Elastic bands across the path:
A new framework and method to lower bound DTW

Chang Wei Tan
chang.tan@monash.edu

François Petitjean
francois.petitjean@monash.edu

Geoffrey I. Webb
geoff.webb@monash.edu

Faculty of IT, Monash University, Melbourne, Australia

Abstract

The Nearest Neighbour algorithm coupled with the Dynamic Time Warping similarity measure (NN-DTW) is at the core of state-of-the-art classification algorithms including Ensemble of Elastic Distances and Collection of Transformation-Based Ensemble. DTW's complexity makes NN-DTW highly computationally demanding. To combat this, lower bounds to DTW are used to minimize the number of times the expensive DTW need be computed during NN-DTW search. Effective lower bounds must balance 'time to calculate' vs 'tightness to DTW.' On the one hand, the tighter the bound the fewer the calls to the full DTW. On the other, calculating tighter bounds usually requires greater computation. Numerous lower bounds have been proposed. Different bounds provide different trade-offs between computational time and tightness. In this work, we present a new class of lower bounds that are tighter than the popular Keogh lower bound, while requiring similar computation time. Our new lower bounds take advantage of the DTW boundary condition, monotonicity and continuity constraints. In contrast to most existing bounds, they remain relatively tight even for large windows. A single parameter to these new lower bounds controls the speed-tightness trade-off. We demonstrate that these new lower bounds provide an exceptional balance between computation time and tightness for the NN-DTW time series classification task, resulting in greatly improved efficiency for NN-DTW lower bound search.

1 Introduction

Dynamic Time Warping (DTW) lower bounds play a key role in speeding up many forms of time series analytics [6, 10, 11, 17]. Several lower bounds have been proposed [6–8, 15, 19]. Each provides a different trade-off between compute time (speed) and tightness. Figure 1 illustrates this, plotting average tightness ($\mathrm{LB}(A,B)/\mathrm{DTW}(A,B)$) against the average time to compute for alternative lower bounds. As shown in Figure 2, different bounds have different relative tightness at different window sizes (the tighter the better).

In this paper, we present a family of lower bounds, all of which are of $O(L)$ time complexity and are in practice tighter than LB_KEOGH [6]. Our new lower bound is parameterized, with a tightness parameter, V controlling a useful speed-tightness trade-off. Two of these, LB_ENHANCED[1] and LB_ENHANCED[2], where $V=\{1,2\}$ respectively, have very similar compute time to LB_KEOGH, while providing tighter bounds, mean-
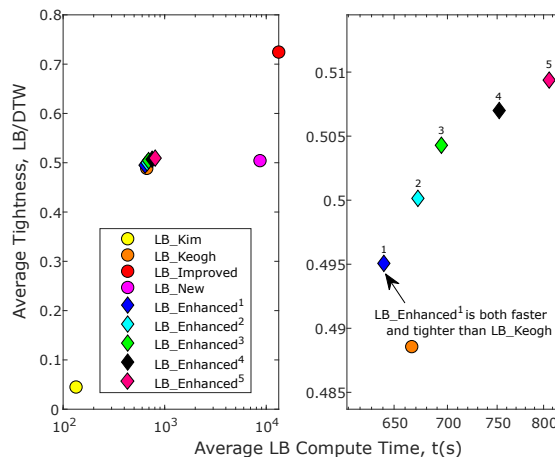

Figure 1: Tightness-Compute Time comparison of existing and our lower bounds at $W=0.1 \cdot L$ over 250,000 time series pairs with $L=256$ randomly sampled from the benchmark UCR time series archive [3]. Figure on the right shows the zoomed in plot. Our LB_ENHANCED[1] is faster and tighter than LB_KEOGH. Our other variants are even tighter at a cost of being slightly slower.
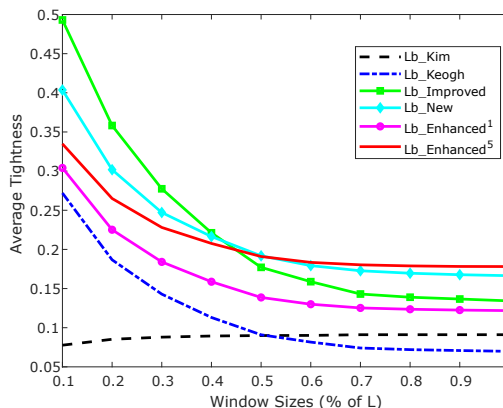

Figure 2: Tightness of different lower bounds at differing window sizes averaged across all UCR datasets [3]. Our very efficient LB_ENHANCED[5] is tighter than any alternative at large window sizes.

ing that their performance should dominate that of LB_KEOGH on any standard time series analysis task.

We focus on the application of lower bounds to DTW in Nearest Neighbor (NN-DTW) Time Series Classification (TSC). NN-DTW is in its own right a useful TSC algorithm and is a core component of the most accurate current TSC algorithms, COTE [2] and EE [9], which are ensembles of TSC algorithms. NN-DTW is thus at the core of TSC algorithms, but is extremely costly to compute [16, 17]. Given a training set with N time series and length L, a single classification with standard NN-DTW requires $O(N \cdot L^2)$ operations. Besides, NN-DTW is only competitive when used with a warping window, W learned from the training set [1]. Learning the best warping window is very time consuming as it requires the enumeration of numerous windows in the range of 0% to 100% of L and is extremely inefficient for large training sets [16].

There has been much research into speeding up NN-DTW, tackling either the N part [17] or the L^2 part of the complexity [6–8, 15, 18, 19]. A key strategy is to use lower bound search, which employs lower bounds on DTW to efficiently exclude nearest neighbor candidates without having to calculate DTW [6–8, 15, 19]. We show that different speed-tightness trade-offs from different lower bounds prove most effective at speeding up NN-DTW for different window sizes.

Of the existing widely used lower bounds, LB_KIM [7] is the fastest, with constant time complexity with respect to window size. It is the loosest of the existing standard bounds for very small W, but its relative tightness increases as window size increases. For small window sizes, LB_KEOGH [6] provides an effective trade off between speed and tightness. However, as shown in Figure 2, it is sometimes even looser than LB_KIM at large window sizes. The more computationally intensive, LB_IMPROVED [8] provides a more productive trade-off for many of the larger window sizes. LB_NEW [15] does not provide a winning trade-off for this task at any window size.

The new DTW lower bound that we propose has the same complexity $O(L)$ as LB_KEOGH. At its lowest setting, LB_ENHANCED[1], it is uniformly tighter than LB_KEOGH. It replaces two calculations of the distance of a query point (the first and last) to a target LB_KEOGH envelope with two calculations of distances between a query and a target point. This may or may not be faster, depending whether the query point falls within the envelope, in which case LB_KEOGH does not perform a distance calculation. However, due to its greater tightness, this variant always supports faster NN-DTW. At $V=\{2, 3, 4, 5\}$, our tighter LB_ENHANCED provide the greatest speed-up
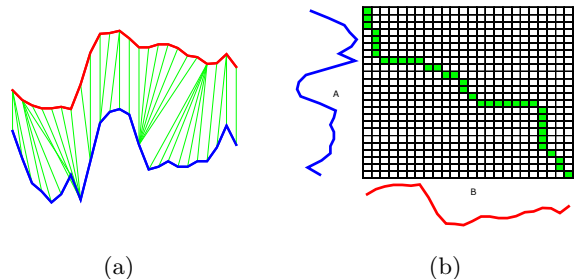

(a)   (b)

Figure 3: (a) DTW warping path for time series A and B (b) Cost matrix D with the warping path $\mathcal{A}$ (green)

out of all standard DTW lower bounds for NN-DTW over a wide range of window sizes.

Our paper is organised as follows. In Section 2, we review relevant background and related work. Then we describe our proposed lower bound in Section 3. Section 4 presents an evaluation of our new lower bound in terms of its utility in TSC with NN-DTW. Lastly, we conclude our paper in Section 5.

2 Background and Related Work

We let $A = \langle A_1, \ldots, A_L \rangle$ and $B = \langle B_1, \ldots, B_L \rangle$ be a pair of time series A and B that we want to compare. Note that, for ease of exposition, we assume that the two series are of equal length, but the techniques trivially generalize to unequal length series.

2.1 Dynamic Time Warping The Dynamic Time Warping (DTW) similarity measure was first introduced in [14] for aligning and comparing time series with application to speech recognition. DTW finds the global *alignment* of a time series pair, A and B, as illustrated in Figure 3a. The *warping path* of A and B is a sequence $\mathcal{A} = \langle \mathcal{A}_1, \ldots, \mathcal{A}_P \rangle$ of *links*. Each link is a pair $\mathcal{A}_k = (i, j)$ indicating that A_i is aligned with B_j. $\mathcal{A}$ must obey the following constraints:

- **Boundary Conditions**: $\mathcal{A}_1 = (1, 1)$ and $\mathcal{A}_P = (L, L)$.

- **Continuity** and **Monotonicity**: for any $\mathcal{A}_k = (i, j)$, $1 < k \leq P$, $\mathcal{A}_{k+1} \in \{(i+1, j), (i, j+1), (i+1, j+1)\}$.

The cost of a warping path is minimised using dynamic programming by building a cost matrix D, as illustrated in Figure 3b. Each cell (i, j) of the matrix shows the cost of aligning A_i and B_j. The cost of the warping path for a pair of time series A and B is computed recursively using Equation 2.1, where $\delta(A_i, B_j)$ is the L_2-norm of A_i and B_j.
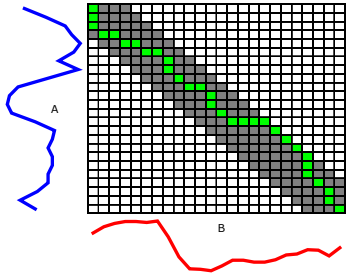
Figure 4: DTW with warping window, $W = 3$

$$(2.1) \quad D(i,j) = \delta(A_i, B_j) + \min \begin{cases} D(i-1, j-1) \\ D(i, j-1) \\ D(i-1, j) \end{cases}$$

Then $\text{DTW}(A, B)$ is calculated using Equation 2.2, where $\mathcal{A}_i^1$ is the first index in $\mathcal{A}_i$ and $\mathcal{A}_i^2$ the second.

$$(2.2) \quad \text{DTW}(A, B) = \sqrt{D(L, L)} = \sqrt{\sum_{i=1}^{L} \delta(A_{\mathcal{A}_i^1}, B_{\mathcal{A}_i^2})}$$

A global constraint on the warping path can be applied to DTW, such that A_i and B_j can only be aligned if they are within a window range, W. This limits the distance in the time dimension that can separate A_i from points in B with which it can be aligned [6,14]. This constraint is known as the warping window, W (previously Sakoe-Chiba band) [14] and we write this as $\text{DTW}_W(A, B)$. Note that we have $1 \leq W \leq L$; DTW_0 corresponds to the Euclidean distance; and DTW_L is equivalent to unconstrained DTW. Figure 4 shows an example with warping window $W=3$, where the alignment of A and B is constrained to be inside the gray band.

Constraining the warping path has two main benefits: (1) increasing NN-DTW accuracy by preventing pathological warping of a pair of time series A and B. (2) speeding up NN-DTW by reducing the complexity of DTW from $O(L^2)$ to $O(W \cdot L)$ [16].

Alternative constraints have also been developed, such as the Itakura Parallelogram [5] and the Ratanamahatana-Keogh band [12]. In this paper, we focus on the Sakoe-Chiba Band which, arguably, is by far the most used constraint in the literature [13,16,18].

2.2 Existing DTW Lower Bounds NN-DTW with lower bound minimises the number of DTW computations. In this section, we review the existing lower bounds for DTW. For the rest of the paper, we will refer a lower bound as LB_⟨name⟩ and consider A as the query time series that is compared to B.

2.2.1 Kim Lower Bound (LB_KIM) [7] is the simplest lower bound for DTW with constant $O(1)$ complexity. LB_KIM extracts four features – distances of the first, last, minimum and maximum points from the time series. Then the maximum of all four features is the lower bound for DTW.

$$\text{LB\_KIM}(A, B) = \max \begin{cases} \delta(A_1, B_1) \\ \delta(A_L, B_L) \\ \delta(A_{\min}, B_{\min}) \\ \delta(A_{\max}, B_{\max}) \end{cases}$$

2.2.2 Yi Lower Bound (LB_YI) [19] takes advantage that all the points in A that are larger than $B_{\max}$ or smaller than $B_{\min}$ shown in Equation 2.2.2, must at least contribute to the final DTW distance. Thus the sum of their distances to $B_{\max}$ or $B_{\min}$ forms a lower bound for DTW.

$$\text{LB\_YI}(A, B) = \sum_{i=1}^{L} \begin{cases} \delta(A_i, B_{\max}) & \text{if } A_i > B_{\max} \\ \delta(A_i, B_{\min}) & \text{if } A_i < B_{\min} \\ 0 & \text{otherwise} \end{cases}$$

2.2.3 Keogh Lower Bound (LB_KEOGH) [6] first creates two new time series, upper $\mathbb{U}^B$ (Equation 2.3) and lower $\mathbb{L}^B$ (Equation 2.4) envelopes. These are the upper and lower bounds on B within the window of each point in A. Then the lower bound is the sum of distances to the envelope of points in A that are outside the envelope of B.

$$(2.3) \quad \mathbb{U}_i^B = \max_{\max(1, i-W) \leq j \leq \min(L, i+W)} (B_j)$$

$$(2.4) \quad \mathbb{L}_i^B = \min_{\max(1, i-W) \leq j \leq \min(L, i+W)} (B_j)$$

$$\text{LB\_KEOGH}_W(A, B) = \sum_{i=1}^{m} \begin{cases} \delta(A_i, \mathbb{U}_i^B) & \text{if } A_i > \mathbb{U}_i^B \\ \delta(A_i, \mathbb{L}_i^B) & \text{if } A_i < \mathbb{L}_i^B \\ 0 & \text{otherwise} \end{cases}$$

2.2.4 Improved Lower Bound (LB_IMPROVED) [8] first computes LB_KEOGH and finds A', the projection of A on to the envelope of B using Equation 2.5, where $\mathbb{U}_i^B$ and $\mathbb{L}_i^B$ are the envelope of B. Then it builds the envelope for A' and computes LB_KEOGH(B, A'). Finally, LB_IMPROVED is the sum of the two LB_KEOGHs.

$$A'_i = \begin{cases} \mathbb{U}^B_i & \text{if } A_i > \mathbb{U}^B_i \\ \mathbb{L}^B_i & \text{if } A_i < \mathbb{L}^B_i \\ A_i & \text{otherwise} \end{cases} \tag{2.5}$$

$$\text{LB\_Improved}_W(A,B) = \text{LB\_Keogh}_W(A,B) + \\ \text{LB\_Keogh}_W(B, A')$$

LB_Improved is tighter than LB_Keogh, but has higher computation overheads, requiring multiple passes over the series. However, to be effective, it is usually used with an early abandon process, whereby the bound determined in the first pass is considered, and if it is sufficient to abandon the search, the expensive second pass is not performed.

2.2.5 New Lower Bound (LB_New) [15] takes advantage of the boundary and continuity conditions for a DTW warping path to create a tighter lower bound than LB_Keogh. The boundary condition requires that every warping path contains (A_1, B_1) and (A_L, B_L). The continuity condition ensures that every A_i is paired with at least one B_j in $\mathcal{B}_i$, where $j \in \{\max(1, i-W) \ldots \min(L, i+W)\}$.

$$\text{LB\_New}_W(A,B) = \delta(A_1, B_1) + \delta(A_L, B_L) + \\ \sum_{i=2}^{L-1} \min_{b \in \mathcal{B}_i} \delta(A_i, b)$$

2.2.6 Cascading Lower Bounds Instead of using standalone lower bounds, multiple lower bounds with increasing complexity can be cascaded, to form an overall tighter lower bound [11]. This greatly increases the pruning power and reduces the overall classification time. The UCR Suite [11] cascades LB_Kim, LB_Keogh(A,B) and LB_Keogh(B,A) to achieve a high speed up in time series search.

3 Proposed DTW Lower Bound

Our proposed lower bounds are based on the observation that the warping paths are very constrained at the start and the end of the series. Specifically, the boundary constraints require that the first link, $\mathcal{A}_1$, is $(1,1)$, the top left of a cost matrix. The continuity and monotonicity constraints ensure that $\mathcal{A}_2 \in \{(1,2), (2,1), (2,2)\}$. If we continue this sequence of sets we get the *left bands*,

$$\mathcal{L}_i^W = \{(\max(1, i-W), i), (\max(1, i-W)+1, i), \ldots, (i, i), \\ (i, i-1), \ldots, (i, \max(1, i-W)\}.$$

These are the alternating bands through the cost matrix shown in Figure 5. We can use these bands to define a lower bound on DTW as explained in Theorem 3.1.
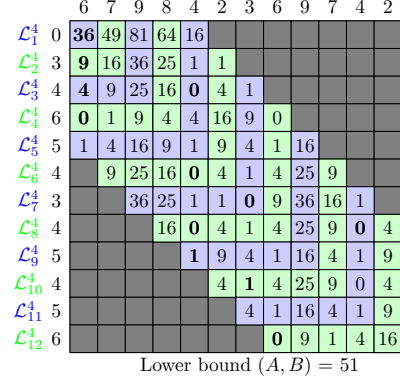
		6	7	9	8	4	2	3	6	9	7	4	2
$\mathcal{L}_1^4$	0	**36**	49	81	64	16							
$\mathcal{L}_2^4$	3	**9**	16	36	25	1	1						
$\mathcal{L}_3^4$	4	**4**	9	25	16	**0**	4	1					
$\mathcal{L}_4^4$	6	**0**	1	9	4	4	16	9	0				
$\mathcal{L}_5^4$	5	1	4	16	9	1	9	4	1	16			
$\mathcal{L}_6^4$	4		9	25	16	**0**	4	1	4	25	9		
$\mathcal{L}_7^4$	3			36	25	1	1	**0**	9	36	16	1	
$\mathcal{L}_8^4$	4				16	**0**	4	1	4	25	9	**0**	4
$\mathcal{L}_9^4$	5					1	9	4	1	16	4	1	9
$\mathcal{L}_{10}^4$	4						4	**1**	4	25	9	0	4
$\mathcal{L}_{11}^4$	5							4	1	16	4	1	9
$\mathcal{L}_{12}^4$	6								0	9	1	4	16

Lower bound $(A,B) = 51$

Figure 5: The cost matrix for calculating a lower bound using *left* bands with $W=4$.

THEOREM 3.1. $\sum_{i=1}^{L} \min_{(j,k) \in \mathcal{L}_i^W} \delta(A_j, B_k)$ *is a lower bound on* $\text{DTW}_W(A,B)$.

Proof. The continuity constraint requires that for all $1 \le i \le L$, any warping path $\mathcal{A}$ must include (i, p) and (q, i), for some $i-W \le p \le i+W$ and $i-W \le q \le i+W$. Either the indexes for both A and B reach i in the same pair and $p = q = i$, or one of the indexes must reach i before the other, and $p < q$ or $p > q$. If $p = q = i$, $(i,i) \in \mathcal{A}$. If $p < q$, $\mathcal{A}$ must contain one of $(i, \max(1, i-W)), \ldots (i, i-1)$. If $p > q$, $\mathcal{A}$ must contain one of $(\max(1, i-W), i), \ldots (i-1, i)$. Thus, $\mathcal{A}$ must contain (at least) one of $\mathcal{L}_i^W$. It follows that

$$\text{DTW}_W(A,B) = \sum_{(j,k) \in \mathcal{A}} \delta(A_j, B_k)$$
$$= \sum_{i=1}^{L} \sum_{(j,k) \in \mathcal{L}_i^W \cap \mathcal{A}} \delta(A_j, B_k)$$
$$\ge \sum_{i=1}^{L} \min_{(j,k) \in \mathcal{L}_i^W} \delta(A_j, B_k).$$

$\square$

Figure 5 illustrates this lower bound in terms of the cost matrix with $W=4$. The columns are elements of A and rows the elements of B. The elements in the matrix show the pairwise distances of each point in the time series pair A and B. Successive $\mathcal{L}_i^W$ are depicted in alternating colors. The minimum distance in each $\mathcal{L}_i^W$ is set in bold type. The sum of these minimums provides a lower bound on the DTW distance.

Working from the other end, the boundary constraints require that $\mathcal{A}_P = (L, L)$, the bottom right of a cost matrix. Continuity and monotonicity constraints ensure that at least one of the *right* band

$$\mathcal{R}_i^W = \{(\min(L, i+W), i), (\min(L, i+W)+1, i), \ldots, (i, i), \\ (i, i-1), \ldots, (i, \min(L, i+W)\}$$

Figure 6: The cost matrix for calculating a lower bound using *right* bands with $W=4$.

	6	7	9	8	4	2	3	6	9	7	4	2	
0	36	49	81	64	**16**								$\mathcal{R}_1^4$
3	9	16	36	25	1	**1**							$\mathcal{R}_2^4$
4	4	9	25	16	**0**	4	1						$\mathcal{R}_3^4$
6	**0**	1	9	4	4	16	9	0					$\mathcal{R}_4^4$
5	1	4	16	9	**1**	9	4	1	16				$\mathcal{R}_5^4$
4		9	25	16	**0**	4	1	4	25	9			$\mathcal{R}_6^4$
3			36	25	1	1	**0**	9	36	16	1		$\mathcal{R}_7^4$
4				16	**0**	4	1	4	25	9	0	4	$\mathcal{R}_8^4$
5					**1**	9	4	1	16	4	1	9	$\mathcal{R}_9^4$
4						4	1	4	25	9	**0**	4	$\mathcal{R}_{10}^4$
5							4	1	16	4	**1**	9	$\mathcal{R}_{11}^4$
6								**0**	9	1	4	16	$\mathcal{R}_{12}^4$

Lower bound $(A,B) = 20$

	6	7	9	8	4	2	3	6	9	7	4	2
0	36	49	81	64	16							
3	9	16	36	25	1	**1**						
4	4	9	25	16	0	4	1					
6	0	**1**	**9**	**4**	4	16	9	0				
5	1	4	16	9	1	9	4	1	16			
4		9	25	16	0	4	1	4	25	9		
3			36	25	1	1	0	9	36	16	1	
4				16	0	4	1	4	25	9	0	**4**
5					1	9	4	1	16	4	1	9
4						4	1	4	25	9	0	4
5							4	1	16	4	1	9
6								0	**9**	**1**	4	16

LB_KEOGH$_4(A,B) = 29$

Figure 7: The cost matrix for calculating LB_KEOGH$_4(A,B)$ with $W=4$.

is in every warping path. Thus,

$$(3.6) \qquad \mathrm{DTW}_W(A,B) \geq \sum_{i=1}^{L} \min_{(j,k)\in\mathcal{R}_i^W} \delta(A_j, B_k).$$

Figure 6 illustrates this lower bound in terms of the cost matrix. The proof of correctness of this bound is a trivial variant of the proof for Theorem 3.1.

On the other hand, LB_KEOGH uses the minimum value from each band in Figure 7 so long as $A_i > \mathbb{U}_i$ or $A_i < \mathbb{L}_i$. When $\mathbb{L}_i^B \leq A_i \leq \mathbb{U}_i^B$, the band is set in gray. For other bands, the minimum is set in bold. Then the sum over all minimum distances in non-gray bands gives LB_KEOGH. It is notable that the leftmost of the left bands and the rightmost of the right bands contain fewer distances than any of the LB_KEOGH bands. All things being equal, on average the minimum of a smaller set of distances should be greater than the minimum of a larger set. Further, because there are fewer distance computations in these few bands, it is feasible to take the true minimum of the band, rather than taking an efficiently computed lower bound on the minimum, as does LB_KEOGH.

	6	7	9	8	4	2	3	6	9	7	4	2
0	**36**	49	81	64	16							
3	**9**	16	36	25	1	**1**						
4	**4**	9	25	16	0	4	1					
6	**0**	1	9	4	4	16	9	0				
5	1	4	16	9	1	9	4	1	16			
4		9	25	16	0	4	1	4	25	9		
3			36	25	1	1	0	9	36	16	1	
4				16	0	4	1	4	25	9	0	4
5					1	9	4	1	16	4	**1**	9
4						4	1	4	25	9	**0**	4
5							4	1	16	4	1	9
6								0	9	1	4	**16**

LB_ENHANCED$_4^4(A,B) = 68$

Figure 8: Cost matrix for calculating LB_ENHANCED$_4^4(A,B)$ with $V=4$ and $W=4$.

3.1 Enhanced Lower Bound Based on these observations, our proposed lower bound exploits the tight leftmost and rightmost bands, but uses the LB_KEOGH bands in the centre section where the left and right bands are larger, and hence less tight and more expensive to compute. This is illustrated in Figure 8.

LB_ENHANCED is parametrized by a tightness parameter V, $1 \leq V \leq L/2$, that specifies how many *left* and *right* bands are utilized. This controls the speed-tightness trade-off. Smaller Vs require less computation, but usually result in looser bounds, while higher values require more computation, but usually provide tighter bounds, as illustrated in Figure 1. LB_ENHANCED is defined as follows

$$(3.7)\quad \mathrm{LB\_ENHANCED}_W^V(A,B) =$$

$$\sum_{i=1}^{L} \begin{cases} \min_{(j,k)\in\mathcal{L}_i^W} \delta(A_j, B_k) & \text{if } i \leq V \\ \min_{(j,k)\in\mathcal{R}_i^W} \delta(A_j, B_k) & \text{if } i > L - V \\ \delta(A_i, \mathbb{U}_i^B) & \text{if } A_i > \mathbb{U}_i^B \\ \delta(A_i, \mathbb{L}_i^B) & \text{if } A_i < \mathbb{L}_i^B \\ 0 & \text{otherwise} \end{cases}$$

where $\mathbb{U}_i^B$ and $\mathbb{L}_i^B$ are defined in Equations 2.3 and 2.4 respectively.

THEOREM 3.2. *For any two time series A and B of length L, for any warping window, $W \leq L$, and for any integer value $V \leq L/2$, the following inequality holds:* LB_ENHANCED$_W^V(A,B) \leq \mathrm{DTW}_W(A,B)$

Proof. From the proof for Theorem 3.1, for every $1 \leq i \leq L$, $\mathcal{A}$ must contain (at least) one of $\mathcal{L}_i^W$ and with trivial recasting this also establishes that for every $1 \leq i \leq L$, $\mathcal{A}$ must contain (at least) one of $\mathcal{R}_i^W$.

To address the contribution of the LB_KEOGH inspired bridge between the $\mathcal{L}$s and $\mathcal{R}$s, we introduce the notion of a vertical band $\mathcal{V}_i$ for element A_i. $\mathcal{V}_i = \{(i,j) : \max(1, i - W) \leq j \leq \min(L, i + W)\}$ is the set

of pairs containing A_i that may appear in a warping path. Note that $\mathcal{L}_1^W, \ldots, \mathcal{L}_V^W$, $\mathcal{V}_{V+1}, \ldots, \mathcal{V}_{L-V}$ and $\mathcal{R}_{L-V+1}^W, \ldots, \mathcal{R}_L^W$ are all mutually exclusive. None of these sets intersects any of the others. It follows

$$\mathrm{DTW}_W(A, B)$$
$$= \sum_{(j,k) \in \mathcal{A}} \delta(A_j, B_k)$$
$$\geq \sum_{i=1}^{V} \sum_{(j,k) \in \mathcal{L}_i^W \cap \mathcal{A}} \delta(A_j, B_k)$$
$$+ \sum_{i=V+1}^{L-V} \sum_{(j,k) \in \mathcal{V}_i \cap \mathcal{A}} \delta(A_j, B_k)$$
$$+ \sum_{i=L-V+1}^{L} \sum_{(j,k) \in \mathcal{R}_i^W \cap \mathcal{A}} \delta(A_j, B_k)$$
$$\geq \sum_{i=1}^{V} \min_{(j,k) \in \mathcal{L}_i^W} \delta(A_j, B_k)$$
$$+ \sum_{i=V+1}^{L-V} \min_{(j,k) \in \mathcal{V}_i} \delta(A_j, B_k)$$
$$+ \sum_{i=L-V+1}^{L} \min_{(j,k) \in \mathcal{R}_i^W} \delta(A_j, B_k)$$
$$\geq \sum_{i=1}^{L} \begin{cases} \min_{(j,k) \in \mathcal{L}_i^W} \delta(A_j, B_k) & \text{if } i \leq V \\ \min_{(j,k) \in \mathcal{R}_i^W} \delta(A_j, B_k) & \text{if } i > L - V \\ \delta(A_i, \mathbb{U}_i^B) & \text{if } A_i > \mathbb{U}_i^B \\ \delta(A_i, \mathbb{L}_i^B) & \text{if } A_i < \mathbb{L}_i^B \\ 0 & \text{otherwise} \end{cases}$$

$\square$

To illustrate our approach, we present the differences between LB_KEOGH and LB_ENHANCED$_4^4$ with respect to A in Figures 7 and 8 respectively.

In Figure 7, the ith column represents $\mathcal{V}_i$, the possible pairs for A_i. The columns are greyed out if $\mathbb{L}_i^B \leq A_i \leq \mathbb{U}_i^B$, showing that they do not contribute to LB_KEOGH. For the remaining columns, the numbers in bold are the minimum distance of A_i to a B_j within A_i's window, either $\mathbb{L}_i^B - A_i$ or $A_i - \mathbb{U}_i^B$. The lower bound is the sum of these values.

In Figure 8, alternate bands are set in alternating colors. The columns are greyed out if $V < i \leq L-V$ and $\mathbb{L}_i^B \leq A_i \leq \mathbb{U}_i^B$, showing that they do not contribute to LB_ENHANCED$_4^4$. For the remaining columns, the numbers in bold are the minimum distance of A_i to B_j within the band. The lower bound is the sum of these values. These figures clearly show the differences

---

Algorithm 1: LB_ENHANCED$(A, B, \mathbb{U}_i^B, \mathbb{L}_i^B, W, V, \mathcal{D})$

   Input: A: Time series A
   Input: B: Time series B
   Input: $\mathbb{U}_i^B$: Upper Envelope for B
   Input: $\mathbb{L}_i^B$: Lower Envelope for B
   Input: W: Warping window
   Input: V: Speed-Tightness parameter
   Input: $\mathcal{D}$: Current distance to NN

1 $res \leftarrow \delta(A_1, B_1) + \delta(A_L, B_L)$
2 $n\_bands \leftarrow \min(L/2, V)$
    // Do $\mathcal{L}$, $\mathcal{R}$ bands
3   **for** $i \leftarrow 2$ **to** n_bands **do**
4 $min_\mathcal{L} \leftarrow \delta(A_i, B_i)$
5 $min_\mathcal{R} \leftarrow \delta(A_{L-i+1}, B_{L-i+1})$
6   **for** $j \leftarrow \max(1, i-W)$ **to** $i-1$ **do**
7 $min_\mathcal{L} \leftarrow \min(min_\mathcal{L}, \delta(A_i, B_j))$
8 $min_\mathcal{L} \leftarrow \min(min_\mathcal{L}, \delta(A_j, B_i))$
9 $min_\mathcal{R} \leftarrow \min(min_\mathcal{R}, \delta(A_{L-i+1}, B_{L-j+1}))$
10 $min_\mathcal{R} \leftarrow \min(min_\mathcal{R}, \delta(A_{L-j+1}, B_{L-i+1}))$
11 $res \leftarrow res + min_\mathcal{L} + min_\mathcal{R}$
12   **if** $res \geq \mathcal{D}$ **then return** ∞
    // Do LB_KEOGH
13   **for** $i \leftarrow n\_bands + 1$ **to** $L - n\_bands$ **do**
14   **if** $A_i > \mathbb{U}_i^B$ **then** $res \leftarrow res + \delta(A_i, \mathbb{U}_i^B)$
15   **else if** $A_i < \mathbb{L}_i^B$ **then** $res \leftarrow res + \delta(A_i, \mathbb{L}_i^B)$
16   **return** res

---

of LB_KEOGH and LB_ENHANCED$_4^4$, where we take advantage of the tighter *left* and *right* bands.

We apply a simple technique to make LB_ENHANCED more efficient and faster. In the naïve version, LB_ENHANCED has to compute the minimum distances of $\mathcal{L}$s and $\mathcal{R}$s. Usually, these computations are very fast as $\mathcal{L}$s and $\mathcal{R}$s are much smaller compared to $\mathcal{V}$, especially when L is long. To optimise LB_ENHANCED, we can first sum the minimum distances for the $\mathcal{L}$s and $\mathcal{R}$s. Then, if this sum is larger than the current distance to the nearest neighbor, $\mathcal{D}$, we can abort the computation for $\mathcal{V}$.

Algorithm 1 describes our proposed lower bound. First, we compute the distance of the first and last points as set by the boundary condition. In line 2, we define the number of $\mathcal{L}$ and $\mathcal{R}$ bands to utilise. This number depends on the warping window, W, as we can only consider the points within W no matter how big V is. Line 3 to 11 computes the sum of the minimum distances for $\mathcal{L}$ and $\mathcal{R}$. If the sum is larger than the current distance to the nearest neighbor, we skip the computation in line 12. Otherwise, we do standard LB_KEOGH for the remaining columns in lines 13 to 15.

4 Empirical Evaluation

Our experiments are divided into two parts. We first study the effect of the tightness parameter V in LB_ENHANCED. Then we show how well LB_ENHANCED can speed up NN-DTW compared to the other lower bounds. We used all the 85 UCR benchmark datasets [3] and the given train/test splits. The relative performance of different lower bounds varies greatly with differing window sizes. In consequence we conduct experiments across a variety of different window sizes, drawn from two sets of values. The set $W=\{1, \ldots, 10\}$, spans the best warping windows for most of the UCR benchmark datasets. The set, $W=\{0.1 \cdot L, 0.2 \cdot L, \ldots, L\}$ shows that using NN-DTW with LB_ENHANCED is always faster across the broad spectrum of all possible windows.

A NN-DTW with lower bound search can be further sped up by ordering the candidates in the training set based on a proxy for their relative distances to the query, such as a lower bound on that distance [17]. However, using a lower bound to order the candidates would unfairly advantage whichever bound was selected. Hence we order the training set by their Euclidean distance to the query time series (an upper bound on their true distance) and start with the candidate that gives the smallest Euclidean distance. Note that our LB_ENHANCED has even greater advantage if random order is employed.

All experiments were optimised and implemented in Java 8 and conducted on a 64-bit Linux AMD Opteron 62xx Class CPU @2.4GHz machine with 32GB RAM and 4 CPUs. Our source code are open-source at https://github.com/ChangWeiTan/LbEnhanced and the full results at http://bit.ly/SDM19.

4.1 How to choose the right tightness parameter for LB_Enhanced? Recall that our LB_ENHANCED is parameterized by a tightness parameter V that specifies the number of bands to be used. This parameter controls the speed tightness trade-off. Higher values require more computations but usually gives tighter bounds. We conducted a simple experiment by recording the classification time of NN-DTW with LB_KEOGH as the baseline and LB_ENHANCED with different tightness parameter in the range of $V=\{1, \ldots, 20\}$. Note that all the required envelopes have been pre-computed at training time and the time is not included in the classification time. Then the classification time of NN-DTW with LB_ENHANCED is normalised by LB_KEOGH. Finally the geometric mean is computed over all 85 datasets.

The results are presented in Figure 9 where we show the performance for a subset of windows. The
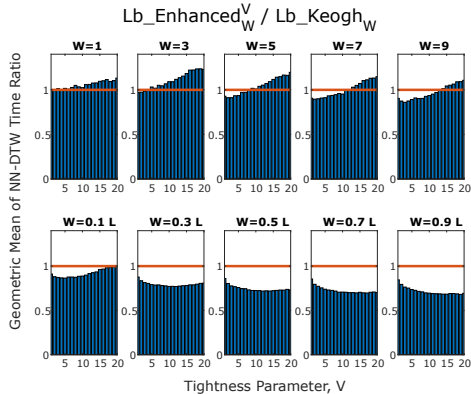

Figure 9: Geometric mean of NN-DTW classification time with LB_ENHANCED of different tightness parameter V normalised by LB_KEOGH. Smaller ratio means faster, and values below 1 (under the red line) indicate LB_ENHANCED is faster than LB_KEOGH.

x axis shows the different Vs, the y axis shows the geometric mean of the normalised time. Ratios below 1 (under the red line) means that LB_ENHANCED is faster than LB_KEOGH and smaller ratio means faster LB_ENHANCED.

These plots show that the optimal value of V increases with W. At $W=1$, only $V=1$ and $V=2$ outperform LB_KEOGH. At $W=3$, all $V < 5$ prove to speed up NN-DTW more than LB_KEOGH, and subsequently $V < 10$ for $W=5$ and $V < 15$ for $W=7$ and $W=9$. At larger window sizes, all $V < 20$ are more effective at speeding up NN-DTW than LB_KEOGH. For $W=0.1 \times L$, $V = 5$ proves to be most effective. For $W \geq 0.3 \times L$, there are a wide range of values of V with very similar performance. One reason for this is that window size is not the only factor that affects the optimal value of V. Series length and the amount of variance in the sequence prefixes and suffixes are further relevant factors. $V=5$ provides strong performance across a wide range of window sizes. In consequence, in the next section we choose $V=5$ and compare the performance of LB_ENHANCED[5] to other lower bounds.

4.2 Speeding up NN-DTW with LB_Enhanced We compare our proposed lower bounds against four key existing alternatives, in total 5 lower bounds:

- **LB_Kim:** The original LB_KIM proposed in [7] is very loose and incomparable to the other lower bounds. To make it tighter and comparable, instead of the maximum, we take the sum of all the four features without repetitions.

- **LB_Keogh:** We use the original implementation of LB_KEOGH proposed in [6].

- **LB_Improved:** We use the original implementation of LB_Improved and the optimised algorithm to compute the projection envelopes for LB_Improved proposed in [8].

- **LB_New:** We use the original implementation of LB_New proposed in [15].

- **LB_Enhanced:** We use LB_Enhanced[5], selecting $V=5$ as it provides reasonable speed up across a wide range of window sizes.

Note that LB_Yi was omitted because it is similar to LB_Keogh when $W=L$. Similar to before, we record the classification time of NN-DTW with the various lower bounds. For each dataset we determine the rank of each bound, the fastest receiving rank 1 and the slowest rank 5. Figure 10 shows the critical difference (CD) diagram comparing the classification time ranks of each lower bound for $W=\{3, 6, 10, 0.1 \cdot L, 0.5 \cdot L\}$. The results for the remaining of the windows can be found in our supplementary paper and http://bit.ly/SDM19. Each plot shows the average rank of each lower bound (the number next to the name). Where the ranks are not significantly different (difference less than CD), their corresponding lines are connected by a solid black line [4]. Thus, for $W=10$, LB_Enhanced[5] has significantly lower average rank than any other bound and the average ranks of LB_Keogh and LB_Improved do not differ significantly but are significantly lower than those of LB_New and LB_Kim; which in turn do not differ significantly from one another.

Our LB_Enhanced[5] has the best average rank of all the lower bounds at all window sizes, significantly so at $W=6$ to 10 and $0.5 \cdot L$ to L. For smaller windows, LB_Keogh is the best of the remaining bounds. For larger windows, the tighter, but more computationally demanding LB_Improved comes to the fore.

We further extend our analysis by computing the speed up gained from LB_Enhanced[5] relative to the other lower bounds. We compute the speedup of LB_Enhanced[5] to all the other lower bounds for $W = \{1, \ldots 10\}$ and present the geometric mean (average) in Figure 11. The results show that LB_Enhanced[5] is consistently faster than all the other lower bounds.

It might be thought that our experimental comparison has unfairly penalized LB_Keogh and LB_New relative to LB_Improved and LB_Enhanced, as only the latter use a form of early abandoning [11]. However, LB_Improved starts with LB_Keogh and LB_Enhanced uses LB_Keogh for most of the sequence. Hence, each of these could benefit as much as would LB_Keogh from the adoption of early abandoning in the LB_Keogh process.
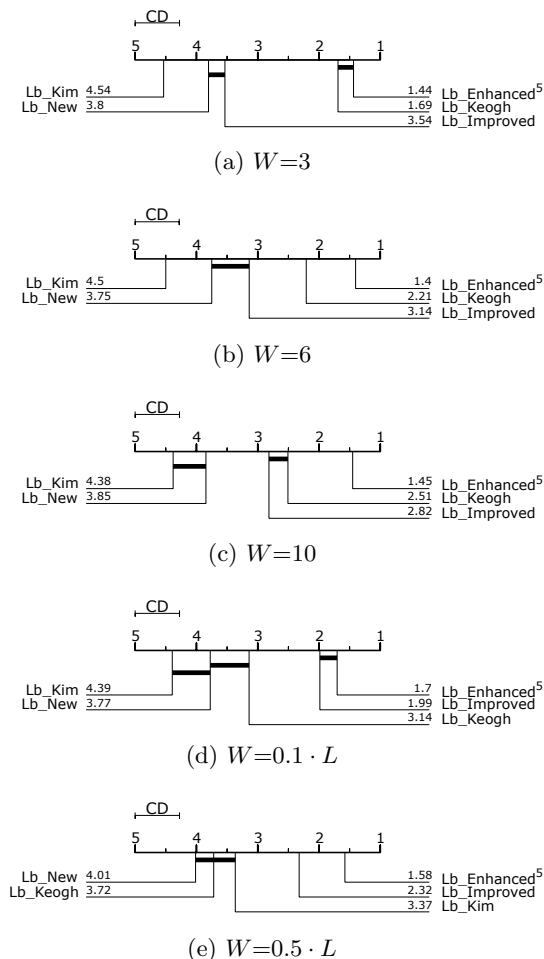

(a) $W=3$

(b) $W=6$

(c) $W=10$

(d) $W=0.1 \cdot L$

(e) $W=0.5 \cdot L$

Figure 10: Ranking of all lower bounds in terms of NN-DTW classification time

5 Conclusion and future work

In conclusion, we proposed LB_Enhanced, a new lower bound for DTW. The speed-tightness trade-off of LB_Enhanced results in faster lower bound search for NN-DTW than any of the previous established bounds at all window sizes. We expect it to be similarly effective at a wide range of nearest neighbor retrieval tasks under DTW. We showed that choosing a small tightness parameter V is sufficient to effectively speed up NN-DTW. Although it is possible to learn the best V for a dataset (which will be future work), our results show that when $V=5$, NN-DTW with LB_Enhanced[5] is faster and more efficient than with the existing lower bounds for all warping window sizes across 85 benchmark datasets.

In addition, there is potential to replace LB_Keogh by LB_Improved within LB_Enhanced. This would increase the computation time, but the strong performance of LB_Improved suggests it should result
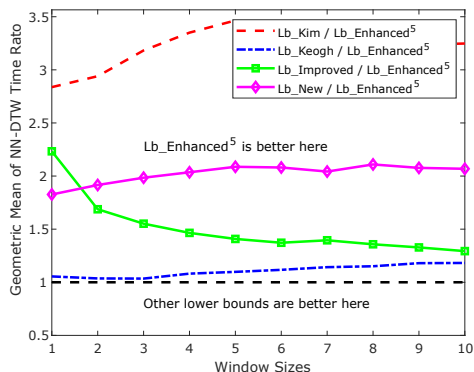
Figure 11: Geometric mean (average) ratio of classification time for major existing lower bounds to our proposed new lower bound LB_ENHANCED[5] across all benchmark datasets [3] at $W = \{1, \ldots 10\}$.

in a powerful trade-off between time and tightness especially at larger windows. Finally, since LB_KEOGH is not symmetric with respect to A and B, LB_ENHANCED is not symmetric too. Thus, $\max(\text{LB\_ENHANCED}(A, B), \text{LB\_ENHANCED}(B, A))$ is also a useful bound. Our proposed lower bound could also be cascaded [11], which may further improve pruning efficiency.

6 Acknowledgement

References

[1] A. BAGNALL, J. LINES, A. BOSTROM, J. LARGE, AND E. KEOGH, *The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances*, Data Mining and Knowledge Discovery, 31 (2017), pp. 606–660.

[2] A. BAGNALL, J. LINES, J. HILLS, AND A. BOSTROM, *Time-series classification with COTE: the collective of transformation-based ensembles*, IEEE Transactions on Knowledge and Data Engineering, 27 (2015).

[3] Y. CHEN, E. KEOGH, B. HU, N. BEGUM, A. BAGNALL, A. MUEEN, AND G. BATISTA, *The UCR Time Series Classification Archive*, 7 2015. www.cs.ucr.edu/~eamonn/time_series_data/.

[4] J. DEMŠAR, *Statistical comparisons of classifiers over multiple data sets*, Journal of Machine learning research, 7 (2006), pp. 1–30.

[5] F. ITAKURA, *Minimum prediction residual principle applied to speech recognition*, IEEE Transactions on Acoustics, Speech, and Signal Processing, 23 (1975), pp. 67–72.

[6] E. KEOGH AND C. A. RATANAMAHATANA, *Exact indexing of dynamic time warping*, Knowledge and information systems, 7 (2005), pp. 358–386.

[7] S.-W. KIM, S. PARK, AND W. W. CHU, *An index-based approach for similarity search supporting time warping in large sequence databases*, in Data Engineering, 2001. Proceedings. 17th International Conference on, IEEE, 2001, pp. 607–614.

[8] D. LEMIRE, *Faster retrieval with a two-pass dynamic-time-warping lower bound*, Pattern recognition, 42 (2009), pp. 2169–2180.

[9] J. LINES AND A. BAGNALL, *Time series classification with ensembles of elastic distance measures*, Data Mining and Knowledge Discovery, 29 (2015), pp. 565–592.

[10] F. PETITJEAN, G. FORESTIER, G. I. WEBB, A. E. NICHOLSON, Y. CHEN, AND E. KEOGH, *Dynamic time warping averaging of time series allows faster and more accurate classification*, in Data Mining (ICDM), 2014 IEEE International Conference on, IEEE, 2014, pp. 470–479.

[11] T. RAKTHANMANON, B. CAMPANA, A. MUEEN, G. BATISTA, B. WESTOVER, Q. ZHU, J. ZAKARIA, AND E. KEOGH, *Searching and mining trillions of time series subsequences under dynamic time warping*, in Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2012, pp. 262–270.

[12] C. RATANAMAHATANA AND E. KEOGH, *Making time-series classification more accurate using learned constraints*, in SIAM SDM, 2004.

[13] ——, *Three myths about DTW data mining*, in SIAM SDM, 2005, pp. 506–510.

[14] H. SAKOE AND S. CHIBA, *A dynamic programming approach to continuous speech recognition*, in International Congress on Acoustics, vol. 3, 1971, pp. 65–69.

[15] Y. SHEN, Y. CHEN, E. KEOGH, AND H. JIN, *Accelerating time series searching with large uniform scaling*, in Proceedings of the 2018 SIAM International Conference on Data Mining, SIAM, 2018, pp. 234–242.

[16] C. W. TAN, M. HERRMANN, G. FORESTIER, G. I. WEBB, AND F. PETITJEAN, *Efficient search of the best warping window for dynamic time warping*, in Proceedings of the 2018 SIAM International Conference on Data Mining, SIAM, 2018, pp. 225–233.

[17] C. W. TAN, G. I. WEBB, AND F. PETITJEAN, *Indexing and classifying gigabytes of time series under time warping*, in Proceedings of the 2017 SIAM International Conference on Data Mining, SIAM, 2017, pp. 282–290.

[18] X. XI, E. KEOGH, C. SHELTON, L. WEI, AND C. RATANAMAHATANA, *Fast time series classification using numerosity reduction*, in ICML, 2006, pp. 1033–1040.

[19] B.-K. YI, H. JAGADISH, AND C. FALOUTSOS, *Efficient retrieval of similar time sequences under time warping*, in Data Engineering, 1998. Proceedings., 14th International Conference on, IEEE, 1998, pp. 201–208.