Network Based Sampling for Time Series Classification

BY

SAMUEL HARFORD
B.S., UNIVERSITY OF ILLINOIS AT CHICAGO, 2016

THESIS

Submitted as partial fulfillment of the requirements
for the degree of Master of Science in Industrial Engineering
in the Graduate College of the
University of Illinois at Chicago, 2021

Chicago, Illinois

Defense Committee:

Houshang Darabi, Chair and Advisor
Michael Scott, Mechanical and Industrial Engineering
Ugo Buy, Computer Science

This thesis is dedicated to my parents, Tom and Michelle Harford, who have been a constant source of love and encouragement in my life.

ACKNOWLEDGMENTS

I would like to seize this opportunity and thank several people without whom this work would never be possible.

First, I would like to thank my thesis adviser Professor Houshang Darabi who offered his continuous advice and encouragement throughout the course of this thesis. I thank him for his excellent guidance, understanding, and support.

I would like to thank the rest of my committee members, Michael Scott and Ugo Buy generously offering their time throughout the preparation and review of this document.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

1-NN	First-Nearest-Neighbor
BOSSVS	Bag-of-SFA-Symbols in Vector Space
CID	Complexity Invariant Distance
DTW	Dynamic Time Warping
DDDTW	Derivative DTW
DTDc	Derivative Transform Distance
ED	Euclidean Distance
FS	Fast Shapelets
KNN	K-Nearest-Neighbor
LS	Learned Shapelets
MSM	Move-Split-Merge
NBS	Network Based Sampling
ST	Shapelet Transform
TSC	Time Series Classification
TWE	Time Warp Edit
WDTW	Weighted DTW

SUMMARY

A time series is an ordered collection of data points collected by observers or sensing devices. Time series classification aims to label time series instances based on previously seen examples. The two primary goals of time series classification are to obtain highly accurate results while reducing the time of calculations. 1-Nearest-Neighbor Dynamic Time Warping (1-NN DTW) is the most widely used classification method on time series and serves as a benchmark when compared to emerging techniques. Several studies have shown that for the task of time series classification, 1-NN DTW is hard to beat with respects to both goals.

Although 1-NN DTW achieves accurate results, it comes with a high cost of processing. With the increased need for machine learning based algorithms to run on edge devices (low capacity), there is a need to reduce the processing requirements of classification algorithms. The focus of this dissertation is to reduce the processing time of time series classification algorithms. This is achieved through a method called Network Based Sampling (NBS), which limits the number of training examples used to classify each instance.

Network Based Sampling is a preprocessing attachment that can be used in conjugation with any classification algorithm. NBS works by analyzing the training data of a time series classification problem to rank each training instance. The ranking process first generates a network of all training instances based on their class labels and Euclidean distances to each other. Then the most prominent instances are determined through an analysis of the network connections. From this network, the training set is sampled from based on a percentage specified by the user that corresponds to the average time savings.

The effectiveness of NBS is tested as an attachment on 1-NN DTW time series classification. The 85 time series benchmarks from the University of California, Riverside (UCR) archive is used as a source of data for classification evaluation.

1 INTRODUCTION

This chapter introduces Time Series Classification (TSC) and the significance of efficient classification algorithms.

1.1 Time Series Classification

The modern age has seen a drastic increase in the digital sensing devices that has resulted in the continuous growth of sensed data (Gregor, 2016), a significant portion of which is in the form of time series. These sensing devices include robot sensors, wearable sensors, smart meters, satellites, smart phones, and more. This growth in data has resulted in the improvement of querying methods (Wilson, 2017) (Wang, Mueen, Ding, Trajcevski, Scheuermann and Keogh, 2013), approaches to data transformation (Nair, Kumar, Sakthivel and Vipin, 2017) (Lin, Keogh, Lonardi and Chiu, 2003) (Liao, 2005), and data mining techniques (Keogh and Kasetty, 2003) (Fu, 2011). Time series data is used for analysis (including forecasting, anomaly detection, signal estimation, etc.) and classification. With the increase in time series data, the field of time series classification (TSC) has undergone a surge of importance as data accumulates from sensing in personalized medicine (The BIDMC Congestive Heart Failure Database, 2014), and human walking motions (CMU Graphics Lab Motion Capture Database, 2014), for example. TSC seeks to establish machine learning models that can replicate human like understanding of similarities between ordered data series.

At a high level, effective models for preforming TSC seek to achieve a high prediction accuracy (Xing, Pei, Yu and Wang, 2011) while providing a quick modeling and/or testing time. When exploring the world of TSC models, one of the most predominantly used methods is 1-Nearest Neighbor Dynamic Time Warping (1-NN DTW) (Berndt and Clifford, 1994). 1-NN DTW

has been a difficult algorithm to beat in terms of both accuracy and speed. Algorithms that aim to decrease the processing time of traditional time series classification algorithms are introduced in the next subsection.

Several methods have been introduced in the past year that sacrifice speed with the aim of increasing accuracy. An example of such methods includes various of Symbolic-Fourier-Approximations like WEASEL, BOSS, and BOSSVS introduced by Patrick Schafer's group (Schafer and Leser, 2017) (Schafer, 2016) (Schafer, 2015). Schafer's methods rely on a transformation of time series data and the ensembling of methods with varying parameters to increase classification accuracy. The current most successful time series classification algorithm for classification accuracy is a neural network architecture introduced in (Karim, Majumdar, Darabi and Chen, 2018). This architecture uses a combination of Convolutional Neural Networks and Long Short-Term Memory Networks to effectively learn to classify time series classification problem in spite of the limited training instances.

1.2 Importance of Efficiency

Where 1-NN DTW can be beaten, it is usually by a slight margin of accuracy compared to the added cost of processing time. The core drawback of DTW is its computational complexity. A high complexity makes methods like 1-NN DTW impractical on low resource devices like tablets, smart phones and other portable devices that are prominent in today's world.

There have been several approaches tested to overcome the shortcoming of 1-NN DTW: pruning techniques in calculation of the exact DTW, feature selection, and DTW approximation. Lower bounding methods such as Kim, Yi, and Keogh lower bounding (Ding, Trajcevski, Scheuermann, Wang and Keogh, 2008) (Keogh, Wei, Xi, Vlachos, Lee and Protopapas, 2009)

(Keogh and Ratanamahatana, 2005) (Tavenard and Amsaleg, 2015) reduce processing time by limiting distance calculations preformed in the classification process. Upper bounding techniques eliminates calculations in 1-NN DTW by first calculating the Euclidean distance between time series to prune unpromising warping paths in the DTW matrix, are examples of pruning techniques in calculation of the exact DTW. Shapelets (Ye and Keogh, 2009) are an example of the feature selection framework in time series classification. In Shapelets, each class of a time series benchmark is analyzed to find the predominant selection of data to represent each class. Different approximation methods have been developed to make DTW faster while maintaining its accuracy, including FastDTW (Salvador and Chan, 2007a), BDTW (Sharabiani, Darabi, Harford, Douzali, Karim, Johnson and Chen, 2018), 3D DTW (Sharabiani, Darabi, Rezaei, Harford, Johnson and Karim, 2017), and more (Salvador and Chan, 2007b) (Keogh and Pazzani, 2000) (Chu, Keogh, Hart and Pazzani, 2002) (Tavenard and Amsaleg, 2015). Incorporating constraint bands to limit the search area in DTW matrix (Constrained warping) and using DTW on reduced representation of the time series are examples of these approximation methods. The use of the Network Based Sampling framework aims to reduce the time required for computation by only analyzing necessary training instances within a time series database. This eliminates the need to process training instances of little importance while maintaining preformation. This is especially important in settings where real time classification of incoming data is required from a massive training database.

1.3 Chapter Synopsis

The rest of this research is organized as follows: In Chapter 2, background and related works are reviewed. Chapter 3 explains the Network Based Sampling framework. Chapter 4 goes over the results of Network Based Sampling on a time series benchmark. Finally, Chapter 5 concludes the thesis and talks about future work.

2 LITERATURE REVIEW

This chapter introduces relevant terminology, time series classification algorithms, and speed up methods in time series classification algorithms.

2.1 Graph Terminology

A graph is a set of connected nodes and links. In graph theory, a directed graph (digraph) is a graph where the links point from one node to another (Cook and Holder, 2006). Formally, a digraph is an ordered pair G = (V, A) where V is a set of nodes and A is a set of ordered pair representing links. Digraphs are used to express the relationship between training instances.

2.2 Distance Metrics

This subsection introduces relevant distance metrics used in the field of time series classification.

2.2.1. Euclidean Distance (ED)

A time series, $T = t_1, t_2, \ldots t_n$, is defined as an ordered list of observations made at successive equally spaced points in time. For two time series $X = x_1, x_2, \ldots x_n$ and $Y = y_1, y_2, \ldots y_n$ both of length n, the most common and intuitive method to measure their distance is referred to as the Euclidean Distance (Kalpakis, Gada and Puttagunta, 2001). ED is derived based on the square root of the sum of squared differences.

$$ED(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

2.2.2. Dynamic Time Warping (DTW)

Dynamic Time Warping is a distance metric used to non-linearly align two time series. The non-linearity allows for time series of different lengths to be compared. DTW outputs a matrix of the distance path and the shortest distance between series (Keogh and Ratanamahatana, 2005). For two time series $X = x_1, x_2, \ldots x_n$ and $Y = y_1, y_2, \ldots y_m$ of lengths n and m respectively, a DTW matrix can be calculated of size n *x* m. Each cell of the DTW matrix represents an alignment between two points of the corresponding time series. Matrix calculations must follow the following conditions:

Boundary Condition: The paths should start from the beginning of each time series (x1; y1) and finish at the last point of each time series (x_n, y_m).

Continuity Condition: The paths should have no jumps in steps; the points to consider for distance at the (i, j) point are (i-1, j), (i, j-1), and (i-1, j-1).

Monotonicity Condition: The warping paths can only go forward in time.

P is defined as a continuous path of cell in the matrix from (x_1, y_1) to (x_n; y_m). The s^{th} element of P is defined as $p_s = d(i, j)_s$; where $d(i, j) = (x_i - y_j)^2$, S is the length of $P = p_1, p_2, \ldots p_S$. The distance for DTW is equal to sqrt(D(n,m)), where the distance of each cell is as follows:

$$D(i,j) = \left(x_i - y_j\right)^2 + min[D(i-1, j-1), D(i-1, j), D(i, j-1)]$$

Initiated by the following conditions:

$$D(1,1) = 0; \; D(1, 2 \ldots . m) = 1; \; D(2 \ldots n; 1) = 1$$

An example of the DTW matrix between two short time series is presented in Figure 1.

	1	5	4	3	2	4	1
2	1	9	4	1	0	4	1
5	16	0	1	4	9	1	16
3	4	4	1	0	1	1	4
2	1	9	4	1	0	4	1
1	0	16	9	4	1	9	0
3	4	4	1	0	1	1	4

	1	5	4	3	2	4	1
2	1	10	14	15	15	19	20
5	17	1	2	6	15	16	32
3	21	5	2	2	3	4	8
2	22	14	6	3	2	6	5
1	22	30	15	7	3	11	5
3	26	26	16	7	4	4	8

Figure 1. DTW Example (left) Distance of each individual cell (right) Matrix with path sums presented

2.3 Review of Time Series Classification Algorithms

This subsection introduces relevant algorithms used in the field of time series classification.

2.3.1 K- Nearest Neighbor Classification

KNN classification is a simple classification algorithm that as proven very effective in time series classification. KNN is a type of instance-based learning where the function is only approximated locally, and all computations are deferred until classification. The algorithm works by finding the K nearest training instances to the test instance. Time series classification literature has found that a K = 1 is generally effective. With a K of 1, the algorithm finds the single closest training instance, based on the selected distance metric, and assigns the label of that training instance to the corresponding test instance.

Several variations from the classic 1-NN DTW have been introduced that aim to give a better approximate of the distance between two series. Some of the variations include:

-*Complexity Invariant Distance* (CID) (Batista, 2014): CID computes complexity differences between time series pairs, then uses it as a correction weight in the distance calculation. The complexity is calculated using the sum of squares of the first difference of time series. As the difference in complexity of time series increase, the distance also increases.

-*Derivative DTW* (DDDTW) (Górecki, 2013): The DDDTW uses a weighted combination of original time series distances and first order differences in a nearest neighbor classification. These two distances are combined with a weighted parameter that is determined by through leave-one-out cross validation.

-*Derivative Transform Distance* (DTDc) (Górecki, 2014): DTDc is an extension of DDDTW that uses DTW in conjuncture with derivatives.

-*Move-Split-Merge* (MSM) (Stefan, 2013): MSM distance offers a measure that is like common distance-based approaches, where similarity is determined through Move and Split operations to transform a given time series.

-*Time Warp Edit* (TWE) (Marteau, 2009): TWE is an elastic distance matric that contains features of the longest common subsequence and DTW. It contains a stiffness parameter that controls the elasticity. This parameter allows TWE to be a middle ground between the stiff Euclidian distance and the flexible DTW.

-*Weighted DTW* (WDTW) (Jeong, 2011): WDTW is a form of DTW that adds a weight to nearer neighbors. This weight depends on the difference of phases between a reference point and a query point. WDTW penalizes the points distances according to the phase difference.

2.3.2 Bag of Word Classifiers

Several time series classification algorithms work by first transforming time series data into bag of words approximation, then performing a classification method. Such methods include

-*Shotgun Classifier* (Schafer, 2014): The Shotgun classifier is based on 1-NN classification with a Shotgun distance. The shotgun distance between two time series is given by aggregating the minimal Euclidean distance between each disjoint query window and each offset, represented by the sliding windows.

-*Shotgun Ensemble Classifier* (Schafer, 2014): Shotgun Ensemble classifier ensembles over the Shotgun classifier with varying windowing parameters for each component of the classifier.

-*BOSSVS Classifier* (Schafer, 2015): The BOSSVS classifier is an ensemble classier that uses a set of windowing parameters to bin the time series data using equal-depth binning. Windowing parameters that prove promising through cross validation are used for 1-NN classification where the distance metric compares bags of words based on Euclidean distance.

-*BOSS Ensemble Classifier* (Schafer, 2016): BOSS Ensemble classification is similar to BOSSVS. The primary difference between the two methods is that BOSSVS uses fewer windowing parameters to decrease the modeling time. The increased number of windowing parameters allows for the BOSS Ensemble classifier to provide a higher accuracy.

-*WEASEL Classifier* (Schafer, 2017): The WEASEL classifier is a bag of word classifier that has two main improvements over the BOSS Ensemble classifier. First, the binning process is calculated using information gain. Second, the 1-NN classification process is substituted for a Linear SVM classifier.

-*MUSE Classifier* (Schafer, 2017): The MUSE classifier is an implementation of the WEASEL classifier that is used of multivariate time series datasets.

2.3.3 Shapelet Based Classifiers

Shapelets are time series subsequences that are discriminatory of the membership to a class. Shapelets can be used as the splitting criterion when applying a decision tree for classification.

-*Fast Shapelets* (FS) (Rakthanmanon , 2013): FS is an extension of the decision tree shapelet that accelerates shapelet discovery. In FS a frequency count histogram is built using multiple random projections.

-*Shapelet Transform Ensemble* (ST) (Bagnall, 2015): ST splits the shapelet discovery by finding the top k shapelets on each run. ST can be utilized using k-NN, Naïve Bayes, decision tree, support vector machine, etc. Each classifier is assigned a weight based on the cross-validation training accuracy.

-*Learned Shapelets* (LS) (Grabocka, 2014) [55]: LS finds k shapelets using k-means clustering of candidates from the training data. A logistic loss function is the objective function according to a logistic regression model for each class. The weights of the regression and shapelets are learned by the algorithm. A check is performed at certain intervals as to whether divergence has occurred.

3 Network Based Sampling

This chapter introduces Network Based Sampling (NBS), a method of reducing the processing time of time series classification algorithms.

Network Based Sampling is a method of determining which training instances in a dataset have the highest priority when used for classification. NBS works by constructing a network based on the training instances, then sampling from the training instances based on the network connections. Figure 1 illustrates how NBS fits into the typical classification pipeline. The following subsections detail the procedure within the Network Generation block and the NBS block.

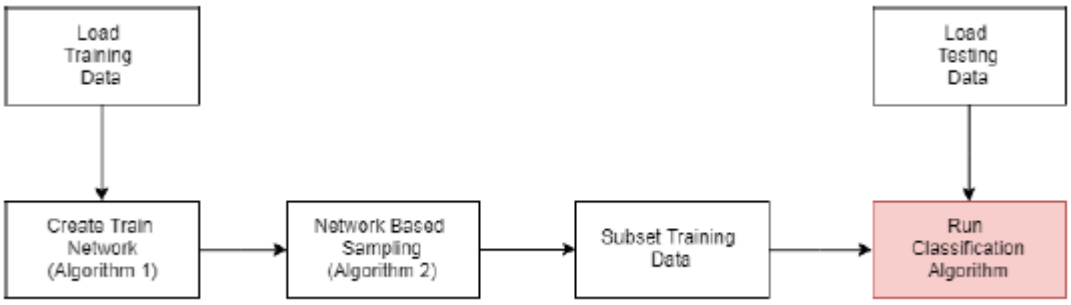

Figure 2. NBS Classification Framework

3.1 Training Network Generation

The network generation works by computing the distance of each training instance to all other training instances excluding itself. This distance is computed using Euclidean Distance. ED is used for network generation because it is inexpensive to compute, relative to DTW. Following the distance calculating in the network generation process, there is a need for a parameter k. k is the number of nearest training instances the network maintains for each instance. Parameter k is calculated as:

$$k = round(\log_5(train\ instances) + \log_{10}(series\ length))$$

Algorithm 1 gives the full process of generating the training network. The network contains the k nearest training instances to each instance. The network is stored as a digraph, where connections point toward the instances that they are closest.

Algorithm 1 Network Generation of Training Instances

```
function TrainNetwork(train, k)
    n = train.instances
    E = Array{(n x n), Inf} //Array of Infintes
    for i ∈ 0...n − 1 do
        for j ∈ 0...n − 1 do
            if i! = j then
                E[i][j] = ED(train[i], train[j]) //Calculate Eucluidean Distance
            end if
        end for
    end for
    connections =[ ]
    for i ∈ 0...n − 1 do
        for j ∈ 0...k − 1 do
            distance, index = findmax(E[i])
            correct = train[i].label == train[index].label
            connections.append([i, index, distance, correct])
            E[i][index] = Inf
        end for
    end for
    return connections
end function
```

Figure 3. Algorithm 1 Network Generation of Training Instances

3.2 Sampling and Classification

The next step in the classification process is to sample from the training set based on the generated network, the training labels, and p (percent of the training set to keep). The value of p is calculated as follows, where n1 is the number of test instances, n2 is the number of training instances, and m is the series length.

$$p = round\left(\frac{n1 * n2 * m^2 - n2 * n2 * m}{n1 * n2 * m2}, 2\right) - threshold$$

The threshold parameter of the p calculation corresponds to the expected time savings in the classification process. The value of threshold must follow 0<threshold<1. As a base the threshold is set to 15% in our tests, however the value is up to the user based on the required time savings. Algorithm 2 goes through the process for Network Based Sampling. NBS starts by calculating a limit to the number instances per each class in the sampled training set, c limit. This limit is used to ensure that the class weight is similar to that of the original training set. The need for this constraint is apparent in classification examples where there exists a clear minority class. In this case, the NBS algorithm would likely eliminate the instances of the minority class during sampling. NBS then goes through every training instance and assigns a weight based on the number of indegree instances that point to this training instance and have the same class label. Given the weights and sampling constraints, the algorithm then determines which training instances to maintain. The output of this algorithm is the indices of the training set that should be kept for classification. The full classification process with NBS is expressed in

Algorithm 3.

Algorithm 2 Network Based Sampling

```
function NBS(connections, labels, p)
    weights = labels.weights //Dictionary of class frequency (between 0 and 1)
    c_margin = 1.05
    c_limit = {e : round((weights[e] * labels.length * p) * c_margin) for e ∈ weights.keys}
    c_count = {e : 0 for e ∈ weights.keys}
    t_limit = round(labels.length * p)
    t_count = 0
    Holder = Array{(labels.length x 3), 0}
    Holder[:, 0] = [i for i ∈ 0 : (labels.length − 1)]
    for i ∈ 0 . . . (labels.length − 1) do
        friends = connections[connections['Last']. == i, :] //Get this instances friends
        f = friends.instances
        if friends.instances! = 0 then
            Holder[i, 1] = sum(friends['Correct'])/friends['Correct'].length) * (f)
        end if
        Holder[i, 2] = labels[i]
    end for
    holder = Holder[Holder[:, 1].sort[end : −1 : 1], :]
    keep =[ ]
    for i ∈ 0 . . . (holder.size[0] − 1) do
        l = holder[i, 2]
        if c_count[l] < c_limit[l] and t_count < t_limit then
            keep.append(holder[i, 0])
            t_count+=1
            c_count[l]+=1
        end if
    end for
    return keep
end function
```

Figure 4. Algorithm 2 Network Based Sampling

Algorithm 3 Classification Process with Network Based Sampling

function CLASSIFICATION($train$, $test$, $threshold$)
$n1 = train.instances$ //Number of training examples
$n2 = test.instances$
$m = train[0].length$ //Length of single time series
$p = round((n1 * n2 * m^2 - n2^2 * m)/(n1 * n2 * m^2), 2) - threshold$
$k =$
$connections = TrainNetwork(train, k)$
$index = NBS(connections, train.labels, p)$
$train = train[index]$ //Subset training examples
$accuracy =$ classifier($train$, $test$) //Desired Classification Method
return $accuracy$
end function

Figure X. Algorithm 3 Classification Process with Network Based Sampling

4 Experiments and Results

The NBS framework is tested on the UCR Time Series Classification Archive (Chen, Keogh, Hu, Begum, Bagnall, Mueen and Batista, 2015). The UCR archive consists of 85 time series classification benchmarks from a variety of fields. These benchmarks vary widely in terms of time series length, training size and testing size. NBS is applied to benchmark training sets and used to evaluate the classification accuracy of testing sets. The NBS framework is implemented in both Python (3.5.2) and Julia (0.6.2). All experiments were performed using a single core on an Intel Xeon CPU E5-2620 processor sing the Julia code base.

4.1 UCR Archive Results

Table 1 summarizes the results obtained from various threshold on NBS classification. Figure 2 [left] illustrates how NBS preforms in terms of both accuracy and time. At a threshold of 15% NBS improves in both speed and accuracy of 39 datasets, improves accuracy but not speed in 1 dataset, and improves time but not accuracy in 45 datasets. Increasing the threshold to 30% and 50% results in a decrease in classification accuracy but a clear increase in the speed of the NBS algorithm. Figure 2 [right] illustrates these results in a different way. In this representation, green points undergo an expected reduced in the time saved. Formally, with a threshold of 15%, the classification time of green data point is < 85% the time required for the full 1-NN DTW classification.

Table 1: NBS Classification Results 50words-Lightning2

Name	ACC_Full	Time_Full	ACC_15	Time_15	ACC_30	Time_30	ACC_50	Time_50
50words	0.69	304.622	0.655	283.578	0.662	194.997	0.644	160.359
Adiac	0.604	106.803	0.578	87.031	0.558	63.638	0.509	48.897
ArrowHead	0.703	8.789	0.697	7.35	0.714	5.692	0.629	4.029
Beef	0.633	4.447	0.567	3.271	0.567	2.778	0.567	1.926
BeetleFly	0.7	2.782	0.7	1.985	0.65	1.674	0.6	1.268
BirdChicken	0.75	2.446	0.75	1.926	0.75	1.428	0.65	1.319
Car	0.733	25.581	0.6	21.907	0.55	16.507	0.567	11.868
CBF	0.997	12.415	0.997	10.033	0.997	8.361	0.999	5.921
ChlorineConcentration	0.648	1415.462	0.595	1040.274	0.555	789.754	0.534	579.331
CinC_ECG_torso	0.651	3209.453	0.528	2616.455	0.499	1867.744	0.4	1401.87
Coffee	1	1.442	1	1.332	1	1.061	0.929	0.822
Computers	0.7	714.255	0.648	578.058	0.656	432.063	0.632	298.244
Cricket_X	0.754	277.754	0.718	227.934	0.71	192.018	0.649	147.538
Cricket_Y	0.744	290.162	0.715	226.567	0.679	172.951	0.654	146.607
Cricket_Z	0.754	278.96	0.733	227.983	0.726	173.366	0.726	147.927
DiatomSizeReduction	0.967	12.518	0.967	10.222	0.892	7.985	0.892	5.093
DistalPhalanxOutlineAgeGroup	0.793	9.87	0.783	6.907	0.838	5.534	0.825	4.28
DistalPhalanxOutlineCorrect	0.768	28.214	0.773	19.357	0.772	19.039	0.78	15.056
DistalPhalanxTW	0.71	9.988	0.735	6.09	0.738	6.536	0.748	4.481
Earthquakes	0.742	272.592	0.752	204.178	0.711	152.925	0.711	131.046
ECG200	0.77	2.29	0.79	2.115	0.8	1.374	0.78	1.233
ECG5000	0.924	1086.667	0.936	875.29	0.94	727.349	0.922	523.387
ECGFiveDays	0.768	9.461	0.753	7.271	0.741	6.238	0.721	4.665
ElectricDevices	0.596	16684.002	0.589	11799.963	0.584	9159.576	0.564	6379.88
FaceAll	0.808	418.951	0.778	379.234	0.802	269.745	0.793	236.4
FaceFour	0.83	6.398	0.648	4.527	0.648	4.026	0.625	2.707
FacesUCR	0.905	195.734	0.895	169.703	0.878	123.348	0.818	91.514
fish	0.823	143.039	0.817	111.776	0.783	82.38	0.754	61.902
FordA	0.562	25796.179	0.549	19554.312	0.524	14441.847	0.524	12917.324
FordB	0.594	15926.718	0.578	12223.279	0.584	10918.759	0.572	8012.09
Gun_Point	0.907	3.803	0.907	3.414	0.86	2.352	0.773	1.768
Ham	0.467	49.9	0.505	40.227	0.486	30.112	0.59	23.587
HandOutlines	0.798	54819.232	0.809	46841.018	0.805	30448.548	0.806	25356.591
Haptics	0.377	1097.398	0.403	1011.903	0.409	652.622	0.416	546.489
Herring	0.531	22.644	0.547	16.73	0.547	13.343	0.547	10.832
InlineSkate	0.384	3746.995	0.349	3431.861	0.365	2383.038	0.329	1619.196
InsectWingbeatSound	0.355	643.968	0.352	504.557	0.358	410.906	0.332	314.118
ItalyPowerDemand	0.95	2.436	0.944	0.816	0.941	0.632	0.93	0.601
LargeKitchenAppliances	0.795	1534.486	0.789	1291.902	0.773	902.402	0.763	703.216
Lighting2	0.869	34.348	0.869	27.481	0.869	19.656	0.869	14.982

Table 2: NBS Classification Results Lightning7-yoga

Name	ACC_Full	Time_Full	ACC_15	Time_15	ACC_30	Time_30	ACC_50	Time_50
Lighting7	0.726	12.061	0.712	9.323	0.699	6.549	0.616	5.559
MALLAT	0.934	3162.481	0.919	2608.97	0.815	1867.846	0.819	1424.846
Meat	0.933	16.092	0.933	12.881	0.933	9.646	0.933	6.981
MedicalImages	0.737	70.059	0.7	50.995	0.659	42.359	0.642	35.053
MiddlePhalanxOutlineAgeGroup	0.75	9.632	0.75	8.34	0.745	6.518	0.75	5.197
MiddlePhalanxOutlineCorrect	0.648	27.258	0.678	23.487	0.682	20.888	0.637	16.157
MiddlePhalanxTW	0.584	9.561	0.584	7.747	0.584	7.912	0.576	5.868
MoteStrain	0.835	4.749	0.837	3.367	0.865	3.537	0.904	1.858
NonInvasiveFatalECG_Thorax1	0.79	39048.467	0.781	34746.679	0.777	24282.966	0.757	19376.977
NonInvasiveFatalECG_Thorax2	0.865	38538.87	0.863	29608.566	0.85	24184.046	0.831	17993.307
OliveOil	0.833	6.469	0.833	4.684	0.833	4.278	0.833	2.816
OSULeaf	0.591	198.668	0.55	163.329	0.521	118.985	0.512	84.94
PhalangesOutlinesCorrect	0.728	264.333	0.76	142.969	0.745	154.909	0.746	163.957
Phoneme	0.228	11077.128	0.222	9449.488	0.228	7464.473	0.206	5256.398
Plane	1	5.441	1	4.785	1	3.218	1	2.52
ProximalPhalanxOutlineAgeGroup	0.805	12.677	0.849	9.531	0.844	6.373	0.844	7.003
ProximalPhalanxOutlineCorrect	0.784	27.264	0.825	23.22	0.814	14.18	0.808	16.588
ProximalPhalanxTW	0.738	12.791	0.77	13.114	0.765	8.598	0.8	7.008
RefrigerationDevices	0.464	1562.009	0.48	1378.232	0.464	1007.618	0.445	706.55
ScreenType	0.397	1373.522	0.4	1037.923	0.411	799.273	0.411	706.759
ShapeletSim	0.65	25.068	0.622	21.909	0.656	16.968	0.667	11.725
ShapesAll	0.768	2229.043	0.742	1853.537	0.71	1354.149	0.678	1022.99
SmallKitchenAppliances	0.643	1471.849	0.656	1223.359	0.645	904.487	0.621	676.039
SonyAIBORobotSurface	0.725	2.042	0.73	1.298	0.681	1.055	0.699	0.709
SonyAIBORobotSurfaceII	0.831	3.54	0.805	2.456	0.773	1.889	0.771	1.746
StarLightCurves	0.907	182185.837	0.913	144017.825	0.918	106812.766	0.918	81469.584
Strawberry	0.94	276.657	0.91	204.167	0.884	174.152	0.886	134.878
SwedishLeaf	0.792	109.736	0.795	103.368	0.787	86.546	0.766	59.035
Symbols	0.95	84.47	0.94	69.874	0.944	50.816	0.927	37.859
synthetic_control	0.993	12.339	0.993	8.627	0.99	4.282	0.99	5.707
ToeSegmentation1	0.772	15.141	0.772	13.631	0.746	12.566	0.754	7.304
ToeSegmentation2	0.838	11.477	0.815	10.774	0.808	9.038	0.8	5.597
Trace	1	17.931	0.97	14.182	0.97	10.757	0.95	8.231
Two_Patterns	1	1805.672	1	1527.167	1	1184.738	1	850.14
TwoLeadECG	0.904	4.905	0.822	3.728	0.829	3.076	0.773	2.217
uWaveGestureLibrary_X	0.728	7026.19	0.722	5606.22	0.716	4390.115	0.702	3172.555
uWaveGestureLibrary_Y	0.634	6690.92	0.635	5960.82	0.626	4364.935	0.608	3202.459
uWaveGestureLibrary_Z	0.658	6064.547	0.661	4802.642	0.656	3649.945	0.656	3263.36
uWaveGestureLibraryAll	0.892	56138.737	0.881	47549.629	0.87	36970.035	0.837	25150.741
wafer	0.98	3001.005	0.974	2468.341	0.972	2074.744	0.97	1544.846
wine	0.574	3.626	0.519	2.765	0.463	2.318	0.481	1.823
WordsSynonyms	0.649	229.946	0.641	196.324	0.614	178.662	0.578	132.32
Worms	0.464	256.086	0.481	220.94	0.442	155.234	0.453	116.935
WormsTwoClass	0.663	256.583	0.685	210.014	0.657	154.356	0.608	118.077
yoga	0.837	3531.35	0.806	3007.693	0.785	2098.382	0.77	1607.097

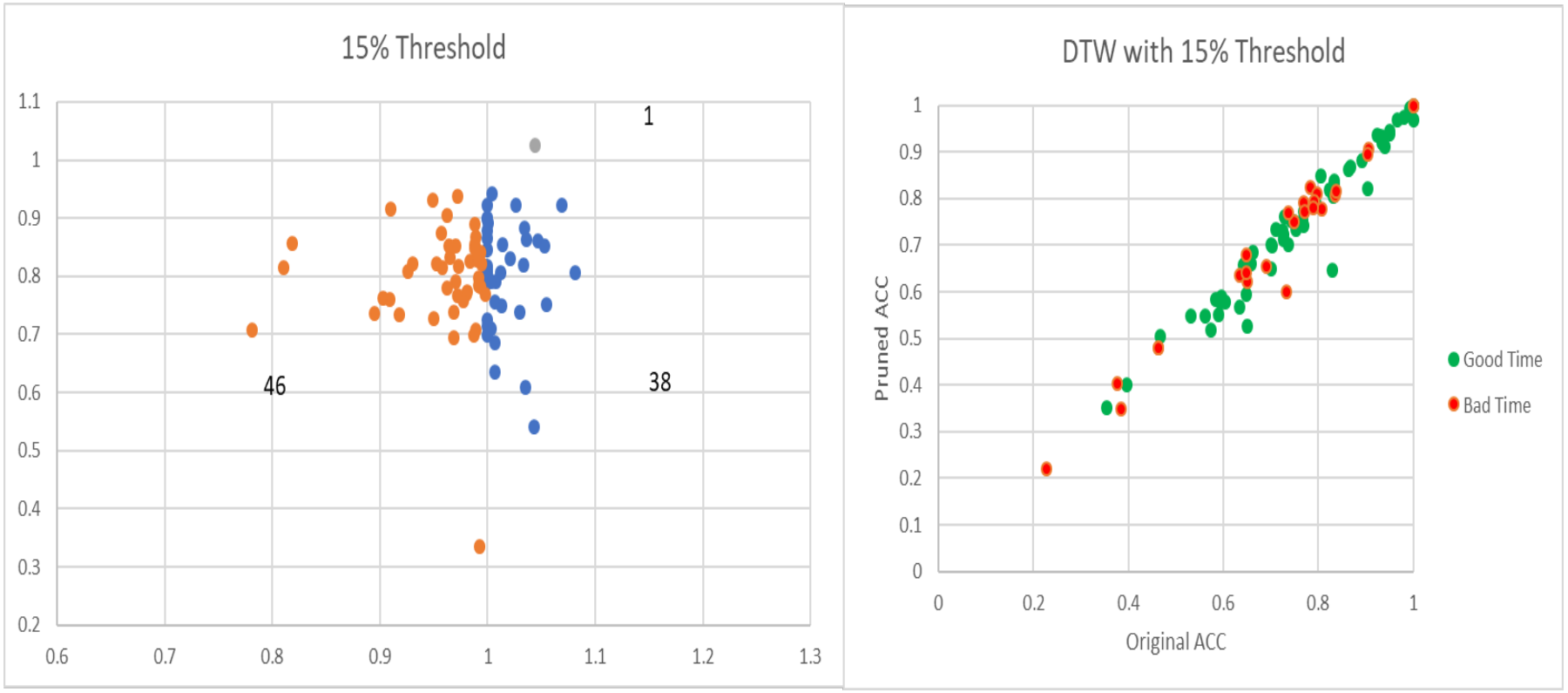

Figure 6. NBS Classification Results of threshold of 15%

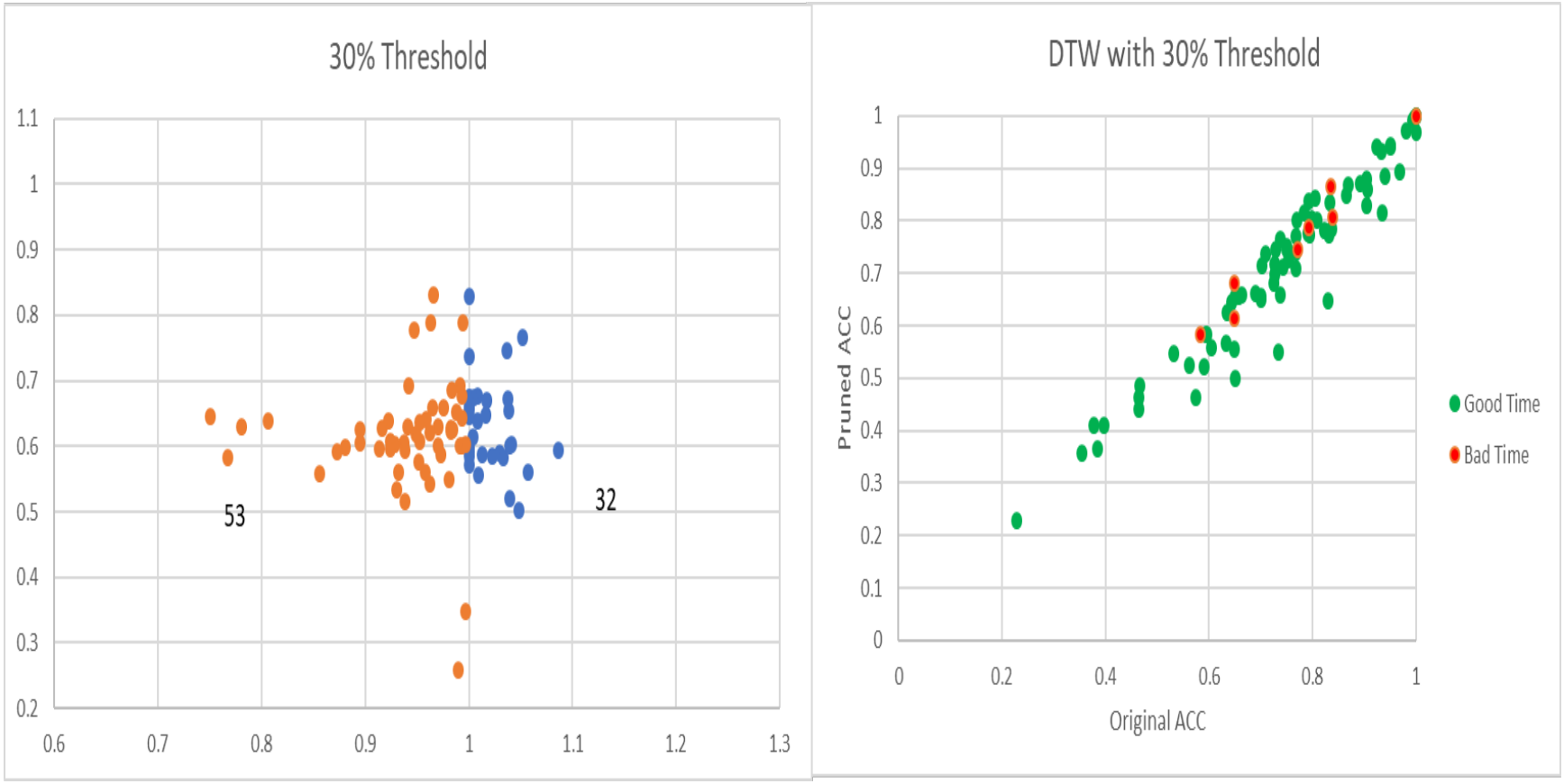

Figure 7. NBS Classification Results of threshold of 30%

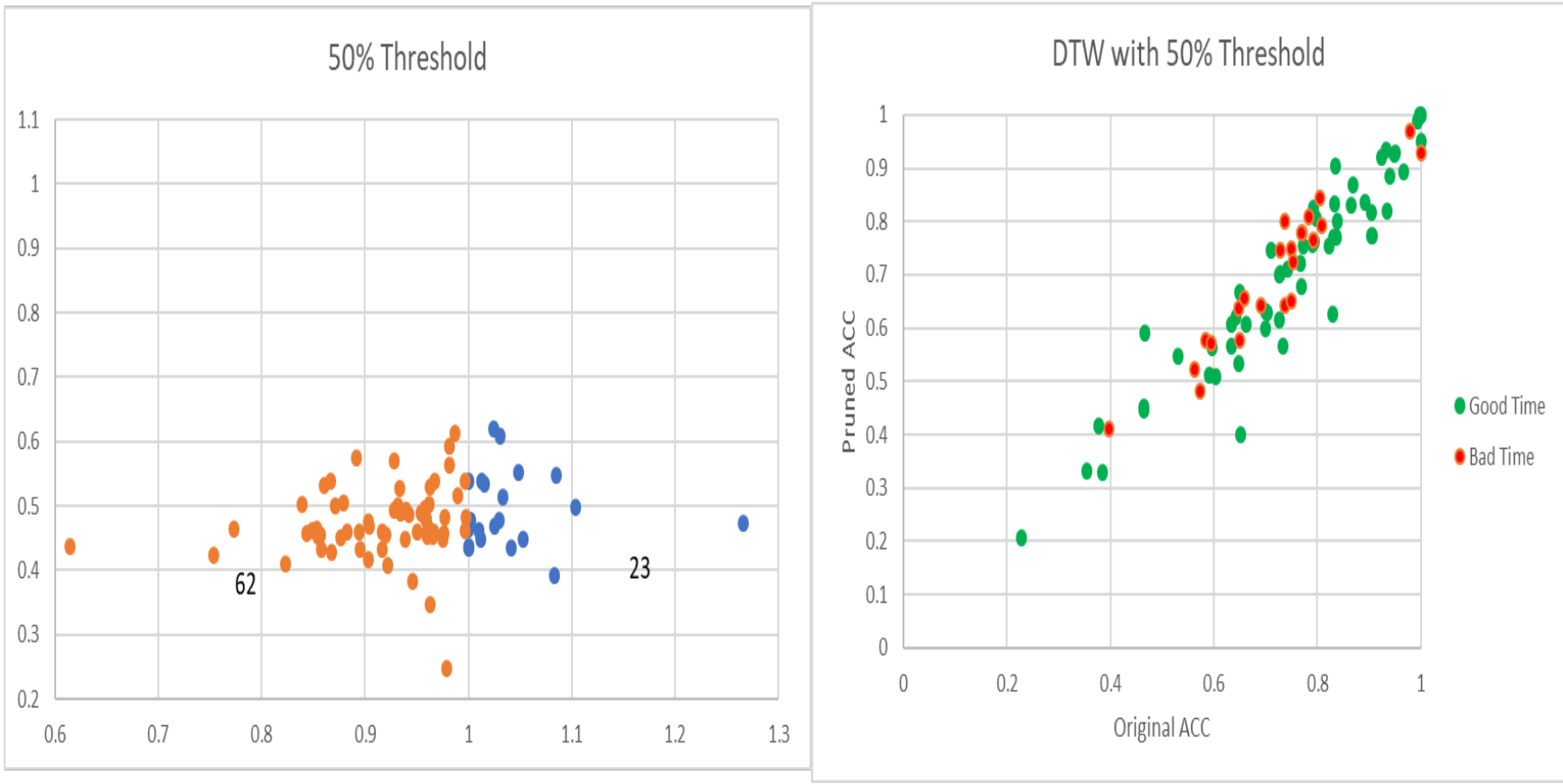

Figure 8. NBS Classification Results of threshold of 50%

5 Conclusion

This dissertation introduces Network Based Sampling for time series classification as a method of reducing computation in traditional classification algorithms. The developed framework is applicable on any time series database. NBS is tested on the 85 benchmarks available in the UCR Time Series Archive. Our results indicate that the framework improves or maintains accuracy for 39 out of 85 datasets at a threshold of 15% with an expected time savings that linearly corresponds to the threshold. This ratio decreases as the threshold is increased, but there is a benefit of faster classification time.

5.1 Future Work

Future improvements to NBS may be accomplished by the following:

Optimizing parameter selection: The NBS framework relies on several parameters the influence the potential time saved and effects on classification accuracy. In practice, optimizing parameters would allow for sufficient time savings while maintaining the required accuracy for a give situation.

Multivariate time series adjustment: The NBS framework can to extended to accommodate multivariate time series data sources. The investigation of this extension can be considered part of this studies future work.

CITED LITERATURE

1. Bagnall, A; Lines, Jason; Hills, Jon; Bostrom, Aaron. "Time-series classification with COTE: the collective of transformation-based ensembles." IEEE Transactions on Knowledge and Data Engineering 27, no. 9 (2015): 2522-2535.

2. Batista, Gustavo EAPA, Eamonn J. Keogh, Oben Moses Tataw, and Vinícius MA de Souza. "CID: an efficient complexity-invariant distance for time series. "Data Mining and Knowledge Discovery 28, no. 3 (2014): 634-669.

3. Berndt, D. J. and Clifford, J. (1994), Using dynamic time warping to find patterns in time series., in `KDD workshop', Vol. 10, Seattle, WA, pp. 359-370.

4. Chen, Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A. and Batista, G. (2015), `The ucr time series classification archive', http://www.cs.ucr.edu/~eamonn/time_series_data/. (Accessed May 2018)

5. Chu, S., Keogh, E., Hart, D. and Pazzani, M. (2002), Iterative deepening dynamic time warping for time series, in `Proceedings of the 2002 SIAM International Conference on Data Mining', SIAM, pp. 195-212.

6. CMU Graphics Lab Motion Capture Database (2014), http://mocap.cs.cmu.edu/.

7. Cook, D. J. and Holder, L. B. (2006), Mining graph data, John Wiley & Sons.

8. Ding, H., Trajcevski, G., Scheuermann, P.,Wang, X. and Keogh, E. (2008), `Querying and mining of time series data: experimental comparison of representations and distance measures', Proceedings of the VLDB Endowment 1(2), 1542-1552.

9. Fu, T.-c. (2011), `A review on time series data mining', Engineering Applications of Artificial Intelligence 24(1), 164-181.

10. Grabocka, Josif; Schilling, Nicolas; Wistuba, Martin; Schmidt-Thieme, Lars. Learning time-series shapelets. In Proc. 20th SIGKDD, 2014

11. Górecki, Tomasz and Maciej Luczak. Using derivatives in time series classification. Data Mining and Knowledge Discovery, 26(2):310–331, 2013.

12. Górecki, Tomasz and Maciej Luczak. Non-isometric transforms in time series classification using DTW. Knowledge-Based Systems, 61:98–108, 2014

13. Gregor, H. (2016), `An evaluation of combinations of lossy compression and change-detection approaches for time-series data'.

14. Jeong, Y; Myong J; and Olufemi, O. "Weighted dynamic time warping for time series classification." Pattern Recognition 44, no. 9 (2011): 2231-2240.

15. Kalpakis, K., Gada, D. and Puttagunta, V. (2001), Distance measures for effective clustering of arima time-series, in `Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on', IEEE, pp. 273-280.

16. Karim, F., Majumdar, S., Darabi, H. and Chen, S. (2018), `Lstm fully convolutional networks for time series classification', IEEE Access 6, 1662-1669.

17. Keogh, E. J. and Pazzani, M. J. (2000), Scaling up dynamic time warping for datamining applications, in `Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, pp. 285-289.

18. Keogh, E. and Kasetty, S. (2003), `On the need for time series data mining benchmarks: a survey and empirical demonstration', Data Mining and knowledge discovery 7(4), 349-371.

19. Keogh, E. and Ratanamahatana, C. A. (2005), `Exact indexing of dynamic time warping', Knowledge and information systems 7(3), 358-386.

20. Keogh, E., Wei, L., Xi, X., Vlachos, M., Lee, S.-H. and Protopapas, P. (2009), `Supporting exact indexing of arbitrarily rotated shapes and periodic time series under Euclidean and warping distance measures', The VLDB journal 18(3), 611-630.

21. Liao, T. W. (2005), `Clustering of time series data a survey', Pattern recognition 38(11), 1857-1874.

22. Lin, J., Keogh, E., Lonardi, S. and Chiu, B. (2003), A symbolic representation of time series, with implications for streaming algorithms, in `Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery', ACM, pp. 2-11.

23. Marteau, P. Time warp edit distance with stiffness adjustment for time series matching. IEEE Transactions on Pattern Analysis and Machine Intelligence, 31(2):306–318, 2009.

24. Nair, B. B., Kumar, P. S., Sakthivel, N. and Vipin, U. (2017), `Clustering stock price time series data to generate stock trading recommendations: An empirical study', Expert Systems with Applications 70, 20-36.

25. Rakthanmanon, Thanawin and Eamonn Keogh. Fast-shapelets: A fast algorithm for discovering robust time series shapelets. *In Proc. 13th SDM*, 2013.

26. Salvador, S. and Chan, P. (2007a), `Toward accurate dynamic time warping in linear time and space', Intelligent Data Analysis 11(5), 561-580.

27. Schäfer, P. "Experiencing the Shotgun Distance for Time Series Analysis." Trans. MLDM 7.1 (2014): 3-25.

28. Schäfer, P. "The BOSS is concerned with time series classification in the presence of noise." Data Mining and Knowledge Discovery 29.6 (2015): 1505-1530.

29. Schäfer, P. "Scalable time series classification." Data Mining and Knowledge Discovery 30.5 (2016): 1273-1298.

30. Schäfer, P; and Leser, U. "Multivariate time series classification with weasel+ muse." arXiv preprint arXiv:1711.11343 (2017).

31. Schafer, P. and Leser, U. (2017), Fast and accurate time series classification with weasel, in `Proceedings of the 2017 ACM on Conference on Information and Knowledge Management', ACM, pp. 637-646.

32. Schäfer, Patrick, and Ulf Leser. "Multivariate time series classification with weasel+ muse." arXiv preprint arXiv:1711.11343 (2017).

33. Sharabiani, A., Darabi, H., Harford, S., Douzali, E., Karim, F., Johnson, H. and Chen, S. (2018), `Asymptotic dynamic time warping calculation with utilizing value repetition', Knowledge and Information Systems pp. 1-30.

34. Sharabiani, A., Darabi, H., Rezaei, A., Harford, S., Johnson, H. and Karim, F. (2017), `Efficient classification of long time series by 3-d dynamic time warping', IEEE Transactions on Systems, Man, and Cybernetics: Systems 47(10), 2688-2703.

35. Stefan, A; Vassilis, A; and Gautam D. The Move-Split-Merge metric for time series. IEEE Transactions on Knowledge and Data Engineering, 25(6):1425–1438, 2013.

36. Tavenard, R. and Amsaleg, L. (2015), `Improving the efficiency of traditional dtw accelerators', Knowledge and Information Systems 42(1), 215-243.

37. The BIDMC Congestive Heart Failure Database (2014), http://www.physionet.org/physiobank/database/chfdb/.

38. Wang, X., Mueen, A., Ding, H., Trajcevski, G., Scheuermann, P. and Keogh, E. (2013), `Experimental comparison of representation methods and distance measures for time series data', Data Mining and Knowledge Discovery 26(2), 275-309.

39. Wilson, S. J. (2017), `Data representation for time series data mining: time domain approaches', Wiley Interdisciplinary Reviews: Computational Statistics 9(1).

40. Xing, Z., Pei, J., Yu, P. S. and Wang, K. (2011), Extracting interpretable features for early classification on time series, in `Proceedings of the 2011 SIAM International Conference on Data Mining', SIAM, pp. 247-258.

41. Ye, L. and Keogh, E. (2009), Time series shapelets: a new primitive for data mining, in `Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, pp. 947-956.

VITA

Samuel Harford

SEL 4209, 842 W Taylor, Chicago, IL 60607. Phone 312-355-4677
Email: sharfo2@uic.edu

Education

University of Illinois at Chicago (UIC) Chicago, IL (Aug. 2016 -Aug. 2018)

M.S. Industrial Engineering

University of Illinois at Chicago (UIC) Chicago, IL (Aug. 2012 -Aug. 2016)

B.S. Industrial Engineering

Publications

Journal

- "Network Based Time Series Sampling." Knowledge and Information Systems (In Review).
- "Multivariate LSTM-FCNs for Time Series Classification." IEEE Transactions on Neural Networks and Learning Systems (In Review).
- "Asymptotic Dynamic Time Warping calculation with utilizing value repetition." Knowledge and Information Systems (2018): 1-30.
- "Efficient Classification of Long Time Series by 3-D Dynamic Time Warping." IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2017.

Conference

- "Beyond Grade Point Average and Standardized Testing: Incorporating a Socio-Economic Factor in Admissions to Support Minority Success." 2017 ASEE Annual Conference & Exposition. 2017.
- "Life after University for Engineering Graduates." 2017 ASEE Annual Conference & Exposition. 2017.
- "A Comparative Analysis of Underrepresented Engineering Applicants Ad-mission Practices and their Academic Performance at the University of Illinois at Chicago." 123rd ASEE Annual Conference and Exposition. American Society for Engineering Education, 2016.