

On the Effect of Endpoints on Dynamic Time Warping

Diego Furtado Silva
Universidade de São Paulo
diegofsilva@icmc.usp.br

Gustavo E. A. P. A. Batista
Universidade de São Paulo
gbatista@icmc.usp.br

Eamonn Keogh
University of California, Riverside
eamonn@cs.ucr.edu

ABSTRACT

While there exist a plethora of classification algorithms for most data types, there is an increasing acceptance that the unique properties of time series mean that the combination of nearest neighbor classifiers and Dynamic Time Warping (DTW) is very competitive across a host of domains, from medicine to astronomy to environmental sensors. While there has been significant progress in improving the efficiency and effectiveness of DTW in recent years, in this work we demonstrate that an underappreciated issue can significantly degrade the accuracy of DTW in real-world deployments. This issue has probably escaped the attention of the very active time series research community because of its reliance on static highly contrived benchmark datasets, rather than real world dynamic datasets where the problem tends to manifest itself. In essence, the issue is that DTW's eponymous invariance to warping is only true for the main "body" of the two time series being compared. However, for the "head" and "tail" of the time series, the DTW algorithm affords no warping invariance. The effect of this is that tiny differences at the beginning or end of the time series (which may be either consequential or simply the result of poor "cropping") will tend to contribute disproportionately to the estimated similarity, producing incorrect classifications. In this work, we show that this effect is real, and reduces the performance of the algorithm. We further show that we can fix the issue with a subtle redesign of the DTW algorithm, and that we can learn an appropriate setting for the extra parameter we introduced. We further demonstrate that our generalization is amiable to all the optimizations that make DTW tractable for large datasets.

Categories and Subject Descriptors

H.2.8 [Information Systems]: Database Application – Data Mining

Keywords

Time Series, Dynamic Time Warping, Similarity Measures

1. INTRODUCTION

Following the huge growth of applications based on temporal measurements, such as Quantified Self and Internet of Things [23], time series data are becoming ubiquitous even in our quotidian lives. It is increasingly difficult to think of a human interest or endeavor, from medicine to astronomy, that does not produce copious amounts of time series.

Among all the time series mining tasks, *query-by-content* is the most basic. It is the fundamental subroutine used to support nearest-neighbor classification, clustering, etc. The last decade has seen

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SIGKDD MiLeTS'16, August 13–17, 2016, San Francisco, CA, USA.

Copyright 2016 ACM 1-58113-000-0/00/0010 ...\$15.00.

DOI: <http://dx.doi.org/10.1145/12345.67890>

mounting empirical evidence that the unique properties of time series mean that Dynamic Time Warping (DTW) is the best distance measure for time series across virtually all domains, from activity recognition for dogs [11] to classifying star light curves to ascertain the existence of exoplanets [5].

However, virtually all current research efforts assume a perfect segmentation of the time series. This assumption is engendered by the availability of dozens of contrived datasets from the UCR time series archive [4]. Improvements on this (admittedly very useful) resource have been seen as *sufficient* to warrant publication of a new idea, but it would be better to see success on these benchmarks as being only *necessary* to warrant consideration of a new approach.

In particular, the way in which the majority of the datasets were created and "cleaned" means that algorithms that do well on these datasets can still fail when applied to real world streaming data.

The issue lends itself to a visually intuitive explanation. Figure 1 shows two examples from the Australian Sign Language dataset aligned by DTW. We can see the utility of DTW here, as it aligns the later peak of the blue (bold) time series to the earlier occurring peak in the red (fine) time series. However, this figure also illustrates a weakness of DTW. Because every point must be matched, the first few points in the red sequence are forced to match the first point in the blue sequence.

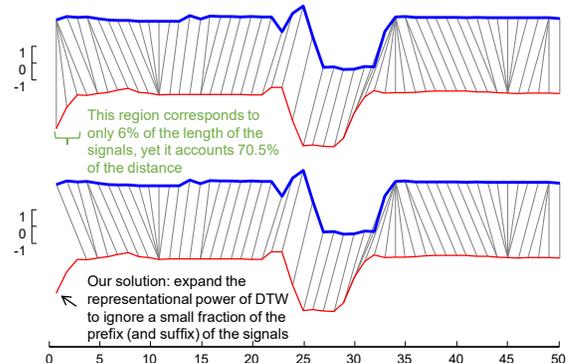


Figure 1. top) Two time series compared with DTW. While the prefix of the red (fine) time series consists of only 6% of the length, it is responsible for 70.5% of the error. **bottom)** We propose to address this disproportionate appointing of error by selectively ignoring parts of the prefix (and/or suffix)

While Figure 1 does show the problem on a real data object, the reader may wonder how common this issue is "in the wild" (again, for the most part, the UCR Archive datasets have been carefully contrived to avoid this issue). We claim that at least in some domains, this problem is very common. For example, heartbeat extraction algorithms often segment the signal to begin at the maximum of the QRS complex [22]. However, as shown in Figure 2 this location has the greatest viability in its prefixes and suffixes.

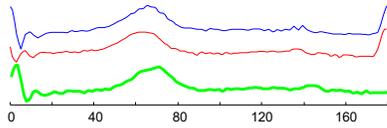


Figure 2. Three heartbeats taken from a one-minute period of a healthy male. The beats were extracted by a state-of-the-art beat extraction algorithm [19], but there is significant variation in the prefix (all three) and in the suffix (green vs. the other two).

Similar remarks apply to gait cycle extraction algorithms [24]. Likewise, star light curves, for which DTW is known to be very effective, have cycles extracted by a technique called *universal phasing* [18]. However, universal phasing has the unfortunate side effect of placing the maximum variance at the prefix and suffix of the signals.

In this work, we address this problem of uninformative and undesirable “information” contained just *before* and just *after* the temporal measurement of informative data. For the sake of clarity, we will refer to these unwanted values as *prefix* and *suffix* information, and use *endpoints* to refer to both.

Our approach is simple and intuitive, but highly effective. We modify the endpoint constraint of Dynamic Time Warping (DTW) to provide endpoint invariance. The main idea behind our proposal is allowing DTW to ignore some leading/trailing values in one or both of the two time series under comparison. While our idea is simple, it must be carefully executed. It is clear that ignoring too much (useful) data is just as undesirable as paying attention to spurious data.

We note that somewhat similar observations were known to the signal processing community when DTW was the state-of-the-art technique for speech processing (in the 1980’s and 90’s before being superseded by Markov models [15]). However, the importance of endpoint invariance for time series seems to be largely unknown or underappreciated [10][16][17].

We can summarize the main contributions of this paper as follows:

- We draw the data mining community’s attention to the endpoint invariance for what is, to the best of our knowledge, the first time;
- We propose a modification of the well-known algorithm Dynamic Time Warping to provide invariance to suffix and prefix;
- Although simple and intuitive, we show that our method can considerably improve the classification accuracy when warranted, and just as importantly, our ideas do not reduce classification accuracy if the dataset happens to not need endpoint invariance;
- Unlike other potential fixes, our distance measure respects the property of symmetry and, consequently, can be applied in a multitude of data mining algorithms with no pathological errors caused by the order of the objects in the dataset;
- In spite of the fact that we must add a parameter to DTW, we show that it is possible to robustly learn a good value for this parameter using only the training data.

The remainder of this paper is organized as follows. Section 2 formalizes the concept of time series suffix and prefix and shows intuitive examples of how it affects the distance measurement, and therefore, the classification accuracy. Section 3 summarizes the main concepts necessary to understand our proposal (in particular, a detailed review of the Dynamic Time Warping algorithm). Section 4 places our ideas in the context of related work. We

explain our proposed method in detail in Section 5. In Section 6, we empirically verify the utility of our ideas on synthetic and real data. Having shown our ideas are *effective*, Section 7 explains how to adapt state-of-the-art lower bound functions to the distance measure proposed in this paper, a critical step to maintain efficiency. Finally, in Section 8 we offer conclusions and directions for future work.

2. Time Series Suffix and Prefix

Most research efforts for time series classification assume that all the time series in the training and test sets are carefully segmented by using the precise endpoints of the desirable event [17][18][26][28]. Despite the ubiquity of time series datasets that fulfill such an assumption, in practical situations the exact endpoints of events are difficult to detect. In general, a perfectly segmented dataset can only be achieved by manual segmentation or some contrivance that uses external information.

To see this, we revisit the Gun-Point dataset, which has been used in more than two hundred papers to test the accuracy of time series classification [4]. As shown in Figure 3, the data objects considered here do have perfectly flat prefixes and suffixes. However, these were obtained only by carefully prompting the actor’s movements with a metronome that produced an audible cue every five seconds.

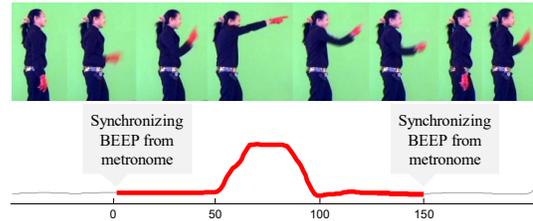


Figure 3. The ubiquitous Gun-Point dataset was created by tracking the hand of an actor (*top*). However, the perfectly flat prefix and suffix were due to carefully training the actor to have her hand immobile by her side one second before and one second after the cue from a metronome (*bottom*)

In more realistic scenarios, the movement of pointing a gun/finger must be detected among several different movements. Before drawing the weapon, the actor could be running, talking on a cell phone, etc.

For example, consider the scenario in which some movement was performed just before the weapon was aimed. In addition, another movement started immediately after the gun was returned to the holster. In this case, the time series could have a more complex shape as shown in Figure 4. As visually explained in Figure 1, it is clear that prefix and suffix would greatly prejudice the distance estimation in this case.



Figure 4. Example of a time series containing the event to be classified (in blue) and prefix and suffix information (in red)

Another possible issue that can result from automatic segmentation is illustrated in Figure 5. In this case, the algorithm used to extract the time series was too “aggressive” and made the mistake of truncating the last few observations of the event of interest. Obviously, a similar issue could also happen at the *beginning* of the signal.



Figure 5. An example of an extracted time series containing an incomplete subset of the shape to be classified (in blue) and a prefix information (in red)

In this case, the time series is missing its true suffix. Even with such missing information, the shape that describes the beginning of the action *may* be enough such that it will be classified correctly. However, the object that would otherwise be considered its nearest neighbor may contain information of the entire movement, as shown in Figure 4. To classify the badly cropped item in Figure 5 correctly, a distance measure must avoid matching the last few observations of the complete event to the values observed in our badly segmented event. In Section 5 we will show how our method can solve these issues.

3. Definitions and Background

A time series x is a sequence of n ordered values such that $x = (x_1, x_2, \dots, x_n)$ and $x_t \in \mathbb{R}$ for any $t \in [1, n]$. We assume that two consecutive values are equally spaced in time or the interval between them can be disregarded without loss of generality. For clarity, we refer each value x_t as *observation*.

The Dynamic Time Warping (DTW) algorithm is arguably the most useful distance measure for time series analysis¹. For example, mounting empirical evidence strongly suggest that the simple nearest neighbor algorithm using DTW outperforms more “sophisticated” time series classification methods in a wide range of application domains [28].

In contrast to other distance measures, such as those in the L_p -norm family, the DTW computes a non-linear alignment between the observations of the two time series being compared. In other words, while L_p -norm distances are only able to compare the value x_t to a value y_t of a time series y , DTW is able to compare x_t to y_s such that $t \approx s$.

To compute the optimal non-linear alignment between a pair of time series x and y , with lengths n and m respectively, the DTW typically bound to the following constraints:

- **Endpoint constraint.** The matching is made for the entire length of time series x and y . Therefore, it starts at the pair of observations $(1,1)$ and ends at (n, m) ;
- **Monotonicity constraint.** The relative order of observations must be preserved, i.e., if $s_1 < s_2$, the matching of x_t with y_{s_1} is done before matching x_t with y_{s_2} ;
- **Continuity constraint.** The matching is made in one-unit steps. It means that the matching never “jumps” one or more observations of any time series.

The calculation of DTW distance is performed by a dynamic programming algorithm. The initial condition of such an algorithm is defined by Equation 1.

$$dtw(i, j) = \begin{cases} \infty, & \text{if } (i = 0 \text{ or } j = 0) \text{ and } i \neq j \\ 0, & \text{if } i = j = 0 \end{cases} \quad (1)$$

In order to find the optimal non-linear alignment between the observations of the time series x and y , DTW follows the recurrence relation defined by Equation 2.

$$dtw(i, j) = c(x_i, y_j) + \min \begin{cases} dtw(i-1, j) \\ dtw(i, j-1) \\ dtw(i-1, j-1) \end{cases} \quad (2)$$

where $i \in [1, n]$ and $j \in [1, m]$, m being the length of the time series y . The partial $c(x_i, y_j)$ represents the cost of matching two observations x_i and y_j and is calculated by the squared Euclidean distance between them. Finally, the DTW distance returned is $DTW(x, y) = dtw(n, m)$.

An additional constraint commonly applied to DTW is the *warping* constraint. This constraint limits the time difference that the algorithm is allowed to match the observations. In the matrix view of DTW, this constraint limits the algorithm to calculate the values of the DTW matrix in a region close to its main diagonal. The benefit of using a warping constraint is two fold: the DTW calculation takes less time (as it is not necessary to calculate values for the entire distance matrix) and it avoids pathological alignments. For example, when comparing heartbeats, we want to allow a little warping flexibility to be invariant to small (and medically irrelevant) changes in timing. However, it never makes sense to attempt to align ten heartbeats to twenty-five heartbeats. The *warping* constraint prevents such degenerate solutions. As a practical confirmation of its utility using the constraint, we note that it has been shown to improve classification accuracy [17].

The most common warping constraint for DTW is the Sakoe-Chiba warping window [20]. The use of warping constraints adds a parameter to be set by the user. However, several studies show that small windows (usually smaller than 10%) are usually a good choice for nearest neighbor classification [17].

4. Related Work

The utility of relaxing the endpoint constraint of DTW has been previously noticed by the signal processing community, in the context of speech [7] and music analysis [13]. However, the issue seems to be unknown or glossed over in time series data mining.

The time series mining method that shares more similarities to our proposal is the open-end DTW (OE-DTW) [25]. However, OE-DTW was proposed to match incomplete time series to complete references. In other words, such a method is based on the assumption that we can construct a training set with carefully cropped time series and we can know the exact point that represents the beginning of the time series to be classified.

Specifically, OE-DTW is a method that allows ignoring any amount of observations at the end of the training time series. The final distance estimate is the value represented by $\min_{0 \leq i \leq m} DTW(n, i)$, i.e., the final distance is the minimum value in the last column of the DTW matrix.

A weakness of OE-DTW is that it does not consider the existence of prefix information. A modification of the OE-DTW called open-begin-end DTW (OBE-DTW) or subsequence DTW [12] mitigates this issue. OBE-DTW allows the match of observations to start at any position of the training time series. To allow DTW to do this, the algorithm needs to initialize the entire first column of the DTW matrix with zeros.

Although OBE-DTW recognizes that both prefix and suffix issues may exist, it only addresses the problem in the training time series. A more important observation is that OBE-DTW is not symmetric, which severely affects its utility. For example, the results obtained

¹ Note that DTW subsumes the *second* most useful measure, the Euclidean distance, as a special case.

by OBE-DTW in any clustering algorithm are dependent on the order in which the algorithm processes the time series. To see this, consider the hierarchical single-linkage clustering algorithm [29]. Figure 6 shows the result of clustering the same set of five time series objects from the Motor Current dataset (c.f. Section 6.2.1), presented in different orders to the clustering algorithm. Specifically, the distance between the time series x and y is calculated by $OBE DTW(x, y)$ in the first case and by $OBE DTW(y, x)$ in the second. Note that the results are completely different, a very undesirable outcome.

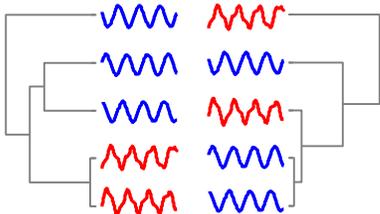


Figure 6. Clustering results of the same dataset by using OBE-DTW. The difference between the results is given by the fact that they were obtained by presenting the time series in a different order to the clustering algorithm

In addition to this issue, OBE-DTW has one other fatal flaw. In essence, it can be “too invariant,” potentially causing meaningless alignments in some cases. Figure 7 shows an extreme example of this. In the top figure, all observations of flat line match to a single observation in the sine wave, and the DTW distance obtained is 0.07. In the bottom figure, we reverse the roles of reference and query. This time, all observations of the sine wave match to a single observation in the flat line, and the DTW distance obtained is 69.0. We observe a three orders of magnitude difference in the DTW results.

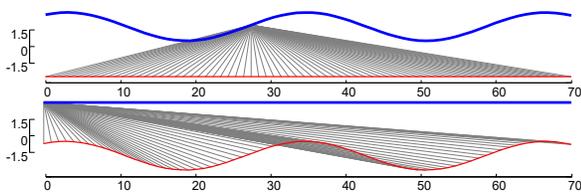


Figure 7. The OBE-DTW alignment for the same pair of time series. In the first (top), a sine wave is used as reference and the flat line is used as query. In the second (bottom), the same sine wave is used as query while the flat line is used as reference

Similar to the OBE-DTW, the method proposed in this paper is based on a relaxation of the endpoint constraint. However, our method is symmetric and strictly limits the amount of the signals that can be ignored, preventing the meaningless alignments shown in Figure 7. Figure 8 shows a comparison of the results obtained by the classic DTW, the OBE-DTW, and the distance measure proposed in this work when used to cluster the time series data considered in Figure 6.

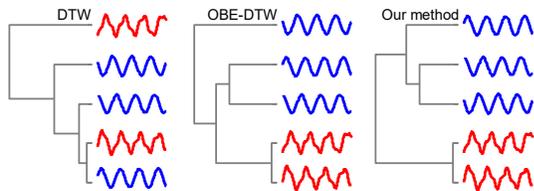


Figure 8. Clusterings on a toy dataset using the classic DTW (left), OBE-DTW (center), and the distance measure proposed in this paper (right). Note that our method achieves a perfect and intuitive separation of the different classes

5. Prefix and Suffix-Invariant DTW (ψ -DTW)

While there are many different methods proposed for time series classification (decision trees, etc.), it is known that the simple nearest neighbor is extremely competitive in a wide range of applications and conditions [28]. Given this, the only decision left to the user is the choice of the distance measure.

In most cases, this choice is guided by the invariances required by the task and domain [1]. In conjunction with simple techniques, such as z-normalization, DTW can provide several invariances like amplitude, offset and the warping (or local scaling) itself.

In this work, we address what we feel is the “missing invariance,” the invariance to spurious prefix and suffix information. Given the nature of our proposal, we call our method Prefix and Suffix-Invariant DTW, or simply PSI-DTW (or ψ -DTW).

The relaxed version of the endpoint constraint proposed in this work is defined as the following.

Relaxed endpoint constraint. Given an integer value r , the alignment path between the time series x and y starts at any pair of observations in $\{(1, c_1 + 1)\} \cup \{(c_1 + 1, 1)\}$ and ends at any pair in $\{(n - c_2, m)\} \cup \{(n, m - c_2)\}$, such that c_1 and $c_2 \in [0, r]$.

This relaxation of the endpoint constraint can avoid undesirable matches at the beginning and the end of any x or y time series by removing the obligation for the alignment path to start and end with specific pairs of observation, namely the first and the last pairs. The value r used in this definition is the *relaxation factor* parameter that needs to be defined by the user.

We recognize the general undesirability of adding a new parameter to an algorithm. However, we argue it is necessary (c.f. Section 4). In addition, we show that we are able to learn an appropriate r solely from the training data. We will return to this topic in Section 6.3.

An important aspect of the proposed endpoint constraint is the fact that, by definition, the same number of cells is “relaxed” for both column and row in the cumulative cost matrix. This is what guarantees the symmetry of ψ -DTW. If the number of relaxed columns and rows was different, the starting and finishing cells of the alignment found by ψ -DTW(x, y) could be outside of the region defined by the endpoint constraint in the cost matrix used by ψ -DTW(y, x).

The relaxation of endpoints slightly affects the initialization of the DTW estimation algorithm defined in Equation 1. To accomplish the new constraint, the initialization of DTW needs to be changed to Equation 3.

$$dtw(i, j) = \begin{cases} \infty, & \text{if } i = 0 \text{ and } j > r \\ 0, & \text{if } i = 0 \text{ and } j \leq r \\ \infty, & \text{if } j = 0 \text{ and } i > r \\ 0, & \text{if } j = 0 \text{ and } i \leq r \end{cases} \quad (3)$$

After this initialization, the recurrence relation to fill the matrix is unchanged; it is exactly the same as defined by Equation 2.

Finally, the ultimate distance estimate is not necessarily obtained by retrieving the value in $dtw(n, m)$. This minor modification can be directly obtained by the definition of the proposed relaxed endpoint constraint. Formally, the final distance calculation is given by Equation 4.

$$\psi - DTW(x, y, r) = \min_{(i, j) \in finalSet} [dtw(i, j)], \quad (4)$$

$$finalSet = \{(n - c, m)\} \cup \{(n, m - c)\} \forall c \in [0, r].$$

The algorithm to calculate ψ -DTW is simple. For concreteness, Table 1 describes it in detail.

Table 1. ψ -DTW algorithm

Procedure ψ-DTW(x,y,r)	
Input: Two user provided time series, x and y and the relaxation factor parameter r	
Output: The ψ -DTW distance between x and y	
1	$n \leftarrow \text{length}(x), m \leftarrow \text{length}(y)$
2	$M \leftarrow \text{infinity_matrix}(n+1, m+1)$
3	$M([0,r], 0) \leftarrow 0$
4	$M(0, [0,r]) \leftarrow 0$
5	for $i \leftarrow 1$ to n
6	for $j \leftarrow 1$ to m
7	$M(i,j) \leftarrow c(x_i, y_j) + \min(M(i-1, j-1), M(i, j-1), M(i-1, j))$
8	$\min X \leftarrow \min(M([n-r, n], m)), \min Y \leftarrow \min(M(n, [m-r, m]))$
9	return $\min(\min X, \min Y)$

The algorithm starts by defining the variables used to access the length of time series (line 1) and the DTW matrix according to Equation 3 (lines 2 to 4). The *for* loops (lines 5 to 7) fill the matrix according to the recurrence relation defined in Equation 2. Finally, the algorithm finds the minimum value in the region defined by the new endpoint constrained and returns it as the distance estimate (lines 8 and 9). To implement the constrained warping version of this algorithm, one only needs to modify the interval of the second *for* loop (line 6) according to the constraint definition.

Note that the proposed method is a generalization of DTW, thus it is possible to obtain the classic DTW by our method. Specifically, if $r = 0$, the final result of our algorithm is exactly the same as the classic DTW.

6. Experimental Evaluation

We begin this section by reviewing our experimental philosophy. We are committed to reproducibility, thus we have made available all the source code, datasets, detailed results and additional experiments in a companion website for this work [21]. In addition to reproducing our experiments, the interested reader can use our code on their own datasets. We implemented all our ideas in Matlab, as it is ubiquitous in the data mining community.

To test the robustness of our method, we compare its performance against the accuracy obtained by the classic, unconstrained DTW. In addition, we present results obtained using constrained-warping. We refer to the constrained versions of the algorithms with names containing the letter *c*. Specifically, *cDTW* refers to the DTW with warping constraint. Similarly, ψ -*cDTW* stands for the constrained version of ψ -DTW.

We are not directly interested in studying the effect of warping window width on classification accuracy. The value of the warping window width parameter has been shown to greatly affect accuracy, but it has also been shown to be easy to learn a good setting for this parameter with cross validation [17][26][28]. For simplicity, we fixed it as 10% of the length of the query time series by default.

However, this setting limits the choice of the relaxation factor to ψ -DTW. For any relaxation factor that is greater than or equal to the warping length, the distance is the same. For this reason, when we wanted to test the effect of larger relaxation factors, the warping window used in the experiment was set by the same value as r .

We divide our experimental evaluation into two sections.

- In order to clearly demonstrate that our algorithm is doing what we claim it can, we take *perfectly* cropped time series data and add increasing amounts of spurious endpoint data. This experiment simulates the scenario in which the

segmentation of time series is not perfect, i.e., there are endpoints that may represent random behaviors.

- The experiments above will be telling, but unless *real* datasets have the spurious endpoint problem, they will be of little interest to the community. Thus, we apply ψ -DTW on real datasets that we suspect have a high probability of the presence of spurious endpoints.

For clarity of presentation, we have confined this work to the single dimensional case. However, our proposal can be easily generalized to multidimensional data.

6.1 The Effect of ψ -DTW on Different Lengths of Endpoints

As noted above, the UCR Time Series Archive has been useful to the community working on time series classification [4]. However, in general, the highly contrived procedures used to collect and/or clean most of the datasets prevent the appearance of prefixes and suffixes (recall Figure 3). For this reason, the impact of endpoints cannot be directly evaluated by the use of such datasets.

However, such “endpoint-free” data create a perfect starting point to understand how different amounts of uninformative data can affect both DTW and ψ -DTW. To see this, we consider some datasets that are almost certainly free of specious prefix or suffix information. To these we prepend and postpend random walk subsequences with length varying from 0% to 50% of the original data. Next, we compared the accuracy obtained using the nearest neighbor classification on the modified datasets using both DTW and ψ -DTW. At each length of added data, we average over three runs with newly created data.

At this point, we are not learning the parameter r . Instead, we fixed both the relaxation factor and warping constraint length as 10% of the time series being compared.

Intuitively, as we add more and more spurious data, we expect to see greater and greater decreases in accuracy. However, we expect that ψ -DTW degrades slower. In fact, this is the *exact* behavior observed in our experiments. Figure 9 shows the results on the Cricket X dataset.

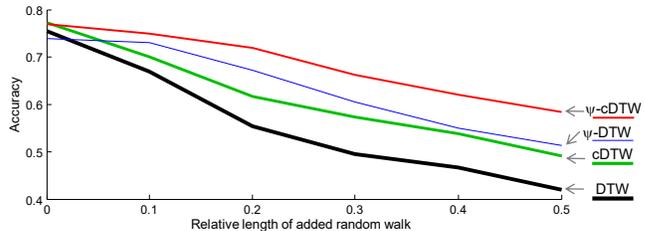


Figure 9. The accuracy after padding the Cricket X dataset with increasing lengths of random walk data. When no such spurious data is added, the accuracy obtained by the classic DTW is very slightly better. As we encounter increasing amounts of spurious data, ψ -DTW and ψ -cDTW degrade less than DTW and cDTW

For brevity, here we show the results on only one dataset. However, we note that this result describes the general behavior of the results obtained in other datasets. We invite the interested reader to review some additional experiments in our website [21].

6.2 Case Studies

In the previous experiment, we showed the robustness of ψ -DTW in the presence of spurious prefix and suffix information in artificially contrived time series data. In this section, we evaluate our method on real data.

The datasets we consider were extracted in a scenario in which we do not have perfect knowledge or control over the events' endpoints. In some cases, the original datasets were obtained by recording sessions similar to the Gun-Point dataset (c.f. Section 2), in which the invariance to endpoints is enforced by the data collection procedure. In this case, we model the real world conditions by ignoring the external cues or annotations. In particular, we simulated a randomly-ordered stream of events followed by a classic subsequence extraction step. For this phase, we considered the simple sliding window approach. For additional details on the extraction phase, please refer to [21].

In keeping with common practice, we adopted the use of dictionaries as training data. A data dictionary is a subset of the original training set containing only its most relevant examples. The utility of creating dictionaries is two-fold [8]: it makes the classifier faster and the accuracy obtained by dictionaries is typically better than that obtained by using *all* the data, which may contain outliers or mislabeled data.

To compute the relevance of training examples to the classification task, we used the SimpleRank function [26]. This function returns a ranking of exemplars according to their estimated contribution to the classification accuracy. Then, we selected the top- k time series of each class in the dictionary, with k empirically discovered for each dataset.

The main intuition behind SimpleRank is to define a score for each exemplar based on its "neighborhood." For each exemplar t_j , its nearest neighbor s is "rewarded" if it belongs to the same class, i.e., s is used to correctly classify t_j . Otherwise, s is "penalized" by having its score decreased. Equation 5 formally defines the SimpleRank function.

$$rank(s) = \sum_j \begin{cases} 1, & \text{if } class(s) = class(t_j) \\ -\frac{2}{num_classes - 1}, & \text{otherwise} \end{cases} \quad (5)$$

The length of subsequences and the size of the dictionary for each dataset were chosen in order to obtain the best accuracy in the training set by using constrained DTW. In addition, the SimpleRank used to construct the dictionaries was also implemented by using the classic constrained DTW instead of the distance measure proposed in this work. This was done to ensure we are not biasing our experimental analysis in favor of our method.

6.2.1 Motor Current Data

Our first case study considers electric motor current signals. This dataset has long been a staple of researchers interested in prognostics and novelty detection [14]. We refer the reader interested in the procedure to generate such data to [6].

The data in question includes 21 classes representing different operating conditions. In addition, a class that represents to (a slight) diversity of healthy operation, the other classes represent different defects in the apparatus (in particular, one to ten broken bars and one to ten broken end-ring connectors).

The original data used in this study is segmented, but with no attention paid to avoiding suffix or prefix inconsistencies. Therefore, in this case, we did not use the approach of simulating a data stream. We segmented the original time series using a static window placed in the middle of each time series. With this procedure, the signals have different endpoints in each different length we consider. Figure 10 shows the classification results.

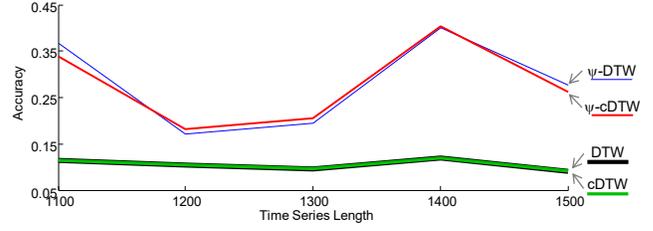


Figure 10. Classification results obtained by varying the time series length on the Motor Current dataset

Given that this dataset is a very clear case of badly-defined endpoints, these results show the robustness of our proposal. Over all lengths we experimented with, ψ -DTW beats DTW by a large margin. Specifically, ψ -DTW can achieve accuracy rates as high as 40% while the best result achieved by the classic DTW is lower than 12%.

6.2.2 Robot Surface and Activity Identification

In this case study, we consider the classification of signals collected by the accelerometer embedded in a Sony ERS-210 Aibo Robot [27]. This robot is a dog-like model equipped with a tri-axial accelerometer to record its movements.

Using the streaming data sets collected by this robot, we evaluated the classification accuracy in two different scenarios: *surface* and *activity* recognition. In the former scenario, the goal is to identify the type of surface in which the robot is walking on. Specifically, the target classes for this problem are carpet, field, and cement. Figure 11 shows the results for this dataset.

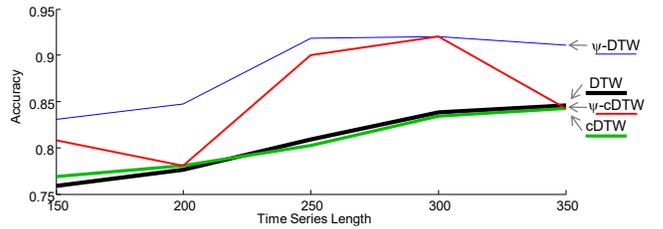


Figure 11. Classification results obtained by varying the time series length on the Sony AIBO Robot Surface dataset

In the second scenario, the aim is the identification of the activity performed by the robot. In this case, the target classes are the robot playing soccer, standing in a stationary position, trying to walk with one leg hooked, and walking straight into a fixed wall. Figure 12 shows the results obtained in this scenario.

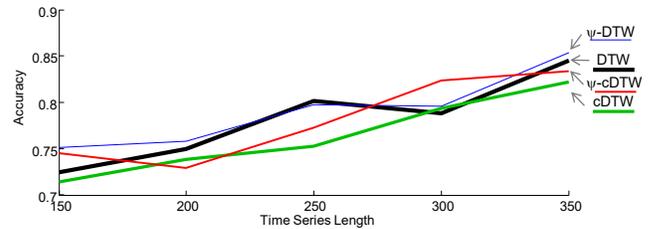


Figure 12. Classification results obtained by varying the time series length on the Sony AIBO Robot Activity dataset

In both scenarios evaluated in this study, the results obtained by ψ -DTW are generally better than the classic DTW. However, there is an important caveat to discuss. Despite the improvements in accuracy in most time series lengths, the accuracy obtained by ψ -DTW was the same or slightly worse than the performance of the classic DTW in a few experiments. This happened because our procedure to learn the relaxation factor was not able to find a more

suitable value in these cases. Even in these cases, the poor choice of r did not significantly affect the classification accuracy. Even so, these results highlight the importance of the parameter learning procedure, which we describe in detail in Section 6.3.

6.2.3 Gesture Recognition

Gesture recognition is one of the most studied tasks in the time series classification literature. The automatic identification of human gestures has become an increasingly popular mode of human-computer interaction.

In this study, we used the Palm Graffiti Digits dataset [1], which consists of recordings of different subjects “drawing” digits in the air while facing a 3D camera. The goal of this task is the classification of the digits drawn by the subjects. Figure 13 shows the results.

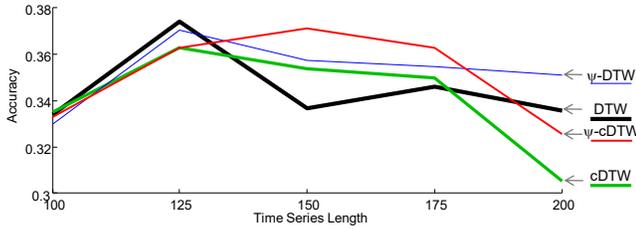


Figure 13. Classification results obtained by varying the time series length on the Palm Graffiti Digits dataset

Similar to our findings with the robot data, the accuracy rates obtained by our proposal are usually better than the obtained by the classic DTW. In few cases, the accuracy is slightly worse. However, most important is the robustness of ψ -DTW to the cases where the prefixes and suffixes seem to significantly affect the classification. For instance, there is an expressive loss of accuracy obtained by the classic DTW in the dataset containing time series with 150 observations. The lost is notably less drastic when we using ψ -DTW.

6.2.4 Sign Language Recognition

Another specific scenario with gesture data used in this work is the recognition of sign language. A sign language is an alternative way to communicate by gestures and body language that replace (or augment) the acoustic communication. In this work we used a dataset of Australian Sign Language (AUSLAN) [9]. The original dataset is composed of signs separately recorded in different sections. We used 10 arbitrarily chosen signs of each recording session displaced as a data stream. Figure 14 shows the results.

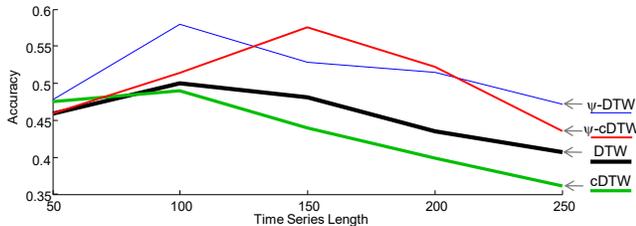


Figure 14. Classification results obtained by varying the time series length on the AUSLAN dataset

In contrast to the previous gesture recognition case, the accuracies obtained by relaxing the endpoint constraint are always better for this dataset. More importantly, the best accuracy rates were significantly superior when using ψ -DTW.

6.2.5 Human Activity Recognition

Due to the growth in the use of mobile devices containing movement sensors (such as accelerometers and gyroscopes), there

is also a notable increase in the interest of human activity analyses using this kind of equipment.

In this final case study, we investigate the robustness of ψ -DTW in the recognition of human activities using smartphone accelerometers. For this purpose, we used a the dataset that first appeared in [2]. Originally, the recordings are composed of 128 observations of three coordinates of the device’s accelerometers. In our study, we used the x-coordinate disposed in a streaming fashion. Figure 15 shows the results.

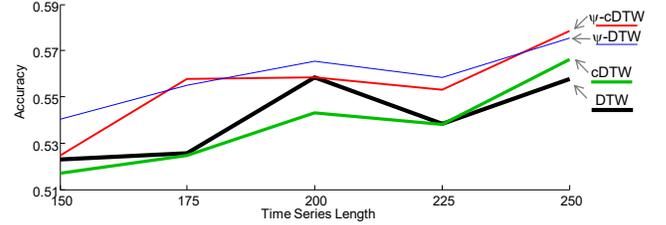


Figure 15. Classification results obtained by varying the time series length on the Human Activity Recognition dataset

Again, the accuracy obtained by ψ -DTW is better than the obtained by the classic DTW in all the cases for this dataset. This success of these results is due to, in part, to a good choice of value to the relaxation factor. This is the main topic of the following section.

6.3 On Learning the Relaxation Factor

The choice of the method to learn a value to set the parameter r may be a critical step to the use of ψ -DTW. For this reason, we devote this section to discuss this topic in details.

We start by demonstrating the sensitivity of ψ -DTW to the relaxation factor. In this experiment, we executed the classification of five random test sets for each dataset used as a study case. For each execution, we annotated the best and worst result, i.e., the accuracy obtained by the best and the worst choice of r . Figure 16 shows these results on the AUSLAN dataset. In this case, we can see that a bad choice of r always results in worse accuracy rates than the classic DTW. On the other hand, a good choice will improve the classification accuracy in all the cases. The goal of learning the relaxation factor is to approximate as much as possible to the best case.

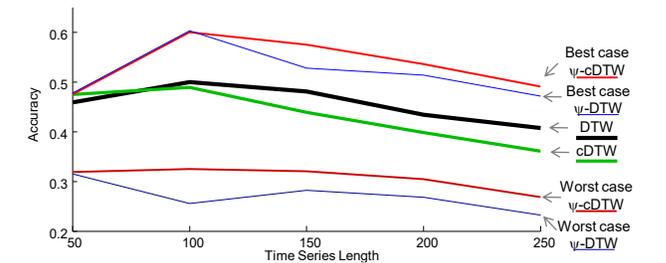


Figure 16. Accuracies obtained in the AUSLAN dataset by the best and worst values of relaxation factor

For some datasets, such as Motor Current, a poor choice of r does not result in worse accuracy than the classic DTW. In fact, the worst case for any time series length in this dataset is given by choosing $r = 0$, i.e., the classic DTW. However, the optimal choice of parameter value has a highly positive impact on the classification.

In our experiments, we experimented with a wide range of possible values to r . We set r as a relative value to the length of the time series under comparison. Specifically, we used a set of values $rl_r \in \{0, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5\}$, such that $r = \lceil n * rl_r \rceil$, where n is the length of the time series.

We limited the value of r to be at most half the number of observations of the time series in order to avoid meaningless alignments, such as the ones obtained by OBE-DTW in the example illustrated in Figure 7. Considering the same time series of that example, if we have decided to use $rl_r = p\%$, $\forall p \in [0,100]$, exactly $p\%$ of the values of the time series would be ignored.

Besides defining a range of values to evaluate, we need to define a procedure to perform such evaluation. The first obvious choice is to use the same training data and compute the accuracy obtained by different values of the parameter by applying a cross-validation or a leave-one-out procedure.

However, recall that we are using dictionaries as training data. Note that the choice of the size of the dictionary is a crucial determinant of the time complexity of the algorithm. For this reason, the number of examples in the dictionary tends to be small in order to keep the algorithm fast, which makes learning r difficult if we use the data in the dictionary exclusively.

For clarity, we measured the accuracy obtained by learning the relaxation factor by varying the size of the validation set. In this experiment, we used the training time series outside the dictionary to learn the parameter. Given a choice of the validation set size, we randomly chose examples to compose the validation set and only took into account the accuracy resulted from the best choice of the relaxation factor. In order to avoid results obtained by chance, we repeated this procedure 50 times. Figure 17 illustrates an example of the results obtained by this procedure.

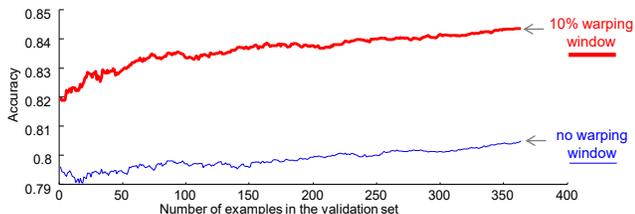


Figure 17. Accuracy obtained by learning the relaxation factor using different sizes of validation set for Sony Aibo Robot Surface dataset with time series length of 250 observations

We performed this experiment on a wide range of datasets (c.f. [21]). The results for all these datasets confirm the generality of the behavior of increasing accuracy according to the increase in the number of objects in the validation set. For this reason, in order to learn the value of r , we used a validation set containing all the training time series but those chosen as part of the dictionary. The use of the exemplars in the dictionary creates a bias for learning $r = 0$ when the nearest neighbor of a time series in the dictionary would be itself, independent of the relaxation factor. In this case, we would choose for the smallest value, i.e., $r = 0$.

7. Lower Bounding of ψ -DTW

One of the biggest concerns while designing a new distance measure is time efficiency. This is more prevalent in our case since we are proposing a modification of Dynamic Time Warping, an $O(n^2)$ algorithm. In fact, a straightforward implementation of the nearest neighbor algorithm under DTW makes its use impractical on large datasets. For this reason, the community has proposed several methods to improve the efficiency of the similarity search under DTW.

A recent paper on speeding-up similarity search [16] shows that the combination of few simple techniques makes possible to handle truly massive data under DTW. We claim that all these methods can be applied to the ψ -DTW with simple or no modifications. Some of the most important speed-up methods rely on the use of a

lower bound (LB) function. A LB function returns a value certainly lower or equal to the true DTW between two objects. Our algorithm is amenable to adaptation of LB functions.

Before explaining how to adapt LB functions to ψ -DTW, we briefly explain the intuition behind the use of LB on time series similarity search. Consider that we have a variable *best-so-far* that stores the distance to the nearest neighbor know up to the current iteration of the search algorithm. We can use this information to decide if we can avoid the expensive calculation of DTW. In order to do this, for each time series in the training set, we first calculate the LB of the distance between it and the query. Clearly, if the LB function returns a value greater than the best-so-far, the training object is not the nearest neighbor of the query. Therefore, the current object can be discarded before having its distance to the query estimated. We can extend this to a k -nearest neighbor scenario by simply replacing the best-so-far by the distance to the k -th nearest object known at that moment.

This approach for pruning DTW calculations is only effective if the LB function has the following properties: (i) its calculation is fast; (ii) and it is tight, i.e., its value is close to the true DTW. Clearly, these requirements imply a tightness-efficiency trade-off. For instance, by simply using the value 0, we have an instantly-calculated LB. However, this will never prune any distance calculation. On the other hand, the classic DTW is also a LB function itself, with the exact value as the actual distance. However, its calculation is slow.

Now we are in position to answer the following question. *How can we use previously proposed LB functions with ψ -DTW?*

We first note that ψ -DTW actually lower bounds the DTW, as exemplified in Figure 18. From a practical standpoint, the alignment path that starts at the first pair of observations and finishes by matching the last one is a possible alignment found by ψ -DTW that correspond to the exact classic DTW. Any other alignment found is considered optimal only in the case in which it provides a smaller value than the one obtained by DTW. This situation occurs when our method disregards some pair of observations that contributes to the total cost of matching.

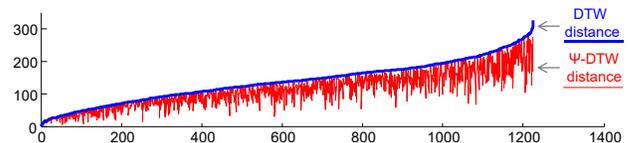


Figure 18. The distance between all the pairs of fifty time series objects in the AUSLAN dataset sorted by their DTW distances. In this experiment, we used both warping constraint and relaxation factor as 10% of the length of time series

For this reason, it is not possible to apply most of the known LB functions directly to our method. Adapting a LB function to ψ -DTW requires the careful analysis of the possible first and last pairs of observations. For sake of exemplification, we will adopt the most widely used LB function, the LB_Keogh [10]. This LB function has arguably the best tightness-efficiency trade-off.

The calculation of LB_Keogh consists of two main steps. The first step is the estimation of an envelope to a given query time series q of length n . Specifically, the envelope is composed of an upper sequence $U = (U_1, U_2, \dots, U_n)$ and a lower sequence $L = (L_1, L_2, \dots, L_n)$ defined by Equation 6.

$$\begin{aligned} U_i &= \max_{i-w \leq j \leq i+w} (q_j) \\ L_i &= \min_{i-w \leq j \leq i+w} (q_j) \end{aligned}, 1 \leq i \leq n \quad (6)$$

where w is the length of the warping constraint window. Clearly, the partials $i - w$ and $i + w$ are restricted to the extent of the query. Figure 19 exemplifies the upper and lower sequences of a given query time series.

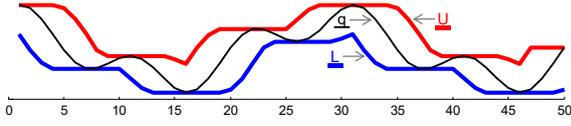


Figure 19. Upper and lower sequences of a given query time series q estimated by LB_Keogh

Once the envelope is calculated, we are in the position to estimate the value of the LB function. For each time series t to be compared to the query q , the value LB_Keogh is calculated as the Euclidean distance between the observations of t that falls outside the envelope and the nearest upper or lower sequence. Formally, the LB_Keogh between t and q is defined by Equation 7.

$$LB_Keogh(q, t) = \sum_{i=1}^n \begin{cases} (t_i - U_i)^2, & \text{if } t_i > U_i \\ (L_i - t_i)^2, & \text{if } t_i < L_i \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

A careful reader might notice that this is the squared value of the original LB_Keogh. This is because the DTW defined in Section 3 also returns the squared version of the traditional DTW. Figure 20 illustrates this step in the comparison of the previously used query q and a specific time series t .

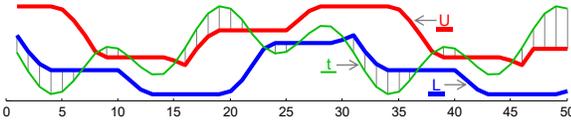


Figure 20. The LB_Keogh is calculated by using the values of the time series t that fall outside the region bounded by the envelope

The only issue in directly applying LB_Keogh to lower bound ψ -DTW is the fact that it is constrained by the classic endpoint constraint of DTW. Therefore, in order to adapt LB_Keogh to our method, we need to relax its endpoints. Since ψ -DTW can skip the matching of the first and last r observations in either q or t , the LB function should ignore these values. We call the adapted LB function ψ -LB_Keogh, and define it formally in Equation 8.

$$\psi - LB_Keogh(q, t) = \sum_{i=r+1}^{n-r} \begin{cases} (t_i - U_i)^2, & \text{if } t_i > U_i \\ (L_i - t_i)^2, & \text{if } t_i < L_i \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

The only difference between Equations 7 and 8 is the range of i . Figure 21 illustrates the ψ -LB_Keogh between q and t .

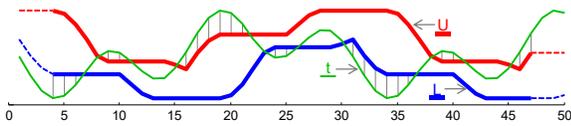


Figure 21. ψ -LB_Keogh ignores the values in the dashed regions

To visualize the tightness of ψ -LB_Keogh, Figure 22 shows the distances and their respective LB for all the pairs of objects in the training set of AUSLAN data.

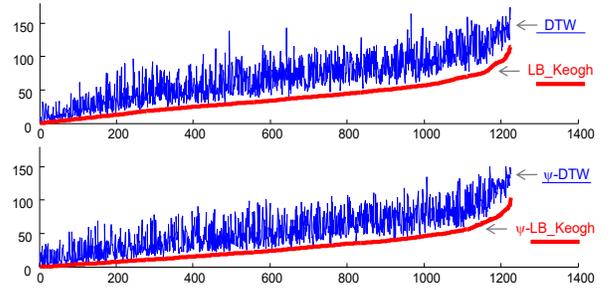


Figure 22. DTW (top) ψ -DTW (bottom) with the respective lower bound values, sorted by the LB

To further demonstrate the tightness of ψ -LB_Keogh, we compared the tightness of ψ -LB_Keogh with that of LB_Keogh for all the study cases in Section 6.2. We quantified the tightness of the LBs by dividing them by the corresponding DTW distances. In this experiment, we set the warping window as 10% of the time series length. The relaxation factor takes the same value. Table 2 shows the results obtained in the training set with the shortest time series used in each study case.

Table 2. Tightness of LB_Keogh and ψ -LB_Keogh

Dataset	Tightness of LB_Keogh	Tightness of ψ -LB_Keogh
AUSLAN	0.522	0.484
Human Activity	0.173	0.152
Motor Current	0.259	0.292
Palm Graffiti Digits	0.549	0.490
Sony Robot Activity	0.120	0.110
Sony Robot Surface	0.174	0.151

From these results, we can note that the tightness of both methods is similar. In fact, ψ -LB_Keogh is even tighter than LB_Keogh in one of the experimented datasets. This indicates that endpoint constraint relaxation does not impair the tightness of ψ -LB_Keogh.

8. Conclusion

In this paper, we proposed a modification of the endpoint constraint of DTW to make it suffix- and prefix-invariant. In addition to be simple and intuitive, our method is quite effective. Experimental results show that our method outperforms the classic DTW-by a large margin in various datasets that contain spurious endpoints. In addition, we demonstrated that the distance obtained by our method can be tightly lower bounded by a slight modification of the current lower bounds of DTW, which indicates that our modified DTW is tractable for large datasets.

Though the results are very promising, we believe that there is room for improvement. Specifically, we plan to investigate alternative ways to learn the relaxation factor parameter, especially when faced with small amounts of training data.

Finally, for the sake of clarity and brevity in this work we only discussed the application of our algorithm to classification. However, it can also be applied to a large variety of tasks, such as clustering, motif discovery, outlier detection, etc. We leave those explorations for future work.

9. REFERENCES

- [1] Alon, J., Athitsos, V., Yuan, Q., and Sclaroff, S. 2009. A unified framework for gesture recognition and spatiotemporal gesture segmentation. *IEEE T. Pattern. Anal.* 31, 9, 1685-1699.

- [2] Anguita, D., Ghio, A., Oneto, L., Parra, X. and Reyes-Ortiz, J. L. 2012. Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine. In *Proceedings of the International Workshop of Ambient Assisted Living*. IWALL'12 Springer Berlin Heidelberg, 216-223.
- [3] Batista, G. E. A. P. A., Keogh, E. J., Tataw, O. M., and Souza, V. M. 2014. CID: an efficient complexity-invariant distance for time series. *Data Min. Knowl. Discov.*, 28, 3, 634-669.
- [4] Chen, Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A. and Batista, G. E. A. P. A. 2015. The UCR Time Series Classification Archive. URL www.cs.ucr.edu/~eamonn/time_series_data/
- [5] Debray, A. and Wu, R. 2013. Astronomical implications of Machine Learning.
- [6] Demerdash, N.A. and Bangura, J.F., 1999. Characterization of induction motors in adjustable-speed drives using a time-stepping coupled finite-element state-space method including experimental validation. *IEEE T. Ind. Appl.* 35, 4, 790-802.
- [7] Haltsonen, S. 1984. An endpoint relaxation method for dynamic time warping algorithms. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*. ICASSP'84. IEEE, 360-363.
- [8] Hu, B., Chen, Y., and Keogh, E. 2013. Time Series Classification under More Realistic Assumptions. In *Proceedings of the SIAM International Conference on Data Mining*. SDM'13. SIAM, 578-586.
- [9] Kadous, M. W. 2002. *Temporal Classification: Extending the Classification Paradigm to Multivariate Time Series*. Doctoral Thesis. University of New South Wales.
- [10] Keogh, E. and Ratanamahatana, C. A. 2005. Exact indexing of dynamic time warping. *Knowl. Inf. Syst.*, 7, 3, 358-386.
- [11] Kiyohara, T., Orihara, R., Sei, Y., Tahara, Y., and Ohsuga, A. 2015. Activity Recognition for Dogs Based on Time-series Data Analysis. In *Proceedings of the International Conference on Agents and Artificial Intelligence*. ICAART'15. Springer, 163-184.
- [12] Müller, M. 2007. Dynamic time warping. *Information retrieval for music and motion*. Springer, 69-84.
- [13] Myers, C. S. 1980. *A comparative study of several dynamic time warping algorithms for speech recognition*. Doctoral Thesis. Massachusetts Institute of Technology.
- [14] Povinelli, R. J., Johnson, M. T., Lindgren, A. C., and Ye, J. 2004. Time series classification using Gaussian mixture models of reconstructed phase spaces. *IEEE T. Knowl. Data. En.*, 16, 6, 779-783.
- [15] Rabiner, L. R. and Juang, B. 1993. Pattern-comparison techniques. *Fundamentals of speech recognition*. Prentice-Hall, Inc., 141-241.
- [16] Rakthanmanon, T., Campana, B., Mueen, A., Batista, G., Westover, B., Zhu, Q., Zakaria, J., and Keogh, E. 2012. Searching and mining trillions of time series subsequences under dynamic time warping. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD'12. ACM, 262-270.
- [17] Ratanamahatana, C. A. and Keogh, E. 2005. Three Myths about Dynamic Time Warping Data Mining. In *Proceedings of the SIAM International Conference on Data Mining*. SDM'05. SIAM, 506-510.
- [18] Rebbapragada, U., Protopapas, P., Brodley, C., and Alcock, C. 2009. Finding anomalous periodic time series. *Mach. Learn.*, 74, 3, 281-313.
- [19] Saini, I., Singh, D., and Khosla, A. 2013. QRS detection using K-Nearest neighbor algorithm (KNN) and evaluation on standard ECG databases. *Journal of Advanced Research*, 4, 331-344
- [20] Sakoe, H. and Chiba, S. 1978. Dynamic Programming Algorithm Optimization for Spoken Word Recognition. *IEEE T. Acoust. Speech*, 26, 1, 43-49.
- [21] Silva, D. F., Batista, G. E. A. P. A., Keogh, E. 2016. Complementary website for this work. URL <https://sites.google.com/site/relaxedboundarydtw/>
- [22] Suhrbier, A., Heringer, R., Walther, T., Malberg, H., and Wessel, N. 2006. Comparison of three methods for beat-to-beat-interval extraction from continuous blood pressure and electrocardiogram with respect to heart rate variability analysis. *Biomed. Tech.*, 51, 2, 70-76.
- [23] Swan, M. 2012. Sensor Mania! The Internet of Things, Wearable Computing, Objective Metrics, and the Quantified Self 2.0. *J. Sens. Actuator Netw.*, 1, 3, 217-253
- [24] Taborri, J., Palermo, E., Rossi, S. and Cappa, P. 2016. Gait Partitioning Methods: A Systematic Review. *Sensors*, 16,1, 66.
- [25] Tormene, P., Giorgino, T., Quaglini, S., and Stefanelli, M. 2009. Matching incomplete time series with dynamic time warping: an algorithm and an application to post-stroke rehabilitation. *Artif. Intell. Med.*, 45, 1, 11-34.
- [26] Ueno, K., Xi, X., Keogh, E., and Lee, D.-J. 2006. Anytime Classification Using the Nearest Neighbor Algorithm with Applications to Stream Mining. In *Proceedings of the IEEE International Conference on Data Mining*. ICDM'06. IEEE, 623-632.
- [27] Vail, D. and Veloso, M. 2004. Learning from accelerometer data on a legged robot. In *Proceedings of the IFAC/EURON Symposium on Intelligent Autonomous Vehicles*.
- [28] Wang, X., Mueen, A., Ding, H., Trajcevski, G., Scheuermann, P., and Keogh, E. 2013. Experimental comparison of representation methods and distance measures for time series data. *Data Min. Knowl. Discov.*, 26, 2, 275-309.
- [29] Xu, R., and Wunsch, I. 2005. Survey of clustering algorithms. *IEEE T. Neural Networ.*, 16, 3, 645-678.