How I Did It

Victor von Frankenstein

Master of Science

School of Informatics

University of Edinburgh

2013

Abstract

This doctoral thesis will present the results of my work into the reanimation of lifeless human tissues.

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.

(*Victor von Frankenstein*)

Table of Contents

Chapter 1

Introduction

jdcsdc

2

Chapter 2

Background

cdscsdcds

Chapter 3

Improving DTW

The Dynamic Time Warping(DTW) algorithm is the one of the oldest algorithms that is used to compare and cluster sequences varying in time, length and speed. Formally, given two temporal sequences, the algorithm utilises the technique of dynamic programming to find an optimum alignment between through the computation of local distances between the points in each sequence. The time and computational complexity of this algorithm is O(mn) where m and n denote the length of the sequences that are being compared. Thus for high dimensional time series sequences, the time and computational costs incurred by the algorithm are quite high which makes DTW a very unattractive choice for clustering or discovering motifs in high dimensional data sets. Intuitively speaking, DTW is a clustering algorithm that clusters similar patterns varying in time and speed. Another drawback for working in high-dimensional spaces is the contrast between the distances of nearest and furthest points. The distances between such points become increasingly smaller as the dimensionality increases. This makes it difficult to construct meaningful cluster groups in such spaces.

To address the issue of the curse of dimensionality, DTW algorithms employ a window constraint to reduce the search space. The window constraints determine the allowable shapes that a warping path can take. As the dimensionality of the data 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. For clustering data sets such as speech utterances, the effect produced by such global constraints is highly undesirable. If we consider two utterances of a word spoken at different time frames,

the patterns 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. Thus it is necessary to explore techniques other than window constraints that can improve the performance of the DTW algorithm in terms of both accuracy and time.

Before investigating methods to improve a technique, it is highly necessary to first understand the nature of the data itself. In this chapter, I investigate data-driven preprocessing techniques that attempt to understand the underlying intrinsic structure of the lower-dimensional space on which the data lives. By achieving a thorough understanding of the data,we can achieve dimensionality reduction by isolating and identifying smaller set of new(current) features that are more relevant for the problem in hand.

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. One of the most popular approaches to feature selection 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 maximize our natural pattern-recognition abilities, such as using multiple plots per page.

Feature extraction processes on the other hand are concerned with a 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 $\{x_n\}$ typi-

cally 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 variable. Hence 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 as shown by figure 1:

In the rest of this chapter, I explore a range of feature selection and extraction methods and investigate whether their application can improve the performance of the DTW algorithm in terms of both accuracy and time complexity.

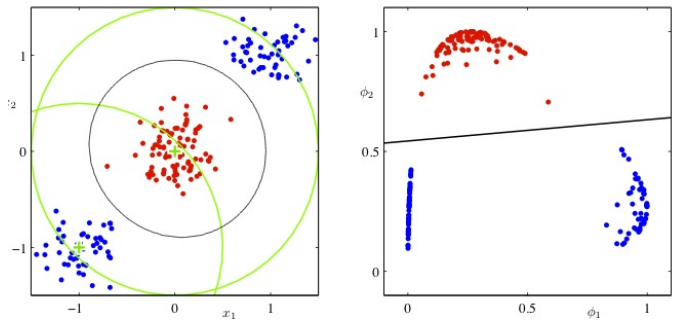


Figure 3.1: The figure on the right corresponds to location of the data points in the feature space spanned by gaussian basis functions $\phi_1(x)$ and $\phi_2(x)$

3.1 Feature Selection

The computational and time complexity associated with the DTW algorithm is governed by the dimensionality of the time series. To get a feel of the data, I employed exploratory data analysis on the isolated word utterances belonging to the test and training data sets that I constructed from the TIDIGITS corpus. The aim here to identify and isolate redundant features from the time series data. To get an idea about the structure of the data, I have studied the plots of the time series sequences along with performing auditory perception on the individual samples. Figure 2 gives 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. These durations of silence segments are considerably long compared to the interesting

regions in the acoustic signal that actually contain information about the spoken digit . Removing these silence segments not only reduce the dimensionality of the time series but also results in minimal loss of information.

- Through auditory perception of numerous samples, I have discovered that the recordings are highly distorted when played in matlab even when the data is scaled so that the sound is played as loud as possible without clipping. The distorted signal fails to provide any time auditory clue about category of the speaker i.e whether the speaker belongs to $\{$ boy,girl, men,women$\}$ and the signal must be played multiple times for its class to be correctly identified.

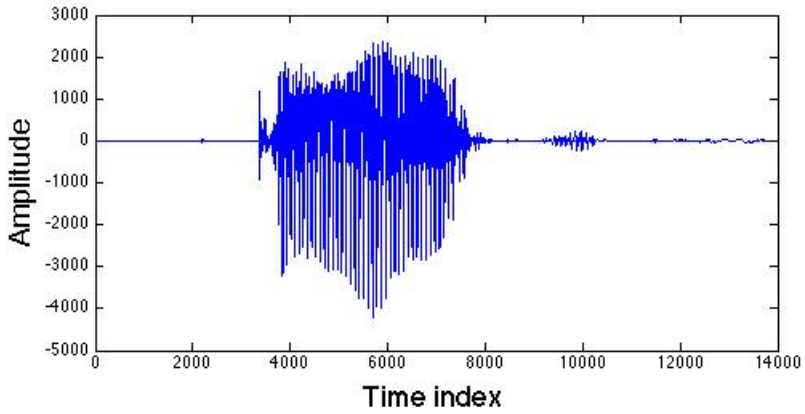


Figure 3.2: 'Raw 'signal

From further experiments, I have seen that if I down-sample the utterances by $\frac{1}{2}$ which in other words means decreasing the sampling frequency by half, the resultant sampled signal is much clearer to understand. Sub-sampling the utterances by half involves removing every other sample from the time series. From the observation of figure 3, it can be sen that this technique keeps the global trend of the signal intact but results in the loss of local information. Furthermore through auditory perception of the sampled signals, I have discovered that losing some **local information** actually cleans the signal in a manner that allows the listener to identity the speaker's category and the utterance's class with ease.

With this knowledge I have constructed the following algorithm: 'signal filter' that achieves feature selection by removing segments of silence and downsampling the remaining segments by half. An outline of the algorithm is as follows:

- The algorithm removes all samples in the times series sequence whose magnitude is less than the threshold. The threshold used is an adaptive parameter.

Algorithm 1 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: signalSil_R← removeSilence(rawSignal,threshold)
6: **return** output← downsample signalSil_R by $\frac{1}{2}$
7: **end procedure**

By using the information of the signal's maximum amplitude the algorithm sets the threshold accordingly. It raises the threshold for signals corresponding to speakers having a loud and deep voice and lowers the threshold for signals corresponding to speakers having gentle and low voice.

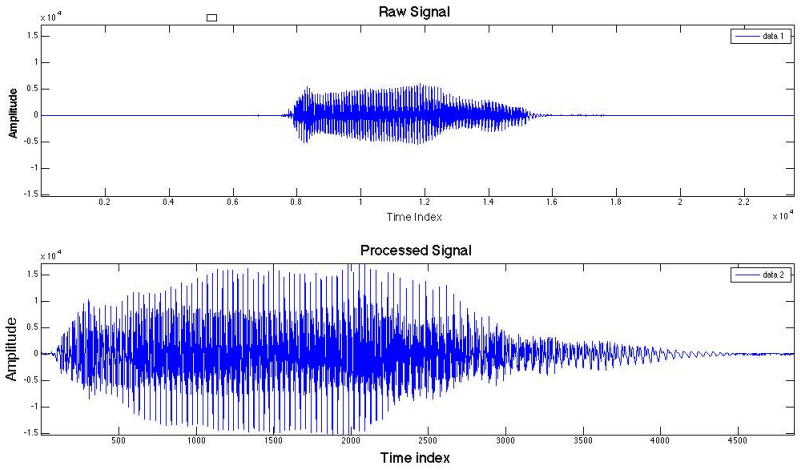


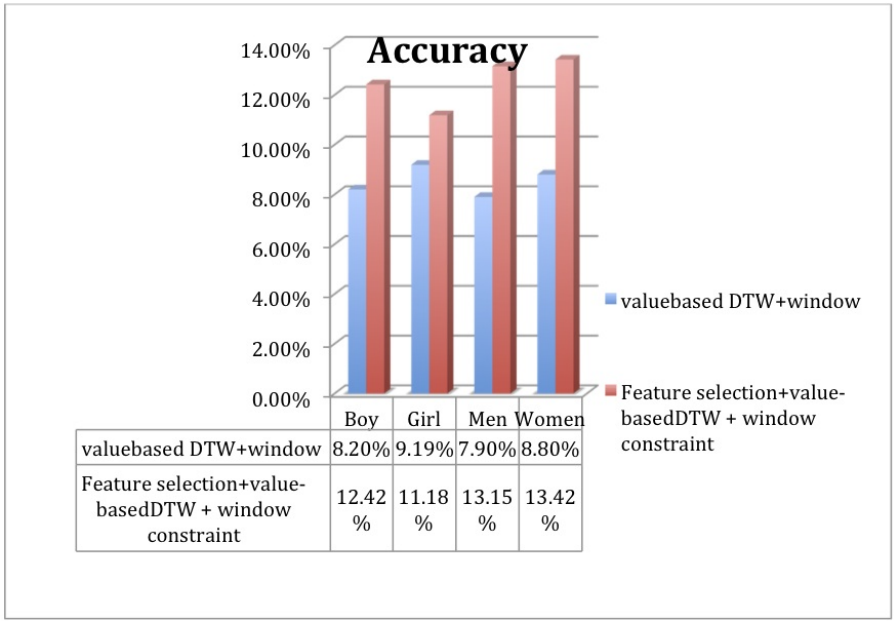
Figure 3.3: shows the raw acoustic signal corresponding to the utterances of the digit '5' alongside with the version that has its dimensionality reduced by the filter discussed above. From the comparison of the plots, it can be observed that the filter preserves the interesting patterns associated with the utterance while succeeding in reducing the dimensionality of the data.

To analyse how introducing this feature selection process effects the performance of a DTW algorithm in working with high dimensional data, I have run the baseline value-added DTW(i.e DTW using raw values) twice: once using the feature selection process as a preprocessing step and once without the feature selection process. An outline of the algorithm is given below.

Algorithm 2 Value-Based DTW

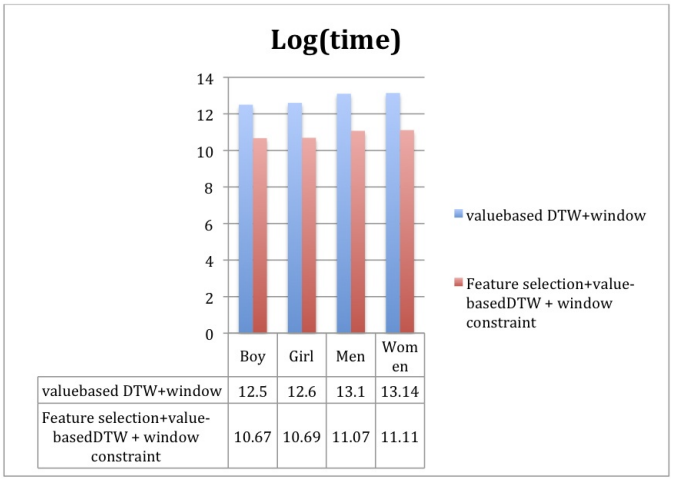
1: **procedure** VALUE-BASED($seq1, seq2$) ▷ two raw sequences
2: w = max($\lceil 0.1 * max(n.m) \rceil$, abs(n-m)) ▷ Window constraint
3: **for** i=1: to length(seq1) **do** ▷ Initialise the DTW cost matrix
4: DTW(i,0) = ∞
5: **end for**
6: **for** i=1 to length(seq2) **do**
7: DTW(0,i) = ∞
8: **end for**
9: **for** i=2 to length(seq1) **do**
10: **for** j=max(2, i-w) to min(length(seq2), i+w) **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 = $\frac{\text{DTW(n,m))}}{nm}$ ▷ n=length(seq1), m=length(seq2)
15: **end procedure**

The focus of my research here 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 the dimensionality of the time series sequences is still very high. Thus for these experiments I have employed the most rigid window constraint w = max($\lceil 0.1 * max(n.m) \rceil$, abs(n-m)) that keeps frames from one sequence from getting too far from the other. The results found from the experiments are as follows:



	Boy	Girl	Men	Women
valuebased DTW+window	8.20%	9.19%	7.90%	8.80%
Feature selection+value-basedDTW + window constraint	12.42 %	11.18 %	13.15 %	13.42 %

Figure 3.4



	Boy	Girl	Men	Women
valuebased DTW+window	12.5	12.6	13.1	13.14
Feature selection+value-basedDTW + window constraint	10.67	10.69	11.07	11.11

Figure 3.5

Observations:

- Employing the prior feature selection process allows DTW subjected to global window constraint to improve both its accuracy and time complexity.

  Explanations:

  - The DTW algorithm,due to the high dimensionality of the time series sequences is subjected to a window constraint that forces the algorithm to operate on the diagonal region of the DTW cost matrix. Removing these

redundant features increases the accuracy because these features primarily represent segments of silence and since all utterances contain silence segments, taking these silences into account degrades the performance as they bring i an unwanted notion of similarity in dissimilar patterns.

– The size of the DTW cost matrix is $O(mn)$. Achieving dimensionality reduction through feature selection reduces the size of the cost matrix and thus decreases the computational cost.

3.2 Feature extraction

To improve the performance of the DTW algorithm even further, in this section I investigate domain-independent and domain dependent feature extraction methodologies that employ an appropriate functional mapping to extract features that capture the intrinsic patterns of the data. The motivation behind this approach is to investigate to what degree we can improve the performance of the standard algorithm across different domains without making changes to the algorithm itself.

3.2.1 Domain-independent feature extraction

The fundamental problem of baseline (value-based) DTW is that the numerical value of a data point in a 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 know as *Derivative* DTW is proposed but it 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). Ideally we need to use features that contains information about the overall shapes of the sequences plus the local trend around the points. This allows the DTW to built a complete picture of the data point in relation to the rest of the sequence and hence achieve a better optimal alignment between the two sequences.

For feature extraction, the methodology that I have used for this setup is based on Xie and Wiltgen's paper[]. In their paper, the authors highlight a domain-independent feature extraction process where each point in the time series sequence is replaced by

a 4 dimensional vector. In this vector, the first two features correspond to information regarding the local trends around a point and the last two features reflect the position of that point in the global shape of the sequence. From the experiments conducted on the UCR data sets, it has been observed that embedding DTW with this feature extraction process yields greater accuracy across all datasets.

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

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

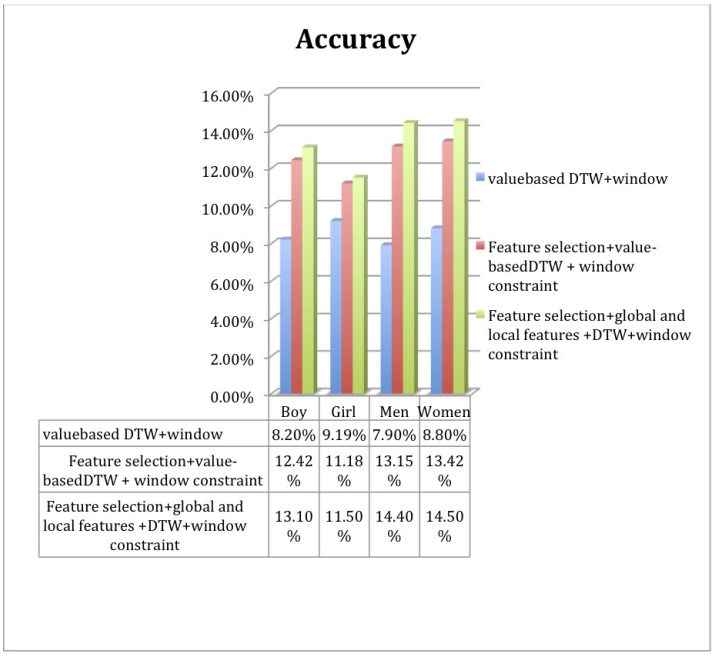
The extraction of global features is constrained by two factors: the features that reflect information about global trends and the features must be in the same scaling order as the local features. Being in the same scale allows them to be combined with local features. In [] the authors used the following method to extract global features from the time series sequence:

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

Note : The local and global features have no definition for the first and last points in a sequence.

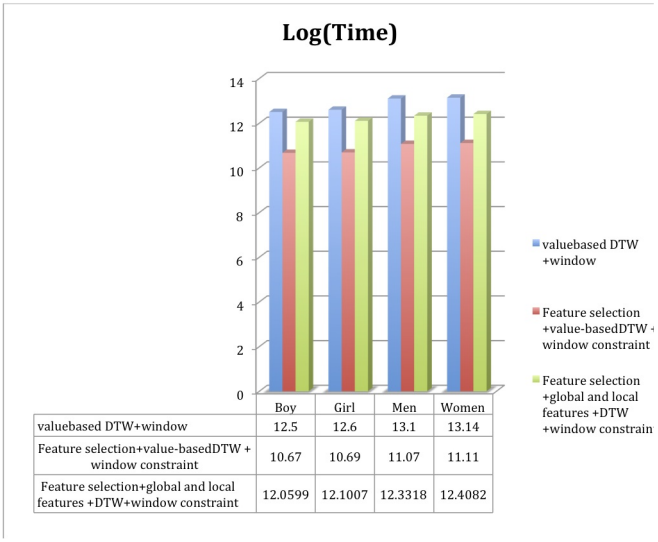
When working with high dimensional time series data, the main drawback of employing this feature extraction method is that it does not offer the advantage of dimensionality reduction. The dimensionality of the feature space is just two dimensions less than the dimensionality of the original data. The DTW algorithm combined with this feature extraction process therefore suffers from the curse of dimensionality as before. To tackle this issue, as a prior step to the feature extraction process, I applied the feature selection process that I have discussed in the previous section to remove redundant features.

The length of the resultant sequence of the vectors is still large. To address the issue of high computational cost, I have used the same window constraint applied the DTW algorithm equipped the two preprocessing stages of feature selection and extraction on the test data set that I had constructed from the TIGITS corpus. A summary of the results are given below:



	Boy	Girl	Men	Women
valuebased DTW+window	8.20%	9.19%	7.90%	8.80%
Feature selection+value-basedDTW + window constraint	12.42 %	11.18 %	13.15 %	13.42 %
Feature selection+global and local features +DTW+window constraint	13.10 %	11.50 %	14.40 %	14.50 %

Figure 3.6



	Boy	Girl	Men	Women
valuebased DTW+window	12.5	12.6	13.1	13.14
Feature selection+value-basedDTW + window constraint	10.67	10.69	11.07	11.11
Feature selection+global and local features +DTW+window constraint	12.0599	12.1007	12.3318	12.4082

Figure 3.7

Observation:

- Equipping DTW with a preprocessing stage that involves feature selection and extraction of local and global features result in a greater improvement in the accuracy of the algorithm.

  Explanation:

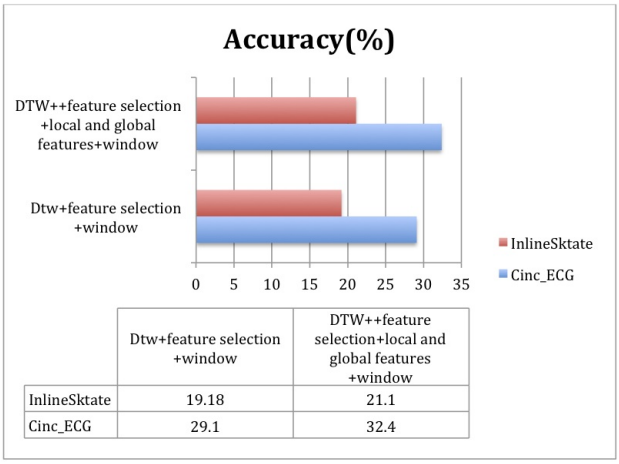
  - The algorithm now warps the time axis to match the local and global trends between points in the sequences instead of just their values. Since similar patterns will share the same trends, adding the feature extraction phase therefore improves the accuracy.

- The computational cost incurred by the algorithm is higher than the previous version that uses only the feature selection process.

  Explanation:

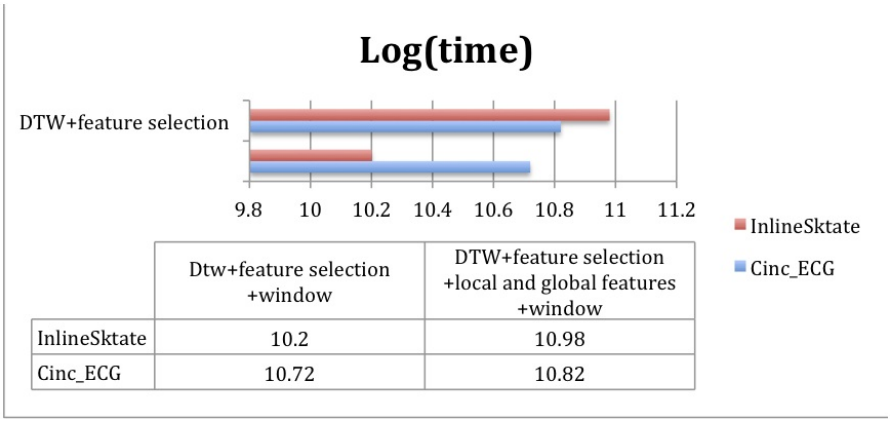
  - The cost of applying the euclidean metric on vectors >cost of applying the euclidean metric on points. Since the euclidean metric is applied *mn* times. The overall computational cost increases.

The problem of working with a single data set is that if the model design is iterated many times using a limited size data set, then some over-fitting to the validation data can occur. To ensure that our model has not over fitted to the test set, I ran the two versions of DTW :one that uses the preprocessing steps of feature selection and feature extraction while being subjected to a window constraint while the other omits the feature extraction phase and uses the feature selection phase along with the window constraint, on the InlineSkate and Cinc_Ecg_Torso time series datasets[]. The results found are as follows:



	Dtw+feature selection +window	DTW++feature selection+local and global features +window
InlineSktate	19.18	21.1
Cinc_ECG	29.1	32.4

Figure 3.8



	Dtw+feature selection +window	DTW+feature selection +local and global features +window
InlineSktate	10.2	10.98
Cinc_ECG	10.72	10.82

Figure 3.9

The differences between the two versions of DTW in terms of accuracy and time complexity are consistent with the observations made in the previous experiment with the TIDIGITS data set. This gives valid supports to the analysis of the tests that I have conducted so far

3.2.2 Domain-dependent feature extraction

The main data set that I am working with is the TIGITS corpus which is composed of speech utterances. DTW formally is a domain independent algorithm. It will interesting to investigate to what degree is the performance of the algorithm effected if we model the information of the domain into the DTW algorithm. Since at the moment, i am investigating techniques that improve the performance of the DTW algorithm on high dimensional data without making changes to the algorithm itself, in this section, I investigate domain-dependent feature extraction methods that embed the knowledge of the domain in the feature extraction phase.

For speech, the most commonly used features are the MFCC features-mel cepstrum ceptral 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.

A sketch of the MFCC feature extraction is given below:

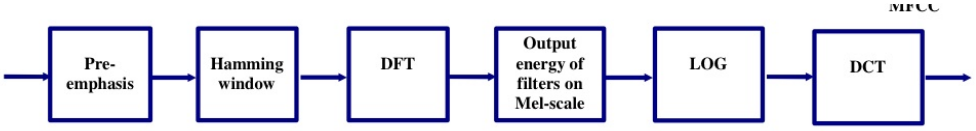


Figure 3.10: MFCC feature extraction

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

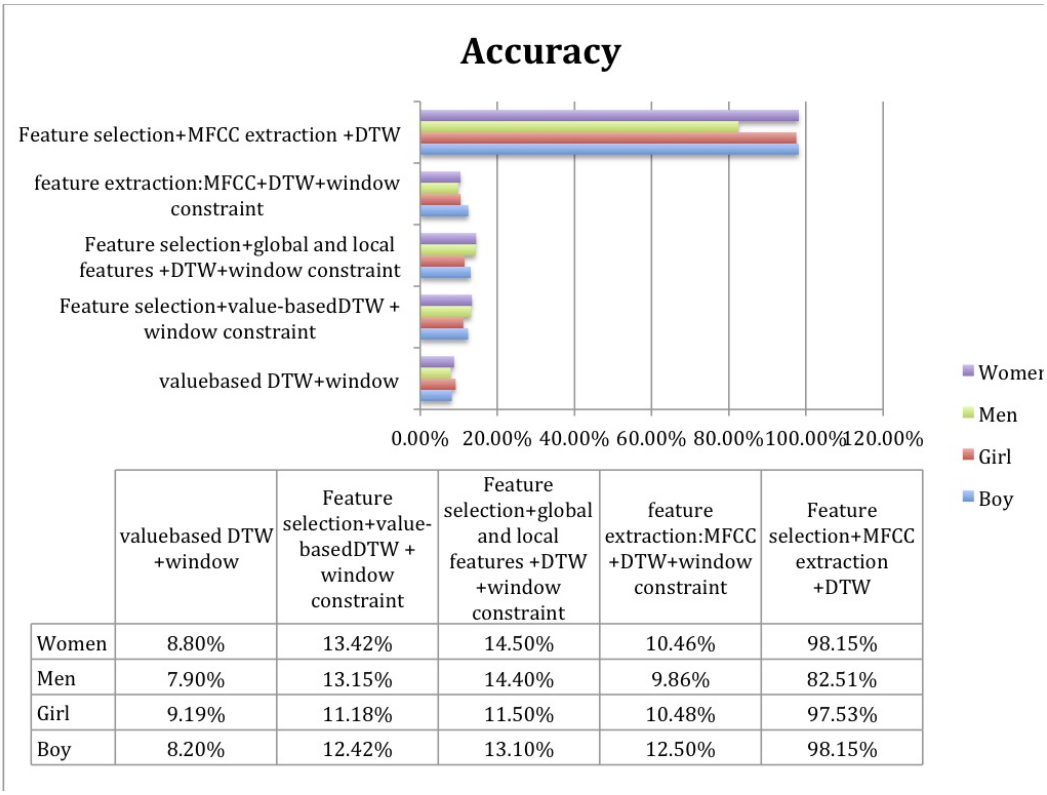
v Log : makes the feature less sensitive to variations in input such as power variations on the speakers mouth.

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

Through the windowing process, the MFCC features extraction achieves dimensionality reduction. Each sequence is segmented into frames of length 20 to 30 ms which are then through appropriate functional mapping are converted into sequences of MFCC feature vectors. Since the result sequence of vectors is much smaller than the length of the original sequence, the size of the DTW cost matrix is much smaller than before, This in turn lowers the time and computation cost incurred by the algorithm. One of the focal points of research for this project is to investigate alternative measures to using global window constraints that minimises the time and a computational cost incurred by the DTW to minimum without decreasing accuracy when working in high dimensional spaces. So the question that now lies is whether we can achieve greater reduction in dimensionality. The feature selection procedure that I discussed in the previous section reduces the size of the sequences by removing segments of silence followed the renaming segments by 1/2. As a prior step to MFCC feature extraction, if we use the first half of this feature selection process(i.e silence removal) and then apply MFCC feature extraction, we achieve a dimensionality reduction which is greater than

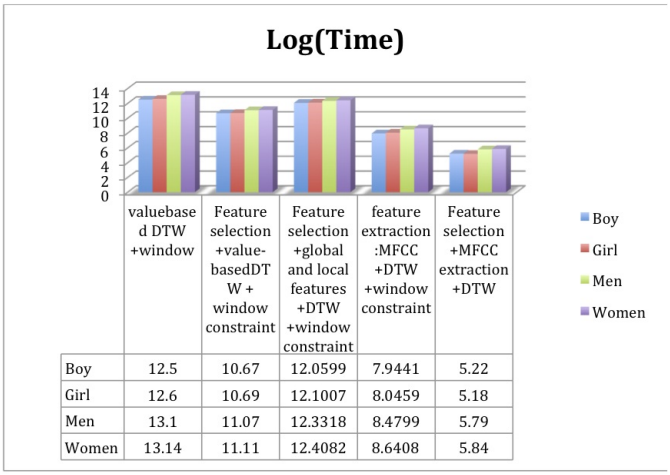
using either of the processes alone.

To test this idea I ran two versions of DTW on the TIGITS test set. The first version is equipped with a two step preprocessing stage: removal of silence followed by MFCC feature extraction while the 2nd version only employs MFCC feature extraction as pre-processing step. For this experiment, I had to impose a window constraint on the later version of DTW to contain the time and computational cost. Although MFCC feature extraction does achieve dimensional reduction. From doing initial tests, I have observed that the time taken by the algorithm to compare any two sequences is still significantly high. To therefore reduce the computational time, I was forced to use the same window constraint that I employed for the previous experiments. A summary of the results of all experiments is given by the figures below:



	valuebased DTW +window	Feature selection+value-basedDTW + window constraint	Feature selection+global and local features +DTW +window constraint	feature extraction:MFCC +DTW+window constraint	Feature selection+MFCC extraction +DTW
Women	8.80%	13.42%	14.50%	10.46%	98.15%
Men	7.90%	13.15%	14.40%	9.86%	82.51%
Girl	9.19%	11.18%	11.50%	10.48%	97.53%
Boy	8.20%	12.42%	13.10%	12.50%	98.15%

Figure 3.11: MFCC feature extraction



	valuebased DTW +window	Feature selection +value-basedDTW + window constraint	Feature selection +global and local features +DTW +window constraint	feature extraction :MFCC +DTW +window constraint	Feature selection +MFCC extraction +DTW
Boy	12.5	10.67	12.0599	7.9441	5.22
Girl	12.6	10.69	12.1007	8.0459	5.18
Men	13.1	11.07	12.3318	8.4799	5.79
Women	13.14	11.11	12.4082	8.6408	5.84

Figure 3.12: MFCC feature extraction

Observations:

- It can be observed that adapting DTW to incorporate a pre-processing stage that involves removal of redundant features through silence removal and employing a feature extraction mapping that integrates domains knowledge in the feature extraction process achieves almost perfects accuracy and incurs the lowest computational time. To be precise, adapting DTW to be domain-dependent allows the algorithm to achieve near perfect accuracy while incurring the minimum computational cost.
- From analysing the time complexity associated with each of the two versions, it can be seen that partitioning the sequences into frames actually leads to greater reduction in dimensionality of the time series than removing redundant features.

To summarise, from the observation of the results gathered from the experiments it can be concluded that employing preprocessing steps that include the domain knowledge greatly improves the accuracy and time complexity of the DTW algorithm when high dimensional spaces and in some domain such the TIGITS corpus eliminates the necessity of impose global window constraints to achieve minimum computational cost.

Chapter 4

Adaptive DTW

n the previous chapter, I have investigated techniques on different pre-processing strategies that can improve the performance of the general DTW algorithm in working with high dimensional time series datasets. These techniques that I am investigated so far primarily focus on the improving the quality of the data rather than the algorithm itself. The results found from the experiments so far, have shown that understanding the intrinsic properties of the data and factoring in the domain information can not only improve the performance of the DTW but may even discard the need for a the global windowing constraint(as we have seen for the model that augments feature selection and MFCC feature extraction) to reduce the time and computational complexity. The DTW algorithm is a memory based algorithm that employs a similarity metric to compare a sequence with all sequences in the training in an iterative manner. Since the whole training set is used during the testing phase, the computational complexity makes the algorithm very attractive to use. The computational cost of a DTW algorithm is (mn) where m and n denote the length of the two time series sequences currently being compared. Using longer sequences increases the size of the DTW cost matrix hence resulting into a greater number of computations. The domain-dependent pre-processing methodology doesnt guarantee the dropping of the global window constraint that is used to reduce the search space when tackling high dimensional data. As we have seen from the experiments,when working with high-dimensional time-series data, the accuracy of the DTW algorithm using a window constraint suffers greatly even if it's equipped with domain dependent/independent features. Hence from a scientific point of view, it is of great interest to research methods to improve the DTW algorithm so it can constraint the time and computational complexity associated with

high-dimensional data without degrading the accuracy by too much. In this chapter, I investigate an unsupervised methodology that:

- incorporates information about local and global trends in the feature extraction process
- employs an adaptive DTW that tackles the issue of the large time and computational complexity by moving from working on time series sequences to sequences of segmented time-slices. To counter the tradeoff in the decrease in accuracy, the algorithm is equipped with a kernel function(self-proposed) that is designed to measure the similarity of sub-sequences more accurately than standard euclidean metric by being invariant toward time-dilation and scale.

4.1 Feature extraction

For feature extraction, the methodology that I have used for this setup is based on Xie and Wiltgen's paper[] that I have already discussed in the previous section. Each point in the time series sequence is replaced by a 4 dimensional vector where the first two features correspond to information regarding the local trends around a point and the last two features reflect the position of that point in the global shape of the sequence.

Definition of local feature as given in [3.2.1] :

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

Definition of global feature:

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

4.2 Adaption of DTW

The feature extraction methodology discussed above maps the time series sequence to a time series sequence of vectors whose length is $\|X_n\| - 2$. (where $\|X_n\|$ denotes the length of the original time series sequence). The DTW augmented with these features will still suffer from large time and computational complexity if the dimensionality of

the data is high. In the MFCC feature extraction process, the time series sequence is first segmented into series of frames of length 20ms. Through appropriate functional mapping, each frame is then mapped to a vector. Because the length of the resultant sequence of vectors is much smaller than the length of the original time series, the size of the DTW cost matrix is reduced resulting in lower time and computational cost associated with each comparison.

Similar to the MFCC extraction process, the time series of 4d vectors extracted in the feature extraction stage are segmented using windows of width 5ms. The original time series is reduced to series of matrices where the length of the new series is 5 times smaller than before. Now if we adapt the cost function of DTW to work on series of matrices rather than series of vectors we can achieve a large improvement in the time and computational cost associated in the testing phase without imposing a **window** constraint.

The problem now can be shifted to finding an appropriate kernel that can be used to compute the similarity between matrices composed of feature vectors. Ideally, we want a metric that takes into account the variation of speed and time when comparing two similar subsequences. We will want to compare the global and local properties associated with a point in one subsequence with the global and local properties of points at different regions in the second sub-sequence illustrated by figure 2. 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 will be matched based on a linear match of the two temporal dimensions. In our context, we need a kernel that computes the similarity between two sub-sequences by warping the time axis.

The motivation behind the kernel that I propose for aiding DTW to tackle high-dimensional data comes from the polynomial kernel.
Let x and z be two dimensional vectors. 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 ϕ that contains terms of order 2. Now, if we generalise this notion then $k(x,z) = (x^T z)^M$ contains 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:.

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

where n denotes the length of the window and x_i and z_j represents the 4-dimensional features indexed by the points in two sub-sequences.
To motivate the reasoning behind the construction of this particular kernel lets consider the following signals:

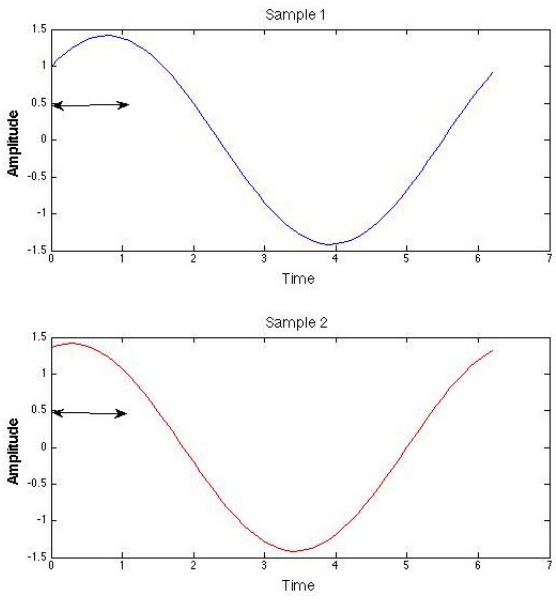


Figure 4.1: Two signals separated by translation

The signal denoted by the 'red' color is a 'slower' version of the signal denoted by the 'blue' color . In the above example, if we are comparing the similarity between the time slices spanned by the arrows, an ideal kernel must be invariant to the time offsets of the signals and thus should consider all possible pairings between the vectors in the

subsequences. Intuitively speaking, the kernel must behave like a DTW algorithm..
For time slices of width n, the kernel metric can be expanded and expressed as :

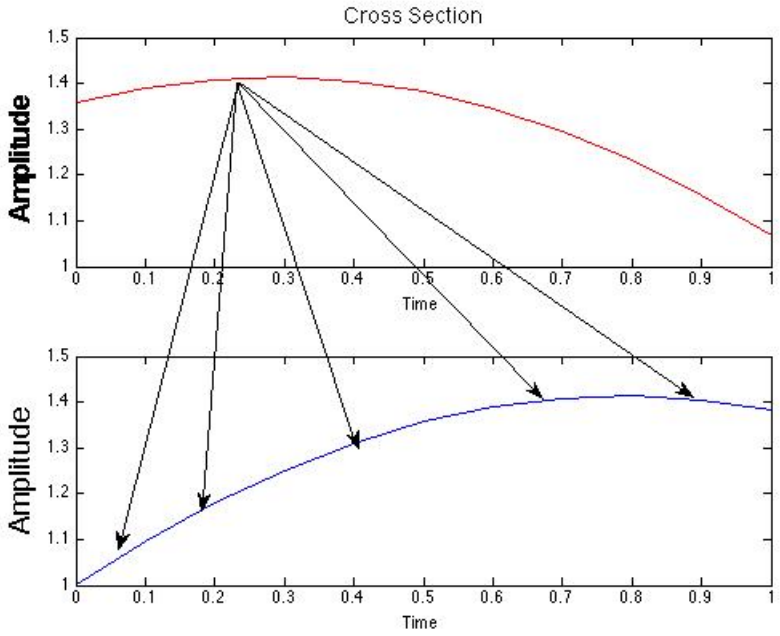


Figure 4.2: Two identical subsequences varying in time

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

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 $\{(x_i z_i)|x_i \in \text{seq1}, z_i \in \text{seq2}\}$. Similar to the polynomial kernel, the proposed kernel allows us to match all possible pairs of vectors belonging to the two sub-sequences given by the matrices. It is easy to check that this proposed kernel is in fact a valid kernel:

- K(x,z)= K(z,x) $\Rightarrow$the function is symmetric.
- The kernel satisfies Mercer's theorem : 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 high-dimensional time sequences with using a window constraint. However the accuracy and computa-

tional cost of the DTW is now dependent on the size of the time slices used to segment the original sequences:

- The accuracy of DTW increases as the width of the windows decrease. Using subsequence allows the similarity measure to be dominated by the dot products of points whose local and global features are most alike. However using smaller windows achieve lesser dimensionality reduction. Thus the time and computational complexity suffers.

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 ideal cost function that make use of dot product sis the *arc-cosine*. Hence I embedded the kernel function in the cosine distance:

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

where $X = \sum_{i=1}^{n} x_i$ and $Z = \sum_{j=1}^{n} z_i$

A formal outline of the algorithm is as follows:

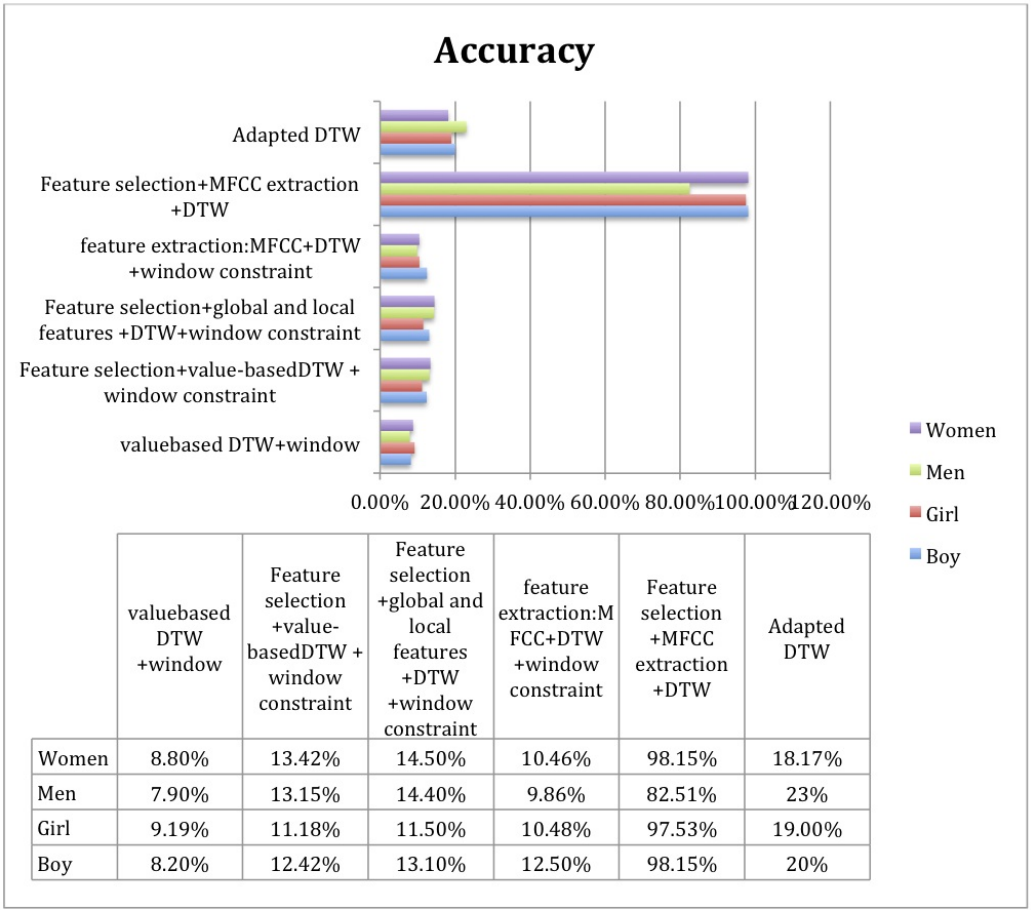
Algorithm 3 Adpated DTW

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

4.3 Experimental results

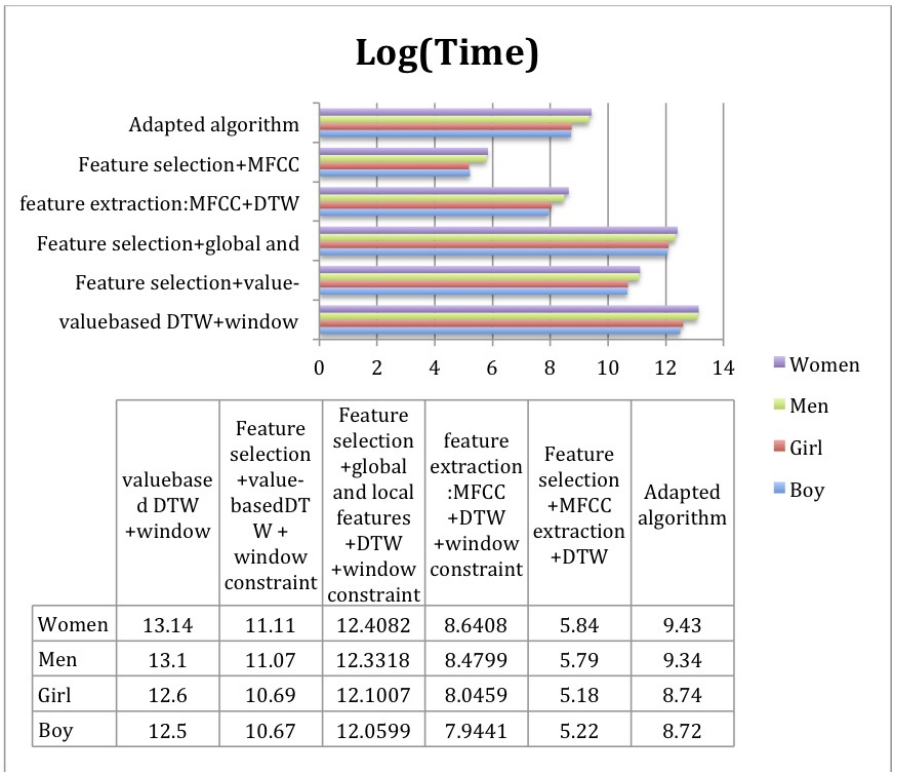
The changes that I have introduced, in the previous section to the 'Dynamic Time Warping' algorithm is aimed to improve the algorithm's performance in handling high dimensional time series data. In this section, I investigate the investigate the performance of my proposed algorithm by running it on the test data set that I have constructed from TIDIGITS test corpus and the time complexities and accuracies against the methodologies that I have investigated so far:

- Approach 1
  - Apply feature selection process to remove segments of silence and down sample the remaining segment to improve the quality.
  - Apply value-based DTW(which we denote as baseline) using the most constrained window size and a euclidean metric.
- Approach 2
  - Apply no feature selection process
  - Perform feature extraction by extracting MFCC features
  - Apply DTW using the most constrained window size and a euclidean metric.
- Approach 3
  - Apply feature selection process that only removes segments of silence
  - Extract MFCC features
  - Apply DTW augmented with the euclidean metric.
- Approach 4
  - Apply feature selection process to remove segment of utterance and down sample the remaining segment to improve the quality.
  - Apply the feature extraction process discussed in [3.2.1] to extract local and global features.
  - Apply DTW using the most constrained window size and a euclidean metric.



	valuebased DTW +window	Feature selection +value-basedDTW + window constraint	Feature selection +global and local features +DTW +window constraint	feature extraction:MFCC+DTW +window constraint	Feature selection +MFCC extraction +DTW	Adapted DTW
Women	8.80%	13.42%	14.50%	10.46%	98.15%	18.17%
Men	7.90%	13.15%	14.40%	9.86%	82.51%	23%
Girl	9.19%	11.18%	11.50%	10.48%	97.53%	19.00%
Boy	8.20%	12.42%	13.10%	12.50%	98.15%	20%

Figure 4.3: Accuracy



	valuebased DTW +window	Feature selection +value-basedDTW + window constraint	Feature selection +global and local features +DTW +window constraint	feature extraction :MFCC +DTW +window constraint	Feature selection +MFCC extraction +DTW	Adapted algorithm
Women	13.14	11.11	12.4082	8.6408	5.84	9.43
Men	13.1	11.07	12.3318	8.4799	5.79	9.34
Girl	12.6	10.69	12.1007	8.0459	5.18	8.74
Boy	12.5	10.67	12.0599	7.9441	5.22	8.72

Figure 4.4: Time complexity

There are quite number of interesting observations that can made from the graphs and the tables given by figures [4.3,4.4].

- The proposed algorithm achieves better accuracy for test samples belonging to all categories than any of algorithms that employ the rigid window constraint of $w = max(\lceil 0.1 * max(n.m) \rceil, abs(n-m))$. The most interesting result is that the new algorithm incurs a lower computational cost than any of these DTW algorithms.

- From examining the improvements in accuracy across all categories, it can be concluded that for the TIGITS dataset, the new algorithm is invariant to variations introduced in the acoustic signals by different speakers .

- The methodology behind my proposed approach and that of approach 4 includes the same pre-processing stage. Th input to the DTW algorithm is constructed by using the domain dependent feature selection process mentioned in section followed by a domain independent feature extraction mapping (i.e the local and global features defined in section). Both approaches differ however, in their use of a cost function and a window constraint. The proposed approach doesn't subject DTW to any window constraints and utilised the kernel function that I constructed in the cost metric. Approach 4 on the other hand, employs the euclidean metric an subjects DTW to the window constraint of of $w = \max(|n-m|, 0.1 * \max(n,m))$.

  The proposed DTW segments sequences into frames and employ a cost function on the frames. This reduces the dimensionality of the time series sequences and allows the algorithm to achieve a time and computational cost than is lower than the cost incurred by any of the investigated adaptations of the DTW algorithm that employs window constraints.

To confirm that the performance of the new algorithm is not tailored for this particular time-series data set, I have applied the tested the algorithm on the two largest time series data sets in UCR database.

The description of the data sets used for the next set of experiements are as follows:

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

   Unlike the speech utterances, the time series sequences within each data set share the same length.

4.3.1 Experimental setup

The focus now is to check how well/ bad is the performance of the new DTW algorithm against DTW algorithms using window constraints when applied to datasets that belong to other application domains . The domain -dependent pre-processing policies are dropped for this set of experiments as these procedures were specifically designed for speech data. Thus in this section, I compare my proposed approach against:

- Approach 1
- Approach 4

As a I mentioned, the domain-dependent feature selection process of silence removal and subsampling is dropped from all approaches. However, the feature extraction process that involves extracting local and global features is kept since this procedure is domain independent.

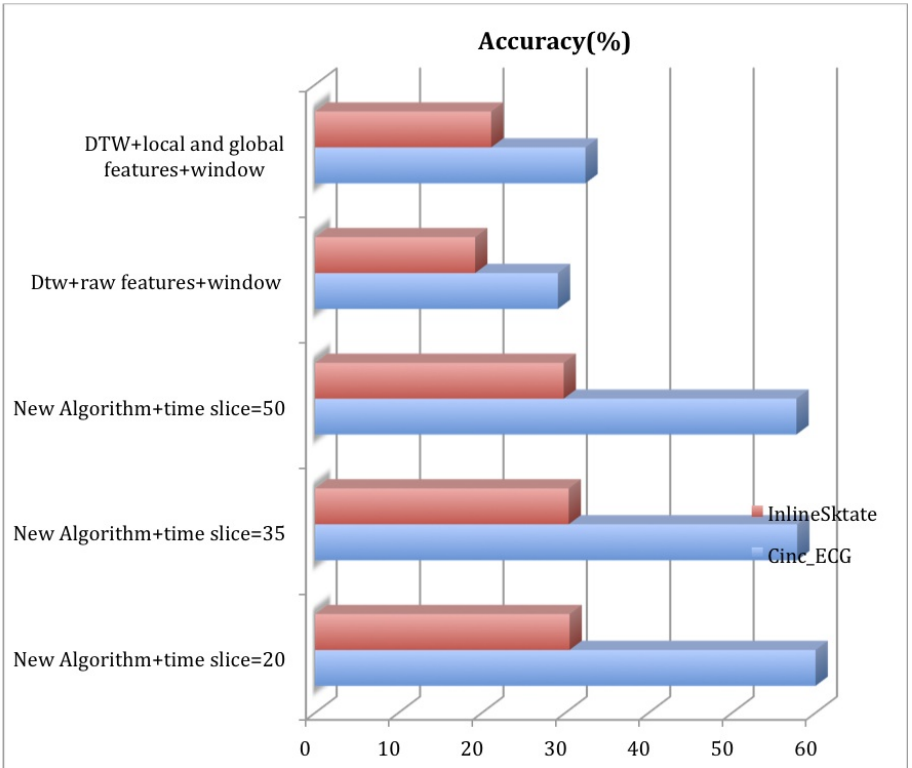
One of the factors that I have also investigated in these experiments is the size of the time slices used in the algorithm that I proposed. For the TIDIGITS data set, I have used window slices of 50 ms. Reducing the size of the windows should principally increase both accuracy and time-complexity . The 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 . Having a smaller window allows the sum to be dominated by dot products of vectors that are most similar. However smaller window frames results in longer time series sequence of frames. This in turn increases the time and computational complexity incurred by the DTW algorithm.

Figure [4.5] and figure[4.6] shows the accuracies and log(time) of the algorithms for the two datasets:

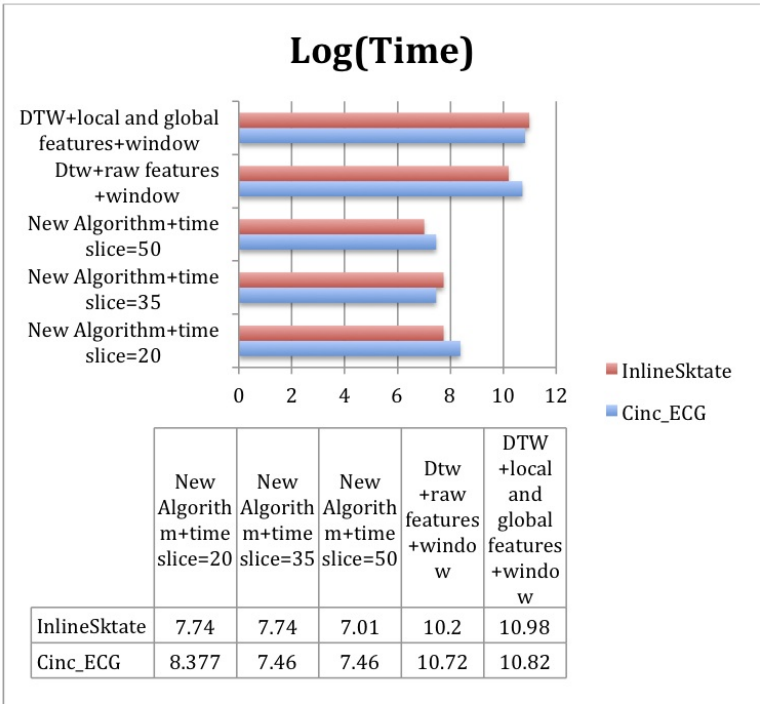


	New Algorithm+time slice=20	New Algorithm+time slice=35	New Algorithm+time slice=50	Dtw +raw features +window	DTW +local and global features +window
InlineSktate	30.5	30.4	29.8	19.18	21.1
Cinc_ECG	60	57.8	57.7	29.1	32.4

Figure 4.5: Accuracy

Observation:

- The new algorithm indeed achieves better accuracy than any versions of the window constrained DTW algorithm employing domain-independent feature extraction
- Comparatively, the performance of the new algorithm does improve if smaller window slices are employed to partition the time series sequence of vectors.



	New Algorithm+time slice=20	New Algorithm+time slice=35	New Algorithm+time slice=50	Dtw +raw features +window	DTW +local and global features +window
InlineSktate	7.74	7.74	7.01	10.2	10.98
Cinc_ECG	8.377	7.46	7.46	10.72	10.82

Figure 4.6: Time complexity

Observation:

- The new algorithm incurs less time and computational cost than any versions of the window constrained DTW algorithm employing domain-independent feature extraction
- Comparatively, the time complexity of the new algorithm increases when smaller window slices are used to partition the sequence