TRAINABLE TIME WARPING: ALIGNING TIME-SERIES IN THE CONTINUOUS-TIME DOMAIN

Soheil Khorram, Melvin G McInnis, Emily Mower Provost

University of Michigan, Ann Arbor, USA

ABSTRACT

DTW calculates the similarity or alignment between two signals, subject to temporal warping. However, its computational complexity grows exponentially with the number of time-series. Although there have been algorithms developed that are linear in the number of time-series, they are generally quadratic in time-series length. The exception is generalized time warping (GTW), which has linear computational cost. Yet, it can only identify simple time warping functions. There is a need for a new fast, high-quality multi-sequence alignment algorithm. We introduce *trainable time warping (TTW)*, whose complexity is linear in both the number and the length of time-series. TTW performs alignment in the continuous-time domain using a sinc convolutional kernel and a gradient-based optimization technique. We compare TTW and GTW on 85 UCR datasets in time-series averaging and classification. TTW outperforms GTW on 67.1% of the datasets for the averaging tasks, and 61.2% of the datasets for the classification tasks.

Index Terms— dynamic time warping, DTW, trainable time warping, TTW, shifted sinc kernel.

1. INTRODUCTION

Time-series alignment is an important question in many different applications, including bioinformatics [1], computer vision [2], speech recognition [3], speech synthesis [4, 5] and human action recognition [6]. The most commonly used method to perform this alignment is dynamic time warping (DTW), which uses the Bellman's dynamic programming technique to search for the optimal time warping [7, 8]. However, it is difficult to apply DTW over large sets of time-series because the computational complexity[1] of DTW grows exponentially with the number of time-series to be aligned [9, 10].

In this paper, we introduce a new time warping algorithm, called *trainable time warping (TTW)* with linear computational complexity in both N and T, where N and T are the number and the length of time-series, respectively[2]. TTW first aligns the input sequences and then takes an average over the synchronized sequences to calculate the centroid signal. We propose a sinc convolutional kernel that is able to warp signals in a nonlinear (elastic) manner. TTW uses this convolutional kernel along with a gradient-based optimization technique to synchronize multiple time-series.

DTW extensions for multiple time-series include a progressive DTW-based method, called NonLinear Alignment and Averaging Filters (NLAAF) [11]. In each iteration of NLAAF, time-series are randomly grouped into a set of pairs and then each pair is fused using the DTW averaging. This procedure is repeated until the centroid time-series is achieved. However, although NLAAF is practical for averaging multiple sequences, it has two problems: (1) ordering - the centroid is sensitive to the order of pairing time-series and (2) error propagation - there is no feedback in the algorithm to reduce the errors that occur at early steps.

Many papers presented methods to address these two weaknesses. Niennattrakul et al. [12], addressed the first problem, introducing a method that uses hierarchical clustering to automatically create a reasonable order for pairing time-series. Petitjean et al. [13, 14], addressed both, introducing an averaging method, DTW Barycenter Averaging (DBA), which iterates over two steps: (1) aligning: DBA computes a DTW mapping between the current centroid and each input time-series and (2) updating: DBA uses the mappings obtained from the previous step to align the time-series and update the centroid sequence. However, DBA is sensitive to the quality of the initial average signal. When the initial average is considerably different from the best solution, DBA is highly likely to converge to a weak local minimum [15]. Extensions have been proposed to improve the performance of the DBA algorithm (e.g., soft-DTW [15], stochastic subgradient (SSG) [8] and constraint DTW [7]). However, all have the computational complexity of $\mathcal{O}(NT^2)$. The complexity is quadratic in T, which is problematic in many applications that must process long-term time-series [16].

To the best of our knowledge, generalized time warping (GTW) [17, 18] is the only algorithm that is able to align multiple signals with linear complexity in T. GTW approximates the optimal temporal warping by linearly combining a fixed set of monotonic basis functions. They introduced a Gauss-Newton-based procedure to learn the weights of the basis functions. However, in the cases where the temporal relationship between the time-series is complex, GTW requires a large number of complex basis functions to be effective; defining these basis functions is very difficult.

We evaluate the efficacy of TTW on 85 datasets of the UCR archive [19] for two applications: DTW averaging and time-series classification. We implement GTW as the baseline system, because GTW has similar computational complexity. The results show that TTW is able to outperform GTW on most of the datasets.

2. PROBLEM SETUP

Given a set of N time-series of length T, $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ where $\boldsymbol{x}_n = [\, x_n[1], \ldots, x_n[\mathrm{T}] \,]$, the goal of DTW averaging is to estimate an optimal average signal $\hat{\boldsymbol{y}}$ that minimizes the following cost:

$$\hat{\boldsymbol{y}} = \arg\min_{\boldsymbol{y}} \sum_{n=1}^{N} \mathcal{D}_{dtw}(\boldsymbol{x}_n, \boldsymbol{y}), \qquad (1)$$

where $\mathcal{D}_{dtw}(\boldsymbol{x}_n, \boldsymbol{y})$ is the DTW distance between $\boldsymbol{x}_n$ and $\boldsymbol{y}$ [13, 20, 21]. In this paper, we approximate $\hat{\boldsymbol{y}}$ by estimating a set of synchronized signals, $\widetilde{\boldsymbol{X}} = \{\widetilde{\boldsymbol{x}}_1, \ldots, \widetilde{\boldsymbol{x}}_N\}$, where $\widetilde{\boldsymbol{x}}_n = [\widetilde{x}_n[1], \ldots, \widetilde{x}_n[\mathrm{T}]\,]$, and then taking average of this set: $\hat{\boldsymbol{y}} = \frac{1}{N}\sum_{n=1}^{N} \widetilde{\boldsymbol{x}}_n$.

[1]Computational complexity refers to both time and space complexities throughout this paper.

[2]For the sake of simplicity in notations, we assume time-series have the same length. Extending TTW to variable-length signals is straightforward. One method is to interpolate the signals to the maximum length of them.

3. METHODS

In this section, we first describe how the shifted sinc kernel can be used to introduce a learnable uniform shift to a time-series. We then extend to a more general non-linear time warping case. Finally, we introduce the TTW algorithm.

3.1. Shifted Sinc Kernel

We first consider a simple case where the time alignment between the signals can be performed by applying a uniform delay τ_n (i.e., $\widetilde{x}_n[t] = x_n[t-\tau_n]$). We can implement this alignment by convolving the input signal $\boldsymbol{x_n}$ with a shifted dirac-delta, $\delta[t-\tau_n]$, kernel:

$$\forall n \in \{1, ..., N\}, \quad \widetilde{x}_n[t] = x_n[t] * \delta[t-\tau_n], \tag{2}$$

where $*$ is the convolution operation.

In order to perform temporal alignment, we must estimate the best value for the shift parameter τ_n, which can be done through a search algorithm [22]. However, search algorithms are inefficient when there are multiple input signals, because the search space grows exponentially with the number of the signals (similar to the standard DTW). Gradient-based optimization techniques are commonly used to deal with very large search spaces. However, τ_n is not learnable in this way because: (1) τ_n is an integer, while gradient-based optimization strategies have been designed for a continuous search space and (2) $\delta(t)$ is not differentiable whenever $t = 0$.

As a solution, we propose to transfer the time-series to the continuous-time domain and perform the temporal alignment in that domain. In this approach, τ_n is applied to a continuous time-series and can be any real number. To implement this idea, three main steps are needed: (1) interpolation: transferring the input signals from discrete to continuous-time domains; (2) shifting: shifting the signals in the continuous-time domain; (3) sampling: returning the signals to the original discrete-time domain.

Interpolation – In the first step, we use sinc interpolation to transfer to the continuous-time domain. Ideal sinc interpolation preserves all frequency components of the input signals [23]. The relationship between discrete and continuous time signals in the sinc interpolation technique is[3]:

$$\bar{x}_n(t) = \sum_{m=1}^{T} x_n[m] sinc\left(\tfrac{t}{T_s} - m\right), \tag{3}$$

where $x_n[m]$ is the m-th sample of the n-th signal, $\bar{x}_n(t)$ is an interpolation of the n-th signal at time t, T_s is the sampling rate and the $sinc$ function is defined as:

$$sinc(t) = \begin{cases} \frac{sin(\pi t)}{(\pi t)} & t \neq 0 \\ 1 & t = 0 \end{cases}. \tag{4}$$

Shifting – In the second step, we temporally align the continuous-time signals obtained from the previous step. In the simplest case, the alignment is performed by applying a fixed shift, i.e.,

$$\widetilde{x}_n(t) = \bar{x}_n(t - \tau_n T_s) = \sum_{m=1}^{T} x_n[m] sinc\left(\tfrac{t}{T_s} - \tau_n - m\right). \tag{5}$$

In this equation, the size of the shift is τ_n samples or $\tau_n T_s$ seconds. We can simplify Equation (5) using properties of the sinc function:

$$\begin{aligned} \widetilde{x}_n(t) &= \left(\textstyle\sum_{m=1}^{T} x_n[m] sinc(\tfrac{t}{T_s} - m)\right) * sinc\left(\tfrac{t}{T_s} - \tau_n\right) \\ \widetilde{x}_n(t) &= \bar{x}_n(t) * sinc\left(\tfrac{t}{T_s} - \tau_n\right). \end{aligned} \tag{6}$$

[3]In this paper, we use [.] for descrete-time signals and (.) for continuous-time signals. For example, $f[m]$ is the m-th sample of a descrete-time signal $\boldsymbol{f}$, and $\bar{f}(t)$ is the value of a continuous-time signal, $\bar{\boldsymbol{f}}$, at time t.

This equation shows that we can use a shifted low-pass (sinc) filter to introduce a fixed shift to the interpolated signals.

Sampling – Finally, in the third step, we transfer the shifted signals to the original discrete-time domain using uniform sampling. Uniform sampling is performed by replacing t with tT_s in Equation (6) which results in:

$$\widetilde{x}_n[t] = x_n[t] * sinc(t - \tau_n) = \sum_{m=1}^{T} x_n[m] sinc(t - \tau_n - m). \tag{7}$$

This equation is the final expression for the shifted sinc kernel. In the case of integer τ_n, the kernel (Equation (7)) is equal to the dirac-delta function in the discrete-time domain (Equation (2)), and therefore the sinc kernel just introduces a fixed shift. In other cases, the kernel interpolates the signal to continuous-time domain, introduces a shift there and returns the signal to the discrete-time domain.

Equation (7) shows that introducing a shift in the continuous-time domain is equivalent to convolving the signals with a shifted sinc kernel in the discrete-time domain. Therefore, *to align the signals with a simple shift, we just need to convolve them with the shifted sinc kernels (Equation (7)) and learn the shift parameters* τ_n. The sinc kernel gives us the ability to introduce real-valued shifts to discrete-time signals. It is also a differentiable function and can therefore be used in gradient-based optimization techniques.

3.2. Non-linear Temporal Warping Using Sinc Kernel

The shifted sinc kernel is able to apply a fixed shift; however, a fixed shift is generally not sufficient for aligning time-series. In DTW-based algorithms, alignment is performed by applying an *elastic (non-linear)* transform to the time axis of the signals. This transform is called time warping or time alignment function. Inspired by these algorithms, we extend our linear transform ($t \rightarrow t - \tau_n$), to a more general nonlinear transform ($t \rightarrow \tau_n[t]$) by replacing the term $t - \tau_n$ with $\tau_n[t]$ in Equation (7):

$$\widetilde{x}_n[t] = \sum_{m=1}^{T} x_n[m] sinc(\tau_n[t] - m). \tag{8}$$

We represent the set of all warping functions with $\boldsymbol{\mathcal{T}} = \{\boldsymbol{\tau}_1, ..., \boldsymbol{\tau}_N\}$, where $\boldsymbol{\tau}_n = [\tau_n[1], ..., \tau_n[\mathrm{T}]]$ is the warping function that we use to align the n-th signal. Equation (8) states that the t-th sample of the synchronized signal, $\widetilde{x}_n[t]$, will be equal to an interpolation of the original signal at time $\tau_n[t]$ (i.e., $\widetilde{x}_n[t] = \bar{x}_n(\tau_n[t])$). Note that we use ideal sinc interpolation to estimate the value of the original signal at time $\tau_n[t]$.

However, calculating the synchronized signals through Equation (8) is computational expensive. For a signal of length T, it needs $\mathcal{O}(T^2)$ arithmetic operations, which is problematic when the algorithm processes long-term signals. In order to overcome this problem, we approximate the ideal sinc with a windowed sinc function. We know that the sinc function is a sine wave that decays by increasing the time; more than 99% of its power lies in the range of $[-10, 10]$. Therefore, we assume $sinc(t) = 0$ whenever $|t| > 10$. Using this assumption we can approximate Equation (8) by:

$$\widetilde{x}_n[t] = \sum_{m=\lfloor \tau_n[t] \rfloor - 10}^{\lfloor \tau_n[t] \rfloor + 10} x_n[m] sinc(\tau_n[t] - m), \tag{9}$$

where $\lfloor . \rfloor$ denotes the floor operator. In other words, we use a truncated sinc (windowed sinc with a rectangular window) to reduce the computational complexity of Equation (9) from $\mathcal{O}(T^2)$ to $\mathcal{O}(T)$.

3.3. DTW Averaging Using Sinc Kernel

To accomplish DTW averaging, we first synchronize the given time-series through an optimal set of time warping functions, $\boldsymbol{\mathcal{T}}_{\boldsymbol{opt}}$; we then take average of the synchronized sequences. We estimate the optimal set of time warping functions by numerically solving the following optimization problem using the Adam algorithm [24]:

$$\boldsymbol{\mathcal{T}}_{\boldsymbol{opt}} = \arg\min_{\boldsymbol{\mathcal{T}}} \mathcal{D}(\widetilde{\boldsymbol{X}}) \quad S.T. \quad \textbf{\textit{DTW Constraints}}, \tag{10}$$

$$\mathcal{D}(\widetilde{\boldsymbol{X}}) = \frac{1}{NT}\sum_{n=1}^{N}\sum_{t=1}^{T}(\widetilde{x}_n[t]-y[t])^2, \;\; y[t] = \frac{1}{N}\sum_{n=1}^{N}\widetilde{x}_n[t],$$

where $\mathcal{D}(\widetilde{\boldsymbol{X}})$ is the within-group mean square error (MSE) of the synchronized signals, $\widetilde{\boldsymbol{X}}$, and $y[t]$ is the t-th sample of the average signal. According to this Equation, we find the optimal alignments that satisfy the DTW conditions and minimizes the within-group distance of the synchronized signals. The DTW conditions guarantee that the synchronized sequences preserve the shape of the original time-series. The next section will explain these conditions in detail.

3.4. Time Warping Constraints

In this section, we ensure that our time warping algorithm satisfies the DTW constraints including *continuity*, *monotonicity* and *boundary* conditions [17]. This mitigates concerns regarding unreasonable choices of $\boldsymbol{\mathcal{T}}$, which could otherwise change the shape of the input time-series in an unsatisfactory manner.

Continuity: Sudden changes in the warping function distort the overall shape of the input time-series [17, 7]. In order to avoid sudden changes, we enforce a smoothness over $\tau_n[t]$. There are different methods for generating a smooth sequence. In this paper, we propose to estimate the first K coefficients of the *discrete sine transform (DST)* of the warping functions instead of estimating all samples [25]. This guarantees the smoothness of the alignment by eliminating high frequency components. In other words, we assume that the warping functions can be written as:

$$\tau_n[t] = t + \sum_{k=1}^{K} a_k^n sin\left(\frac{\pi k(t-1)}{(T-1)}\right), \tag{11}$$

where $\boldsymbol{a}_n = [a_1^n, ..., a_K^n]$ is the DST coefficients of $\tau_n[t]$. In the alignment process, we find these coefficients instead of $\tau_n[t]$. The parameter K (number of sine components) controls the smoothness of the warping function such that with a smaller value of K we generate a smoother warping function. By increasing the value of K, we can capture higher-frequency variations of the input signals. However, our experiments show that TTW with a higher value of K is more likely to converge to a weak local minimum. In addition, The DST expressed by Equation (11) significantly reduces the number of the learning parameters, which enables us to leverage more complex (e.g., quasi-Newton) optimization algorithms.

Monotonicity: Time warping algorithms must preserve the order of the samples. This constraint guarantees that the time warping algorithm provides consistent temporal changes between input and output sequences. The warping function must be monotonically increasing to satisfy this constraint [17] (i.e., for all $t \in \{2, ..., T\}$, $\Delta\tau_n[t] \geq 0$). In the TTW algorithm, we enforce this constraint in our optimization process; after each iteration (update) of the Adam algorithm, we traverse our warping functions from beginning to end; if a sample does not follow the monotonicity condition, we set the sample to its previous sample; more precisely, for all $t \in \{2, ..., T\}$ and $\Delta\tau_n[t] < 0$, we set $\tau_n[t]$ to $\tau_n[t-1]$.

Boundary conditions: $\tau_n[1] = 1$ and $\tau_n[T] = T$. This constraint (along with the other constraints) guarantees that the aligned signal contains all parts of the original signal [26]. Our algorithm is guaranteed to satisfy this constraint because the DST transform expressed by Equation (11) generates 1 and T for the starting ($\tau_n[1]$) and the ending ($\tau_n[T]$) points of the warping functions.

3.5. Trainable Time Warping Algorithm

The goal is to synchronize a set of N time-series of length T, $\boldsymbol{X} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_N\}$ where $\boldsymbol{x}_n \in \mathbb{R}^T$, and fuse them into a single centroid time-series $\boldsymbol{y} \in \mathbb{R}^T$. To do so, we define K learning parameters, $\boldsymbol{a}_n = \{a_1^n, ..., a_K^n\}$, for each input signal $\boldsymbol{x}_n$ and initialize them with zero. TTW finds the optimal set of learning parameters through iterating over the following forward, backward and update steps.

Forward: takes the given signals $\boldsymbol{X}$ and current values of the learning parameters, $\boldsymbol{A} = \{\boldsymbol{a}_1, ..., \boldsymbol{a}_N\}$ as the input to compute the current average sequence. In the forward step, TTW generates $\boldsymbol{\mathcal{T}}$ using Equation (11); fixes the monotonicity problems of $\boldsymbol{\mathcal{T}}$ using the algorithm explained in Section 3.4 (Monotonicity); generates $\widetilde{\boldsymbol{X}}$ using Equation (9); and calculates the within-group distance $\mathcal{D}$ and the current mean signal $\boldsymbol{y}$ using Equation (10).

Backward: takes all intermediate signals of the forward step to generate the derivatives of the within-group distance $\mathcal{D}(\widetilde{\boldsymbol{X}})$ with respect to the learning parameters $\boldsymbol{A}$, (i.e., $\frac{\partial\mathcal{D}}{\partial\boldsymbol{A}}$). $\frac{\partial\mathcal{D}}{\partial\boldsymbol{A}}$ is an N-by-K matrix that can be calculated through the following expression:

$$\frac{\partial\mathcal{D}}{\partial\boldsymbol{A}}_{N\times K} = \frac{\partial\mathcal{D}}{\partial\boldsymbol{\mathcal{T}}}_{N\times T} \times \frac{\partial\boldsymbol{\mathcal{T}}}{\partial\boldsymbol{A}}_{T\times K}, \tag{12}$$

where $\times$ denotes matrix multiplication and $\frac{\partial\mathcal{D}}{\partial\boldsymbol{\mathcal{T}}}$ is an N-by-T matrix that contains the derivatives of the within-group distance $\mathcal{D}(\widetilde{\boldsymbol{X}})$ with respect to the warping functions; the following equation calculates the (n, t)-th element of this matrix:

$$\frac{\partial\mathcal{D}}{\partial\tau_n[t]} = \frac{2}{NT}(\widetilde{x}_n[t]-y[t])\frac{\partial\tilde{x}_n[t]}{\partial\tau_n[t]},$$
$$\frac{\partial\tilde{x}_n[t]}{\partial\tau_n[t]} = \sum_{m=\lfloor\tau_n[t]\rfloor-10}^{\lfloor\tau_n[t]\rfloor+10} x_n[m]sinc'(\tau_n[t]-m), \tag{13}$$

where $sinc'$ is the derivative of the sinc function. In addition, $\frac{\partial\boldsymbol{\mathcal{T}}}{\partial\boldsymbol{A}}$ denotes the derivatives of the warping functions with respect to the learning parameters which is a T-by-K matrix with the following (t, k)-th element:

$$\frac{\partial\tau_n[t]}{\partial a_k^n} = sin\left(\frac{\pi k(t-1)}{(T-1)}\right). \tag{14}$$

Update: We use the Adam algorithm [24] with the step size of 0.01 to update the learning parameters. The Adam algorithm requires $\frac{\partial\mathcal{D}}{\partial\boldsymbol{A}}$ obtained in the backward step and it performs $\mathcal{O}(NK)$ arithmetic operations to update the parameters, where NK is the number of learning parameters.

Computational Complexity of TTW – Calculating Equation (12) is the most computationally expensive step in the TTW algorithm. Time and space complexities needed to calculate this step is $\mathcal{O}(KNT)$. We run this step in a loop for I iterations ($I = 100$ in our experiments); therefore, time and space complexities of TTW are $\mathcal{O}(IKNT)$ and $\mathcal{O}(KNT)$ respectively. According to our experiments, a good choice for K is a number between 8 and 16 for the datasets of the UCR archive.

4. EXPERIMENTS

We use the UCR (University of California, Riverside) time-series classification archive to conduct experiments of this paper [19]. This archive contains 85 datasets from a wide variety of applications (e.g.,

Table 1: Results of the DTW averaging experiment. AVG refers to the conventional sample-by-sample averaging technique. We report two numbers in each cell (α-β)%: α is the percentage of the datasets on which the TTW system is significantly better; β is the percentage that the baseline is significantly better. K denotes the number of DST coefficients in Equation (11). Optimal K refers to the experiment in which we validate K for each dataset.

TTW	K = 4	K = 8	K = 16	K = 32	Optimal K
AVG	(65-13)%	(66-15)%	(72-19)%	(62-26)%	***(82-4)%***
GTW	(49-31)%	(53-31)%	(53-40)%	(38-55)%	***(57-25)%***

medical imaging, geology and astronomy). Each dataset is split into train and test sets and both sets contain class labels up to 60 classes. All time-series in a dataset of UCR have the same length; for different datasets this length varies from 24 to 2,709 samples.

Baseline System: We implement the GTW algorithm, proposed by Zhou et al. [17], as the baseline method for comparison. The basis functions and other parameters of the implemented GTW are consistent with Section 5.4 of their paper [17].

4.1. DTW Averaging Experiment

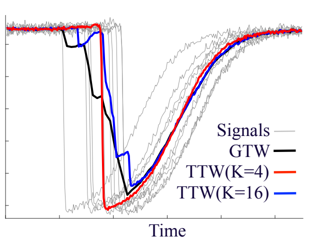
The goal of DTW averaging task is to find a time-series that have the minimum DTW distance with the given signals [27, 28]. In this section, we compare three time-series averaging techniques (i.e., TTW, GTW and AVG) for the DTW averaging task. AVG is the straightforward sample-by-sample averaging technique. We selected GTW and AVG as the baseline systems because their computational complexity is similar or better than TTW.

For each class of a dataset, we randomly select ten sets of time-series, each set contains ten signals. We estimate the DTW average signal of each set using TTW, GTW and AVG methods. We calculate the DTW distance (Equation (1)) for each generated average signal. We finally compare our proposed method with baseline systems (i.e., GTW and AVG) in each dataset by applying pair-wise t-test on the DTW distances. For each dataset, we identify if the systems are comparable (two-sided p-value > 0.05) or one of them (the one with lower mean DTW distance) is significantly better.

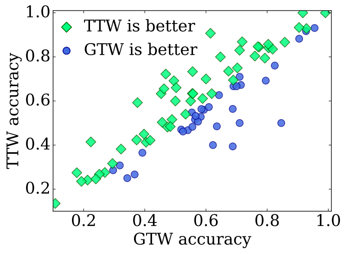
Table 1 shows the percentage of the datasets on which TTW vs. the other baseline system is significantly better. In this table, we report the results of TTW with different numbers of DST coefficients (i.e., K). According to Table 1, TTW outperforms both GTW and AVG for all values of K except 32. GTW approximates the temporal warping by combining a set of basis functions. The results show that the GTW algorithm is not flexible enough to find optimal warping functions in many datasets; however, TTW offers a more flexible time warping that is able to learn complex temporal mappings.

The performance of the TTW improves by increasing K up to 8; however, we observe diminishing gains in performance for $K > 16$. It is because TTW with a smaller K, estimates a smoother warping function that is able to smooth out local optima and provide a better optimization landscape. Figure 1(a) shows an example of aligning ten time-series using GTW, TTW(K=4) and TTW(K=16). In this example TTW(K=16) converges to a local minimum and cannot find its best solution; however, TTW(K=4) provides a better alignment and considerably outperforms the other methods.

It is important to run TTW with a reasonable value of K: TTW with a small K is not flexible enough to estimate complex warping functions and TTW with a large K is more likely to converge to a local minimum. A straightforward method to define K is to tune it on a subset of the dataset. We use the training sets to tune the value of K and we compare different methods on the test sets. We increase K exponentially from 2^0 to 2^4 and select the one that



(a)



(b)

Fig. 1: **(a)** An example of applying DTW averaging techniques on ten randomly selected signals. **(b)** Results of the nearest centroid classification experiment. Each point reports the result of one dataset; Green rectangular nodes (52 out of 85 datasets) represent the datasets on which TTW outperforms GTW.

minimizes the DTW distance of the training set of each dataset. We found that by tuning the value of K for each dataset, our system outperforms GTW on 67% of the datasets (significantly for 57% of the datasets). Table 1 (Optimal K column) summarizes the results of this experiment; the results confirm the importance of tuning K for each dataset.

4.2. Classification Experiment

In this section we investigate the performance of the TTW algorithm in a nearest centroid classification application [29]. We divide the training part of each UCR dataset into two equal parts randomly. We use the first part to find the centroid time-series using TTW and GTW methods. We validate the parameter K on the second part. We select K from five log scale values between 2^0 and 2^4. We evaluate our centroids using the UCR test partition. In the test phase, conventional DTW distance assigns each sample to a centroid.

Figure 1(b) summarizes the results of this experiment. There are 85 points in this figure; each point compares the classification accuracy of TTW and GTW on a specific dataset. The green rectangular points represent a database for which our proposed algorithm outperforms the GTW algorithm. The results show that the TTW algorithm is able to improve the GTW classification in 61.2% of the datasets.

5. CONCLUSION

This paper offers a new solution to the problem of aligning multiple time-series. We first introduce a convolutional sinc kernel that is able to apply non-linear temporal warping functions to time-series. We then use this kernel along with a gradient-based optimization technique to develop a new multiple sequence alignment algorithm: trainable time warping (TTW). The computational complexity of TTW is linear with the number and the length of the input time-series. Our experiments show that TTW provides an effective time alignment; it outperforms GTW (the previous time-warping algorithm that has similar computational complexity) on DTW averaging and nearest centroid classification tasks.

In this paper, we used a sinc filter to approximate the dirac-delta function. Dirac-delta can also be approximated by other functions such as Gaussian. Future work will explore the effect of using different types of kernels.

6. ACKNOWLEDGEMENT

This work was partially supported by the National Science Foundation (NSF CAREER 1651740), NIMH R34MH100404, the HC Prechter Bipolar Research Program, and Richard Tam Foundation.

7. REFERENCES

[1] John Aach and George M Church, "Aligning gene expression time series with time warping algorithms," *Bioinformatics*, vol. 17, no. 6, pp. 495–508, 2001.

[2] Toni M Rath and Raghavan Manmatha, "Word image matching using dynamic time warping," in *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*. IEEE, 2003, vol. 2, pp. II–II.

[3] Lindasalwa Muda, Mumtaj Begam, and Irraivan Elamvazuthi, "Voice recognition algorithms using mel frequency cepstral coefficient (mfcc) and dynamic time warping (dtw) techniques," *arXiv preprint arXiv:1003.4083*, 2010.

[4] Soheil Khorram, Fahimeh Bahmaninezhad, and Hossein Sameti, "Speech synthesis based on gaussian conditional random fields," in *International Symposium on Artificial Intelligence and Signal Processing*. Springer, 2013, pp. 183–193.

[5] Soheil Khorram, Hossein Sameti, Fahimeh Bahmaninezhad, Simon King, and Thomas Drugman, "Context-dependent acoustic modeling based on hidden maximum entropy model for statistical parametric speech synthesis," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2014, no. 1, pp. 12, 2014.

[6] Samsu Sempena, Nur Ulfa Maulidevi, and Peb Ruswono Aryan, "Human action recognition using dynamic time warping," in *Electrical Engineering and Informatics (ICEEI), 2011 International Conference on*. IEEE, 2011, pp. 1–5.

[7] Marion Morel, Catherine Achard, Richard Kulpa, and Séverine Dubuisson, "Time-series averaging using constrained dynamic time warping with tolerance," *Pattern Recognition*, vol. 74, pp. 77–89, 2018.

[8] David Schultz and Brijnesh Jain, "Nonsmooth analysis and subgradient methods for averaging in dynamic time warping spaces," *Pattern Recognition*, vol. 74, pp. 340–358, 2018.

[9] Brijnesh J Jain and David Schultz, "Asymmetric learning vector quantization for efficient nearest neighbor classification in dynamic time warping spaces," *Pattern Recognition*, vol. 76, pp. 349–366, 2018.

[10] Markus Brill, Till Fluschnik, Vincent Froese, Brijnesh Jain, Rolf Niedermeier, and David Schultz, "Exact mean computation in dynamic time warping spaces," in *Proceedings of the 2018 SIAM International Conference on Data Mining*. SIAM, 2018, pp. 540–548.

[11] Lalit Gupta, Dennis L Molfese, Ravi Tammana, and Panagiotis G Simos, "Nonlinear alignment and averaging for estimating the evoked potential," *IEEE Transactions on Biomedical Engineering*, vol. 43, no. 4, pp. 348–356, 1996.

[12] Vit Niennattrakul and Chotirat Ann Ratanamahatana, "Shape averaging under time warping," in *Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, 2009. ECTI-CON 2009. 6th International Conference on*. IEEE, 2009, vol. 2, pp. 626–629.

[13] François Petitjean, Alain Ketterlin, and Pierre Gançarski, "A global averaging method for dynamic time warping, with applications to clustering," *Pattern Recognition*, vol. 44, no. 3, pp. 678–693, 2011.

[14] Saeid Soheily-Khah, Ahlame Douzal Chouakria, and Eric Gaussier, "Progressive and iterative approaches for time series averaging.," in *AALTD@ PKDD/ECML*, 2015.

[15] Marco Cuturi and Mathieu Blondel, "Soft-dtw: a differentiable loss function for time-series," *arXiv preprint arXiv:1703.01541*, 2017.

[16] Stan Salvador and Philip Chan, "Toward accurate dynamic time warping in linear time and space," *Intelligent Data Analysis*, vol. 11, no. 5, pp. 561–580, 2007.

[17] Feng Zhou and Fernando De la Torre, "Generalized time warping for multi-modal alignment of human motion," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 1282–1289.

[18] Feng Zhou and Fernando Torre, "Canonical time warping for alignment of human behavior," in *Advances in neural information processing systems*, 2009, pp. 2286–2294.

[19] Yanping Chen, Eamonn Keogh, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, and Gustavo Batista, "The ucr time series classification archive," July 2015.

[20] Eamonn Keogh and Chotirat Ann Ratanamahatana, "Exact indexing of dynamic time warping," *Knowledge and information systems*, vol. 7, no. 3, pp. 358–386, 2005.

[21] Brijnesh J Jain and David Schultz, "Optimal warping paths are unique for almost every pair of time series," *arXiv preprint arXiv:1705.05681*, 2017.

[22] Soroosh Mariooryad and Carlos Busso, "Correcting time-continuous emotional labels by modeling the reaction lag of evaluators," *IEEE Transactions on Affective Computing*, vol. 6, no. 2, pp. 97–108, 2015.

[23] LP Yaroslavsky, "Efficient algorithm for discrete sinc interpolation," *Applied optics*, vol. 36, no. 2, pp. 460–463, 1997.

[24] Diederik Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[25] A Gupta and K Raghava Rao, "A fast recursive algorithm for the discrete sine transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 38, no. 3, pp. 553–557, 1990.

[26] Diego F Silva, Gustavo EAPA Batista, and Eamonn Keogh, "Prefix and suffix invariant dynamic time warping," in *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 2016, pp. 1209–1214.

[27] Vit Niennattrakul and Chotirat Ann Ratanamahatana, "Inaccuracies of shape averaging method using dynamic time warping for time series data," in *International conference on computational science*. Springer, 2007, pp. 513–520.

[28] François Petitjean, Germain Forestier, Geoffrey I Webb, Ann E Nicholson, Yanping Chen, and Eamonn Keogh, "Faster and more accurate classification of time series by exploiting a novel dynamic time warping averaging algorithm," *Knowledge and Information Systems*, vol. 47, no. 1, pp. 1–26, 2016.

[29] Ping Li, Jianping Gou, and Hebiao Yang, "The distance-weighted k-nearest centroid neighbor classification," *J. Intell. Inf. Hiding Multimedia Sig. Process*, vol. 8, no. 3, pp. 611–622, 2017.