BRIEF ARTICLE

THE AUTHOR

1. Report

In the previous chapter, we have seen that using a MFCC representation of utterances with regions of silence removed leads to a large improvement in accuracy, time and computational complexity in the performance of DTW algorithm augmented with a euclidean metric. But this model cannot to extended to data sets that are not speech/ From the experiments conducted it is clear that the main contributing factor behind the large time and computational complexity experienced by the base line DTW is the dimensionality of the time series sequences. 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 base-line DTW algorithm is a domain independent algorithm that uses a similarity metric to compare any two sequences through comparison of their global trends. The algorithm employes dynamic programming to search a space of mapping between the time axis of the two respective sequences to determine the optimum alignment between them. The only difference between MFCC-augmented DTW and baseline DTW is the feature extraction stage. In machine learning, feature extraction refers to the pre-processing stage that involves the extraction of new features from a set of raw attributes through a suitable functional mapping. The extraction phase of MFCC features involves a segmentation of the time series followed by a functional mapping on the segmented windows. The resultant sequence of extracted feature vectors has a much smaller length compared to the length of the original sequence. Evident from the experiments done in the previous chapters, the use mel-cepstrum features extracted on 'cleaned' signals not only increases the accuracy of DTW but also reduces the time and computational cost through reduction of dimensionality of the original sequence.

Time series sequences are embedded with local and global trends. Unsupervised methods using DTW focus on exploiting the information of either these trends for pattern

extraction and mining. From the work conducted by Xie and Wiltgen on the time series datasets[], it has been shown that using DTW equipped with features that incorporate information of **both** local trends and global shapes in the clustering /classification process greatly improves the performance of DTW . Unlike the MFCC, the features constructed from local and global trends are domain independent and thus can be applied to any time of data. However, when working 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.

2. Feature extraction

The fundamental problem of baseline 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 the fundamental problem with this DTW is that it fails to detect significant 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[]. 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 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.

3. 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:

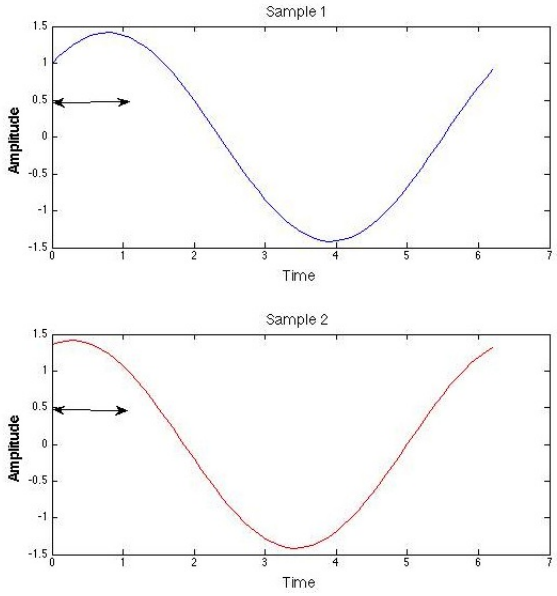


FIGURE 1. Two signals separated by translation

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 :

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

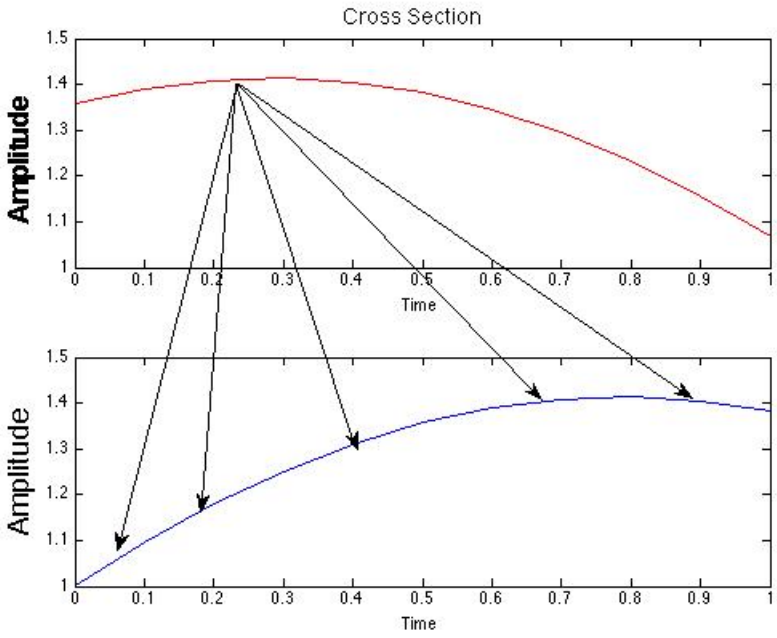


FIGURE 2. Two identical subsequences varying in time

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) ⇒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 computational 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:

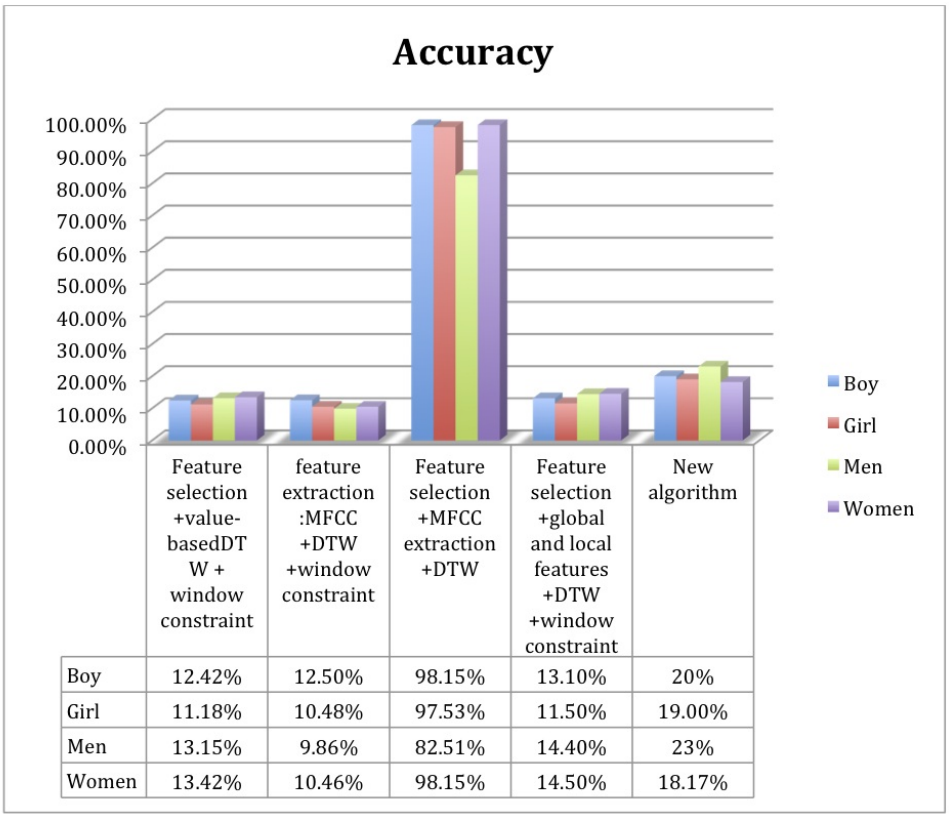
Algorithm 1 Adpated DTW

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(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**
            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. Experimental results

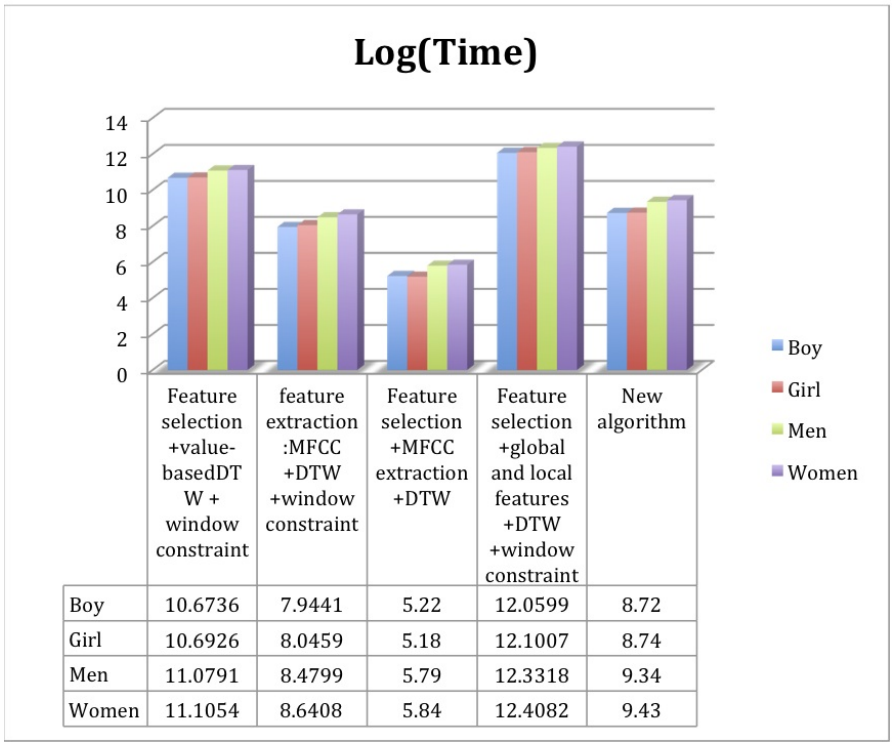
The changes that I have introduced, in the previous section to the 'Dynamic Time Warping' algorithm is aimed to improve the algorithm's performance in handling high dimensional time series data. In this section, I investigate the investigate the performance of my proposed algorithm by running it on the 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 [] to extract local and global features.
  - Apply DTW using the most constrained window size and a euclidean metric.



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

FIGURE 3. Accuracy



	Feature selection +value-basedDTW + window constraint	feature extraction :MFCC +DTW +window constraint	Feature selection +MFCC extraction +DTW	Feature selection +global and local features +DTW +window constraint	New algorithm
Boy	10.6736	7.9441	5.22	12.0599	8.72
Girl	10.6926	8.0459	5.18	12.1007	8.74
Men	11.0791	8.4799	5.79	12.3318	9.34
Women	11.1054	8.6408	5.84	12.4082	9.43

FIGURE 4. Time complexity

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

The proposed algorithm achieves better accuracy for test samples belonging to all categories than any of algorithms that employ the rigid window constraint of of $w = \max(|n - m|, 0.1 * \max(n, m))$. 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.

4.1. **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

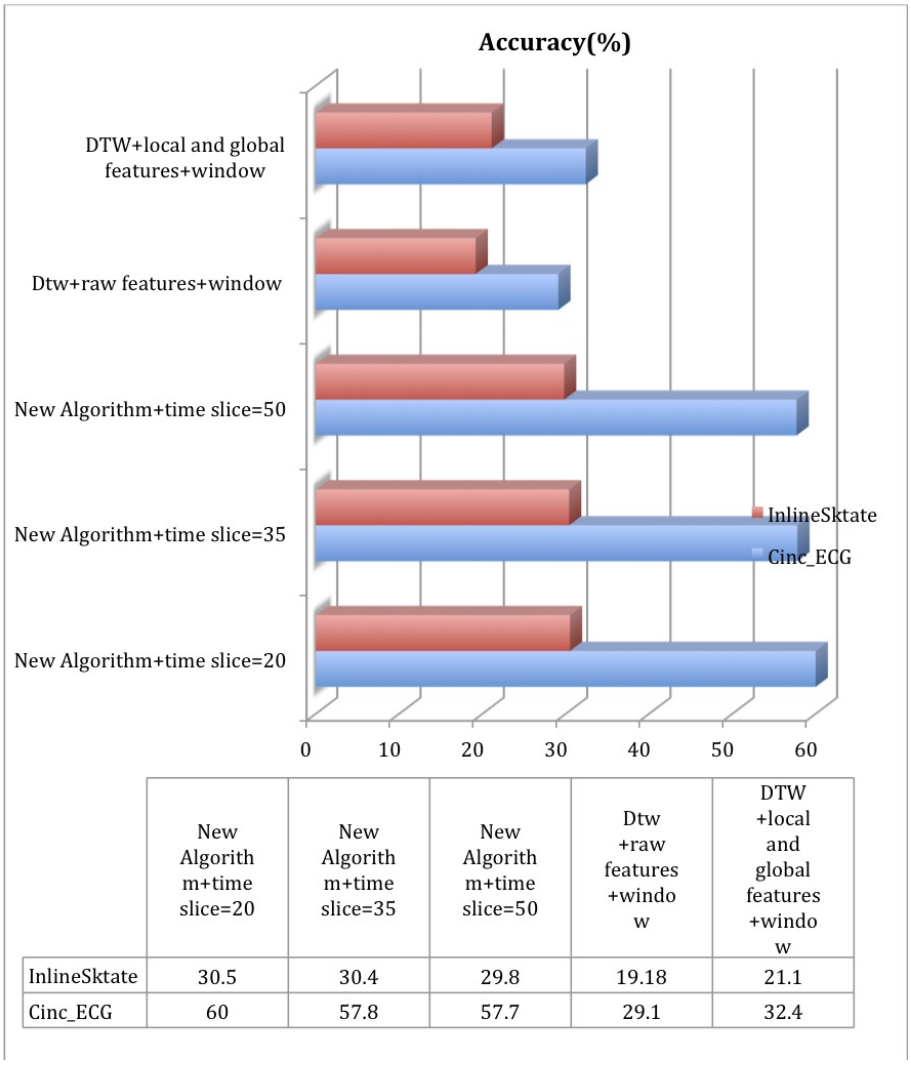
As a I mentioned, the domain-dependent feature selection process of silence removal and subsampling is dropped from all approaches. However, the feature extraction process that involves extracting local and global features is kept since this procedure is domain independent.

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

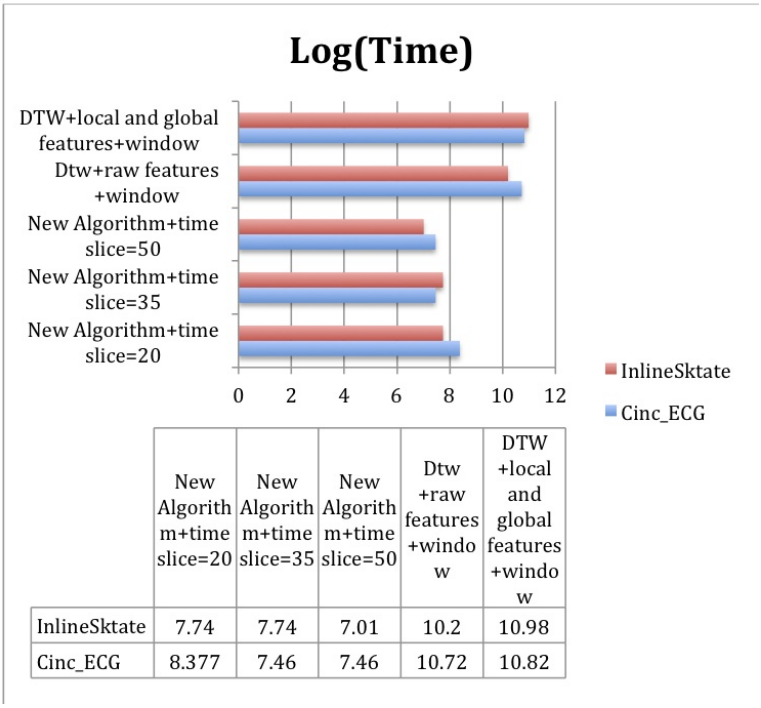


	New Algorithm+time slice=20	New Algorithm+time slice=35	New Algorithm+time slice=50	Dtw +raw features +window	DTW +local and global features +window
InlineSktate	30.5	30.4	29.8	19.18	21.1
Cinc_ECG	60	57.8	57.7	29.1	32.4

FIGURE 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 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