ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/216301292

Everything you know about dynamic time warping is wrong

Conference Paper · January 2004

CITATIONS
271

READS
15,639

2 authors, including:

Chotirat Ann Ratanamahatana
Chulalongkorn University
82 PUBLICATIONS **5,391** CITATIONS

SEE PROFILE

All content following this page was uploaded by Chotirat Ann Ratanamahatana on 05 June 2014.

The user has requested enhancement of the downloaded file.

Everything you know about Dynamic Time Warping is Wrong

Chotirat Ann Ratanamahatana Eamonn Keogh

Department of Computer Science and Engineering
University of California, Riverside
Riverside, CA 92521

{ ratana, eamonn }@cs.ucr.edu

ABSTRACT

The Dynamic Time Warping (DTW) distance measure is a technique that has long been known in speech recognition community. It allows a non-linear mapping of one signal to another by minimizing the distance between the two. A decade ago, DTW was introduced into Data Mining community as a utility for various tasks for time series problems including classification, clustering, and anomaly detection. The technique has flourished, particularly in the last three years, and has been applied to a variety of problems in various disciplines.

In spite of DTW's great success, there are still several persistent "myths" about it. These myths have caused confusion and led to much wasted research effort. In this work, we will dispel these myths with the most comprehensive set of time series experiments ever conducted.

Keywords

Dynamic Time Warping. Data Mining. Experimentation.

1. INTRODUCTION

In recent years, classification, clustering, and indexing of time series data have become a topic of great interest within the database/data mining community. The Euclidean distance metric has been widely used [17], in spite of its known weakness of sensitivity to distortion in time axis [15]. A decade ago, the Dynamic Time Warping (DTW) distance measure was introduced to the data mining community as a solution to this particular weakness of Euclidean distance metric [3]. This method's flexibility allows two time series that are similar but locally out of phase to align in a non-linear manner. In spite of its $O(n^2)$ time complexity, DTW is the best solution known for time series problems in a variety of domains, including bioinformatics [1], medicine [5], engineering, entertainment [30], etc.

The steady flow of research papers on data mining with DTW became a torrent after it was shown that a simple lower bound allowed DTW to be indexed with no false dismissals [15]. The lower bound requires that the two sequences being compared are of the same length, and that the amount of warping is constrained. This work allowed practical applications of DTW, including real-time query-by-humming systems [30], indexing of historical handwriting archives [24], and indexing of motion capture data [6].

In spite of the great success of DTW in a variety of domains, there still are several persistent myths about it. These myths have caused great confusion in the literature, and led to the publication of papers that solve apparent problems that do not actually exist. The three major myths are:

Myth 1: *The ability of DTW to handle sequences of different lengths is a great advantage, and therefore the simple lower bound that requires different-length sequences to be reinterpolated to equal length is of limited utility* [18][27][28]. In fact, as we will show, there is no evidence in the literature to suggest this, and extensive empirical evidence presented here suggests that comparing sequences of different lengths and reinterpolating them to equal length produce no statistically significant difference in accuracy or precision/recall.

Myth 2: *Constraining the warping paths is a necessary evil that we inherited from the speech processing community to make DTW tractable, and that we should find ways to speed up DTW with no (or larger) constraints*[27]. In fact, the *opposite* is true. As we will show, the 10% constraint on warping inherited blindly from the speech processing community is actually too large for real world data mining.

Myth 3: *There is a need (and room) for improvements in the speed of DTW for data mining applications*. In fact, as we will show here, if we use a simple lower bounding technique, DTW is essentially $O(n)$ for data mining applications. At least for CPU time, we are almost certainly at the asymptotic limit for speeding up DTW.

In this paper, we dispel these DTW myths above by empirically demonstrate our findings with a comprehensive set of experiments. In terms of number of objective datasets and size of datasets, our experiments are orders of magnitude greater than anything else in the literature. In

particular, our experiments required more than eight billion DTW comparisons.

Before beginning our deconstruction of these myths, it would be remiss of us not to note that several early papers by the second author are guilty of echoing them. This work is part of an effort to redress these mistakes. Likewise, we have taken advantage of the informal nature of a workshop to choose a tongue-in-cheek attention grabbing title. We do not really mean to imply that the entire community is ignorant of the intricacies of DTW.

The rest of the paper is organized as follows. In Section 2, we give an overview of Dynamic Time Warping (DTW) and its related work. The next three sections consider each of the three myths above. Section 6 suggests some avenues for future researches, and Section 7 gives conclusions and directions for future work. Because we are testing on a wide range of real and synthetic datasets, we have placed the details about them in Appendix A to enhance the flow of the paper.

2. BACKGROUND AND RELATED WORK

The measurement of similarity between two time series is an important subroutine in many data mining applications, including classification [11][14], clustering [1][10], anomaly detection [9], rule discovery [8], and motif discovery [7]. The superiority of DTW over Euclidean distance metric for these tasks has been demonstrated by many authors [1][2][5][29]. We will first begin with a review of some background material on DTW and its recent extensions, which contributes to our main motivation of this paper.

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 first 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 retrieve a path through the matrix that minimizes the total cumulative distance between them as illustrated in Figure 1. In particular, the optimal path is the path that minimizes the warping cost

$$DTW(Q,C) = \min\left\{\sqrt{\sum_{k=1}^{K} w_k}\right\} \quad (3)$$

where w_k is the matrix element $(i,j)_k$ that also belongs to k^{th} element of a warping path W, a contiguous set of matrix elements that represent a mapping between Q and C.

This warping path can be found using dynamic programming to evaluate the following recurrence.

$$\gamma(i,j) = d(q_i,c_j) + \min\{\gamma(i-1,j-1), \gamma(i-1,j), \gamma(i,j-1)\} \quad (4)$$

where $d(i,j)$ is the distance found in the current cell, and $\gamma(i,j)$ is the cumulative distance of $d(i,j)$ and the minimum cumulative distances from the three adjacent cells.

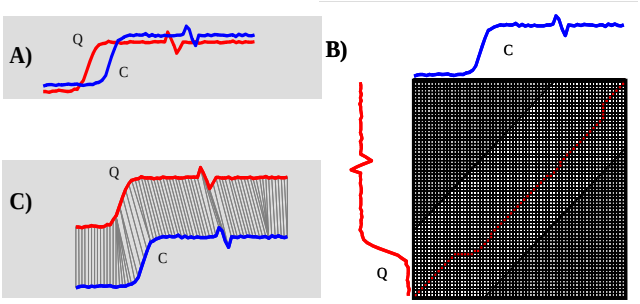

Figure 1. A) Two similar sequences *Q* and *C*, but out of phase. B) To align the sequences, we construct a warping matrix and search for the optimal warping path, shown with solid squares. Note that the 'corners' of the matrix (shown in dark gray) are excluded from the search path as part of an Adjustment Window condition. C) The resulting alignment.

To reduce the number of paths to consider during the computation, several well-known constraints (*Boundary Conditions*, *Continuity condition*, *Monotonic condition*, *and Adjustment Window Condition)* have been applied to the problem to restrict the moves that can be made from any point in the path and so restrict the number of paths that need to be considered. Figure 1 B) illustrates a particular example of the Adjustment Window Condition (or Warping Window Constraints) with the Sakoe-Chiba Band [26]. The width of this constraint is often set to 10% of the length of the time series [1][22][26].

2.2 LOWER BOUNDING THE DTW DISTANCE

A recent extension to DTW that significantly speeds up the DTW calculation is a lower bounding technique based on the warping window (envelope) [15].

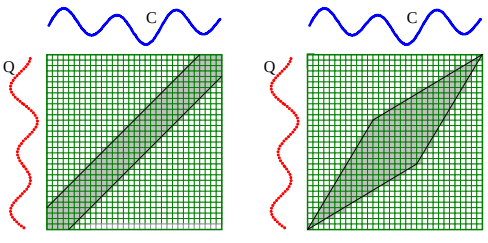

Figure 2. The two most common constraints in the literature are the Sakoe-Chiba Band and the Itakura Parallelogram

Figure 2 illustrates two of the most frequently used global constraints in the literature, the Sakoe-Chiba Band [26] and

the Itakura Parallelogram [13]. The latter is widely used in the speech community.

The lower bound is only defined for sequences of the same length; if the sequences are of different lengths, one of them must be reinterpolated. This lower bounding technique uses the warping window 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.

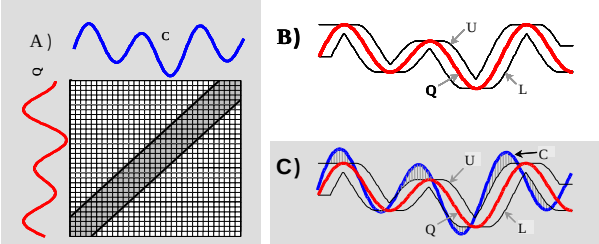

Figure 3. The Sakoe-Chiba Band A) can be used to create an envelope B) around a query sequence *Q*. The Euclidean distance between any candidate sequence *C* and the closest external part of the envelope C) is a lower bound for the DTW distance

For clarity, in Table 1, we will show a trivial algorithm that can exploit *any* lower bound to do faster sequential search. This algorithm is taken from Table 2 of [15].

Table 1. An algorithm that uses a lower bounding distance measure to speed up the sequential scan search for the query Q

```
Algorithm Lower_Bounding_Sequential_Scan(Q)
1.  best_so_far = infinity;
2.  for all sequences in database
3.   LB_dist = lower_bound_distance(Ci, Q);
4.      if LB_dist < best_so_far
5.          true_dist = DTW(Ci, Q);
6.          if true_dist < best_so_far
7.              best_so_far = true_dist;
8.              index_of_best_match = i;
9.          endif
10.     endif
11. endfor
```

Note that the tightness of these lower bounds and pruning power essentially depend on the size of the warping window used as well. In general, the smaller the area of allowed warping, the more we can take advantage of pruning. As noted above, the best size of this warping is subject to controversy; we will examine the question in Section 4.

For clarity, we will summarize before continuing. If we are willing to force the sequences to be of the same length, and to constrain the warping, then we have a simple solution for speeding up similarity search under DTW. We will call this solution **4S** (Simple Straw man for Similarity Search). As we shall see, the papers that try to speed up this simple approach, or relax its two assumptions, are motivated and misled by the myths discussed above.

3. DOES COMPARING SEQUENCES OF DIFFERENT LENGTHS HELP OR HURT?

Many recent papers suggest that the ability of classic DTW to deal directly with sequences of different length is a great advantage; some paper titles even contain the phrase "*…of different lengths"* [4][21] showing their great concerns in solving this issue. As further examples, consider the following quotes taken from recent papers: "*Time warping enables sequences with similar patterns to be found even when they are of different lengths*" [18], or " *(DTW is) a more robust distance measure than Euclidean distance in many situations, where sequences may have different lengths*" [28] or "*(DTW) can be used to measure similarity between sequences of different lengths*". Some of these papers further suggest that the simple **4S** solution to DTW similarity search is not useful because it requires that sequences of different lengths to be reinterplolated to the same length, and use this fact to motive new approaches: for example "***(4S)** only works when the data and query sequences are of the same length."* [27].

These claims are surprising in that they are not supported by any empirical results in the papers in question. Furthermore, an extensive literature search through more than 500 papers dating back to the 1960's failed to produce any theoretical or empirical results to suggest that simply making the sequences to be of the same length has any detrimental effect.

To test our claimed hypothesis that there is no significant difference in accuracies between using variable-length time series and equal-length time series in DTW calculation, we carry out an experiment as follows.

For all variable-length time series datasets (Face, Leaf, Trace, and Wordspotting – See Appendix A for dataset details), we compute 1-nearest-neightbor classification accuracies (leaving-one-out) using DTW for all warping window sizes (1% to 100%) in two different ways:

1) The **4S** way; we simply reinterpolated the sequences to have the same length.
2) By comparing the sequences directly using their original lengths.

The latter case is not as simple as one might think since we need to normalize the returned distance in some way. All things being equal, we would expect longer sequences to be further apart than short sequences, since they have more dimensions on which to accumulate noise. The following normalization policies have appeared in the literature or are common sense ideas.

- No normalization on the distance.
- Normalize by the length of the optimal warping path.
- Normalize by the length of the shorter time series (for each pair of the time series during each DTW computation).
- Normalized by the length of the longer time series.

To give the benefit of the doubt to different-length case, for each warping window size, we do all four possible normalizations above, and the best performing of the four options is recorded as the accuracy for the variable-length DTW calculation.

For completeness, we test over every possible warping constraint size. Note that we start the warping window size of 1% instead of 0% since 0% size is Euclidean distance metric, which is undefined when the time series are not of the same length. Also, when measuring the DTW distance between two time series of different lengths, the percentage of warping window applied is based on the length of the longer time series to ensure that we allow adequate amount of warping for each pair and deliver a fair comparison.

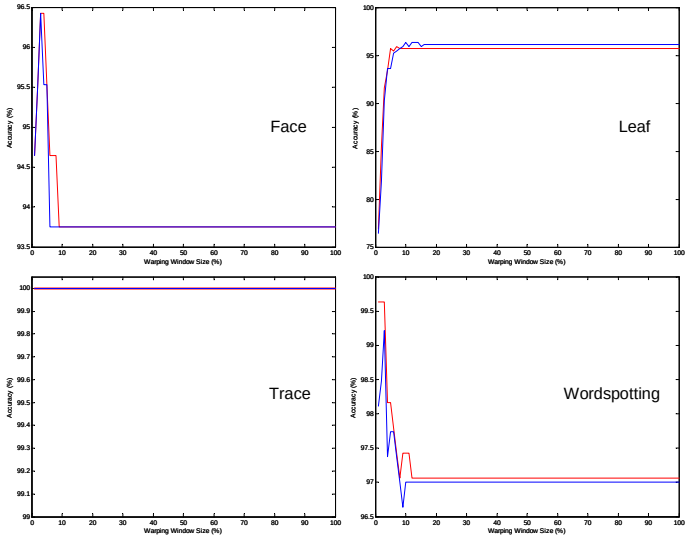

Figure 4. A comparison of the classification accuracies between variable-length datasets (dotted lines) and the (reinterpolated) equal-length datasets (solid lines). The two options produce such similar results that in many places the lines overlap.

The variable-length datasets are then linearly reinterpolated to have the same length of the longest time series within each dataset. After that, we simply compute the classification accuracies using DTW for all warping window sizes (1% to 100%) for each dataset. The results are shown in Figure 4.

Note that the experiments do strongly suggest that changing the amount of warping allowed does affect the accuracy (an issue that will be discussed in depth in the next section), but over the entire range on possible warping widths, the two approaches are nearly indistinguishable. Furthermore, a two-tailed test using a significance level of 0.05 between each variable-length and equal-length pair indicates that there is no statistically significant difference between the accuracy of the two sets of experiments. An even more telling result is the following. In spite of extensive experience with DTW and an extensive effort, we were unable to create an artificial problem where reinterpolating made a significant difference in accuracy. To further reinforce our claim, we also reinterpolate the datasets to have the equal length of the shortest and averaged length of all time series within the dataset. We still achieve similar findings.

These results strongly suggest that work allowing DTW to support similarity search that does require reinterpolation, is simply solving a problem that does not exist. Subsequently, while Wong and Wong claimed, "*(DTW is useful) to measure similarity between sequences of different lengths*" [28] we must recall that two *Wongs* do not make a right[1]. The often-quoted utility of DTW being able to support the comparison of sequences of different lengths is simply a myth.

4. ARE NARROW CONSTRAINTS BAD?

Apart from (slightly) speeding up the computation, warping window constraints were originally applied mainly to prevent pathological warping (where a relatively small section of one sequence maps to a much larger section of another). The vast majority of the data mining researchers have used a Sakoe-Chiba Band with a 10% width for the global constraint [1][22][26]. This setting seems to be the result of historical inertia, inherited from the speech processing community, rather than some remarkable property of this particular constraint.

Some researchers believe that having wider warping window contributes to improvement in accuracy [30]. Or without realizing the great effect of the warping window size on accuracies, some applied DTW with no warping window constraints [20], or did not specify the window size used in the experiments [19] (the latter case makes it particularly difficult for others to reproduce the experiment results). In [27], the authors bemoan the fact that "*(4S) cannot be applied when the warping path is not constrained*" and use

[1] Yes, this is a very poor joke!

this fact to justify introducing an alterative approach that works for the unconstrained case.

To test the effect of the warping window size to the classification accuracies, we performed an empirical experiment on all seven classification datasets. We vary the warping window size from 0% (Euclidean) to 100% (no constraint/full calculation) and record the accuracies.

Since we have shown in Section 3 that reinterpolation of time series into the same length is at least as good as (or better than) using the original variable-length time series, we linearly interpolate all variable-length datasets to have the same length of the longest time series within the dataset and measure the accuracy using the 1-nearest-neighbor with leaving-one-out classification method. The results are shown in Figure 5.

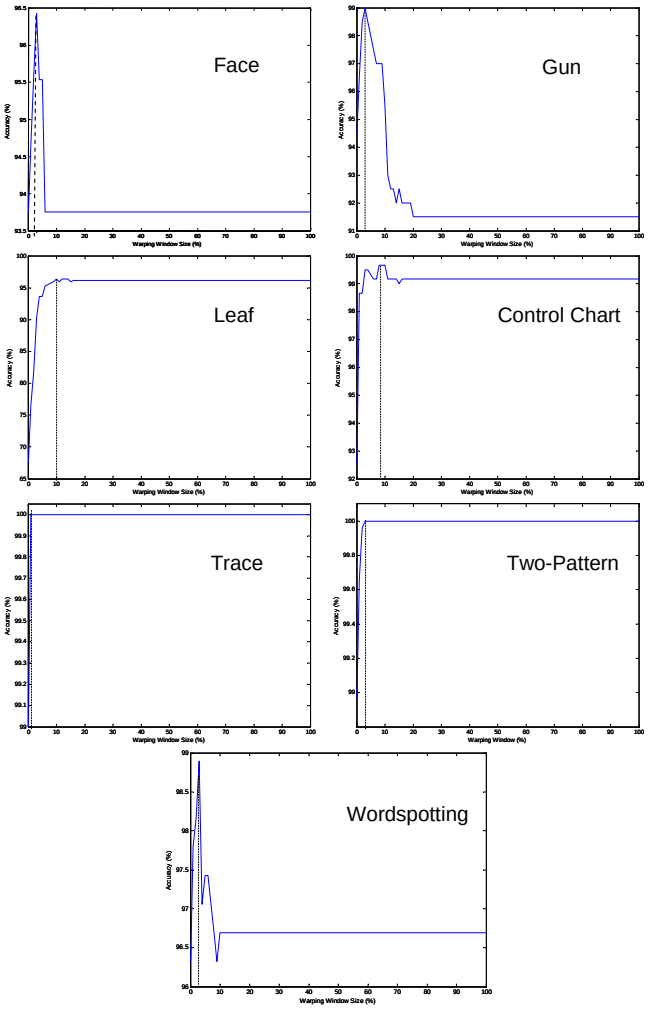

Figure 5. The classification accuracies for all warping window sizes. All accuracies peak at very small window sizes.

As we hypothesized, wider warping constraints do not always improve the accuracy, as commonly believed [30]. More often, the accuracy peaks very early at much smaller window size, as shown in Table 2 below. In essence, the results can be summarized by noting that a little warping is a good thing, but too much warping is a bad thing.

Table 2. The warping window size that yields maximum classification accuracy for each dataset, using DTW with Sakoe-Chiba Band.

Dataset	Max Accuracy (%)	Warping Window Size (%)
Face	96.43	3
Gun	99.00	3
Leaf	96.38	10
Syn_contrl_chrt	99.67	8
Trace	100.00	1
TwoPatterns	100.00	3
Wordspotting	98.90	3

We also did an additional experiment, where half of the objects in the databases are randomly removed from the database in each iteration. We measure the classification accuracies for each database size, as shown in Figure 6. As the database size decreases, the classification accuracy also declines and the peak appears at larger warping window size.

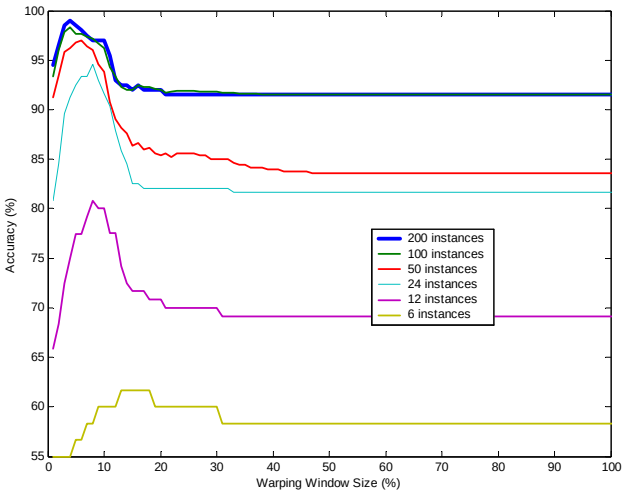

Figure 6. With fewer objects in the databases, the accuracies become less accurate and peak at larger window size

This finding suggests that warping window size adjustment does affect accuracy, and that the effect also depends on the database size. This in turn suggests that we should find the best warping window size on realistic (for the task at hand) database sizes, and not try to generalize from toy problems.

To summarize, there is no evidence to support the idea that we need to be able to support wider constraints. While it is possible that there exist some datasets somewhere that

could benefit from wider constraints, we found no evidence for this in a survey of more than 500 papers on the topic. More tellingly, in spite of extensive efforts, we could not even create a large synthetic dataset for classification that needs more than 10% warping.

All the evidence suggests that narrow constraints are *necessary* for accurate DTW, and the "need" to support wide (or no) constraints is just a myth.

5. CAN DTW BE FURTHER SPEEDED UP?

Smaller warping windows speed up the DTW calculations simply because there is less area of the warping matrix to be searched. Prior to the introduction of lower bounding, the amount of speedup was directly proportional to the width of the warping window. For example, a nearest neighbor search with a 10% warping constraint was almost exactly twice as fast as a search done with a 20% window. However, it is important to note that with the introduction of lower bounding based on warping constraints (i.e. **4S**), the speedup is now highly nonlinear in the size of the warping window. For example, a nearest neighbor search with a 10% warping constraint may be many times faster than twice a search done with a 20% window.

In spite of this, many recent papers still claim that there is a need and room for further improvement in speeding up DTW. For example, a recent paper suggested "*dynamic time warping incurs a heavy CPU cost…*" Surprisingly, as we will now show, the amortized CPU cost of DTW is essentially O(*n*) if we use the trivial **4S** technique.

To really understand what is going on, we will avoid measuring the efficiency of DTW when using index structures. The use of such index structures opens the possibility of implementation bias [17]; it is simply difficult to know if the claimed speedup truly reflects a clever algorithm, or simply the care in choice of buffer size, caching policy, etc.

Instead, we measure the computation time of DTW for each pair of time series in terms of the amortized percentage of the warping matrix that needs to be visited for each pair of sequences in our database. This number depends only on the data itself and the usefulness of the lower bound. As a concrete example, if we are doing a one nearest neighbor search on 120 objects with a 10% warping window size, and the **4S** algorithm only needs to examine 14 sequences (pruning the rest), then the amortized cost for this calculation would be (w * 14) / 120 = 0.12*w, where w is the area (in percentage) inside the warping window constraint along the diagonal (Sakoe-Chiba band). Note that 10% warping window size does not always occupy 10% of the warping matrix; it mainly depends on the length of the sequence as well (longer sequences give smaller w). In contrast, if **4S** was able to prune all but 3 objects, the amortized cost would be (w * 3) / 120 = 0.03*w.

The amount of pruning we should actually expect depends on the lower bounds. For example, if we used a trivial lower bound hard-coded to zero (pointless, but perfectly legal), then line 4 of Table 1 would always be true, and we would have to do DTW for every pair of sequences in our dataset. In this case, amortized percentage of the warping matrix that needs to be accessed for each sequence in our database would exactly be the area inside the warping window. If, on the other hand, we had a "magic" lower bound that returned the true DTW distance minus some tiny epsilon, then line 4 of the Table 1 would rarely be true, and we would have to do the full DTW calculation only rarely. In this case, the amortized percentage of the warping matrix that needs to be accessed would be very close to zero.

We measured the amortized cost for all our datasets, and for every possible warping window size. The results are shown in Figure 7. Figure 8 shows the zoom-in of the results from 0 to 10 % warping window size

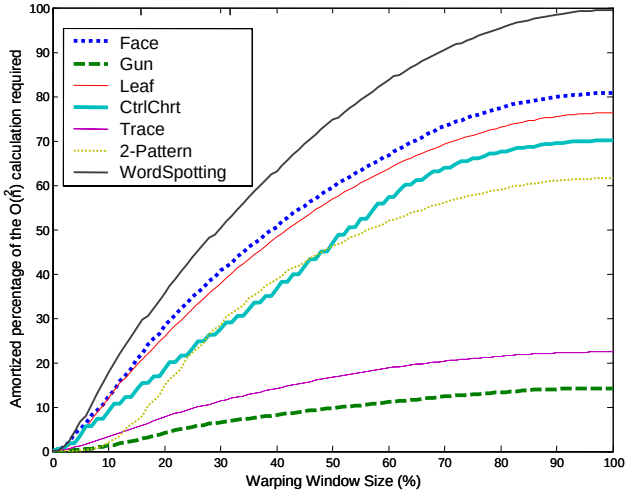

Figure 7. The amortized percentage of warping matrix that needs to be accessed during the DTW calculation for each warping window size. The use of a lower bound helps prune off numerous unnecessary calculations.

The results are very surprising. For reasonably large datasets, simply using a good lower bound insures that we rarely have to use the full DTW calculation. In essence, we can say that DTW is effectively O(n), and not O(n^2), when searching large datasets.

For example, in the Gun, Trace, and 2-Pattern problems (all maximum accuracy at 3% warping), we only need to do much less than half a percent of the O(n^2) work that we would have been forced to do without lower bounding. For some of the other datasets, it may appear that we need to do a significant percentage of the CPU work. However, as we will see below, these results are pessimistic in that they reflect the small size of these datasets.

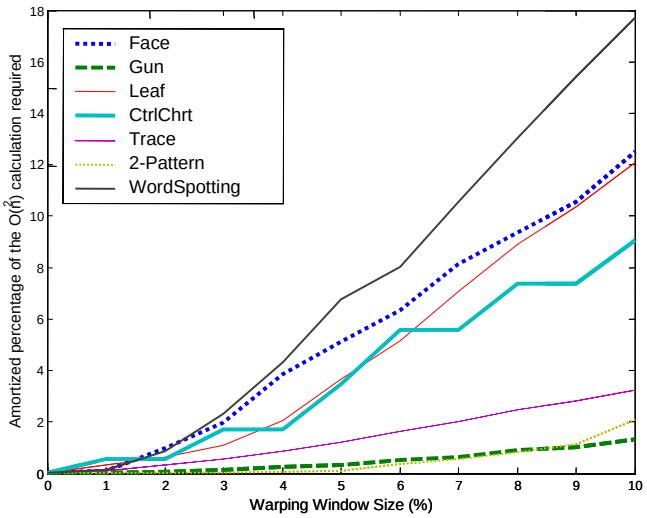

Figure 8. Zoom-in of Figure 6, from 0 to 10% warping window size. Note that from Section 4, we recognize that the most accurate results for all datasets happen in this range.

If the amortized cost of DTW is linear, where does the claimed improvement from recent papers come from? It is true that these approaches typically use indexes, rather than sequential search, but an index must do costly random access rather than the optimized linear scans of sequential search. In order to simply break even in terms of disk access time, they must avoid looking at more than 10% of the data [16], but for time series where even the reduced dimensionality (i.e. the Fourier or wavelet coefficients) is usually greater than 20 [17], it is not obvious that this is possible.

Some recent papers that claim speedups credit the improved lower bounds, for example "*…we present progressively tighter lower bounds… that allow our method to outperform (4S)* " [27]. Indeed, it might be imagined that speedup could be obtained by having tighter lower bounds. Surprisingly, this is not true! We can see this with the following simple experiment. Let us imagine that we have a wonderful lower bound, which always returns a value that is within 1% of the correct value (more concretely, a value uniformly distributed between 99% and 100% of the true DTW value). We will call this idealized lower bound *LB_Magic*. In contrast, the current best-known lower bounds typically return a value between 40% and 60% of the true value [15].

We can compare the speedup obtained by LB_Magic with the current best lower bound, LB_Keogh [15], on 1-nearest neighbor search. Note that we have to cheat for LB_Magic by doing the full DTW calculation then assigning it a value up to 1% smaller. We will use a warping constraint of 5%, which is about the mean value for the best accuracy (cf. Section 4). As before, we measured the amortized percentage of the warping matrix that needs to be accessed for each sequence in our database. For this experiment, we use a *randomwalk* data of length 128 data points, and vary the database size from 10 objects to 40,960 objects. Figure 9 shows the results.

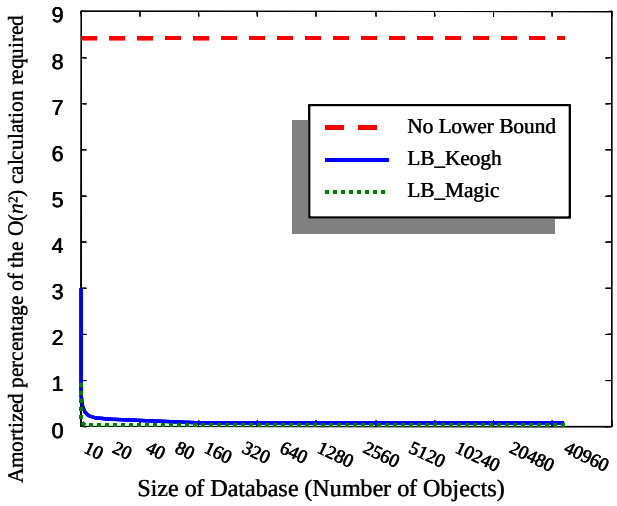

Figure 9. Amortized percentage of the warping matrix that needs to be accessed. As the size of the database is increasing, the amortized percentage of the warping matrix accessed becomes closer to zero.

Once again, the results are very surprising. The idealized LB_Magic allows a very impressive speedup; for the largest size database, it eliminates 99.997% of the CPU effort. However, the very simple lower bounding technique that has been in the literature for several years is able to eliminate 99.369% of the CPU effort! The difference is not quite so dramatic for very small datasets, say less than 160 objects. But here we can do unoptimized search in much less than a hundredth of a second. Note that we obtain similar results for other datasets.

To summarize, for problems involving a few thousand sequences or more, each with a few hundred data points, the "*significant CPU cost of DTW*" is simply non-issue (as for problems involving less than a few thousand sequences, we can do them in less than a second anyway).

The lesson for the data mining community from this experiment is the following; it is almost certainly pointless to attempt to speed up the CPU time for DTW by producing tighter lower bounds. Even if you could produce a magical lower bound, the difference it would make would be tiny, and completely dwarfed by minor implementation choices.

6. AVENUES FOR FUTURE RESEARCH

In this section, we will attempt to redress the somewhat negative tone of this paper by suggesting many avenues for future research with DTW. Since it has been demonstrated by many authors that DTW is the solution to many data mining problems, we would like to present some of the

other applications or problems that can effectively benefit from DTW distance measure.

6.1 VIDEO RETRIEVAL

Generally, research on content-based video retrieval represents the content of the video as a set of frames, leaving out the temporal features of frames in the shot. However, for some domains, including motion capture editing, gait analysis, and video surveillance, it may be fruitful to extract time series from the video, and index *just* the time series (with pointers back to the original video). Figure 10 shows an example of a video sequence that is transformed into a time series. This example is the basis for the Gun dataset discussed in Appendix A.

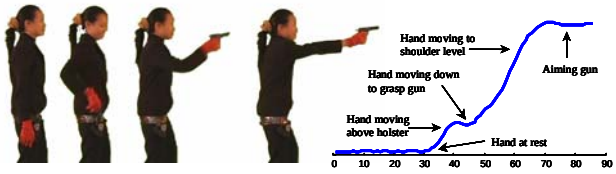

Figure 10. Stills from a video sequence; the right hand is tracked, and converted into a time series

One obvious reason why using time series representation may be superior to working with the original data is the massive reduction in dimensionality, which enhances the ease of storage, transmission, analysis, and indexing. Moreover, it is much easier to make the time series representation invariant to distortions in the data, such as time scaling and time warping.

6.2 IMAGE RETRIEVAL

For some specialized domains, it can be useful to convert the images into "pseudo time series". For example, consider Figure 11 Below. Here, we have converted an image of a leaf into a time series by measuring the local angle of a trace of its perimeter. The utility of such a transform is similar to that for video retrieval.

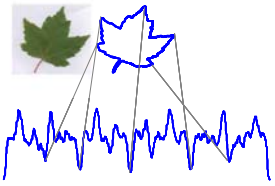

Figure 11. An example of a leaf image converted into a "pseudo time series"

6.3 HANDWRITING RETRIEVAL

The problem of transcribing and indexing existing historical archives is still a challenge. For even such a major historical figure as Isaac Newton, there exists a body of unpublished, handwritten work exceeding one million words. For other historical figures, there are even larger collections of handwritten text. Such collections are potential goldmines for researchers and biographers. Recent work by [24] suggests that DTW may be best solution to this problem.

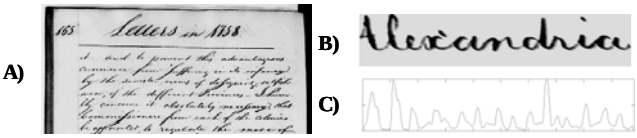

Figure 12. A) An example of handwritten text by George Washington. B) A zoom-in on the word "Alexandria", after being processed to remove slant. C) Many techniques exist to convert 2-D handwriting into a time series; in this case, the projection profile is used (Fig. created by R. Manmatha).

6.4 TEXT MINING

Surprisingly, we can also transform text into a time series representation. For instance, we consider a problem of translating biblical text in two different languages (English and Spanish). The bible text is converted into bit streams according to the occurrences of the chosen word in the text. For example, subsection of the bible containing the word 'God' in "In the beginning God created the heaven and the earth" will be represented by "0001000000". Then the bit streams are converted into time series by recording the number of word occurrences within the predefined sliding window.

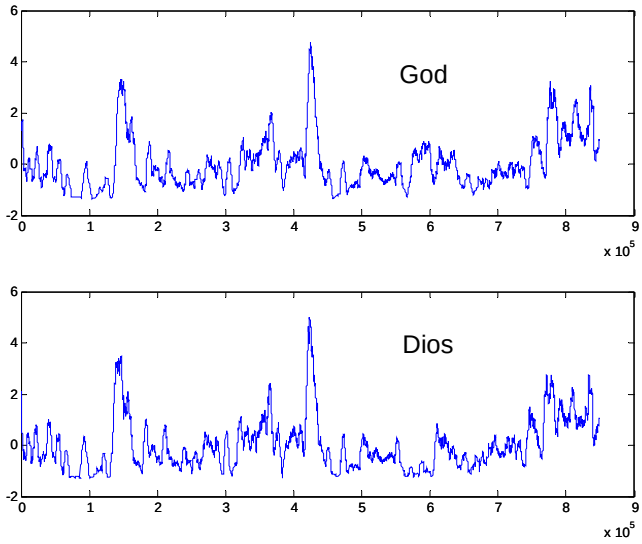

Figure 13. Times series of the number of occurrences of the word 'God' in English (top) and 'Dios' in Spanish (bottom) bible text using 6,000 words as the window size (z-normalized and reinterpolated to the same length). The two time series are almost identical.

The intuition behind this approach is that for each appearance of each word in English, there must be a corresponding Spanish word that also appears in the same vicinity in the Spanish bible text. However, there can be some discrepancies in the number of words in the entire text as well as the position of the word within the sentence

between the two languages. This can be overcome by DTW technique.

These suggestions are only subsets of variety of problems that could benefit from transforming them into time series, which DTW could be trivially applied to. There are still considerable amount of applications that are left to be explored.

7. CONCLUSIONS AND FUTURE WORK

In this work, we have pointed out and investigated some of the myths in Dynamic Time Warping measure. We empirically validated our three claims.

We hope that our results will help researchers focus on more useful problems. For example, while there have been dozens of papers on speeding up DTW in the last decade, there has only been one on making it more accurate [23]. Likewise, we feel that the speed and accuracy of DTW that we have demonstrated in this work may encourage researchers to apply DTW to a wealth of new problems/domains.

8. ACKNOWLEDGMENTS

All datasets used in this paper are available upon request, by emailing either author. We thank Yasushi Sakurai for his useful comments. We note that some of the claims in this paper might be controversial. We welcome comments and criticism, and will be happy to run experiments on your favorite dataset. The online version on this paper (at www.cs.ucr.edu/~eamonn/selected_publications.php) will be updated within 48 hours of any contradictory evidence being found.

9. REFERENCES

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

[2] 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.

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

[4] Bozkaya, T, Yazdatani, Z, and Ozsoyoglu, Z.M. (1997). Matching and Indexing Sequences of Different Lengths. CIKM-97.

[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] Cardle, M. (2003). Music-Driven Animation. Ph.D. Thesis, Cambridge University.

[7] 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.

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

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

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

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

[12] Geurts, P. (2002). Contributions to decision tree induction: bias/variance tradeoff and time series classification. Ph.D. thesis, Department of Electrical Engineering and Computer Science, University of Liege, Belgium.

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

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

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

[16] Keogh, E., Chakrabarti, K., Pazzani, M., and Mehrotra, S. (2001). Locally adaptive dimensionality reduction for indexing large time series databases. In Proceeding so ACM SIGMOD International conference on Management of data, pp. 151-162.

[17] Keogh, E. and Kasetty, S. (2002). On the Need for Time Seires Data Mining Benchmarks: A Survey and Empirical Demonstration. In the 8th ACM SIGKDD, pp. 102-111.

[18] Kim, S.W., Park, S., & Chu, W.W. (2004). Efficient processing of similarity search under time warping in sequence databases: an index-based approach. Inf. Syst. 29(5): 405-420.

[19] Kornfield, E.M, Manmatha, R., and Allan, J. (2004). Text Alignment with Handwritten Documents. 1st International workshop on Document Image Analysis for Libraris (DIAL), pp. 195-209.

[20] Laaksonen, J., Hurri, J., and Oja, Erkki. (1998). Comparison of Adaptive Strategies for On-Line Character Recognition. In proceedings of ICANN'98, pp. 245-250.

[21] Park, S.,, Chu, W, Yoon, J., and Hsu, C (2000). Efficient searchs for similar subsequences of different lengths in sequence databases. In ICDE-00.

[22] 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.

[23] Ratanamahatana, C.A. & Keogh, E. (2004). Making Time-series Classification More Accurate Using Learned Constraints. In proceedings of SDM International conference, pp. 11-22.

[24] Rath, T. & Manmatha, R. (2003). Word image matching using dynamic time warping. CVPR, Vol. II, pp. 521-527.

[25] Roverso, D. (2000). Multivariate temporal classification by windowed wavelet decomposition and recurrent neural networks. In 3rd ANS International Topical Meeting on Nuclear Plant Instrumentation, Control and Human-Machine Interface, 2000.

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

[27] Shou, Y., Mamoulis, N., and Cheung, D.W. (2004). Efficient Warping of Segmented Time-series, HKU CSIS Tech report, TR-2004-01, March 2004.

[28] Wong, T.S.F & Wong, M.H. (2003). Efficient Subsequence Matching for Sequences Databases under Time Warping. IDEAS '03: 139-148.

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

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

10. APPENDIX A: A DESCRIPTION OF THE DATASETS USED

We have chosen seven classification datasets to be tested in our work. Note that we Z-normalized (mean = 0 and standard deviation = 1) all the datasets used in this work.

10.1 FACE DATASET

This is a face classification problem based on the head profiles. We took a number of photos (20-35) of four different individuals (1 female, 3 males) making different expressions on the face, e.g. talking, smiling, frowning, laughing, etc. We then convert each head profile, starting from the neck area into a "pseudo time series" by measuring the local angle of a trace of its perimeter, as shown in Figure 14. The dataset contains 112 instances in total with the length of each instance ranges from 107 to 240 data points.

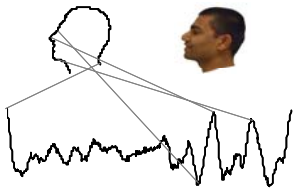

Figure 14. Starting from the neck area, the head profile is converted into a "pseudo time series"

10.2 GUN PROBLEM

This dataset comes from the video surveillance domain. The dataset has two classes, each containing 100 instances. 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.

Point: The actors have their gun 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 actor's right hands in both X- and Y-axes, which appear to be highly correlated; therefore, in this experiment, we only consider the X-axis for simplicity.

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 15.

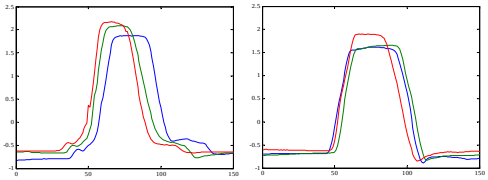

Figure 15. (left) some examples from the Gun-Draw data. (right) some examples from the Point data

The dataset contains 200 instances, 100 for each class. Since each actor was periodically signaled every 5 seconds before each repetition of Gun-Draw/Point, the video clip (captured at 30 frames per second) was easily segmented into 150 data points for each instance.

10.3 LEAF DATASET

This dataset contains a collection of six different species of leaf images, including 2 genera of plant, i.e. oak and maple (all original images can be found at [http://web.engr.oregonstate.edu/~tgd/leaves/dataset/herbari

um]). The dataset comprises four different species of Maple and two species of Oak, with 442 instances in total. We convert each leaf image into "pseudo time series" using similar method as the Face Dataset. Figure 11 shows an example of a Glabrum maple image converted into a 'time series'. The length of each time series ranges from 22 to 475 data points.

10.4 SYNTHETIC CONTROL CHART

This six-class dataset was retrieved from the UCR Time Series Data Mining Archive [http://www.cs.ucr.edu/~eamonn/TSDMA/]. It contains 600 instances in total with 100 instances in each class. Each instance has the same length of 60 data points.

10.5 TRACE DATASET

This dataset is a subset of the Transient Classification Benchmark first introduced by Davide Roverso [25]. This is a synthetic dataset designed to simulate instrumentation failures in a nuclear power plant. 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. The length of each instance ranges between 279 and 386 data points.

10.6 TWO-PATTERN DATASET

This four-class dataset contains 5,000 instances. Each instance has the same length of 128 data points. The dataset was introduced in [12]. Each class is characterized by the presence of two patterns in a definite order, down-down, up-down, down-up, and up-up.

10.7 WORDSPOTTING DATASET

This is a subset of the WordSpotting Project dataset created by Rath and Manmatha [24]

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. Once combined, we obtain a dataset of 272 instances, with the length of each instance ranges from 41 to 192 data points.

View publication stats