ENSEMBLE-BASED FAST SHAPELET APPROXIMATION

by

Berk Görgülü

B.S., Industrial Engineering, Boğaziçi University, 2017

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Industrial Engineering
Boğaziçi University

2018

ACKNOWLEDGEMENTS

First and foremost, I would like to express my sincere gratitude to *Mustafa Gökçe Baydoğan* for being my thesis advisor, guiding me throughout all the phases of this study as well as many others. His guidance helped me to discover what I really want to do in my future studies. He always considered my opinions and offered me great research opportunities. I felt his trust throughout our research and in every conversation we had. He has not been only an advisor, but he has also been a mentor and a friend. I am grateful for his friendship and all of his support.

My sincere thanks also goes to *Gönenç Yücel* who provided me an opportunity to join SESDYN, and always supported me during my undergraduate and graduate studies. Working with him on various research activities has been an honor and very valuable experience. I always felt his confidence in me. His door was always open when I needed help. Without his precious support, it would not be possible to conduct this and many other research.

I would like to thank *Enis Kayış* for taking part in my thesis committee and providing valuable comments.

I would like to thank all of the professors and staff in Industrial Engineering department for great 6 years. I especially thank *Yaman Barlas* for everything he has taught me during my studies and assistantship. I admire his hardworking character, knowledge and intellectual capacity. Every academic or non-academic conversation with him has been very influential.

I must express my greatest gratitude to my family. I want to thank to my mother *Evlin* and my father *Aşkın* for their lifelong emotional and physical effort that they put for me. I won't be able to stand where I am now without their never ending support. And most importantly, I want to thank my best friend and my life partner *Billur* for her understanding, patience, love and all happiness she brought into my life.

I am very lucky to have strong and beautiful friendships. I want to thank my dearest friends *Süleyman* and *Serkan* for their unconditional support and joy they brought into my life for more than 10 years. I also want to thank *Osman*, *Erdem*, *Berkay*, *Alperen* and *Cafer* for all the good times we spent. This process would not be this fun without you. I want to thank *Dilara* for her support throughout my undergraduate studies and for her sincere friendship. Lastly, I want to thank *Özberk*, *Baturay* and *Beril* for making my Boğaziçi years memorable.

Furthermore, I am proud to be a member of SESDYN and I would like to thank all SESDYN members. I want to thank *İpek Dursun* for her friendship, her support in my research, and for her food and water stock that I looted. I also want to thank *Oylum Şeker* for being the best teaching assistant, welcoming me to SESDYN and helping me when I needed.

Lastly, I would like to thank the Scientific and Technological Research Council of Turkey (TÜBİTAK) for providing the financial means throughout this study.

ABSTRACT

ENSEMBLE-BASED FAST SHAPELET APPROXIMATION

Similarity search and classification on time series databases have received great interest over the past decade. The definition of similarity between the series is a major problem in this context. Nearest-neighbor (NN) classifiers with alternative dissimilarity measures are widely used because of their simplicity and known success. However, these approaches compute the similarity over the whole time series which might be problematic with the long time series and relatively short features of interest. Moreover, NN classifiers are not directly interpretable as they do not describe why a time series is assigned to a certain class. To overcome these problems, researchers focus on discovering discriminative subsequences, namely shapelets, from the time series. Instead of pairwise distance calculations between the whole time series, shapelet-based approaches map time series to a feature vector based on the existence of the shapelets. In the recent years, shapelet discovery approaches have focused on the evaluation of the segmented subsequences in terms of their discriminative power. As this approach may be time-consuming depending on the size of the time series database, recent attempts exploit the change of time-series representations for faster discovery of shapelets. In this sense, piecewise constant approximations are shown to provide significantly faster results with a low-dimensional representation. This study proposes a novel supervised piecewise approximation to identify shapelets related to the class. After utilizing a simple piecewise linear model to characterize the time series, the segments from the model are determined to be potential candidates for shapelets. Proposed piecewise approximation scheme is notably different than the traditional methods. Ensembles of regression trees are utilized to learn a piecewise approximation to identify the shapelets in a supervised manner. Experimental results show that proposed Ensemble-based Fast Shapelet Approximation (EFSA) provides fast and competitive results on benchmark datasets from different domains.

ÖZET

TOPLULUK TEMELLİ ŞEKİLCİK YAKLAŞIKLAMA

Son yıllarda, zaman serisi veri tabanlarında benzerlik araştırması ve zaman serilerinin sınıflandırılması büyük ilgi görmektedir. Bu bağlamda zaman serileri arasındaki benzerliğin tanımlanması ve ölçülmesi önemli bir hal almıştır. En yakın komşu (NN) sınıflandırıcıları, başarısı ve sadeliği sebebiyle yaygın olarak kullanılmaktadır. Ancak bu sınıflandırıcılar, benzerlik hesaplamalarını zaman serilerinin tümü üzerinden yaptığından, kısa ilgi alanları olan uzun zaman serilerinde iyi performans gösteremeyebilir. Ayrıca, NN sınıflandırıcıları bir zaman dizisinin neden belirli bir sınıfa atanmış olduğu bilgisini de sağlamaz. Bu problemlerin üstesinden gelmek için, zaman serisi sınıflarını birbirinden ayıran ayırımcı alt serileri, yani şekilcikleri keşfetmeye odaklanılmıştır. Şekilcik bazlı yöntemler benzerlik hesaplamak yerine, zaman serilerinin bu şekilcikleri içlerinde barındırıp barındırmadığını araştırır ve zaman serilerini şekilciklerin varlığına dayalı bir öznitelik vektörüne dönüştürür. Son yıllarda, şekilcik keşif yöntemleri, şekilciklerin ayırıcı özelliklerini ölçme odaklıdır. Bu yöntemler, zaman serisi veri tabanlarının büyüklüğüne göre çok zaman alabileceğinden, şekilciklerin daha hızlı keşfine olanak sağlayan farklı gösterimler bulmaya yönelik çalışmalar önem kazanmıştır. Bunlar arasında, parçalı sabit yaklaşıklama yöntemlerinin çok daha hızlı sonuçlar sağlayan düşük boyutlu gösterimler oluşturduğu gözlenmiştir. Bu çalışmada, sınıfları tanımlayan şekilcikler oluşturmak için eğitmenli parçalı yaklaşıklama yöntemi önerilmektedir. Bu yöntem zaman serilerini sınıflandırmak için basit bir parçalı doğrusal yaklaşıklama modeli kullanılarak potansiyel şekilcikler belirler. Önerilen metot, geleneksel yöntemlerden farklı olmamakla birlikte parçalı bir yaklaşıklama yaparak şekilcikleri eğitmenli bir şekilde oluşturmak için regresyon ağaç toplulukları kullanır. Deney sonuçları, önerilen topluluk temelli hızlı şekilcik yaklaşıklama (EFSA) metodunun farklı alanlardaki referans veri setlerinde hızlı ve rekabetçi sonuçlar sağladığını göstermektedir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

c_n	Class label of time series n
D_t^n	A univariate time series n
$D(x^n, \Psi_j^l)$	Best matching distance between time series x^n and shapelet Ψ_j^l
DD_t^n	A univariate time series n
I_{max}	Maximum length of the shapelets
I_{min}	Minimum length of the shapelets
J_p	Number of trees in the random forest classifier
k	Number of unique classes in a time series database
N	Number of univariate time series
Q_j^l	Length of the shapelet Ψ_j^l
S	Shapelet set
t	Time
T_n	Number of observations in time series n
x^n	A univariate time series n
x_t^n	Observation of time series n on time t
X	Time series database
γ	Number of sampling in the shapelet generation procedure
η	Number of shapelets
Ψ_j^l	Shapelet generated from regression tree j terminal node l

LIST OF ACRONYMS/ABBREVIATIONS

APCA	Adaptive Piecewise Constant Approximation
BMD	Best Matching Distance
BMS	Best Matching Segment
BMT	Best Matching Time
Diff	First order difference
DDiff	Second order difference
DTW	Dynamic Time Warping
ECG	Electrocardiography
EFSA	Ensemble Based Fast Shapelet Approximation
EFSA1NN	Ensemble Based Fast Shapelet Approximation with 1 Nearest Neighbor
FS	Fast Shapelet
GRSF	Generalized Random Shapelet Forest
LS	Learned Shapelet
NN	Nearest Neighbor
NNDTW	Nearest Neighbor Dynamic Time Warping
Obs	Observations
OOB	Out of Bag
PLA	Piecewise Linear Approximation
SAX	Symbolic Aggregate Approximation
SSE	Sum of Squared Errors
ST	Shapelet Transform
TS	Time Series

1. INTRODUCTION

Time series data mining has recently been drawing major attention due to its numerous important applications including finance, science, medicine and multimedia. A time series is a set of sequential observations. Time series data is usually large and high dimensional with continuous updates. In addition, its sequential nature requires time series to be considered as a whole rather than collection of individual features [1]. Therefore, time series data mining aims to utilize tools that take this temporal relationship into account. Time series are grouped as univariate or multivariate time series according to the dimensionality; and numeric or categorical according to the type of their observations.

The main challenge behind all time series data mining tasks is finding a suitable representation with reduced dimensionality for the time series. Many researchers have come up with various representation schemes for different types of data mining tasks. These techniques can be grouped as approximation techniques, salient points discovery in the time series and removal of the unimportant parts, categorization of the time series for dimensionality reduction and utilization of transformation techniques to come up with a new representation [1].

Time series data mining can be analyzed under four tasks: Pattern discovery and clustering, classification, rule discovery and summarization [1]. Among these four tasks, the major ones can be considered as clustering and classification tasks. Each time series contain a sequential observations, and these sequential observation may exhibit some common or distinguishing patterns. Discovery of these patterns constitutes the basis for both tasks.

Clustering can be defined as finding a structure in a collection of unlabeled data and creating clusters (groups) that contain similar observations. Therefore, measuring the similarity between each observation yields a great importance. Due to sequential nature of the time series, similarity measurement in time series carries additional diffi-

culties which causes time series clustering task interesting. For example, similar time series are expected to demonstrate similar patterns, however these patterns can shift in time and magnitude. Discovery and definition of these patterns requires extensive work. Detailed review of the works has been provided by Fu [1].

Classification, i.e. assigning a class label to a set of unclassified cases using a prelabeled training set, constitutes the primary goal in many time series data mining applications. For example, a cardiologist might be interested in analysis of electrocardiography (ECG) signals to identify whether patients have different temporal pattern in their heart signals than a control group and classify them accordingly [2] or seismologists aim at discriminating the nature of the seismic waves to classify events such as earthquakes, mining explosions or nuclear explosions [3].

Previous time series classification methods can be categorized into instance-based and feature-based approaches. Instance-based approaches focus on similarity of a test instance to the training instances and assign labels based on the labels of similar training instances with a nearest-neighbor (NN) classifier [4, 5]. Distance measure selection plays the most important role in the performance of a NN classifier. Most frequently used distances include Euclidean and Dynamic Time Warping (DTW) distance. The popularity of the Euclidean distance is mainly due to its simplicity, and it performs well for certain applications as the training data size increases [6]. However in general, it is sensitive to noise, scaling, translation and dilation of the patterns within the time series. As a more complex distance measure, Dynamic Time Warping distance has been widely, and successfully used with instance-based classifiers, such as nearest neighbor classifiers (e.g. NNDTW [7–11], DTW [12]). DTW provides a measure of similarity independent of certain non-linear variations in the time dimension, and is considered as a strong solution for the problems related to time series [13]. However, instance-based approaches require training data to be stored throughout the classification process, which damages the efficiency of classification due to the time and memory requirements. Moreover, understanding what exactly relates to the class is not clear when a NN classifier is used. Such information may be crucial in many application areas. For example, in the health domain, it is important for a medical specialists to

understand the underlying reasons of a certain diagnosis to be provided by a classifier. In that respect, the interpretability of classifiers to be used in this domain is of primary importance for their acceptance and usability.

Feature-based approaches transform the time series observations to a new representation (i.e. feature vectors) and train traditional classifiers (e.g. decision trees). These approaches are in general interpretable, and computationally more efficient than instance-based classifiers. However, determining which features to extract is a very important and challenging step. Features such as wavelet coefficients and interval statistics have been widely used in earlier studies [14, 15]. These approaches have potential to fail for the cases in which a pattern that defines a certain class shifts over time [16]. In other words, a feature-based classifier can identify the patterns in the training instances, and classify this set well. However, it may fail for a test instance which has a pattern at a different location when a feature vector of fixed dimension is used.

Recently, researchers have concentrated on the extraction of interpretable patterns to classify large time series databases as a midway approach to instance-based and feature-based classifiers [16–25]. An approach to find subsequences of the time series which are thought to be maximally representative of a class was first proposed by Ye *et al.* [24, 25] and these interpretable subsequences are called shapelets. Given a good shapelet set, shapelet-based classifiers outperform instance-based and feature-based approaches. The information provided by shapelets is limited to their presence or absence which makes shapelet discovery the most important phase. Most of the recent shapelet discovery approaches have focused on the evaluation of the subsequences of the time series in terms of their discriminative power. For example, Shapelet Transform [16] and Generalized Shapelet Forest [19] methods find shapelets by pruning the subsequence space according to the information gain. Pruning of the subsequence space requires evaluation of exponential number of possible subsequences and discovery of the relationships between these subsequences. To reduce computational complexity, piecewise approximation methods have been proposed for shapelet discovery [21, 26]. After representing time-series in a low-dimensional space with piecewise approxima-

tion, search for shapelets is performed on the new space. This way, computational burden for shapelet discovery reduces significantly. For example, Fast Shapelets [21] use "symbolic aggregate approximation" (SAX) [14] to generate shapelets efficiently.

Existing approximation-based shapelet discovery approaches perform transformation to low-dimensional space without use of the class information. After obtaining the representation for each series, evaluation of candidate shapelets are performed as in traditional approaches [14, 20]. This approach has potential to miss certain important shapelets depending on the approximation parameters. For example, SAX requires the alphabet size and desired dimensionality parameter [14]. Alphabet size affects the number of distinct mean levels to consider in a piecewise constant approximation setting where desired dimensionality determines the number of segments to consider in the approximation. Figure 1.1 illustrates the piecewise constant approximation by SAX. Although approximation methods reduce the computational burden, they are not learned with the objective of improving the classification performance in the approximation stage. This potentially decreases the quality of shapelet candidate set and yields relatively low classification performance compared to other discovery methods. To address this problem, Grabocka *et al.* [17] proposes a shapelet learning framework. Without performing approximation, shapelets are learned in an optimization framework in this approach. However, the method requires selection of many parameters which damages the time complexity.

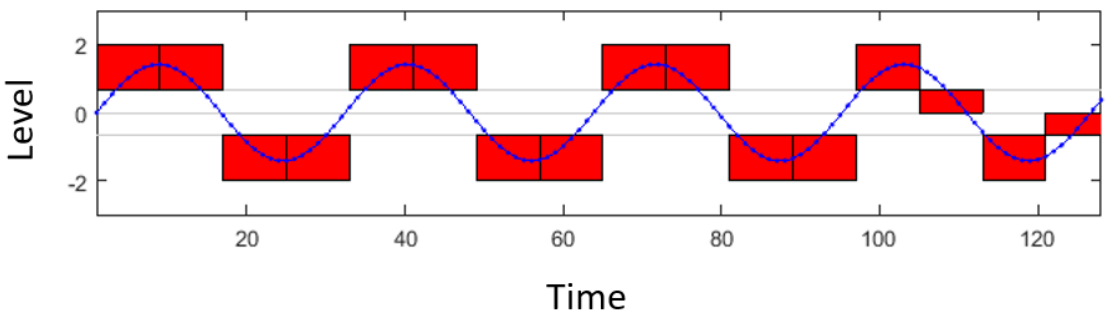


Figure 1.1. Sample SAX representation for an alphabet size of four and desired dimensionality of 16. The time series length is 128. Horizontal grids differentiate the mean levels implied by each symbol and red boxes visualizes symbol assignments.

This work proposes a novel supervised piecewise approximation to identify shapelets related to a class. After utilizing a simple piecewise linear model to characterize the time series, the segments from the model are determined to be potential candidates for shapelets. Suggested approach, namely Ensemble-based Fast Shapelet Approximation (EFSA) is notably different than the traditional approximation approaches. Ensembles of regression trees are utilized to learn a piecewise approximation to identify the shapelets in a supervised manner. Experiments demonstrate the effectiveness of the proposed approach in terms of accuracy and computation times on a full set of benchmark datasets from Bagnall *et al.* [27].

2. BACKGROUND

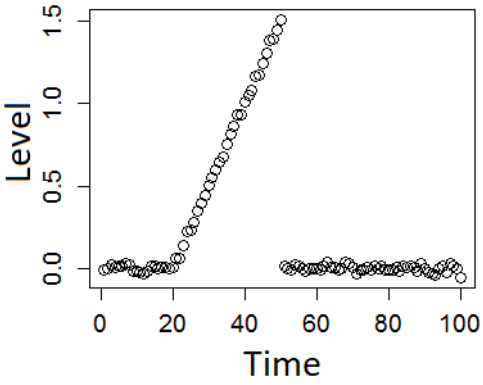
Approximation using regression trees is the core element of our shapelet discovery procedure. This chapter provides some background information about tree-based piecewise approximations. Before going into details of piecewise approximation, it is required to define some concepts.

A univariate time series, $x^n = (x_1^n, x_2^n, \ldots, x_t^n, \ldots, x_{T_n}^n)$ is an ordered set of T_n values. It is assumed that time series are measured at equally-spaced time points. A time series database, X, stores N univariate time series. Note that, the time series can be of different lengths, namely T_n and equally-spaced time points assumption is not restrictive for the proposed approach. Each time series is assigned to a class c_n, $c_n \in \{1, 2, \ldots, k\}$ and k is the number of unique classes in X.

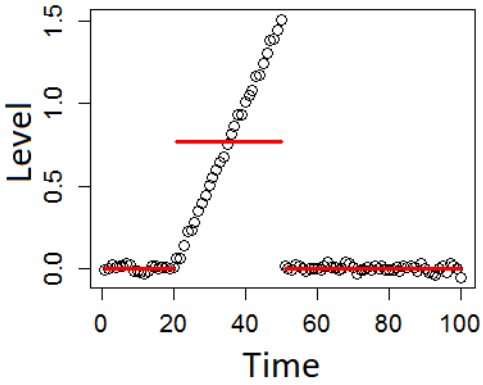
2.1. Piecewise Approximations

Time series representation has been a very active research topic due to its importance in efficient mining of the time series databases. Detailed categorization and description of these approaches can be found in Ratanamahatana *et al.* [28]. Among those, piecewise approximations are commonly used to represent the local characteristics of time series. These approaches segment the time series into intervals and fit a polynomial function of a certain degree for each interval. The most prominent representation of the time series intervals are obtained by constant and linear functions [8].

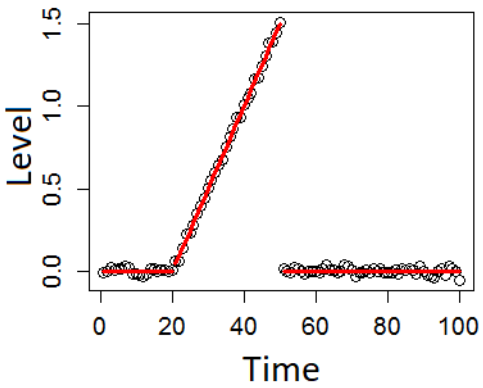
Piecewise constant approximations such as Adaptive Piecewise Constant Approximation (APCA) [26] or Symbolic Aggregate Approximation (SAX) [14] segment the series into intervals and represent each interval with the mean of the observations. A sample adaptive piecewise constant approximation is illustrated in Figure 2.1(b). The other option is to segment the time series and learn a Piecewise Linear Approximation (PLA) [8] as illustrated in Figure 2.1(c). Piecewise linear approximation has been used to solve several time series data mining problems [8] because of its simplicity.



(a) Original time series.



(b) Piecewise constant approximation (red) of the original time series.



(c) Piecewise linear approximation (red) of the original time series.

Figure 2.1. Piecewise approximation examples: Original time series Figure 2.1(a), original time series with piecewise constant approximation Figure 2.1(b) and original time series with piecewise linear approximation Figure 2.1(c).

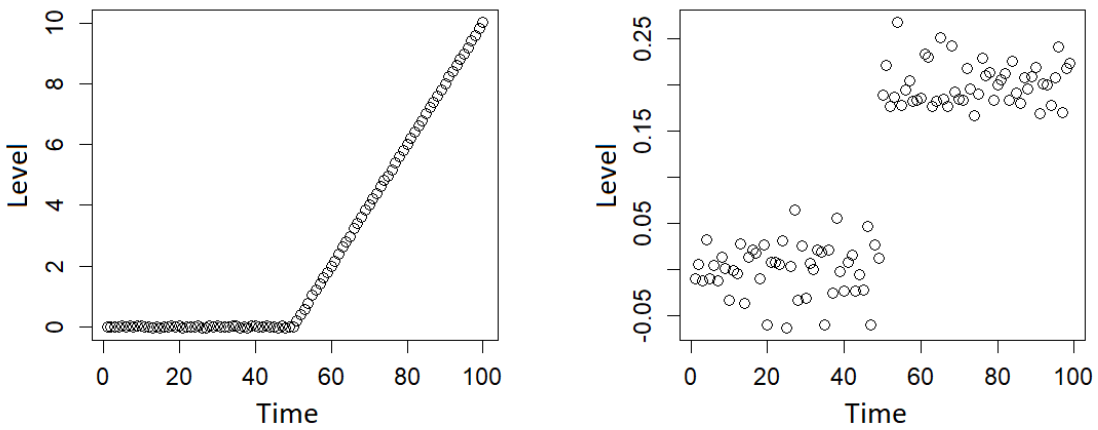
2.2. Piecewise Approximation with Regression Trees

In previous studies, regression trees [29] are used to approximate a piecewise constant model for the time series [30]. These approaches aim to minimize the sum of squared error (SSE) of the constant approximation by using the time index t as the

predictor and the observations x_t^n as the target [30]. For example, a sample regression tree fitted to time series in Figure 2.2(a) provides the constant approximation in Figure 2.3(a). Terminal nodes of the trained regression tree in Figure 2.3(b) implies this constant approximation.

Regression trees are flexible in terms of learning the trend and curvature information with a simple change of time series representation. In other words, it is possible to obtain a piecewise approximation of the time series in second degree. From the time series demonstrated in Figure 2.2(a), it can be observed that time series carry a clear trend when time is larger than 50 and constant approximation is not representative enough.

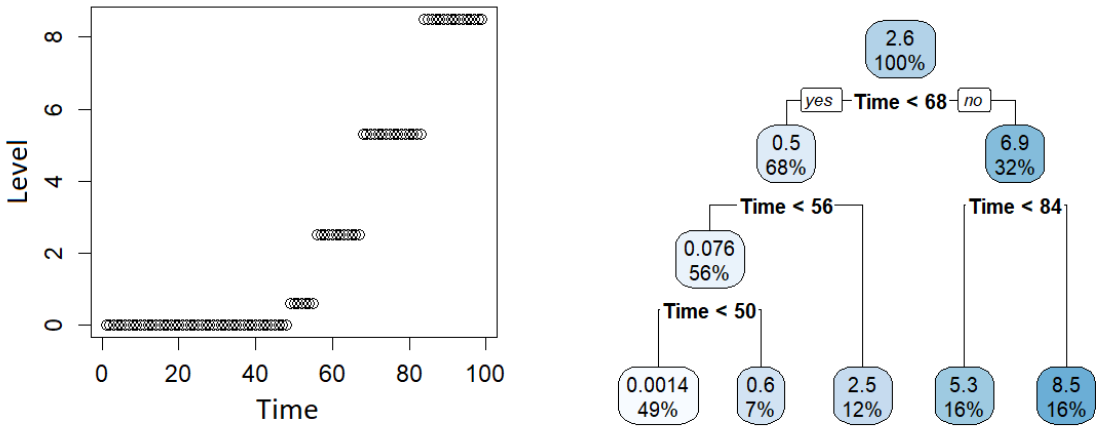
Suppose that the time series representation is changed to a new series illustrated in Figure 2.2(b) by taking the difference of consecutive observations. The new time series is an empirical approximation for the first order derivative. The terminal nodes of the regression tree trained on this new representation provided in Figure 2.4(b). Merging the mean and trend information obtained from both trees, piecewise linear approximation of the time series is obtained and it is illustrated in Figure 2.4(a).



(a) An example time series.

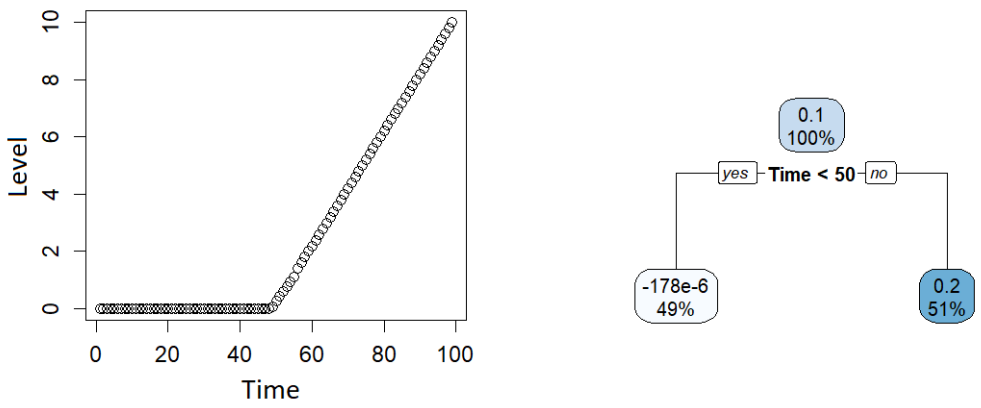
(b) First order difference of time series.

Figure 2.2. Example time series and its first order difference.



(a) Constant approximation of the time series. (b) Visual representation of the regression tree.

Figure 2.3. Constant approximation of the time series provided in Figure 2.2(a) and the stucture of the regression tree trained on the same time series.



(a) Linear approximation of the time series. (b) Visual representation of the regression tree.

Figure 2.4. Linear approximation of the time series provided in Figure 2.2(a) and the stucture of the regression tree trained on the first order difference of the time series provided in Figure 2.2(b).

Higher order polynomial approximations can be approximated by further differencing the time series observations and aggregating the information from the regression trees. Computational complexity of learning a regression tree on T_n observations is $O(T_n log(T_n))$ which makes it very efficient compared to existing piecewise polynomial approximation approaches.

3. RELATED WORK

There are various studies about discovering the best shapelets and using them in the classification phase [16, 17, 19–22, 24, 25]. These studies aim at finding the representative subsequences of the time series to be used in supervised learners. Among the aforementioned studies, the state of the art shapelet-based classification approaches are Learned Shapelets [17], Shapelet Transform [16], Generalized Random Shapelet Forest [19] and Fast Shapelets [21]. In order to demonstrate the performance of the method introduced in this work, it is compared to these state-of-the-art methods. Therefore, in this section, these state-of-the-art methods will be briefly introduced.

3.1. Learned Shapelets

Learned Shapelets (LS) [17], provides an algorithm that learns shapelets by using a non-linear optimization method. Instead of constructing the shapelet set by searching the complete subsequence space, LS generates its continuous valued shapelet set by minimizing a logistic loss function. Optimization of this logistic model is realized iteratively using a gradient descent algorithm. LS learns shapelets and optimal decision boundary simultaneously in an optimization framework. This allows finding shapelets that are not part of the original time series and helps to prevent over-fitting. However with large data sizes non-linear optimization becomes computationally heavy and this damages the time performance of the algorithm.

3.2. Shapelet Transform

Hills *et al.* [16] tries to construct best decision tree by enumerating all possible shapelets in certain length intervals. For each time series and for each length in a given interval, all possible shapelet candidates are generated. These candidates then are assessed according to their discriminative properties using their distance to each of the time series. For each length, a single best subsequence is selected and stored using the assessment. Then similar subsequences are eliminated in order to prevent repetition.

By repeating this procedure for each time series and merging the subsequences, shapelet set is generated. These shapelet set is then used to come up with a new representation of time series dataset by measuring the distance of all shapelets to all time series. Using this new representation, ST constructs a classifier using weighted ensemble of standard classifiers.

3.3. Generalized Random Shapelet Forest

Generalized Random Shapelet Forest (GRSF) by Karlsson *et al.* [19] aims to find best shapelet set by pruning the subsequence space. For a given minimum and maximum subsequence length and number of shapelet candidates, a subsequence set is generated. By calculating the distance of each subsequence in the subsequence space to the time series, predetermined number of trees are constructed considering information gain from each split. GRSF algorithm requires tuning of some parameters such as minimum length, maximum length and sample size. Parameter selection is computationally heavy process which makes the algorithm computationally inefficient as data size increases.

3.4. Fast Shapelets

To reduce the complexity, there are some algorithms that use piecewise approximations to generate shapelets. Fast Shapelets (FS) [21] uses unsupervised constant approximation by transforming time series into symbolic representation that is called "symbolic aggregate approximation"(SAX) [14]. By using this symbolic approximation, FS decreases the dimension of the time-series and makes the classification process computationally more efficient.

EFSA is very similar to Fast Shapelets since it also uses approximation to generate shapelets. However, EFSA uses a supervised approximation method rather than unsupervised one, in other words class labels are used as a predictor to approximate shapelets in EFSA. Also rather than constant approximation, EFSA uses a shapelet discovery method that creates shapelets with a polynomial of second degree.

4. ENSEMBLE-BASED FAST SHAPELET APPROXIMATION

The objective of the study is to utilize a piecewise representation for discovery of the regions of the time series with distinct class information. This discovery relies on the results of regression trees. The patterns implied by the terminal nodes of the regression trees are used as a reference to transform the observation space to a distance space based on the distance between each pattern and the time series. Any supervised learner can be trained on the transformed features to find the labels of the time series. Tree-based ensembles (namely a random forest classifier) are utilized because of its known success in classification. Moreover, the structure of the classifier allows to identify important patterns.

4.1. Shapelet Discovery

A shapelet, denoted as Ψ_j^l, is a vector of values obtained from the linear model implied by the terminal node l of a regression tree j. Discovery of shapelets can be considered as the most essential part of the shapelet based classification methods. Their similarity with the original time-series are used as an input to the various classifiers depending on the method. Therefore, discovering the best representing shapelet set directly affects the ability to classify time-series correctly. Before proceeding to the shapelet discovery procedure, set of time-series is transformed into a new feature representation in which each observation has corresponding class label and time of the observation. Table 4.1 illustrates an example of transformed feature representation.

Shapelet discovery process of EFSA utilizes a supervised second order tree-based approximation on transformed feature representation. Unlike the unsupervised approximation procedures that only use time index t as predictor, EFSA proposes a supervised tree-based approximation procedure which predicts the observations x_t^n using not only the time index t but also the class labels c_n.

The first step of the discovery process aims to capture distinct mean levels between different classes of time series. In order to capture distinct mean levels, a regression tree is trained on the set of time series by predicting the observations x_t^n using the class labels c_n and time index t as predictors. Regression model tries to split the observations using the class label c_n and time index t in order to minimize the sum of squared error. Resulting terminal nodes contain varying number of consequent observations. By taking the mean of the observations with respect to the class labels c_n in each terminal nodes, constant approximation of the time-series can be obtained.

Usually the shapelets obtained by constant approximation carry exclusive information about the class of time series. However, a set of subsequences obtained from different classes can have same constant approximation, even though they demonstrate different dynamics due to their trend and curvature. In order to capture information provided by the trend and curvature additional steps are taken by EFSA.

In the second step, EFSA aims to capture important trend information related to each class of time-series. To achieve this, another regression tree is trained on the difference levels $D_t = Y_t - Y_{t-1}$ of the same time series. This tree is utilized to

Train a regression tree $\boldsymbol{\tau_1}$ on the data to predict observation values $\boldsymbol{x_t^n}$ using class $\boldsymbol{c_n}$ and time index $\boldsymbol{t}$.
Generate mean shapelets by using terminal nodes of $\boldsymbol{\tau_1}$.
Train a regression tree $\boldsymbol{\tau_2}$ on the first order difference data to predict observation values $\boldsymbol{D_t^n = x_t^n - x_{t-1}^n}$ using class $\boldsymbol{c_n}$ and time index $\boldsymbol{t}$.
Generate trend shapelets by using terminal nodes of $\boldsymbol{\tau_2}$.
Train a regression tree $\boldsymbol{\tau_3}$ on the second order difference data to predict observation values $\boldsymbol{DD_t^n = D_t^n - D_{t-1}^n}$ using class $\boldsymbol{c_n}$ and time index $\boldsymbol{t}$.
Generate curvature shapelets by using terminal nodes of $\boldsymbol{\tau_3}$.
Form shapelet set $\boldsymbol{S}$ by combining the mean shapelets, trend shapelets and curvature shapelets.
Eliminate shapelets shorter than $\boldsymbol{I_{min}}$ and longer than $\boldsymbol{I_{max}}$.

Figure 4.1. Pseudocode of the three step shapelet discovery procedure.

predict the differences in the observations D_t^n using again class labels c_n and time t as predictors. By taking the mean of the observations in the terminal nodes with respect to their class labels, meaningful trend information related to each class is being discovered.

In the last step, same procedure is applied to the second order difference of the time-series in order to approximate the curvature. Curvature discovery is realized again by utilizing a regression tree. Regression tree is trained to predict the second order differences $DD_t = D_t - D_{t-1}$ using class labels c_n and time parameter t. Mean of the observations with respect to classes is used as curvature information.

Table 4.1. Transformed representation of the time series observations into a single matrix.

Time	**Class**	**Obs**	**Diff**	**DDiff**
1	1	-1.7463	0.0136	0.0050
2	1	-1.7413	0.0055	0.0186
3	1	-1.7227	-0.0027	0.0241
4	1	-1.6986	0.0254	0.0214
5	1	-1.6772	0.0042	0.0468
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
1	2	-1.7170	0.0559	-0.0111
2	2	-1.7281	-0.0283	0.0448
3	2	-1.6833	0.0035	0.0165
4	2	-1.6668	0.0353	0.0200
5	2	-1.6468	-0.0276	0.0553
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.

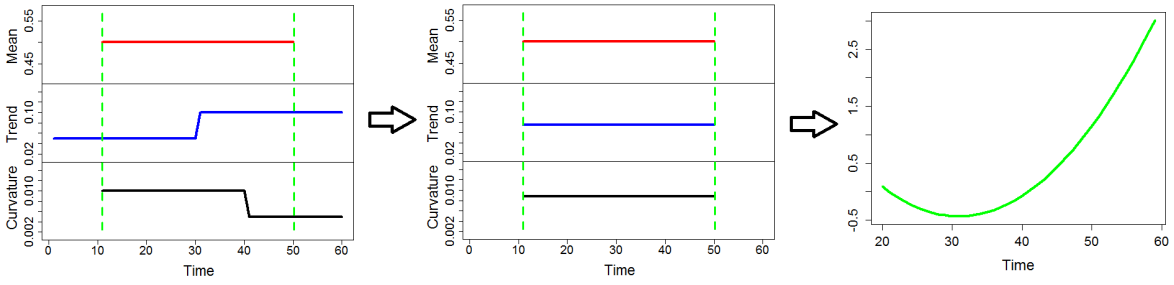


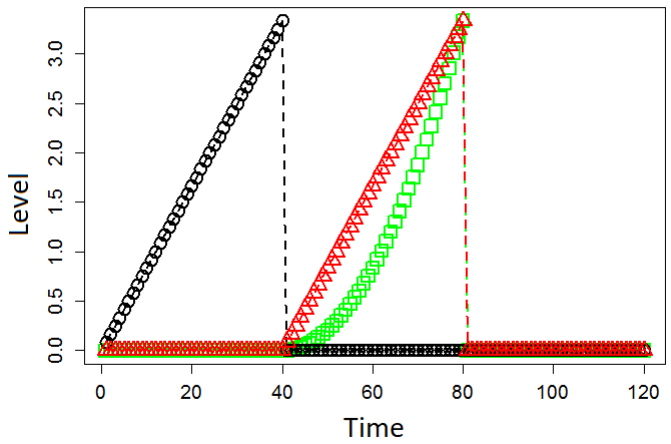
Figure 4.2. Shapelet generation by merging mean, trend and curvature informations.

Approximated mean, trend and curvature informations are combined to create complete shapelets with possibly first or second order polynomial. Merging is realized by updating the constant subsequences approximated in the first step with corresponding trend and curvature, therefore mean levels determine the length of the shapelets. As illustrated in Figure 4.2, for each mean level, overlapping trend and curvature information is averaged separately in order to obtain single trend and curvature values. These mean, trend and curvature levels are used to generate a second order shapelet with corresponding mean, trend and curvature.

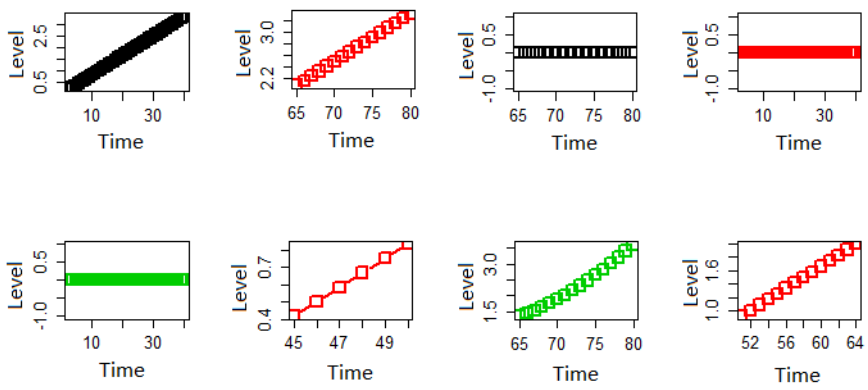
Supervised shapelet generation, in other words, using class labels in approximation allows discovering subsequences that separates each class of time series from the others. Consider the example provided in Figure 4.3. Shapelet discovery procedure is applied on the time-series from three different classes (Figure 4.3(a)) as a demonstration. Figure 4.3(b) demonstrates the subsequences generated by the approximation method. It can be observed that these generated subsequences represent the distinguishing parts of each class of time-series.

Suggested approach applies shapelet discovery procedure γ times to randomly selected time series. This selection is done in a stratified manner for which one time series from each class is selected randomly. Shapelet discovery procedure is then applied on the randomly selected time series. This provides certain benefits for the algorithm. First of all, working on the subset of data is computationally more efficient. Also, when all time series are used to train models, it becomes harder to identify the shapelets

because of the translations over time or potential dilations. For example, two time series from the same class may exhibit different behavior in the same time segment. As regression tree splits are only based on the class and time information (which are the same for this particular example), they will fail to find an appropriate model. Hence, random selection of one time series from each class allows EFSA to avoid this particular problem.



(a) Time series from three different class.



(b) Shapelets generated by three step shapelet discovery procedure.

Figure 4.3. Example time series from three different class and shapelets generated from these time series by using three step shapelet discovery procedure.

Shapelet set S is constructed by combining all generated shapelets. Shapelets that are longer than I_{min} and shorter than I_{max} are used in classification. I_{min} and I_{max} do not require any parameter optimization since they are not for determining the subsequence length. Tree-based piecewise approximation is capable of constructing subsequences with varying length therefore I_{min} and I_{max} are used only for providing logical borders. I_{min} is set as five because shapelets that are smaller than five time points are too short to be informative. I_{max} is set as 75% of the time series length, because, using complete time series as shapelet is not the aim of the procedure.

4.2. Classification

The classification process starts by mapping NxT dimensional time-series dataset to $Nx2\eta$ dimensional new representation using η number of shapelets. New representation contains closeness information between each time-series and shapelet pairs. In this work, closeness is represented by the best matching distance (BMD) and the best matching time (BMT). First η column of this mapping corresponds to best matching distance (BMD) and columns between $\eta + 1$ to 2η represents best matching time (BMT) between time-series and shapelet pairs. An illustration of new representation is provided in Table 4.2.

Best Matching Segment (BMS) of time series x_n, is the segment that has the minimum distance to the shapelet Ψ_j^l. The minimum distance is referred to as the Best Matching Distance (BMD) and denoted as $D(x_n, \Psi_j^l)$. Here $D(x_n, \Psi_j^l)$ is the minimum of the distances computed by sliding Ψ_j^l with length Q_j^l over the time series x_n. The segment providing the minimum distance is called the BMS of x_n to Ψ_j^l and the first time index of BMS of x_n is called the Best Matching Time (BMT). The distance measure considered in this study is the Euclidean distance.

$$D(x_n, \Psi_j^l) = \min_{t=1,\dots,T_n-Q_j^l+1} \sum_{q=1}^{Q_j^l} (x_{n,t+q-1} - \Psi_{j,q}^l)^2 \tag{4.1}$$

A supervised learner is trained to predict the class of the time series using the features provided in new representation and the trained learner is used as a predictive model. Any supervised learner can be used to make this prediction, however it is preferred to use tree-based ensembles (namely a random forest classifier) because of its known success in classification.

Table 4.2. Transformed feature space.

	Best Matching Distance (BMD)				Best Matching Time (BMT)				
	f_1	f_2	...	f_η	$f_{\eta+1}$	$f_{\eta+2}$	...	$f_{2\eta}$	Class
1	50.857	7.758	...	22.839	127	65	...	189	0
2	26.714	7.532	...	24.183	114	69	...	188	1
3	34.365	5.158	...	21.746	115	66	...	188	2
...	...	...	...	...	...	...	...	...	
$N-2$	60.630	2.422	...	24.781	133	108	...	190	0
$N-1$	33.866	6.260	...	23.444	113	68	...	187	1
N	30.638	7.444	...	21.450	114	69	...	188	1

for $\boldsymbol{\gamma}$ times **do**

Randomly sample a single time-series from each class.

Construct shapelets using three step shapelet discovery procedure.

Add shapelets to the set $\boldsymbol{S}$.

end for

Generate a new feature set $\boldsymbol{D(x^n, \Psi_k)}$ using the BMD and BMT of all time series $\boldsymbol{x^n}$ to each shapelet $\boldsymbol{\Psi_k \in S}$.

Train a random forest classifier, with $\boldsymbol{J_P}$ number of trees on the transformed feature set.

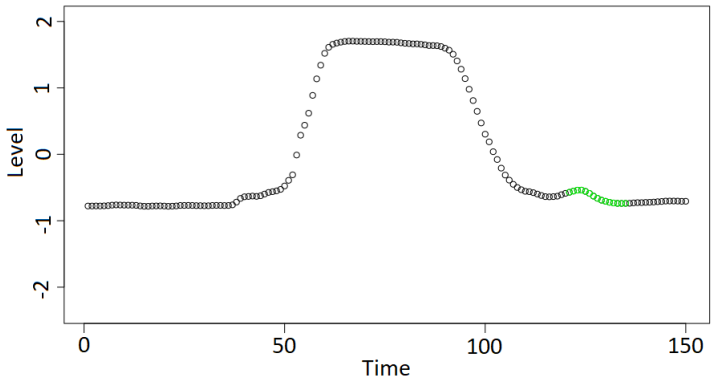
Figure 4.4. Pseudocode of the Ensemble-Based Fast Shapelet Approximation (EFSA) algorithm.

4.3. Interpretation

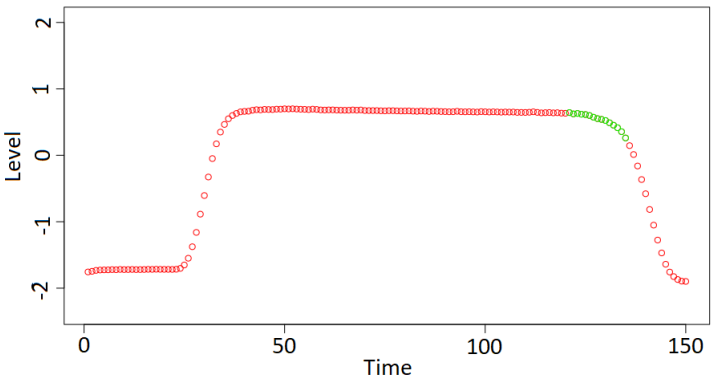
In this section interpretability of EFSA is demonstrated over two well-known datasets Gun-Point and Sony AIBO Robot Surface 1. Gun-Point dataset is one of the most studied time series classification problem [4]. Gun-Point dataset contains time series from two classes namely 'Gun' or 'NoGun'. These time series are generated by mapping the motion of two actors. For the Gun class, the actors "have their hands by their sides, draw a gun from a hip-mounted holster, point it at a target for approximately one second, and then return the gun to the holster and their hands to their sides" [25]. In the NoGun class, actors perform the same movements without a gun by using their index finger to point to a target.

Gun class contains the motion of drawing the gun from holster and returning it back unlike NoGun class. EFSA performs successfully on the Gun-Point dataset with classification accuracy of 97%. As an interpretable method, EFSA is expected to generate shapelets that focus on the parts where gun is drawn and returned. In order to illustrate the logical relationship between the classes and the shapelets, the most important patterns generated by the shapelet discovery process is schematized with the one instance from each class of time series in Figure 4.5. It can be observed from the Figure 4.5 that the shapelet generated using EFSA refers to the action which shows that suggested discovery process is successful in understanding the distinguishing parts of the given classes.

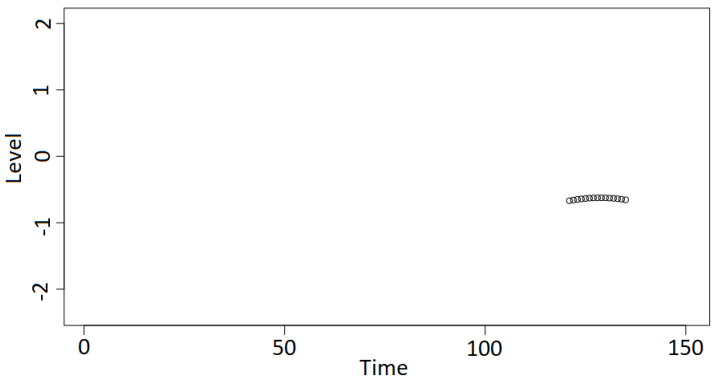
Sony AIBO Robot Surface 1 dataset is created by [31] with the aim of classifying the surface types using the measurements of the tri-axial accelerometer from Sony AIBO Robot [20]. The dataset contains measurements from two types of surfaces, carpet and cement. Cement floors are harder than floors with carpets, therefore sharper changes are demonstrated in the acceleration [20]. EFSA obtains test accuracy of 81.8%. Figure 4.6 represent the most important shapelet introduced by the supervised learner that refer to different shifts-of-weight in the walk cycle on the carpet floor.



(a) Time series from Gun class.

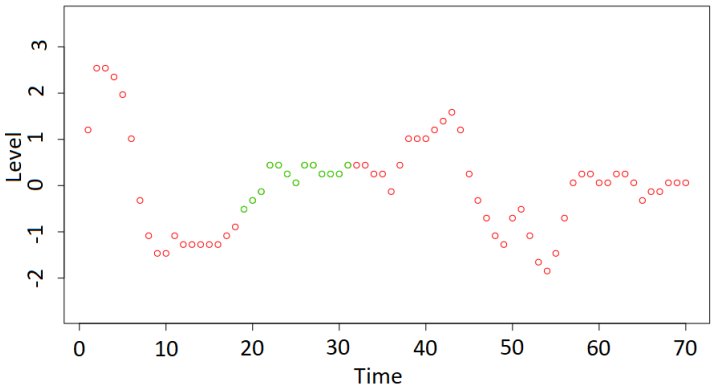


(b) Time series from NoGun class.

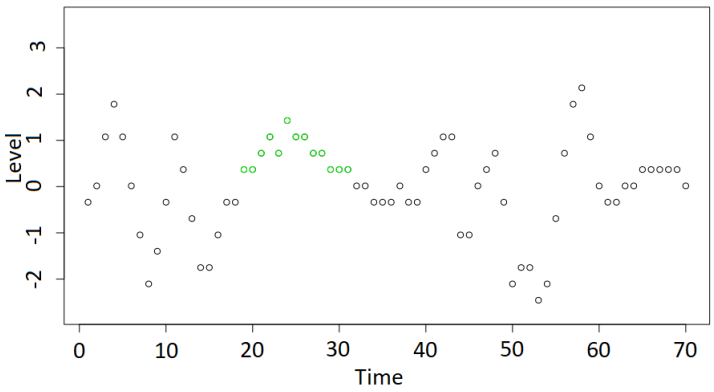


(c) Most important shapelet.

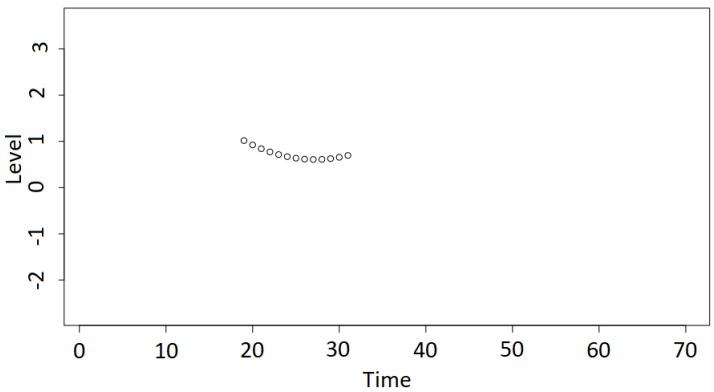
Figure 4.5. Two different time series examples from different classes of the Gun Point dataset and the most important shapelet generated from the Gun Point dataset.



(a) Time series from carpet floor class.



(b) Time series from cement floor class.



(c) Most important shapelet.

Figure 4.6. Two different time series examples from different classes of the Sony AIBO Robot Surface 1 dataset and the most important shapelet generated from the dataset.

5. EXPERIMENTS

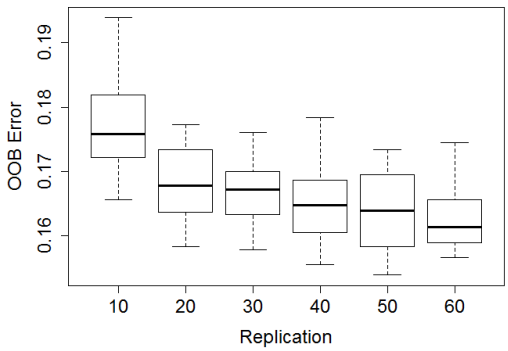
Experiments chapter contains three parts. In the first part, best performing state of the art methods Learned Shapelets (LS) [17], Fast Shapelets (FS) [21], Shapelet Transform (ST) [32] and Generalized Random Shapelet Forest (GRSF) [19] are compared to EFSA by evaluating each of them on 85 datasets from Bagnall *et al.* [27]. In the second part, empirical complexity of the EFSA is analyzed and computational performance of the algorithm with changing data size and time series length is reported. Lastly, in the third section computational performances of the all algorithms are compared. EFSA is implemented by using R programming language and experimentation is done in an Ubuntu 16.04 system with 4 GB RAM, Intel Core^{TM} i5-3337U CPU @ 1.80GHz $\times$ 4. Although the CPU can handle four threads in parallel, only a single thread is used in the experiments. The source codes can be found in Gorgulu *et al.* [33].

5.1. Classification Accurracy

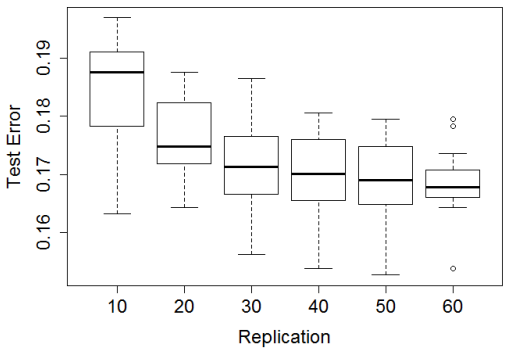
EFSA is compared to state-of-the-art shapelet-based approaches on the full set of benchmark datasets from Bagnall *et al.* [27]. Classification accuracies of LS, ST and FS that are used in comparison are kindly provided us by their authors. Since GRSF [19] does not report the performance on all 85 datasets, suggested parameter tuning is followed in Karlsson *et al.* [19] and all of the experiments are rerun using the authors' implementation.

Although EFSA does not require intense parameter tuning due to its non-parametric nature, number of sampling (γ) and number of trees in the random forest are required to be specified. Performance of EFSA is not effected by the number of trees in the random forest as long as there is sufficient number of trees. In the experimentation it is provided as 300 and it can be used as is in any application. For the number of sampling (γ) parameter, an upper bound is set and all of the experiments are conducted accordingly. By looking at the OOB and test errors of the more stable datasets with high number of training/test instances and balanced class distribution, provided in Figure

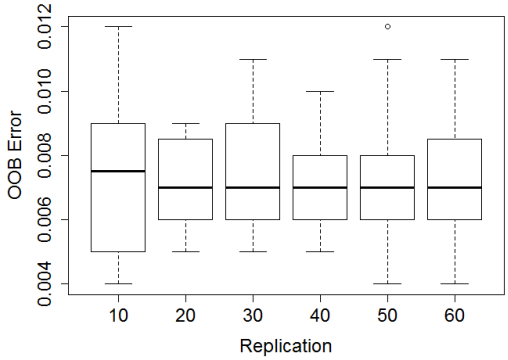
5.1, upper bound is specified as 50. Although it is possible to obtain the same performance by lower number of sampling in smaller datasets, it only makes EFSA faster without damaging its accuracy. Because of the random nature of EFSA, classification accuracies are reported as an average of five replications in each dataset.



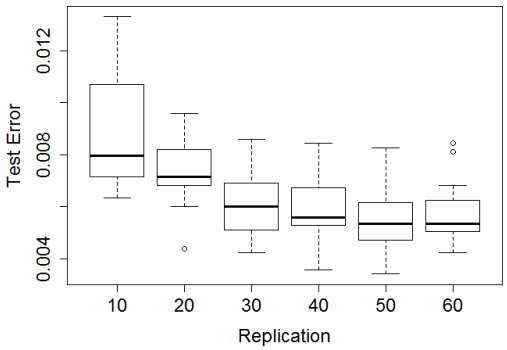
(a) OOB error rates for Phalanges Outlines Correct dataset.



(b) Test error rates for Phalanges Outlines Correct dataset.



(c) OOB error rates for Wafer dataset.



(d) Test error rates for Wafer dataset.

Figure 5.1. OOB and test error rates of Phalanges Outline Correct and Wafer datasets with changing sampling parameter.

Classification performance of all mentioned methods are provided in the Table 5.1. Classifiers are statistically compared by using a Friedman test followed by a Nemenyi post-hoc test as suggested by Demsar in [34] and illustrated in Figure 5.2. And pairwise comparison of the algorithms by number of better classified datasets provided in Table 5.2.

Table 5.1. Classification accuracies of EFSA, ST, LS, FS and GRSF on the UCR datasets.

	EFSA	**ST**	**LS**	**FS**	**GRSF**
Adiac	0.803*	0.783	0.522	0.593	0.726
ArrowHead	0.773	0.737	0.846*	0.594	0.717
Beef	0.793	0.900*	0.867	0.567	0.700
BeetleFly	0.805	0.900*	0.800	0.700	0.870
BirdChicken	0.885*	0.800	0.800	0.750	0.810
Car	0.812	0.917*	0.767	0.750	0.843
CBF	0.985	0.974	0.991*	0.940	0.978
ChlorineConc	0.691	0.700*	0.592	0.546	0.669
CinCECGtorso	0.818	0.954*	0.870	0.859	0.832
Coffee	0.986	0.964	1.000*	0.929	0.929
Computers	0.737	0.736	0.584	0.500	0.739*
CricketX	0.705	0.772*	0.741	0.485	0.771
CricketY	0.732	0.779*	0.718	0.531	0.756
CricketZ	0.708	0.787*	0.741	0.464	0.756
DiatomSize	0.911	0.925	0.980*	0.866	0.969
DistPhalanxAge	0.852*	0.770	0.719	0.655	0.852
DistPhalanxOut	0.830*	0.775	0.779	0.750	0.826
DistPhalanxTW	0.789	0.662	0.626	0.626	0.797*
Earthquakes	0.807	0.741	0.741	0.705	0.818*
ECG200	0.854	0.830	0.880*	0.810	0.828
ECG5000	0.943	0.944*	0.932	0.923	0.940
ECGFiveDays	0.922	0.984	1.000*	0.998	0.999
ElectricDevices	0.760*	0.747	0.587	0.579	0.736
FaceAll	0.803*	0.779	0.749	0.626	0.749
FaceFour	0.863	0.852	0.966	0.909	0.998*

Table 5.1. Classification accuracies of EFSA, ST, LS, FS and GRSF on the UCR datasets. (cont.)

	EFSA	**ST**	**LS**	**FS**	**GRSF**
FacesUCR	0.914	0.906	0.939*	0.706	0.861
FiftyWords	0.706	0.705	0.730*	0.481	0.723
Fish	0.927	0.989*	0.960	0.783	0.959
FordA	0.842	0.971*	0.957	0.787	0.927
FordB	0.832	0.807	0.917*	0.728	0.900
GunPoint	0.970	1.000*	1.000*	0.947	0.995
Ham	0.737*	0.686	0.667	0.648	0.764
HandOutlines	0.870	0.932*	0.481	0.811	0.882
Haptics	0.475	0.523*	0.468	0.393	0.460
Herring	0.634	0.672*	0.625	0.531	0.619
InlineSkate	0.354	0.373	0.438*	0.189	0.365
InsectWingbeat	0.626	0.627*	0.606	0.489	0.634
ItalyPower	0.953	0.948	0.960*	0.917	0.944
LargeKitchen	0.875*	0.859	0.701	0.560	0.871
Lightning2	0.766	0.738	0.820*	0.705	0.761
Lightning7	0.778	0.726	0.795*	0.644	0.707
Mallat	0.942	0.964	0.950	0.976*	0.940
Meat	0.940*	0.850	0.733	0.833	0.927
MedicalImages	0.728*	0.670	0.664	0.624	0.718
MidPhalanxAge	0.798*	0.643	0.571	0.545	0.780
MidPhalanxOutline	0.786	0.794*	0.780	0.729	0.740
MidPhalanxTW	0.627	0.519	0.506	0.532	0.632*
MoteStrain	0.899	0.897	0.883	0.777	0.914*
NonInvThorax1	0.913	0.950*	0.259	0.710	0.917
NonInvThorax2	0.931	0.951*	0.770	0.754	0.935

Table 5.1. Classification accuracies of EFSA, ST, LS, FS and GRSF on the UCR datasets. (cont.)

	EFSA	**ST**	**LS**	**FS**	**GRSF**
OliveOil	0.913*	0.900	0.167	0.733	0.867
OSULeaf	0.717	0.967*	0.777	0.678	0.883
Phalanges	0.833	0.763	0.765	0.744	0.841*
Phoneme	0.318	0.321*	0.218	0.174	0.305
Plane	1.000*	1.000*	1.000*	1.000*	1.000*
ProxPhalanxAge	0.830	0.844*	0.834	0.780	0.841
ProxPhalanxOut	0.866	0.883*	0.849	0.804	0.862
ProxPhalanxTW	0.805*	0.805	0.776	0.702	0.804
Refr.Devices	0.584*	0.581	0.515	0.333	0.563
ScreenType	0.495	0.520	0.429	0.413	0.573*
ShapeletSim	1.000*	0.956	0.950	1.000*	1.000*
ShapesAll	0.848*	0.842	0.768	0.580	0.830
SmallKitchen	0.842*	0.792	0.664	0.333	0.806
SonyRobot1	0.818	0.844	0.810	0.686	0.878*
SonyRobot2	0.852	0.934*	0.875	0.790	0.912
StarLightCurves	0.977	0.979*	0.947	0.918	0.977
Strawberry	0.957	0.962*	0.911	0.903	0.957
SwedishLeaf	0.938*	0.928	0.907	0.768	0.906
Symbols	0.955*	0.882	0.932	0.934	0.933
SyntheticControl	0.994	0.983	0.997*	0.910	0.993
ToeSegmentation1	0.912	0.965*	0.934	0.956	0.940
ToeSegmentation2	0.852	0.908	0.915*	0.692	0.885
Trace	1.000*	1.000*	1.000*	1.000*	1.000*
TwoLeadECG	0.972	0.997*	0.996	0.924	0.992
TwoPatterns	0.997	0.955	0.993	0.908	0.998*

Table 5.1. Classification accuracies of EFSA, ST, LS, FS and GRSF on the UCR datasets. (cont.)

	EFSA	**ST**	**LS**	**FS**	**GRSF**
UWaveAll	0.963*	0.942	0.953	0.789	0.955
UWaveX	0.813*	0.803	0.791	0.695	0.804
UWaveY	0.717	0.730*	0.703	0.596	0.714
UWaveZ	0.749*	0.748	0.747	0.638	0.742
Wafer	0.995	1.000*	0.996	0.997	1.000*
Wine	0.667	0.796*	0.500	0.759	0.715
WordSynonyms	0.596	0.571	0.607	0.431	0.607*
Worms	0.575	0.740*	0.610	0.649	0.556
WormsTwoClass	0.739	0.831*	0.727	0.727	0.746
Yoga	0.851*	0.818	0.834	0.695	0.838

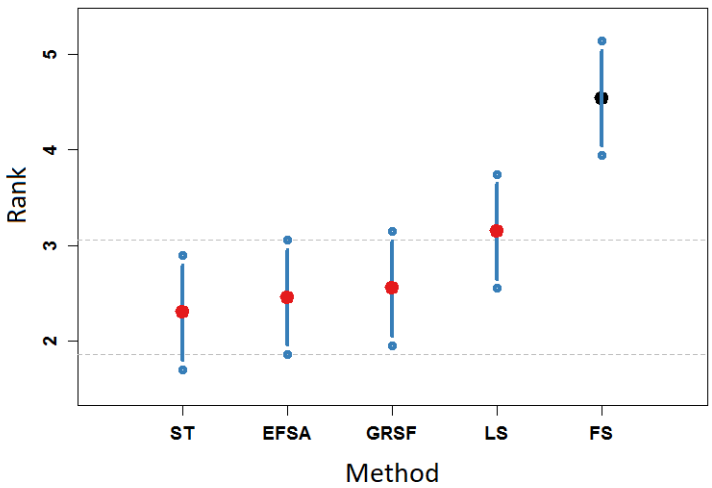
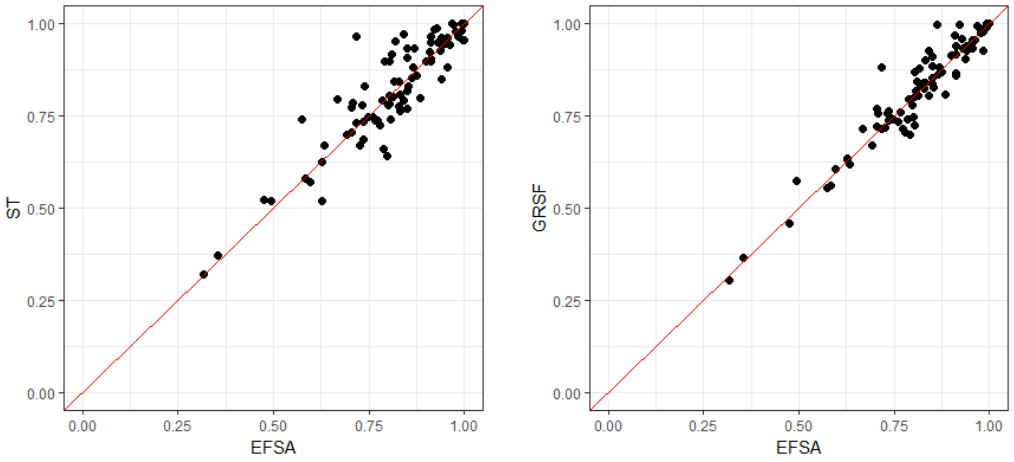


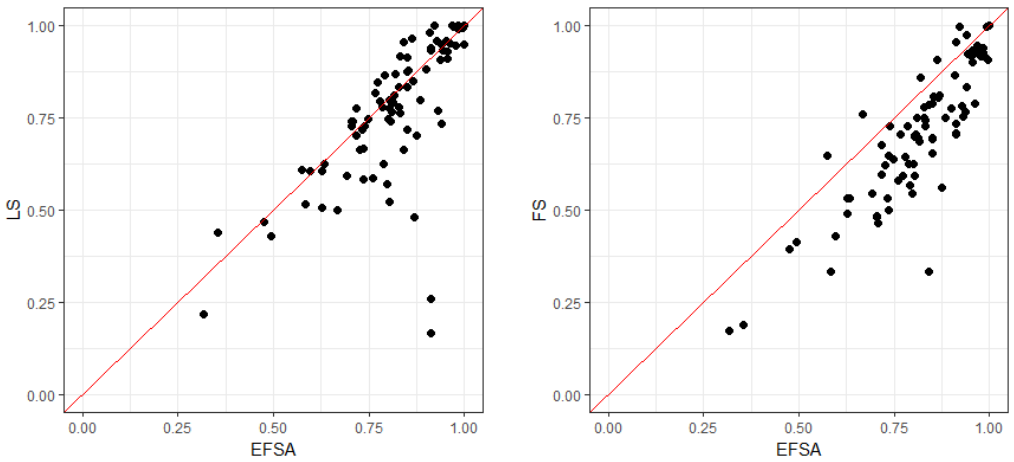
Figure 5.2. Results of Friedman-Nemenyi rank test on accuracies from UCR datasets.

From Table 5.2, it can be observed that EFSA has classified more datasets with high accuracy than each of the methods. Also, it can be noted from Figure 5.3 and Table 5.2 that, differences in terms of classification accuracy between EFSA, ST and GRSF are not significant.



(a) EFSA vs. ST.

(b) EFSA vs. GRSF.



(c) EFSA vs. LS.

(d) EFSA vs. FS.

Figure 5.3. Classification accuracies of EFSA (mean of 10 replications) versus ST, GRSF, LS, FS.

Table 5.2. Pairwise comparison of ST, GRSF, LS and FS against EFSA in terms of classification accuracies.

		ST	**GRSF**	**LS**	**FS**
EFSA	Better	43*	43*	51*	74*
	Worse	40	39	32	8
	Equal	2	3	2	3

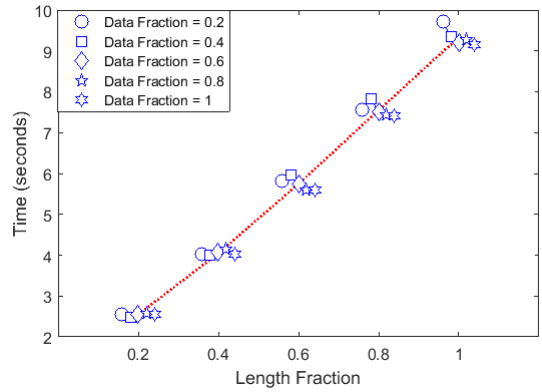
Table 5.3. Average ranks of five classifiers for the all datasets by the problem category.

	EFSA	**ST**	**LS**	**FS**	**GRSF**
Image	2.14*	2.55	3.17	4.59	2.52
Spectro	1.86*	2.57	3.14	4.71	2.71
Sensor	2.63	1.94*	2.75	4.56	2.44
Simulated	2.50*	2.67	3.33	3.17	2.83
ECG	3.14	2.14*	2.86	4.28	2.57
Device	1.83*	2.33	4.00	5.00	1.83*
Motion	2.78	1.57*	2.86	4.50	2.50
Overall	2.40	2.24	3.08	4.48	2.49

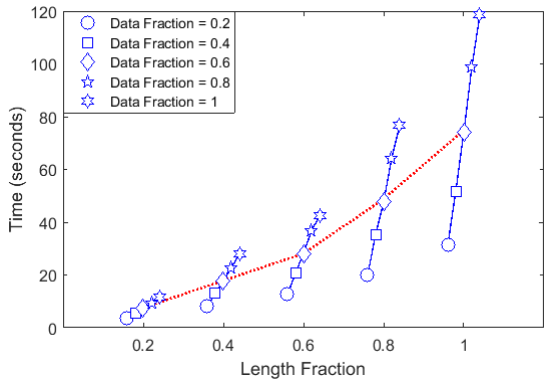
In addition to the comparison based on all time-series, Table 5.3 provides average ranks of five classifier for all datasets by problem category. EFSA significantly outperforms other five classifiers in Image, Simulated and Device categories of the datasets. In the other categories, ST outperforms other classifiers, however, the difference between EFSA and ST is not significant except Motion category.

5.2. Complexity Analysis

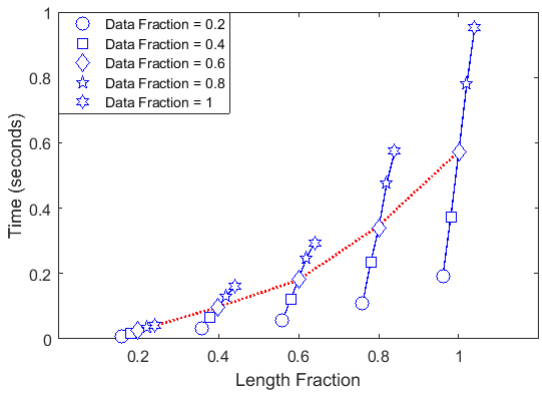
In this section, time performance of EFSA with respect to changing data size and time-series lengths is analyzed. Inline Skate dataset from UCR datasets is used to demonstrate the effect of the data-size and time-series length on the performance of EFSA due to its balanced train/test set distribution and sufficient length. Shapelet generation, training, testing and total elapsed time are reported as a result of this analysis. As the name implies shapelet generation time is the elapsed time until shapelets are generated using supervised tree-based approximation algorithm. Training time is the amount of time required to train random forest using the distances of the time series to the generated shapelets. And testing time is the time needed to classify all test instances in the dataset.



(a) Shapelet generation time.



(b) Training time.



(c) Testing time.

Figure 5.4. Computational times (sec) with changing data size and time series length.

From Figure 5.4(a) it can be observed that shapelet generation time increases linearly as time series length increases. This demonstrates the speed and efficiency of the shapelet approximation procedure of EFSA. Training and testing times do not

increase exponentially with increasing time-series length Figure 5.4. Figure 5.4(b) and Figure 5.4(c) show quadratic increase on the running times as time series length increases which again reinforces the claim about speed and efficiency of the algorithm.

Increasing data size does not affect shapelet generation time due to the sampling method that EFSA uses in shapelet generation part. In addition, it can be observed from Figure 5.4 that training, testing and overall running times increases linearly with increasing data-size.

5.3. Comparison of Computational Times with Other Three Methods

From the accuracy comparison it can be observed that EFSA, ST and GRSF performs significantly better than the FS and LS. Therefore, in this section, LS is left out from this analysis and only computational times of EFSA, ST, GRSF and FS are compared. Computational comparison is realized in Ubuntu 16.04 system with 4 GB RAM, Intel Core^{TM} i5-3337U CPU @ 1.80GHz $\times$ 4 setting. Runs of all three classifiers are taken by using the same setting. Using Yoga dataset from UCR database, different datasets with differing data sizes are constructed. Experiment are made by using source codes provided by GRSF, ST and FS. Average running times from 10 replications are illustrated in Table 5.4.

Table 5.4. Computation times (sec) of three classifiers for Yoga dataset with different data rates.

Rate	**EFSA**	**ST**	**FS**	**GRSF**
0.2	3.91*	14.67	63.40	2995.27
0.4	5.04*	57.28	129.82	8410.79
0.6	6.17*	166.68	230.42	15224.08
0.8	7.22*	379.52	322.73	22868.43
1	8.43*	741.85	421.42	30888.67

Although ST and GRSF work with multiple core and EFSA works with single core, EFSA performs computationally more efficient than ST and GRSF. Due to computationally inefficient parameter optimization procedure, computation time of GRSF takes considerably longer time than the other methods.

5.4. EFSA vs. FS

Fast Shapelet method approximates shapelets by using a piecewise constant approximation in an unsupervised manner. Approximated shapelet set is used to construct a distance matrix using BMD and 1NN classifier is trained on the distance matrix. In order to compare the power of the shapelet generation procedures, EFSA is run with 1NN final classifier. EFSA uses BMD and BMT as its default features. However, in this case, results are obtained by only using BMD. Comparison is made on the 84 datasets from UEA and UCR databases and the test accuracies are demonstrated in Figure 5.5. Even without BMT, EFSA with 1NN classifier outperforms FS in terms of test accuracies on 55 of 85 datasets. This result demonstrates the EFSA's ability of approximating better shapelet set than FS.

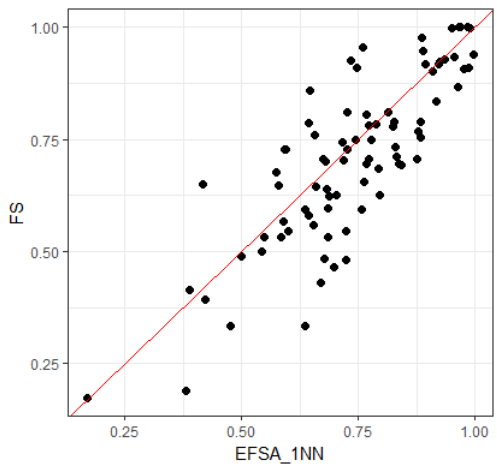


Figure 5.5. Graphical comparison of classification accuracies of EFSA with 1NN classifier against Fast Shapelet.

Table 5.5. Classification accuracies of EFSA with 1NN classifier vs Fast Shapelet on the UCR datasets.

	EFSA1NN	**FS**
Adiac	0.637*	0.593
ArrowHead	0.757*	0.594
Beef	0.590*	0.567
BeetleFly	0.680	0.700*
BirdChicken	0.745	0.750*
Car	0.777*	0.750
CBF	0.997*	0.940
ChlorineConc	0.599*	0.546
CinCECGtorso	0.646	0.859*
Coffee	0.936*	0.929
Computers	0.544*	0.500
CricketX	0.676*	0.485
CricketY	0.685*	0.531
CricketZ	0.698*	0.464
DiatomSizeReduction	0.963*	0.866
DistPhalanxAge	0.762*	0.655
DistPhalanxOutline	0.743	0.750*
DistPhalanxTW	0.704*	0.626
Earthquakes	0.674	0.705*
ECG200	0.814*	0.810
ECG5000	0.925*	0.923
ECGFiveDays	0.950	0.998*
ElectricDevices	0.643*	0.579
FaceAll	0.795*	0.626
FaceFour	0.748	0.909*

Table 5.5. Classification accuracies of EFSA with 1NN classifier vs Fast Shapelet on the UCR datasets. (cont.)

	EFSA1NN	**FS**
FacesUCR	0.877*	0.706
FiftyWords	0.723*	0.481
Fish	0.789*	0.783
FordA	0.644	0.787*
FordB	0.594	0.728*
GunPoint	0.889	0.947*
Ham	0.580	0.648*
HandOutlines	0.725	0.811*
Haptics	0.421*	0.393
Herring	0.547*	0.531
InlineSkate	0.380*	0.189
InsectWingbeat	0.499*	0.489
ItalyPower	0.922*	0.917
LargeKitchen	0.654*	0.560
Lightning2	0.772*	0.705
Lightning7	0.659*	0.644
Mallat	0.887	0.976*
Meat	0.917*	0.833
MedicalImages	0.687*	0.624
MidPhalanxAge	0.723*	0.545
MidPhalanxOutline	0.726	0.729*
MidPhalanxTW	0.584*	0.532
MoteStrain	0.824*	0.777
NonInvThorax1	0.832*	0.710
NonInvThorax2	0.883*	0.754

Table 5.5. Classification accuracies of EFSA with 1NN classifier vs Fast Shapelet on the UCR datasets. (cont.)

	EFSA1NN	**FS**
OliveOil	0.830*	0.733
OSULeaf	0.573	0.678*
Phalanges	0.715	0.744*
Phoneme	0.168	0.174*
Plane	0.969	1.000*
ProxPhalanxAge	0.774	0.780*
ProxPhalanxOutline	0.768	0.804*
ProxPhalanxTW	0.717*	0.702
Refr.Devices	0.476*	0.333
ScreenType	0.388	0.413*
ShapeletSim	0.966	1.000*
ShapesAll	0.836*	0.830
SmallKitchen	0.637*	0.333
SonyRobot1	0.792*	0.686
SonyRobot2	0.826*	0.790
StarLightCurves	0.893	0.918*
Strawberry	0.909*	0.903
SwedishLeaf	0.879*	0.768
Symbols	0.956*	0.934
SyntheticControl	0.986*	0.910
ToeSegmentation1	0.758	0.956*
ToeSegmentation2	0.843*	0.692
Trace	0.983	1.000*
TwoLeadECG	0.733	0.924*
TwoPatterns	0.977*	0.908

Table 5.5. Classification accuracies of EFSA with 1NN classifier vs Fast Shapelet on the UCR datasets. (cont.)

	EFSA1NN	**FS**
UWaveAll	**0.883**	0.789
UWaveX	0.769*	0.695
UWaveY	0.684*	0.596
UWaveZ	0.682*	0.638
Wafer	0.990	0.997*
Wine	0.656	0.759*
WordSynonyms	0.669*	0.431
Worms	0.416	0.649*
WormsTwoClass	0.591	0.727*
Yoga	0.836*	0.695

6. CONCLUSION

In this study, an accurate and efficient method for time-series classification based on shapelets is proposed. Shapelet-based classification has been attracting a major interest due to its success in time-series classification. There are various well-known methods for shapelet-based classification however, these methods are either computationally heavy or inaccurate. Accuracy of the shapelet-based methods are directly related to the quality of the shapelet set that is used in classification. Therefore, efficient construction of the shapelet space yields a great importance. Current state-of-the-art methods construct the shapelet set by pruning the shapelet space using various methods or by generating them using optimization techniques. Learned Shapelets (LS) [17] tries to find best shapelets by optimizing a logistic cost function. These kind of approaches contain too many parameters which requires computationally heavy parameter selection procedure. Shapelet Transform (ST) tries to prune the shapelet space efficiently, however, evaluation the exponential number of shapelet candidate and their interaction is computationally inefficient even it is better than explicit enumeration. In order to decrease the computational burden, sampling methods are applied on the shapelet space. Fast Shapelets (FS) randomly samples shapelets from shapelet spaces and uses them in classification. Even though its computational performance is not heavy due to complete random sampling, it provides inaccurate results. Generalized Random Shapelet Forest (GRSF) merges the idea of sampling with shapelet space pruning by using forest of decision trees. GRSF again contains too many parameters. Although GRSF works well and considerably more efficient than ST when parameter settings for each dataset is known, if they are unknown time complexity increases drastically for GRSF to learn and predict by scratch.

In order to generate a powerful shapelet set efficiently, a non-parametric approach namely supervised tree-based approximation is suggested. Supervised tree-based approximation efficiently generates distinctive shapelets with various lengths without specifying any parameters. Applying this tree-based approximation procedure to the stratified samples recursively helps to create more robust model with higher

predictive performance. By utilizing a random forest on the similarity vector obtained by BMD between shapelets and time-series, a successful classification results are obtained. EFSA is a computationally more efficient and an accurate model compared to the state-of-the-art approaches. Shapelet discovery method of EFSA is an original and very efficient method which can also be used in other studies in order to increase the quality of shapelet sets and decrease the time complexity.

REFERENCES

1. Fu, T., "A review on time series data mining", *Engineering Applications of Artificial Intelligence*, Vol. 24, pp. 164–181, 2011.

2. Khadra, L., A. Al-Fahoum and S. Binajjaj, "A quantitative analysis approach for cardiac arrhythmia classification using higher order spectral techniques", *Biomedical Engineering, IEEE Transactions on*, Vol. 52, No. 11, pp. 1840 –1845, November 2005.

3. Kakizawa, Y., R. H. Shumway and M. Taniguchi, "Discrimination and Clustering for Multivariate Time Series", *Journal of the American Statistical Association*, Vol. 93, No. 441, pp. pp. 328–340, 1998.

4. Ding, H., G. Trajcevski, P. Scheuermann, X. Wang and E. Keogh, "Querying and mining of time series data: experimental comparison of representations and distance measures", *Proc. VLDB Endow.*, Vol. 1, pp. 1542–1552, August 2008.

5. Xi, X., E. Keogh, C. Shelton, L. Wei and C. Ratanamahatana, "Fast time series classification using numerosity reduction", *Proceedings of International Conference on Machine Learning (ICML06)*, pp. 1033–1040, ACM, 2006.

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

7. Jeong, Y.-S., M. K. Jeong and O. A. Omitaomu, "Weighted dynamic time warping for time series classification", *Pattern Recognition*, Vol. 44, No. 9, pp. 2231–2240, 2011.

8. Keogh, E. and S. Kasetty, "On the Need for Time Series Data Mining Benchmarks:

A Survey and Empirical Demonstration", *Data Mining and Knowledge Discovery*, Vol. 7, No. 4, pp. 349–371, 2003.

9. Ratanamahatana, C. and E. Keogh, "Making time-series classification more accurate using learned constraints", *Proceedings of SIAM International Conference on Data Mining (SDM04)*, pp. 11–22, 2004.

10. Ueno, K., X. Xi, E. Keogh and D. Lee, "Anytime classification using the nearest neighbor algorithm with applications to stream mining", *IEEE International Conference on Data Mining (ICDM06)*, pp. 623–632, IEEE, 2007.

11. Xing, Z., J. Pei and P. S. Yu, "Early prediction on time series: a nearest neighbor approach", *Proceedings of International Joint Conference on Artifical Intelligence (IJCAI09)*, pp. 1297–1302, Morgan Kaufmann, 2009.

12. Sakoe, H., "Dynamic programming algorithm optimization for spoken word recognition", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. 26, pp. 43–49, 1978.

13. Ratanamahatana, C. and E. Keogh, "Three myths about dynamic time warping data mining", *Proceedings of SIAM International Conference on Data Mining (SDM05)*, Vol. 21, pp. 506–510, 2005.

14. Lin, J., E. Keogh, L. Wei and S. Lonardi, "Experiencing SAX: a novel symbolic representation of time series", *Data Mining and Knowledge Discovery*, Vol. 15, pp. 107–144, 2007.

15. Rodríguez, J. J. and C. J. Alonso, "Interval and dynamic time warping-based decision trees", *Proceedings of ACM Symposium on Applied Computing (SAC04)*, pp. 548–552, 2004.

16. Hills, J., J. Lines, E. Baranauskas, J. Mapp and A. Bagnall, "Classification of time series by shapelet transformation", *Data Mining and Knowledge Discovery*,

Vol. 28, No. 4, pp. 851–881, 2014.

17. Grabocka, J., B. Schiling, M. Wistuba and L. Schmidt-Thieme, "Learning time-series shapelets", *ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 392–401, 2014.

18. Hidasi, B. and C. Gaspar-Papanek, "ShiftTree: An Interpretable Model-Based Approach for Time Series Classification", *Machine Learning and Knowledge Discovery in Databases*, Vol. 6912 of *Lecture Notes in Computer Science*, pp. 48–64, Springer Berlin / Heidelberg, 2011.

19. Karlsson, I., P. Papapetrou and H. Boström, "Generalized random shapelet forests", *Data Mining and Knowledge Discovery*, Vol. 30, No. 5, pp. 1053–1085, 2016.

20. Mueen, A., E. J. Keogh and N. Young, "Logical-shapelets: an expressive primitive for time series classification.", C. Apté, J. Ghosh and P. Smyth (Editors), *KDD*, pp. 1154–1162, ACM, 2011.

21. Rakthanmanon, T. and E. Keogh, "Fast shapelets: A scalable algorithm for discovering time series shapelets", *proceedings of the 2013 SIAM International Conference on Data Mining*, pp. 668–676, SIAM, 2013.

22. Xing, Z., J. Pei, P. S. Yu and K. Wang, "Extracting Interpretable Features for Early Classification on Time Series", *SDM*, pp. 247–258, 2011.

23. Yamada, Y., H. Yokoi and K. Takabayashi, "Decision-tree induction from time-series data based on standard-example split test", *Proceedings of International Conference on Machine Learning (ICML03)*, pp. 840–847, Morgan Kaufmann, 2003.

24. Ye, L. and E. Keogh, "Time series shapelets: a new primitive for data mining", *Proceedings of the 15th ACM SIGKDD international conference on Knowledge dis-*

covery and data mining, KDD '09, pp. 947–956, 2009.

25. Ye, L. and E. Keogh, "Time series shapelets: a novel technique that allows accurate, interpretable and fast classification", *Data Mining and Knowledge Discovery*, Vol. 22, pp. 149–182, 2011.

26. Chakrabarti, K., E. Keogh, S. Mehrotra and M. Pazzani, "Locally adaptive dimensionality reduction for indexing large time series databases", *ACM Trans. Database Syst.*, Vol. 27, No. 2, pp. 188–228, Jun. 2002.

27. Bagnall, A., J. Lines, A. Bostrom, J. Large and E. Keogh, "The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances", *Data Mining and Knowledge Discovery*, Vol. 31, No. 3, pp. 606–660, 2017.

28. Ratanamahatana, C., J. Lin, D. Gunopulos, E. Keogh, M. Vlachos and G. Das, "Mining Time Series Data", *Data Mining and Knowledge Discovery Handbook*, pp. 1049–1077, Springer US, 2010.

29. Breiman, L., J. Friedman, R. Olshen and C. Stone, *Classification and Regression Trees*, Wadsworth, Belmont, MA, 1984.

30. Geurts, P., "Pattern Extraction for Time Series Classification", *Principles of Data Mining and Knowledge Discovery*, Vol. 2168 of *Lecture Notes in Computer Science*, pp. 115–127, Springer Berlin / Heidelberg, 2001.

31. Vail, D. and M. Veloso, "Learning from Accelerometer Data on a Legged Robot", *In Proceedings of the 5th IFAC/EURON Symposium on Intelligent Autonomous Vehicles*, 2004.

32. Bagnall, A., L. Davis, J. Hills and J. Lines, "Transformation based ensembles for time series classification", *Proceedings of the 2012 SIAM international conference on data mining*, pp. 307–318, SIAM, 2012.

33. Gorgulu, B. and M. G. Baydogan, "EFSA Implementation in R", `https://github.com/gorguluberk/EFSA`, 2018.

34. Demšar, J., "Statistical comparisons of classifiers over multiple data sets", *Journal of Machine learning research*, Vol. 7, No. Jan, pp. 1–30, 2006.