Making Time-series Classification More Accurate Using Learned Constraints

Chotirat Ann Ratanamahatana Eamonn Keogh
University of California - Riverside
Computer Science & Engineering Department
Riverside, CA 92521, USA
{ratana, eamonn}@cs.ucr.edu

Abstract
It has long been known that Dynamic Time Warping (DTW) is superior to Euclidean distance for classification and clustering of time series. However, until lately, most research has utilized Euclidean distance because it is more efficiently calculated. A recently introduced technique that greatly mitigates DTWs demanding CPU time has sparked a flurry of research activity. However, the technique and its many extensions still only allow DTW to be applied to moderately large datasets. In addition, almost all of the research on DTW has focused exclusively on speeding up its calculation; there has been little work done on improving its accuracy. In this work, we target the accuracy aspect of DTW performance and introduce a new framework that learns arbitrary constraints on the warping path of the DTW calculation. Apart from improving the accuracy of classification, our technique as a side effect speeds up DTW by a wide margin as well. We show the utility of our approach on datasets from diverse domains and demonstrate significant gains in accuracy and efficiency.

Keywords: Time Series, Dynamic Time Warping.

1 Introduction.

In recent years, classification and clustering of time series data have become a topic of great interest within the database/data mining community. Although the Euclidean distance metric is widely known to be very sensitive to distortion in time axis [3][9][22][27][44], the vast majority of research has used Euclidean distance metric or some minor variation thereof [2][12] [16][25][29][45]. The ubiquity of Euclidean distance in the face of increasing evidence of its poor accuracy for classification and clustering is almost certainly due to its ease of implementation and its time and space efficiency.

The problem of distortion in the time axis can be addressed by Dynamic Time Warping (DTW), a distance measure that has long been known to the speech processing community [21][31][35][38][41][43] and was introduced to the database community by Berndt and Clifford [4]. This method allows non-linear alignments between two time series to accommodate sequences that are similar, but locally out of phase, as shown in Figure 1.

As Berndt and Clifford originally noted, DTW does not scale very well to large databases because of its quadratic time complexity.

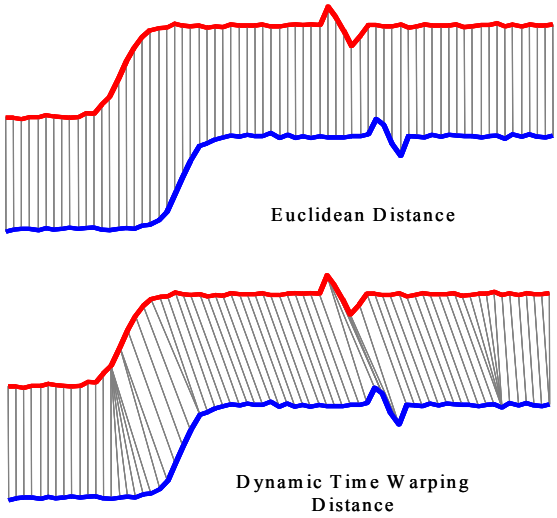

Figure 1: Note that while the two time series have an overall similar shape, they are not aligned in the time axis. Euclidean distance, which assumes the i^{th} point in one sequence is aligned with the i^{th} point in the other, will produce a pessimistic dissimilarity measure. The non-linear Dynamic Time Warped alignment allows a more intuitive distance measure to be calculated.

In spite of its computational lethargy, DTW still is the best way to solve a vast range of time series problems, and it is widely used in various disciplines:- In bioinformatics, Aach and Church successfully applied DTW to cluster RNA expression data [1]. In chemical engineering, it has been used for the synchronization and monitoring of batch processes [19]. DTW has been effectively used to align biometric data, such as gait [18], signatures [34], fingerprints [30], and ECGs [5]. Rath and Manmatha have successfully applied DTW to the problem of indexing repositories of handwritten historical documents [39] (Although handwriting is 2-dimensional, it can be useful to re-represent it as a 1-dimensional time series). DTW is often the technique of choice for indexing video motion streams [36]. In robotics, Schmill et al. demonstrate a technique that utilizes DTW to cluster robots sensory outputs [42]. And finally, in music, Zhu and Shasha (among many others [20]) have exploited DTW to query music databases with snippets of hummed phrases [46].

However, the greater accuracy of DTW comes at a cost. Depending on the length of the sequences, DTW is typically hundreds or thousands of times slower than Euclidean distance. For example, Clote et al. [11] report an experiment using DTW to align gene expression data that required 6 days.

During the past decade, there has been a huge amount of work on speeding up data mining time series under the Euclidean distance [7][16][24][45] (see [25] for a comprehensive listing). However, the first practical technique for speeding up data mining time series under DTW [23] has only been introduced very recently. The technique is based on using DTWs "warping envelope" constraint to lower bound the true DTW distance, hence pruning off many costly distance computations. This work has sparked a flurry of research interest [10][17][20][33][47], ensuring that the warping-envelope lower bounding technique has become a relatively mature technology within a year. However, there are still two areas in which improvements need to be made: scalability to truly massive datasets and classification accuracy.

In this work, we address these problems with a novel technique. Our approach is based on learning arbitrarily shaped warping-envelope constraints: These learned constraints allow:

- Improved accuracy; by allowing some warping that increases intra-class similarity, while discouraging warping that increases inter-class similarity.
- Faster classification and similarity search; by exploiting the constraints to achieve extraordinarily tight lower bounds, thus allowing pruning.

An interesting and useful property of our technique is that it includes the ubiquitous Euclidean distance and classic DTW as special cases.

The rest of the paper is organized as follows. Section 2 gives some background on time series data mining, a review of DTW, and related work. Section 3 reviews the lower bounding measures and its utility. In Section 4, we introduce our approach, a novel framework that we call the *R-K Band*, to the problem. Section 5 contains an empirical evaluation on three real-world datasets. And lastly, Section 6 gives conclusions and directions for future work.

2 Background.

The measurement of similarity between two time series is an important subroutine in many data mining applications, including rule discovery [12], clustering [1] [14], anomaly detection [13], motif discovery [8], and classification [15][22]. The superiority of DTW over Euclidean distance for these tasks has been demonstrated by many authors [1][3][5][27][44]; nevertheless, DTW is less familiar to the data mining community. We will therefore begin with overview of DTW and its recent extensions.

2.1 Review of DTW.

Suppose we have two time series, a sequence Q of length n, and a sequence C of length m, where:

$$Q = q_1, q_2, \ldots, q_i, \ldots, q_n \quad (1)$$

$$C = c_1, c_2, \ldots, c_j, \ldots, c_m \quad (2)$$

To align these two sequences using DTW, we construct an n-by-m matrix where the (i^{th}, j^{th}) element of the matrix corresponds to the squared distance, $d(q_i,c_j) = (q_i - c_j)^2$, which is the alignment between points q_i and c_j. To find the best match between these two sequences, we can find a path through the matrix that minimizes the total cumulative distance between them. A warping path, W, is a contiguous set of matrix elements that characterizes a mapping between Q and C. The k^{th} element of W is defined as $w_k = (i,j)_k$. So we have:

$$W = w_1, w_2, \ldots, w_k, \ldots, w_K \qquad \max(m,n) \le K < m+n-1 \quad (3)$$

By definition, the optimal path W_o is the path that minimizes the warping cost:

$$DTW(Q,C) = \min\left\{ \sqrt{\sum_{k=1}^{K} w_k} \right\} \quad (4)$$

This path can be found using dynamic programming to evaluate the following recurrence which defines the cumulative distance $\gamma(i,j)$ as the distance $d(i,j)$ found in the current cell and the minimum of the cumulative distances of the adjacent elements:

$$\gamma(i,j) = d(q_i,c_j) + \min\{ \gamma(i-1,j-1), \gamma(i-1,j), \gamma(i,j-1) \} \quad (5)$$

In practice, we do not evaluate all possible warping paths, since many of them correspond to pathological warpings (for example, a single point on one ECG mapping to an entire heartbeat in another ECG). Instead, we consider the following constraints that decrease the number of paths considered during the matching process. This reduction in the number of paths considered also has the desirable side effect of speeding up the calculations, although only by a (small) constant factor.

Boundary conditions: The path must start in $w_1 = (1,1)$ and end in $w_K = (m,n)$, that is, the warping path has to start at the bottom left and end at the top right of the matrix.

Continuity condition: Every point in the query and candidate sequences must be used in the warping path, and both i and j indexes can only increase by 0 or 1 on each step along the path. In other words, if we take a point (i, j) from the matrix, the previous point must have been (i-1, j-1), (i-1, j), or (i, j-1).

Monotonic condition: Given $w_k = (a,b)$ then $w_{k-1} = (a',b')$ where $a - a' \ge 0$ and $b - b' \ge 0$. The warping path cannot go backward in time; both i and j indexes either stay the same or increase. They can never decrease.

Slope constraint condition: The path should not be too steep or too shallow. This prevents very short subsequences

to match very long ones. The condition is expressed as a ratio *a/b*, where *b* is the number of steps in the *x* direction and *a* is the number in the *y* direction. After b steps in x, it must make a step in *y*, and vice versa.

Adjustment Window condition: An intuitive alignment path is unlikely to drift very far from the diagonal. The distance that the path is allowed to wander is limited to a window (or "band") of size *r,* directly above and to the right of the diagonal.

By applying these conditions, we can restrict the moves that can be made from any point in the path and therefore reduce the number of paths that need to be considered.

The Euclidean distance between two sequences can be seen as a special case of DTW where the k^{th} element of W is constrained such that $w_k = (i,j)_k$, $i = j = k$. Note that it is only defined in the special case where the two sequences have the same length. The time and space complexity of DTW is O(*nm*). However, the constraints above mitigate this only by a constant factor.

This review of DTW is necessarily brief; we refer the interested reader to [31][37] for more details.

2.2 Related work.

While there has been much work on indexing time series under the Euclidean metric over the last decade [7][16][24][25][45], there has been much less progress on indexing under DTW. Additionally, all of the work on DTW has focused exclusively on speeding up DTW; it does not appear that researchers have considered the possibility of making DTW more accurate.

Keogh [23] introduced a novel technique for exact indexing of DTW using global constraints and Piecewise Constant Approximation [24]. The proposed lower bounding measure, *LB_Keogh*, exploits the global constraints to produce a very tight lower bound that prunes off numerous expensive DTW computations. The method has been re-implemented and extended by several other research groups [17][28][46], and is now the basis of a successful "query-by-humming" system [47] and a system for indexing historical handwriting documents [33]. Because of the power and widespread adoption of this approach, we will utilize LB_Keogh lower bounding function as a starting point for this work.

We note there has been some work on obtaining warping alignments by methods other than DTW [3][32]. For example, Kwong et al. consider a genetic algorithm based approach [32], and recent work by Bar-Joseph et al. considers a technique based on linear transformations of spline-based approximations [3]. However, both methods are stochastic and require multiple runs (possibly with parameter changes) to achieve an acceptable alignment. In addition, both methods are clearly non-indexable. Nevertheless, both works do reiterate the superiority of warping over non-warping for pattern matching.

3 Lower Bounding the DTW Distance.

In this section, we explain the importance of lower bounding and briefly review the LB_Keogh lower bounding distance measure [23].

3.1 The utility of lower bounding measures.

Time series similarity search under the Euclidean metric is heavily I/O bound; however, similarity search under DTW is also very demanding in terms of CPU time. One way to address this problem is to use a fast lower bounding function to help prune sequences that could not possibly be the best match (see [23] for full algorithm detail).

There are only two desirable properties of a lower bounding measure:

- It must be fast to compute. Clearly, a measure that takes as long to compute as the original measure is of little use. In our case, we would like the time complexity to be at most linear in the length of the sequences.
- It must be a relatively tight lower bound. A function can achieve a trivial lower bound by always returning zero as the lower bound estimate. However, in order for the algorithm to be effective, we require a method that more tightly approximates the true DTW distance.

While lower bounding functions for string edit, graph edit, and tree edit distance have been studied extensively [31], there has been far less work on DTW, which is very similar in spirit to its discrete cousins. Below, we will review global constraints, which can be exploited to produce tight lower bounds.

3.2 Existing lower bounding measures.

As previously noted, virtually all practitioners using DTW constrain the warping path in a global sense by limiting how far it may stray from the diagonal [4][9][19][21][27] [35][41][43]. The subset of matrix that the warping path is allowed to visit is called a warping window or a band. Figure 2 illustrates two of the most frequently used global constraints in the literature, the Sakoe-Chiba Band [41] and the Itakura Parallelogram [21].

In addition to helping to speed up the DTW distance calculation, the warping window prevents a pathological warping, where a relatively small section of one sequence maps onto a relatively large section of another. The importance of global constraints was documented by the originators of the DTW algorithm, Sakoe and Chiba, who were exclusively interested in aligning speech patterns [41]. However, it has been empirically confirmed in many other settings, including music [20][47], finance [4], medicine [19], biometrics [18][34], chemistry [19], astronomy, robotics [42], and industry.

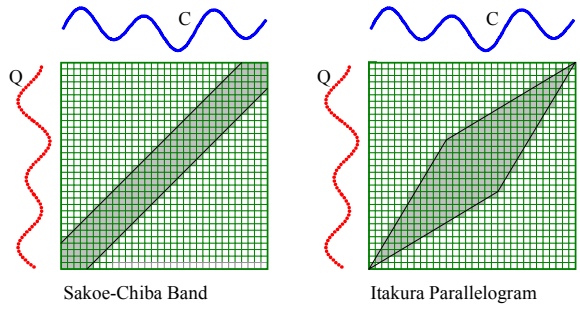

Figure 2: Global constraints limit the scope of the warping path, restricting them to the gray areas. The two most common constraints in the literature are the Sakoe-Chiba Band and the Itakura Parallelogram.

As mentioned earlier, a lower bounding distance measure, LB_Keogh, has been introduced for the task of indexing DTW. This lower bounding technique uses the warping window, e.g. Sakoe-Chiba Band or Itakura Parallelogram, to create a bounding envelope above and below the query sequence. Then the squared sum of the distances from every part of the candidate sequence not falling within the bounding envelope, to the nearest orthogonal edge of the bounding envelope, is returned as its lower bound. The technique is illustrated in Figure 3. This lower bound can prune off numerous number of expensive DTW computations, using the simple algorithm described in [23].

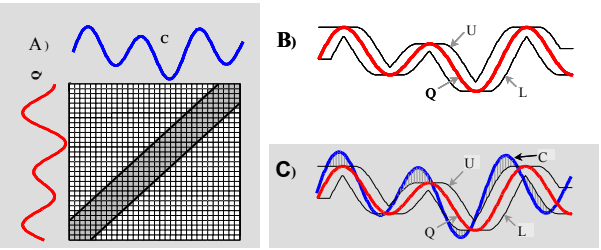

Figure 3: The Sakoe-Chiba Band A) can be used to create an envelope B) around a query sequence Q. The Euclidean distance between any candidate sequence C and the closest external part of the envelope C) is a lower bound for the DTW distance.

4 Ratanamahatana-Keogh Band (*R-K Band*).

The warping window constraints discussed above have been used to restrict the warping paths to force more intuitive alignments, as well as to speed up the calculation. However, surprisingly little research has been done on discovering the best shape and size of the band. Instead, the relatively ad-hoc shapes of the bands introduced exclusively in the context of speech recognition in the 1970s have survived DTWs migration to diverse domains. Many researchers seem to believe that the wider the band, the better classification accuracy [47], and that narrow bands are a necessary evil, required only to make the algorithm tractable. In fact, this is not true; in Section 4.2 we carry out extensive experiments in which we vary the width of Sakoe-Chiba Band. We find that the effect of the band width on accuracy is generally very large, and is heavily domain dependent. This observation motivates our work. If the *width* of the constraint band can greatly affect accuracy, then it is likely that the *shape* of the band can too. If we can somehow find the optimal band shape for a given problem, we may be able to boost the accuracy. Furthermore, if the optimal band shape is tighter than the classic bands, we can simultaneously reduce the CPU time and increase the tightness of the lower bounds, producing speed up.

In order to discuss the effect of constraint shape on accuracy, we must first introduce a representation that allows the specification of arbitrary shaped constraints. For consistency with the literature, we call this representation the Ratanamahatana-Keogh band, and introduce it in the next section.

4.1 A general model of global constraints.

We can view a global constraint as constraining the indices of the warping path $w_k = (i,j)_k$ such that $j-R_i \le i \le j+R_i$, where R_i is a term defining the allowed range of warping, for a given point in a sequence. In the case of the Sakoe-Chiba Band, R is independent of i; for the Itakura Parallelogram, R is a function of i. Here we define a parameter vector R more concretely.

$$R_i = d \quad 0 \le d \le m,\ 1 \le i \le m, \tag{6}$$

where R_i is the height above the diagonal in the y direction, as well as the width to the right of the diagonal in the x direction. Note that $|R| = m$, and the above definition forces R to be symmetric, i.e. the constraint above the diagonal is the mirror image of the one below the diagonal.

As an example, we can create a Sakoe-Chiba Band of overall width of 11 (width 5 strictly above and to the right of the diagonal) with the definition

$$R_i = \begin{cases} 5 & 1 \le i \le m-5 \\ m-i & m-5 < i \le m \end{cases} \tag{7}$$

or an Itakura Parallelogram with the definition

$$R_i = \begin{cases} \left\lfloor \frac{2}{3} i \right\rfloor & 1 \le i \le \left\lfloor \frac{3}{8} m \right\rfloor \\ \left\lfloor \frac{3}{8} m \right\rfloor - \left\lfloor \frac{2}{5} i \right\rfloor & \left\lfloor \frac{3}{8} m \right\rfloor < i \le m \end{cases} \tag{8}$$

Even the Euclidean distance can be defined in terms of $R_i = 0$; $1 \le i \le m$; only the diagonal path is allowed. More generally, we can define any arbitrary global constraint with the vector R. Figure 4 illustrates some examples.

We call a global constraint specified by R a Ratanamahatana-Keogh-Band (which we will abbreviate as an *R-K Band*).

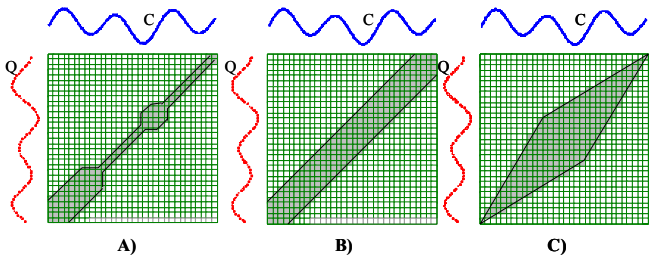

Figure 4: We can use *R* to create arbitrary global constraints. A) Note that the width of the band may increase or decrease. We can use *R* to specify all existing global constraints, including the Sakoe-Chiba Band B) and the Itakura Parallelogram C).

We can exploit *Ratanamahatana-Keogh Bands* for classification. In particular, we can use a different *R-K Band* for each class. We will denote the band learned for the c^{th} class, as the *R-K$_c$ Band*.

Before considering the effects of the *shape* of the bands, we will first consider the effect of the *size* of the bands.

4.2 Are we better off with wider band?

Virtually, all researchers have used a Sakoe-Chiba Band with a 10% width for the global constraint. This setting appears to be the result of historical inertia, rather than some remarkable property of this particular constraint.

To test the effect of the warping window size to the classification accuracies, we perform an empirical experiment on 3 datasets for which class labels are available (the datasets details are fully explained in section 5). We vary the warping window size from 0 (Euclidean) to 100 and record the accuracies. The results are shown in Figure 5.

Surprisingly, wider bands do not always result in increased accuracy, as commonly believed [46]. More often, the accuracy peaks very early at smaller window size.

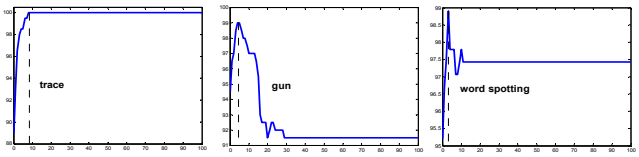

Figure 5: The classification accuracies of the 3 datasets with different warping window sizes (x-axis), using the uniform Sakoe-Chiba Bands. Most accuracies peak at very small window sizes.

Apart from the width, different shapes of the band also give different accuracies as well. For example, one may pick the Itakura Parallelogram over the uniform Sakoe-Chiba band to perform classification in speech recognition problem, since most speech tends to have the most variability in its middle, and very little in the beginning or the end. Consequently, choosing a good *shape* for the band may help improve its classification accuracy as well. Having introduced an *R-K Band*, we can easily represent any shape and size warping windows. However, we are left with the question of how we can discover the best *R-K Band* for the task at hand.

In some cases, we may be able to manually construct the best set of *R-K$_c$ Bands* for classification, based on domain knowledge. Consider the following motivating problem; the 2-class dataset shown in Figure 6 (top): Since we can see from Figure 6 (top) that both classes have similar variability in the x-axis approximately around points 50 to 135, we can allow some warping in those regions of the hand-constructed bands, *R-K$_1$* and *R-K$_2$*. And since class 1 has an extra place of variability in y-axis approximately between data points 150 and 225, we accordingly allow some warping in that section of *R-K$_1$* as well. The shape of hand-created *R-K$_1$* and *R-K$_2$ Bands* are shown in Figure 6 (bottom).

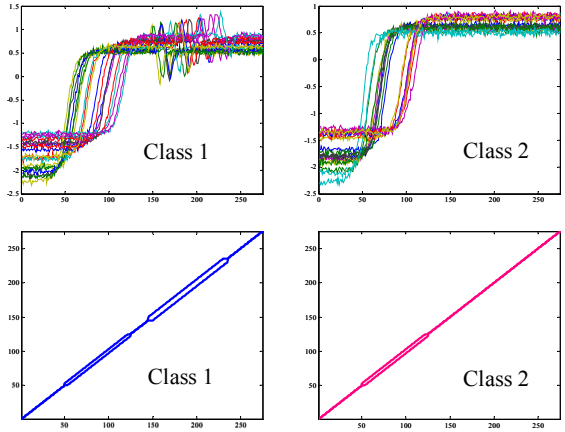

Figure 6: (Top) Some instances from the *trace* dataset. (Bottom) The hand-constructed bands created for each individual class; they achieve 100% accuracies in classification.

The Euclidean accuracy for this problem is only 90%. The hand-constructed *R-K Bands* illustrated in Figure 6 (bottom) give perfect accuracy with the widest portion of the band of 4. However, we have to try several widths and shapes until we obtain the smallest width of the shape we want with the best accuracy. In addition, we make some unintuitive discoveries. We set the *R-K$_2$ Band* to the Euclidean special case, and still get perfect results. So, while there is variability in the time-axis of Class 2, it is not important for discriminating it from Class 1. In general, even for this simplest of problems we have great difficulty in hand crafting high quality *R-K Bands*, and only converge on the fine solution after much tweaking. Note that this is only a 2-class problem; when we try to expand it to a 4-class problem (see Section 5), the difficulties in constructing *R-K$_c$ Bands* appear to increase exponentially.

While the results above are tentative vindication of the *R-K Band* representation, for most real-world problems it is simply not possible to build the high quality *R-K$_c$ Bands* by hand. Instead, in the following sections, we will show how we can *learn* them automatically from the data.

4.3 Learning multiple Ratanamahatana-Keogh bands for classification.

We have shown that it is not generally possible to handcraft accurate *R-K Bands*. Fortunately, as we will show, it is possible to pose the problem as a classic search problem, and thus take advantage of the wealth of research on search from the artificial intelligence community [39].

Using the generic heuristic search techniques elucidated in [39], we merely need to specify the direction of the search, i.e *forward*, *backward* or *bi-directional*, the initial state, heuristic function, the operators, and the terminal test. Forward search starts with the initial Sakoe-Chiba band (uniform) of width 0 (Euclidean), and backward search starts from the uniform band of the maximum width *m*, above and to the right of the diagonal.

Before giving a full detailed explanation of our learning algorithm, we first give a simple intuition behind this approach as illustrated in Figure 7.

For a forward search, we start off with the Euclidean Band, and try to increment the whole section of the envelope before re-evaluating its accuracy. If an improvement is made, we keep on incrementing that whole section of the envelope; otherwise, we split that section in half and recursively increment each portion individually before re-evaluation. Backward search is very similar, except that we start off with a wider band and try to decrement the size instead of incrementing. We do not consider bi-directional search in this work for brevity.

The rest of the components of the search algorithm are enumerated below:

- The **initial state**, which includes the initial shape of the band for each class.
  - Forward Search: $R_i = 0$
  - Backward Search: $R_i = m$
- A **heuristic function**, which is used to evaluate the quality of the operation. While we are ultimately interested in improving accuracy, there may be a danger of overfitting using accuracy as the heuristic function. We therefore consider two different heuristics in this work:
  - *Accuracy metric: h(env)* = estimated accuracy using DTW, based on the envelope *env*.
  - *Distance metric: h(env)* = estimated ratio of the mean DTW distance among the correctly classified and misclassified objects.

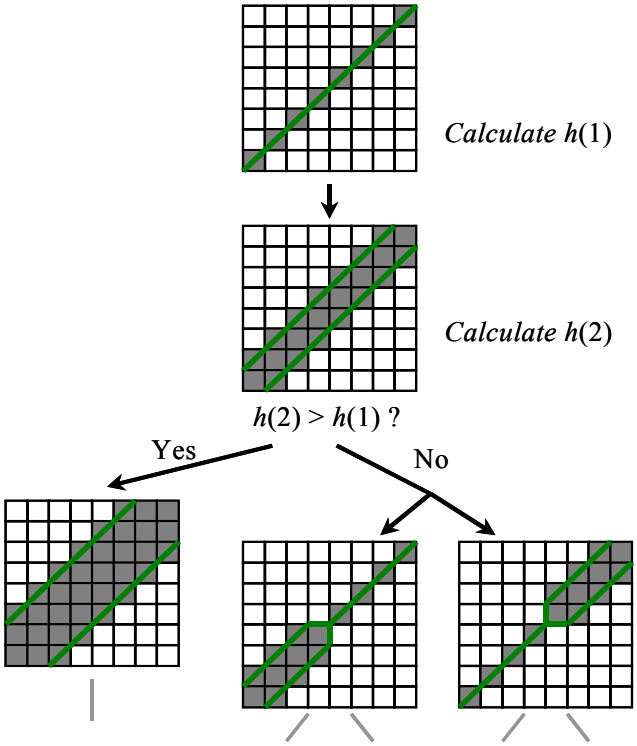

Figure 7: An illustration of our forward search algorithm using the accuracy metric.

- A set of **operators**, which defines all the legal operations that can be taken at each stage. There are two available operations at each stage after the heuristic function is evaluated:
  - If improvement is made, continue working on the same piece of envelope (as specified by *start* and *end* points.
  - If no improvement is made, first the modification to the envelope needs to be undone, then redo the modification on the left half (*start:mid-1*) and the right half (*mid:end*) individually.
- A **terminal test**, which determines when the search is completed.
  - *Forward Search:* The search is complete when one of the following is true:
    - No improvement can be made.
    - The envelope's width reaches *m*, or
    - When a condition $end\text{-}start+1 \leq threshold$ is met for all the pieces of the envelope.
  - *Backward Search:* The search is complete when one of the following is true:

- No improvement can be made.
- The envelope has width 0 (Euclidean), or
- A condition *end-start+1* $\leq$ *threshold* is met for all the pieces of the envelope.

In principle, the threshold could be a single cell, but this would produce a greater danger of overfitting. In this work, we set the *threshold* to be $\frac{\sqrt{m}}{2}$, where *m* is the length of the input sequence.

Table 1: A *backward* hill-climbing search algorithm that finds a proper envelope for each individual class.

```
Algorithm LearnEnv()
1. Evaluate (with initial envelopes);
2. foreachclass i = 1:c
3.   Enqueue(1, m, Q_i);
4. endfor;
5. while !empty(Q_1:c)
6.   foreachclass i = 1:c
7.     if !empty(Q_i)
8.       [start,end] = Dequeue(Q_i);
9.       Decrement the envelope Env_i(start:end)
10.       improve = Evaluate(with the
                    modified envelope Env_i);
11.       if improve
12.         Enqueue(start, end, Q_i);
13.       else
14.         Undo the modification
15.         Enqueue(start, mid-1, Q_i);
16.         Enqueue(mid, end, Q_i);
17.       endif;
18.     endif;
19.   endfor;
20. endwhile;
21. return Env_1:c
```

Table 1 shows the backward learning algorithm that discovers the high-quality envelope for each individual class. The only different between backward and forward search algorithm is on line 9; we *increment* the width of the envelope for forward search, whereas we *decrement* the width of the envelope for backward search.

The algorithm starts off by evaluating all the initial envelopes of all classes and enqueues the whole length of the envelope into the queue that belongs to each individual class (lines 1-4). As long as there is something left in the queue of any classes (line 5), it takes turns, starting from class 1 up to class c, to make some modification and re-evaluate the envelope (lines 6-10). If that envelope provides some improvement, then that piece of envelope is kept and the width is further reduced (increased in forward search) in the next round of iteration (lines 11-12). If it provides no improvement, we have to restore the previous envelope and try to re-modify its smaller pieces (left half and right half) in the next round of iterations (lines 13-16). Note that only one envelope can be modified at each time before the evaluation; the rest of the envelopes of the remaining classes must remain the same since a modification on an individual envelope may affect the classification in other classes as well. This way, when there is some improvement, we know exactly which envelope modification it comes from. The process is repeated until no further change could be made to any of the envelopes that improves the accuracy; the algorithm then returns the resulting envelopes for all classes ($Env_{i:c,}$, cf. Table 1, line 21).

Because "Local maximum" is a well-known drawback of the hill-climbing search, our algorithm does not guarantee an optimal solution. To mitigate this, we can perform all four combinations of the search to find the set of envelopes that yields the best outcome. The four combinations are:

- Forward search with Accuracy metric.
- Forward search with Distance metric.
- Backward search with Accuracy metric, and
- Backward search with Distance metric.

Table 2 shows the *Evaluate* function in greater detail. Lines 2-16 locate the best match for one test instance, according to the DTW distance with LB_Keogh function used to prune off some unnecessary computations. It loops through all examples as it carries out the "leaving-one-out" scheme (line 1). Once the best match is found, the DTW distance is recorded as well as the statistic of correctly classified vs. misclassified examples (lines 17-23). If the new distance metric is smaller or the new accuracy is larger than what we have so far, that means we get some improvement (lines 25-31). In backward search, we are trying to make the envelope smallest possible; this in turn counts identical metric value as an improvement as well; smaller or equal value of *distmetric* and larger or equal value of accuracy are counted as improvement.

Table 2: An algorithm to evaluate both accuracy- and distance-metric heuristic functions for both forward and backward search. If the input envelopes yield some improvement, the algorithm returns 1, otherwise returns 0.

```
Algorithm Evaluate()

1.  for i = 1:num_examples
2.    test_instance = Input[i];
3.    best_so_far = Infinity;
4.    best_index = -1;
5.    for j = 1:num_examples
6.      if i != j
7.        LB_dist = LB_keogh(test_instance,
                      Input[j], Env_Input[j][0]);
8.        if LB_dist < best_so_far
9.          true_dist=dtwDistance(test_instance
                        ,Input[j],Env_Input[j][0]);
10.         if true_dist < best_so_far
11.           best_so_far = true_dist;
```

```
12.           best_index = j;
13.         endif;
14.       endif;
15.     endif;
16.   endfor;
17.   if Input[best index][0]==test instance[0]
18.     num_correct++;
19.     dist += dtwDistance(test_ instance,
              Input[best_index],Env_test_instance[0]);
20.   else
21.     num_wrong++;
22.     wrongdist += dtwDistance(test_instance,
              Input[best_index],Env_test_instance[0]);
23.   endif;
24. endfor;
25. distmetric = (dist*num_wrong)/(wrongdist *
                  num_correct);
26. if smaller distmetric or larger num_correct
27.   improve = 1;
28. else
29.   improve = 0;
30. endif;
31. return improve;
```

In the following section, we evaluate our proposed framework on three real-world datasets.

5 Experimental Evaluation.

In this section, we test our proposed approach with a comprehensive set of experiments.

5.1 Dataset.

We have chosen three datasets to be tested in our work.

5.1.1 Gun Problem

This dataset comes from the video surveillance domain. The dataset has two classes, each containing 100 examples. All instances were created using one female actor and one male actor in a single session. The two classes are:

- **Gun-Draw:** The actors have their hands by their sides. They draw a replicate gun from a hip-mounted holster, point it at a target for approximately one second, then return the gun to the holster, and their hands to their sides. Figure 8 illustrates some snippets from the video.
- **Point:** The actors have their hands by their sides. They point with their index fingers to a target for approximately one second, and then return their hands to their sides.

For both classes, we tracked the centroid of the right hand in both the X- and Y-axes; however, in this experiment, we will consider just the X-axis for simplicity.

Figure 8: Stills from the video *Gun-Draw* problem; the right hand is tracked and converted into motion streams.

The overall motions of both classes are very similar. However, it is possible for human to visually classify the two classes with great accuracy, after noting that the actor must lift his/her hand above a holster, then reach down for the gun, this action creates a subtle distinction between the classes as shown in Figure 9.

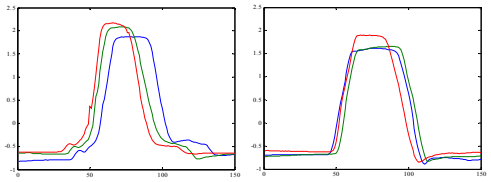

Figure 9: (left) Some time series from the *Gun-Draw* x-axis data. (right) Some snippets from the *Point* x-axis data.

The different in height between the two actors is not too much of a problem, since a standard pre-processing step in matching motion streams is to perform Z-normalization before performing comparisons [6][26].

The dataset contains 200 instances, 100 for each class. Each instance has the same length of 150 data points.

5.1.2 Trace dataset

This dataset is a subset of the Transient Classification Benchmark (trace project).

It is a synthetic dataset designed to simulate instrumentation failures in a nuclear power plant, created by Davide Roverso. The full dataset consists of 16 classes, 50 instances in each class. Each instance has 4 features.

For simplicity, we only use the second feature of class 2 and 6, and the third feature of class 3 and 7 for our experiment. Our dataset contains 200 instances, 50 for each class. All instances are interpolated to have the same length of 275 data points. Examples of each class are shown in Figure 10.

5.1.3 Handwritten Word Spotting data

This is a subset of the WordSpotting Project dataset created by Manmatha and Rath [33].

In the full dataset, there are 2,381 words with four features that represent each word image's profiles or the background/ink transitions.

For simplicity, we pick the "Projection Profile"(feature 1) of the four most common words, "*the*", "*to*", "*be*", and "*that*", to be used in our experiment. "*the*" has 109 instances; "*to*" has 91 instances; "*be*" has 38 instances, and "*that*" has 34 instances. All instances are interpolated to have the same length of 100 data points. Once combined, we obtain a dataset of 272 instances.

5.2 Experimental results.

In this section, we test our proposed approach with a comprehensive set of experiments. On all three datasets mentioned in the previous section, we perform classification using the following approaches:

- Euclidean Distance.
- Dynamic Time Warping with Sakoe-Chiba Band (uniform warping window) of size 1 up to 100. The best accuracy with smallest-size band is to be reported, and
- Dynamic Time Warping with *R-K Bands* that we learn from the input data.

Note that we only compare Dynamic Time Warping with Euclidean Distance metric in this work. It has been forcefully shown in [26] that many of the more complex similarity measures proposed in other work have higher error rates than a simple Euclidean Distance metric, and therefore by transitivity have higher error rates than DTW itself. We therefore exclude those techniques from our consideration in this experiment. The learned bands from each datasets are shown in Figure 10 and Figure 11.

We measure the accuracy and CPU time on each dataset, using the 1-nearest-neighbor with "leaving-one-out" classification method. The lower bounding technique introduced in [23] is also integrated in all the DTW calculations to help achieve some speedup.

Table 3 compares the classification accuracies (Error rates) for all approaches, and Table 4 compares the CPU time for each method to achieve these accuracies, both with using Lower Bounding measure and without. Euclidean distance metric is essentially a DTW with uniform band of width 0 (no warping allowed). For the uniform (Sakoe-Chiba) band, we report the best accuracy within the window width between 1 and 100. We also report the accuracy at 10% warping window size since it is the number most researchers typically have been using in DTW researches [23][38][41].

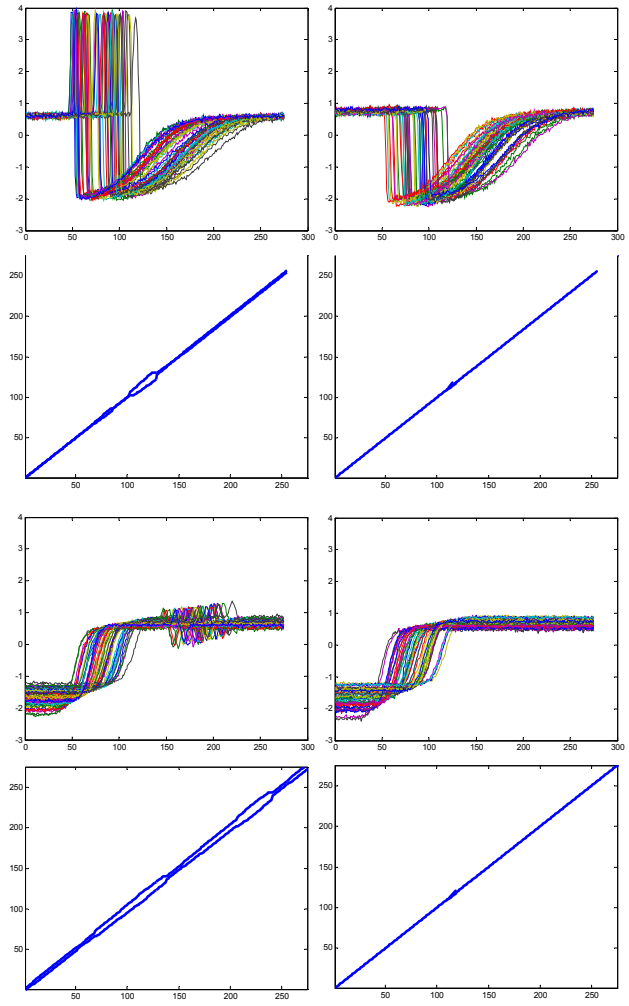

Figure 10: Trace dataset: The *R-K Bands* for all four classes.

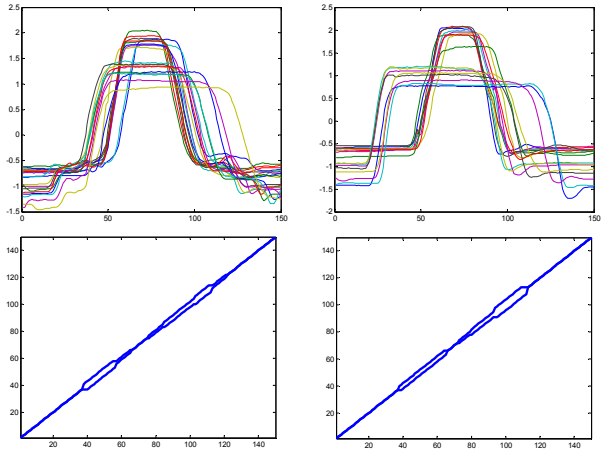

Figure 11: Gun dataset: The *R-K Bands* for the Gun (left) and Point (right) classes.

Table 3: Classification Error Rates (%) for all three datasets, using Euclidean distance, DTW with the best uniform band reported, DTW with 10% uniform band, and DTW with our framework, *R-K$_c$ Bands*.

	Euclidean	Best Uniform	10% Uniform	*R-K Bands*
Gun	5.50%	1.00% at $R_i = 4$	4.50% at $R_i = 15$	0.50% with max(R_i) = 4
Trace	11.00%	0.00% at $R_i = 8$	0.00% at $R_i = 27$	0.00% with max(R_i) = 7
Word Spotting	4.78%	1.10% at $R_i = 3$	2.21% at $R_i = 10$	0.37% with max(R_i) = 4

Table 4: CPU time (msec) for all three datasets, both with and without the use of Lower Bounding measure.

	Euclidean	Best Uniform	10% Uniform	*R-K-Bands*
Gun (LB)	N/A	2,440	5,430	1,440
No LB	60	11,820	17,290	9,440
Trace (LB)	N/A	16,020	34,980	7,420
No LB	210	144,470	185,460	88,630
Word Spotting (LB)	N/A	6,100	14,770	1,940
No LB	40	8,600	12,440	7,480

We can readily see from the above figures and tables that the learned *R-K Bands* usually are of smaller sizes than the uniform case; some portions of the band even have zero width. This speeds up the time needed for classification. We can improve the DTW calculation from running several hundred times slower than Euclidean to running about only 30 times or so slower (or even less than 5 times slower in some other datasets). In addition, classification with *R-K Bands* always achieves higher accuracies than Euclidean or any-size uniform bands (or at least as accurate as the better of the two methods).

In the three datasets discussed above, it is known that there is some distortion in the time axis. Where no distortion exists (and assuming Gaussian noise), Euclidean distance is known to be the optimal metric [16]. If we were to apply DTW in these cases, we may get lower accuracy, and we will certainly waste a lot of extra time. For real-world problems, we usually never know if Euclidean distance is the best approach. So, it would be very desirable if our approach could discover this automatically.

To see if this is the case, we performed an additional classification experiment on the well-known Cylinder-Bell-Funnel datasets [15][22], on which Euclidean distance is known to perform extremely well, with sufficient number of instances. With 100 instances in each of the three classes, the Euclidean distance metric achieves 100% accuracy in 0.16 seconds. DTW also achieves perfect accuracy, but wastes a large amount of classification time without realizing the trivial solution.

The DTW algorithm with 10% warping window size requires 27.63 seconds. Our approach, learning *R-K$_c$ Bands*, quickly discovers that perfect accuracy can be achieved with three bands of size zero. In other words, our approach is capable of learning the optimal-size bands for this problem. With the resultant *R-K$_c$ Bands* of size 0, we also get the perfect accuracy using only 0.89 seconds. This is slightly slower than the Euclidean distance since the *R-K$_c$ Bands* are the special case of DTW thus a distance matrix has to be created during the computation. However, it is still much faster than the classic 10%-uniform DTW. It is also trivial to force the algorithm to perform the original Euclidean metric calculation instead of the DTW calculation of the zero band size.

6 Conclusions and Future Work.

In this work, we have introduced a new framework for classification of time series. The Ratanamahatana-Keogh Band (*R-K Band*) allows for any arbitrary shape and size of the warping band. We have also introduced a heuristic search algorithm that automatically learns the *R-K Bands* from the data. With an extensive empirical evaluation, we have shown that our approach can reduce the error rate by an order of magnitude, and reduce the CPU time of DTW, also by an order of magnitude. An attractive property of our approach is that it includes the two most used distance measures, Euclidean distance and DTW as special cases. One advantage of this fact is that it enables us to simply "slot-in" our representation to the sophisticated techniques available for indexing time series envelopes [10][17][20][33][47], thus achieving even greater speedup than shown here.

We plan to extend this work in several directions. First we intend to investigate the theoretical properties of *R-K Bands*, and the search algorithms defined on them. We also plan to consider a more generalized form of our framework, in which a single *R-K Band* is learned for an application domain. For example, what is the best *single* band for indexing George Washington's handwriting [33], does it differ from the best band for, say Isaac Newton's handwriting?

Finally, for some applications, it may be possible to examine the *R-K Bands* to glean knowledge about the domain. For example, if we learn to classify normal heartbeats versus supraventricular arrhythmias, and discover that *R-K Bands* are narrow at both ends, but wide in the center, this would suggest that the discriminating difference is contained within the T-U wave of the ECG [5].

Acknowledgments: We would like to thank Christos Faloutsos for his useful comments on this work, and thank Bhrigu Celly and Victor Zordan for helping us in creating the gun dataset. We also appreciate the help from Davide Roverso and the donors of the other datasets used in this work. This research was partly funded by the National Science Foundation under grant IIS-0237918.

Reproducible Results Statement: In the interests of competitive scientific inquiry, all datasets used in this work are available, by emailing either author.

References

[1] Aach, J. and Church, G. (2001). *Aligning gene expression time series with time warping algorithms*, Bioinformatics. Volume 17, pp. 495-508.

[2] Agrawal, R., Lin, K. I., Sawhney, H. S., & Shim, K. (1995). *Fast similarity search in the presence of noise, scaling, and translation in times-series databases*. In Proc. 21st Int. Conf. on Very Large Databases, pp. 490-501.

[3] Bar-Joseph. Z., Gerber, G., Gifford, D., Jaakkola T & Simon. I. (2002). *A new approach to analyzing gene expression time series data*. In Proceedings of the 6th Annual International Conference on Research in Computational Molecular Biology. pp. 39-48.

[4] Berndt, D. & Clifford, J. (1994). *Using dynamic time warping to find patterns in time series*. AAAI-94 Workshop on Knowledge Discovery in Databases. pp. 229-248.

[5] Caiani, E.G., Porta, A., Baselli, G., Turiel, M., Muzzupappa, S., Pieruzzi, F., Crema, C., Malliani, A. & Cerutti, S. (1998) *Warped-average template technique to track on a cycle-by-cycle basis the cardiac filling phases on left ventricular volume*. IEEE Computers in Cardiology. pp. 73-76.

[6] Chan, K. & Fu, A. W. (1999). *Efficient time series matching by wavelets*. In proceedings of the 15th IEEE Int'l Conference on Data Engineering. Sydney, Australia. pp. 126-133.

[7] Chan, K.P., Fu, A & Yu, C. (2003). *Haar wavelets for efficient similarity search of time-series: with and without time warping*. IEEE Transactions on Knowledge and Data Engineering.

[8] Chiu, B. Keogh, E., & Lonardi, S. (2003). *Probabilistic Discovery of Time Series Motifs*. In the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. August 24 - 27, 2003. Washington, DC, USA.

[9] Chu, S., Keogh, E., Hart, D., Pazzani, M (2002). *Iterative deepening dynamic time warping for time series*. In Proc 2nd SIAM International Conference on Data Mining.

[10] Clausen, M., Korner, H., & Kurth, F. (2003). *An Efficient Indexing and Search Technique for Multimedia Databases*. SIGIR Multimedia Information Retrieval Workshop 2003.

[11] Clote, P., Straubhaar, J. & Ewell (2003). *An Application of Time Warping to Functional Genomics. Unpublished report*. (EK emailed authors asking for more ref info 2003-08-14)

[12] Das, G., Lin, K., Mannila, H., Renganathan, G. & Smyth, P. (1998). *Rule discovery form time series*. Proc. of the 4th International Conference of Knowledge Discovery and Data Mining. pp. 16-22, AAAI Press.

[13] Dasgupta, D. & Forrest, S. (1999). *Novelty Detection in Time Series Data using Ideas from Immunology*. In Proceedings of The International Conference on Intelligent Systems (1999).

[14] Debregeas, A. & Hebrail, G. (1998). *Interactive interpretation of Kohonen maps applied to curves*. Proc. of the 4th International Conference of Knowledge Discovery and Data Mining. pp 179-183.

[15] Diez, J. J. R. & Gonzalez, C. A. (2000). *Applying boosting to similarity literals for time series Classification*. Multiple Classifier Systems, 1st Inter' Workshop. pp. 210-219.

[16] Faloutsos, C., Ranganathan, M., & Manolopoulos, Y. (1994). *Fast subsequence matching in time-series databases*. In Proc. ACM SIGMOD Conf., Minneapolis. pp. 419-429.

[17] Fung, W.S. & Wong, M.H. (2003). *Efficient Subsequence Matching for Sequences Databases under Time Warping*. The 7th International Database Engineering and Application Symposium. Hong Kong.

[18] Gavrila, D. M. & Davis, L. S. (1995). *Towards 3-d model-based tracking and recognition of human movement: a multi-view approach*. In International Workshop on Automatic Face- and Gesture-Recognition. pp. 272-277.

[19] Gollmer, K., & Posten, C. (1995) *Detection of distorted pattern using dynamic time warping algorithm and application for supervision of bioprocesses*. On-Line Fault Detection and Supervision in Chemical Process Industries.

[20] Hu, N., Dannenberg, R.B., & Tzanetakis, G. (2003). *Polyphonic Audio Matching and Alignment for Music Retrieval*. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA '03).

[21] Itakura, F. (1975). *Minimum prediction residual principle applied to speech recognition*. IEEE Trans. Acoustics, Speech, and Signal Proc., Vol. ASSP-23, pp. 52-72.

[22] Kadous, M. W. (1999). *Learning comprehensible descriptions of multivariate time series*. In Proc. of the 16th International Machine Learning Conference. pp. 454-463.

[23] Keogh, E. (2002). *Exact indexing of dynamic time warping*. In 28th International Conference on Very Large Data Bases. Hong Kong. pp. 406-417.

[24] Keogh, E,. Chakrabarti, K,. Pazzani, M. & Mehrotra (2000). *Dimensionality reduction for fast similarity search in large time series databases*. Journal of Knowledge and Information Systems. pp. 263-286.

[25] Keogh, E,. Chakrabarti, K,. Pazzani, M. & Mehrotra (2001). *Locally adaptive dimensionality reduction for indexing large time series databases*. In Proceedings of ACM SIGMOD Conference on Management of Data, May. pp. 151-162.

[26] Keogh, E. and Kasetty, S. (2002). *On the Need for Time Series Data Mining Benchmarks: A Survey and Empirical Demonstration*. In the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Edmonton, Canada. pp. 102-111.

[27] Keogh, E., & Pazzani, M. (2000). *Scaling up dynamic time warping for data mining applications*. In 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Boston.

[28] Kollios, G., Vlachos, M. & Gunopulos, G. (2002). *Discovering similar multidimensional trajectories*. In Proc 18th International Conference on Data Engineering.

[29] Korn, F., Jagadish, H & Faloutsos. C. (1997). *Efficiently supporting ad hoc queries in large datasets of time sequences*. In Proceedings of SIGMOD '97. pp. 289-300.

[30] Kovacs-Vajna,. Z. M. (2000). *A fingerprint verification system based on triangular matching and dynamic time warping*. IEEE transaction on pattern analysis and machine intelligence, Vol.22, No.11, November, pp. 1266-1276.

[31] Kruskall, J. B. & Liberman, M. (1983). *The symmetric time warping algorithm: From continuous to discrete*. In Time Warps, String Edits and Macromolecules. Addison-Wesley.

[32] Kwong, S. He, Q. & Man, K. (1996). *Genetic Time Warping For Isolated Word Recognition*. International Journal of Pattern Recognition and Artificial Intelligence, vol.10, no. 7, pp. 849-865

[33] Manmatha, R. & Rath, T.M. (2003). *Indexing Handwritten Historical Documents – Recent Progress*. In Proceedings of the Symposium on Document Image Understanding (SDIUT'03), pp. 77-86.

[34] Munich,. M & Perona, P. (1999). *Continuous dynamic time warping for translation-invariant curve alignment with applications to signature verification*. In Proceedings of 7th International Conference on Computer Vision, Korfu, Greece. pp. 108-115.

[35] Myers, C., Rabiner, L & Roseneberg, A. (1980). *Performance tradeoffs in dynamic time warping algorithms for isolated word recognition*. IEEE Trans. Acoustics, Speech, and Signal Proc., Vol. ASSP-28, pp. 623-635.

[36] Ng, J. & Gong, S. (2002). *Learning Intrinsic Video Content using Levenshtein Distance in Graph Partitioning*. In Proc. European Conference on Computer Vision, Volume 4, pp. 670-684, Copenhagen, Denmark, May 2002.

[37] Rabiner, L. & Juang, B. (1993). *Fundamentals of speech recognition*. Englewood Cliffs, N.J, Prentice Hall.

[38] Rabiner, L., Rosenberg, A. & Levinson, S. (1978). *Considerations in dynamic time warping algorithms for discrete word recognition*. IEEE Trans. Acoustics, Speech, and Signal Proc., Vol. ASSP-26, pp. 575-582.

[39] Rath, T. & Manmatha, R. (2002). *Word image matching using dynamic time warping*. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR'03), Vol. II, pp. 521-527.

[40] Russell S, Norvig P (1995) *Artificial Intelligence: A Modern Approach*. Prentice Hall Series in Artificial Intelligence. Englewood Cliffs, New Jersey.

[41] Sakoe, H. & Chiba, S. (1978). *Dynamic programming algorithm optimization for spoken word recognition*. IEEE Trans. Acoustics, Speech, and Signal Proc., Vol. ASSP-26. pp. 43-49.

[42] Schmill, M., Oates, T. & Cohen, P. (1999). *Learned models for continuous planning*. In 7th International Workshop on Artificial Intelligence and Statistics.

[43] Tappert, C. & Das, S. (1978). *Memory and time improvements in a dynamic programming algorithm for matching speech patterns*. IEEE Trans. Acoustics, Speech, and Signal Proc., Vol. ASSP-26, pp. 583-586.

[44] Yi, B, K. Jagadish, H & Faloutsos (1998). *Efficient retrieval of similar time sequences under time warping*. In ICDE 98, pp. 23-27.

[45] Yi, B, K., & Faloutsos, C.(2000). *Fast time sequence indexing for arbitrary Lp norms*. Proceedings of the 26st Intl Conference on Very Large Databases. pp. 385-394.

[46] Zhu, Y. & Shasha, D. (2003). *Warping Indexes with Envelope Transforms for Query by Humming*. SIGMOD 2003. pp. 181-192.

[47] Zhu, Y., Shasha, D., & Zhao, X. (2003). *Query by Humming – in Action with its Technology Revealed*. SIGMOD 2003. pp. 675.