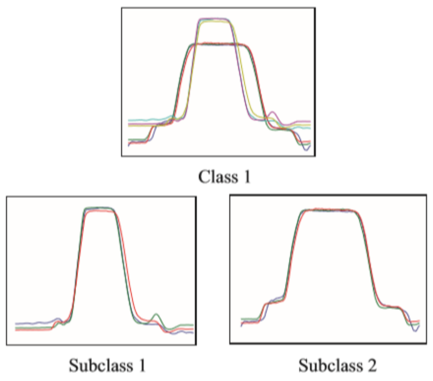


Figure 2.4 Sequence which contains subclasses

Assume there are n_c time series in the given class and each sequence contains m attributes.

$$T = \{T_1, T_2, T_3, \dots.., T_j, \dots., T_m\}$$

$$T_n = \{T_{n_1}, T_{n_2}, T_{n_3}, \dots., T_{n_c}\}$$

Sum up all the values of the T time series and calculate the sum value of the T time series (Sum T).

$$Sum\,T = \sum_{i=1}^{m} T_i \qquad (01)$$

The sum value of every time series can be calculated as below (Sum T$_n$).

$$Sum\,T_n = \sum_{j=1}^{c} Sum\,T_j \qquad (02)$$

Then calculate the mean value of the entire Time Series and the Time Series which is close to the mean value consider as the pivot series.

$$Mean\,T_n = \frac{Sum\,T_n}{n_c} \qquad (03)$$

$$argMin_{i\in(1,2,\ldots\ldots,n_c)}(|Sum\,T_i - Mean_{T_n}|) \qquad (04)$$

After identifying the pivot series Euclidean distance can be calculated for each sequence in the class. Suppose $T_c = \{c_1, c_2, c_3, \ldots\ldots, c_m\}$ is the pivot series of the class. Euclidean distance of the $T_i = \{t_1, t_2, t_3, \ldots\ldots, t_m\}$ can be calculated by using the following formula. Then sort the Euclidean distance values and calculate the adjacent discrepancies of the series.

$$euc\ distance = \sqrt{\sum_{j=1}^{m}(t_j - c_j)^2} \qquad (05)$$

Next, need to calculate standard deviation of the adjacent discrepancies. Calculated standard deviation is used to split the sequences in the class to sub classes.

2.3.6.2 Piecewise Linear Representation (PLR)

A time series can be represented in many forms such as Fourier Transform, Wavelets and Symbolic Mapping [17]. Piecewise Linear Representation (PLR) is another representation of a time Series [18]. Main concept of the representing a time Series in another form is to consider the main shape of the time series and discard the local details. There are many variations of the PLR been published. In this research, the Piecewise Linear Representation method based on Important Data Points [19] is used.

As described above PLR is a representation of time series and segments of the time series are identified based on fitting errors of the time series. Fitting distance of the time series is calculated by identifying the distance between fitting curve and time series. This distance can be calculated through Euclidean Distance, Orthogonal Distance or Vertical Distance [19]. Below equation is used to calculate the vertical distance between fitting curve and time series.

Point one on line – (t_1, v_1) Point two on line – (t_2, v_2)

Point on fitting line - (t_0, v_0)

$$Vertical\ Distance\ (d) = \left| v_1 + \frac{(t_0 - t_1) * (v_2 - v_1)}{(t_2 - t_1)} - v_0 \right| \tag{06}$$

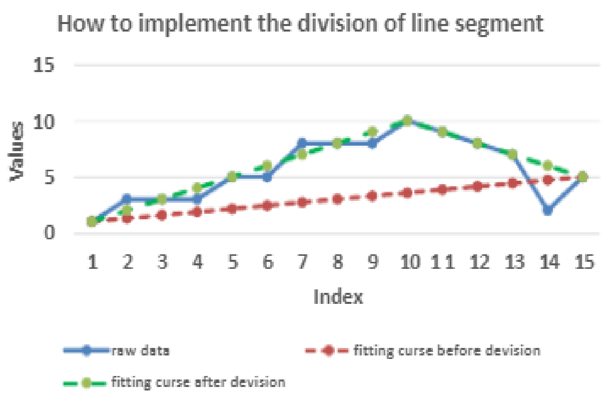


Figure 2.5 Identify IDP Point [19]

Figure 2.5 Depict raw data of the time series in blue color line and initial fitting curve in red line. If we take point 10, fitting distance between fitting curve and time series is 4. This is the maximum fitting distance of the fitting curve in red color. According to the definition 01. point 10 can be considered as an important data point. New fitting curve can be created by considering this IDP (point 10). New fitting curve is depicted in green line.

2.3.6.3 FSS Algorithm Evaluation

There are two ways to reduce the time of the shapelet extraction process. The first approach is to reduce the evaluation time complexity. In [12] two speed up approaches has been introduced. Those are the Subsequence Distance Early Abandon (SDEA) and Admissible Entropy Pruning (AEP). The approach of the SDEA is that when the distance is larger than the current smallest value, a further calculation is stopped. The approach of the AEP means to "calculates the cheap-to-compute upper bound of the information gain and uses this to admissibly prune certain candidates" [20]. Another approach to reducing the time complexity is to reuse the calculation and pruning the search space [10].

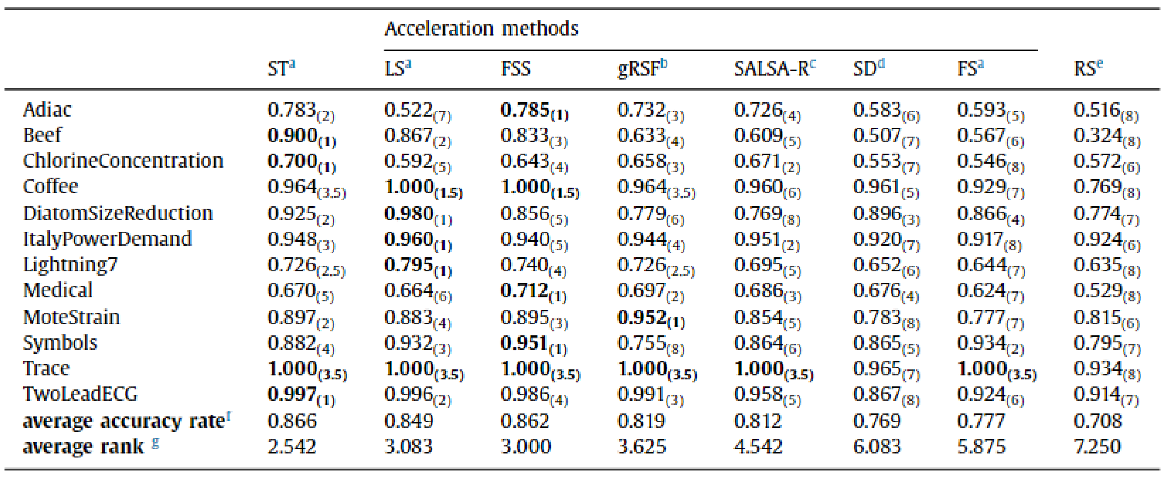
The second approach is to reduce the number of candidates. Setting the values for minimum and maximum length of shapelets can reduce the number of shapelets. Hills et al [13] proposed an algorithm to estimate the values for minimum and maximum length. This approach is discussed under 'Shapelet Transformation Algorithm'. Another way to do this is to select the subsequences which contain one or more Important Data Points (IDP) [19]. Sampling the candidates from the data set is another way to reduce the number of candidates. Different sampling methodologies are proposed in [21, 22, 23].

SALSA-R [21] – This is a sampling-based algorithm which does not process all possible candidates.

Random Shapelet (RS) [22] – This algorithm is based on randomization of the process.

Generalized Random Shapelet Forest gRSF [23] – The algorithm built on using a random subset of shapelets for each decision trees [15].

In [15], the accuracy and performance of each shapelet extract algorithm are compared and evaluated. 12 data sets from the UEA and UCR Time Series Classification Repository [24] has been used for evaluation.



		Acceleration methods						
	ST^{a}	LS^{a}	FSS	gRSF^{b}	SALSA-R^{c}	SD^{d}	FS^{a}	RS^{e}
Adiac	0.783$_{(2)}$	0.522$_{(7)}$	**0.785$_{(1)}$**	0.732$_{(3)}$	0.726$_{(4)}$	0.583$_{(6)}$	0.593$_{(5)}$	0.516$_{(8)}$
Beef	**0.900$_{(1)}$**	0.867$_{(2)}$	0.833$_{(3)}$	0.633$_{(4)}$	0.609$_{(5)}$	0.507$_{(7)}$	0.567$_{(6)}$	0.324$_{(8)}$
ChlorineConcentration	**0.700$_{(1)}$**	0.592$_{(5)}$	0.643$_{(4)}$	0.658$_{(3)}$	0.671$_{(2)}$	0.553$_{(7)}$	0.546$_{(8)}$	0.572$_{(6)}$
Coffee	0.964$_{(3.5)}$	**1.000$_{(1.5)}$**	**1.000$_{(1.5)}$**	0.964$_{(3.5)}$	0.960$_{(6)}$	0.961$_{(5)}$	0.929$_{(7)}$	0.769$_{(8)}$
DiatomSizeReduction	0.925$_{(2)}$	**0.980**$_{(1)}$	0.856$_{(5)}$	0.779$_{(6)}$	0.769$_{(8)}$	0.896$_{(3)}$	0.866$_{(4)}$	0.774$_{(7)}$
ItalyPowerDemand	0.948$_{(3)}$	**0.960$_{(1)}$**	0.940$_{(5)}$	0.944$_{(4)}$	0.951$_{(2)}$	0.920$_{(7)}$	0.917$_{(8)}$	0.924$_{(6)}$
Lightning7	0.726$_{(2.5)}$	**0.795$_{(1)}$**	0.740$_{(4)}$	0.726$_{(2.5)}$	0.695$_{(5)}$	0.652$_{(6)}$	0.644$_{(7)}$	0.635$_{(8)}$
Medical	0.670$_{(5)}$	0.664$_{(6)}$	**0.712$_{(1)}$**	0.697$_{(2)}$	0.686$_{(3)}$	0.676$_{(4)}$	0.624$_{(7)}$	0.529$_{(8)}$
MoteStrain	0.897$_{(2)}$	0.883$_{(4)}$	0.895$_{(3)}$	**0.952$_{(1)}$**	0.854$_{(5)}$	0.783$_{(8)}$	0.777$_{(7)}$	0.815$_{(6)}$
Symbols	0.882$_{(4)}$	0.932$_{(3)}$	**0.951$_{(1)}$**	0.755$_{(8)}$	0.864$_{(6)}$	0.865$_{(5)}$	0.934$_{(2)}$	0.795$_{(7)}$
Trace	**1.000$_{(3.5)}$**	**1.000$_{(3.5)}$**	**1.000$_{(3.5)}$**	**1.000$_{(3.5)}$**	**1.000$_{(3.5)}$**	0.965$_{(7)}$	**1.000$_{(3.5)}$**	0.934$_{(8)}$
TwoLeadECG	**0.997$_{(1)}$**	0.996$_{(2)}$	0.986$_{(4)}$	0.991$_{(3)}$	0.958$_{(5)}$	0.867$_{(8)}$	0.924$_{(6)}$	0.914$_{(7)}$
average accuracy rate^{f}	0.866	0.849	0.862	0.819	0.812	0.769	0.777	0.708
average rank ^{g}	2.542	3.083	3.000	3.625	4.542	6.083	5.875	7.250

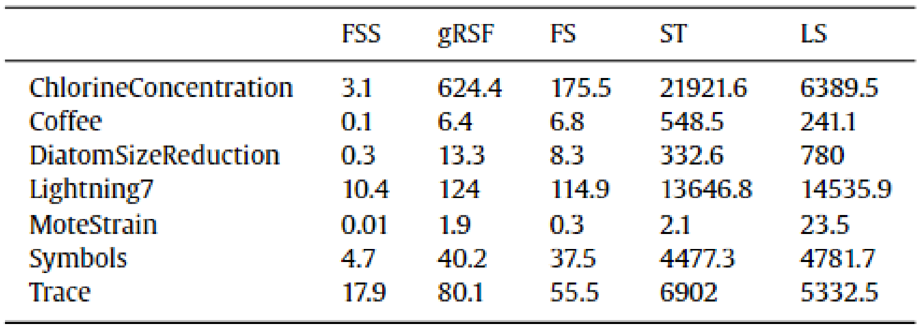
Figure 2.6 Accuracy comparison [15]

To compare the accuracy between different algorithms, average accuracy rate and average accuracy rank have been used. Average accuracy rate and average rank are calculated as below.

average accuracy rate= ∑accuracy rate of the method on each data set / number of data set

average rank= ∑rank of the method on each data set / number of data set

When evaluating the accuracy of FSS algorithm, out of 12 data sets FSS algorithm gives the highest accuracy for five data sets. According to the results, LS algorithm also given the highest accuracy rate for five data sets. The average accuracy rate of the FSS algorithm is higher than the average accuracy of the LS algorithm. Highest accuracy rate is given by the ST algorithm. But the time complexity of this algorithm is considerably high. Due to this reason, it is difficult to use ST algorithm in real-world scenarios. FSS algorithm gives the second-best average accuracy rate. Based on the average accuracy rate calculation and average rank calculation, FSS algorithm gives the best accuracy among others. Running time comparison has been done for FSS, gRSF, FS, ST and LS algorithms. According to the running time evaluation FSS algorithm is significantly efficient than other algorithms.



	FSS	gRSF	FS	ST	LS
ChlorineConcentration	3.1	624.4	175.5	21921.6	6389.5
Coffee	0.1	6.4	6.8	548.5	241.1
DiatomSizeReduction	0.3	13.3	8.3	332.6	780
Lightning7	10.4	124	114.9	13646.8	14535.9
MoteStrain	0.01	1.9	0.3	2.1	23.5
Symbols	4.7	40.2	37.5	4477.3	4781.7
Trace	17.9	80.1	55.5	6902	5332.5

Figure 2.7 Running Time Comparison (Seconds) [15]

According to the evaluation results FSS algorithm outperforms the ST and LS algorithm comprehensively.

Summary :

Shapelet Transformation (ST) algorithm is the most prominent algorithm among the other shapelet extraction algorithms. Due to its higher accuracy, ST algorithm is considered as one of the best solutions for shapelet extraction. The time complexity of the ST algorithm is $O(n^2m^4)$. Where n is the number of series and m is the series length. Its clear that when the size of the data set and the series length is increased, the running time of the ST algorithm is exponentially increased.

To address the computational inefficiency of the Brute Force (BF) algorithm, Fast Shapelet (FS) algorithm was proposed. FS algorithm used the cache-based approach and pruning method to reduce the calculation complexity. The time complexity of the FS algorithm is $O(nm^2)$. This algorithm clearly reduces the time complexity when compared to the ST algorithm which is an extension of the BF algorithm. Running time gain of the FS algorithm can be clearly identified in Figure 2.7. But the accuracy of the FS algorithm is low when compared to the ST algorithm. According to the information available in Figure 2.6, the average accuracy of the FS algorithm is 0.777 and average accuracy of the ST algorithm is 0.864.

Shapelet Learner (LS) algorithm is another shapelet extraction algorithm which was proposed to reduce the time complexity. The time complexity of the LS algorithm is $O(em^2n^2R^2)$, where ‘e’ is max no. of iteration, R is shapelet scale. The obtained results in Figure 2.7 shows that this algorithm couldn’t reduce the time complexity considerably. Also, the average accuracy of the LS algorithm is low compared to the ST algorithm

gRSF, RS and SALSA-R are several algorithms which were proposed to improve running time and accuracy. But none of them were able to achieve the accuracy that ST or FSS algorithms has achieved.

FSS algorithm is the fastest algorithm among the others. Also obtained results show that FSS was able to achieve the second highest accuracy. This accuracy is very near to the accuracy of the ST algorithm. As a conclusion FSS algorithm is the best shapelet extraction algorithm both accuracy wise and time complexity wise.

CHAPTER 3: METHODOLOGY

To Automate the query generating of the CEP from multivariant time series, a methodology which is based on Shaplets, Subclass Splitting and Piecewise Linear Representation is proposed. This chapter will discuss the detail implementation and methodology that has been used for implementation.

3.1 High Level Design

The system consists of two main components. First component is ‘Shaplet Learner” which is responsible to generate shaplets by analyzing labeled data provided as input. Output of the Shaplet Learner module is a list of generated shaplets. Second component is “Query Generator”. Generated shaplets needs to be passed as input and Query Generator module return the generated quires based on the provided shaplets. This module will generate query for each class in data set.

3.1.1 Shaplet Learner Module

This is the main component of the system and accuracy of this component directly impact the outcome of the system. This module doesn’t support any unlabeled data and handling the unsupervised data is not considered in this scope. Since labelled data set should be passed as input parameter of the system. System does support the ‘. arff’ and ‘.csv’ as input file formats. Fast Shaplet Selection (FSS) algorithm has been used to extract shaplets from data set. This FSS algorithm is computationally efficient. Time that needs to be spent to identify shaplets is largely reduced since this algorithm is used for Shaplet Learner. Shaplet extraction process can be divided into five main steps.

Step 01: Select Pivot Time Series

Step 02: Split Subclasses

Step 03: Sample Time Series from Sub Classes

Step 04: Identify End Points

Step 05: Generate Shaplet Candidates

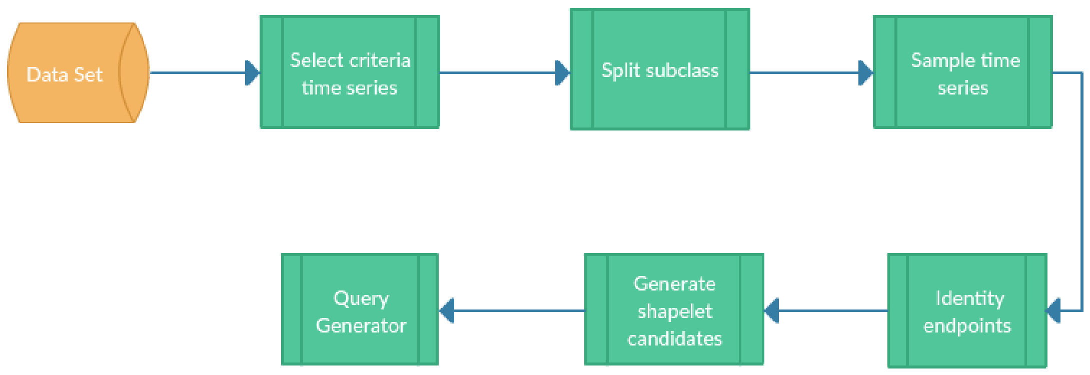


Figure 3.1 Steps of the Process

Step 01: Select Criteria Time Series

As a first step of the FSS algorithm calculate the mean sum value of the given data set. If there are n_c time series in one class and each sequence contains m attributes.

$$T = \{T_1, T_2, T_3, \ldots.., T_j, \ldots., T_m\}$$

$$T_n = \{T_{n_1}, T_{n_2}, T_{n_3}, \ldots., T_{n_c}\}$$

$$Sum\ T = \sum_{i=1}^{m} T_i \qquad (01)$$

$$Sum\ T_n = \sum_{j=1}^{c} Sum\ T_j \qquad (02)$$

By using the equations (01), (02) and (03) can obtain the mean sum value of each class. Then calculate difference between mean value and sum value of each time series. The time series which has minimum difference is selected as the pivot time series (04).

$$Mean\ T_n = \frac{Sum\ T_n}{n_c} \qquad (03)$$

$$argMin_{i\in(1,2,\ldots\ldots,n_c)}(|Sum\ T_i - Mean_{T_n}|) \qquad (04)$$

$$euc\ distance = \sqrt{\sum_{j=1}^{m}(t_j - c_j)^2} \qquad (05)$$

Step 02: Split Subclasses

After identifying the pivot time series, Euclidean distance is calculated for each time series using formula (05). In Algorithm 3, code in line numbers 3-8 are used to calculate the Euclidean distance. Once the Euclidean distance calculation is completed, the time series is sorted based on the value of Euclidean distance. Line 9 shows the distance value sorting. Then the adjacent discrepancy of each time series is calculated, and that value is used to calculate standard deviation. Lines 10-15 are used to calculate adjacent values and standard deviation. Calculated standard deviation is used to split the class in to sub classes. Lines 16-26 are used for subclass splitting

Algorithm 01 Sub Class Splitting Algorithm

```
1: procedure subclassSplitting (D, P) ▷ Time Series Data Set D, Pivot Time Series P
2:      distance ← { }
3:      for each row r in D do
4:          while start ← r.length
5:              distance[r] = distance[r] + $(P[r] – D[r])^2$
6:          end while
7:          distance[r] = sqrt(distance[r])
8:      end for
9:      shortAscending(distance{})
10:     adjacentDistance ← { }
11:     for i = 2 to distance.length do
12:         adjacentDistance[i] = distance[i] – distance[i-1]
13:     end for
14:     stdDivDistance ← 0
15:     stdDivDistance ← calculateStdDiviation(adjacentDistance{})
16:     class  ← {}
17:     class[1] ← 1
18:     c ← 1
19:     for i = 2 to D.length -1 do
20:         if i=1
21:             class[i+1] ← 1
22:         else if  adjacentDistance[i] > stdDivDistance
23:             class  ← c + 1
24:             c ← c + 1
25:         end if
26:     end for
27:     return class                    ▷ Identified Sub Classes will be returned
28: end procedure
```

Algorithm 1 Sub Class Splitting Algorithm

Step 03 : Sample Time Series from Sub Classes

Once the sub classes are identified we can select the sample time series from each sub classes. When sampling the time series, the time series which has minimum sum distance compared to other time series is selected as sample time series. At least one time series needs to be picked from each sub class in minimum. But we can increase the number of samples that we are selecting from each sub class. Selected time series are used to identify end points and generate shaplets.

Step 04: Identify End Points

After selecting the sample time series from sub classes PLR_IDP algorithm is used to identify end points of the sample series. These end points are called as Local Farthest Deviation Points (LFDP). To identify LFDP , the algorithm proposed by C Ji et al.[4](Algorithm 2) is used. Algorithm is starting from first and last end points of the time series. Assume first and last points are the first two LFDP of the time series. Then calculate the fitting error (vertical distance) for each data point by using equation (06). Calculated fitting errors are used to define weight of the time series segment. Equation (07) is used to identify the weight of the segment.

$$weight = \max(dist_{sum}, 2 * dist_{max}) \qquad (07)$$

$dist_{sum}$ − Sum of the fitting errors of the all points in time series segment
$dist_{max}$ − Maximum fitting error of the time series segment

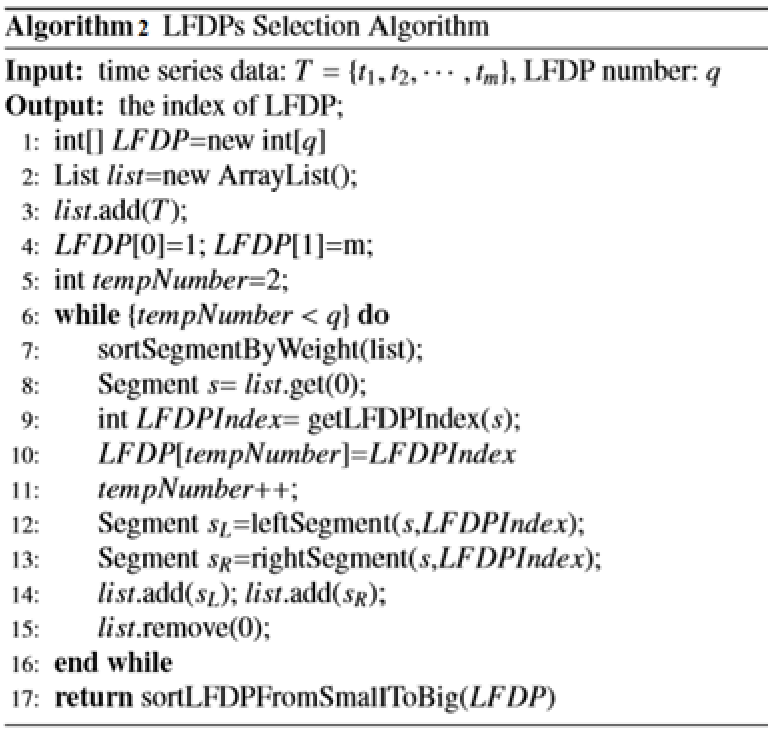
The points which are in time series segment to be considered as LFDP, two conditions need to be satisfied.

- The point has the maximum fitting error (distance)
- The sub sequence which is point reside should have maximum weight

As mentioned above start and end points are considered as starting LFDPs. Then a recursive approach is used to identify LFDPs. During the recursive process sort the line segments by weights and identify the line segment which has maximum weight. Then evaluate all the points of the maximum weight line segment and identify the point which has maximum fitting error (distance) as LFDP. Then LFDP identified line segment split into two-line segments. Add newly generated two-line segments to list which contains the information about line segments and remove the previously split line segment from the list. This recursive process continues until identify given number of LFDPs. Each iteration of the process identifies new LFDPs.

Step 05: Generate Shaplet Candidates

Identified LFPDs use to generate the shaplets.



Algorithm 2 LFDPs Selection Algorithm

```
Input: time series data: $T = \{t_1, t_2, \cdots, t_m\}$, LFDP number: $q$
Output: the index of LFDP;
int[] LFDP=new int[q]
List list=new ArrayList();
list.add(T);
LFDP[0]=1; LFDP[1]=m;
int tempNumber=2;
while {tempNumber < q} do
    sortSegmentByWeight(list);
    Segment s= list.get(0);
    int LFDPIndex= getLFDPIndex(s);
    LFDP[tempNumber]=LFDPIndex
    tempNumber++;
    Segment $s_L$=leftSegment(s,LFDPIndex);
    Segment $s_R$=rightSegment(s,LFDPIndex);
    list.add($s_L$); list.add($s_R$);
    list.remove(0);
end while
return sortLFDPFromSmallToBig(LFDP)
```

Algorithm 2 LFDPs Selection Algorithm [15]

3.1.2 Query Generator Module

Once the shapelets are identified, shapelets are grouped by using the class value of the each shapelet. Single query is generated for each class value. When generating the queries, the minimum and maximum values of each attributes are identified, and those values are used to generate the queries.

CHAPTER 4: RESULTS AND ANALYSIS

This chapter discusses the performance evaluation of the FSS algorithm-based CEP query generation approach. Section 4.1 discusses the evaluation result of the proposed algorithm. Section 4.2 compares the obtained results with previously implemented approaches.

4.1 Evaluation of the Algorithm

During the evaluation of the proposed system, the following two measures have been evaluated.

01 – Total time for shapelet extraction.

02 – Accuracy of the generated queries.

4.1.1 Total Time for Shapelet Extraction

Since the proposed methodology for automating the rule generation for CEP is a shapelet based approach, as first evaluation criteria, the total time spent for shapelet extraction is evaluated. As first step of the process, shapelets which are accurately representing the entire data set needs to be extracted from the data set. Accuracy of the generated queries totally depend on accuracy of the extracted shapelets. If the time complexity of shaplet extraction algorithm increase exponentially when the number of records increase in the data set, that algorithm cannot be used for a large data set. This is directly impacting the accuracy of the generated queries. During the evaluation of the shapelet extraction process, the running time of the shapelet extraction process against the volume of the training data set is evaluated. Also, the behavior of the running time when increasing the training data volume is evaluated.
This is the key evaluation parameter of the entire system. Here the accuracy of the system against the labeled data set is evaluated. When evaluating the accuracy of the system following two parameters has been considered.

- Precision
- Recall
- F1 Score

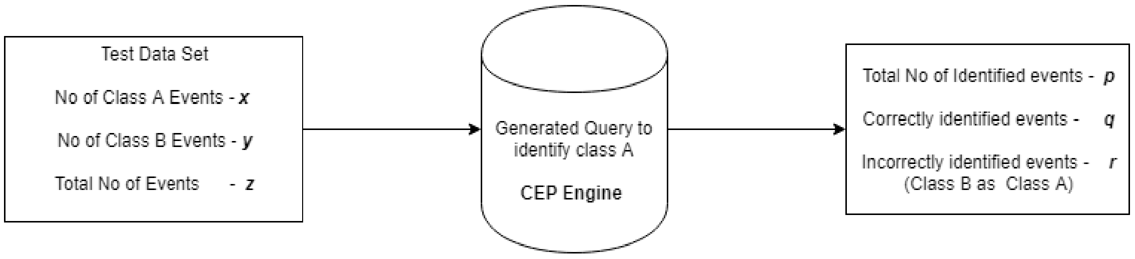

Test Data Set
No of Class A Events - x
No of Class B Events - y
Total No of Events - z
Generated Query to identify class A
CEP Engine
Total No of Identified events - p
Correctly identified events - q
Incorrectly identified events - r
(Class B as Class A)

Figure 4.1 Accuracy Evaluation

Figure 4.1 depicts the system used for accuracy evaluation. As an input to the system, the labeled data set has been passed in. CEP engine contains the generated query which can be used to identify class A events. If the given input event is classified as a class A event by CEP engine, that event is returned as an identified event. If the CEP engine classified the event as a class B event, then CEP engine doesn't return the event as identified event. Table 4.1 explains the calculation performed to obtain recall, precision, false positive and false negative values.

Table 4.1 Calculation Formula

No of events in data set	Z
No of identified events	P
Recall	(q / x) * 100
Precision	(q / (q + r)) * 100
False Positive	R
False Negative	x – q

4.1.2 Data Set

To evaluate the performance of the system two data sets have been used. These two data sets were taken from the UEA and UCR Time Series Classification Repository [24]. Table 4.2 contains information on ‘Occupancy Detection’ data set and Table 4.3 contains the information on ‘EEG-Eye State’ data set.

Table 4.2 Occupancy Detection Data Set

No. of attributes	7
Total number of events	8143
No. of not occupied events	6414 (78%)
No. of occupied events	1729 (22%)

Table 4.3 EEG Eye State Data Set

No. of attributes	15
Total number of events	14980
No. of not eye open events	8257 (55%)
No. of eye close events	6723 (45%)

4.1.3 Evaluation

The proposed system uses the FSS shapelet extraction algorithm to extract shapelets from data set. Running time of the shapelet extraction algorithm is evaluated with different sizes of training data set. The evaluation was carried out in Java on an Intel(R) Core(TM) i5-6200U CPU @ 2.30Ghz CPU with 8GB memory.

Table 4.4 FSS algorithm Running Time

Data Set	No. of Instances	Running Time (ms)
Occupancy Detection	500	1533
EEG Eye State	500	1765

Evaluation results show that when the number of attributes in data set increase, time for the shapelet extraction also slightly increase. When the number of train events increases, shapelet extraction time also increase. Figure 4.2 shows that the running time of the shapelet extraction is not exponentially increased.

Table 4.4 contains the benchmark values of the FSS shapelet extraction algorithm. Benchmark has been done using 500 instances and two data sets. This benchmark values are used to compare the running time of the CEP query generate method proposed in Nawagamuwa et al [7].

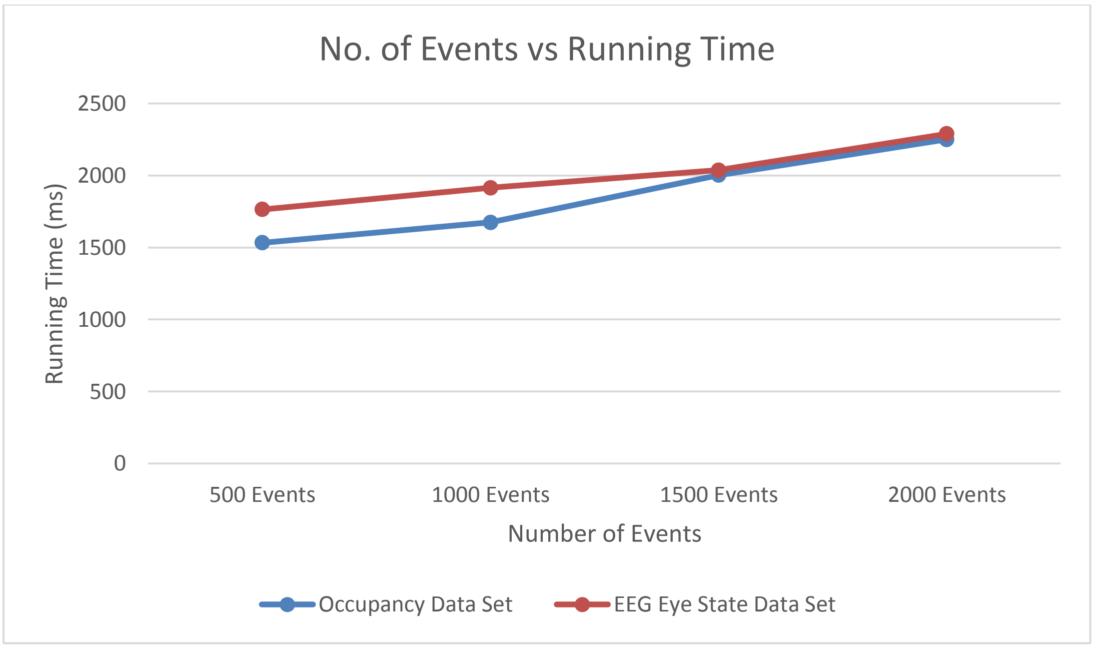


Figure 4.2 Running Time Comparison

Then accuracy is calculated for the two data sets by using the approach mentioned in Section 4.1.1. For the occupancy data set, 6797 events have been used as a training data set and 1345 events has been used as a test data set. Table 4.5 contains the accuracy calculation for occupancy data set.

Table 4.5 FSS - Occupancy Data Evaluation

Event	Metric	Value
Occupied	No of occupied events in the dataset	447
	No of events detected using CEP query	442
	Recall	99.549%
	Precision	0.98.881%
	F1 Score	0.9921
	False positives	2 events
	False negatives	3 events
Not Occupied	No of occupied events in the dataset	898
	No of events detected using CEP query	898
	Recall	100%
	Precision	75.209%
	F1 Score	0.8585
	False positives	296 events
	False negatives	0 events

Table 4.6 contains the accuracy calculation of the EEG Eye State. 11984 events contained in the training data set and 2000 events contained in the test data set.

Table 4.6 FSS-EEG Data Evaluation

Event	Metric	Value
Eye Open	No of occupied events in the dataset	985
	No of events detected using CEP query	884
	Recall	89.74%
	Precision	75.04%
	F1 Score	0.8173
	False positives	294 events
	False negatives	101 events
Eye Close	No of occupied events in the dataset	1014
	No of events detected using CEP query	883
	Recall	87.08%
	Precision	88.03%
	F1 Score	0.8755
	False positives	120 events
	False negatives	131 events

4.2 Result Comparison

To compare the accuracy and time complexity of the new approach, obtained results were compared with another shapelet based approach proposed in [7]. In the proposed method [7], the Shapelet Transformation (ST) algorithm is used to extract shapelets from data set. The result comparison between the new system and the system [7] has been done in two ways. Time complexity between the shapelet transform algorithm and the FSS algorithm has been compared to identify the most efficient algorithm. To do the more fare evaluation same training data sets has been used to calculate the time complexity of the shapelet transformation algorithm.

As explained in Section 4.1.3, 6797 and 11984 events have been used in two train data sets respectively. But this training data couldn’t be used in the shapelet transformation algorithm since running time is exponentially increased for the ST algorithm. For ST algorithm time complexity is evaluated against the different size of training data set. 500, 1000, 1500 and 2000 are the different data set sizes that have been used to evaluate time complexity. Figure 4.3 shows that when the event count is increased running time of the ST algorithm is exponentially increased. It's clearly visible that the running time of the FSS algorithm is much better than the running time of the ST algorithm.

Time complexity of the ST algorithm is O(n^2m^4), where n is number of event instances in data set and m is number of attributes in an event. When the number of attribute count is increased, the running time of the shapelet extraction algorithm is high. Shapelet extraction time of the ST algorithm is considerably high for EEG data set when compared to the Occupancy data set. Attribute count of the EEG data set is 15 and 7 for occupancy data set. According to the evaluation result, when attribute count is increased running time of the ST algorithm increase exponentially.

Table 4.7 Running Time (ms) Comparison between FSS and ST

Algorithm	Data Set	No. of Events			
		500	1000	1500	2000
FSS	Occupancy	1533	1674	2002	2251
	EEG	1765	1915	2037	2289
ST	Occupancy	20103	159798	623073	1048845
	EEG	266507	1254721	Not Calculated	Not Calculated

Table 4.7 clearly shows that FSS shapelet extraction algorithm outperforms the ST shapelet extraction algorithm. Also, the ST algorithm running time is increased when the number of attributes in the data set to increase.

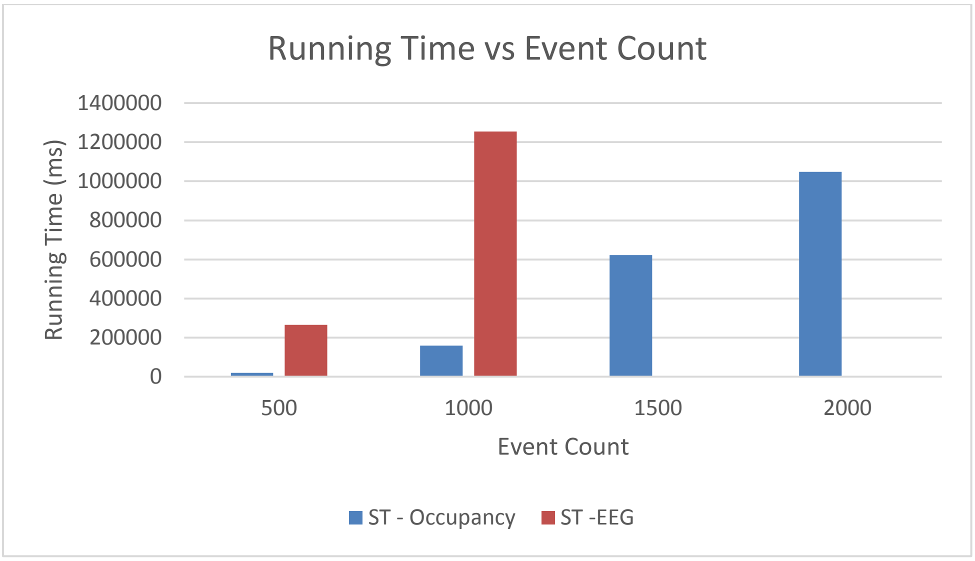
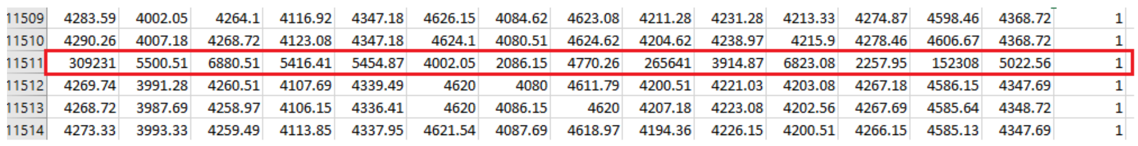


Figure 4.3 ST Algorithm Running Time Comparison

Table 4.8 Accuracy Comparison

Metric	FSS – Occupancy	FSS – EEG	ST - Occupancy	ST -EEG
Recall	99.549%	89.74%	98.28%	97.39%
Precision	98.881%	75.04%	100.00%	100%

Table 4.8 contains the accuracy comparison between shapelet based approach proposed in [15] and the new approach proposed in this paper. Recall of the FSS based solution for occupancy data is nearly 100%. ST based approach has also achieved the same recall percentage. Precision of the FSS algorithm is low when compared to the ST algorithm. During the evaluation, it has been noticed that when outliers exist in the data set accuracy of the output drastically reduced. Since outliers directly impacting the subclass splitting. Data preprocessing needs to be done before using the training data set. EEG data set contains the outlier data points and it was directly impacted to reduce the accuracy.



11509	4283.59	4002.05	4264.1	4116.92	4347.18	4626.15	4084.62	4623.08	4211.28	4231.28	4213.33	4274.87	4598.46	4368.72	1
11510	4290.26	4007.18	4268.72	4123.08	4347.18	4624.1	4080.51	4624.62	4204.62	4238.97	4215.9	4278.46	4606.67	4368.72	1
11511	309231	5500.51	6880.51	5416.41	5454.87	4002.05	2086.15	4770.26	265641	3914.87	6823.08	2257.95	152308	5022.56	1
11512	4269.74	3991.28	4260.51	4107.69	4339.49	4620	4080	4611.79	4200.51	4221.03	4203.08	4267.18	4586.15	4347.69	1
11513	4268.72	3987.69	4258.97	4106.15	4336.41	4620	4086.15	4620	4207.18	4223.08	4202.56	4267.69	4585.64	4348.72	1
11514	4273.33	3993.33	4259.49	4113.85	4337.95	4621.54	4087.69	4618.97	4194.36	4226.15	4200.51	4266.15	4585.13	4347.69	1

Figure 4.4 Outlier of the EEG Data Set

In order to identify the sub classes of the system, mean value of the time series needs to be calculated. Mean value of the time series is calculated by using equation (1,2,3), that has been mentioned in Chapter 3. If the outliers exist as depicted in Figure 4.4, the mean value is directly impacted. In this example values of the outlier records are considerably large when compared to the other values. Therefore, the calculated mean value is not correctly representing the data set. Due to this issue number of subclasses identified by using mean and standard deviation is not correct. As an example, when the system runs with the outliers, 74 sub classes are identified. But when the system run without outliers, 992 sub classes are identified. Once the subclasses are identified, the sample time series form each subclass is picked. Since subclasses are not correctly identified, sample time series is not correctly representing the entire data set. This can be negatively impacting the accuracy of the generated queries.

4.3 Result Summary

Obtained results proved that the ST algorithm-based query generator is not suitable for large data set even though it shows high accuracy. When the data set contains higher number of attributes and large number of data volumes, ST algorithm cannot be used to extract shaplets. The proposed system in this study can be used for large data sets to generate queries as its running time doesn't increase exponentially.

If the automated query generation system is implemented based on the shaplet approach, shaplet extraction algorithm play a key role of the system performance. The system accuracy and the running time heavily rely on the accuracy and time complexity of the shaplet extraction algorithm. In this study, this point has been considered and the FSS is selected as shaplet extraction algorithm. Evaluation results show that FSS algorithm outperform the ST algorithm time complexity wise. But accuracy wise ST algorithm-based approach is little bit ahead when compared to the FSS algorithm-based approach.

CHAPTER 5: CONCLUSION AND FUTURE WORK

5.1 Conclusion

Writing queries for CEP is a challenging task and it requires the knowledge of the domain as well as technical skills to map that knowledge to CEP query language. Automating query generation of the CEP is the best solution to overcome this problem.

During this study different approaches that have been proposed to solve this problem are evaluated. Those approaches have various drawbacks such as being domain specific, computationally inefficient, not supporting multi-variant time series and requiring manual user intervention in defining parameters. This study proposes a shapelet based approach to automate the query generation of the CEP. Many algorithms have been proposed for shapelets extraction and time series classification. During this study different type of shapelet extraction algorithms were evaluated and FSS shapelet extraction algorithm was selected for implementation. FSS shapelet extraction algorithm is the most efficient algorithm among other shapelet extraction algorithms based on the studies done so far.

By using FSS as shapelet extraction algorithm, the issue of being computationally inefficient which exists in most of the proposed algorithms, is resolved. This shapelet extraction algorithm uses the subclass splitting approach and sample shapelets from each subclass. With this approach the number of shapelets which are extract from the data set is reduced. Since the number of shapelets is less, complex works that need to be done to calculate information gain is reduced.

5.2 Future Work

Since the proposed method is based on sampling shapelets from the data set, this method doesn't consider the interdependencies between two adjacent data points. Thus, this system cannot generate time window-based queries. Further study needs to be done to consider the interdependencies between two adjacent data points when sampling the shapelets from the subclasses. The accuracy of the proposed system is directly impacted when outliers exist in the data set. A proper mechanism needs to be implemented to handle outliers before subclass splitting.

According to the proposed approach shapelets are obtained by sampling the subclasses. Further analysis is needed to identify the sample size which improves the accuracy of the queries without increasing the calculation complexity. Chapter 4 present the obtained results by using two data sets. Further evaluation needs to be done with different time series data set to benchmark the accuracy of the system.

References

[1] G. Cugola, and G. Tamburrelli A. Margara, "Learning From the Past: Automated Rule Generation for Complex Event Processing," in DEBS '14 Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems, 2014, pp. 47-58.

[2] Margara, A., Cugola, G., Tamburrelli, G.: Towards automated rule learning for complex event processing (2013). Technical Report

[3] A. Gal, and S. Wasserkrug Y. Turchin, "Tuning complex event processing rules using the prediction-correction paradigm," in DEBS '09 Proceedings of the 3rd ACM International Conference on Distributed Event-Based Systems, 2009.

[4] M. Philippsen and C.Mutschler, “Learning Event Detection Rules with Noise Hidden Markov Models,” in Proc. 2012 NASA/ESA Conference on Adaptive Hardware and Systems (AHS-2012), 2012.

[5] S. Sen, N. Stojanovic, and L. Stojanovic, "An approach for iterative event pattern recommendation," in DEBS '10 Proceedings of the 4th ACM International Conference on Distributed Event-Based Systems, 2010.

[6] Y. T. R. Mousheimish, K. Zeitouni, "Complex event processing for the non-expert with autocep", in Proc. 10th ACM Intl. Conf. on Distributed and Event-based Systems, 2016.

[7] R.N. Navagamuwa, K.J.P.G. Perera, M.R.M.J. Sally,L.A.V.N. Prashan, and H.M.N.D. Bandara, “Shapelets and Parallel Coordinates Based Automated Query Generation for Complex Event Processing,” in Proc. 2016 IEEE Intl. Conf. on Smart Data (SmartData ‘16), Chengdu, China, Dec. 2016.

[8] L. Ye, E.Keogh, "Time series shapelets: a new primitive for data mining," in Proceedings of the 15th ACM SIGKDD international conference on Knowledge

discovery and data mining,2009 ACM. pp. 947–956.

[9] R.N. Navagamuwa, K.J.P.G. Perera, M.R.M.J. Sally,L.A.V.N. Prashan, and H.M.N.D. Bandara, “Shapelets and Parallel Coordinates Based Automated Query Generation for Complex Event Processing,” in Proc. 2016 IEEE Intl. Conf. on Smart Data (SmartData ‘16), Chengdu, China, Dec. 2016, pp. 846-853

[10] Mueen, A., Keogh, E., Young, N., "Logical-shapelets: an expressive primitive for time series classifcation," in DEBS Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM,2011, pp. 1154–1162.

[11] Rakthanmanon, T., Keogh, E., 2013. "Fast shapelets: A scalable algorithm for discovering time series shapelets," in: Proceedings of the 2013 SIAM International Conference on Data Mining, SIAM,2013,pp. 668–676.

[12] Ye, L., Keogh, E., 2011. "Time series shapelets: a novel technique that allows accurate, interpretable and fast classification." in Data mining and knowledge discovery 22, 149–182.

[13] Hills, J., Lines, J., Baranauskas, E., Mapp, J., Bagnall, A., 2014. Classification of time series by shapelet transformation. Data Mining and Knowledge Discovery 28, 851–88

[14] Grabocka,J.,Schilling,N.,Wistuba,M.,Schmidt-Thieme,L., "Learning time-series shapelets," in: Proceedings of the20th ACMSIGKDD international conference on Knowledge discovery and data mining, ACM,2014, pp. 392–401.

[15] C. Ji, S. Liu, C. Yang, L. Pan, L. Wu, X. Meng A shapelet selection algorithm for time series classification: new directions Procedia Comput. Sci., 129 (2018), pp. 461-467

[16] Xing, Z., Pei, J., Yu, P.S., Wang, K.,"Extracting interpretable features for early classification on time series," in: Proceedings of the 2011 SIAM International Conference on Data Mining, SIAM,2011, pp. 247–258.

[17] Yuelong Zhu, De Wu and Shijin Li, A Piecewise Linear Representation Method of Time Series Based on Feature Points, Lecture Notes In Computer Science, Vol. 4693, 2007, pp. 1066-1072

[18] Shatkay, H., Zdonik, S. B.: Approximate queries and representations for large data sequences. In Proceedings of the Data Engineering, 1996

[19] Ji, C., Liu, S., Yang, C., Wu, L., Pan, L., Meng, X.,"A piecewise linear representation method based on importance data points for time series data,"in:2016 IEEE 20th International Conference on Computer Supported Cooperative Work in Design(CSCWD),IEEE,2016,pp.111–116.

[20] C. Ji, S. Liu, C. Yang, L. Pan, L. Wu, X. Meng A shapelet selection algorithm for time series classification.

[21] D.Gordon, D.Hendler, L.Rokach, Fast and space-efficient shapelets-based time-series classification, Intell.Data Anal.19(5)(2015)953–981.

[22] X.Renard, M.Rifqi, W.Erray, M.Detyniecki, "Random-shapelet: an algorithm for fast shapelet discovery," in: 2015 IEEE International Conference on Data Sci-ence and Advanced Analytics, IEEE, 2015,pp.1–10.

[23]I.Karlsson, P.Papapetrou, H.Boström, Generalized random shapelet forests, Data Min.Knowl.Discov.30(5)(2016)1053–1085.

[24] Bagnall, A., Lines, J., Vickers, W., Keogh, E., 2016c. The uea & ucr time series classification repository. URL www.timeseriesclassification.com