Improving the discovery of Motifs in high-dimensional sequences of varying length

M. Adnan Haider

Master of Science
School of Informatics
University of Edinburgh

2013

Abstract

Acknowledgements

Many thanks to my mummy for the numerous packed lunches; and of course to Igor, my faithful lab assistant.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*M. Adnan Haider*)

Table of Contents

Chapter 1

Introduction

Over the course of the last decade, the mining of time-series data have received considerable attention within the data mining and machine learning community. The term 'time series' denotes a set of observations concurring any activity against different periods of time. The duration of time period may be either in the order of milliseconds or monthly or even annually depending on the domain.

Mathematically, a time series is defined by the values $y_1, y_2 ... y_n$ at times $t_1, t_2 ... t_n$ where $y = f(t)$. The time t_i acts as an independent variable to estimate dependent variables y_i. The dimensionality of the series is denoted as **n** where the value of '**n**' is equal to the length of the sequence.

Time series analysis is used in many applications ranging from sales forecasting, budgetary analysis to stock market analysis and many more. One particular domain where the application of time series analysis is currently very popular is *motif* discovery- the problem of efficiently locating frequent/interesting sub-patterns in the data. The knowledge of motifs has been seen to have important applications in various aspects of data mining tasks. For instance motifs can use applied :

- to discover association rules [1] that the reflect information of 'primitive shapes.
- to specify the number of clusters for unsupervised clustering algorithms. The knowledge of motifs give a good approximation on the number of meaningful groups that are present in the data[2].
- to identify important sub-patterns in DNA and gene sequences [3].

In the analysis of speech data, motifs also play a very important role. Recent research have shown that detecting and isolating motifs in speech utterances is equivalent to identifying words that are specific to genres or in the case of

recordings of individual speakers, detecting words that are frequently spoken by them. [4, 5].

To identify and extract motifs from time series data, various clustering algorithms have been proposed. Among them, the most widely used is the Dynamic time warping algorithm(DTW) [6, 7, 8, 9, 10, 11] that clusters similar sequences separated by time shifts, length or scale. For time series sequences, the algorithm performs a much richer comparison between sequences than standard metrics such as the euclidean. By warping the time axis, the DTW algorithm compares each point in one sequence with points at different temporal regions in the second sequence. This allows the algorithm to be invariant to changes of speed, length or scale between similar patterns. Metrics such as the euclidean on the other hand are constrained to performing only linear matches between the temporal dimensions of sequences and they require the all sequences to share the same length. This provides a limitation on the type of time series domains such metrics can be applied to. The speech corpus is an prime example of one such domain where metrics such as the euclidean cannot be applied directly. Data sets comprised of speech utterances are a good example where recorded utterances do not share the same dimensionality (i.e the same length) as acoustically similar signals may be contracted/expanded versions of each other due to speaker variations, context etc.

However, the DTW algorithm also experiences from severe drawbacks. The algorithm suffers from large run-times when the length of the time series sequences are very long. This is because the time complexity of the algorithm is quadratic i.e O(n^2) where n is the length of the sequence. In comparison, standard metrics like the euclidean has a time complexity of O(n), making them computational less expensive to employ. To address the quadratic time complexity of the DTW algorithm window constraints (Itakura parallelogram[13], Sakoe-Chiba band[7]) have been proposed to reduce the size of the search space of DTW by forcing the algorithm to look for optimal paths that are along the diagonal. Although introducing such window constraints does improve the time complexity, the reduction in the search space however leads to a substantial decrease in the accuracy of the algorithm since all optimal paths may not exist around the diagonal region[11].

The goal of this project is to employ machine learning techniques to tackle and resolve the individual drawbacks associated with using the DTW algorithm and the euclidean metric to compute the similarity between high dimensional time series sequences. In the first half of the project, I will be solely concentrating on improving the performance of the DTW algorithm for problem domains where the minimising the time complexity is as high priority as achieving good accuracy. In the latter half of the project, I will be concentrating on improving the performance of the euclidean metric in time series domains where intra

class sequences share the same global shape. To be precise, I will be exploring feature extraction techniques to extract a smaller set of independent features from the raw sequences that can capture information about the global shape allow sequences to be accurately compared only through a linear match of the feature dimensions.

For this project, I will be primarily using 3 time series datasets (details given in chapter 2):

- TIDIGITS corpus
- InLineSkate
- Cinc_ECG_TORSO

To evaluate and compare the merits of different proposed changes, I have partitioned each of the above datasets into a training set and a test set and employed the 1 nearest neighbour classifier to classify the test instances. In the first half of the project, the DTW was used as the similarity metric while for the later analysis, the euclidean metric was employed as the similarity metric. The performance of the classifier has been used to compare different proposed adaptions of the DTW and evaluate the performance of using different feature sets when employing the euclidean metric. The reason for choosing the 1 nearest neighbour classifier is that it shares almost the same methodology as the algorithms used to detect motifs. Motif detection algorithms are memory based i.e like 1NN classifier, they rely on comparing each sequence with other sequences in the dataset. Hence, methodologies that tend to achieve good performance on 1 NN classification can be expected to achieve equivalent performance in motif detection.

The dissertation is organised as follows: Chapter 2 gives a brief description of the time-series datasets used for this project. Chapter 4 provides a detailed background description of the DTW algorithm. Chapter 3 and 4 investigates methods to improve the performance of the DTW algorithm in terms of both accuracy and speed. Chapter 5 explores the noble technique of single value decomposition in the context of time series domains and explores whether it can be applied to extract a smaller set of latent features that can minimise the run time of the 1NN classifier,using the euclidean metric to classify sequences, without decreasing the accuracy. Chapter 6 focusses on improving the accuracy of the classification of 1NN classifier on time series sequences when it employes the euclidean metric. In this chapter, I mainly explore preprocessing techniques that can extract noble features which capture the global properties of sequences. Finally in the last chapter, I present my conclusions and present scope for future work.

Chapter 2

Datasets Used

There are two primary goals of this project :

- improve the performance of the DTW algorithm on long time series sequences.
- adapt SVD to create a unique 'fingerprint' scheme for each distinguished motif/time series pattern.

For each of these algorithms, the performance of different adaptions was evaluated by combining them with the 1 nearest neighbours classifier. As I have mentioned in the previous chapter, the 1 nearest neighbour classifier shares the same methodology as the algorithms used to detect motifs. Thus, if the proposed adaptions improve the performance of 1 nearest neighbour classifier then it is highly likely that they will achieve better performance in detecting motifs. The datasets used for evaluation were carefully chosen so that they contain time series sequences that have very long lengths.

The datasets used for this project are as follows:

1. TIDIGITS speech corpus
2. The UCR data set :CinC_ECG_torso
3. The UCR data set :InlineSkate

All 3 data sets were partitioned into two disjoints sets: test set and training sets. Each sample in the training and test data set corresponds to a particular pattern. The class information of the training samples were used to evaluate the accuracy of the nearest neighbours classifier.

The primary dataset that I have used for this project is the 'TIGITS' corpus. In comparison to the UCR datasets, the length of the time series sequences in the TIGITS corpus are on average 100 times longer than the time series sequences of the two UCR datasets. Description of the data sets is given below:

2.0.1 Brief description of TIDIGITS dataset

The utterances in the TIDIGITS[14] dataset correspond to digit sequences. The corpus consists of :

- 22 isolated digits (two tokens of each of the eleven digits)
- 11 two-digit sequences
- 11 three-digit sequences
- 11 four-digit sequences
- 11 five-digit sequences
- 11 seven-digit sequences

In this corpus, the motifs correspond to individual digits. Thus for my experiments, I have considered only the isolated digit sequences for classification, more specifically the digits from 0 to 9. The database as I have mentioned earlier is divided into two subsets, one to be used for algorithm design and one to be used only for evaluation. Both sets contain samples belonging to different speaker categories i.e (men,women,boy,girl) who in turn have different dialect classifications. The table below shows the partitioning of speakers from different speaker categories in the training and test dataset.:

Subset	Man	Woman	Boy	Girl
Train	55	57	25	26
Test	56	57	25	25

For this project, the entire training set was used for nearest neighbour classification, although only samples from one production was taken. To ensure that the experiments completed with in a reasonable time, I have only used $\frac{1}{3}$ of the samples of the test set.

The test data subset used for evaluation consists of

- 162 random samples from boy category
- 162 random samples from girl category
- 326 random samples from men category
- 326 random samples from women category

Note: The samples were not chosen exactly randomly. When subsampling from each category, I have ensured that:

- the samples belong to a number of different speakers

- the chosen subset consists of enough examples of all 10 digits.

2.0.2 Brief description CinC_ECG_torso and InLineSkate dataset

The UCR data resource has been funded by an NSF(National Science Foundation) since 2003 and is the largest public collection of time series data sets that have been made available to the data mining/machine learning community, to encourage reproducible research for time series classification and clustering. To evaluate the performance of different adaptations of the two algorithms on high dimensional sequences, we require data sets that contain extremely long sequences. The UCR data sets 'CinC_ECG_torso'[15] and the 'InLineSkate' [15] therefore are an excellent choice as they contain time series sequences that have the longest lengths.

1. CinC_ECG_torso
    - Length of the time series:1639
    - Size of test set:1380
    - Size of training set:40
    - Number of classes:4
2. InLineSkate
    - Length of the time series:1882
    - Size of test set:550
    - Size of training set:100
    - Number of classes:7

Chapter 3

Background

3.1 Dynamic Time Warping Algorithm

The Dynamic Time Warping algorithm is a similarity metric that is used cluster similar time series sequences varying in length, scale and speed. The problem formulation [6] of the algorithm is stated as follows: Given two time series X, and Y, of lengths $|X|$ and $|Y|$,

$$X = x_1, x_2 ... x_{|X|} \tag{3.1}$$
$$Y = y_1, y_2 ... y_{|Y|} \tag{3.2}$$

construct a warping path W

$$W = w_1, w_2 ... w_k \text{ where max } (|X|, |Y|) \leq k \leq |X| + |Y|$$

where

- k denotes the length of the warping path
- the mth element of the warping path is $w_m = (i, j) \in [1 : N] \times [1 : M]$ for $l \in [1 : k]$ where i is an index from the time series X and j is an index from the time series Y.

To properly understand the above formulation some key definitions must be stated:

1. Warping path [16]: An (N,M)-warping path (or simply referred to as warping path if N and M are clear from the context) is a sequence $w = (w_1, ..., w_k)$ with $w_l = (i, j) \in [1 : N] \times [1 : M]$ for $l \in [1 : k]$ that satisfies the following three conditions.
   (a) Boundary condition [16, 7, 10, 13]: $w_1 = (1,1)$ and $w_k = (N, M)$.

The boundary condition enforces that the first elements of the series X and Y as well as the last elements of the series X and Y have to be aligned with each other. This prevents the alignment from missing any points.

(b) Monotonicity condition [16, 7, 13]:

$\forall i \in [1:k-1], [|w_{i+1} - w_i| = 1]$. In other words, $w_k = (i,j), w_{k+1} = (i',j'), i \leq i' \leq i+1, j \leq j' \leq j+1$

The condition requires that the path will not turn back on itself. Both the i and j indexes either stay the same or increase but they will never decrease.

(c) Step-size condition [16, 7, 13]: $w_{l+1} - w_l \in \{(1,0),(0,1),(1,1)\}$, $\forall l \in$ [1:k-1].

The step size condition expresses a kind of continuity condition: no element in X and Y can be omitted and there are no replications in the alignment

Intuitively speaking, the (N,M) warping path $w = (w_1,...,w_k)$ defines an alignment between two sequences $X = (x_1,x_2,...,x_N)$ and $Y = (y_1,y_2,...,y_M)$ by assigning each element of X to a unique element in Y.

2. Optimum Warping Path [10, 16]:

The optimal warp path corresponds to the minimum-distance warp path, where the distance of a warp path W is given as

$$Dist(W) = \sum_{m=1}^{K} dist(X,Y)_{|(w_m)}$$

where $w_m = (i,j) \in [1:N] \times [1:M]$

The $dist(X,Y)_{|(w_m)}$ represents the distance computed using an appropriate cost function between the value x_i of sequence X and the value y_j of sequence Y.

$$dist(X,Y)_{|(w_m)} = dist(x_{x_i}, y_{y_j})$$

The goal of the DTW algorithm is to compute the distance of the optimal warping path for given two time series sequences. Instead of attempting to solve the entire problem all at once, the algorithm utilises the technique of dynamic programming to find an optimum alignment between two sequences through the computation of local distances between the points in the temporal sequences. The algorithm proceeds by iteratively filling in values for each cell (i,j) in the $|X|$ by $|Y|$ cost matrix D. The value of the cell (i,j) is given by

$D(x_i, y_j)$ which uses a cost metric to compute the distance of optimum warp path from (1,1) to (i,j) :

$$D(i,j) = Dist(i,j) + min(D(i-1,j), D(i-1,j-1), D(i,j-1))$$

The figure below illustrates the working of the DTW algorithm.

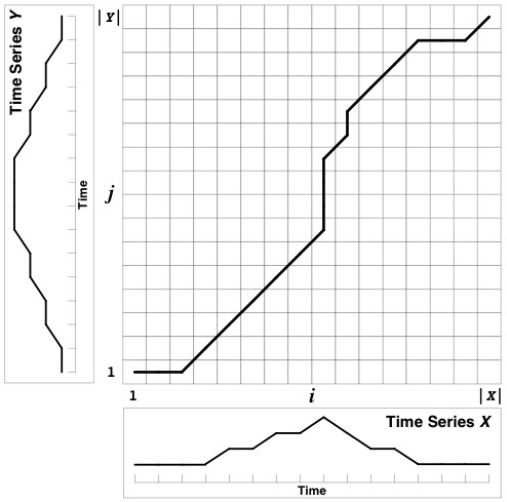


Figure 3.1: The DTW cost matrix with the minimum-distance warp path traced through it.

Generally as a cost metric, the euclidean metric is mostly widely used. For this project, I will hence consider the basic DTW algorithm equipped with the euclidean metric as my baseline model. An outline of the base line DTW algorithm is given below:

Algorithm 1 Value-Based DTW

```
1: procedure VALUE-BASED(seq1, seq2)                    ▷ two raw sequences
2:     DTW= zeros(length(seq1)+1,length(seq2)+1)
3:     for i=1: to length(seq1) do                      ▷ Initialise the DTW cost matrix
4:         DTW(i,0) = ∞
5:     end for
6:     for i=1 to length(seq2) do
7:         DTW(0,i) = ∞
8:     end for
9:     for i=2 to length(seq1) do
10:        for j=2 to length(seq2) do                   ▷ cost(a,b)≡euclid(a,b)
11:            DTW(i,j) = cost(seq1(i),seq2(j)) + min{ DTW(i-1,j)+DTW(i,j-
   1)+DTW(i-1,j-1)}
12:        end for
13:    end for
14:    return result = DTW(n,m) / nm                    ▷ n=length(seq1), m=length(seq2)
15: end procedure
```

The computational complexity of the DTW algorithm is $O(n^2)$ where n denotes the length of the sequences that are being compared. Thus for time series domains having high dimensions i.e.long lengths, the time and computational costs incurred by the algorithm are quite high. To address this issue, two well-known global window constraints are employed: the Sakoe-Chiba band [7] and the Itakura parallelogram [13] . Figure 3.2 gives an illustration of the use of both constraints:

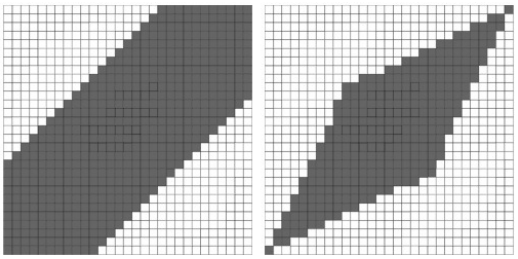


Figure 3.2: Two constraints: Sakoe-Chuba Band (left) and an Itakura Parallelogram (right), both have a width of 5.

The shaded areas in the above figure represent the cells of the cost matrix that are filled in by the DTW algorithm for each constraint. The width of each shaded area/window, is specified by a window parameter. When these constraints are used, the DTW algorithm finds the optimal warp path only through the constraint window. However, the globally optimal warp path will not be found if it is not entirely inside the window[6]. The window parameter

therefore determines the accuracy of the DTW algorithm. Decreasing the width decreases the accuracy. If the width is 0 and the series are of equal length then the DTW generalises to the euclidean distance which is a notoriously inaccurate similarity measure for time series data[17]. To constraint the time complexity to a minimum, the vast majority of the data mining researchers use a Sakoe-Chiba Band with a 10% width. This reduces the time complexity from O(n^2) to O (wn) where w is the size of the window parameter.

3.2 Single Value Decomposition

The formal definition of SVD as stated in[18, 12] is given as follows:

Any $M \times N$ real matrix X can be factorized of the form $X = UDV^T$ where

- U is a $M \times M$ orthogonal matrix. The columns of U are the eigenvectors of XX^T
- D is a $M \times N$ rectangular matrix with :

$$D_{ij} = \begin{cases} \sigma^2, & \text{if } j = i \\ 0, & \text{otherwise} \end{cases}$$

  Here D is a diagonal matrix where the diagonal entities are called singular values and are ranked from greatest to least. The number of non-zero singular values determine the rank of the original matrix.

- V is a $N \times N$ orthogonal matrix. The columns of V are the eigenvectors of X^TX. Since X^TX is the covariance matrix, these eigenvectors span the linear subspace that maximises the variance of the data.

Dimensionality reduction is achieved by discarding the eigenvectors in V i.e the columns that are associated with the smallest or zero singular values. Thus, each sample can now be represented by fewer dimensions. For data matrices where M $\ll$ N, there exists at most M-1 singular values. In such situations, considering only the non zero columns in the diagonal matrix D and the associated columns in V achieves dimensionality reduction without any loss of function. The data matrix X can be factorized as $X = U \times D' \times V'^T$ where D is now an M by K diagonal matrix containing only positive values on its diagonal and V′ is the corresponding N by K matrix. The matrix V'^TV captures the full covariance of the data.

Chapter 4

Optimising DTW

The Dynamic Time Warping(DTW) algorithm is the one of the oldest algorithm that is used to compare and cluster sequences varying in time, scale and speed. Given two temporal sequences, the algorithm utilises the technique of dynamic programming to compute the cost of the optimum alignment path between them. The computed cost gives an indication of the degree of similarity. The smaller the cost, the more similar the sequences are. Intuitively speaking, DTW can be seen as a clustering algorithm that clusters patterns that share roughly the same shape. As I have stated in the previous chapter, the time and computational complexity of this algorithm is $O(n^2)$ where n denotes the length of the sequences that are being compared. Thus for time series domains having high dimensions i.e long sequences, DTW becomes a very unattractive choice as it suffers from high computational and time costs.

To address the issue of the curse of dimensionality, DTW algorithms employ a window constraint to reduce the search space. The most commonly used are Sakoe-Chuba Band[7] and the Itakura window constraint [13]. Figure[3.2] gives an illustration of the nature of these window constraints. Such constraints determine the allowable shapes that a warping path can take by restricting the DTW to construct optimal warping paths only through a restricted number of cells around the diagonal region of the cost matrix. As the dimensionality(length) of the sequences increases, the size of the window is adjusted accordingly. Rigid window constraints impose a more rigid alignment that prevent an overly temporal skew between two sequences, by keeping frames of one sequence from getting too far from the other. The vast majority of the data mining researchers use a Sakoe-Chiba Band with a 10% width for the global constraint [19] to constraint the time complexity of DTW to a minimum. For finding motifs in datasets comprising of speech utterances, the use of rigid window constraints is highly undesirable. Utterances corresponding to the same lexical identity may suffer from large variations in speed, scale and time

due to the word(s) being spoken in difference contexts, by different speakers, in different environments etc. Thus it is necessary to explore alternative techniques to window constraints that can reduce the time complexity of DTW without degrading its accuracy.

Before investigating methods to improve the DTW algorithm itself, it is highly necessary to first understand the nature of the data sequences that the DTW is presented with. Achieving a thorough understanding of the data can result in the extraction of a smaller set of relevant features that can be used to achieve better discrimination between different classes/motifs. In this chapter, I investigate domain-dependent preprocessing techniques to improve the performance of the baseline DTW algorithms in clustering patterns that have long lengths.

There are presently two groups of preprocessing techniques commonly used to address this issue:

- Feature Selection
- Feature Extraction

Feature selection techniques involve selecting only a subset of attributes from the original data. With respect to the time series data, the technique is analogous to sub-sampling the sequence. To remove redundant features, one of the most popular feature selection approach is the exploratory data analysis(EDA). EDA is an approach to data analysis that postpones the usual assumptions about what kind of model the data follows with the more direct approach of allowing the data itself to reveal its underlying structure and models. The particular techniques employed in EDA are often quite simple, consisting of various techniques of:

1. Plotting the raw data such as data traces, histograms, histograms, probability plots, lag plots, block plots, and Youden plots.
2. Plotting simple statistics such as mean plots, standard deviation plots, box plots, and main effects plots of the raw data.
3. Positioning such plots so as to maximise our natural pattern-recognition abilities, such as using multiple plots per page.

Apart from removing redundant features, we can also construct more useful features from the existing ones. Feature extraction processes are concerned with the range of techniques that apply an appropriate functional mapping to the original attributes to extract new features. The intuition behind feature extraction is that the data vectors $\{x_n\}$ typically lie close to a non- linear manifold whose intrinsic dimensionality is smaller than that of the input space as a result of strong correlations between the input features. Hence

by using appropriate functional mapping, we obtain a smaller set of features that capture the intrinsic correlation between the input features. By doing so, we move from working in high dimensional spaces to working in low dimensional spaces. The choice of appropriate functional mapping can also improve the clustering of data. For example, lets consider figure 4.1: The left- hand plot represents the locations of two dimensional data points in the original input space. The colours red and blue denote the classes to which the data points belong to. To cluster the data with respect to their classes, it will be ideal if we can partition the input space into disjoint regions where intra class variation is small and inter class separation in large. For this example, this is achieved by mapping the points to a feature space spanned by two gaussian basis functions(shown on the right). Now, we can partition the feature space into two disjoint regions, one of each cluster.

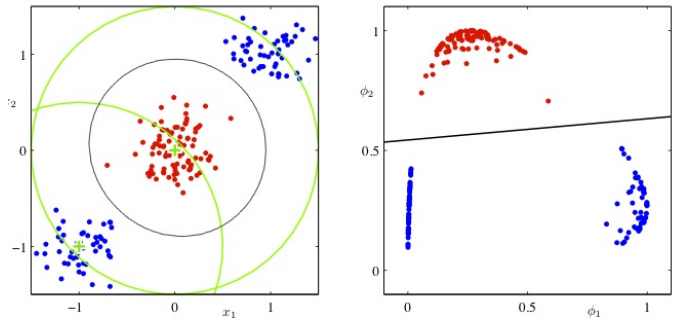


Figure 4.1: The figure on the right corresponds to location of the data points in the feature space spanned by gaussian basis functions $\phi_1(x)$ and $\phi_2(x)$

In the following sections, I investigate whether the application of exploratory data analysis and the construction of features that integrate metadata mainly knowledge of the domain can aid the baseline DTW algorithm is achieving low run times and good accuracy on long time series sequences. The primary data set that I will be using is the TIDIGITS corpus as the time series sequences of this dataset have lengths that are in the order of 10^4.

4.1 Feature Selection

The computational and time complexity associated with the DTW algorithm is governed by the length of the time series. Removing redundant features can result in a great reduction in the time complexity without any negative impact on the accuracy. To get an idea about the underlying structure of the data, I studied the plots of the time series sequences along with listening to

the individual samples. Figure 4.2 shows the plot of raw signal corresponding to the word '8' by a speaker from the *boy* category.

From the visual and auditory analysis, I have made the following observations:

- Long durations of silence occupy the beginning and end of each utterance. The durations of the interesting regions that actually contain information about the spoken digit is quite small is comparison to the durations of silence regions. Removing these silence segments allow reduction in the dimensionality of the time series sequences with minimal loss of information.
- The recordings are highly distorted when played in *matlab*. The distorted signals fail to provide any type of auditory clue about category of the speaker i.e whether the speaker belongs to { boy,girl, men,women} the signal has to be played multiple time to correctly identify the spoken word.

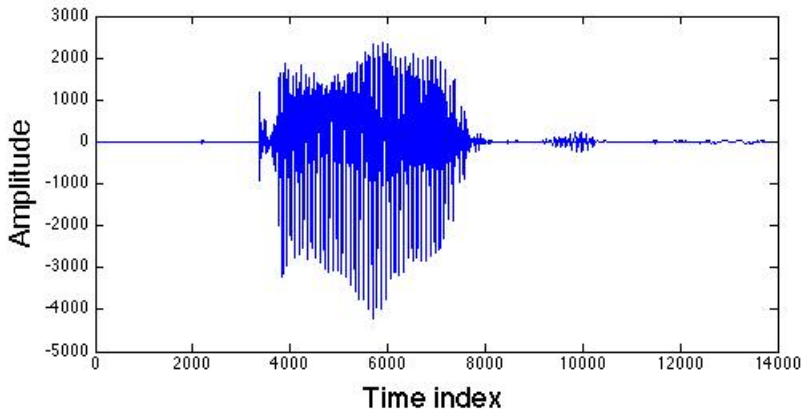


Figure 4.2: 'Raw 'signal

4.1.1 Signal Filter

To remove the redundant attributes from the time series utterances, I have constructed the following algorithm: 'SignalFilter' that performs feature selection by removing segments corresponding to durations of silence. An outline of the algorithm is as follows:

The algorithm employes an adaptive threshold to select and remove redundant attributes. The value of the threshold is dependent on the maximum amplitude of the sequence. The value set is comparatively higher for utterances of speakers having a loud and deep voice and lower for utterances for speakers having gentle and low voice.

Algorithm 2 SignalFilter

1: **procedure** SIGNALFILTER(*signal*) ▷ raw signal
2: *threshold* = 0
3: maxAmplitude= max(rawSignal)
4: Adapt the threshold based on the value taken by the maximum amplitude
5: output← removeSilence(rawSignal,threshold)
6: **return** output
7: **end procedure**

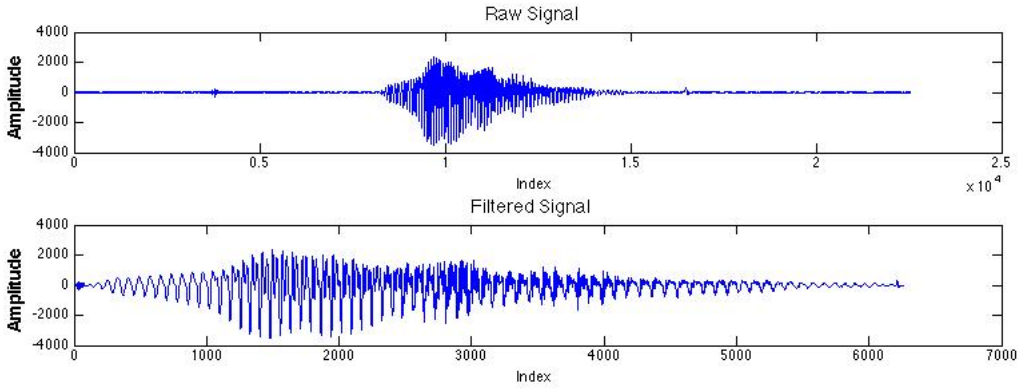


Figure 4.3: shows the raw acoustic signal corresponding to the utterances of the digit '8' alongside with the filtered signal at the bottom.

From figure 4.3, it can be observed that the filter preserves the interesting patterns associated with the utterance while succeeding in reducing the dimensionality of the data. To investigate the effect of introducing this prior feature selection step on the performance of the baseline DTW algorithm, I conducted the following experiment:

SETUP:

- Dataset used: TIDIGITS Test-set size : 976 samples

category	sample size
boy	162
girl	162
men	326
women	326

Training data set size: 1467

category	sample size
boy	225
girl	234
men	495
women	513

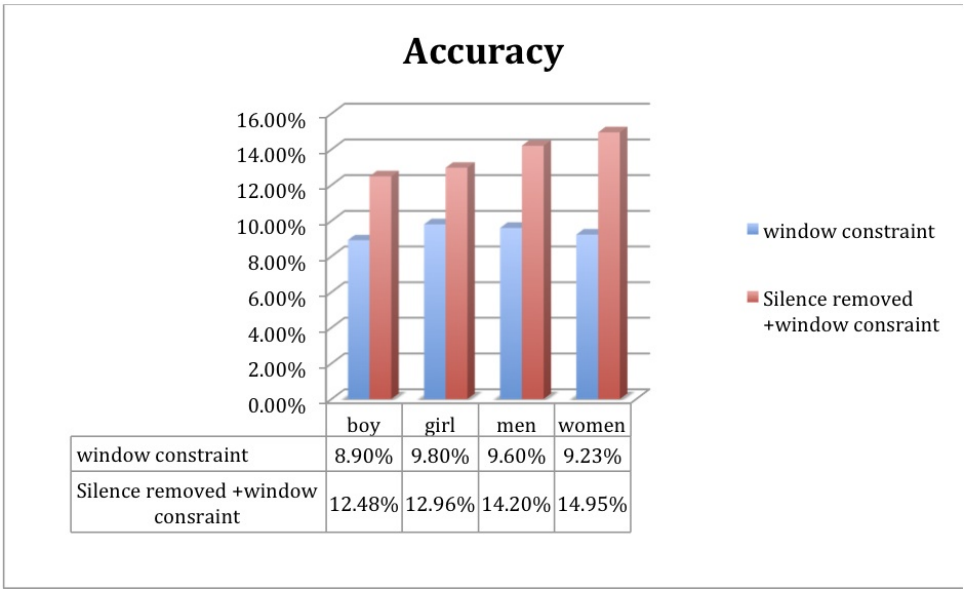
- An outline of DTW used algorithm used for this experiment is given below.

Algorithm 3 Value-Based DTW

procedure VALUE-BASED($seq1, seq2$) ▷ two raw sequences
  DTW= zeros(length(seq1)+1,length(seq2)+1)
  w = max($\lceil 0.1 * max(n.m) \rceil$, abs(n-m)) ▷ Window constraint
  for i=1: to length(seq1) **do** ▷ Initialise the DTW cost matrix
    DTW(i,0) = ∞
  end for
  for i=1 to length(seq2) **do**
    DTW(0,i) = ∞
  end for
  for i=2 to length(seq1) **do**
    for j=max(2, i-w) to min(length(seq2), i+w) **do** ▷ cost(a,b)≡euclid(a,b)
      DTW(i,j) = cost(seq1(i),seq2(j)) + min{ DTW(i-1,j)+DTW(i,j-1)+DTW(i-1,j-1)}
    end for
  end for
  return result = $\frac{\text{DTW(n,m)}}{nm}$ ▷ n=length(seq1), m=length(seq2)
end procedure

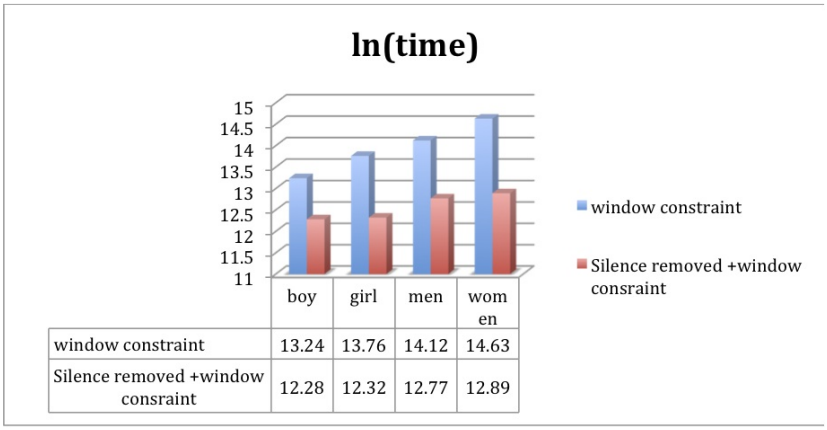
Note : The main objective of my research is to improve the accuracy of the DTW algorithm while reducing the time and computational cost to a **minimum**. Even after applying the feature selection process, from initial experiments on few samples, I have found that the computational cost incurred by the algorithm is still very high. Hence to minimise run time, I employed the Sakoe-Chuba band that has an adaptive window size of : w = max($\lceil 0.1 * max(n.m) \rceil$, abs(n-m)). The lower bound of the window size is set to 10% of the size of the longest sequence which is the standard size that vast majority of the data mining researchers [19] use to keep the time complexity of DTW to a minimum.

- RESULTS:



	boy	girl	men	women
window constraint	8.90%	9.80%	9.60%	9.23%
Silence removed +window consraint	12.48%	12.96%	14.20%	14.95%

Figure 4.4



	boy	girl	men	wom en
window constraint	13.24	13.76	14.12	14.63
Silence removed +window consraint	12.28	12.32	12.77	12.89

Figure 4.5

Observations:

- The DTW algorithm achieves very poor accuracy. The poor accuracy is a result of one or a combination of the following three factors:
  - the use of raw values a features: the numerical value associated with each time point is not a complete picture of the data point in relation to the rest of the sequence.
  - the use of the rigid window constraint- the low accuracy may be attributed to the optimum warping path between similar patterns lying outside boundaries of the Sakoe-Chuba bands[5, 4].
  - not integrating knowledge of the domain: The data set is comprised of speech utterances. It is a widely known fact that for speech data,

the MFCC feature vectors capture the information of phones that make up a word. Since different lexical identities are composed of different phones, these use of MFCC vectors can increase the inter class variance of the samples. (details of MFCC to follow)[4, 20, 5, 21, 22].

- Removing 'silence' segments improve **both** the accuracy and the run-time of the algorithm. While the decrease in run-time is quite obvious, the increase in accuracy is not. All utterances contain durations of silence. Taking the silence regions into account decreases the inter class variance and in turn increase intra class variance as they bring in an unwanted notion of similarity in dissimilar patterns.

4.1.2 Downsampling

From conducting further exploratory data analysis, I have observed that if I down-sample the utterances by $\frac{1}{2}$ which in other words corresponds to decreasing the sampling frequency by half, the global shape is still preserved even though some local information is lost as shown in figure 4.6. This results in further reduction in the dimensionality of the sequences i.e the length of the sequences.

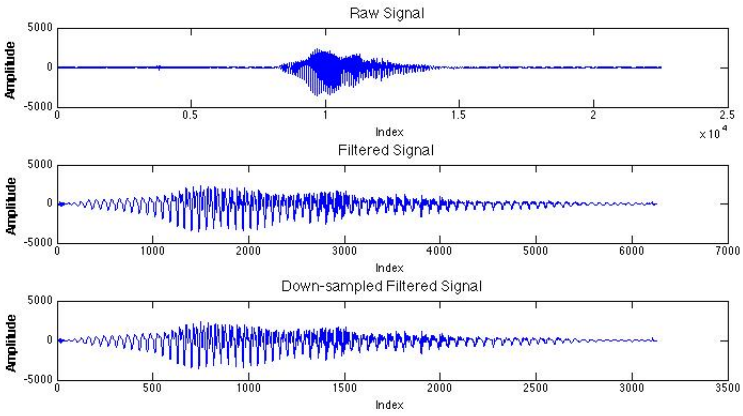


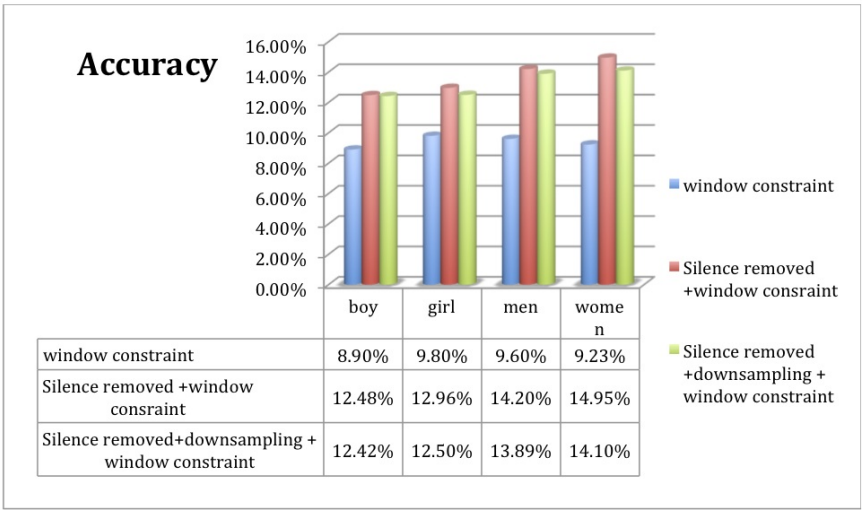
Figure 4.6: shows the raw acoustic signal of the digit '8' (top figure), the silence removed version of the signal(middle) and the silence removed and down sampled version of the acoustic signal (bottom)

To investigate the effect of performing further dimensionality reduction on the time series sequences through down sampling on the accuracy of the DTW, I performed the following experiment:

Dataset: The TIDIGITS training and set used in 4.1

Algorithm : baseline DTW augmented with an adaptive window constraint(4.1)
Approaches being compared: Removing silence utterances VS Removing silence utterance + downsampling VS baseline

Results:

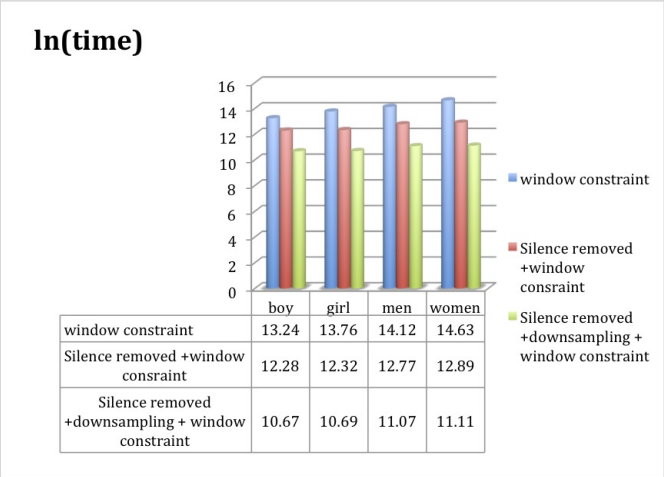


	boy	girl	men	women
window constraint	8.90%	9.80%	9.60%	9.23%
Silence removed +window consraint	12.48%	12.96%	14.20%	14.95%
Silence removed+downsampling + window constraint	12.42%	12.50%	13.89%	14.10%

Figure 4.7

Observations:

Performing the two stage feature selection step that involves removing 'silence' utterances followed by downsampling allow DTW to still construct more optimal paths that are aligned along the main diagonal for similar patterns than using the entire 'raw' time series sequences. This procedure results in a 4% increase in accuracy on average. Furthermore, the loss of local information through downsampling the non-silence regions leads to a minimal decrease in the accuracy of the algorithm in comparison to using the entire non silence regions for pattern matching. This supports the claim that the classes are differentiated mainly by their global shape.



	boy	girl	men	women
window constraint	13.24	13.76	14.12	14.63
Silence removed +window consraint	12.28	12.32	12.77	12.89
Silence removed +downsampling + window constraint	10.67	10.69	11.07	11.11

Figure 4.8: Integrating downsampling decreases the ln(run time)

Augmenting the down sampling procedure to the feature selection process leads to a further decrease of 1.5 in the ln(run time) of the DTW algorithm in comparison to just removing silence regions as a preprocessing step. This is expected since in the first stage of the preprocessing phase, redundant features are dropped which reduces the dimensionality(i.e length) of the sequences. The length of the sequences is reduced even further through downsampling the the output sequences of stage 1. Since the computational cost of DTW is directly dependent on the length of the sequences, thus decrements in dimensionality leads to a decrease in the computational cost.

4.2 Domain Dependent Feature extraction

From the analysis conducted so far, it can be concluded that heuristically selecting only significant attributes from the time series sequences does **improve** both the accuracy and the speed of the DTW. However, from the observation of the experimental results gathered so far, it is quite clear that the accuracy of the algorithm is very low. So far, I have investigated the effect on using 'raw' values of the time series sequences on the performance of the DTW. In this section, I investigate the contribution of using the window constraint and the contribution of using domain dependent features on the performance of the DTW algorithm. There are two motivations behind conducting this analysis:

- The primary motivation is to improve the speed of the DTW algorithm without degrading the accuracy. Choosing an appropriate functional mapping, can help map the data to a lower dimensional feature space that can capture the intrinsic qualities of the data. Thus constructing appropriate functional mappings can achieve both dimensionality reduction

and higher accuracy.

- The other motivation is to investigate to what degree is the low accuracy error credited to not using domain dependent features and to using a rigid window constraint has on the low accuracy(The features that we have considered so far are the raw values indexed by time). Understanding the underlying factors that that influence the performance of the algorithm can provide insights on what aspect of the algorithm needs to be changed to gain better performance.

4.2.1 Mel Cepstrum Cepstral Coefficients

The primary dataset that I am working with is the TIGITS corpus which is composed of speech utterances. For speech, the most commonly used features are the MFCC features-mel cepstrum cepstral coefficients. This feature representation is based on the idea of the cepstrum. For human speech, a speech waveform is created when a glottal source waveform of a particular frequency is passed through the vocal tract which because of its shape has a particular filtering characteristic. The exact position of the vocal tract is in fact the key attribute in providing useful information about phones(units of sounds). Cepstrum provides a useful way to separate the information of the vocal tract from the glottal source.

An outline of the MFCC feature extraction is given below:

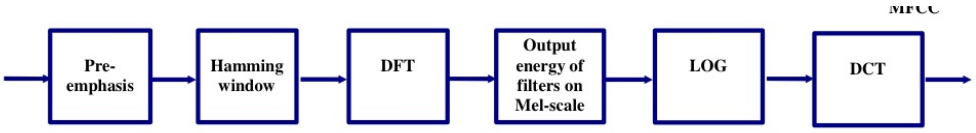


Figure 4.9: MFCC feature extraction

i Pre-emphasis: boosts the energy of the signal at high frequencies to improve phone detection

ii Windowing: partitions the time series sequence into frames using a hamming window

iii DFT: extracts spectral information at different frequency bands

iv Mel scale : transforming to mel scale improves recognition performance by allowing the model to take into account the property of human hearing

v Log : makes the feature less sensitive to variations in input such as power variations on the speakers mouth.

vi Cepstrum : separate the information of the vocal tract from the glottal source. The first 12 cepstral values from spectrum of the log of the spectrum are used

The windowing process results in dimensionality reduction i.e the length of the time-series sequences is reduced. Each sequence is segmented into frames of length 20 to 30 ms. These frames are then mapped using the procedure above to MFCC feature vectors. Since the result sequence of vectors is much smaller than the length of the original sequence resulting in the size of the DTW cost matrix is much smaller than before which in turn lowers the time and computation cost incurred by the algorithm. The time complexity of DTW is now reduced from O(n^2) to O (u^2) where $u = \frac{n}{l}$ (l denotes the length of the frame.

The experiments conducted in section 4.1.1 and 4.1.2, have shown that the DTW algorithm performs very poorly in terms of accuracy on the TIDIGITS test data when it employs the narrow Sakoe-Chuba band to minimise the time complexity. The reason for this low accuracy was credited to the influence of one or a combination of the following factors : using a narrow window constraint, using raw attribute values and not using domain-dependent features. Having already investigated the influence of using raw values(4.1), for this part of the project, I constructed the following 3 models in order to investigate the relative isolated impacts of the other two factors. The models were tested on the TIDIGITS dataset, thus allowing the results to be compared with the results of the previous experiments so far :

i Model 1 employs the MFCC feature extraction mapping as a preprocessing step. The extracted features are used by the valued-based DTW algorithm augmented with the adaptive Sakoe-Chuba band constraint described in 4.1.1 to cluster similar patterns. The performance of this model can be compared to the performance of the base line model to measure the relative impact on the accuracy and run-time of the DTW of replacing each numerical value associated with each time index with a domain dependent feature vector.

ii Model 2 employs a two stage preprocessing step. The feature selection procedure discussed in 4.1.1 is first applied to remove redundant features followed by MFCC feature extraction that achieves further dimensionality reduction. In this model dimensionality reduction occurs at both stages of the pre-processing step. The downsampling method discussed in 4.1.2 was deemed not necessary as a feature selection step since the feature extraction phase allows further reduction in dimensionality without any loss of information. The sequence of vectors are then used by the DTW algorithm to find optimal warping along the main diagonal of the DTW

cost matrix.

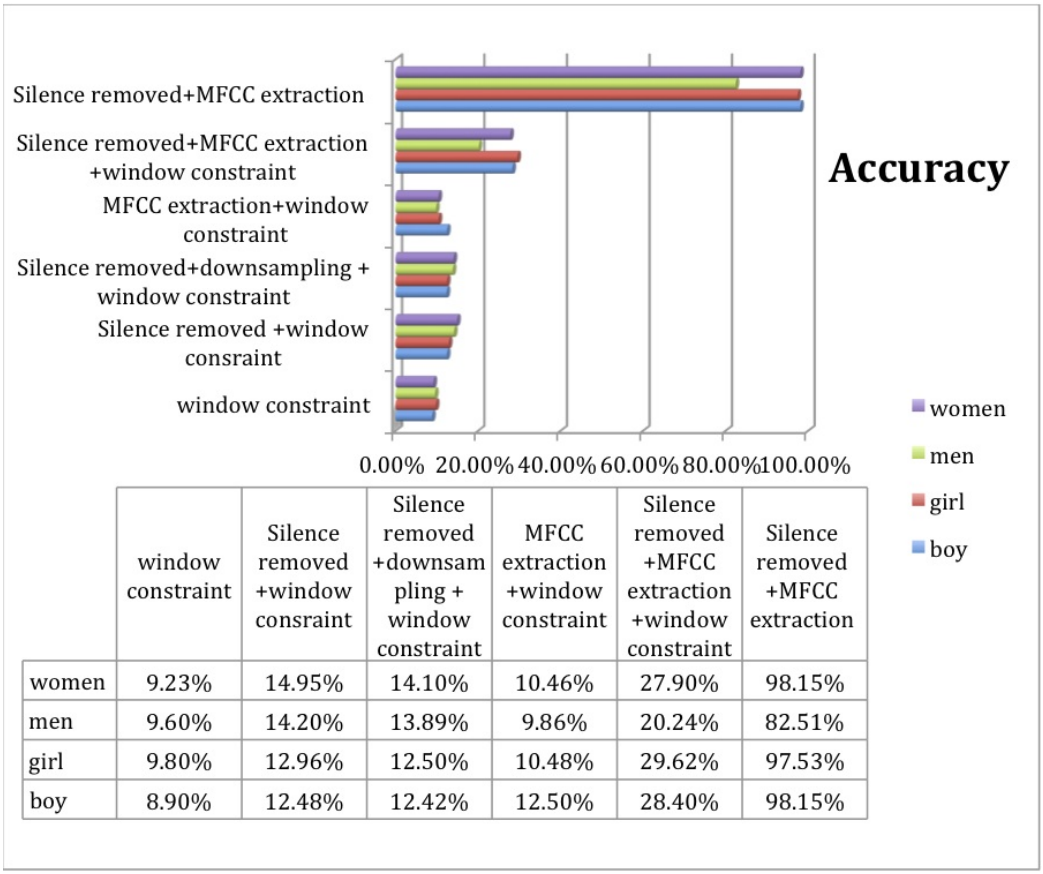
By comparing the results of this model with the results of model 1 and the model discussed in 4.1, the optimum preprocessing strategy can be determined.

iii Model 3 is identical to the version 2 with the exception that this version does not employ the window constraint. The main purpose of using this model is to check the relative impact of using window constraints. The difference in the performance between using model 2 and model 3 can be used to check to what degree is employing such a rigid window constraint affect the accuracy and run-time of the algorithm.

Experimental setup:

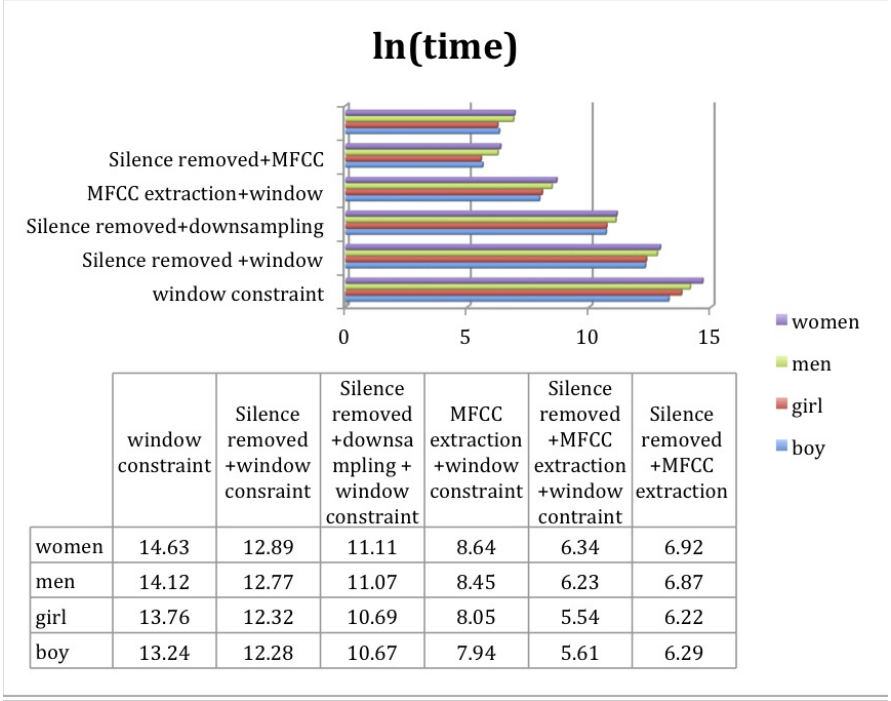
Dataset : The TIDIGITS training and test set (Chapter 2, 4.1.1)

RESULTS:



	window constraint	Silence removed +window consraint	Silence removed +downsampling + window constraint	MFCC extraction +window constraint	Silence removed +MFCC extraction +window constraint	Silence removed +MFCC extraction
women	9.23%	14.95%	14.10%	10.46%	27.90%	98.15%
men	9.60%	14.20%	13.89%	9.86%	20.24%	82.51%
girl	9.80%	12.96%	12.50%	10.48%	29.62%	97.53%
boy	8.90%	12.48%	12.42%	12.50%	28.40%	98.15%

Figure 4.10: Accuracy achieved by all models



	window constraint	Silence removed +window consraint	Silence removed +downsampling + window constraint	MFCC extraction +window constraint	Silence removed +MFCC extraction +window contraint	Silence removed +MFCC extraction
women	14.63	12.89	11.11	8.64	6.34	6.92
men	14.12	12.77	11.07	8.45	6.23	6.87
girl	13.76	12.32	10.69	8.05	5.54	6.22
boy	13.24	12.28	10.67	7.94	5.61	6.29

Figure 4.11: The ln(run time) incurred by all models

Observations:

- Making DTW domain-dependent has little significance on the accuracy when constraints are employed to minimise the time complexity.

  -In comparison to the base-line model, the accuracy of the window-constrained DTW only increases by 1.3% on average across all 4 categories.

  -The run-time on the other hand has improved considerably. In comparison to the baseline model, the average ln(run time) has decreased by 5.7. This is obvious since the acoustic signal is mapped to smaller sequence of frames.

- Removing redundant features have greater relative impact on the accuracy than employing domain dependent MFCC features. This is evident if we compare the results of using model 1 with the model discussed in 4.1.1. The presence of silence regions force the optimal warping paths between patterns of the same lexical identity to lie outside the regions in the cost matrix bounded by the Sakoe-Chuba band. This forces the DTW to find an **approximate** optimal path with in the allowed regions which in turn leads to an increase in the error rate.

- Combining attribute/feature selection with MFCC feature extraction as a preprocessing step achieves greater improvement in accuracy and speed

than using either of these approaches alone. In comparison to just using MFCC feature extraction as a preprocessing step, the accuracy has been boosted up by 15.17% on average while the $ln(time)$ have been reduced by 2.36. Similarly, in comparison to just using feature selection as a preprocessing step, the accuracy has increased by 12.9% on average while the $ln(time)$ have been reduced by 6.5.

- From the above observation alone, we can deduce two facts for this dataset, one of which is not that obvious: removing redundant features have a greater relative impact on improving the performance of a constrained DTW than employing domain dependent features and the second fact which is quite obvious is the strong dependency between the algorithm's accuracy and the size of the window constraint. It can be concluded that the use of the rigid window constraint is the main contributing factor for achieving small accuracy rates.

  Since the main goal is to improve **both** the accuracy and speed of DTW in handling sequences of high-dimensionality i.e long lengths, removal of silence segments provides an ideal mechanism to improve **both** the time complexity and the accuracy of the algorithm. Dropping the window constraint although will lead to considerable increase in the accuracy by will result in longer run-times. This is not ideal when working on large high dimensional datasets.

- Model 3 achieves almost near perfect accuracy. Dropping the window constraint improves the accuracy by 67.15% on average over model 2. In other words 67.15% of the time on average, the optimum warping path lie outside the regions bounded by the Sakoe-Chuba band constraints, This confirms that patterns belonging to the same lexical identity can have an overly temporal askew between them as result of the different contexts in which the words are spoken and/or as a result of different speakers speaking the same word.

- The run time of model 3 is lower than the run time of all models with the exception of model 2. Model 3 therefore provides the best balance in the fulfilling the two conflicting objectives of high accuracy and high speed that any of the other investigated models.

Conclusion

To summarise, to improve the performance of the DTW algorithm:

- In 4.1, I have investigated ways to remove redundant features by conducting exploratory data analysis.

- In 4.2, I introduced domain dependent feature extraction methods and

investigated the influence of the use of such features on the performance of DTW

- I have investigated the investigated the relative isolated impacts of using raw values, employing a rigid window constraint and using domain dependent features on the performance of the DTW algorithm
- I have observed that although the DTW algorithm is domain independent, augmenting the DTW with a domain dependent preprocessing technique can greatly improve its performance in terms of **both** accuracy and speed with out the need of any global constraints. For the TIDIGITS dataset, employing the preprocessing steps of 'silence' removal followed by MFCC feature extraction allows DTW to improve not only its accuracy but also speed even without the use of global window constraints.

Chapter 5

Improving DTW

So far, we have investigated methodologies that improve both the speed and accuracy of the DTW algorithm by integrating meta data (i.e knowledge of the domain) in the preprocessing stage. The problem with such methodologies is that the same algorithm cannot be extended across multiple domains: the MFCC feature extraction discussed in the previous chapter is only applicable to time series sequences that correspond to speech utterances. To develop a framework that can be extended across multiple domains, it is necessary to use features that are entirely data driven. In the first half of this chapter, I investigate in detail a recently proposed data driven feature extraction methodology [9] that is aimed to improve the accuracy of the DTW across multiple domains. In the second half of this chapter, I investigate alternative measures to using window constraints that can improve the performance of the algorithm in terms of **both** time and accuracy across different time series domains.

5.1 Domain-independent feature extraction

Ideally, we require features that reflect information about the structure of the data. This allows the DTW to built a complete picture of the data point in relation to the rest of the sequence and hence achieve optimal alignments that are close to the diagonal for similar sequences. The fundamental problem of baseline (value-based) DTW is that the 'raw' numerical value associated with each point in the time series sequence is not a complete picture of the data point in relation to the rest of the sequence. The context such as the position of the points in relation to their neighbours is ignored. To fix this issue, an alternative form of DTW known as *derivative* DTW[23] is proposed but it too fails to achieve better performance across all domains as it ignores to take into account the common sub-patterns between two sequences (mainly global

trends).

To address these drawbacks, recent works have been conducted to extract good feature extraction methods that are entirely data-driven. One particular approach that has been seen to achieve good accuracies across multiple domains is the method proposed Xie and Wiltgen in their paper [9]. The authors highlight a domain-independent feature extraction process where each point in the time series sequence is replaced by a 4 dimensional vector. This 4-d vector attempts to provide a complete picture of a point in the relation to the rest of the sequence. The first two features in this vector hold information of the local trends around a point while the last two features reflect information about the global trends. From experiments conducted on the UCR datasets, they have observed that embedding DTW with this feature extraction process yields greater accuracy across all datasets.

Definition of local feature given in [9] is as follows:

$$f_{\text{local}}(r_i) = (r_i - r_{i-1}, r_i - r_{i+1})$$

The first feature reflects the difference between the values of the current index and the previous index while the second feature corresponds to the difference between the value of the current index and the succeeding index.

The extraction of global features however, is constrained by two factors:

- the features must reflect information about global trends
- the features must be in the same scaling order as the local features. Being in the same scale allows them to be combined with local features.

In [9], the authors used the following method to extract global features from the time series sequence:

$$f_{\text{global}}(r_i) = (r_i - \sum_{k=1}^{i-1} \frac{r_k}{i-1}, r_i - \sum_{k=i+1}^{M} \frac{r_k}{M-i})$$

The first feature represents the deviation of the value at time i from the mean of the values of the sequence that has been seen so far while the second feature represents the deviation of the value at time i from the mean of the values that is yet to be seen. This formulation allows the detection of significant 'drops' or 'rises' in the series.

Note : The local and global features have no definition for the first and last points in a sequence and to keep the terminology clear, when I refer to the dimension of the time series, I mean the length of the time series.

One of the drawbacks of using this feature extraction methodology is the

absence of achieving dimensionality reduction. The length of the sequence of vectors is approximately the same as the dimensionality of the original sequence. The DTW algorithm combined with this feature extraction process therefore suffers from the curse of dimensionality as before. Since minimising the time complexity is a major priority for this analysis, the DTW algorithm is subjected to the adaptive Sakoe-Chuba band window constraint discussed in 4.1.1 to minimise the run time.

Xie and Witgen [9] have already shown that augmenting this feature extraction methodology to the DTW algorithm improves its accuracy across different domains. However, in their analysis due to the availability of sufficient computing power, they didn't use any window constraints when performing their experiments. For problem scenarios where availability of computing power is limited and the speed of the DTW is considered an equal priority along with the accuracy, it will be interesting to investigate whether this methodology can allow DTW to achieve better performance in terms of accuracy over the base line method.

To investigate the effect of introducing this prior feature selection step, I conducted the following 2 experiments:

Objective Compare the affect of using global and local features to using raw values on the performance of a **window constrained** DTW algorithm.

- Experiment 1

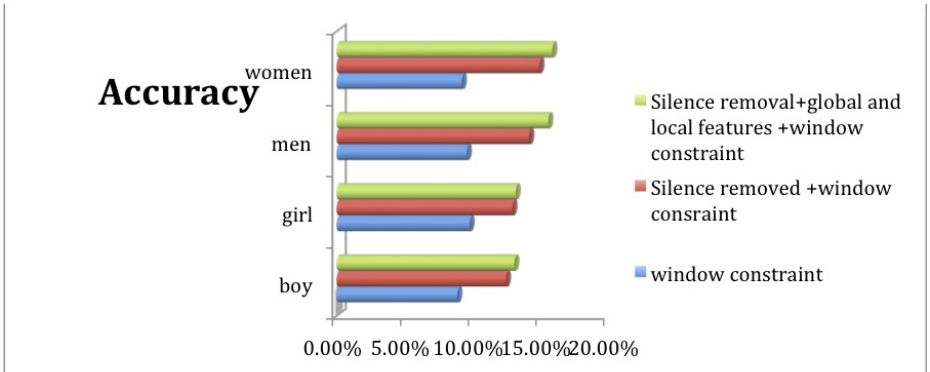
  Datasets: TIDIGITS dataset(4.1.1)

  Setup Prior to extracting these domain independent features , I performed the feature selection step of removing segments of utterances that correspond to silence. When conducting experiments using MFCC feature vectors, removing redundancy allows the optimal warping paths between similar patterns to lie closer to the main diagonal. Using this feature selection step also has the advantage of dimensionality reduction.

  By comparing the experimental results of this model with the baseline model and the model of 4.1, we can observe whether for this dataset, removing redundant features is more significant to using features that capture information about the global and local trends. Also we be able to investigate whether using such feature allows a constrained DTW algorithm to achieve higher accuracies than using raw values.

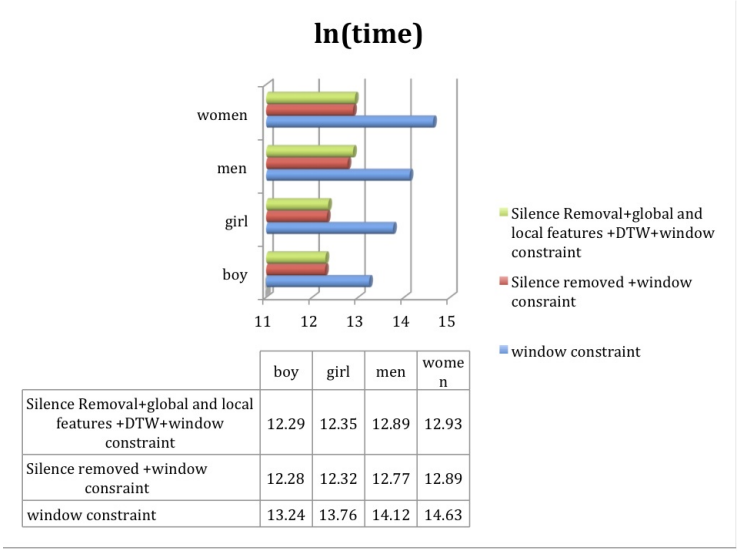
  RESULTS

  A summary of the results are given below:



	boy	girl	men	women
Silence removal+global and local features +window constraint	13.10%	13.20%	15.60%	15.90%
Silence removed +window consraint	12.48%	12.96%	14.20%	14.95%
window constraint	8.90%	9.80%	9.60%	9.23%

Figure 5.1



	boy	girl	men	women
Silence Removal+global and local features +DTW+window constraint	12.29	12.35	12.89	12.93
Silence removed +window consraint	12.28	12.32	12.77	12.89
window constraint	13.24	13.76	14.12	14.63

Figure 5.2: ln of the run time

Observations:

- Removing redundant features i.e removing regions of silence is a greater contributing factor in improving the performance of a window constrained DTW than applying a feature extraction process that integrates meta data(4.2.1) or that captures information about local and global trends.
- All 3 models achieve very low accuracies. As concluded from previous experiments, the low accuracy is primarily credited to the use of the rigid window constraint

– The computational cost incurred by the algorithm is higher than the version used for the model of 4.1.1 One possible explanation is that the cost of applying the euclidean metric on vectors >cost of applying the euclidean metric on points. Since the euclidean metric is applied mn times. The overall computational cost increases.

- Experiment 2

  Datasets: UCR datasets: InlineSkate and Cinc_ECG_Torso

The results are as follows:

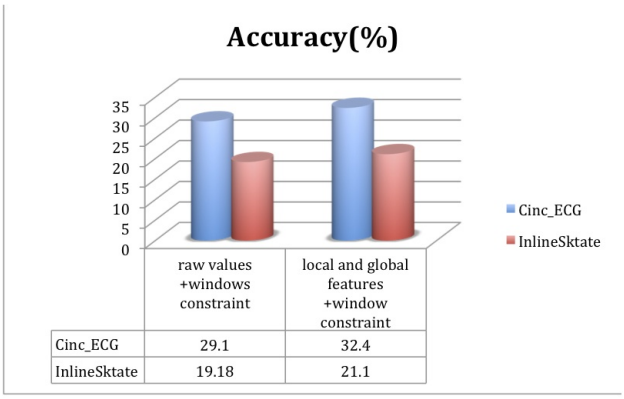


Figure 5.3: Using features that reflect information of trends improves the accuracy of the algorithm

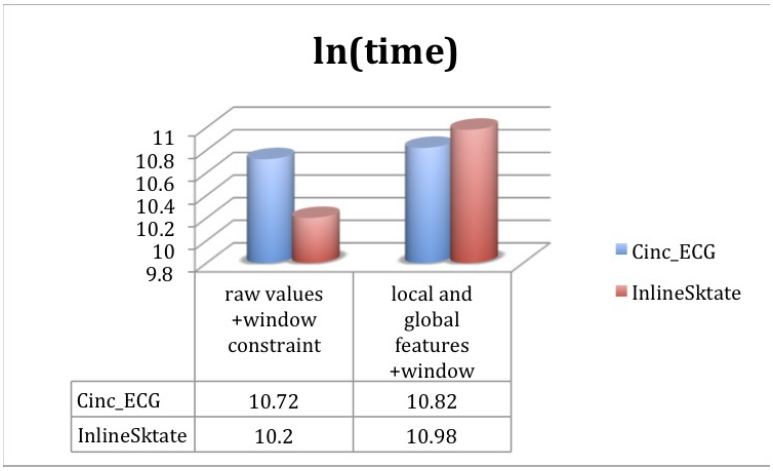


Figure 5.4: The computation time incurred

Observations

- The differences between performances of the two versions of DTW are consistent with the observations made in the previous experiment.

- In comparison to the base line method, there is only a minimum improvement in the accuracy of the window constrained DTW. For the Cinc_ECG_Torso time series dataset, using global and local features improves the accuracy by only 3.2% whereas for the InlineSkate dataset, the accuracy improves by only 1.92%.

Conclusion:

The experiments conducted by Xie and Witgen[9] on the UCR datasets have shown that using their feature extraction methodology allow DTW to achieve much higher accuracies than using just 'raw' values . Optimal paths that closer to the diagonal region of the cost matrix have smaller distances than paths that are further away from the diagonal. The improved accuracy is credited to the construction of optimal paths that are closer to the diagonal than before for sequences belonging to the same class. However the results from their experiments do not provide any information on the degree of translation undergone by these warping paths. The results from my experiments provides this insight. Using the domain independent methodology only leads to a fractional increase in the accuracy of the DTW when it employs a window constraint. This shows that for similar patterns, the degree of translation undergone by the optimal warping paths is very small as very few ideal optimal paths enter the region defined by Sakoe-Chuba band.

The low accuracies of the DTW can be improved by increasing the width of the Sakoe-Chuba band. But increasing the width also leads to a reduction in speed. This provides the motivation to investigate alternative methods that can a better balance between the two conflicting goals of accuracy and speed. In the next section, I investigate a self-proposed method that is an alternative approach to using window constraints to speed up DTW. The new proposed methodology uses the data-driven feature extraction process that is discussed in the current section. The aim here to construct a domain independent framework that allows DTW to attain improvements in both speed and accuracy over the Sakoe-Chuba band window constrained DTW.

5.2 Extending DTW

The feature extraction methodology discussed in the previous section offers no advantage of dimensionality reduction. The time series sequence is mapped to sequence of vectors whose length is $\|X_n\| - 2$. (where $\|X_n\|$ denotes the length of the original time series sequence). The MFCC feature extraction process (4.2) on the other hand, does provide the advantage of dimensionality reduction. The time series sequence is first segmented into series of frames of length 20ms

i.e 200 points. Through appropriate functional mapping, each frame is then mapped to a vector. Hence the time complexity of the DTW algorithm have been reduced from O(n^2) to O (u^2) where $u = \frac{n}{l}$ (l denotes the length of the frame).

Using the MFCC feature extraction method as motivation, I propose the following methodology:

i Apply the feature extraction process of 5.1 to embed the information of local and global trends.

ii Partition the sequence of vectors into frames of width m. The default value of m is set to 50. For speech utterances in the TIGITS dataset, this is equivalent to having frames of size 5ms. Unlike the windowing process of MFCC extraction, the frames here correspond to sequence of vectors rather sequence of numerical values. This partitioning process reduces the length of the sequences by m times and hence decreases the time complexity of the DTW algorithm from O(n^2) to O$(\frac{n}{m})^2$.

iii Adapt the cost metric of the DTW algorithm to work on series of frames rather than series of vectors

The problem can now be shifted to finding an appropriate kernel that can be used to compute the similarity between frames composed of feature vectors. Ideally, we want a metric that takes into account the variation of speed and time when comparing two similar subsequences. To understand why we need the metric to be invariant to variations in speed and time, lets consider the following two signals:

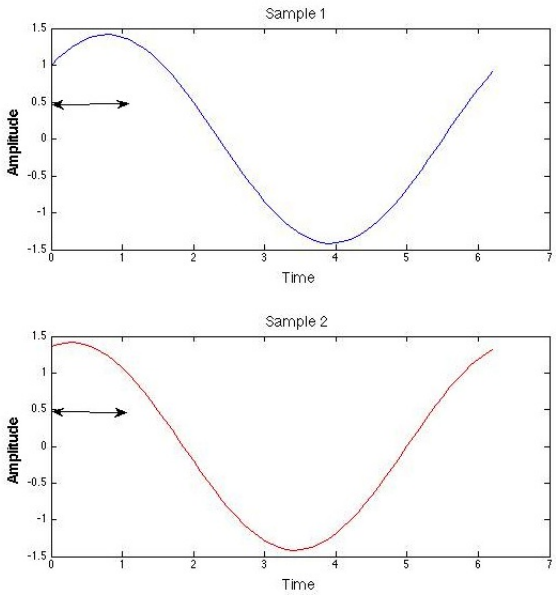


Figure 5.5: Two signals separated by translation

The signal denoted by the 'red' color is a 'slower' version of the signal denoted by the 'blue' color . Using a euclidean metric in this scenario is inappropriate. The euclidean metric in this context is identical to linear time warping where the two subsequences/frames will be compared based on a linear match of the temporal dimensions of the two sequences/frames. In our context, we need a kernel that computes the similarity between two sub-sequences by warping the time axis. Intuitively speaking, the kernel must behave like a DTW algorithm that compares the global and local properties associated with a point in one frame with the global and local properties of all points in the second frame as illustrated by the figure below.

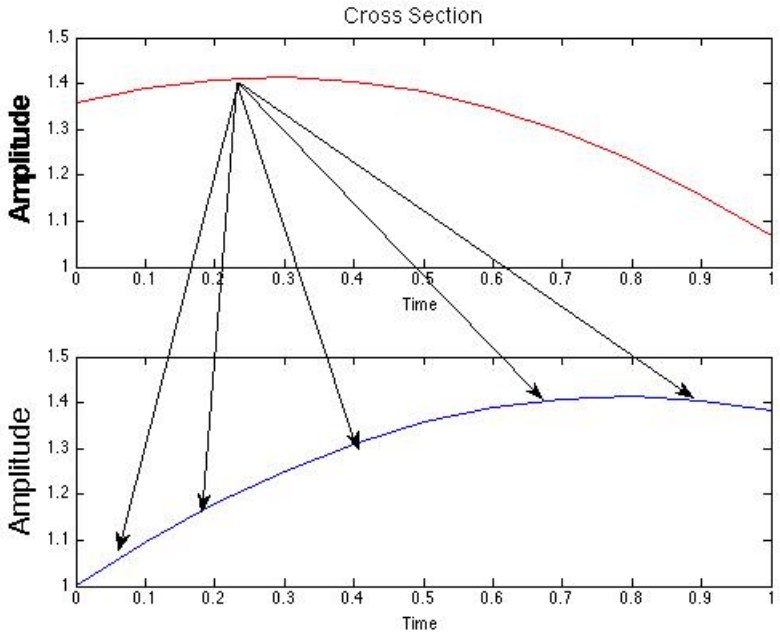


Figure 5.6: Two identical subsequences varying in time

The motivation behind my proposed kernel function comes from the polynomial kernel. Let x and z be two two-dimensional vectors and consider the simple polynomial kernel of degree 2 :$k(x,z) = (x^T z)^2$. This kernel can expressed as :

$$\begin{aligned} k(x,z) &= (x^T x')^2 \\ &= (x_1 z_1 + x_2 z_2)^2 \\ &= x_1^2 z_1^2 + 2x_1 z_1 x_2 z_2 + x_2^2 z_2^2 \\ &= (x_1^2, 2x_1 x_2, x_2^2)(z_1^2, 2z_1 z_2, z_2^2)^T \\ &= \phi(x)^T \phi(z) \end{aligned}$$

The 2nd order polynomial kernel is equivalent to a corresponding feature mapping $\phi(x)$ that maps a two dimensional vector to $(x_1^2, 2x_1x_2, x_2^2)$ where each attribute is monomial of order 2 . Generalising this notion to order M then $k(x,z) = (x^T z)^M$ represents the sum of all monomials of order M. If we imagine x and z to be two images, then the polynomial kernel represents a particular **weighted** sum of all possible products of M pixels in the first image with M pixels in the second image.
Using this as motivation I propose the following kernel:.

$$k(x,z) = < \sum_{i=1}^{n} x_i, \sum_{j=1}^{n} z_j >$$

Here n denotes the length of the frame while x_i and z_j correspond to the individual 4-dimensional feature vectors that make up the frame. The above kernel corresponds to the dot product between the individual sums of the two frames. Although it may not look obvious, this construction actually allows the comparison between all feature vectors in the first frame with all feature vectors in the second frame as shown below:

$$\begin{aligned} k(x,z) &= < \sum_{i=1}^{n} x_i, \sum_{j=1}^{n} z_i > \\ &= < (x_1 + x_2 + x_3 + ..), (z_1 + z_2 + z_3 + ..) > \\ &= < x_1, z_1 > + < x_1, z_2 > + < x_1, z_2 > ... + < x_2, z_1 > + < x_2, z_2 > + ... \end{aligned}$$

From above expression, we can see that the proposed kernel corresponds to a sum of all possible dot products of pairs belonging to the set $\{(x_i z_i) | x_i \in \text{seq1}, z_i \in \text{seq2}\}$. It is easy to check that this proposed kernel is in fact a valid kernel function:

- K(x,z)= K(z,x) $\Rightarrow$the function is symmetric.
- The kernel can be written as a scalar product in feature space: K(x,z) $=\phi(x)^T \phi(x)$ where the feature mapping corresponds to a finite summation of vectors $\phi(y) = \sum_{i=1}^{n} y_i$.

Augmenting the kernel to the DTW algorithm allows DTW to work on long time sequences without using a window constraint. To use this kernel as an appropriate cost function in the DTW algorithm, we need a functional mapping that:

1. constraints the codomain to be in the range from 0 to ∞.
2. ensures larger values given by the function signify great degree of dissimilarity and smaller values signify a high degree of similitude.

An inspection of the kernel function shows the function represents the sum of dot products of all vectors in one frame with all vectors in the second frame. An ideal cost function that make use of dot products is the *arc-cosine*. Using an appropriate substitution, I have thus constructed the following cost metric:

$$\theta = \frac{< X, Z >}{|X||Z|}$$

where θ is the *arc-cosine* distance, $X = \sum_{i=1}^{n} x_i$ and $Z = \sum_{j=1}^{n} z_i$.

A formal outline of the entire methodology is given by the algorithm below:

Algorithm 4 Proposed DTW

procedure VALUE-BASED($seq1, seq2$) ▷ two sequences of feature vectors
  seq_1←segment(seq1,n) ▷ Segment the sequences using a window of size n
  seq_2←segment(seq2,n)
  for i=1: to length(seq_1) **do** ▷ Initialise the DTW cost matrix
    DTW(i,0) = ∞
  end for
  for i=1 to length(seq_2) **do**
    DTW(0,i) = ∞
  end for
  for i=2 to length(seq_1) **do**
    for j=max(2, i-w) to min(length(seq_2), i+w) **do**
      DTW(i,j) = $\theta = \frac{<X,Z>}{|X||Z|}$ + min{ DTW(i-1,j)+DTW(i,j-1)+DTW(i-1,j-1)}
    end for ▷ $X = \sum_{i=1}^{n} x_i$ and $Z = \sum_{j=1}^{n} z_i$
  end for
  return result = $\frac{\text{DTW(n,m)}}{nm}$ ▷ n=length(seq1), m=length(seq2)
end procedure

5.2.1 Testing the methodology

The primary objective is to improve the performance of the DTW algorithm in problem domains where minimising time complexity has a high priority. To investigate whether the proposed changes does provide a better alternative to using the Sakoe-Chuba band window constraint, I have conducted the following experiments:

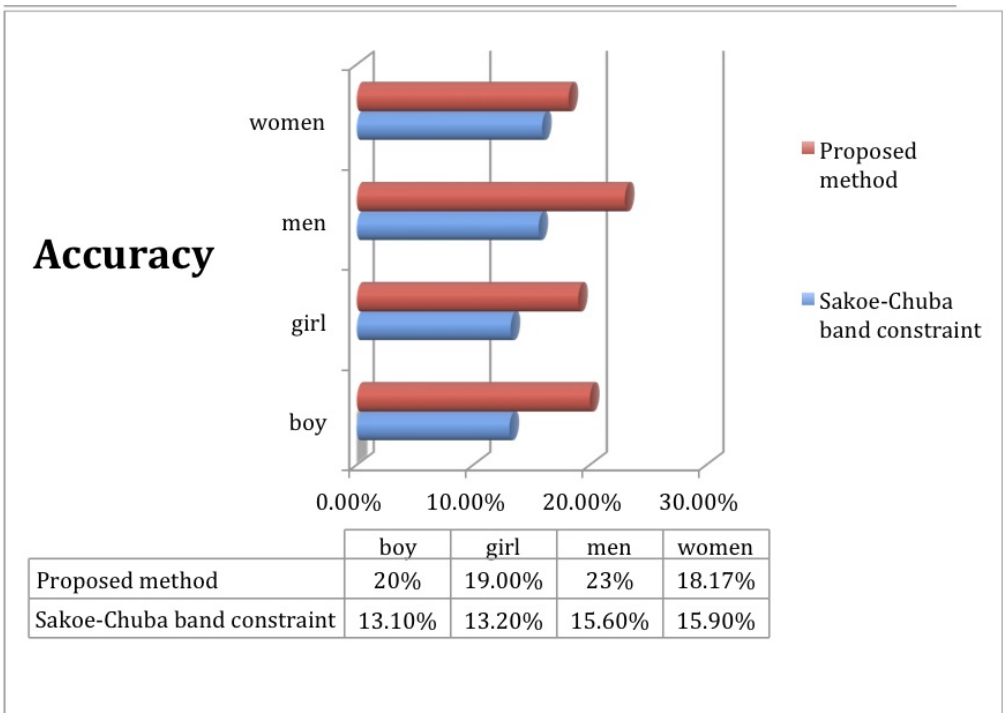
Experiment 1

Dataset used : The TIDIGITS dataset.

Model used : The model discussed in section 5.1: the framework consists of a two stage preprocessing step that involves feature selection process consisting of 'silence' removal followed by the domain independent feature extraction methodology proposed in [9]

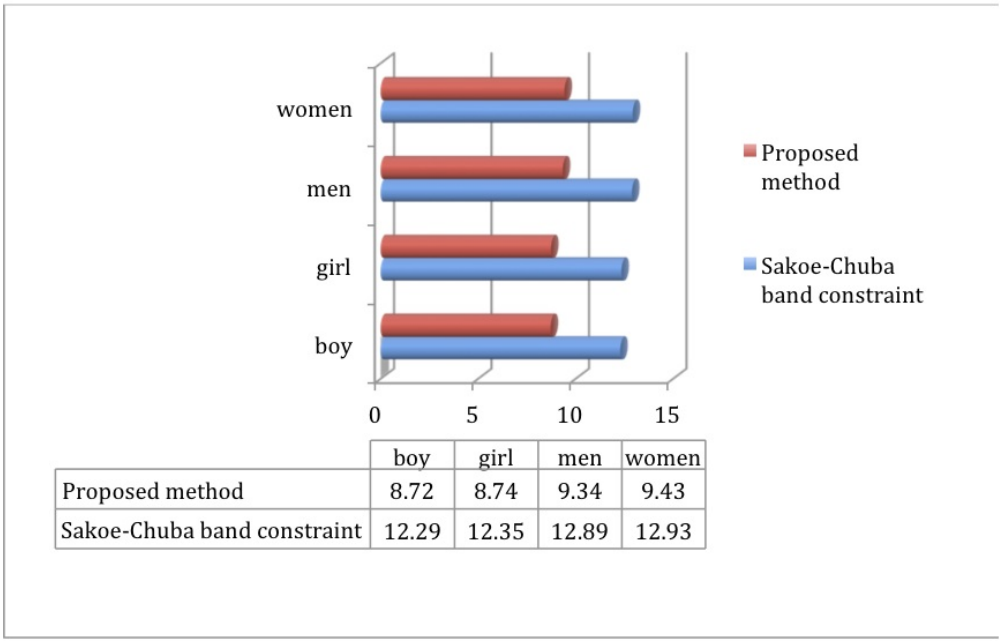
Variables being compared: Sakoe-Chuba band window constraint VS the proposed methodology

RESULTS:



	boy	girl	men	women
Proposed method	20%	19.00%	23%	18.17%
Sakoe-Chuba band constraint	13.10%	13.20%	15.60%	15.90%

Figure 5.7: Accuracy



	boy	girl	men	women
Proposed method	8.72	8.74	9.34	9.43
Sakoe-Chuba band constraint	12.29	12.35	12.89	12.93

Figure 5.8: $log_e(\text{run time})$

There are quite number of interesting observations that can made from figures 5.7 and 5.8.

- In comparison to employing the Sakoe-Chuba band window constraint, the alternative method allows DTW to improve its accuracy across all 4 categories by 6.54% on average.
- Equipped with the new changes, the run time of the algorithm has decreased exponentially by an **order** of 3.1 on average. The reduction of the time complexity is mainly due to the partitioning of the sequence into time slices of width 5 ms. The reduction in the length of the sequences by an order of 50 results in the shrinkage of the search space of DTW thus causing the algorithm to improves its speed.

The proposed methodology can also be adapted to use domain-dependent features. As I have stated earlier, the main motivation for constructing this framework is to provide better performance in terms of both accuracy and run-time for time series problem domains where the use of rigid window constraints is deemed necessary. In the previous chapter, we have seen that when the MFCC feature extraction is used as the only preprocessing step, to minimise the run-time, the use of the adaptive window constraint was deemed necessary. However from the experimental results, it can been seen enforcing such a rigid constraint results in a high error rate. To check that the new changes allow DTW to perform equally well when using domain dependent features, I have repeated the same experiment again but this time, as a preprocessing step I have just used the MFCC feature extraction.

A summary of the results are as follows:

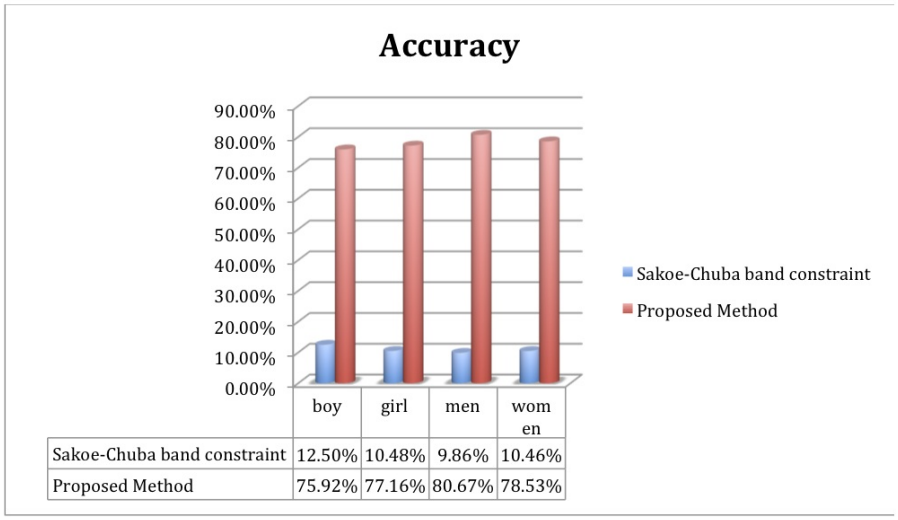
Experiment 2

Dataset used : The TIDIGITS test and training data.

Model : The model framework consists of single preprocessing step that involves the extraction of MFCC features

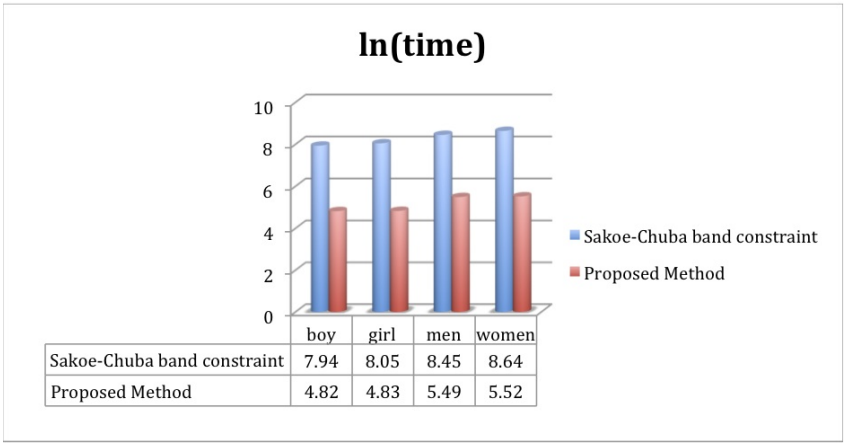
Variables being compared: employing window constraints vs the new proposed changes

Results



	boy	girl	men	women
Sakoe-Chuba band constraint	12.50%	10.48%	9.86%	10.46%
Proposed Method	75.92%	77.16%	80.67%	78.53%

Figure 5.9: Accuracy



	boy	girl	men	women
Sakoe-Chuba band constraint	7.94	8.05	8.45	8.64
Proposed Method	4.82	4.83	5.49	5.52

Figure 5.10: ln of the run time: $log_e(time)$

Observation The results are consistent with the findings in the previous experiment. The accuracy of the DTW algorithm has improved by over 60%

on average and the average run time has decreased exponentially by an order of 3.12. Hence in comparison to the window constrained DTW, the new framework takes better advantage of employing domain dependent features.

However, since the tests so far has been conducted on the TIDIGITS test set, it is possible that this new approach is only tailored for this particular dataset. Thus to confirm that the boost in the performance is not tailored for this particular time-series dataset, I have the following experiments using the model of experiments on the UCR datasets: InlineSkate and CINC_ECG_TORSO.

Experiment 3

Datasets used : InlineSkate and CINC_ECG_TORSO

Setup :

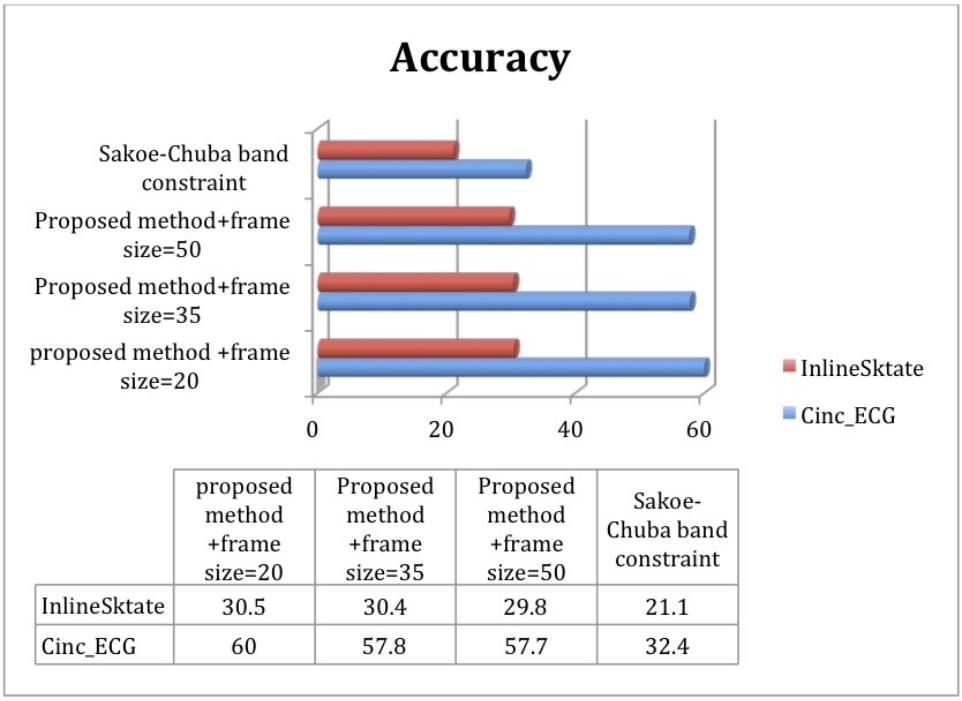
Model used : The model discussed in section 5.1: the framework consists of a two stage preprocessing step that involves feature selection process consisting of 'silence' removal followed by the domain independent feature extraction methodology proposed in [9]

In the proposed approach, the width of the time slices has been kept fixed at a default value of 50 so far. In this experiment, I also investigate the influence of this parameter on the performance of the DTW. Decreasing its value reduces the size of the time slices which in principal should increase both accuracy and time-complexity . The core kernel used by the new algorithm is based on the function:

$$k(x,z) = < \sum_{i=1}^{n} x_i, \sum_{j=1}^{n} z_j >$$

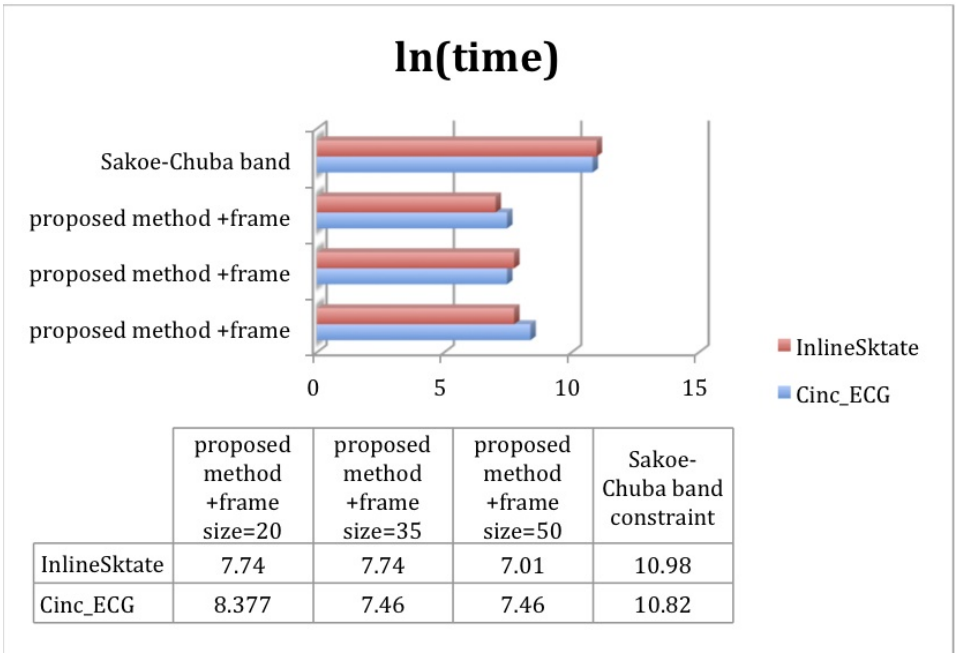
k(x,z) represents the sum of all possible dot-products. Using smaller subsequences allow the similarity measure to be dominated by the dot products of points whose local and global features are most alike. However, this suffers from the drawback of achieving lesser dimensionality reduction. Thus the time and computational complexity suffers.

RESULTS



	proposed method +frame size=20	Proposed method +frame size=35	Proposed method +frame size=50	Sakoe-Chuba band constraint
InlineSktate	30.5	30.4	29.8	21.1
Cinc_ECG	60	57.8	57.7	32.4

Figure 5.11: Accuracy



	proposed method +frame size=20	proposed method +frame size=35	proposed method +frame size=50	Sakoe-Chuba band constraint
InlineSktate	7.74	7.74	7.01	10.98
Cinc_ECG	8.377	7.46	7.46	10.82

Figure 5.12: Run time in ln

Observation

- The proposed changes does provide a **better** alternative approach to using rigid constraints for problem domains where the time complexity is of high priority and the use of window constraints can not be avoided.

The accuracy of the DTW algorithm under the new methodology is significantly higher then employing a window constrained DTW on sequences of extracted local and global feature vectors.

- Decreasing the size of the time slices only leads to a minimal increase in accuracy. Thus using the default value of 50 seems to be safe option as the algorithm better accuracy and speed than using the rigid Sakoe-Chuba band constraint .

Conclusion

- In chapter 3, we have seen that under the window constraint the time complexity of the algorithm decreases from O(n^2) to O(wn). From the experimental results, it can be observed that the new proposed changes allow DTW to achieve run times that are exponential lower than the run times achieved by the window constrained DTW. Thus it can be concluded that the time complexity of the proposed DTW variant is smaller that O(wn).
- We have seen so far that increasing the speed of DTW negatively impacts the accuracy of the algorithm . The new methodology actually provides an exception. The use of the kernel function improves the accuracy of the DTW which implies that matching frames using the new cost function is a better alternative than employing the euclidean distance to find local optimal warp path only through the constraint window. Combining this approach with the feature extraction method discussed in [9] results in the construction of a model that can applied for **different types** of time series datasets.

Chapter 6

Improving the speed of the 1 nearest neighbour classifier

High dimensional data vectors $\{x_n\}$ typically lie close to a non- linear manifold [24] whose intrinsic dimensionality is smaller than that of the input space as a result of strong correlations between the input variables. In data mining, to understand this intrinsic dimensionality of the data, the Single Value Decomposition(SVD) algorithm is widely used. SVD simplifies the data representation by showing only the number of important dimensions that captures the **variance** of the data. Intuitively speaking, SVD projects the high dimensional data vectors to an orthogonal subspace that maximises the variance of the data. Recent works[12] have shown that SVD can be applied to achieve dimensionality reduction in time sequences given that the sequences satisfy certain constraints. Applying SVD as a preprocessing step to such time series datasets can result in a significant reduction of the runtime of a classification/clustering algorithm. The focus of this chapter is to investigate compatible preprocessing methods that can be used alongside SVD to improve the accuracy of the 1 nearest neighbour classifier while minimising its run time.

6.1 Motivation

Unlike other datasets, clustering/classifying time series sequences by conventional distance metrics, such as the euclidean, may not result in high accuracy. The scope of such metrics is quite limited in the fact that they require sequences to share the same length and comparison of sequences is based on a linear match of the temporal dimensions of the two sequences. This represents a substantial drawback as the metrics fail to take into account that similar

patterns may vary in terms of speed, length and scale. Therefore, this provides the motivation to look for preprocessing feature extraction techniques that can extract translation and scale invariant features that capture information about the intrinsic trends in the data.

However, recent works have shown that there exists time series domains where the use of conventional methods can still achieve accurate results. Korn in his paper [12] has shown that for time series sequences that share the same **speed** and **length**, the conventional methods can achieve good clustering/classification performance. If intra class sequences have the same speed i.e no time offsets then they will also share the same time localised trends(see figure 6.1). Applying SVD as a preprocessing step, results in the extraction of latent factors that represent the localised patterns in time. Through mapping the time series sequences into the latent space, we can therefore construct a simplified representation of the data which allows conventional methods such as the euclidean metric to cluster time sequences with lesser run time.

To see how SVD accomplishes this feat, lets consider the following toy example:

Let X be a matrix where rows correspond to customers, columns correspond to days and the values are the dollar amounts spent on phone each day.

X =

customer	wed	thurs	frid	sat	sun
A	1	1	1	0	0
B	2	2	2	0	0
C	1	1	1	0	0
D	5	5	5	0	0
E	0	0	0	2	2
F	0	0	0	3	3
G	0	0	0	1	1

Observation

- There are only distinct time localised trends. From this we can infer that there are two groups of customers: one group makes calls only during week days while the other group makes calls only during weekends.

Using SVD to factorize the matrix gives:

$$
\begin{aligned}
X &= U \times D \times V^T \\
X &= \begin{pmatrix} 0.18 & 0 \\ 0.36 & 0 \\ 0.18 & 0 \\ 0.90 & 0 \\ 0 & 0.53 \\ 0 & 0.80 \\ 0 & 0.27 \end{pmatrix} \times \begin{pmatrix} 9.64 & 0 \\ 0 & 5.29 \end{pmatrix} \times \begin{pmatrix} 0.58 & 0 \\ 0.58 & 0 \\ 0.58 & 0 \\ 0 & 0.71 \\ 0 & 0.71 \end{pmatrix}^T
\end{aligned}
$$

Note: The columns in the orthogonal matrix V that are the eigenvectors of the zero singular values in the diagonal matrix have been omitted. Omitting these column vectors results in no loss of information.

Key observations

- The rank of the matrix X is 2. This shows that there are effectively two groups of customers: weekday(business) and weekend(residential) callers.
- The U matrix can be thought of as a customer to pattern similarity matrix. The first column vector of U corresponds to the weekday customers while the second column vector represents the weekend customers
- The V matrix can be thought of as day to pattern similarity matrix. In the case of days, there are two patterns: the weekday pattern i.e {Wed,Thurs,Frid} and the weekend pattern {Sat,Sun}

For SVD to be applied successfully across all time series datasets , there are two fundamental issues that must be addressed:

1. Time complexity: The time complexity of SVD is O(D^3) where D represents the dimensionality of the data. For domains where the dimensionality of the data is really high, applying SVD as a preprocessing step increases the run time of a clustering/classification algorithm.
2. Variable dimensionality: SVD cannot be directly applied to datasets where the individual data points do not share the same dimensionality. This issue is specially seen in time series data sets such as speech. The acoustics samples in such datasets vary in length.

To address the above issues and the drawbacks associated with performing nearest neighbours classification on time series sequences, in the following three sections, I investigate :

i methods to improve the time complexity of SVD in handling high dimensional sequences when the size of the dataset is very small.

ii preprocessing methods that can used along side the SVD feature extraction scheme to boost the accuracy of the classifier

iii techniques to adapt SVD in handling time series sequences of variable length

6.2 Improving Time Complexity

For datasets where N(the number of samples) $\ll$ D(dimensionality of the data), Bishop [24] offers the following solution that reduces the time complexity of computing SVD from O(D^3) to O(N^3):

Let X be the constructed D by N data matrix where D is the number dimensions and N is the number of samples and D$\ll$N. By factorizing X, we get X=$U \times D \times V^T$. To achieve dimensionality reduction, we need to consider only the subset of columns vectors of V that are associated with the non-zero singular values. Since N $\ll$D, there will at most N-1 of these singular values. As stated earlier, the columns of the V are the eigenvectors of the covariance matrix XX^T.

The eigen-decomposition of the covariance matrix is $XX^T u_i = \lambda_i u_i$. XX^T is D by D matrix and the computational complexity of computing the eigenvectors is O (D^3).

Now if we multiply both sides by X^T we get :

$$(X^T X)X^T u_i = \lambda_i X^T u_i$$

Setting $v_i = X^T u_i$ gives us the eigenvectors of $X^T X$.

From the above expression, it can be observed that the eigenvectors of $X^T X$ and XX^T share the **same** eigenvalues. Using this insight, we can thus solve the eigenvector problem in spaces of lower dimensionality with computational cost O(N^3) instead of O(D^3). The required eigenvectors of the covariance matrix can then be computed by the using the following expression:

$$u_i = \frac{1}{\lambda^{(\frac{1}{2})}} X v_i$$

To verify the above approach, I have conducted the following experiment :

Objective : The experiment is designed to verify whether applying SVD as a preprocessing step can improve the run time of 1 nearest neighbour classifier on time series datasets, without affecting its accuracy.

Datasets used : InLineSkate and Cinc_ECG_TORSO.

Unlike the speech corpus, the intra class variance of the time series sequences in the UCR datasets are much lower. This is primarily due to the fact that sequences of the same class can be described by the same time-localised trends. Figure 6.1 shows examples of two sequences belonging to class '1' in the Cinc_ECG_TORSO dataset. Apart from having the same shape, the two sequences also posses the same speed i.e they have no time offset. From figure 6.1,it can seen that at around t=600, both patterns undergo a dip followed by a sharp peak. The fact that all sequences in the dataset share the same length and intra class sequences have no time offset allow classification algorithms to treat the problem as a normal classification problem. This also provide the context to apply SVD directly to achieve dimensionality reduction. For this experiment, SVD has been applied to extract all the eigenvectors of the covariance matrix that corresponds to non-zero singular values. Since the number of samples $\ll$ the length of the sequences, the size of the latent space will be smaller than the original input space leading to dimensionality reduction without any loss of information.

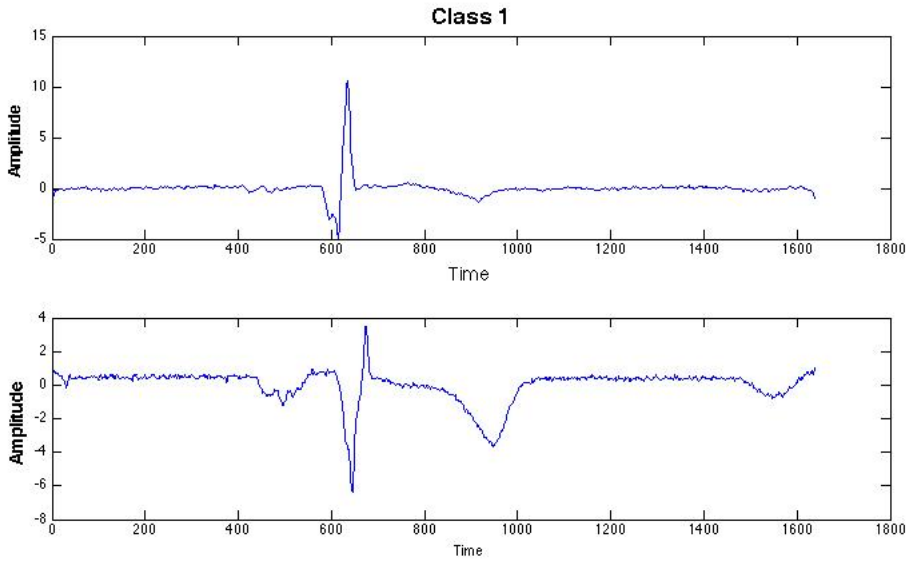


Figure 6.1: Time series sequences corresponding to class '1' in the Cinc_TORSO_ECG dataset

Results:

	raw features	latent features
Cinc_ECG_TORSO	89.71	89.71
InlineSkate	34.18	34.18

Table 6.1: Accuracy

	latent features	raw features
Cinc_ECG_TORSO	0.24	0.72
InlineSkate	0.21	0.92

Table 6.2: Run times in seconds

Observations

- The accuracy of the 1 nearest classifier in both cases is consistent with the 1NN error rate given for each of the datasets in the UCR data source webpage.
- Applying SVD as a preprocessing step has no effect on the accuracy on the algorithm but reduces the run time of the classifier. Thus using Bishop's[24] formulation, we can use SVD to reduce the time complexity of clustering/classification algorithms for high dimensional time series datasets that satisfy the constraint identified by Korn[12].

Chapter 7

Improving the accuracy of the 1 nearest neighbour classifier

In the last chapter, we have seen that the 1 nearest neighbour classifier using single value decomposition achieves good performance in both accuracy and run time in classifying sequences belonging to the Cinc_ECG_TORSO. The Cinc_ECG_TORSO is a rare example of a time series data set where there is no translation offset between sequences of the same class and intra class sequences share the same **time localised** local and global trends(see figure 6.1). This allows sequences to be fairly accurately classified/clustered based only on a linear match of their temporal dimensions.

For most time series data sets however, the above approach presents a limitation. The intra class variances in most time series datasets will be much higher as sequences of the same class may vary in size, shape or speed. The InlineSkate dataset is a prime example of such a dataset. Figure 6.2 shows examples of two instances of class 2 in the InlineSkate dataset. The two instances of class '2' share the same time localised global trend but posses different local time localised trends: in the period between t= 400 to 600, the first sequence is experiencing an upward trend while the second sequence is seen to experience a downward trend. Performing comparison by conducting only a linear match of the temporal dimensions is inappropriate in such cases as they increase the degree of dissimilarity between similar patterns. This creates the motivation to investigate feature extraction methodologies that minimises the intra class variance of time series sequences by mapping the data to a feature space that captures information about common class attributes such as global shape, trends etc. The construction of a such discrete feature sets allows the time series classification problem to be approached as a general classification problem.

In the following chapter, I will be concentrating on constructing feature extrac-

tion techniques that capture information about the **global shape** of time series sequences. In the first half of this chapter, I will be exploring wavelet-based feature extraction techniques and will subsequently look into how they can be used alongside SVD as preprocessing step to improve the accuracy of the 1 nearest neighbour classifier.

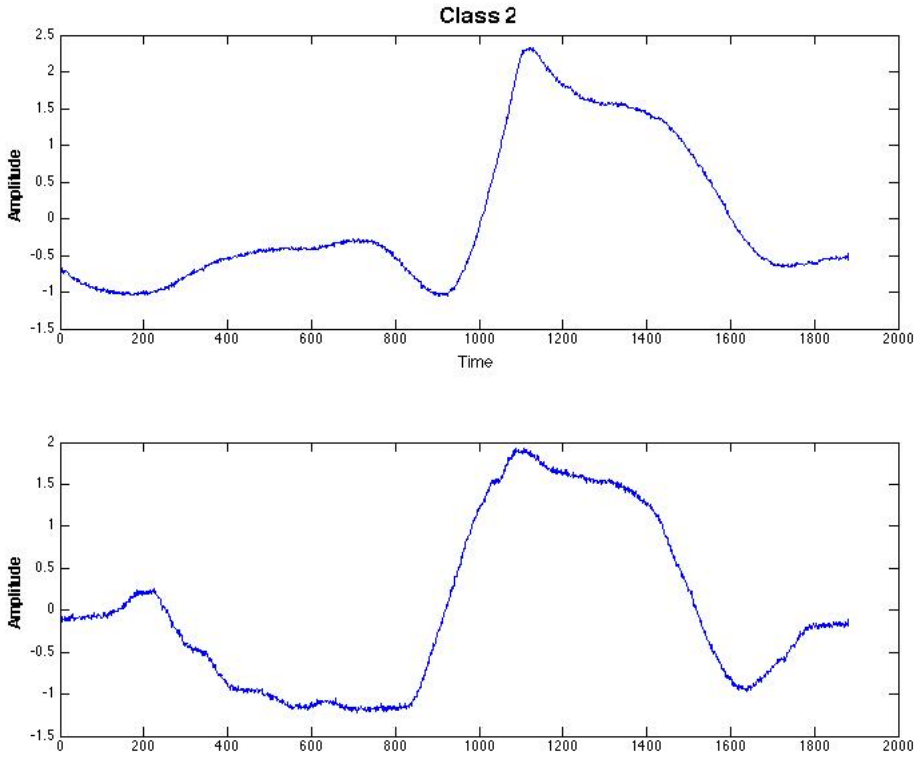


Figure 7.1: Time series sequences corresponding to class '2' in the InlineSkate dataset

7.1 Wavelet-based Feature Extraction

7.1.1 Background

The inspiration for the wavelet transform came from the idea of multiresolution analysis[25]. A multiresolution analysis decomposes a signal into a smoothed version of the original signal and a set of detail information at different scales.This type of decomposition is most easily understood by thinking of a picture (can be thought of as a two dimensional signal) and removing from the picture information that distinguishes the sharpest edges, leaving a new image that is slightly blurred. This blurred version of the original picture is a rendering at a slightly coarser scale. We then recursively repeat the procedure.

The wavelet transform specifies such a multiresolution decomposition, with the wavelet functions defining the bandpass filter that determines the detail information. and associated with each wavelet is a smoothing function, which defines the complementary lowpass filter.

Mathematically, we can motivate the definition of the wavelet transform as follows:

Let $L^2(R)$ be the set of all discrete functions with finite energy. Any discrete function $f \in L^2(R)$, can be written as a linear combination of translations of the scaling function $\phi(st)$ as shown below:

$f(t) = \sum_{n=0}^{N} \alpha_n \phi(st - n)$ where $\phi(st)$ is defined as :

$$\phi(st) = \begin{cases} 1 & \text{if } 0 \leq st < 1 \\ 0 & \text{otherwise} \end{cases}$$

Under this representation, the following facts hold.

i The scaling function is orthogonal to its integer translates.

ii As we increase the scale, the support of the function decreases.

iii The subspaces spanned by the scaling function at low scales are nested within those spanned at higher scales. For example if represent s as powers of 2 then: $\phi_{-1}(t) = \phi(2^{-1}t) = \phi_0(t) + \phi_0(t-1)$.

Before proceeding any further, it is necessary to define the concept of the wavelet. A wavelet [26, 27, 28] is a smooth and quickly oscillating function with good localisation in both frequency and time. To perform wavelet transformation the Haar function is commonly chosen as the mother wavelet ψ. For any scale s, the function $\psi(st)$ is orthogonal to $\phi(st)$ defined at the same scale but can be expressed in terms of higher order scaling functions $\phi(s't)$ where $s' > s$. Using this knowledge alongside (iii), the wavelet transformation is applied to decompose a finite signal f by projecting f onto two orthogonal subspaces W_1 and $V_1$of $L^2(R)$ where $L^2(R) = V_1 \oplus W_1$. If $f(t) = \sum_{n=0}^{N} \alpha_n \phi(st - n)$ then W_1 is spanned by $\{\psi(s't - n) | n \in [1,..N]\}$ and V_1 is spanned by $\{\phi(s't) | n \in [1,,N]\}$ where $s' < s$.

Note :To ensure that $\{\psi(s't - n) | n \in [1,..N]\}$ forms an orthogonal basis, the scales are restricted to powers of 2[29, 26].

The process is recursive. After removing the first set of detail information from f, we are left with a slightly smoothed version of f. We iteratively remove detail information by applying the transform on the smoothed versions at each scale to progressively extract coarser details of f . We stop the process once we have enough detail information or a smooth enough version to

do the analysis we desire. Thus the subspaces $\{V_m\}|_{m=1}^{M}$ satisfy $V_{j+1} \subset V_j$. The details of f at any scale **m** is a projection of f onto the subspace W_M. The projection is represented by the map $Q_M : L^2(R) \rightarrow W_M$ where $Q_M f = \sum_{n=1}^{K} < f, \psi_{Mn} > \psi_{Mn}$(the subspaces W_m are spanned by dilations and translations of the mother wavelet ψ).

Furthermore there exists another operator P_M that maps f to its approximation at scale **m**: $P_M : L^2(R) \rightarrow V_M \subseteq L^2(R)$. The functional space $L^2(R)$ can therefore be expressed as $L^2(R) = \oplus_{m=1}^{M} W_m \oplus V_M$. where any finite signal $f \in L^2(R)$ can be expressed as :

$$f(t) = \sum_{n=0}^{N} \alpha_n \phi(t-n) = P_M f + \sum_{m=1} Q_m f$$

7.1.2 Wavelet-based Features

The wavelet transformation allows a time series sequence to be described in terms of an approximation of the original sequence plus a set of details ranging from fine to coarse. Given any time series X, the wavelet coefficients $H_j(X)$ at any scale j can be represented by the series $\{A_j, D_j, D_{j-1}...D_1\}$ where A_j are the coefficients of the scaling function at scale j and $D_j...D_1$ are the coefficients of Haar wavelet at different scales . The broad trends of the sequence such as global shape are **preserved** in the approximation part of the signal i.e $A_j(X)$ while the **localised** changes are kept in the detail parts i.e $\{D_j..., D_1\}$[27]. .The decomposition presents **no loss** of information. The original signal can be fully reconstructed from the approximation part and the detail parts.

The transformation has the property of achieving both time and frequency localisation of a time series. Each scale corresponds to particular band of frequencies. The coefficients of ψ and ϕ at different scales hence capture the behaviour of the signal at different frequency bands. For signals that are nonzero only during finite spans of time, the wavelet transform has nonzero elements that are concentrated around that time.This allows us to analyse singularities at particular points in time. Thus as a feature extraction step, performing wavelet decomposition can be highly advantageous as it allows us to decompose a signal and inspect its behaviour at different frequencies and time.

As we have discussed in the previous section, the intra class sequences in the Cinc_ECG_TORSO dataset share both local and global trends. In such problem domains, the coefficients of both ψ and ϕ at different scales are required to form accurate comparison. Figure 7.2 show examples of two instances belonging to class '3' and two instances belonging to class '4'. From an observation

of the plots, it can been seen that apart from sharing global trends, intra class sequences also tend to share similar local trends. Information about the global shape is captured by the coefficients of ϕ which behave as global trend descriptors while the finer details are expressed by the coefficients of ψ. Thus, to make accurate comparison, it is necessary to take the coefficients of both ψ and ϕ into account.

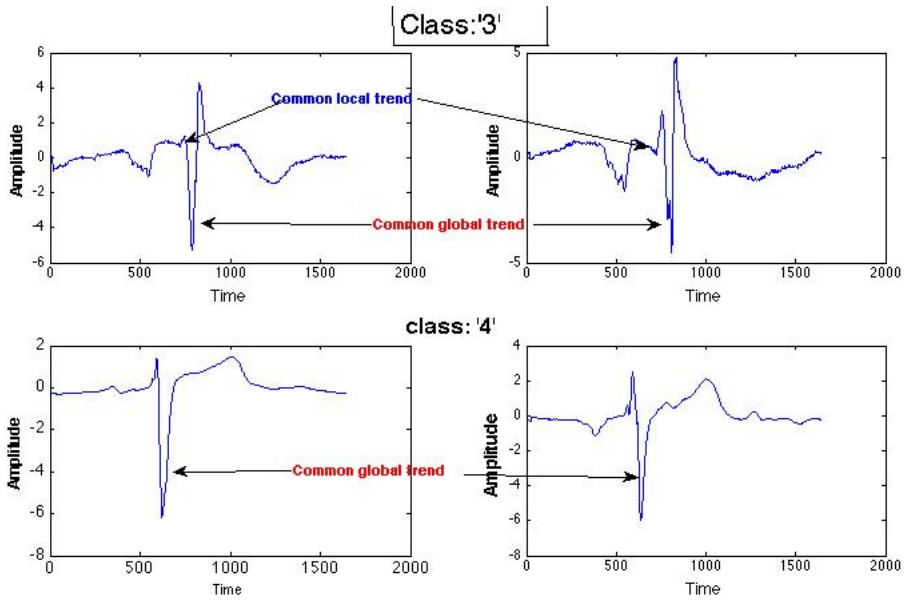


Figure 7.2: The top two plots corresponds to instances of class '3' while the bottom two plots correspond to instances of class '4'

However, for the the InlineSkate dataset, the global shape is the only commonality between sequences in the same class. Here intra class sequences only share global time-localised trends. Figure 7.3 shows examples of samples belonging to the class '2'. The red lines represent smoothed fitted curves to the time sequences to to capture the overall shape. Both instances share a global trend(shown in the figure) around the same time but differ in the local trends.

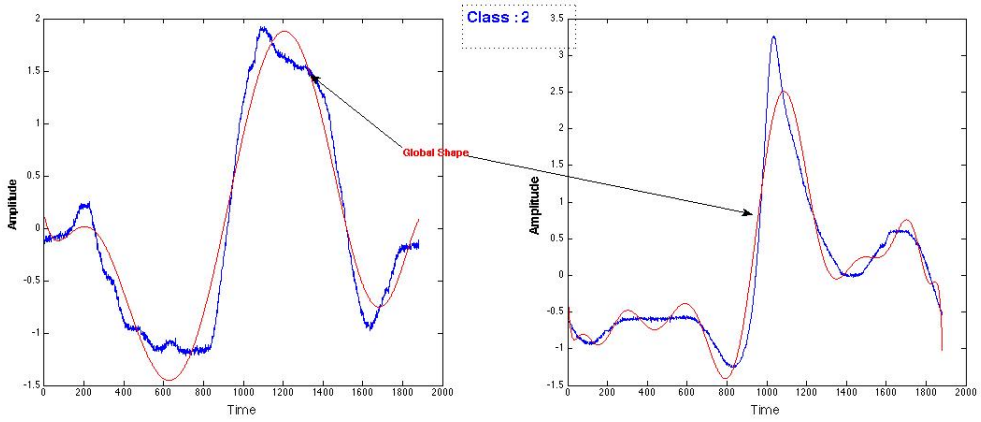


Figure 7.3: The plots correspond to instances of class '2' of the InlineSkate dataset

In such cases, it can be assumed that the coefficients of ϕ play a greater role in discriminating classes the coefficients of the wavelets.

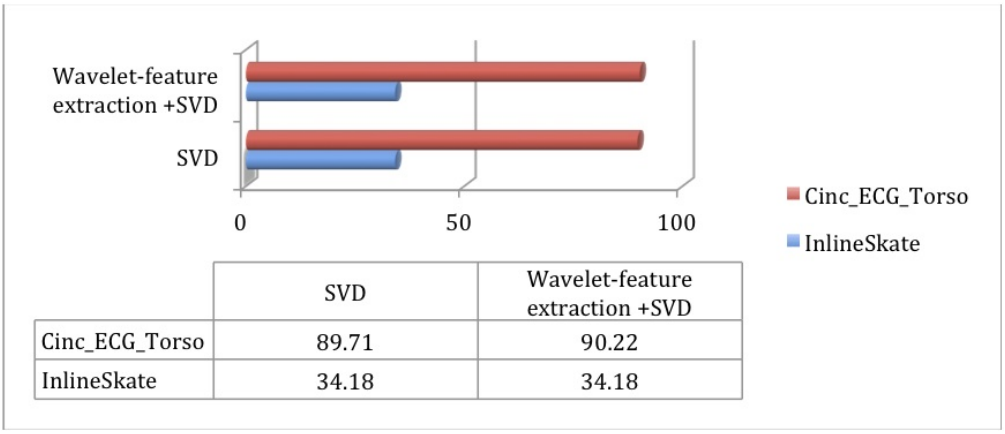
To investigate whether applying wavelet decomposition prior to SVD as preprocessing step improves the performance of the 1 nearest neighbour, I conducted the following experiment:

Datasets: Cinc_ECG_TORSO and InlineSkate

Preprocessing: Wavelet decomposition followed by single value decomposition.

- For sequences in the Cinc_ECG_TORSO dataset, I have applied wavelet decomposition up to the lowest possible scale. Applying the transformation on the smoothed versions at each scale progressively extracts coarser details from the signal. The local trends will be captured by the detail coefficients in the upper levels while the coarser details such as global trends will be captured by the detail coefficients in the lower levels. For this analysis, I have omitted the detail coefficients corresponding to scale 1 i.e D_1. The reason being regions in high frequency have a lot of noise embedded in them. Since the first level details capture information about the behaviour of the signal in the highest frequency bands, discarding those details achieves reduction in both the noise and the dimensionality of the sequence.
- For sequences in the InLineSkate dataset, I have applied wavelet decomposition up to a depth of 4 levels. Since the commonality of the classes is captured by the global shape. I have only considered the approximation coefficients ϕ at scale 4.

The results of the experiments are as follows:



	SVD	Wavelet-feature extraction +SVD
Cinc_ECG_Torso	89.71	90.22
InlineSkate	34.18	34.18

Figure 7.4: Accuracy

	SVD	Wavelet-feature extraction +SVD
Cinc_ECG_TORSO	0.24s	9.22s
InlineSkate	0.21s	2.22s

Table 7.1: Run time

Observation

- As evident from the run time results, applying the wavelet feature extraction as prior step to SVD does increase the run time but this increase is minimal since the run time only increases by few seconds.
- For the Cinc_ECG_TORSO data set, removing the detail coefficients of scale 1 i.e D_1 leads to a fractional improvement in the accuracy of the 1 nearest neighbour classifier. From this observation, we can deduce two facts: the information contained in the finer details of sequences fail to serve as a discriminator between inter class sequences and removing such information has the affect of decreasing the intra class variance as evident in the increase in accuracy.
- In the case of the InlineSkate data set, considering only the approximation part of signal i.e A_4 have no effect on the accuracy of the 1 nearest neighbour. This shows that when comparing sequences using the euclidean metric, the details in the first 3 levels: D_3, D_2 and D_1 play no role in the classification process.

  The sequence A_j corresponds to an approximation(smoothed version) of the signal at scale j. The coefficients of A_j are the amplitude values of the smoothed signal. These coefficients are considered as global trend

descriptors as they preserve information about the global trends of the sequence. However, such information can only be fully extracted by classifiers/clustering algorithms that employ metrics such as the DTW to warp the time axis. By warping the time axis, the DTW algorithm compares each point in one sequence with points at different temporal regions in the second sequence allowing it to distinguish inter class sequences based solely on their global shape [9, 30]. Metrics such as the euclidean will thus be inept in utilising the embedded information about the global shape as they are constrained to conducting only linear matches between temporal dimensions of sequences.

7.2 Curvature-based Feature extraction

In the previous section, we have observed that for datasets such as the InlineSkate dataset, the main commonality between intra class sequences is the global shape. Applying wavelet decomposition and considering only the approximation part of the signal A_4 although allows us to achieve dimensionality reduction without any decrement in accuracy but at the same time doesn't lead to any improvement in recall. The low accuracy achieved by the classifier is primarily due to the limitations incurred in comparing sequences using the euclidean metric. The individual amplitude values on their own are not reflective of the global shape of the sequence. This creates the motivation to explore feature extraction methodologies that capture information about the shape of the sequence.

To address this issue, I propose the following 4 step feature extraction methodology to extract useful features from the InlineSkate dataset:

i Apply wavelet decomposition up to scale 4 and consider only the approximation part of the signal A_4. This achieves dimensionality reduction by pruning away unwanted details.

ii Replace the amplitude values of the approximated signal with curvature values of the corresponding points(details to follow).

iii Apply fourier transform and consider only the first half of the fourier coefficients(details to follow).

iv Apply SVD on the result sequence of fourier feature descriptors to extract a smaller set of latent vectors that capture the full variance of the feature vectors in the fourier feature space.

In the following sections, I provide a detail description of the procedures specified in step (ii) and step (iii) of my proposed methodology.

Replacing the individual amplitude values with the curvature estimates of the corresponding points can allow the euclidean metric to take into account the **shape** of the signals at corresponding points when conducting a linear match of their temporal dimensions. The curvature of a geometric object is a quantitive description of the shape of that object[31]. It dictates the amount by which a geometric object deviates from being flat or straight. Formally, the curvature of a curve can be defined as follows:

Let γ be a differentiable curve on $\mathbf{E}^3$(Euclidean Space) such that

$$\gamma : (-\epsilon, \epsilon) \rightarrow U \subseteq \mathbf{E}^3$$

$$\gamma(t) = \begin{pmatrix} x(t) \\ y(t) \\ z(t) \end{pmatrix}$$

The derivative of this function $\gamma'(t) = \begin{pmatrix} x'(t) \\ y'(t) \\ z'(t) \end{pmatrix}$ is the velocity of this curve.

Thus by assuming that this curve is parameterised by arc length (i.e. the curve is parameterised to have unit velocity $\|\gamma'(t)\| = 1$) we define the *curvature* of as curve as

$$\kappa = \|\gamma''(t)\|$$

which in fact is none other than the magnitude of the acceleration of a curve.

Geometrically one can imagine κ in the case of curve as a parameter which value determines the amount of the bending of that the curve experiences in the flat euclidean space.

7.2.1 Extracting curvature based features

To extract features that capture the **shape** of the dominant trends in a sequence, I performed the following two step feature extraction process.

Step 1: Extraction of curvature estimates

To compute the curvature estimates at each point in a time series sequence, I have fitted a **smoothed cubic spline** to each sequence. The motivation behind the use of spline function is as follows: a spline is a polynomial function that is piecewise-defined, and possesses a high degree of smoothness at the places where the polynomial pieces connect [32]. A cubic spline is a specific type of spline which is constructed using piecewise polynomials of order 3. Fitting a cubic spline directly on the sequences will not be ideal as real datasets have

noise embedded in them. The fitted will therefore not be smooth. To overcome this issue, I have hence applied smoothed cubic spline to fit smooth curve to each sequence.

Mathematically, the smoothing spline estimate $\hat{g}(x)$ of the actual function $g(x)$ is defined to be the minimiser of :

$$\lambda \sum_{i=1}^{n} \{y_i - \hat{g}(x_i)\}^2 + (1-\lambda) \int_a^b \{\hat{g}''(t)\}^2 dt$$

where λ is a fixed constant

- As $\lambda \to 1$,the smoothing spline converges to the interpolating spline.
- As $\lambda \to 0$,the roughness penalty becomes paramount and the estimate converges to a linear least squares estimate.

The critical parameter here is λ. The values of λ determines the nature of the fit. For the sequences in the InLineSkate dataset, exact fitting is unnecessary as the sequences have noise embedded in them. Ideally, we want to choose a value of λ that allows the estimated curvatures to directly correlate with significant events/trends in the time series sequence.

For my analysis, I have chosen the value of 0.25. Through conducting experiments, I have observed that setting λ to this value allows the computed curvatures to reflect significant events in the time series sequence. Figure 7.5 shows the plots of an instance of class '7' from the InlineSkate dataset. The left hand plot corresponds to the raw signal while the righthand plot represents the curvatures computed at each of the time stamps. From the observation of the figure, it can be seen that the computed curvatures do reflect information about significant events in the time series sequence. At the global maximum, the curvature estimate of the corresponding point is close to 0 while the peak value on the curvature plot at t=1650 is in accordance with the sharp bend in the time series curve at the same time.

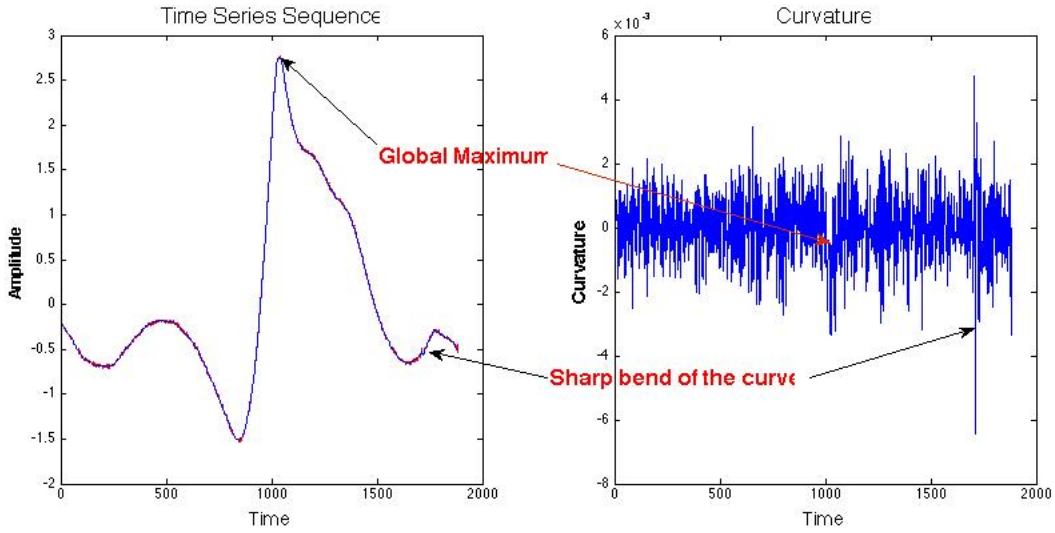


Figure 7.5: The left hand plot represents the time series sequence while the right plot corresponds to curvatures computed at each point in the time series sequence

The curvatures κ at each point of the fitted smooth curve is computed using the following expression:

$$\kappa = \frac{\hat{g}''(x)}{(1 + \hat{g}'(x)^2)^{\frac{3}{2}}}$$

Step 2: Extract features that reflect the global shape

The main objective here is to extract features that reflect the **global** shape of the time series sequence. The above mechanism embeds the information of local curvatures to each point of the time series sequence. But what we are really interested in is in the extraction of global shape descriptors. The shapes of the global trends of a signal are preserved in the low frequency bands while the higher frequency bands encode information of local shapes [33]. Thus, to extract global shape descriptors, I have applied the discrete fourier transform[34] on the extracted curvature sequences.

The Fourier transform maps time into frequency and phase. For each frequency the fourier transform yields an amplitude and a phase value. Thus, the transform captures the behaviour of the signal at different frequency band. Since the amplitudes represent curvature values estimated at step 1, the fourier transform therefore capture the **behaviour** of the shape of the signal at different **frequency** bands. The curvature-sequences can now be represented as the sum of sine and cosine waves whose phase and 'amplitude' are given by the fourier transform.

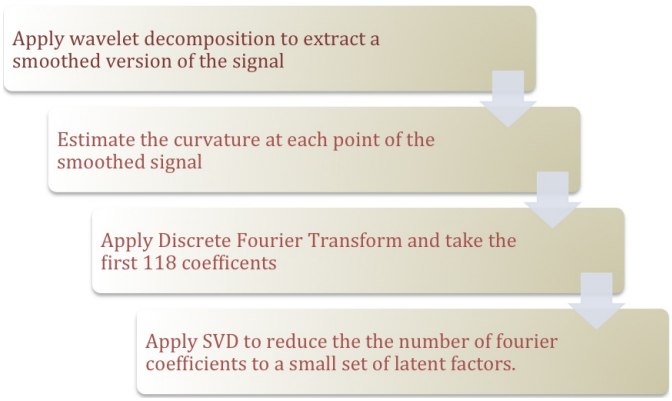
For any time series sequence, the coefficients of the discrete fourier transform

for each frequency band is computed as follows:

$$X_k = \sum_{i=1}^{n} x_n . e^{-2\pi i k \frac{n}{N}}$$

To extract global shape descriptors only the magnitudes of the fourier coefficients have been used and since global trends [26, 33] are preserved in the lower bands, I have only selected the first 118 coefficients as features for nearest neighbour classification. The set of features are reduced by SVD to a smaller latent feature that maps the data points in the 118 dimensional feature space to orthogonal plane that maximises the variance of the feature vectors.

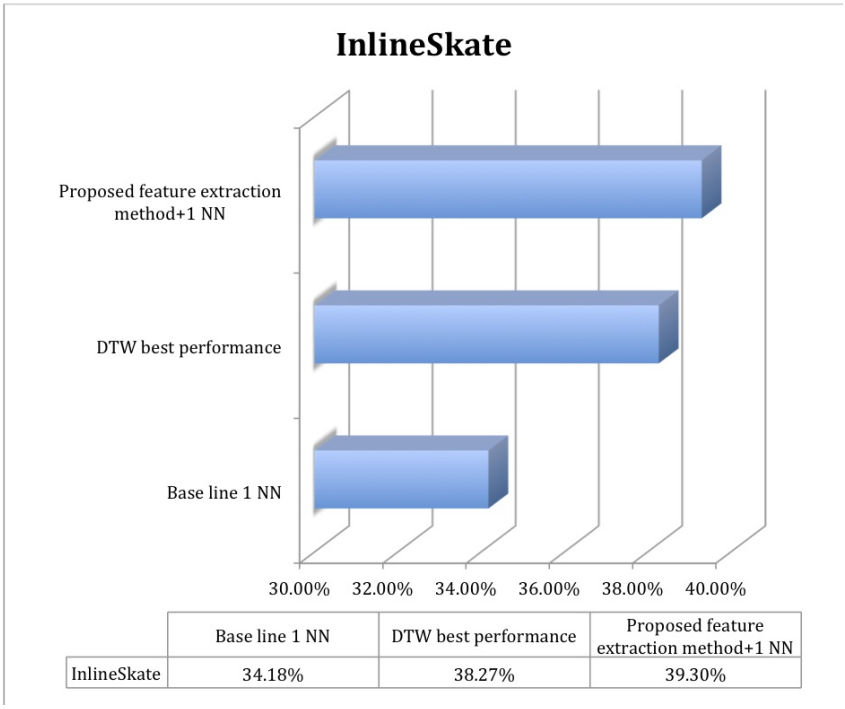
The entire feature extraction process can hence be summarised as follows:



Note: Dimensionality reduction occurs at step 1,3 and 4. Redundant features are pruned away at this steps to finally result a small set of latent factors that be used to cluster/classify inter class sequences that are distinguished by global trends.

7.2.2 Experimental results

To investigate whether using proposed feature extraction method improves the accuracy of the 1 nearest neighbour classifier, using the new extracted feature, I performed 1 nearest neighbour classification on the sequences on the InlineSkate test dataset. The accuracy results were compared with the accuracy of the base line 1 nearest classifier which employs no preprocessing technique and the highest recorded accuracy of the DTW algorithm[15] for this dataset. A summary of the results is as follows:



	Base line 1 NN	DTW best performance	Proposed feature extraction method+1 NN
InlineSkate	34.18%	38.27%	39.30%

Figure 7.6: Accuracy

Observation

- The 1 NN classifier achieves better accuracy when it uses the latent factors to conduct a linear match of the temporal dimensions for given any two sequences as opposed to using just amplitude values. From the table above, it can be seen that the classifier's accuracy has improved by 5.12%. This shows that the extracted global shape descriptors are better in clustering intra class sequences than the raw amplitude values.

- By employing the proposed feature extraction methodology, the 1 NN classifier using the euclidean metric achieves **better** accuracy than the version of the algorithm that employs the DTW algorithm, using the best warping window, to compute the similarity between the raw sequences.

  As we have discussed in the previous section, for time series sequences, the DTW algorithm makes a richer comparison than the euclidean metric. By warping the time axis, the DTW algorithm compares each point in one sequence with points at different temporal regions in the second sequence whereas the euclidean metric is only restricted to conducting linear matches between the temporal dimensions. The experimental results shows that by using the latent global shape descriptors, for the InlineSkate datasets, using the euclidean is equivalent to applying the DTW algorithm equipped with the best warping window on the raw sequences.

Bibliography

[1] R. Gautam Das, King ip Lin, Mannila, H., "Rule Discovery From Time Series," *4th Int'l Conference on Knowledge Discovery and Data*, 1998.

[2] U. Fayyad, C. Reina, and P. S. Bradley, "Initialization of Iterative Refinement Clustering Algorithms," 1998.

[3] A. Bazma, I. JONASSEN, I. EIDHAMMER, and D. GILBERT, "Approaches to the Automatic Discovery of Patterns in Biosequences," *Journal of Computational Biology*, vol. 5, pp. 279–305, Jan. 1998.

[4] A. S. Park and J. R. Glass, "Unsupervised Pattern Discovery in Speech," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, pp. 186–197, Jan. 2008.

[5] Y. Zhang and J. R. Glass, "Towards multi-speaker unsupervised speech pattern discovery," in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 4366–4369, IEEE, 2010.

[6] S. Salvador and P. Chan, "FastDTW : Toward Accurate Dynamic Time Warping in Linear Time and Space,"

[7] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," pp. 159–165, May 1990.

[8] L. Rabiner, A. Rosenberg, and S. Levinson, "Considerations in dynamic time warping algorithms for discrete word recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, pp. 575–582, Dec. 1978.

[9] Y. Xie and B. Wiltgen, "Adaptive Feature Based Dynamic Time Warping," vol. 10, no. 1, pp. 264–273, 2010.

[10] X. Xi, E. Keogh, C. Shelton, L. Wei, and C. A. Ratanamahatana, "Fast time series classification using numerosity reduction," in *Proceedings of the 23rd international conference on Machine learning - ICML '06*, (New York, New York, USA), pp. 1033–1040, ACM Press, June 2006.

[11] A. W.-C. Fu, E. Keogh, L. Y. H. Lau, C. A. Ratanamahatana, and R. C.-W. Wong, "Scaling and time warping in time series querying," *The VLDB Journal*, vol. 17, pp. 899–921, Mar. 2007.

[12] F. Korn, H. V. Jagadish, and C. Faloutsos, "Efficiently supporting ad hoc queries in large datasets of time sequences," in *Proceedings of the 1997 ACM SIGMOD international conference on Management of data - SIGMOD '97*, vol. 26, (New York, New York, USA), pp. 289–300, ACM Press, June 1997.

[13] F. Itakura, "Minimum prediction residual principle applied to speech recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 23, pp. 67–72, Feb. 1975.

[14] R. G. L. Doddington and George, "TIDIGITS Readme File," 1993.

[15] L. . R. Keogh, E., Zhu, Q., Hu, B., Hao. Y., Xi, X., Wei, "UCR Time Series Data sets," 2011.

[16] E. Keogh and C. A. Ratanamahatana, "Exact indexing of dynamic time warping," *Knowledge and Information Systems*, vol. 7, pp. 358–386, May 2004.

[17] M. Vlachos, G. Kollios, and D. Gunopulos, "Discovering similar multidimensional trajectories," in *Proceedings 18th International Conference on Data Engineering*, pp. 673–684, IEEE Comput. Soc, 2002.

[18] D. J. Marchette, "Local dimensionality reduction," vol. 14, ch. Dimensiona, p. 469, 1999.

[19] E. K. Chotirat Ann Ratanamahatana, "Three Myths about Dynamic Time Warping Data," *Mining, in the Proceedings of SIAM International Conference on Data Mining*, 2005.

[20] M. De Wachter, M. Matton, K. Demuynck, P. Wambacq, R. Cools, and D. Van Compernolle, "Template-Based Continuous Speech Recognition," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, pp. 1377–1390, May 2007.

[21] M. De Wachter, K. Demuynck, P. Wambacq, and D. Van Compernolle, "A locally weighted distance measure for example based speech recognition," in *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, pp. I–181–4, IEEE, 2004.

[22] M. A. Carlin, S. Thomas, A. Jansen, and H. Hermansky, "Rapid Evaluation of Speech Representations for Spoken Term Discovery,"

[23] E. Keogh and M. Pazzani, "Derivative dynamic time warping," *the 1st SIAM Int. Conf. on Data Mining (SDM- . . .*, 2001.

[24] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*, vol. 1. springer New York, 2006.

[25] L. Chun-Lin, "Chapter 1 Overview," ch. wavelets I, 2010.

[26] S. Mallet, *A Wavelet Tour of Signal Processing*. Academic Press ,London, 1998.

[27] H. Zhang, "Unsupervised Feature Extraction for Time Series Clustering Using Orthogonal Wavelet Transform.," *Informatica (03505596)*, vol. 30, no. 3, pp. 305–319, 2006.

[28] K.-P. Chan and A.-C. Fu, "Efficient time series matching by wavelets," in *Proceedings 15th International Conference on Data Engineering (Cat. No.99CB36337)*, pp. 126–133, IEEE, 1999.

[29] I. Daubechies, "Orthonormal bases of compactly supported wavelets," *Communications on Pure and Applied Mathematics*, vol. 41, pp. 909–996, Oct. 1988.

[30] J. Gorman, O. Mitchell, and F. Kuhl, "Partial shape recognition using dynamic programming," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, pp. 257–266, Mar. 1988.

[31] M. A. Haider, *Morphometric analysis of temporally varying shapes in 3 dimensions*. Bsc honors project, University of Edinburgh, 2012.

[32] C. D. Boor, *A practical guide to splines*. 1978.

[33] S. Osowski and D. D. Nghia, "Fourier and wavelet descriptors for shape recognition using neural networksa comparative study," *Pattern Recognition*, vol. 35, pp. 1949–1957, Sept. 2002.

[34] R. Bracewell, *The Fourier transform and its applications*. 1986.