# Amercing: An Intuitive and Effective Constraint for Dynamic Time Warping

Matthieu Herrmann*, Geoffrey I. Webb

*Department of Data Science and Artificial Intelligence and Monash Data Futures Institute, Monash University, Melbourne, VIC 3800, Australia*

**Abstract**

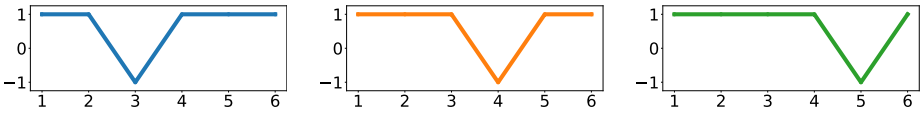
Dynamic Time Warping (DTW) is a time series distance measure that allows non-linear alignments between series. Constraints on the alignments in the form of windows and weights have been introduced because unconstrained DTW is too permissive in its alignments. However, windowing introduces a crude step function, allowing unconstrained flexibility within the window, and none beyond it. While not entailing a step function, a multiplicative weight is relative to the distances between aligned points along a warped path, rather than being a direct function of the amount of warping that is introduced. In this paper, we introduce *Amerced Dynamic Time Warping* (ADTW), a new, intuitive, DTW variant that penalizes the act of warping by a fixed additive cost. Like windowing and weighting, ADTW constrains the amount of warping. However, it avoids both abrupt discontinuities in the amount of warping allowed and the limitations of a multiplicative penalty. We formally introduce ADTW, prove some of its properties, and discuss its parameterization. We show on a simple example how it can be parameterized to achieve an intuitive outcome, and demonstrate its usefulness on a standard time series classification benchmark. We provide a demonstration application in C++ [1].

*Keywords:* Time Series, Dynamic Time Warping, Elastic Distance

*Corresponding author

*Email addresses:* matthieu.herrmann@monash.edu (Matthieu Herrmann), geoff.webb@monash.edu (Geoffrey I. Webb)

Preprint of paper published in Pattern Recognition, 2023.




(a) $S = \{1, 1, -1, 1, 1, 1\}$ (b) $T = \{1, 1, 1, -1, 1, 1\}$ (c) $U = \{1, 1, 1, 1, -1, 1\}$

Figure 1: Three time series $S$, $T$ and $U$. No current variant of DTW captures the intuition that $S$ is identical to itself, and not identical to either $T$ or $U$, and that $S$ is more similar to $T$ than it is to $U$.

## 1. Introduction

Dynamic Time Warping (DTW) is a distance measure for time series. First developed for speech recognition [2, 3], it has been adopted across a broad spectrum of applications including gesture recognition [4], signature verification [5], shape matching [6], road surface monitoring [7], neuroscience [8] medical diagnosis [9], seismic imaging [10], and vision [11].

DTW sums pairwise-distances between aligned points in two series. To allow for similar events that unfold at different rates, DTW provides elasticity in which points are aligned. However, it is well understood that DTW can be too flexible in the alignments it allows. We illustrate this with respect to three series shown in Figure 1. One possible expectation for a time series distance function "dist" is that $\text{dist}(S, S) = 0 < \text{dist}(S, T) < \text{dist}(S, U)$. However, unless a constraint is applied, $\text{DTW}(S, S) = \text{DTW}(S, T) = \text{DTW}(S, U) = 0$. While this may be appropriate for some tasks, where all that matters is that these sequences all contain five 1s and one -1, it is inappropriate for others.

When a window $w$ is introduced, $\text{DTW}_w$ constrains the alignments from warping by more than $w$ time steps. Within the window, any warping is allowed, and none beyond it. Depending on $w$, there are three possibilities for how DTW might assess our example:

1. For $w \geq 2$, $\text{DTW}_w(S, S) = \text{DTW}_w(S, T) = \text{DTW}_w(S, U) = 0$.
2. $\text{DTW}_1(S, S) = \text{DTW}_1(S, T) = 0 < \text{DTW}_1(S, U) = 8$ (see Figure 3)
3. $\text{DTW}_0(S, S) = 0 < \text{DTW}_0(S, T) = \text{DTW}_0(S, U) = 8$.

The weighted variant of DTW, WDTW, applies a multiplicative penalty to alignments that increases the more the alignment is warped. However, because the penalty is multiplicative, the perfectly matched elements in our example still lead to $\text{WDTW}(S, S) = \text{WDTW}(S, T) = \text{WDTW}(S, U) = 0$ (see Figure 4).

Neither WDTW nor DTW can be parameterized to obtain the natural expectation that $\text{dist}(S, S) = 0 < \text{dist}(S, T) < \text{dist}(S, U)$.

In this paper, we present Amerced DTW (ADTW), an intuitive and effective variant of DTW that applies a tunable additive penalty $\omega$ for warping an alignment. To align the $-1$ in $S$ with the $-1$ in $T$, it is necessary to warp the alignments by 1 step, incurring a cost of $\omega$. A further compensatory warping step is required to bring the two end points back into alignment, incurring another cost of $\omega$. Thus, the total cost of aligning the two series is $\text{ADTW}_\omega(S, T) = 2\omega$. Aligning the $-1$ in $S$ with the $-1$ in $U$ requires warping by two steps, incurring a penalty of $2\omega$, with a further $2\omega$ penalty required to realign the end points, resulting in $\text{ADTW}_\omega(S, U) = 4\omega$. Thus, ADTW can be parameterized to be in line with natural expectations for our example, or to allow the flexibility to treat the three series as equivalent if that is appropriate for an application:

1. For $0 < \omega < 4$ (see Figure 5), $\text{ADTW}_\omega(S, S) = 0 < \text{ADTW}_\omega(S, T) = 2\omega < \text{ADTW}_\omega(S, U) = \min(4\omega, 8)$. We have $\text{ADTW}_\omega(S, U) = \min(4\omega, 8)$ because the path will not be warped if the resulting cost of 8 is cheaper than 4 warping penalties.
2. For $\omega \geq 4$, $\text{ADTW}_\omega(S, S) = 0 < \text{ADTW}_\omega(S, T) = \text{ADTW}_\omega(S, U) = 8$, because the penalty for warping is greater than the cost of not warping.
3. For $\omega = 0$, $\text{ADTW}_0(S, S) = \text{ADTW}_0(S, T) = \text{ADTW}_0(S, U) = 0$, because there is no penalty for warping.

We show that this approach is not only intuitive, it often provides superior outcomes in practice.

The remainder of this paper is organised as follows. In Section 2, we review the literature related to DTW and its variants. ADTW is presented Section 3, and Section 4 discusses how to parameterize it. We then present the results of

our experiments in Section 5, and conclude in Section 6.

## 2. Background and related Work

DTW is a foundational technique for a wide range of time series data analysis tasks, including similarity search [12], regression [13], clustering [14], anomaly and outlier detection [15], motif discovery [16], forecasting [17], and subspace projection [18]. Not only is DTW widely used in many fields, it also remains an active area of research, e.g. SoftDTW [19], a differentiable version of DTW used as loss function, published in 2017.

In this paper, for ease of exposition, we only consider univariate time series, although ADTW extends directly to the multivariate case. We denote series by the capital letters $S$, $T$ and $U$. The letter $\ell$ denotes the length of the series. Subscripting (e.g. $\ell_S$) is used to disambiguate between the length of different series. The elements $S_1, S_2, \dots S_i \dots S_{\ell_S}$ with $1 \leq i \leq \ell_S$ are the elements of the series $S = (S_1, S_2, \dots S_i \dots S_{\ell_S})$, drawn from a domain $\mathbb{D}$ (e.g. real numbers).

### *2.1. Dynamic Time Warping*

Dynamic Time Warping (DTW) [2, 3] handles alignment of series with distortion and disparate lengths. An *alignment* $\mathcal{A}(S,T) = ((i_1, j_1), \dots, (i_\lambda, j_\lambda))$ between two series $S$ and $T$ is made of tuples $\mathcal{A}(S,T)_{1 \leq k \leq \lambda} = (i_k, j_k)$ representing the point-to-point alignments of $S_{i_k}$ with $T_{j_k}$. A *warping path* is an alignment $\mathcal{A}(S,T)$ with the following properties:

- Series extremities are aligned with each other, i.e.

$$\mathcal{A}(S,T)_1 = (1,1) \quad \text{and} \quad \mathcal{A}(S,T)_\lambda = (\ell_S, \ell_T)$$

- It is continuous, i.e.

$$\forall k \in \{2, \lambda\} \ (i_{k-1} \leq i_k \leq i_{k-1} + 1) \ \wedge \ (j_{k-1} \leq j_k \leq j_{k-1} + 1)$$

- It is monotonic, i.e.

$$\forall k \in \{2, \lambda\} \ \ \mathcal{A}_k \neq \mathcal{A}_{k-1}$$

Given a cost function $\gamma : \mathbb{D} \times \mathbb{D} \to \mathbb{R}$, DTW finds a warping path $\mathcal{A}(S,T)$ minimizing the cumulative sum of $\gamma(S_{i_k}, T_{j_k})$:

$$\mathrm{DTW}(S,T) = min_{\mathcal{A}(S,T)} \sum_{k=1}^{\lambda} \gamma(S_{i_k}, T_{j_k}) \tag{1}$$

In this paper, following common practice in time series classification, we use the squared L2 norm as the cost function for all distances.

$\mathrm{DTW}(S,T)$ can be computed on a 0-indexed $(\ell_S + 1) \times (\ell_T + 1)$ *cost matrix* $M_{\mathrm{DTW}(S,T)}$. A cell $M_{\mathrm{DTW}(S,T)}(i,j)$ represents the minimal cumulative cost of aligning the first $i$ points of $S$ with the first $j$ points of $T$. It follows that

$$\mathrm{DTW}(S,T) = M_{\mathrm{DTW}(S,T)}(\ell_S, \ell_T)$$

The cost matrix $M_{\mathrm{DTW}(S,T)}$ is defined by a set of recursive equations (2). The first two equations (2a) and (2b) define the border conditions. The third equation (2c) computes the cost of a cell $(i,j)$ by adding the cost of aligning $S_i$ with $T_j$, given by $\gamma(S_i, T_j)$, to the cost of its smallest predecessors. Figure 2 shows examples of DTW cost matrices.

$$M_{\mathrm{DTW}(S,T)}(0,0) = 0 \tag{2a}$$

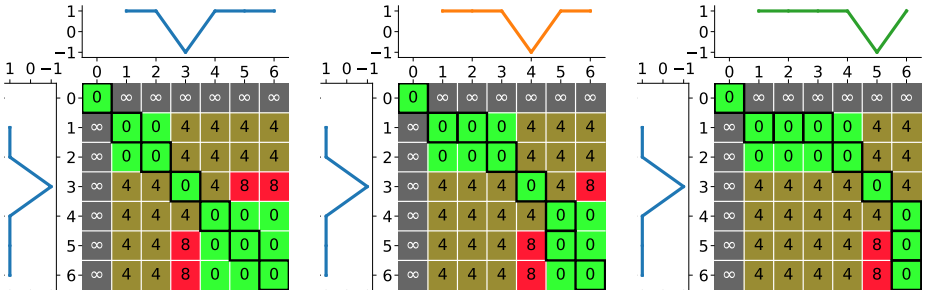
$$M_{\mathrm{DTW}(S,T)}(i,0) = M_{\mathrm{DTW}(S,T)}(0,j) = +\infty \tag{2b}$$

$$M_{\mathrm{DTW}(S,T)}(i,j) = \gamma(S_i, T_j) + \min \begin{cases} M_{\mathrm{DTW}(S,T)}(i-1, j-1) \\ M_{\mathrm{DTW}(S,T)}(i-1, j) \\ M_{\mathrm{DTW}(S,T)}(i, j-1) \end{cases} \tag{2c}$$

An efficient implementation technique [20] allows DTW and its variants, including ADTW, to be computed with an $O(\ell)$ space complexity, and that the $O(\ell^2)$ worst case time complexity can usually be avoided.

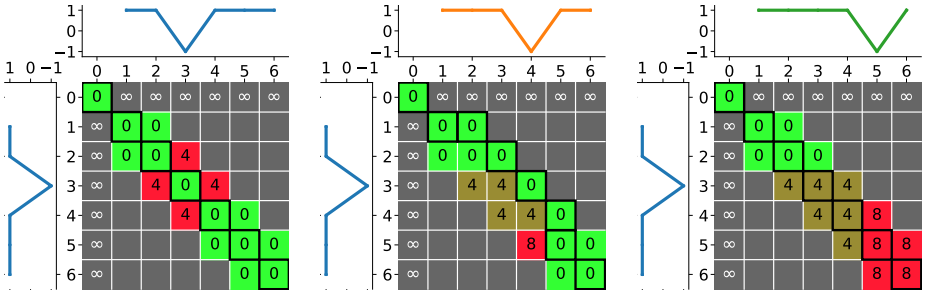
### *2.2.* DTW *parameterization*

DTW is constrained by a *warping window* $w$. This window is a parameter that restricts the warping path to a subarea of the cost matrix known as the Sakoe-Chiba band [3], defined by how far the warping path can deviate from



(a) DTWFull$(S, S) = 0$ (b) DTWFull$(S, T) = 0$ (c) DTWFull$(S, U) = 0$

Figure 2: Cost matrices and warping paths for DTWFull on $(S, S)$, $(S, T)$, and $(S, U)$.

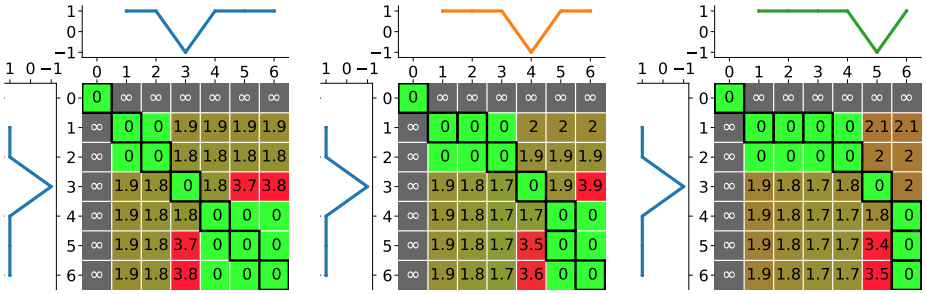


(a) $\text{DTW}_1(S, S) = 0$ (b) $\text{DTW}_1(S, T) = 0$ (c) $\text{DTW}_1(S, U) = 8$

Figure 3: Cost matrices and warping paths for $\text{DTW}_1$ on $(S, S)$, $(S, T)$, and $(S, U)$.

the diagonal. Given a cost matrix $M_{\text{DTW}_w(S,T)}$, a line index $1 \leq i \leq \ell$ and a column index $1 \leq j \leq \ell$, we have $(i, j)$ by $|i - j| \leq w$. A window of 0 forces the warping path to be on the diagonal, while a window larger than $\ell - 1$ means no constraint, which we denote as DTWFull (Figure 2). For example with $w = 1$, the warping path can only step one cell away from each side of the diagonal (Figure 3).

An effective window increases the usefulness of DTW by preventing spurious alignments. For example, nearest neighbor classification under DTW with a tuned window (NN-DTW) is significantly more accurate than nearest neighbor classification using DTWFull (NN-DTWFull, see Section 5). DTW is also usually faster to compute than DTWFull, as cells of the cost matrix beyond the window are ignored. However, the window is a parameter that must be learned.



(a) $\mathrm{WDTW}_{0.1}(S, S)$ (b) $\mathrm{WDTW}_{0.1}(S, T)$ (c) $\mathrm{WDTW}_{0.1}(S, U)$

Figure 4: Cost matrices and warping paths for $\mathrm{WDTW}_{0.1}(S, S)$, $\mathrm{WDTW}_{0.1}(S, T)$, and $\mathrm{WDTW}_{0.1}(S, U)$.

The Sakoe-Chiba band is part of the original definition of DTW and its defacto constraint, although other constraints with different subarea "shapes" exist [3, 21].

### *2.3. Weighted DTW*

Weighted Dynamic Time Warping (WDTW) [22] penalizes phase difference between two points. An alignment cost of a cell $(i, j)$ is weighted according to its distance to the diagonal, $\delta = |i - j|$. In other words, WDTW relies on a weight (3a) function to define a new cost function $\gamma'$ (3b). A large weight decreases the chances of a cell to be on an optimal path. The *weight factor* parameter $g$ controls the penalization, and usually lies within $0.01 - 0.6$ [22]. Figure 4 shows several WDTW cost matrices.

$$\mathrm{weight}(\delta) = \frac{1}{1 + \exp^{-g \times (\delta - \ell/2)}} \tag{3a}$$

$$\gamma'(S_i, T_j) = \gamma(S_i, T_j) * \mathrm{weight}(|i - j|) \tag{3b}$$

WDTW applies a penalty to every pair of aligned points that are off the diagonal. Hence, the longer the off diagonal path is, the greater the penalty. Thus, it favors many small deviations from the diagonal over fewer longer deviations.

*2.4. Squared Euclidean Distance*

One simple distance measure is to simply sum the cost of aligning successive points in the two series. This is equivalent to DTW with $w = 0$, and is only defined for same length series. When the cost function for aligning two points $S_i$ and $T_j$ is $\gamma(S_i, T_j) = (S_i - T_j)^2$, this distance measure is known as the Squared Euclidean Distance:

$$\mathrm{SQED}(S, T) = \sum_{i=1}^{\ell} (S_i - T_i)^2 \tag{4}$$

## 3. Amerced Dynamic Time Warping

Amerced Dynamic Time Warping (ADTW) provides a new, intuitive, and effective mechanism for constraining the alignments within the DTW framework. It achieves this by the introduction of a novel constraint, *amercing* — the application of a simple additive penalty $\omega$ every time an alignment is warped ($i - j$ changes). This addresses the following problems:

- DTWFull can be too flexible in its alignments;
- DTW uses a crude step function — any flexibility is allowed within the window and none beyond it;
- WDTW applies a multiplicative weight, and hence promotes large degrees of warping if they lead to low cost alignments — the penalty incurred for warping is dependent on the costs of the warped alignments. Further, as the penalty is paid for every off-diagonal alignment (where $i \neq j$), WDTW penalizes off diagonal paths for their length, rather than for the number of times they adjust the alignment [1].

[1] By *adjust the alignment* we mean having successive alignments $\mathcal{A}_k$ and $\mathcal{A}_{k+1}$ such that $i_{k+1} - i_k \neq j_{k+1} - j_k$.

### 3.1. Formal definition of ADTW

Given two times series $S$ and $T$, a cost function $\gamma : \mathbb{D} \times \mathbb{D} \to \mathbb{R}$, and an amercing penalty $\omega \in \mathbb{R}$ (see Section 4), ADTW finds a warping path $\mathcal{A}(S,T)$ minimizing the cumulative sum of amerced costs:

$$\mathrm{ADTW}_\omega(S,T) = \min_{\mathcal{A}(S,T)} \left[ \gamma(S_{i_1}, T_{j_1}) + \sum_{k=2}^{\lambda} \gamma(S_{i_k}, T_{j_k}) + 1(i_k - i_{k-1} \neq j_k - j_{k-1})\omega \right] \quad (5)$$

where $1(i_k - i_{k-1} \neq j_k - j_{k-1})\omega$ indicates that the amercing penalty $\omega$ is only applied if a step from $(i_{k-1}, j_{k-1})$ to $(i_k, j_k)$ is not an increment on both series, i.e. it is *not* the case that $i_k = i_{k-1} + 1$ and $j_k = j_{k-1} + 1$. $\mathrm{ADTW}_\omega(S.T)$ can be computed on a cost matrix $M_{\mathrm{ADTW}_\omega(S,T)}$ (6) with

$$\mathrm{ADTW}_\omega(S,T) = M_{\mathrm{ADTW}_\omega(S,T)})(\ell_S, \ell_T).$$

$M_{\mathrm{ADTW}}$ has the same border conditions as $M_{\mathrm{DTW}}$ (equations (6a) and (6b)). The other matrix cells are computed recursively, *amercing* the off diagonal alignments by the penalty $\omega$ (6c). Methods for choosing $\omega$ are discussed Section 4.

$$M_{\mathrm{ADTW}_\omega(S,T)}(0,0) = 0 \quad (6a)$$

$$M_{\mathrm{ADTW}_\omega(S,T)}(i,0) = M_{\mathrm{ADTW}_\omega(S,T)}(0,j) = +\infty \quad (6b)$$

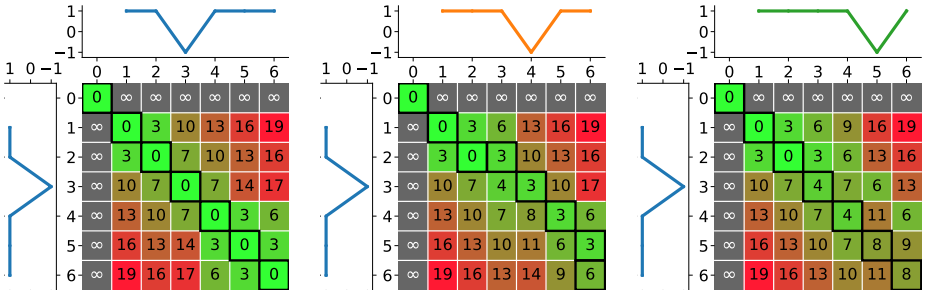
$$M_{\mathrm{ADTW}_\omega(S,T)}(i,j) = \min \begin{cases} M_{\mathrm{ADTW}_\omega(S,T)}(i-1,j-1) + \gamma(S_i, T_j) \\ M_{\mathrm{ADTW}_\omega(S,T)}(i-1,j) + \gamma(S_i, T_j) + \omega \\ M_{\mathrm{ADTW}_\omega(S,T)}(i,j-1) + \gamma(S_i, T_j) + \omega \end{cases} \quad (6c)$$

### 3.2. Properties of ADTW

$\mathrm{ADTW}_\omega(S,T)$ is monotonic with respect to $\omega$.

$$\alpha < \beta \equiv \mathrm{ADTW}_\alpha(S,T) \leq \mathrm{ADTW}_\beta(S,T) \quad (7)$$

*Proof.* For any warping path $\mathcal{A}(S,T)$ that minimizes (5) under $\omega = \beta$, the same path will result in an equivalent or lower score for $\omega = \alpha$. Hence the minimum score for a warping path under $\alpha$ cannot exceed the minimum score under $\beta$. □



(a) $\mathrm{ADTW}_3(S, S)$ (b) $\mathrm{ADTW}_3(S, T)$ (c) $\mathrm{ADTW}_3(S, U)$

Figure 5: Cost matrices and warping paths for $\mathrm{ADTW}_3$ on $(S, S)$, $(S, T)$, and $(S, U)$.

$$\mathrm{ADTW}_0(S, T) = \mathrm{DTWFull}(S, T) \tag{8}$$

*Proof.* With the amercing term $\omega$ set to zero, (5) is equivalent to (1). □

If $\ell_S = \ell_T$,

$$\mathrm{ADTW}_\infty(S, T) = \mathrm{SQED}(S, T) \tag{9}$$

*Proof.* With $\omega = \infty$, any off diagonal alignment will receive an infinite penalty. Hence the minimal cost warping path must follow the diagonal. □

When $0 \leq \omega \leq \infty$,

$$\mathrm{DTWFull}(S, T) \leq \mathrm{ADTW}_\omega(S, T) \leq \mathrm{SQED}(S, T) \tag{10}$$

*Proof.* This follows from (7), (8) and (9). □

This observation provides useful and intuitive bounds for the measure.

ADTW is symmetric with respect to the order of the arguments.

$$\mathrm{ADTW}_\omega(S, T) = \mathrm{ADTW}_\omega(T, S) \tag{11}$$

*Proof.* If we consider the cost matrix, such as illustrated in Figure 5, swapping $S$ for $T$ has the effect of flipping the matrix on the diagonal. This does not affect the cost of the minimal path. □

That a distance measure should satisfy this symmetry is intuitive, as it does not seem appropriate that the order of the arguments to a distance function should affect its value. This symmetry also holds for the other variants of DTW.

ADTW is symmetric with respect to the order in which the series are processed. With $\text{reverse}(S) = (S_{\ell_S}, S_{\ell_S-1}, \ldots, S_1)$, we have:

$$\text{ADTW}_\omega(S, T) = \text{ADTW}_\omega(\text{reverse}(S), \text{reverse}(T)) \tag{12}$$

*Proof.* For any warping path $\mathcal{A}(S, T) = ((i_1, j_1), \ldots, (i_\lambda, j_\lambda))$, there is a matching $\mathcal{A}(\text{reverse}(S), \text{reverse}(T)) = ((i_\lambda, j_\lambda), \ldots, (i_1, j_1))$. As these warping paths both contain the same alignments, the cost terms $\gamma$ will be the same. As $1(i_k - i_{k-1} \neq j_k - j_{k-1})\omega = 1(i_{k-1} - i_k \neq j_{k-1} - j_k)\omega$, the amercing penalty terms will also be identical. □

This symmetry is also intuitive, as it does not seem appropriate that the direction from which one calculates a time series distance measure should affect its value. Neither DTW nor WDTW has this symmetry when $\ell_S \neq \ell_T$. This is because both measures apply constraints relative to the diagonal, which is different depending on whether one traces it from the top left or the bottom right of the cost matrix. Unlike $\text{DTW}(S, T)$, $\text{ADTW}(S, T)$ is well defined when $\ell_S \neq \ell_T$. If $w < |\ell_S - \ell_T|$ then $\text{DTW}_w(S, T)$ is undefined.

## 4. Parameterization

As shown in Section 3.2, $\text{ADTW}_\omega(S, T)$ can be parameterized so as to range from being as flexible as DTWFull, to being as constrained as SQED. If the situation requires a large amount of warping, a *small* penalty should be used. Reciprocally, a *large* penalty helps to avoid warping. Hence, $\omega$ must be tuned for the task at hand, ideally using expert knowledge. Without the latter, one has to fallback on some automated approach. In this paper, we evaluate ADTW on a classification benchmark (Section 5). Our automated parameter selection

method has been designed to work in this context. It remains an important topic for future investigation on how to best parameterize ADTW in other scenarios.

The penalty range $0 \leq \omega \leq \infty$ is continuous, and hence cannot be exhaustively assessed unless the objective function is convex with respect to $\omega$, or there are other properties that can be exploited for efficient complete search. Instead, we use a similar method to the defacto standard for parameterizing DTW and WDTW. To this end we define a range from small to large penalties. We select from this range the one that achieves the lowest error when used as the distance measure for nearest neighbor classification, as assessed through leave-one-out cross validation (LOOCV) on the training set.

This leads to a major issue with an additive penalty such as $\omega$: its scale. A small penalty in a given context may be huge in another, as the impact of the penalty depends on the magnitude of the costs at each step along a warping path. Hence, we must find a reasonable way to determine the scale of penalties. We achieved this by defining a maximal penalty $\omega'$, and multiplying it by a ratio $0 \leq r \leq 1$ (13). Our penalty range is now $0 \leq \omega \leq \omega'$. This leads to two questions: how to define $\omega'$, and how to sample $r$.

$$\omega = \omega' * r \tag{13}$$

Let us first consider two series $S$ and $T$. As $\text{ADTW}_\omega(S,T) \leq \text{SQED}(S,T)$, taking $\omega = \text{SQED}(S,T)$ does ensure that $\text{ADTW}_\omega(S,T) = \text{SQED}(S,T)$. Indeed, taking one single *warping step* costs as much as the full diagonal. In turn, this ensures that we do not need to test any other value beyond $\omega' = \text{SQED}(S,T)$. However, we need to have a single penalty for all series from $\mathcal{D}$, so we sample random pairs of series, and set $\omega'$ to their average SQED, $\omega' = \text{mean}_{S,T\sim\mathcal{D}}\,\text{SQED}(S,T)$.

We now have to find the best $r$ — we use a detour to better explain it. Intuitively, $\omega'$ represents the cost of $\ell$ steps along the diagonal, and we can define an *average step cost* $\alpha = \frac{\omega'}{\ell}$. In turn, we have

$$\omega = \alpha \times (\ell \times r') \tag{14}$$

where $0 \leq r' \leq 1$. As $\alpha$ remains a large penalty, we need to start sampling $r'_1$ at

a value below $\frac{1}{\ell}$. Without expert knowledge, our *best guess* is to start at several orders of magnitude $m$ lower. Simplifying equation (14) back to equation (13), $r_1$ must be small enough to account for both $m$, and the magnitude of $\ell$. Then, successive $r_i$ must gradually increase toward 1. If several "best" $r_i$ are found, we take the median value.

As leave-one-out cross validation is time consuming, and to ensure a fair comparison with the methods for parameterizing DTW and WDTW, we limit the search to a hundred values [23, 24]. The formula $r_i = (\frac{i}{100})^5$ for $1 \leq i \leq 100$ fulfills our requirements, covering a range from $1E-10$ to 1 and favoring smaller penalties, which tend to be more useful than large ones, as any sufficiently large penalty confines the warping path to the diagonal.

## 5. Experiments

One difficulty with assessing the quality of alternative time series distance measures is that many of their applications lack well agreed objective criteria for such assessment. Due to its clearly defined objective evaluation measures, we evaluate ADTW against other distances on a classification benchmark, using the UCR archive [25] in its 128 dataset versions. It is important to note that in general distance-based classifiers are now outperformed by more recent specialized methods for time series classification such as HIVE-COTE [26], TS-CHIEF [27] and MultiRocket [28]. However, the success of HIVE-COTE demonstrates the power of assembling multiple forms of time series classifier and there is great potential for distance-based approaches to contribute to future such approaches. Further, there remain some time series classification tasks for which DTW-1NN and its variants outperform the state of the art. We provide one such example at the end of this section.

We retained 109 datasets from the UCR archive after filtering out those with series of variable length, containing missing data, or having only one training exemplar per class (leading to spurious results under LOOCV). We compare the distances when used in nearest neighbor classifiers. The archive provides

default train and test splits. For each dataset, the parameters $w$, $g$ and $\omega$ are learned on the training data $\mathcal{D}$. We report the accuracy obtained on the test split. Our results are available online [1].

The parameter $\omega$ is learned as described Sections 4. The warping window $w$ of DTW is learned following the defacto method for time series classification [25], i.e. applying LOOCV over the training data for all $w \in \{0, 0.01\ell, 0.02\ell, \ldots, \ell\}$ and selecting the $w$ with the lowest error. On ties, the smallest $w$ is selected as it leads to the fastest computation time. Finally, the parameter $g$ of WDTW is selected in a similar fashion, for all $g \in \{0.01, 0.02, \ldots 1.0\}$, as employed in the Elastic Ensemble [23]. Again, on ties, the smallest $g$, which leads to the fastest computation times in an EAP setting [20], is selected.

We present both the accuracy results and the impact of the distances on the running time (LOCCV train time and NN1 test time).

### *5.1. Accuracy results*

Following Demsar [29], we compare classifiers using the Wilcoxon signed-rank test. The result can be graphically presented in mean-rank diagrams (Figures 6 and 9), where classifiers not significantly different (at a 0.05 significance level, adjusted with the Holm correction for multiple testing) are connected by a horizontal bar. A rank closer to 1 indicates a more accurate classifier.

We compare ADTW against SQED, DTWFull, DTW and WDTW, both on the raw series and on their first derivative [30]. leading to the derivative version of the distances. Derivative distances are represented with the suffix "-D1" Figure 6 summarizes the outcome of this comparison. It shows that ADTW applied to the raw series attains a significantly better rank on accuracy than any other measure. Further, ADTW applied to the first derivative attains a significantly better rank than any other measure applied to the first derivative.

Mean rank diagrams are useful to summarize relative performance, but do not tell the full story — a classifier consistently outperforming another would be deemed as significantly better ranked, even if the actual gain is minimal. Accuracy scatter plots (see figures 7 and 8) allow to visualize how a classifier

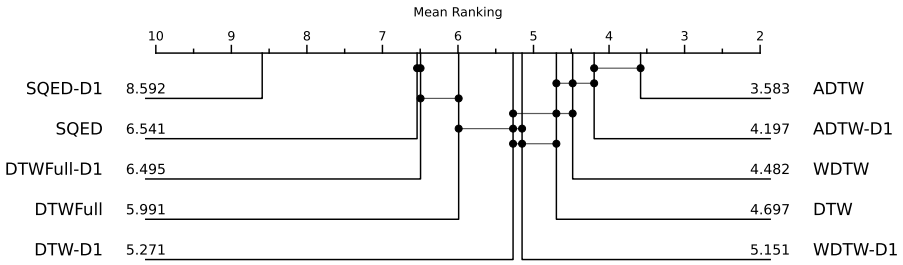
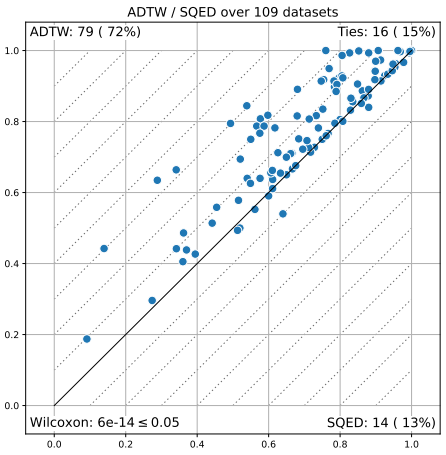



Figure 6: Test accuracy ranking of NN1 classifiers over 109 datasets from UCR128.

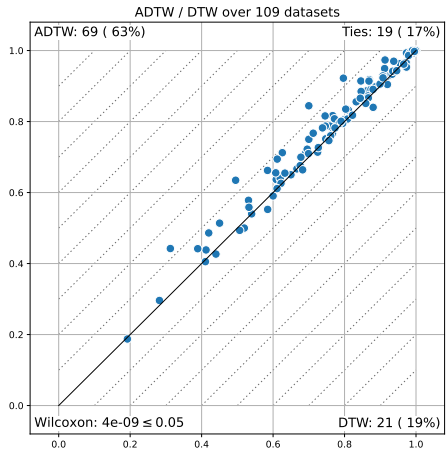
performs relatively to another one. The space is divided by a diagonal representing equal accuracy, and each dot represents a dataset. Points close to the diagonal indicates comparable accuracy between classifiers; conversely, points far away indicates different accuracies.

Figure 7 shows the accuracy scatter plot of ADTW against other distances. Points above the diagonal indicate datasets where ADTW is more accurate, and conversely below it. We also indicate the numbers of ties and *wins* per classifier, and the resulting Wilcoxon score. ADTW is almost always more accurate than SQED and DTWFull — usually substantially so, and the majority of points remain well above the diagonal for DTW and WDTW, albeit by a smaller margin. We note a point well below the diagonal in Figures 7a and 7c. This it the "Rock" dataset, for which our parameterization process fails to select a high enough penalty. Although ADTW has the capacity to behave like SQED if appropriately parameterized, our parameterization process fails to achieve this for "Rock." Effective parameterization remains an open problem.

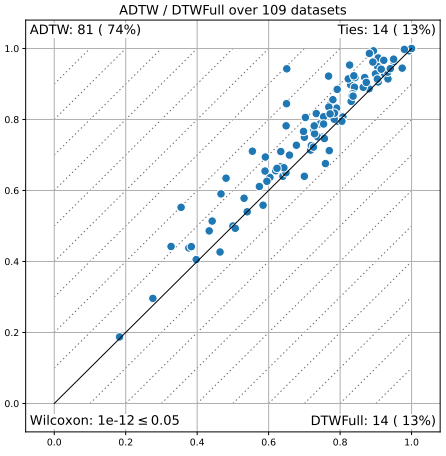
We also compare against a hypothetical BEST classifier, a classifier that uses the most accurate among all ADTW competitors. Such a classifier is not feasible to obtain in practice as it is not possible to be certain which classifier will obtain the lowest error on previously unseen test data. The accuracy scatter plots against the hypothetical BEST classifier over the raw series and the first derivative of the series are shown in Figure 8. In both cases, ADTW is on par with this hypothetical classifier, which suggests that ADTW is a good default



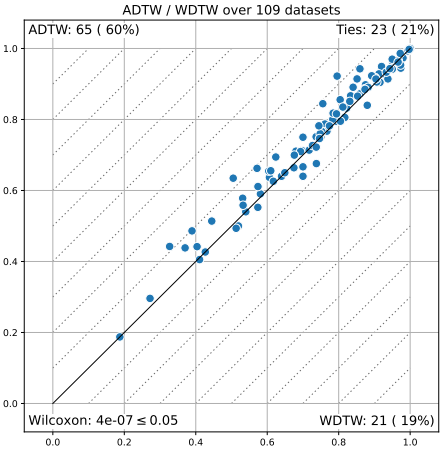

(a) ADTW vs. SQED




(b) ADTW vs. DTW

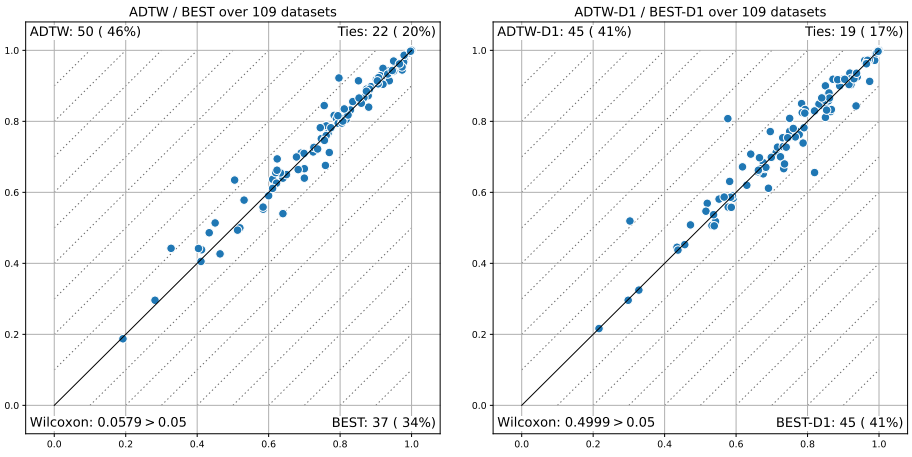



(c) ADTW vs. DTWFull




(d) ADTW vs. WDTW

Figure 7: Accuracy of NN1-ADTW vs. NN1 with other distances on 109 datasets from UCR128. Each point represents a dataset. Points above the diagonal indicate that ADTW gives better accuracy than the alternative, and reciprocally below the diagonal.




(a) ADTW vs. BEST

(b) ADTW-D1 vs. BEST-D1

Figure 8: Accuracy of NN1-ADTW vs. the best classifier per dataset among competitors NN1-{SQED, DTWFull, DTW, WDTW}, on 102 datasets from UCR128, on raw series (left) and first derivative of the series (right). A point above the diagonal indicates that ADTW gives better accuracy than any competitor, and a point below the diagonal indicates that the best alternative gives higher accuracy than ADTW.

choice of DTW variant.

Finally, we present a sensitivity analysis for the exponent used to tune $\omega$. Figure 9 shows the effect of using different values for exponent $e$ when sampling the ratio $r$ under LOOCV at train time. All exponents above 1 lead to comparable results. While $e = 4$ leads to the best results on the benchmark, we have no theoretical grounds on which to prefer it and in other applications we have examined, slightly higher values lead to slightly greater accuracy.

### *5.2. Timing results*

Training NN1 classifiers with LOOCV is notoriously slow, although recent progress have been made [31]. In this experiment, we trained ADTW, DTW and WDTW on both raw series and the first derivatives with a LOOCV technique similar to the one described in UltraFastWWSearch [31]. We used the same search table, but omitted the window validity trick for DTW, ensuring that all the distances are compared under the exact same conditions. We parallelized

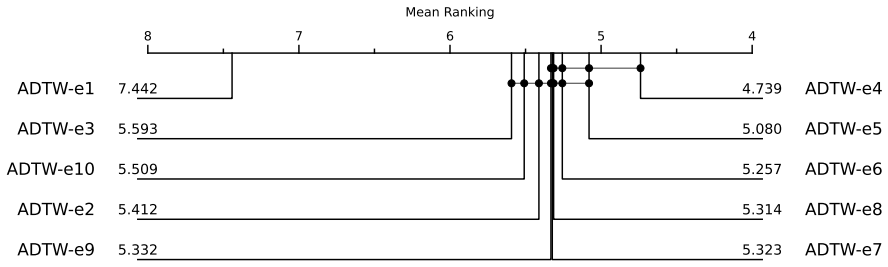



Figure 9: Test accuracy ranking of ADTW-$e$ when varying the exponent $e$ used at train time, over 109 datasets from UCR128

the process, and used an Early Abandoned and Pruned (EAP [20]) version for all the distances, rendering lower bounding techniques redundant.

Figure 10 presents the LOOCV training time of our NN1 classifiers. We used an AMD EPYC@2.2Ghz with 32 cores. The train time of ADTW is on par with DTW while WDTW is substantially slower. This is due to the multiplicative penalty of WDTW, which often leads to lower intermediate values, which in turn delay early abandoning and pruning. We refer the reader to the EAP paper [20] for a detailed discussion on early abandoning and pruning.

Finally, Figure 11 shows the testing time of our NN1 classifiers when used with the parameters selected by LOOCV. Again, the process is parallelized over 32 threads, making it quite tractable. Working with the derivative is always slower than using the raw series due to its smoothing effect (it takes longer to reach high enough cost allowing to prune or abandon a computation). ADTW is the slowest distance at test time. This is due to two main factors. One is that DTW and WDTW select among parameter values with the same LOOCV result the value leading to the fastest computation, whereas our ADTW parameterization picks the median. The other is that the ADTW penalties will increase the costs of the best path, relative to the DTWFull cost, thereby delaying opportunities for pruning subsequent paths, which are only pruned when they exceed the best so far.

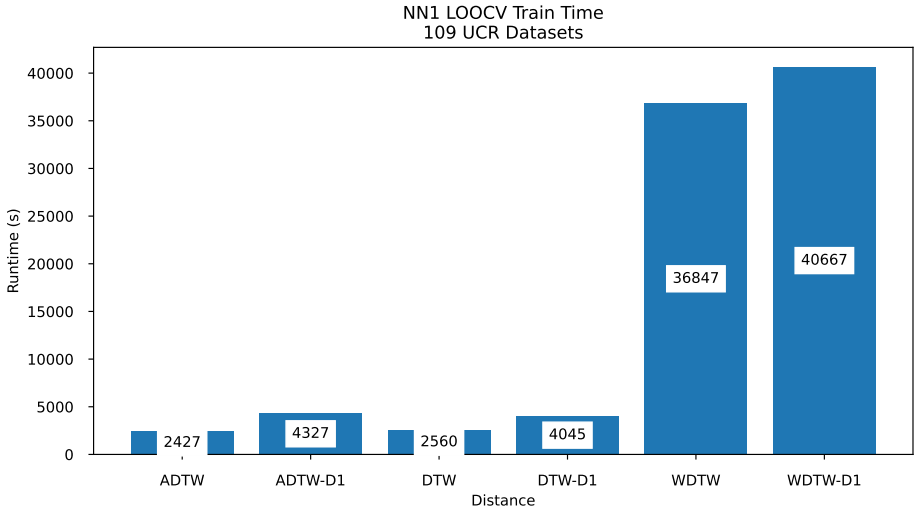



Figure 10: LOOCV Training time (in seconds) of parameterized NN1 classifiers over 109 datasets from UCR128

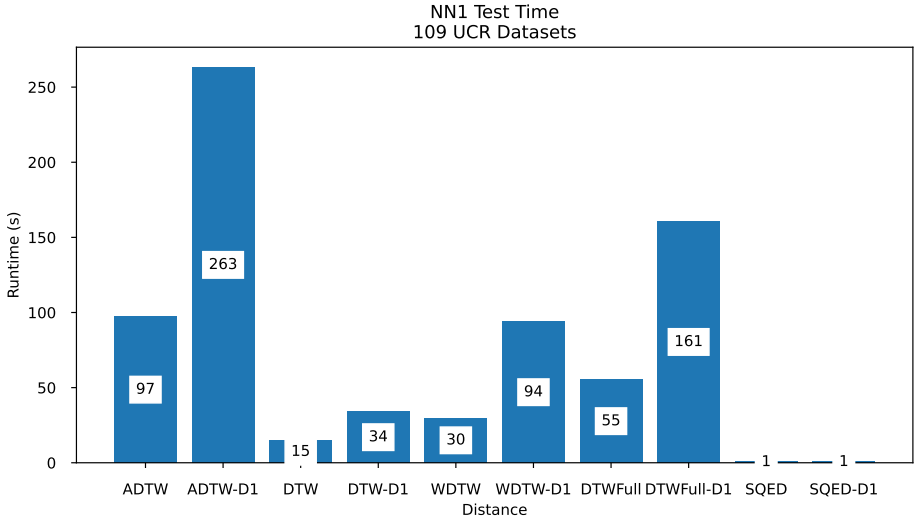



Figure 11: NN1 test time (in seconds) of parameterized NN1 classifiers over 109 datasets from UCR128 with learned parameter

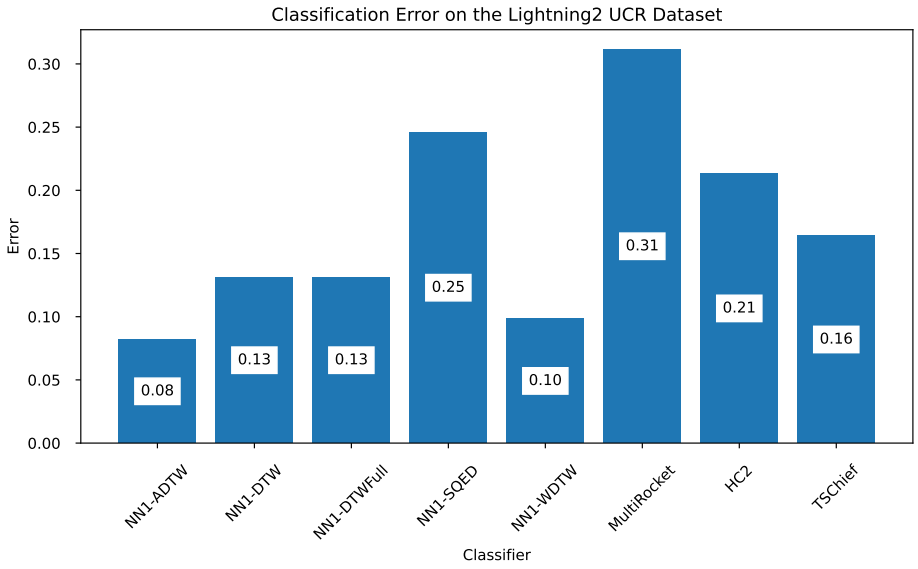



Figure 12: Classification error rates for the UCR Lightning2 dataset (lower is better).

### *5.3. An example where 1NN-DTW and its variants outperform alternatives*

In general, specialized state of the art time series classification methods significantly outperform simple NN1 classifiers, and should be used by default. However, there remain classification tasks for which 1NN-DTW and its variants strongly outperform the alternatives. The "Lightning2" dataset from the UCR archive is one example where this is the case. Figure 12 shows the error rate of several classifiers on this dataset. Among the specialized methods, TSChief, which embeds NN1 classifiers, is the better performer. Of the 1NN-DTW variants, 1NN-ADTW is most accurate.

## 6. Conclusions

DTW is a popular time series distance measure that allows flexibility in the series alignments [2, 3]. However, it can be too flexible for some applications, and two main forms of constraint have been developed to limit the alignments. Windows provide a strict limit on the distance over which points may be aligned

[3, 21]. Multiplicative weights penalize all off diagonal points proportionally to their distance from the diagonal and the cost of their alignment [22].

However, windows introduce an abrupt discontinuity. Within the window there is unconstrained flexibility and beyond it there is none. Multiplicative weights allow large warping at little cost if the aligned points have low cost. Further, they penalize the length of an off-diagonal path rather than the number of times the path deviates from the diagonal. Whats more, neither windows nor multiplicative weights are symmetric with respect to reversing the series (they do not guarantee $\text{dist}(S,T) = \text{dist}(\text{reverse}(S), \text{reverse}(T))$ when $\ell_S \neq \ell_T$).

Amerced DTW introduces a tunable additive weight $\omega$ that produces a smooth range of distance measures such that $\text{ADTW}_\omega(S,T)$ is monotonic with respect to $\omega$, $\text{ADTW}_0(S,T) = \text{DTW}(S,T)$, and $\text{ADTW}_\infty(S,T) = \text{SQED}(S,T)$. It is symmetric with respect to the order of the parameters and reversing the series, irrespective of whether the series share the same length. It has the intuitive property of penalizing the number of times a path deviates from simply aligning successive points in each series, rather than penalizing the length of the paths following such a deviation.

As a proxy for assessing the quality of the distance measurements, we assessed the accuracy obtained when ADTW is employed for nearest neighbor classification on the widely used UCR benchmark. This benchmark demonstrated that ADTW results in more accurate classification significantly more often than any other DTW variant.

This improvement in the quality of distance measurement does come at a slight computational cost. In some contexts ADTW lends itself less to state of the art methods for speeding up DTW computations, resulting in acceptable compute time. Whether the improved quality of measurement warrants the increased compute overhead should be assessed on a case by case basis.

The application of ADTW in the many other types of task to which DTW is often applied remains a productive direction for future investigation. These include similarity search [12], regression [13], clustering [14], anomaly and outlier detection [15], motif discovery [16], forecasting [17], and subspace projection

[18]. One issue that will need to be addressed in each of these domains is how best to tune the amercing penalty $\omega$, especially if a task does not have objective criteria by which utility may be judged. We hope that ADTW will prove as effective in these other applications as it has proved to be in classification. A C++ implementation of ADTW is available at [1].

**Acknowledgments**

This work was supported by the Australian Research Council award DP210100072. We would like to thank the maintainers and contributors for providing the UCR Archive, and Hassan Fawaz for the critical diagram drawing code [32]. We are also grateful to Eamonn Keogh, Chang Wei Tan, and Mahsa Salehi for insightful comments on an early draft.

**References**

[1] M. Herrmann, Nn1 adtw demonstration application and results, `https://github.com/HerrmannM/paper-2021-ADTW` (2021).

[2] H. Sakoe, S. Chiba, Recognition of continuously spoken words based on time-normalization by dynamic programming, Journal of the Acoustical Society of Japan 27 (9) (1971) 483–490.

[3] H. Sakoe, S. Chiba, Dynamic programming algorithm optimization for spoken word recognition, IEEE Transactions on Acoustics, Speech, and Signal Processing 26 (1) (1978) 43–49. `doi:10.1109/TASSP.1978.1163055`.

[4] H. Cheng, Z. Dai, Z. Liu, Y. Zhao, An image-to-class dynamic time warping approach for both 3d static and trajectory hand gesture recognition, Pattern Recognition 55 (2016) 137–147.

[5] M. Okawa, Time-series averaging and local stability-weighted dynamic time warping for online signature verification, Pattern Recognition 112 (2021) 107699.

[6] Z. Yasseen, A. Verroust-Blondet, A. Nasri, Shape matching by part alignment using extended chordal axis transform, Pattern Recognition 57 (2016) 115–135.

[7] G. Singh, D. Bansal, S. Sofat, N. Aggarwal, Smart patrolling: An efficient road surface monitoring using smartphone sensors and crowdsourcing, Pervasive and Mobile Computing 40 (2017) 71–88.

[8] Y. Cao, N. Rakhilin, P. H. Gordon, X. Shen, E. C. Kan, A real-time spike classification method based on dynamic time warping for extracellular enteric neural recording with large waveform variability, Journal of Neuroscience Methods 261 (2016) 97–109.

[9] R. Varatharajan, G. Manogaran, M. K. Priyan, R. Sundarasekar, Wearable sensor devices for early detection of alzheimer disease using dynamic time warping algorithm, Cluster Computing 21 (1) (2018) 681–690.

[10] L. Liu, Y. Li, W. He, Y. Luo, Data-domain traveltime inversion of reflected waves using segment dynamic image warping, IEEE Geoscience and Remote Sensing Letters 19 (2022) 1–5. doi:10.1109/LGRS.2021.3111688.

[11] H. Wei, L. Meng, An accurate stereo matching method based on color segments and edges, Pattern Recognition 133 (2023) 108996.

[12] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, E. Keogh, Searching and mining trillions of time series subsequences under dynamic time warping, in: Proc. 18th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2012, pp. 262–270.

[13] C. W. Tan, C. Bergmeir, F. Petitjean, G. I. Webb, Time series extrinsic regression, Data Mining and Knowledge Discovery 35 (3) (2021) 1032–1060. doi:10.1007/s10618-021-00745-9.

[14] F. Petitjean, A. Ketterlin, P. Gançarski, A global averaging method for dynamic time warping, with applications to clustering, Pattern Recognition 44 (3) (2011) 678–693.

[15] D. M. Diab, B. AsSadhan, H. Binsalleeh, S. Lambotharan, K. G. Kyriakopoulos, I. Ghafir, Anomaly detection using dynamic time warping, in: 2019 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC), IEEE, 2019, pp. 193–198.

[16] S. Alaee, R. Mercer, K. Kamgar, E. Keogh, Time series motifs discovery under DTW allows more robust discovery of conserved structure, Data Mining and Knowledge Discovery 35 (3) (2021) 863–910.

[17] K. Bandara, H. Hewamalage, Y.-H. Liu, Y. Kang, C. Bergmeir, Improving the accuracy of global forecasting models using time series data augmentation, Pattern Recognition 120 (2021) 108148.

[18] H. Deng, W. Chen, Q. Shen, A. J. Ma, P. C. Yuen, G. Feng, Invariant subspace learning for time series data based on dynamic time warping distance, Pattern Recognition 102 (2020) 107210. doi:https://doi.org/10.1016/j.patcog.2020.107210.

[19] M. Cuturi, M. Blondel, Soft-dtw: A differentiable loss function for time-series, in: Proceedings of the 34th International Conference on Machine Learning - Volume 70, ICML'17, JMLR.org, 2017, p. 894–903.

[20] M. Herrmann, G. I. Webb, Early abandoning and pruning for elastic distances including dynamic time warping, Data Mining and Knowledge Discovery 35 (6) (2021) 2577–2601. doi:10.1007/s10618-021-00782-4.
URL https://doi.org/10.1007/s10618-021-00782-4

[21] F. Itakura, Minimum prediction residual principle applied to speech recognition, IEEE Transactions on Acoustics, Speech, and Signal Processing 23 (1) (1975) 67–72. doi:10.1109/TASSP.1975.1162641.

[22] Y.-S. Jeong, M. K. Jeong, O. A. Omitaomu, Weighted dynamic time warping for time series classification, Pattern Recognition 44 (9) (2011) 2231–2240. doi:10.1016/j.patcog.2010.09.022.

[23] J. Lines, A. Bagnall, Time series classification with ensembles of elastic distance measures, Data Mining and Knowledge Discovery 29 (3) (2015) 565–592. doi:10.1007/s10618-014-0361-2.

[24] C. W. Tan, F. Petitjean, G. I. Webb, FastEE: Fast Ensembles of Elastic Distances for time series classification, Data Mining and Knowledge Discovery 34 (1) (2020) 231–272. doi:10.1007/s10618-019-00663-x.

[25] H. A. Dau, A. Bagnall, K. Kamgar, C.-C. M. Yeh, Y. Zhu, S. Gharghabi, C. A. Ratanamahatana, E. Keogh, The UCR Time Series Archive, arXiv:1810.07758 [cs, stat] (Sep. 2019). arXiv:1810.07758.

[26] M. Middlehurst, J. Large, M. Flynn, J. Lines, A. Bostrom, A. Bagnall, Hive-cote 2.0: a new meta ensemble for time series classification, Machine Learning 110 (11) (2021) 3211–3243.

[27] A. Shifaz, C. Pelletier, F. Petitjean, G. I. Webb, TS-CHIEF: A scalable and accurate forest algorithm for time series classification, Data Mining and Knowledge Discovery 34 (3) (2020) 742–775. doi:10.1007/s10618-020-00679-8.

[28] C. W. Tan, A. Dempster, C. Bergmeir, G. I. Webb, Multirocket: multiple pooling operators and transformations for fast and effective time series classification, Data Mining and Knowledge Discovery 36 (2022) 1623–1646. doi:10.1007/s10618-022-00844-1.

[29] J. Demšar, Statistical Comparisons of Classifiers over Multiple Data Sets, Journal of Machine Learning Research 7 (2006) 1–30.

[30] E. J. Keogh, M. J. Pazzani, Derivative Dynamic Time Warping, in: Proceedings of the 2001 SIAM International Conference on Data Mining, Society for Industrial and Applied Mathematics, 2001, pp. 1–11. doi:10.1137/1.9781611972719.1.

[31] C. W. Tan, M. Herrmann, G. I. Webb, Ultra fast warping window optimization for dynamic time warping, in: 2021 IEEE International Conference on

Data Mining (ICDM), 2021, pp. 589–598. doi:10.1109/ICDM51629.2021.00070.

[32] H. Ismail Fawaz, Critical difference diagram with Wilcoxon-Holm post-hoc analysis, https://github.com/hfawaz/cd-diagram (2019).