Scaling up Dynamic Time Warping to Massive Datasets

Eamonn J. Keogh and Michael J. Pazzani

Department of Information and Computer Science
University of California, Irvine, California 92697 USA
{eamonn, pazzani}@ics.uci.edu

Abstract. There has been much recent interest in adapting data mining algorithms to time series databases. Many of these algorithms need to compare time series. Typically some variation or extension of Euclidean distance is used. However, as we demonstrate in this paper, Euclidean distance can be an extremely brittle distance measure. Dynamic time warping (DTW) has been suggested as a technique to allow more robust distance calculations, however it is computationally expensive. In this paper we introduce a modification of DTW which operates on a higher level abstraction of the data, in particular, a piecewise linear representation. We demonstrate that our approach allows us to outperform DTW by one to three orders of magnitude. We experimentally evaluate our approach on medical, astronomical and sign language data.

1 Introduction

Time series are a ubiquitous form of data occurring in virtually every scientific discipline and business application. There has been much recent work on adapting data mining algorithms to time series databases. For example, Das et al (1998) attempt to show how association rules can be learned from time series. Debregeas and Hebrail (1998) demonstrate a technique for scaling up time series clustering algorithms to massive datasets. Keogh and Pazzani (1998) introduced a new, scaleable time series classification algorithm. Almost all algorithms that operate on time series data need to compute the similarity between time series. Euclidean distance, or some extension or modification thereof, is typically used. However, Euclidean distance can be an extremely brittle distance measure. Consider the clustering produced by Euclidean distance in Fig 1. Sequence 3 is judged as most similar to the line in sequence 4, yet it appears more similar to 1 or 2.

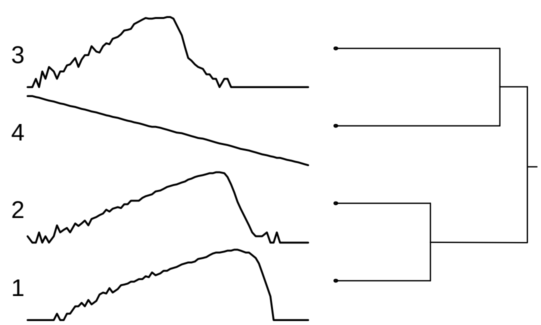

Fig. 1. An unintuitive clustering produced by the Euclidean distance measure. Sequences 1, 2 and 3 are astronomical time series (Derriere 1998). Sequence 4 is simply a straight line with the same mean and variance as the other sequences

The reason why Euclidean distance may fail to produce an intuitively correct measure of similarity between two sequences is because it is very sensitive to small distortions in the time axis. Consider Fig 2.A. The two sequences have approximately the same overall shape, but those shapes are not exactly aligned in the time axis. The nonlinear alignment shown in Fig 2.B would allow a more sophisticated distance measure to be calculated.

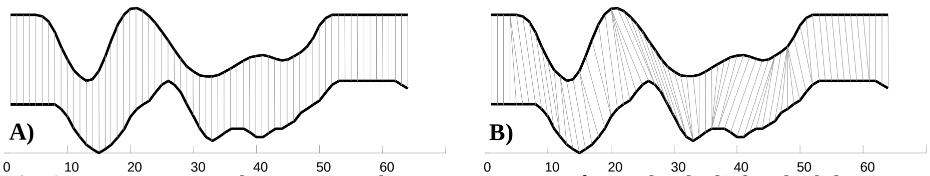

Fig. 2. Two sequences that represent the Y-axis position of an individual's hand while signing the word "pen" in Australian Sign Language. Note that while the sequences have an overall similar shape, they are not aligned in the time axis. Euclidean distance, which assumes the i^{th} point on one sequence is aligned with i^{th} point on the other (A), will produce a pessimistic dissimilarity measure. A nonlinear alignment (B) allows a more sophisticated distance measure to be calculated

A method for achieving such alignments has long been known in the speech processing community (Sakoe and Chiba 1978). The technique, Dynamic Time Warping (DTW), was introduced to the data mining community by Berndt and Clifford (1994). Although they demonstrate the utility of the approach, they acknowledge that the algorithms time complexity is a problem and that "…performance on very large databases may be a limitation".

As an example of the utility of DTW compare the clustering shown in Figure 1 with Figure 3.

In this paper we introduce a technique which speeds up DTW by a large constant. The value of the constant is data dependent but is typically one to three orders of magnitude. The algorithm, Segmented Dynamic Time Warping (SDTW), takes advantage of the fact that we can efficiently approximate most time series by a set of piecewise linear segments.

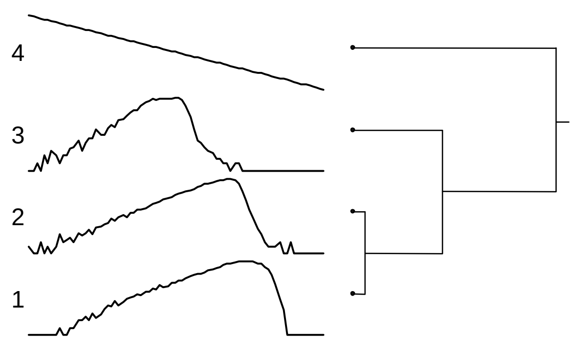

Fig 3. When the dataset used in Fig. 1 is clustered using DTW the results are much more intuitive

The rest of this paper is organized as follows. Section 2 contains a review of the classic DTW algorithm. Section 3 introduces the piecewise linear representation and SDTW algorithm. In Section 4 we experimentally compare DTW, SDTW and Euclidean distance on several real world datasets. Section 5 contains a discussion of related work. Section 6 contains our conclusions and areas of future research.

2 The Dynamic Time Warping Algorithm

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

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

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

To align two sequences using DTW we construct an n-by-m matrix where the (i^{th}, j^{th}) element of the matrix contains the distance $d(q_i,c_j)$ between the two points q_i and c_j (With Euclidean distance, $d(q_i,c_j) = (q_i - c_j)^2$). Each matrix element (i,j) corresponds to the alignment between the points q_i and c_j. This is illustrated in Figure 4. A warping path W, is a contiguous (in the sense stated below) set of matrix elements that defines a mapping between Q and C. The k^{th} element of W is defined as $w_k = (i,j)_k$ so we have:

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

The warping path is typically subject to several constraints.

- **Boundary Conditions:** $w_1 = (1,1)$ and $w_K = (m,n)$, simply stated, this requires the warping path to start and finish in diagonally opposite corner cells of the matrix.
- **Continuity:** Given $w_k = (a,b)$ then $w_{k-1} = (a',b')$ where $a–a' \leq 1$ and $b-b' \leq 1$. This restricts the allowable steps in the warping path to adjacent cells (including diagonally adjacent cells).
- **Monotonicity:** Given $w_k = (a,b)$ then $w_{k-1} = (a',b')$ where $a–a' \geq 0$ and $b-b' \geq 0$. This forces the points in W to be monotonically spaced in time.

There are exponentially many warping paths that satisfy the above conditions, however we are interested only in the path which minimizes the warping cost:

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

The K in the denominator is used to compensate for the fact that warping paths may have different lengths.

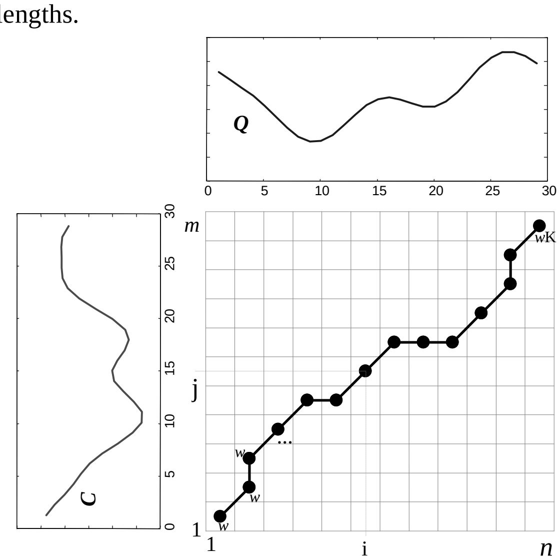

Fig. 4. An example warping path

This path can be found very efficiently 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:

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

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 $w_k = (\mathrm{i,j})_k$, i = j = k. Note that it is only defined in the special case where the two sequences have the same length.

The time complexity of DTW is O(nm). However this is just for comparing two sequences. In data mining applications we typically have one of the following two situations (Agrawal et. al. 1995).

1) **Whole Matching:** We have a query sequence *Q*, and X sequences of approximately the same length in our database. We want to find the sequence that is most similar to *Q*.

2) **Subsequence Matching:** We have a query sequence *Q*, and a much longer sequence *R* of length X in our database. We want to find the subsection of *R* that is most similar to *Q*. To find the best match we "slide" the query along *R*, testing every possible subsection of *R*.

In either case the time complexity is O(n^2X), which is intractable for many real-world problems.

This review of DTW is necessarily brief; we refer the interested reader to Kruskall and Liberman (1983) for a more detailed treatment**.**

3 Exploiting a Higher Level Representation

Because working with raw time series is computationally expensive, several researchers have proposed using higher level representations of the data. In previous work we have championed a piecewise linear representation, demonstrating that the linear segment representation can be used to allow relevance feedback in time series databases (Keogh and Pazzani 1998) and that it allows a user to define probabilistic queries (Keogh and Smyth 1997).

3.1 Piecewise Linear Representation

We will use the following notation throughout this paper. A time series, sampled at *n* points, is represented as an italicized uppercase letter such as *A*. The segmented version of A, containing *N* linear segments, is denoted as a bold uppercase letter such as **A**, where **A** is a 4-tuple of vectors of length *N*.

$$\mathbf{A} \equiv \{\mathbf{A}\mathrm{XL}, \mathbf{A}\mathrm{XR}, \mathbf{A}\mathrm{YL}, \mathbf{A}\mathrm{YR}\}$$

The i^{th} segment of sequence A is represented by the line between ($\mathbf{A}\mathrm{XL}_i$,$\mathbf{A}\mathrm{YL}_i$) and ($\mathbf{A}\mathrm{XR}_i$,$\mathbf{A}\mathrm{YR}_i$). Figure 5 illustrates this notation.

We will denote the ratio n/N as c, the compression ratio. We can choose to set this ratio to any value, adjusting the tradeoff between compactness and fidelity. For brevity we omit details of how we choose the compression ratio and how the segmented representation is obtained, referring the interested reader to Keogh and Smyth (1997) instead. We do note however that the segmentation can be obtained in linear time.

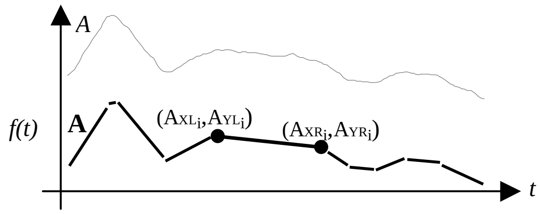

Fig. 5. We represent a time series by a sequence of straight segments

3.2 Warping with the Piecewise Linear Representation

To align two sequences using SDTW we construct an N-by-M matrix where the (i^{th}, j^{th}) element of the matrix contains the distance $d(\mathbf{Q}_i,\mathbf{C}_j)$ between the two segments $\mathbf{Q}_i$ and $\mathbf{C}_j$. The distance between two segments is defined as the square of the distance between their means:

$$d(\mathbf{Q}_i,\mathbf{C}_j) = [((\mathbf{Q}YL_i + \mathbf{Q}YR_i)/2) - ((\mathbf{C}YL_j + \mathbf{C}YR_j)/2)]^2 \tag{6}$$

Apart from this modification the matrix-searching algorithm is essentially unaltered. Equation 5 is modified to reflect the new distance measure:

$$\gamma(i,j) = d(\mathbf{Q}_i,\mathbf{C}_j) + \min\{\gamma(i\text{-}1,j\text{-}1), \gamma(i\text{-}1,j), \gamma(i,j\text{-}1)\} \tag{7}$$

When reporting the DTW distance between two time series (Eq. 4) we compensated for different length paths by dividing by K, the length of the warping path. We need to do something similar for SDTW but we cannot use K directly, because different elements in the warping matrix correspond to segments of different lengths and therefore K only approximates the length of the warping path. Additionally we would like SDTW to be measured in the same units as DTW to facilitate comparison.

We measure the length of SDTW's warping path by extending the recurrence shown in Eq. 7 to return and recursively sum an additional variable, $\max([\mathbf{Q}XR_i - \mathbf{Q}XL_i],[\mathbf{C}XR_j - \mathbf{C}XL_j])$, with the corresponding element from $\min\{\gamma(i\text{-}1,j\text{-}1), \gamma(i\text{-}1,j), \gamma(i,j\text{-}1)\}$. Because the length of the warping path is measured in the same units as DTW we have:

$$\text{SDTW}(\mathbf{Q},\mathbf{C}) \cong \text{DTW}(Q,C) \tag{8}$$

Figure 6 shows strong visual evidence that SDTW finds alignments that are very similar to those produced by DTW.

The time complexity for a SDTW is $O(MN)$, where $M = m/c$ and $N = n/c$. This means that the speedup obtained by using SDTW should be approximately c^2, minus some constant factors because of the overhead of obtaining the segmented representation.

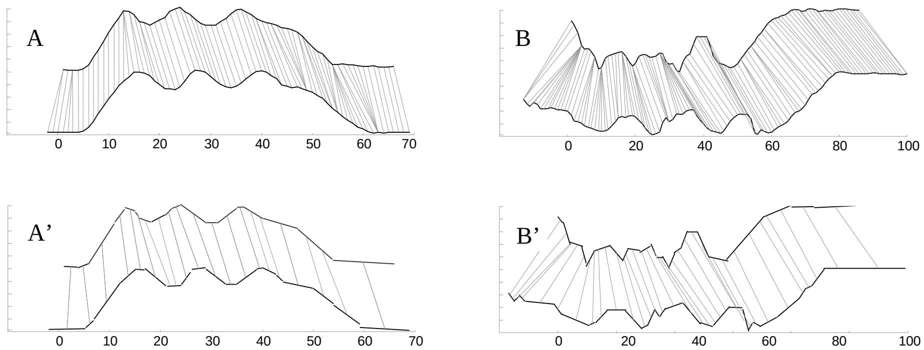

Fig. 6. **A** and **B** both show two similar time series and the alignment between them, as discovered by DTW. **A'** and **B'** show the same time series in their segmented representation, and the alignment discovered by SDTW. This presents strong visual evidence that SDTW finds approximately the same warping as DTW

4 Experimental results

We are interested in two properties of the proposed approach. The speedup obtained over the classic DTW algorithm and the quality of the alignment. In general, the quality of the alignment is subjective, so we designed experiments that indirectly, but objectively measure it.

4.1 Clustering

For our clustering experiment we utilized the Australian Sign Language Dataset from the UCI KDD archive (Bay 1999). The dataset consists of various sensors that measure the X-axis position of a subject's right hand while signing one of 95 words in Australian Sign Language (There are other sensors in the dataset, which we ignored in this work). For each of the words, 5 recordings were made. We used a subset of the database which corresponds to the following 10 words, "*spend*", "*lose*", "*forget*", "*innocent*", "*norway*", "*happy*", "*later*", "*eat*", "*cold*" and "*crazy*".

For every possible pairing of words, we clustered the 10 corresponding sequences, using group average hierarchical clustering. At the lowest level of the corresponding dendogram, the clustering is subjective. However, the highest level of the dendogram (i.e. the first bifurcation) should divide the data into the two classes. There are 34,459,425 possible ways to cluster 10 items, of which 11,025 of them correctly partition the two classes, so the default rate for an algorithm which guesses randomly is only 0.031%. We compared three distance measures:

1) **DTW**: The classic dynamic time warping algorithm as presented in Section 2.

2) **SDTW**: The segmented dynamic time warping algorithm proposed here.

3) **Euclidean**: We also tested Euclidean to facilitate comparison to the large body of literature that utilizes this distance measure. Because the Euclidean distance is only defined for sequences of the same length, and there is a small variance in the length of the sequences in this dataset, we did the following. When comparing sequences of different lengths, we "slid" the shorter of the two sequences across the longer and recorded the minimum distance.

Figure 7 shows an example of one experiment and Table 1 summarizes the results.

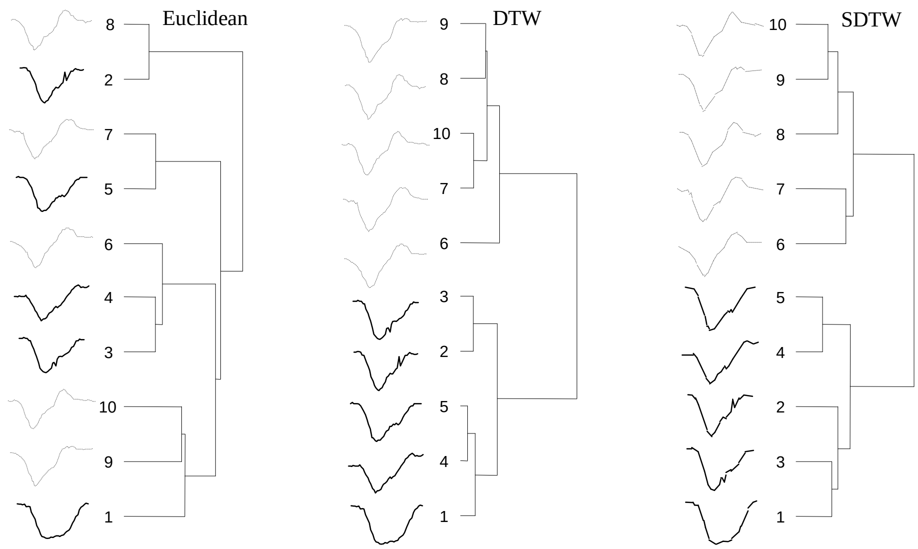

Fig. 7. An example of a single clustering experiment. The time series 1 to 5 correspond to 5 different readings of the word "*norway*", the time series 6 to 10 correspond to 5 different readings of the word "*later*". Euclidean distance is unable to differentiate between the two words. Although DTW and SDTW differ at the lowest levels of the dendrogram, were the clustering is subjective, they both correctly divide the two classes at the highest level

Distance measure	Mean Time (Seconds)	Correct Clusterings (Out of 45)
Euclidean	3.23	2
DTW	87.06	22
SDTW	4.12	21

Table 1: A comparison of three distance measures on a clustering task

Although the Euclidean distance can be quickly calculated, it performance is only slightly better than random. DTW and SDTW have essentially the same accuracy but SDTW is more than 20 times faster.

4.2 Query by Example

The clustering example in the previous section demonstrated the ability of SDTW to do whole matching. Another common task for time series applications is subsequence matching, which we consider here.

Assume that we have a query *Q* of length *n*, and a much longer reference sequence *R*, of length X. The task is to find the subsequence of *R*, which best matches *Q*, and report it's offset within *R*. If we use the Euclidean distance our distance measure, we can use an indexing technique to speedup the search (Faloutsos et. al. 1994, Keogh & Pazzani 1999). However, DTW does not obey the triangular inequality and this makes it impossible to utilize standard indexing schemes. Given this, we are resigned to using sequential search, "sliding" the query along the reference sequence repeatedly recalculating the distance at each offset. Figure 8 illustrates the idea.

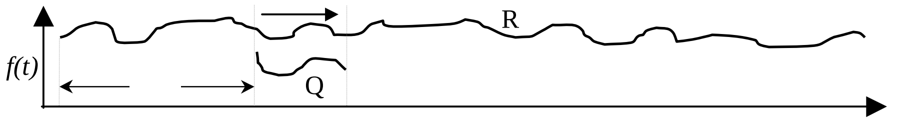

Fig. 8. Subsequence matching involves sequential search, "sliding" the query Q against the reference sequence R, repeating recalculating the distance measure at each offset.

Brendt and Clifford (1994) suggested the simple optimization of skipping every second datapoint in R, noting that as Q is slid across R, the distance returned by DTW changes slowly and smoothly. We note that sometimes it would be possible to skip much more than 1 datapoint, because the distance will only change dramatically when a new feature (i.e. a plateau, one side of a peak or valley etc.) from R falls within the query window. The question then arises of how to tell where features begin and end in R. The answer to this problem is given automatically, because the process of finding obtaining the linear segmentation can be considered a form of feature extraction (Hagit & Zdonik 1996).

We propose searching R by anchoring the leftmost segment in Q against the left edge of each segment in R. Each time we slid the query to measure the distance at the next offset, we effectively skip as many datapoints as are represented by the last anchor segment. As noted in section 3 the speedup for SDTW over DTW is approximately c^2, however this is for whole matching, for subsequence matching the speedup is approximately c^3.

For this experiment we used the EEG dataset from the UCI KDD repository (Bay 1999). This dataset contains a 10,240 datapoints. In order to create queries with objectively correct answers. We extracted a 100-point subsection of data at random, then artificially warped it. To warp a sequence we begin by randomly choosing an anchor point somewhere on the sequence. We randomly shifted the anchor point W time-units left or right (with W = 10, 20, 30). The other datapoints were moved to compensate for this shift by an amount that depended on their inverse squared distance to the anchor point, thus localizing the effect. After this transformation we interpolated the data back onto the original, equi-spaced X-axis. The net effect of this transformation is a smooth local distortion of the original sequence, as shown in Figure 9. We repeated this ten times for each for W.

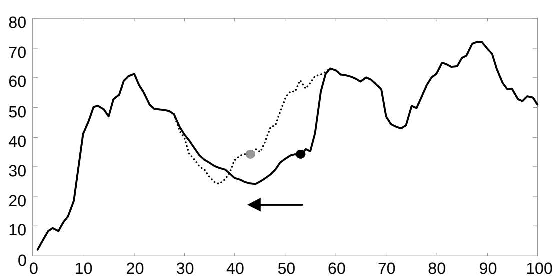

Fig. 9. An example of an artificially warped time series used in our experiments. An anchor point (black dot) is chosen in the original sequence (solid line). The anchor point is moved W units (here W = 10) and the neighboring points are also moved by an amount related to the inverse square of their distance to the anchor point. The net result is that the transformed sequence (dashed line) is a smoothly warped version of the original sequence

As before, we compared three distance measures, measuring both accuracy and time. The results are presented in Table 2.

Distance measure	Mean Accuracy ($W = 10$)	Mean Accuracy ($W = 20$)	Mean Accuracy ($W = 30$)	Mean Time (Seconds)
Euclidean	20%	0%	0%	147.23
DTW	100%	90%	60%	15064.64
SDTW	100%	90%	50%	26.16

Table 2: A comparison of three distance measures on query by example

Euclidean distance is fast to compute, but its performance degrades rapidly in the presence of time axis distortion. Both DTW and SDTW are able to detect matches in spite of warping, but SDTW is approximately 575 times faster.

5 Related Work

Dynamic time warping has enjoyed success in many areas where it's time complexity is not an issue. It has been used in gesture recognition (Gavrila & Davis 1995), robotics (Schmill et. al 1999), speech processing (Rabiner & Juang 1993), manufacturing (Gollmer & Posten 1995) and medicine (Caiani et. al 1998).

Conventional DTW, however, is much too slow for searching large databases. For this problem, Euclidean distance, combined with an indexing scheme is typically used. Faloutsos et al, (1994) extract the first few Fourier coefficients from the time series and use these to project the data into multi-dimensional space. The data can then be indexed with a multi-dimensional indexing structure such as a R-tree. Keogh and Pazzani (1999) address the problem by de-clustering the data into bins, and optimizing the data within the bins to reduce search times. While both these approaches greatly speed up query times for Euclidean distance queries, many real world applications require non-Euclidean notions of similarity.

The idea of using piecewise linear segments to approximate time series dates back to Pavlidis and Horowitz (1974). Later researchers, including Hagit and Zdonik (1996) and Keogh and Pazzani (1998) considered methods to exploit this representation to support various non-Euclidean distance measures, however this paper is the first to demonstrate the possibility of supporting time warped queries with linear segments.

6 Conclusions and Future Work

We demonstrated a modification of DTW that exploits a higher level representation of time series data to produce one to three orders of magnitude speed-up with no appreciable decrease in accuracy. We experimentally demonstrated our approach on several real world datasets.

Future work includes a detailed theoretical examination of SDTW, and extensions to multivariate time series.

References

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 *VLDB*, September.

Bay, S. (1999). UCI Repository of Kdd databases [http://kdd.ics.uci.edu/]. Irvine, CA: University of California, Department of Information and Computer Science.

Berndt, D. & Clifford, J. (1994) Using dynamic time warping to find patterns in time series. *AAAI-94 Workshop on Knowledge Discovery in Databases (KDD-94)*, Seattle, Washington.

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. Vol. 25 Cat. No.98CH36292, NY, USA.

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

Debregeas, A. & Hebrail, G. (1998). Interactive interpretation of Kohonen maps applied to curves. *Proceedings of the 4rd International Conference of Knowledge Discovery and Data Mining*. pp 179-183, AAAI Press.

Derriere, S. (1998) D.E.N.I.S strip 3792: [http://cdsweb.u-strasbg.fr/DENIS/qual_gif/cpl3792.dat]

Faloutsos, C., Ranganathan, M., & Manolopoulos, Y. (1994). Fast subsequence matching in time-series databases. In *Proc. ACM SIGMOD Conf.*, Minneapolis, May.

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*. IEEE Computer Society, Zurich.

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 the Chemical Process Industries (Edited by: Morris, A.J.; Martin, E.B.).

Hagit, S., & Zdonik, S. (1996). Approximate queries and representations for large data sequences. *Proc. 12th IEEE International Conference on Data Engineering*. pp 546-553, New Orleans, Louisiana, February.

Keogh, E., & Pazzani, M. (1998). An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback. *Proceedings of the 4[rd] International Conference of Knowledge Discovery and Data Mining*. pp 239-241, AAAI Press.

Keogh, E., & Pazzani, M. (1999). An indexing scheme for fast similarity search in large time series databases. To appear in *Proceedings of the 11[th] International Conference on Scientific and Statistical Database Management.*

Keogh, E., Smyth, P. (1997). A probabilistic approach to fast pattern matching in time series databases. *Proceedings of the 3[rd] International Conference of Knowledge Discovery and Data Mining*. pp 24-20, AAAI Press.

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

Pavlidis, T., Horowitz, S. (1974). Segmentation of plane curves. IEEE Transactions on Computers, Vol. C-23, NO 8, August.

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

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

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