Querying and Mining of Time Series Data: Experimental Comparison of Representations and Distance Measures

Hui Ding§ Goce Trajcevski§[*] Peter Scheuermann§[†] Xiaoyue Wang¶ Eamonn Keogh¶[‡]

§hdi117, goce, peters@eecs.northwestern.edu
Northwestern University
Evanston, IL 60208

¶xwang, eamonn@cs.ucr.edu
University of California, Riverside
Riverside, CA 92517

ABSTRACT

The last decade has witnessed a tremendous growths of interests in applications that deal with querying and mining of time series data. Numerous representation methods for dimensionality reduction and similarity measures geared towards time series have been introduced. Each individual work introducing a particular method has made specific claims and, aside from the occasional theoretical justifications, provided quantitative experimental observations. However, for the most part, the comparative aspects of these experiments were too narrowly focused on demonstrating the benefits of the proposed methods over some of the previously introduced ones. In order to provide a comprehensive validation, we conducted an extensive set of time series experiments re-implementing 8 different representation methods and 9 similarity measures and their variants, and testing their effectiveness on 38 time series data sets from a wide variety of application domains. In this paper, we give an overview of these different techniques and present our comparative experimental findings regarding their effectiveness. Our experiments have provided both a unified validation of some of the existing achievements, and in some cases, suggested that certain claims in the literature may be unduly optimistic.

1. INTRODUCTION

Today time series data are being generated at an unprecedented speed from almost every application domain, e.g., daily fluctuations of stock market, traces of dynamic processes and scientific experiments, medical and biological experimental observations, various readings obtained from sensor networks, position updates of moving objects in location-based services etc. As a consequence, in the last decade there has been a dramatically increasing amount of interest in querying and mining such data which, in turn, resulted in a large amount of work introducing new methodologies for indexing, classification, clustering and approximation of time series [13, 17, 22].

Two key aspects for achieving *effectiveness* and *efficiency* when managing time series data are *representation methods* and *similarity measures*. Time series are essentially *high dimensional* data [17] and directly dealing with such data in its raw format is very expensive in terms of processing and storage cost. It is thus highly desirable to develop representation techniques that can reduce the dimensionality of time series, while still preserving the fundamental characteristics of a particular data set. In addition, unlike canonical data types, e.g., nominal or ordinal variables, where the distance definition is straightforward, the *distance* between time series needs to be carefully defined in order to reflect the underlying (dis)similarity of such data. This is particularly desirable for similarity-based retrieval, classification, clustering and other mining procedures of time series [17].

Many techniques have been proposed in the literature for representing time series with reduced dimensionality, such as *Discrete Fourier Transformation* (DFT) [13], *Single Value Decomposition* (SVD) [13], *Discrete Cosine Transformation* (DCT) [29], *Discrete Wavelet Transformation* (DWT) [33], *Piecewise Aggregate Approximation* (PAA) [24], *Adaptive Piecewise Constant Approximation* (APCA) [23], *Chebyshev polynomials* (CHEB) [6], *Symbolic Aggregate approXimation* (SAX) [30], *Indexable Piecewise Linear Approximation* (IPLA) [11] and etc. In conjunction with this, there are over a dozen distance measures for similarity of time series data in the literature, e.g., *Euclidean distance* (ED) [13], *Dynamic Time Warping* (DTW) [5, 26], distance based on *Longest Common Subsequence* (LCSS) [39],*Edit Distance with Real Penalty* (ERP) [8], *Edit Distance on Real sequence* (EDR) [9], DISSIM [14], *Sequence Weighted Alignment model* (Swale) [31], *Spatial Assembling Distance* (SpADe) [12] and similarity search based on *Threshold Queries* (TQuEST) [4]. Many of these work and some of their extensions have been widely cited in the literature and applied to facilitate query processing and data mining of time series data.

However, with the multitude of competitive techniques, we believe that there is a strong need to compare what might have been omitted in the comparisons. Every newly-introduced representation method or distance measure has claimed a particular superiority. However, it has been demon-

[*]Research supported by the Northrop Grumman Corp. contract: $P.O.8200082518$

[†]Research supported by the NSF grant $IIS - 0325144/003$

[‡]Research supported by NSF Career Award 0237918

strated that some empirical evaluations have been inadequate [25] and, worse yet, some of the claims are even contradictory. For example, one paper claims *"wavelets outperform the DFT"* [34], another claims *"DFT filtering performance is superior to DWT"* [19] and yet another claims *"DFT-based and DWT-based techniques yield comparable results"* [41]. Clearly these claims cannot all be true. The risk is that this may not only confuse newcomers and practitioners of the field, but also cause a waste of time and research efforts due to assumptions based on incomplete or incorrect claims.

Motivated by these observations, we have conducted the most extensive set of time series experiments to-date, re-evaluating the state-of-the-art representation methods and similarity measures for time series that appeared in high quality conferences and journals. Specifically, we have re-implemented 8 different representation methods for time series, and compared their *pruning power* over various time series data sets. We have re-implemented 9 different similarity measures and their variants, and compared their effectiveness using 38 real world data sets from highly diverse application domains. All our source code implementations and the data sets are publicly available on our website [1].

The rest of this paper is organized as follows. Section 2 reviews the concept of time series, and gives an overview of the definitions of different representation techniques and similarity measures investigated in this work. Section 3 and Section 4 present the main contribution of this work – the results of the extensive experimental evaluations of different representation methods and similarity measures, respectively. Section 5 concludes the paper and discusses possible future extensions of the work.

2. PRELIMINARIES

Typically, most of the existing work on time series assume that *time* is discrete. For simplicity and without any loss of generality, we make the same assumption here. Formally, a *time series* data is defined as a sequence of pairs $T = [(p_1, t_1), (p_2, t_2), ..., (p_i, t_i), ..., (p_n, t_n)]$ $(t_1 < t_2 < ... < t_i < ... < t_n)$, where each p_i is a data point in a d-dimensional data space, and each t_i is the time stamp at which p_i occurs. If the sampling rates of two time series are the same, one can omit the time stamps and consider them as sequences of d-dimensional data points. Such a sequence is called the *raw representation* of the time series. In reality however, sampling rates of time series may be different. Furthermore, some data points of time series may be dampened by noise or even completely missing, which poses additional challenges to the processing of such data. For a given time series, its number of data points n is called the *length*. The portion of a time series between two points p_i and p_j (inclusive) is called a *segment* and is denoted as s_{ij}. In particular, a segment between two consecutive points is called a *line segment*.

In the following subsections, we briefly review the representation methods and similarity measures studied in this work. We observe that this is not meant to be a complete survey for the respective field and is only intended to provide the readers with a necessary background for following our experimental evaluations.

2.1 Representation Methods for Time Series

There are many time series representations proposed to support similarity search and data mining. Figure 1 shows a classification of the major techniques arranged in a hierarchy.

- **Data Adaptive**
  - Piecewise Polynomials
    - Interpolation*
    - Regression
  - Adaptive Piecewise Constant Approximation*
  - Singular Value Decomposition*
  - Symbolic
    - Natural Language
    - Strings
      - Non-Lower Bounding
      - SAX*
      - Clipped Data*
  - Trees
- **Non-Data Adaptive**
  - Wavelets*
  - Random Mappings
  - Spectral
    - DFT*
    - DCT*
    - Chebyshev Polynomials*
  - Piecewise Aggregate Approximation*

Figure 1: *A Hierarchy of Representation Methods*

The representations annotated with an asterisk $(*)$ in Figure 1 have the very desirable property of allowing *lower bounding*. That is to say, we can define a distance measurement on the reduced-size (i.e., compressed) representations that is guaranteed to be less than or equal to the true distance measured on the raw data. It is this lower bounding property that allows using representations to index the data with a guarantee of no false negatives [13]. The list of representations considered in this study includes (in approximate order of introduction) DFT, DCT, DWT, PAA, APCA, SAX, CHEB and IPLA. The only lower bounding omissions from our experiments below are the eigenvalue analysis techniques such as SVD and PCA [29]. While such techniques give optimal linear dimensionality reduction, we believe they are untenable for large data sets. For example, while [38] notes that they can transform 70000 time series in under 10 minutes, the assumption is that the data is memory resident. However, transforming out-of-core (disk resident) data sets using these methods becomes unfeasible. Note that the available literature seems to agree with us on this point. For (at least) DFT, DWT and PAA, there are more than a dozen projects each that use the representations to index more than 100000 objects for query-by-humming [18, 45], Mo-Cap indexing [7] etc. However we are unaware of any similarly scaled projects that use SVD.

2.2 Similarity Measures for Time Series

In this section, we review the 9 similarity measures evaluated in this work, summarized in Figure 2. Given two time series T_1 and T_2, a similarity function $Dist$ calculates the distance between the two time series, denoted by $Dist(T_1, T_2)$. In the following we will refer to distance measures that compare the $i-$th point of one time series to the $i-$th point of another as lock-step measures (e.g., Euclidean distance and the other Lp norms), and distance measures that allow comparison of one-to-many points (e.g., DTW) and one-to-many/one-to-none points (e.g., LCSS) as elastic measures.

The most straightforward similarity measure for time series, is the *Euclidean Distance* [13] and its variants, based on the common L_p-norms [43]. In particular, in this work we used L_1 (Manhattan), L_2 (Euclidean) and L_∞ (Maxi-

- **Lock-step Measure**
  - L_p-norms
    - L_1-norm (Manhattan Distance)
    - L_2-norm (Euclidean Distance)
    - L_{inf}-norm
  - DISSIM
- **Elastic Measure**
  - Dynamic Time Warping (DTW)
  - Edit distance based measure
    - Longest Common SubSequence (LCSS)
    - Edit Sequence on Real Sequence (EDR)
    - Swale
    - Edit Distance with Real Penalty (ERP)
- **Threshold-based Measure**
  - Threshold query based similarity search (TQuEST)
- **Pattern-based Measure**
  - Spatial Assembling Distance (SpADe)

Figure 2: ***A Summary of Similarity Measures***

mum) norms (c.f. [43]). In the sequel, the terms Euclidean distance and L_2 norm will be used interchangeably. Besides being relatively straightforward and intuitive, Euclidean distance and its variants have several other advantages. The complexity of evaluating these measures is linear, and they are easy to implement and indexable with any access method and, in addition, are parameter-free. Furthermore, as we will present, the Euclidean distance is surprisingly competitive with other more complex approaches, especially if the size of the training set/database is relatively large. However, since the mapping between the points of two time series is *fixed*, these distance measures are very sensitive to noise and misalignments in time, and are unable to handle *local time shifting*, i.e., similar segments that are out of phase.

The DISSIM distance [14] aims at computing the similarity of time series with different sampling rates. However, the original similarity function is numerically too difficult to compute, and the authors proposed an approximated distance with a formula for computing the error bound.

Inspired by the need to handle time warping in similarity computation, Berndt and Clifford [5] introduced DTW, a classic speech recognition tool, to the data mining community, in order to allow a time series to be "stretched" or "compressed" to provide a better match with another time series. Several lower bounding measures have been introduced to speed up similarity search using DTW [21, 26, 27, 44], and it has been shown that the amortized cost for computing DTW on large data sets is linear [21, 26]. The original DTW distance is also parameter free, however, as has been reported in [26, 40] enforcing a *temporal constraint* δ on the warping window size of DTW not only improves its computation efficiency, but also improves its accuracy for measuring time series similarity, as extended warping may introduce pathological matchings between two time series and distort the true similarity. The constraint warping is also utilized for developing the lower-bounding distance [26] as well as for indexing time series based on DTW [40].

Another group of similarity measures for time series are developed based on the concept of the *edit distance* for strings. The best known such distance is the LCSS distance, utilizing the *longest common subsequence* model [3, 39]. To adapt the concept of matching characters in the settings of time series, a *threshold parameter* ε was introduced, stating that two points from two time series are considered to match if their distance is less than ε. The work reported in [39] also considered constraining the matching of points along the temporal dimension, using a *warping threshold* δ. A lower-bounding measure and indexing technique for LCSS are introduced in [40].

EDR [9] is another similarity measure based on the edit distance. Similar to LCSS, EDR also uses a threshold parameter ε, except its role is to quantify the distance between a pair of points to 0 or 1. Unlike LCSS, EDR assigns penalties to the gaps between two matched segments according to the lengths of the gaps.

The ERP distance [8] attempts to combine the merits of DTW and EDR, by using a *constant reference point* for computing the distance between gaps of two time series. Essentially, if the distance between two points is too large, ERP simply uses the distance value between one of those point and the reference point.

Recently, a new approach for computing the edit distance based similarity measures was proposed in [31]. Whereas traditional tabular dynamic programming was used for computing DTW, LCSS, EDR and ERP, a matching threshold is used to divide the data space into grid cells and, subsequently, matching points are found by hashing. The similarity model Swale is proposed that rewards matching points and penalizes gaps. In addition to the matching threshold ε, Swale requires the tuning of two parameters: the *matching reward* weight r and the *gap penalty* weight p.

The TQuEST distance [4] introduced a rather novel approach to computing the similarity measure between time series. The idea is that, given a threshold parameter τ, a time series is transformed into a sequence of so-called *threshold-crossing* time intervals, where the points within each time interval have a value greater than a given τ. Each time interval is then treated as a point in a two dimensional space, where the starting time and ending time constitute the two dimensions. The similarity between two time series is then defined as the Minkowski sum of the two sequences of time interval points.

Finally, SpADe [12] is a pattern-based similarity measure for time series. The algorithm finds out matching segments within the entire time series, called *patterns*, by allowing shifting and scaling in both the temporal and amplitude dimensions. The problem of computing similarity value between time series is then transformed to the one of finding the most similar set of matching patterns. One disadvantage of SpADe is that it requires tuning a number of parameters, such as the temporal scale factor, amplitude scale factor, pattern length, sliding step size etc.

3. COMPARISON OF TIME SERIES REPRESENTATIONS

We compare all the major time series representations, including SAX, DFT, DWT, DCT, PAA, CHEB, APCA and IPLA. Note that any of these representations can be used to index the Euclidean Distance, the Dynamic Time Warping, and at least some of the other elastic measures. While various subsets of these representations have been compared before, this is the first time they are all been compared together. An obvious question to consider is what metric should should be used for comparison. We believe that wall clock time is a poor choice, because it may be open to implementation bias [25]. Instead, we believe that using the *tightness of lower bounds* (TLB) is a very meaningful measure [24], and this also appears to be the current consensus in the literature [6, 8, 11, 21–23, 26, 35, 40]. We note that all the representation methods studied in this paper allow lower bounding.

$$TLB = LowerBoundDist(T, S)/TrueEuclideanDist(T, S)$$

where T and S are the two time series. The advantage of TLB is two-fold:

1. It is a completely implementation-free measure, independent of hardware and software choices, and is therefore completely reproducible.
2. The TLB allows a very accurate prediction of indexing performance.

If the value of TLB is zero, then any indexing technique is condemned to retrieving every time series from the disk. If the value of TLB is one, then after some trivial processing in main memory, we could simply retrieve one object from disk and guarantee that we had the true nearest neighbor. Note that the speedup obtained is generally non-linear in TLB, that is to say, if one representation has a lower bound that is twice as large as another, we can usually expect a much greater than two-fold decrease in disk accesses. We randomly sampled T and S (with replacement) 1000 times for each combination of parameters. We vary the time series length $\{480, 960, 1440, 1920\}$ and the number of coefficients per time series available to the dimensionality reduction approach $\{4, 6, 8, 10\}$ (each coefficient takes 4 bytes). For SAX, we hard coded the cardinality to 256. Figure 3 shows the result of one such experiment with an ECG data set.

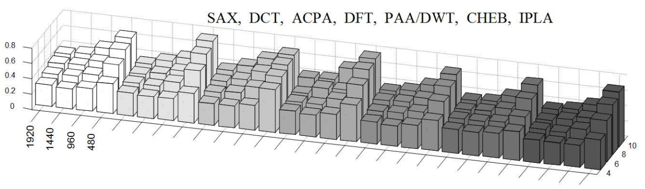

Figure 3: ***The tightness of lower bounds for various time series representations on an ECG data set***

The result of this experiment may at first be surprising, it shows that there is very little difference between representations, in spite of claims to the contrary in the literature. However, we believe that most of these claims are due to some errors or bias in the experiments. For example, it was recently claimed that DFT is much worse than all the other approaches [11], however it appears that the complex conjugate property of DFT was not exploited. As another example, it was claimed *"it only takes* 4 *to* 6 *Chebyshev coefficients to deliver the same pruning power produced by* 20 *APCA coefficients"* [6], however this claim has since been withdrawn by the authors [2]. Of course there is some variability and difference depending on data set. For example, on highly periodic data set the spectral methods are better, and on bursty data sets APCA can be significantly better, as shown in Figure 4.

In contrast, in Figure 5 we can see that highly periodic data can slightly favor the spectral representations (DCT, DFT, CHEB) over the polynomial representations (SAX, APCA, DWT/PAA, IPLA).

However it is worth noting that the differences presented in these figures are the most extreme cases found in a search over 80 diverse data sets from the publicly available UCR Time Series Data Mining Archive [20]. In general, there

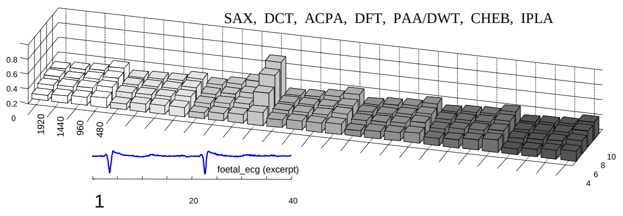

Figure 4: ***The tightness of lower bounds for various time series representations on a relatively bursty data set (see inset)***

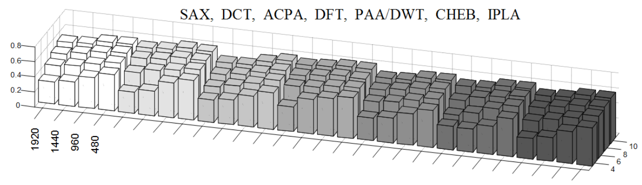

Figure 5: ***The tightness of lower bounds for various time series representations on a periodic data set of tide levels***

is very little to choose between representations in terms of pruning power.

4. COMPARISON OF TIME SERIES SIMILARITY MEASURES

In this section we present our experimental evaluation on the accuracy of different similarity measures.

4.1 The Effect of Data Set Size on Accuracy and Speed

We first discuss an extremely important finding which, in some circumstances makes some of the previous findings on efficiency, and the subsequent findings on accuracy, moot. This finding has been noted before [35], but does not seem to be appreciated by the database community.

For an elastic distance measure, both the *accuracy* of classification (or precision/recall of similarity search), and the *amortized speed*, depend critically on the size of the data set. In particular, as data sets get larger, the amortized speed of elastic measures approaches that of lock-step measures, however the accuracy/precision of lock-step measures approaches that of the elastic measures. This observation has significant implications for much of the research in the literature. Many papers claim something like *"I have shown on these* 80 *time series that my elastic approach is faster than DTW and more accurate that Euclidean distance, so if you want to index a million time series, use my method"*. However our observation suggests that even if the method is faster than DTW, the speed difference will decrease for larger data sets. Furthermore, for large data sets, the differences in accuracy/precision will also diminish or disappear. To demonstrate our claim we conducted experiments on two highly warped data sets that are often used to highlight the superiority of elastic measures, Two-Patterns and CBF. Because these are synthetic data sets we have the luxury of creating as many instances as we like, using the data generation algorithms proposed in the original papers [15, 16].

However, it is critical to note that the same effect can be seen on all the data sets considered in this work. For each problem we created 10000 test time series, and increasingly large training data sets of size 50, 100, 200, ..., 6400. We measured the classification accuracy of 1NN for the various data sets (explained in more detail in Section 4.2.1), using both Euclidean distance and DTW with 10% warping window, Figure 6 shows the results.

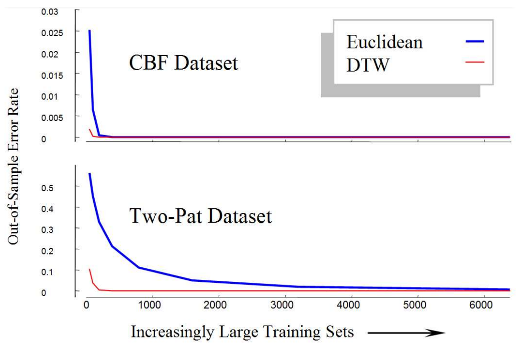

Figure 6: ***The error rate for 1-Nearest Neighbor Classification for increasingly large instantiations of two classic time series benchmarks***

Note that for small data sets, DTW is significantly more accurate than Euclidean distance in both cases. However, for CBF, by the time we have a mere 400 time series in our training set, there is no statistically significant difference. For Two-Patterns it takes longer for Euclidean Distance to converge to DTW's accuracy, nevertheless, by the time we have seen a few thousand objects there is no statistically significant difference.

This experiment can also be used to demonstrate our claim that amortized speed of a (lower-boundable) elastic method approaches that of Euclidean distance. Recall that Euclidean distance has a time complexity of $O(n)$ and that a single DTW calculation has a time complexity of $O(nw)$, where w is the warping window size. However for similarity search or 1NN classification, the amortized complexity of DTW is $O((P \cdot n) + (1 - P) \cdot nw)$, where P is the fraction of DTW calculations pruned by a linear time lower bound such as LB_Keogh. A similar result can be achieved for LCSS as well. In the Two-Pattern experiments above, when classifying with only 50 objects, $P = 0.1$, so we are forced to do many full DTW calculations. However, by the time we have 6400 objects, we empirically find out that $P = 0.9696$, so about 97% of the objects are disposed of in the same time as takes to do a Euclidean distance calculation. To ground this into concrete numbers, it takes less that one second to find the nearest neighbor to a query in the database of 6400 Two-Pat time series, on our off-the-shelf desktop, even if we use the pessimistically wide warping window. This time is for just sequential search with a lower bound, no attempt was made to index the data.

To summarize, many of the claims over who has the fastest or most accurate distance measure have been biased by the lack of tests on very (or even slightly) large data sets.

4.2 Accuracy of Similarity Measures

In this section, we evaluate the accuracy of the similarity measures introduced in Section 2. We first explain the methodology of our evaluation, as well as the parameters that need to be tuned for each similarity measure. We then present the results of our experiments and discuss several interesting findings.

4.2.1 *Accuracy Evaluation Framework*

Accuracy evaluation answers one of the most important questions about a similarity measure: why is this a good measure for describing the (dis)similarity between time series? Surprisingly, we found that accuracy evaluation is usually insufficient in existing literature: it has been either based on subjective evaluation, e.g., [4, 9], or using clustering with small data sets which are not statistically significant, e.g., [31, 40]. In this work, we use an objective evaluation method recently proposed [25]. The idea is to use a one nearest neighbor (1NN) classifier [17, 32] on labelled data to evaluate the efficacy of the distance measure used. Specifically, each time series has a correct class label, and the classifier tries to predict the label as that of its nearest neighbor in the training set. There are several advantages with this approach. First, it is well known that the underlying distance metric is critical to the performance of 1NN classifier [32], hence, the accuracy of the 1NN classifier directly reflects the effectiveness of the similarity measure. Second, the 1NN classifier is straightforward to implement and is parameter free, which makes it easy for anyone to reproduce our results. Third, it has been proved that the error ratio of 1NN classifier is at most twice the Bayes error ratio [36]. Finally, we note that while there have been attempts to classify time series with decision trees, neural networks, Bayesian networks, supporting vector machines etc., the best published results (by a large margin) come from simple nearest neighbor methods [42].

Algorithm 1 Time Series Classification with 1NN Classifier

```
Input: Labelled time series data set $\mathbb{T}$, similarity measure
    operator $SimDist$, number of crosses $k$
Output: Average 1NN classification error ratio and standard deviation
 1: Randomly divide $\mathbb{T}$ into $k$ stratified subsets $\mathbb{T}_1, \ldots, \mathbb{T}_k$
 2: Initialize an array $ratios[k]$
 3: for Each subset $\mathbb{T}_i$ of $\mathbb{T}$ do
 4:   if $SimDist$ requires parameter tuning then
 5:     Randomly split $\mathbb{T}_i$ into two equal size stratified subsets $\mathbb{T}_{i1}$ and $\mathbb{T}_{i2}$
 6:     Use $\mathbb{T}_{i1}$ for parameter tuning, by performing a
        leave-one-out classification with 1NN classifier
 7:     Set the parameters to values that yields the minimum error ratio from the leave-one-out tuning process
 8:   Use $\mathbb{T}_i$ as the training set, $\mathbb{T} - \mathbb{T}_i$ as the testing set
 9:   $ratio[i] \leftarrow$ the classification error ratio with 1NN classifier
10: return Average and standard deviation of $ratios[k]$
```

To evaluate the effectiveness of each similarity measure, we use a cross-validation algorithm as described in Algorithm 1, based on the approach suggested in [37]. We first use a stratified random split to divide the input data set into k subsets for the subsequent classification (line 1), to minimize the impact of skewed class distribution. The number of cross validations k is dependent on the data sets and we explain shortly how we choose the proper value for k. We then carry out the cross validation, using one subset at a time for the training set of the 1NN classifier, and the rest $k-1$ subsets as the testing set (lines $3-9$). If the similarity measure *SimDist* requires parameter tuning, we divide the training set into two equal size stratified subsets, and use one of the subset for parameter tuning (lines $4-7$). We perform an exhaustive search for all the possible (combinations of) value(s) of the similarity parameter, and conduct a leave-one-out classification test with a 1NN classifier. We record the error ratios of the leave-one-out test, and use the parameter values that yield the minimum error ratio. Finally, we report the average error ratio of the 1NN classification over the k cross validations, as well as the standard deviation (line 10).

Algorithm 1 requires that we provide an input k for the number of cross validations. In our experiments, we need to take into consideration the impact of training data set size discussed in Section 4.1. Therefore, our selection of k for each data set tries to strike a balance between the following factors:

1. The training set size should be selected to enable discriminativity, i.e., one can tell the performance difference between different distance measures.
2. The number of items in the training set should be large enough to represent each class. This is especially important when the distance measure need parameter tuning.
3. The number of cross validations should be between $5-20$ in order to minimize bias and variation, as recommended in [28].

The actual number of splits is empirically selected such that the training error for 1NN Euclidean distance (which we use as a comparison reference) is not perfect, but significantly better than the default rate.

Several of the similarity measures that we investigated require the setting of one or more parameters. The proper values for these parameters are key to the effectiveness of the measure. However, most of the time only empirical values are provided for each parameter in isolation. In our experiments, we perform an exhaustive search for all the possible values of the parameters, as described in Table 1.

Parameter	Min Value	Max Value	Step Size
DTW.δ	1	$25\% \cdot n$	1
LCSS.δ	1	$25\% \cdot n$	1
LCSS.ε	$0.02 \cdot Stdv$	$Stdv$	$0.02 \cdot Stdv$
EDR.ε	$0.02 \cdot Stdv$	$Stdv$	$0.02 \cdot Stdv$
Swale.ε	$0.02 \cdot Stdv$	$Stdv$	$0.02 \cdot Stdv$
Swale.*reward*	50	50	-
Swale.*penalty*	0	*reward*	1
TQuEST.τ	$Avg - Stdv$	$Avg + Stdv$	$0.02 \cdot Stdv$
SpADe.*plength*	8	64	8
SpADe.*ascale*	0	4	1
SpADe.*tscale*	0	4	1
SpADe.*slidestep*	$plength/32$	$plength/8$	$plength/32$

Table 1: *Parameter Tuning for Similarity Measures*

For DTW and LCSS measures, a common optional parameter is the window size δ that constrains the temporal warping, as suggested in [40]. In our experiments we consider both the version of distance measures without warping and with warping. For the latter case, we search for the best warping window size up to 25% of the length of the time series n. An additional parameter for LCSS, which is also used in EDR and Swale, is the matching threshold ε. We search for the optimal threshold starting from $0.02 \cdot Stdv$ up to $Stdv$, where $Stdv$ is the standard deviation of the data set. Swale has another two parameters, the matching reward weight and the gap penalty weight. We fix the matching reward weight to 50 and search for the optimal penalty weight from 0 to 50, as suggested by the authors. Although the warping window size can also be constrained for EDR, ERP and Swale, due to limited computing resource and time, we only consider full matching for these distance measures in our current experiments. For TQuEST, we search for the optimal querying threshold from $Avg - Stdv$ to $Avg + Stdv$, where Avg is the average of the time series data set. For SpADe, we tune four parameters based on the original implementation and use the parameter tuning strategy, i.e. search range, step size, as suggested by the authors. In Table 1, *plength* is the length of the patterns, *ascale* and *tscale* are the maximum amplitude and temporal scale differences allowed respectively, and *slidestep* is the minimum temporal difference between two patterns.

4.3 Analysis of Classification Accuracy

In order to provide a comprehensive evaluation, we perform the experiments on 38 diverse time series data sets, provided by the UCR Time Series repository [20], which make up somewhere between 90% and 100% of all publicly available, labelled time series data sets in the world. For several years everyone in the data mining/database community has been invited to contribute data sets to this archive, and 100% of the donated data sets have been archived. This ensures that the collection represents the interest of the data mining/database community, and not just one group. All the data sets have been normalized to have a maximum scale of 1.0 and all the time series are z-normalized. The entire simulation was conducted on a computing cluster in the Northwestern University, with 20 multi-core workstations running for over a month. The results are presented in Table 2. Due to limited space, we only show the average error ratio of the similarity measures on each data set. More detailed results, such as the standard deviation of the cross validations, are hosted on our web site [1].

To provide a more intuitive illustration of the performance of the similarity measures compared in Table 2, we now use scatter plots to conduct pair-wise comparisons. In a scatter plot, the error ratios of the two similarity measures under comparison are used as the x and y coordinates of a dot, where each dot represent a particular data set. Where a scatter plot has the label "A vs B", a dot above line indicates that A is more accurate than B (since these are error ratios). The further a dot is from the line, the greater the margin of accuracy improvement. The more dots on one side of the line indicates that the worse one similarity measure is compared to the other .

First, we compare the different variances of L_p-norms. Figure 7 shows that the Euclidean distance and the Manhattan distance have a very close performance, while both largely outperforms the L_∞-norm. This is as expected since the L_∞-norm uses the maximum distance between two sets of time series points, and is more sensitive to noise [17].

Next we illustrate the performance of DTW against Euclidean. Figure 8 (a) shows that full DTW is clearly superior over Euclidean on the data sets we tested. Figure 8 (b)

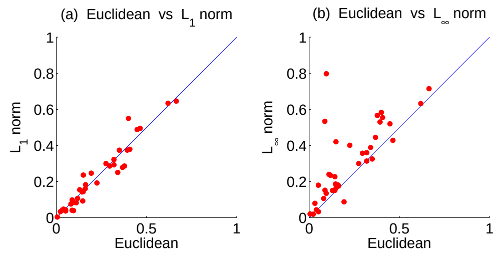

Figure 7: *Accuracy of Various L_p-norms, above the line Euclidean outperforms L_1norm/L_∞norm*

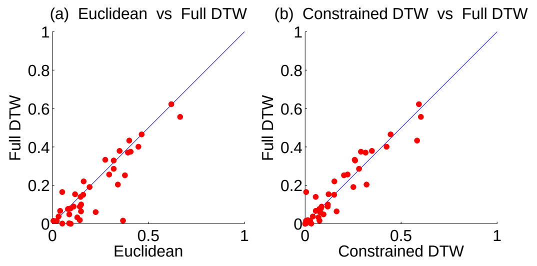

Figure 8: *Accuracy of DTW, above the line Euclidean/Constrained DTW outperforms Full DTW*

shows that the effectiveness of constrained DTW is the same (or even slightly better) than that of full DTW. This means that we could generally use the constrained DTW instead of DTW, to reduce the time for computing the distance and to utilize proposed lower bounding techniques [26].

Unless otherwise stated, in the following we compare the rest of the similarity measures against Euclidean distance and full DTW, since Euclidean distance is the fastest and most straightforward measure, and DTW is the oldest elastic measure.

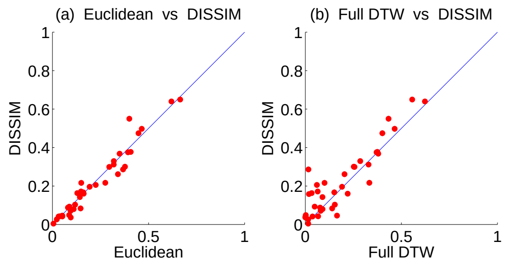

Figure 9: *Accuracy of DISSIM, above the line Euclidean/Full DTW outperforms DISSIM*

The performance of DISSIM against that of Euclidean and DTW are shown in Figure 9. It can be observed that the accuracy of DISSIM is slightly better than Euclidean distance, however, it is apparently inferior to DTW.

The performance of TQuEST against that of Euclidean and DTW are shown in Figure 10. On most of the data sets, TQuEST is worse than Euclidean and DTW distances. While the outcome of this experiment cannot account for the usefulness of TQuEST, it indicates that there is a need to investigate the characteristics of the data set for which TQuEST is a favorable measure.

The performance of LCSS, EDR and ERP against that of Euclidean and DTW are shown in Figure 11, Figure 12 and Figure 13 respectively. All three distances outperform Euclidean distance by a large percentage. On the other hand, while it is commonly believed that these edit distance based similarity measures are superior to DTW [8, 10, 12], our experiments show that this is generally not the case. Only EDR is *potentially* slightly better than full DTW, whereas the performance of LCSS and ERP are very close to DTW. Even for EDR, a more formal analysis using two-tailed,

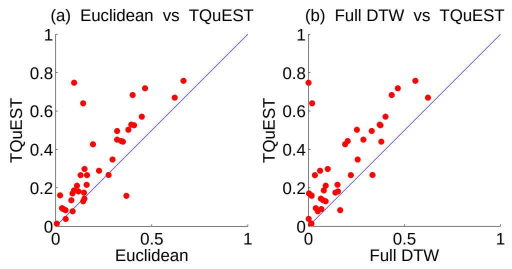

Figure 10: *Accuracy of TQuEST, above the line Euclidean/Full DTW outperforms TQuEST*

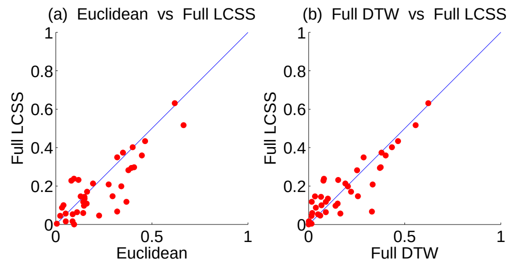

Figure 11: *Accuracy of LCSS, above the line Euclidean/Full DTW outperforms Full LCSS*

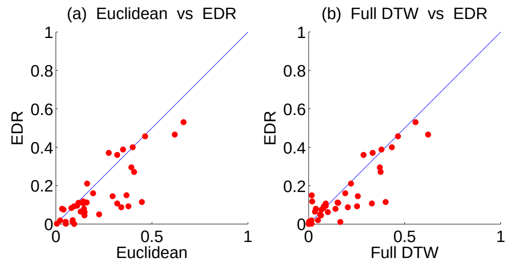

Figure 12: *Accuracy of EDR, above the line Euclidean/Full DTW outperforms EDR*

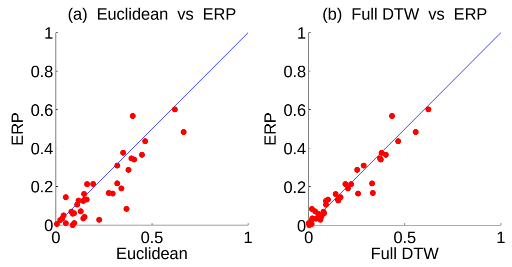

Figure 13: *Accuracy of ERP, above the line Euclidean/Full DTW outperforms ERP*

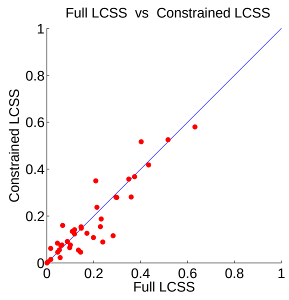

Figure 14: *Accuracy of Constrained LCSS, above the line Full LCSS outperforms Constrained LCSS*

paired t-test is required to reach any statistically significant conclusion [37]. We also studied the performance of constrained LCSS, as shown in Figure 14. It can be observed that the constrained version of LCSS is even slightly better the unconstrained one, while it also reduces the computation cost and gives rise to an efficient lower-bounding measure [40].

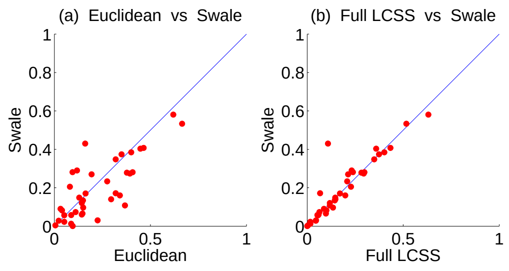

Figure 15: *Accuracy of Swale, above the line Euclidean/Full LCSS outperforms Swale*

Next, we compare the performance of Swale against that of Euclidean distance and LCSS, as Swale aims at improving the effectiveness of LCSS and EDR. The results are shown in Figure 15, and suggests that Swale is clearly superior to Euclidean distance, and yields an almost identical accuracy as LCSS.

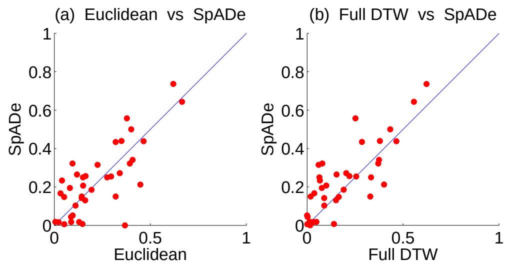

Figure 16: *Accuracy of SpADe, above the line Euclidean/Full DTW outperforms SpADe*

Finally, we compare the performance of SpADe against that of Euclidean distance and DTW. The results are shown in Figure 16. In general, the accuracy of SpADe is close to that of Euclidean but is inferior to DTW distance, although on some data sets SpADe outperforms the other two. We believe one of the biggest challenges for SpADe is that it has a large number of parameters that need to be tuned. Given the small tuning data sets, it is very difficult to pick the right values. However, we note again that the outcome of this experiment cannot account for the utility of SpADe. For example, one major contribution of SpADe is to detect interesting patterns online for stream data.

In summary, we found through experiments that there is no clear evidence that there exists one similarity measure that is superior to others in the literature, in terms of accuracy. While some similarity measures are more effective on certain data sets, they are usually inferior on some others data sets. This does not mean that the time series community should settle with the existing similarity measures, – quite the contrary. However, we believe that more caution need to be exercised to avoid making the kind of mistakes we illustrate in the Appendix.

5. CONCLUSION & FUTURE WORK

In this paper, we conducted an extensive experimental consolidation on the state-of-the-art representation methods and similarity measures for time series data. We re-implemented and evaluated 8 different dimension-reduction representation methods, as well as 9 different similarity measures and their variants. Our experiments are carried on 38 diverse time series data sets from various application domains. Based on the experimental results we obtained, we make the following conclusions.

1. The tightness of lower bounding, thus the pruning power, thus the indexing effectiveness of the different representation methods for time series data have, for the most part, very little difference on various data sets.
2. For time series classification, as the size of the training set increases, the accuracy of elastic measures converge with that of Euclidean distance. However, on small data sets, elastic measures, e.g., DTW, LCSS, EDR and ERP etc. can be significantly more accurate than Euclidean distance and other lock-step measures, e.g., L_∞-norm, DISSIM.
3. Constraining the warping window size for elastic measures, such as DTW and LCSS, can reduce the computation cost and enable effective lower-bounding, while yielding the same or even better accuracy.
4. The accuracy of edit distance based similarity measures, such as LCSS, EDR and ERP are very close to that of DTW, a 40 year old technique. In our experiments, only EDR is potentially slightly better than DTW.
5. The accuracy of several novel types of similarity measures, such as TQuEST and SpADe, are on general inferior to elastic measures.
6. If a similarity measure is not accurate enough for the task, getting more training data *really* helps.
7. If getting more data is not possible, then trying the other measures *might* help, however, extreme care must be taken to avoid overfitting.

As an additional comment, but not something that can be conclusively validated from our experiments, we would like to bring an observation which, we hope, may steer some interesting directions of future work. Namely, when pair-wise comparison is done among the methods, in few instances we have one method that has worse accuracy than the other in majority of the data sets, but in the ones that it is better, it does so by a large margin. Could it be due to some intrinsic properties of the data set? If so, could it be that those properties may have a critical impact on which distance measure should be applied? We believe that in the near future, the research community will generate some important results along these lines.

As an immediate extension, we plan to conduct more rigorous statistical analysis on the experimental results we obtained. We will also extend our evaluation on the accuracy of the similarity measures to more realistic settings, by allowing missing points in the time series and adding noise to the data. Another extension is to validate the effectiveness of some of the existing techniques in expediting similarity search using the respective distance measures.

6. ACKNOWLEDGEMENT & DISCLAIMER

We would like to sincerely thank all the donors of the data sets, without which this work would not have been possible. We would also like to thank Lei Chen, Jignesh Patel, Beng Chin Ooi, Yannis Theodoridis and their respective team members for generously providing the source code to their similarity measures as references and for providing timely help on this project. We would like to thank Michalis Vlachos, Themis Palpanas, Chotirat (Ann) Ratanamahatana and Dragomir Yankov for useful comments and discussions. Needless to say, any errors herein are the responsibility of the authors only. We would also like to thank Gokhan Memik and Robert Dick at Northwestern University for providing the computing resources for our experiments.

We have made the best effort to faithfully re-implement all the algorithms, and evaluate them in a fair manner. The purpose of this project is to provide a consolidation of existing works on querying and mining time series, and to serve as a starting point for providing references and benchmarks for future research on time series. We welcome all kinds of comments on our source code and the data sets of this project [1], and suggestions on other experimental evaluations.

7. REFERENCES

[1] Additional Experiment Results for Representation and Similarity Measures of Time Series. http://www.ece.northwestern.edu/∼hdi117/tsim.htm.

[2] R.T.Ng (2006), Note of Caution. http://www.cs.ubc.ca/∼rng/psdepository/cheby Report2.pdf.

[3] H. André-Jönsson and D. Z. Badal. Using signature files for querying time-series data. In *PKDD*, 1997.

[4] J. Aßfalg, H.-P. Kriegel, P. Kröger, P. Kunath, A. Pryakhin, and M. Renz. Similarity search on time series based on threshold queries. In *EDBT*, 2006.

[5] D. J. Berndt and J. Clifford. Using dynamic time warping to find patterns in time series. In *KDD Workshop*, 1994.

[6] Y. Cai and R. T. Ng. Indexing spatio-temporal trajectories with chebyshev polynomials. In *SIGMOD Conference*, 2004.

[7] M. Cardle. Automated motion editing. In *Technical Report, Computer Laboratory, University of Cambridge, UK*, 2004.

[8] L. Chen and R. T. Ng. On the marriage of lp-norms and edit distance. In *VLDB*, 2004.

[9] L. Chen, M. T. Özsu, and V. Oria. Robust and fast similarity search for moving object trajectories. In *SIGMOD Conference*, 2005.

[10] L. Chen, M. T. Özsu, and V. Oria. Using multi-scale histograms to answer pattern existence and shape match queries. In *SSDBM*, 2005.

[11] Q. Chen, L. Chen, X. Lian, Y. Liu, and J. X. Yu. Indexable PLA for Efficient Similarity Search. In *VLDB*, 2007.

[12] Y. Chen, M. A. Nascimento, B. C. Ooi, and A. K. H. Tung. SpADe: On Shape-based Pattern Detection in Streaming Time Series. In *ICDE*, 2007.

[13] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos. Fast Subsequence Matching in Time-Series Databases. In *SIGMOD Conference*, 1994.

[14] E. Frentzos, K. Gratsias, and Y. Theodoridis. Index-based most similar trajectory search. In *ICDE*, 2007.

[15] P. Geurts. Pattern Extraction for Time Series Classification. In *PKDD*, 2001.

[16] P. Geurts. *Contributions to Decision Tree Induction: bias/variance tradeoff and time series classification*. PhD thesis, University of Lige, Belgium, May 2002.

[17] Jiawei Han and Micheline Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, CA, 2005.

[18] I. Karydis, A. Nanopoulos, A. N. Papadopoulos, and Y. Manolopoulos. Evaluation of similarity searching methods for music data in P2P networks. *IJBIDM*, 1(2), 2005.

[19] K. Kawagoe and T. Ueda. A Similarity Search Method of Time Series Data with Combination of Fourier and Wavelet Transforms. In *TIME*, 2002.

[20] E. Keogh, X. Xi, L. Wei, and C. Ratanamahatana. The UCR Time Series dataset. In *http://www.cs.ucr.edu/∼eamonn/time_series_data/*, 2006.

[21] E. J. Keogh. Exact Indexing of Dynamic Time Warping. In *VLDB*, 2002.

[22] E. J. Keogh. A Decade of Progress in Indexing and Mining Large Time Series Databases. In *VLDB*, 2006.

[23] E. J. Keogh, K. Chakrabarti, S. Mehrotra, and M. J. Pazzani. Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases. In *SIGMOD Conference*, 2001.

[24] E. J. Keogh, K. Chakrabarti, M. J. Pazzani, and S. Mehrotra. Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases. *Knowl. Inf. Syst.*, 3(3), 2001.

[25] E. J. Keogh and S. Kasetty. On the Need for Time Series Data Mining Benchmarks: A Survey and Empirical Demonstration. *Data Min. Knowl. Discov.*, 7(4), 2003.

[26] E. J. Keogh and C. A. Ratanamahatana. Exact indexing of dynamic time warping. *Knowl. Inf. Syst.*, 7(3), 2005.

[27] S.-W. Kim, S. Park, and W. W. Chu. An Index-Based Approach for Similarity Search Supporting Time Warping in Large Sequence Databases. In *ICDE*, 2001.

[28] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *IJCAI*, 1995.

[29] F. Korn, H. V. Jagadish, and C. Faloutsos. Efficiently

supporting ad hoc queries in large datasets of time sequences. In *SIGMOD Conference*, 1997.

[30] J. Lin, E. J. Keogh, L. Wei, and S. Lonardi. Experiencing SAX: a novel symbolic representation of time series. *Data Min. Knowl. Discov.*, 15(2), 2007.

[31] M. D. Morse and J. M. Patel. An efficient and accurate method for evaluating time series similarity. In *SIGMOD Conference*, 2007.

[32] Pang-Ning Tan and Michael Steinbach and Vipin Kumar. *Introduction to Data Mining.* Addison-Wesley, Reading, MA, 2005.

[33] K. pong Chan and A. W.-C. Fu. Efficient Time Series Matching by Wavelets. In *ICDE*, 1999.

[34] I. Popivanov and R. J. Miller. Similarity Search Over Time-Series Data Using Wavelets. In *ICDE*, 2002.

[35] C. A. Ratanamahatana and E. J. Keogh. Three myths about dynamic time warping data mining. In *SDM*, 2005.

[36] Richard O. Duda and Peter E. Hart. *Pattern Classification and Scene Analysis.* John Wiley & Sons, 1973.

[37] S. Salzberg. On Comparing Classifiers: Pitfalls to Avoid and a Recommended Approach. *Data Min. Knowl. Discov.*, 1(3), 1997.

[38] M. Steinbach, P.-N. Tan, V. Kumar, S. A. Klooster, and C. Potter. Discovery of climate indices using clustering. In *KDD*, 2003.

[39] M. Vlachos, D. Gunopulos, and G. Kollios. Discovering similar multidimensional trajectories. In *ICDE*, 2002.

[40] M. Vlachos, M. Hadjieleftheriou, D. Gunopulos, and E. J. Keogh. Indexing Multidimensional Time-Series. *VLDB J.*, 15(1), 2006.

[41] Y.-L. Wu, D. Agrawal, and A. E. Abbadi. A Comparison of DFT and DWT based Similarity Search in Time-Series Databases. In *CIKM*, 2000.

[42] X. Xi, E. J. Keogh, C. R. Shelton, L. Wei, and C. A. Ratanamahatana. Fast time series classification using numerosity reduction. In *ICML*, 2006.

[43] B.-K. Yi and C. Faloutsos. Fast Time Sequence Indexing for Arbitrary Lp Norms. In *VLDB*, 2000.

[44] B.-K. Yi, H. V. Jagadish, and C. Faloutsos. Efficient retrieval of similar time sequences under time warping. In *ICDE*. IEEE Computer Society, 1998.

[45] Y. Zhu and D. Shasha. Warping Indexes with Envelope Transforms for Query by Humming. In *SIGMOD Conference*, 2003.

APPENDIX

A SOBERING EXPERIMENT

Our comparative experiments have shown that while elastic measures are, in general, better than lock-step measures, there is little difference between the various elastic measures. This result explicitly contradicts many papers that claim to have a distance measure that is better than DTW, the original and simplest elastic measure. How are we to reconcile this two conflicting claims? We believe the following demonstration will shed some light on the issue. We classified 20 of the data sets hosted at the UCR archive, using the suggested two-fold splits that were establish several years ago. We used a distance measure called ANA (explained below) which has a single parameter that we adjusted to get the best performance. Figure 17 compares the results of our algorithm with Euclidean distance.

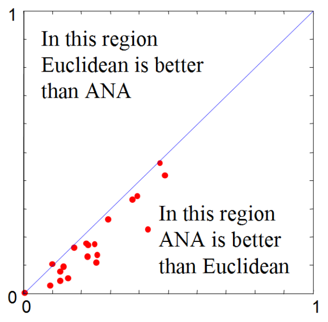

Figure 17: *The classification accuracy of ANA compared to Euclidean distance*

As we can see, the ANA algorithm is consistently better than Euclidean distance, often significantly so. Furthermore, ANA is as fast as Euclidean distance, is indexable and only has a single parameter. Given this, can a paper on ANA be published in a good conference or journal? It is time to explain how ANA works. We downloaded the mitochondrial DNA of a monkey, Macaca mulatta. We converted the DNA to a string of integers, with A (Adenine) = 0, C (Cytosine) = 1, G (Guanine) = 2 and T (Thymine) = 3. So the DNA string $GATCA\ldots$ becomes $2, 0, 3, 1, 0, \ldots$. Given that we have a string of 16564 integers, we can use the first n integers as weighs when calculating the weights of the Euclidean distance between our time series of length n. So ANA is nothing more than the weighed Euclidean distance, weighed by monkey DNA. More concretely, if we have a string S: $S = 3, 0, 1, 2, 0, 2, 3, 0, 1, \ldots$ and some time series, say of length 4, then the weight vector W with $p = 1$ is $\{3, 0, 1, 2\}$, and the ANA distance is simply:

$$ANA(A, B, W) = \sqrt{\sum_{i=1}^{4} (A_i - B_i)^2 \times W_i}.$$

After we test the algorithm, if we are not satisfied with the result, we simply shift the first location in the string, so that we are using locations 2 to $n + 1$ of the weight string. We continue shifting till the string is exhausted and reported the best result in Figure 17. At this point the reader will hopefully say "but that's not fair, you cannot change the parameter after seeing the results, and report the best results". However, we believe this effect may explain many of the apparent improvements over DTW, and in some cases, the authors explicitly acknowledge this [10]. Researchers *are* adjusting the parameters after seeing the results on the test set. In summary, based on the experiments conduced in this paper and all the reproducible fair experiments in the literature, there is no evidence of any distance measure that is systematically better than DTW in general. Furthermore, there is at best very scant evidence that there is any distance measure that is systematically better than DTW in on particular problems (say, just ECG data, or just noisy data).

Finally, we are in a position to explain what ANA stand for. It is an acronym for *Arbitrarily Naive Algorithm.*

Data Set	# of crosses	ED	L_1-norm	L_∞-norm	DISSIM	TQuEST	DTW	DTW (c)[a]	EDR	ERP	LCSS	LCSS (c)	Swale	Spade
50words	5	0.407	0.379	0.555	0.378	0.526	0.375	0.291	0.271	0.341	0.298	0.279	0.281	0.341
Adiac	5	0.464	0.495	0.428	0.497	0.718	0.465	0.446	0.457	0.436	0.434	0.418	0.408	0.438
Beef	2	0.4	0.55	0.583	0.55	0.683	0.433	0.583	0.4	0.567	0.402	0.517	0.384	0.5
Car	2	0.275	0.3	0.3	0.217	0.267	0.333	0.258	0.371	0.167	0.208	0.35	0.233	0.25
CBF	16	0.087	0.041	0.534	0.049	0.171	0.003	0.006	0.013	0	0.017	0.015	0.013	0.044
chlorineconcentration	9	0.349	0.374	0.325	0.368	0.44	0.38	0.348	0.388	0.376	0.374	0.368	0.374	0.439
cinc_ECG_torso	30	0.051	0.044	0.18	0.046	0.084	0.165	0.006	0.011	0.145	0.057	0.023	0.057	0.148
Coffee	2	0.193	0.246	0.087	0.196	0.427	0.191	0.252	0.16	0.213	0.213	0.237	0.27	0.185
diatomsizereduction	10	0.022	0.033	0.019	0.026	0.161	0.015	0.026	0.019	0.026	0.045	0.084	0.028	0.016
ECG200	5	0.162	0.182	0.175	0.16	0.266	0.221	0.153	0.211	0.213	0.171	0.126	0.17	0.256
ECGFiveDays	26	0.118	0.107	0.235	0.103	0.181	0.154	0.122	0.111	0.127	0.232	0.187	0.29	0.265
FaceFour	5	0.149	0.144	0.421	0.172	0.144	0.064	0.164	0.045	0.042	0.144	0.046	0.134	0.25
FacesUCR	11	0.225	0.192	0.401	0.205	0.289	0.06	0.079	0.05	0.028	0.046	0.046	0.03	0.315
fish	5	0.319	0.293	0.314	0.311	0.496	0.329	0.261	0.107	0.216	0.067	0.16	0.171	0.15
Gun_Point	5	0.146	0.092	0.186	0.084	0.175	0.14	0.055	0.079	0.161	0.098	0.065	0.066	0.007
Haptics	5	0.619	0.634	0.632	0.64	0.669	0.622	0.593	0.466	0.601	0.631	0.58	0.581	0.736
InlineSkate	6	0.665	0.646	0.715	0.65	0.757	0.557	0.603	0.531	0.483	0.517	0.525	0.533	0.643
ItalyPowerDemand	8	0.04	0.047	0.044	0.043	0.089	0.067	0.055	0.075	0.05	0.1	0.076	0.082	0.233
Lighting2	5	0.341	0.251	0.389	0.261	0.444	0.204	0.32	0.088	0.19	0.199	0.108	0.16	0.272
Lighting7	2	0.377	0.286	0.566	0.3	0.503	0.252	0.202	0.093	0.287	0.282	0.116	0.279	0.557
MALLAT	20	0.032	0.041	0.079	0.042	0.094	0.038	0.04	0.08	0.033	0.088	0.091	0.09	0.167
MedicalImages	5	0.319	0.322	0.36	0.329	0.451	0.286	0.281	0.36	0.309	0.349	0.357	0.348	0.434
Motes	24	0.11	0.082	0.24	0.08	0.211	0.09	0.118	0.095	0.106	0.064	0.077	0.073	0.103
OliveOil	2	0.15	0.236	0.167	0.216	0.298	0.1	0.118	0.062	0.132	0.135	0.055	0.097	0.207
OSULeaf	5	0.448	0.488	0.52	0.474	0.571	0.401	0.424	0.115	0.365	0.359	0.281	0.403	0.212
plane	6	0.051	0.037	0.033	0.042	0.038	0.001	0.032	0.001	0.01	0.016	0.062	0.023	0.006
SonyAIBORobotSurface	16	0.081	0.076	0.106	0.088	0.135	0.077	0.074	0.084	0.07	0.228	0.155	0.205	0.195
SonyAIBORobotSurfaceII	12	0.094	0.084	0.135	0.071	0.186	0.08	0.083	0.092	0.062	0.238	0.089	0.281	0.322
StarLightCurves	9	0.142	0.143	0.151	0.142	0.13	0.089	0.086	0.107	0.125	0.118	0.124	0.12	0.142
SwedishLeaf	5	0.295	0.286	0.357	0.299	0.347	0.256	0.221	0.145	0.164	0.147	0.148	0.14	0.254
Symbols	30	0.088	0.098	0.152	0.093	0.078	0.049	0.096	0.02	0.059	0.053	0.055	0.058	0.018
synthetic_control	5	0.142	0.146	0.227	0.158	0.64	0.019	0.014	0.118	0.035	0.06	0.075	0.06	0.15
Trace	5	0.368	0.279	0.445	0.286	0.158	0.016	0.075	0.15	0.084	0.118	0.142	0.108	0
TwoLeadECG	25	0.129	0.154	0.151	0.163	0.266	0.033	0.07	0.065	0.071	0.146	0.154	0.149	0.017
TwoPatterns	5	0.095	0.039	0.797	0.036	0.747	0	0	0.001	0.01	0	0	0	0.052
wafer	7	0.005	0.004	0.021	0.005	0.014	0.015	0.005	0.002	0.006	0.004	0.004	0.004	0.018
WordsSynonyms	5	0.393	0.374	0.53	0.375	0.529	0.371	0.315	0.295	0.346	0.294	0.28	0.274	0.322
yoga	11	0.16	0.161	0.181	0.167	0.216	0.151	0.151	0.112	0.133	0.109	0.134	0.43	0.13

Table 2: Error Ratio of Different Similarity Measures on 1NN Classifier

[a]DTW (c) denotes DTW with constrained warping window, same for LCSS.