Noname manuscript No.
(will be inserted by the editor)

A Shape-based Similarity Measure for Time Series Data with Ensemble Learning

Tetsuya Nakamura · Keishi Taki · Hiroki Nomiya · Kazuhiro Seki · Kuniaki Uehara

Received: date / Accepted: date

Abstract This paper introduces a shape-based similarity measure, called the *Angular Metric for Shape Similarity* (AMSS), for time series data. Unlike most similarity or dissimilarity measures, AMSS is based not on individual data points of a time series but on vectors equivalently representing it. AMSS treats a time series as a vector sequence to focus on the shape of the data and compares data shapes by employing a variant of cosine similarity. AMSS is, by design, expected to be robust to time and amplitude shifting and scaling, but sensitive to short-term oscillations. To deal with the potential drawback, ensemble learning is adopted, which integrates data smoothing when AMSS is used for classification. Evaluative experiments reveal distinct properties of AMSS and its effectiveness when applied in the ensemble framework as compared with existing measures.

Keywords Time series analysis · Similarity measures · Machine learning

1 Introduction

A time series is a sequence of values measured at successive time intervals, where the intervals can be either constant or variable. Such time series data can arise in any disciplines, such as agriculture, chemistry, demography, and finance. When analyzing time series data, the most essential yet still open question is how to best compute similarity/dissimilarity between two time series. Defining (dis)similarity is of central importance for many applications including time series classification, clustering, search, trend analysis, and forecasting [15] as their outcomes are directly affected by the definition. However, (dis)similarity computation involves several issues regarding data alignment, outliers, and spatial variation. To tackle them, numbers of (dis)similarity measures have been proposed [2], such as dynamic time warping (DTW) and longest common subsequences (LCSS). Most measures, however, are more or less similar in a sense that they are essentially dealing with individual data points composing the time series. An exception from this viewpoint is derivative DTW (DDTW) [19], which is based on approximated local derivatives instead of data points.

This paper introduces a similarity measure, called the *Angular Metric for Shape Similarity* (AMSS). Similar to DDTW, AMSS is not based on individual data points. Instead, it focuses on a vector sequence equivalently representing time series data. Compared with the existing measures based on data points, an advantage of AMSS is that it better handles spatial variation by converting data points into vectors in comparing two sequences. Also, AMSS adopts a variant of cosine similarity to minimize the influence of outliers in similarity computation, where outliers are defined as much bigger or smaller data points than their immediate neighbors.

Generally speaking, a given (dis)similarity measure is not effective for every kind of time series data; it may be suited for some types of data, and not so for the others. AMSS is not an exception in this regard. Empirical results will show that it is relatively robust to data containing outliers but by nature susceptible to short-term oscillations. Here, short-term oscillations are defined as repetitive, frequent fluctuations with typically low amplitude that do not reflect the characteristics of

T. Nakamura, K. Taki, H. Nomiya, K. Seki, and K. Uehara
Kobe University, 1-1 Rokkodai, Nada, Kobe 657-8501, Japan
Tel.: +81-78-803-6480
Fax: +81-78-803-6316
E-mail: seki@cs.kobe-u.ac.jp

the data (e.g., white noise). Note that "outliers" and "short-term oscillations" are distinguished throughout this paper, although they both can be seen as noise in general. To deal with the problem, ensemble learning is adopted so as to effectively handle data with/without short-term oscillations. This framework integrates different smoothing algorithms or (dis)similarity measures by considering the characteristics of given data.

The rest of this paper is organized as follows: Section 2 summarizes the related work. Section 3 details the algorithm of AMSS and Section 4 introduces the ensemble learning framework. Section 5 evaluates the effectiveness of AMSS in comparison with other representative measures. Finally, Section 6 offers conclusion and possible directions for future work.

2 Related work

There have been many (dis)similarity measures proposed for time series data. Generally, they can be categorized into lock-step, elastic, threshold-based, and patterns-based measures [9].

For lock-step measures, the most widely known one would be Euclidean distance [10], defined as the square root of the sum of the squared differences between corresponding data points in two time series data. The main problem of the measure is that the input sequences need to have the same length. A newer measure in this category, DISSIM [11], provides a solution to this problem. Although DISSIM can compute similarity between two data with different sampling rates, the downside is that the computation is costly. This measure could yield high classification accuracy when there are a large amount of labeled data but in general do not compare favorably with elastic measures discussed next.

The family of elastic measures uses dynamic programming to align sequences with different lengths, including DTW [2], LCSS [7], edit distance with real penalty (ERP) [4], and edit distance on real sequence (EDR) [5]. The proposed measure, AMSS, also falls in this category. DTW computes dissimilarity between two given sequences by finding the best warping path in their distance matrix. LCSS was originally developed as a similarity measure between trajectories of mobile objects. It can reduce the influence of outliers by quantizing the distance between two data points into two values, 0 and 1, representing distant and near, respectively. EDR and ERP are based on edit distance originally proposed for measuring dissimilarity between two character strings. EDR requires a threshold by which the distance between two data points is quantized into 0 or 1. ERP is based on EDR but uses actual distance instead of the dichotomous values. The sequence weighted alignment model (Swale) [22] can be seen as another elastic measure but not employing dynamic programming. A major difference between these elastic measures and AMSS is that the former looks only at individual data points in computing (dis)similarity, where shapes of trajectories are not considered. In this aspect, DDTW, which focuses on approximated local derivatives, is similar to AMSS as mentioned above.

More recently, other types of (dis)similarity measures, TQuEST [1] and SpADe [6], have been proposed, which are categorized as threshold-based and pattern-based measures, respectively. The former accepts a user-provided threshold τ and converts sequence data to so called *threshold crossing*, which are treated as points in two-dimensional space composed only of data points above τ. The output dissimilarity is defined as the Minkowski sum. The latter, SpADe, first finds representative matching segments, called *local patterns*, in a sequence by focusing on amplitude and trajectories. SpADe is similar to AMSS in a sense that it also looks at the shapes of data. A critical difference is, however, SpADe requires many parameters, including the number of local patterns, gap bound, time shifting factor, amplitude shifting factor, time scale factor, and amplitude scale factor, which must be manually tuned for each data set, whereas AMSS has no parameter to tune as will be described shortly.

3 Angular Metric for Shape Similarity

3.1 Basic idea

Unlike other measures, AMSS treats a time series as a vector sequence. This conversion allows us to better handle spatial variation. In addition, AMSS calculates similarity between two vector sequences based on vectors' directions, not on the actual locations of the data points in the d-dimensional space.

Figure 1 presents a simple example to illustrate how the similarity between 1-dimensional time series data C and Q is calculated by AMSS, where C and Q are represented as a sequence of vectors as described shortly in the next section. In Figure 1 (a), C contains successive vectors $\mathbf{c}_1$ to $\mathbf{c}_6$ which appear to be similar to vectors $\mathbf{q}_1$ to $\mathbf{q}_6$ in Q. Although subsequent vectors, $\mathbf{c}_7$ to $\mathbf{c}_{13}$, in C are not quite similar to $\mathbf{q}_7$ to $\mathbf{q}_{14}$ in Q partly due to a sudden increase by $\mathbf{c}_7$, AMSS will be less affected by the large discrepancy. Figure 1 (b) shows how the vector sequences would be aligned each other. Although $\mathbf{c}_7$ and $\mathbf{q}_7$ are quite different in direction and would (locally) lead to low similarity, AMSS can properly compute global similarity between C and Q by pairing similar subsequences and by focusing on the shapes of the subsequences represented by vector directions. In contrast, other data point-based measures, such as DTW, would fail to recognize their similarity since $\mathbf{c}_7$ to $\mathbf{c}_{13}$ and $\mathbf{q}_7$ to $\mathbf{q}_{14}$ are far apart with respect to their values. Note that shape-based measures, such as DDTW and SpADe, would be also effective for these types of data. Section 5 will empirically compares AMSS with these measures and discusses their differences.

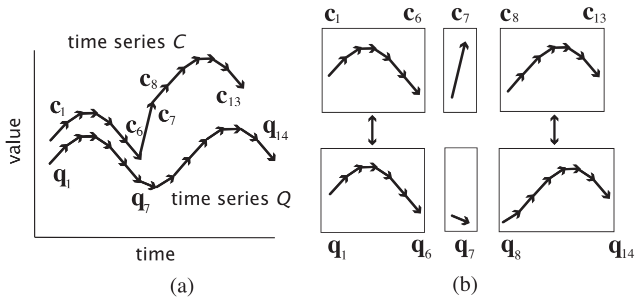


Fig. 1 Two time series data compared by AMSS.

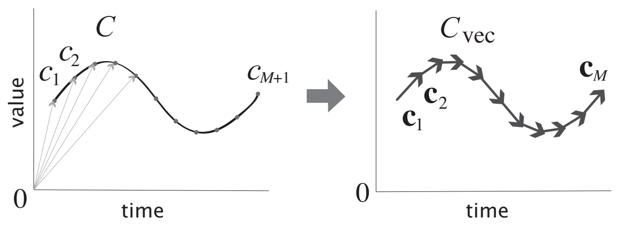


Fig. 2 Time series data C represented as a vector sequence C_{vec}.

3.2 Algorithm

Suppose that a 1-dimensional time series data sequence C is composed of $M+1$ elements as

$$\begin{aligned} C &= ((p_1,t_1),(p_2,t_2),\ldots,(p_{M+1},t_{M+1})) \\ &= (c_1,c_2,\ldots,c_{M+1}) \end{aligned} \tag{1}$$

where p_m is a value (data point) and t_m, $m = 1,2,\ldots,M+1$, is the time when p_m was measured and m-th element (p_m,t_m) of C was denoted as c_m for short. By treating time t_m as if it is another coordinate, the coordinates of each point p_m can be represented as a position vector in the 2-dimensional space. Here, a new symbol $\mathbf{c}_m$ is introduced as a displacement vector connecting two successive points, which can be obtained as their difference, written as $c_{m+1}-c_m$. Then, the whole sequence C can be equivalently represented as a sequence C_{vec} of M vectors.[1]

$$\begin{aligned} C_{\mathrm{vec}} &= ((c_2-c_1),(c_3-c_2),\ldots,(c_{M+1}-c_M)) \\ &= (\mathbf{c}_1,\mathbf{c}_2,\ldots,\mathbf{c}_M) \end{aligned} \tag{2}$$

Figure 2 illustrates an example of the conversion from coordinates to vectors for 1-dimensional time series data.

In the same way, let Q_{vec} be a sequence of vectors converted from $Q=(q_1,q_2,\ldots,q_{N+1})$ as follows.

$$\begin{aligned} Q_{\mathrm{vec}} &= ((q_2-q_1),(q_3-q_2),\ldots,(q_{N+1}-q_N)) \\ &= (\mathbf{q}_1,\mathbf{q}_2,\ldots,\mathbf{q}_N) \end{aligned} \tag{3}$$

Given two vectors $\mathbf{q}_n$, $n=1,2,\ldots,N$, in Q_{vec} and $\mathbf{c}_m$ in C_{vec} as shown in Figure 3 (a), their (dis)similarity can be quantified by many different measures. Among them, most widely used ones are Euclidean distance and cosine similarity. The former is the length between the end points of the two vectors in the Euclidean space, and the latter is the cosine of the angle between the two vectors, concerning only the directions they point to. AMSS aims to be robust to spatial variation, where exact points of the vectors are of less importance than their directions. Thus, this work adopts cosine similarity as a principle measure which disregards exact points or vector length.[2] The definition of the standard cosine similarity is slightly modified as

$$\mathrm{sim}(\mathbf{q}_n,\mathbf{c}_m) = \begin{cases} 0 & \textbf{if } \theta > \pi/2 \\ \cos\theta \left(= \dfrac{\mathbf{q}_n \cdot \mathbf{c}_m}{|\mathbf{q}_n|\,|\mathbf{c}_m|} \right) & \textbf{otherwise} \end{cases} \tag{4}$$

where θ is the angle between $\mathbf{q}_n$ and $\mathbf{c}_m$ as illustrated in Figure 3 (b). The modification sets the similarity to 0 for θ greater than $\pi/2$, which limits the influence of two vectors with widely different directions. Note that θ is less than π because one of the dimensions of the vectors is time, which increases monotonically

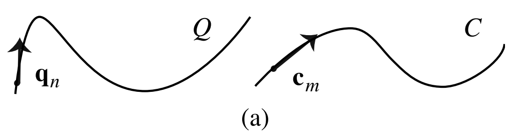
After calculating the similarity for every pair of vectors in the sequences Q_{vec} and C_{vec}, which forms an $N \times M$ similarity matrix, AMSS identifies a warping path W of length K:

$$W = (w_1,w_2,\ldots,w_K) \tag{5}$$

where w_k, $k=1,2,\ldots,K$, is a pair of indices (n_k,m_k) indicating the corresponding vectors $\mathbf{c}_{n_k}$ and $\mathbf{q}_{m_k}$.

[1] To be precise, C and C_{vec} are not exactly the same because the latter does not retain the position of the first element c_1.

[2] Another shape-based similarity measure, DDTW, uses Euclidean distance. However, the distance is measured between the slopes of two vectors, which is similar to (inversed) cosine similarity in concept.



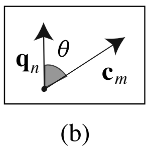


Fig. 3 A similarity measure between two vectors.

A warping path is typically subject to several constraints [17]. The followings are the constraints AMSS also complies with.

- **Boundary condition.** This constraint requires W to start and end in diagonally opposite corner cells of the similarity matrix and is expressed as $w_1 = (1, 1)$ and $w_K = (N, M)$.
- **Continuity.** This constraint requires W to lie across (diagonally) adjacent cells. More formally, given $w_k = (n_k, m_k)$ and $w_{k+1} = (n_{k+1}, m_{k+1})$, $n_{k+1} - n_k$ and $m_{k+1} - m_k$ equal either 0 or 1.
- **Monotonicity.** This constraint requires W to never go back. That is, given $w_k = (m_k, n_k)$ and $w_{k+1} = (m_{k+1}, n_{k+1})$, $m_k \leq m_{k+1}$ and $n_k \leq n_{k+1}$ always hold.

There are many warping paths that satisfy the above conditions, and generally the best path is selected based on some cost function $C(W)$. This function is typically defined as the sum of the similarities/dissimilarities associated with the elements of the warping path. The present work follows this bottom-up approach and defines the overall similarity between Q_{vec} and C_{vec}, denoted as AMSS$(Q_{\text{vec}}, C_{\text{vec}})$, by using the following recursive function:

$$\begin{aligned}&\text{AMSS}(Q_n, C_m) = \qquad (6)\\ &\max\{\text{AMSS}(Q_{n-1}, C_{m-1}) + 2\text{sim}(\mathbf{q}_n, \mathbf{c}_m),\\ &\text{AMSS}(Q_{n-2}, C_{m-1}) + 2\text{sim}(\mathbf{q}_{n-1}, \mathbf{c}_m) + \text{sim}(\mathbf{q}_n, \mathbf{c}_m),\\ &\text{AMSS}(Q_{n-1}, C_{m-2}) + 2\text{sim}(\mathbf{q}_n, \mathbf{c}_{m-1}) + \text{sim}(\mathbf{q}_n, \mathbf{c}_m)\}\end{aligned}$$

where new symbols Q_n and C_m are introduced to denote vector subsequences $(\mathbf{q}_1, \ldots, \mathbf{q}_n)$ and $(\mathbf{c}_1, \ldots, \mathbf{c}_m)$, respectively. Using the new notations, AMSS$(Q_{\text{vec}}, C_{\text{vec}})$ can be expressed as AMSS(Q_N, C_M). To terminate the recursion, AMSS(Q_1,C_1) is defined to be sim$(\mathbf{q}_1, \mathbf{c}_1)$. Also, AMSS$(Q_n, C_m)$ for $n = 0$ or $m = 0$ is defined as $-\infty$ to avoid invalid paths involving undefined vectors q_0 or c_0.[3]

[3] This situation can arise when assessing AMSS(Q_{N-2}, C_{M-1}) or AMSS(Q_{N-1}, C_{M-2}).

The three elements appearing in the max function in Equation (6) represent three different paths reaching an intersection of n- and m-th grid lines as illustrated in Figure 4 (a), where each intersection is associated with a similarity value between the corresponding vectors. The three paths are introduced as the restriction to limit the global warping path to the gray area in Figure 4 (b) and gives a similar effect to Itakura parallelogram [16] to speed up computation. Note that the second terms, sim$(\mathbf{q}_n, \mathbf{c}_m)$, sim$(\mathbf{q}_{n-1}, \mathbf{c}_m)$, and sim$(\mathbf{q}_n, \mathbf{c}_{m-1})$, in each element are multiplied by 2 to give diagonal paths higher weights equivalent to a combination of a vertical and a horizontal paths. AMSS(Q_n, C_m) can be efficiently computed by dynamic programming, yielding the "best" warping path in a sense that it maximizes the overall cumulative similarity.

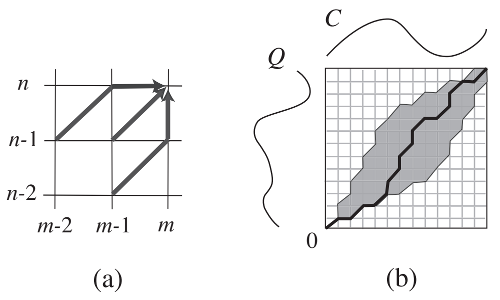


Fig. 4 A constraint on the warping path. The bold line in (b) indicates a hypothetical warping path maximizing AMSS.

Note that for simplicity 1-dimensional data (Equation (1)) is used as an example in this section, but the above discussion can be generalized to multi-dimensional data without any modification. Also note that, in the following sections, C and Q are used to refer to C_{vec} and Q_{vec}, respectively, to keep the notation simple when they cannot be confused from the context.

3.3 An Upper Bound Function for AMSS

A strategy to speed up similarity computation is to limit the search space by imposing a global constraint, such as Itakura parallelogram [16] and Sakoe-Chiba band [27]. AMSS described in the previous section achieves this by way of Equation (6). While this constraint on the search space practically accelerates similarity computation, the time complexity of AMSS is still $O(MN)$ because the gray area in Figure 4 (b) extends as the length of time series increases. Other elastic measures, including DTW, ERP, EDR, and LCSS, utilizing dynamic time warping generally have the same time complexity $O(MN)$, although the aforementioned global constraint can be applied to reduce the computation time.

Another strategy to reduce the computation time is to introduce an upper/lower bound function. This section describes such an upper bound function for AMSS, denoted as $\mathrm{UB}(Q, C)$, based on vector composition. Note that any upper bound function is required to satisfy the following condition to avoid false dismissal.

$$\mathrm{UB}(Q, C) \geq \mathrm{AMSS}(Q, C) \tag{7}$$

With such an upper bound function and a set of time series data $\{S_1, S_2, \ldots\}$, the most similar time series among the set to a given input Q can be efficiently found by the following steps.

1. Calculate $\mathrm{AMSS}(Q, S_1)$ and assume for now that S_1 is most similar to Q.
2. Calculate the upper bound $\mathrm{UB}(Q, S_2)$ of the similarity between Q and S_2.
   - If $\mathrm{UB}(Q, S_2)$ is smaller than $\mathrm{AMSS}(Q, S_1)$ calculated above, rule out S_2 because it follows from Equation (7) that $\mathrm{AMSS}(Q, S_2)$ is smaller than $\mathrm{AMSS}(Q, S_1)$. That is,
     $$\mathrm{AMSS}(Q, S_1) \geq \mathrm{UB}(Q, S_2) \geq \mathrm{AMSS}(Q, S_2). \tag{8}$$
   - If $\mathrm{UB}(Q, S_2)$ is greater than $\mathrm{AMSS}(Q, S_1)$, $\mathrm{AMSS}(Q, S_2)$ is calculated. If $\mathrm{AMSS}(Q, S_2)$ is also greater than $\mathrm{AMSS}(Q, S_1)$, the most similar sequence to S_2 is updated.
3. Repeat from the step 2 above for the remaining sequences in the set.

3.4 Definition of an Upper Bound Function

AMSS compares vector sequences to compute their similarity. Each input vector sequence can potentially be converted to a simpler, shorter sequence if its shape is smooth, which would speed up similarity computation. Consider the example in Figure 5, where a vector sequence Q composed of ten vectors is converted to Q' composed of three vectors by vector composition. There are many ways to combine a subsequence of vectors and one should take the one closely resembling the original shape of the vector sequence Q. The closeness or difference can be measured by the shaded area in Figure 5 (b) between the shapes of the original vector sequences Q and the combined, resultant vector sequences Q'. A predefined threshold is imposed to the area and a compact representation Q' is created. Briefly, consecutive vectors $q_i, \ldots, q_j$, $i, j \in \mathbb{N}$, $i < j$, are incrementally combined to create a resultant vector q'_k when the area enclosed by $q_i, \ldots, q_j$ and q'_k does not exceed the threshold.

To compute the upper bound UB based on resultant vector sequences, a straightforward approach

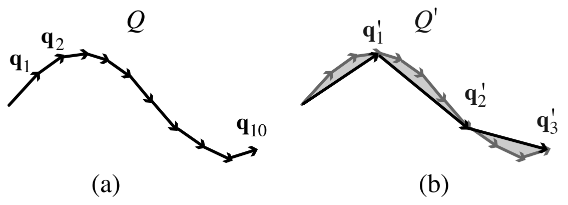


Fig. 5 An example of vector composition applied to Q to yield Q'.

would be to apply the original AMSS to resultant vector sequences by regarding $\mathrm{AMSS}(Q', C')$ as an upper bound function $\mathrm{UB}(Q, C)$. However it does not generally satisfy the condition expressed in Equation (7). For instance, consider the case where a vector $\mathbf{c}_1$ is compared with a resultant vector $\mathbf{q}'_1$ as in Figure 6 (a). In this example, the angle between $\mathbf{c}_1$ and $\mathbf{q}'_1$ is larger than the angle between $\mathbf{c}_1$ and $\mathbf{q}_1$, and thus, $\mathrm{sim}(\mathbf{q}_1, \mathbf{c}_1) > \mathrm{sim}(\mathbf{q}'_1, \mathbf{c}_1)$. This makes the upper bound $\mathrm{UB}(Q, C)$ smaller than $\mathrm{AMSS}(Q, C)$. In addition, because AMSS is defined as a weighted sum of the similarity scores, the similarities associated with q_2 and q_3 need to be added up, which would make the AMSS score for the original vectors, q_1 to q_3, greater than that for the single resultant vector q'_1. This case also violates the aforementioned condition and results in false dismissal in similarity search.

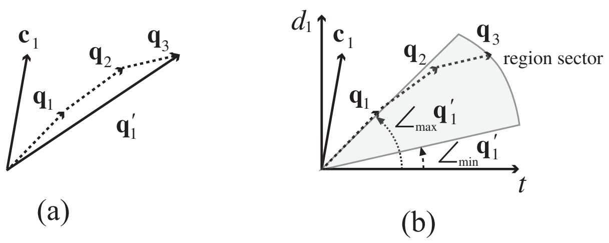


Fig. 6 An example of a resultant vector $\mathbf{q}'_1$ and its associated region sector.

To deal with the above problem, one can take advantage of the angles of the original vectors (i.e., $\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3$) and represents the resultant vector (i.e., $\mathbf{q}'_1$) by a region, named *region sector*, between the smallest and largest angles denoted by $\angle_{\min}\mathbf{q}'_1$ and $\angle_{\max}\mathbf{q}'_1$, respectively. This way, the region sector for $\mathbf{q}'_1$ encloses all the original vectors, and thus, the angle between the region sector and the vector to be compared ($\mathbf{c}_1$ for this example) is always equal or less than that between any original vector and $\mathbf{c}_1$. More formally, when ν-th resultant vector $\mathbf{q}'_\nu$ is created from $\mathbf{q}_i$ to $\mathbf{q}_j$, $i \leq j$, of a sequence Q, $\angle_{\max}\mathbf{q}'_\nu$ and $\angle_{\min}\mathbf{q}'_\nu$ are defined as follows.

$$\angle_{\max}\mathbf{q}'_\nu = \max\{\angle\mathbf{q}_i, \ldots, \angle\mathbf{q}_j\} \tag{9}$$

$$\angle_{\min}\mathbf{q}'_\nu = \min\{\angle\mathbf{q}_i, \ldots, \angle\mathbf{q}_j\} \tag{10}$$

If a region sector is compared with a vector $\mathbf{c}_m$, the smallest angle between them is equal to or smaller than that between the vector $\mathbf{c}_m$ and any of the original vectors $\mathbf{q}_i, \ldots, \mathbf{q}_j$ because the region sector encloses all the original vectors. For the above example of $\mathbf{q}'_1$, it can be seen that the region sector between $\angle_{\max}\mathbf{q}'_1 = \angle\mathbf{q}_1$ and $\angle_{\min}\mathbf{q}'_1 = \angle\mathbf{q}_3$ (the gray area in Figure 6 (b)) contains all the original vectors and that the angle between the region sector and $\mathbf{c}_1$ is equal to the angle between $\mathbf{c}_1$ and $\mathbf{q}_1$.

The above discussion focused on similarity between a vector and a region sector. Now, the concept is extended to define the similarity between two region sectors. Figure 7 illustrates possible three cases in comparing two region sectors associated with two resultant vectors $\mathbf{q}'_\nu$ and $\mathbf{c}'_\mu$ with respect to their relative positions.

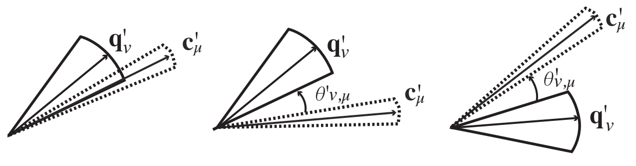


Fig. 7 Possible three cases in comparing two region sectors associated with $\mathbf{q}'_\nu$ and $\mathbf{c}'_\mu$.

When two region sectors overlap as in Figure 7 (a), two sets of the original vectors contained therein can be considered similar. Hence, the angle between the region sectors is defined as 0 and, consequently, their similarity becomes 1. When the region sectors are separated as in Figure 7 (b) and (c), their similarity is calculated based on their smallest angle $\theta'_{\nu,\mu}$ between them defined as in Equation (11).

$$\theta'_{\nu,\mu} = \begin{cases} \angle_{\min}\mathbf{q}'_\nu - \angle_{\max}\mathbf{c}'_\mu & \textbf{if } \angle_{\min}\mathbf{q}'_\nu > \angle_{\max}\mathbf{c}'_\mu \\ \angle_{\min}\mathbf{c}'_\mu - \angle_{\max}\mathbf{q}'_\nu & \textbf{if } \angle_{\min}\mathbf{c}'_\mu > \angle_{\max}\mathbf{q}'_\nu \\ 0 & \textbf{otherwise} \end{cases} \tag{11}$$

This angle is used to compute the similarity between the resultant vectors $\mathbf{q}'_\nu$ and $\mathbf{c}'_\mu$ as

$$\mathrm{sim}'(\mathbf{q}'_\nu, \mathbf{c}'_\mu) = \begin{cases} 0 & \textbf{if } \theta'_{\nu,\mu} > \pi/2 \\ \cos\theta'_{\nu,\mu} & \textbf{otherwise} \end{cases} \tag{12}$$

where the similarity is defined to be 0 for $\theta'_{\nu,\mu} > \pi/2$, following the definition of AMSS (see Equation (4)).

Based on the similarity function sim′ for resultant vectors, UB is defined as a recursive function as in Equation (13), which is slightly different from AMSS such that Equation (7) holds.

$$\begin{aligned} &\mathrm{UB}(Q'_\nu, C'_\mu) = \\ &\quad \max\{\mathrm{UB}(Q'_{\nu-1}, C'_{\mu-1}) + \delta_{\nu,\mu} \cdot \mathrm{sim}'(\mathbf{q}'_\nu, \mathbf{c}'_\mu), \\ &\qquad \mathrm{UB}(Q'_{\nu-1}, C'_\mu) + \upsilon_{\nu,\mu} \cdot \mathrm{sim}'(\mathbf{q}'_\nu, \mathbf{c}'_\mu), \\ &\qquad \mathrm{UB}(Q'_\nu, C'_{\mu-1}) + \eta_{\nu,\mu} \cdot \mathrm{sim}'(\mathbf{q}'_\nu, \mathbf{c}'_\mu)\} \end{aligned} \tag{13}$$

where $Q'_\nu = (\mathbf{q}'_1, \ldots, \mathbf{q}'_\nu)$ and $C'_\mu = (\mathbf{c}_1, \ldots, \mathbf{c}'_\mu)$ are subsequences of Q' and C', respectively, and $\delta_{\nu,\mu}$, $\upsilon_{\nu,\mu}$, and $\eta_{\nu,\mu}$ are weight functions of $\mathbf{q}'_\nu$ and $\mathbf{c}'_\mu$ defined below.

$$\begin{aligned} \delta_{\nu,\mu} &= 2\min(n(\mathbf{q}'_\nu), n(\mathbf{c}'_\mu)) + |n(\mathbf{q}'_\nu) - n(\mathbf{c}'_\mu)| \\ \upsilon_{\nu,\mu} &= 2\min(n(\mathbf{q}'_\nu) - 1, n(\mathbf{c}'_\mu)) + |(n(\mathbf{q}'_\nu) - 1) - n(\mathbf{c}'_\mu)| \\ \eta_{\nu,\mu} &= 2\min(n(\mathbf{q}'_\nu), n(\mathbf{c}'_\mu) - 1) + |n(\mathbf{q}'_\nu) - (n(\mathbf{c}'_\mu) - 1)| \end{aligned} \tag{14}$$

where $n(\mathbf{x})$ denotes the number of the original vectors from which the resultant vector $\mathbf{x}$ is created. Intuitively, these weights are the maximum number of similarity values accumulated to compute the AMSS between the original vector subsequences corresponding to $\mathbf{q}'_\nu$ and $\mathbf{c}'_\mu$.

Note that the vector composition method described above can be seen as a dimensionality reduction method for time series data since they both convert the original data into simpler representation. There has been much work in the area; some examples include discrete wavelet transform (DWT) [3], piecewise linear approximation (PLA) [24], symbolic aggregate approximation (SAX) [20], and discrete Fourier transformation (DFT) [25]. In concept, PLA is most similar to the vector composition method, approximating an original time series with a set of line segments. These dimensionality reduction methods might be used for upper bounding AMSS but a different upper bound function needs to be derived for the specific choice of a dimensionality reduction method to avoid false dismissal.

Section 5.3 will report how much the upper bound function can accelerate the computation by way of execution time.

4 Ensemble for Time Series Data

A potential drawback of AMSS also comes from its advantage to focus on shapes of data. AMSS is designed to recognize similarity between shapes with limited influence on spatial variation and not to be much affected by outliers. However, it is by nature vulnerable to short-term oscillations with low amplitude that do not reflect the characteristics of true data, since AMSS looks at every swing of data in computing similarity no matter how small the amplitude is. An example of such time series data is shown in Figure 8, where the true data

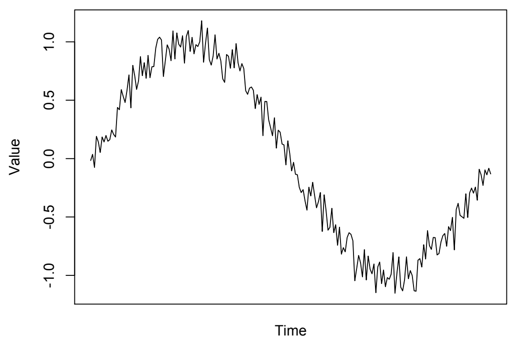


Fig. 8 An example of time series data having short term oscillations with low amplitude.

is given by a sine function but Gaussian noise is imposed on it. Due to the added noise, the time series data exhibit short-term oscillations.

A straightforward solution for the problem regarding short-term oscillations is to smooth out data before computing (dis)similarity. A downside of smoothing is that it may also reduce important, subtle behavior that characterizes some classes. Also, it may be rather harmful to data not involving short-term oscillations. Furthermore, there are many alternative smoothing algorithms, of which more appropriate ones for given data are not known. Therefore, it would be desirable if multiple smoothing algorithms (including not applying one) are considered in computing similarities for the given data.

Another solution is to employ multiple (dis)similarity measures, such as AMSS and DTW. The rationale behind is that, similarly to the idea of using multiple smoothing algorithms, each (dis)similarity measure has different properties and would have some types of data for which it works better. Section 5.4 will later show that different measures are good at different types of data and that their behavioral similarities can be quantified by correlation coefficients.

When AMSS is used for classification, both of the solutions can be implemented in the framework of ensemble learning [8], which is used to integrate multiple classifiers and generally produces superior performance to single classifiers.[4] The simplest form of ensemble learning trains each classifier separately and integrates the output of all classifiers by, for example, voting. However, this framework is not effective when the accuracy of a classifier varies on different data. More effective learning framework considers the interaction among multiple classifiers.

[4] Ensemble approaches have been applied to time-series classification in the past [14,26]. However, none has attempted to incorporate different smoothing techniques in an ensemble framework.

This study utilizes one of such frameworks, called *collaborative ensemble learning* [23], for its characteristics to effectively handle diverse data sets; it takes different types of classifiers and learns their weights through a simultaneous and iterative process for given data. Additionally, in classifying a new test instance, it dynamically gives appropriate weights to the predictions of the classifiers in accordance with their predicted class labels and corresponding confidence values learned in the training phase. These features become crucial when there are different target data sets with different characteristics, where no single classifier (smoothing algorithm or (dis)similarity measure in this case) is satisfactory for all the data sets. There are other ensemble approaches, such as AdaBoost [12] and negative correlation learning [21], which, however, have not been tested with different types of classifiers. The effect of using different smoothing algorithms or (dis)similarity measures in this study could be seen as using different types of classifiers.

In the collaborative ensemble learning, each classifier is learned on a training data set resampled for the classifier through an iterative process examining individual classifiers' performance on given training data. Figure 9 depicts the basic framework of collaborative ensemble learning involving three classifiers, $\mathcal{C}_1$, $\mathcal{C}_2$, and $\mathcal{C}_3$, each having different characteristics due to different training data. In this work, these classifiers are actually of the same type to be described later. Here, a calligraphic letter $\mathcal{C}$ is used to distinguish it from time series C. In essence, when $\mathcal{C}_i$ misclassifies a training instance, the instance will be fed to $\mathcal{C}_j$, $j \neq i$, for training at the next round, where $\mathcal{C}_j$ may correctly classify the instance. Owing to this collaboration, each classifier is trained more effectively for a particular portion of the training data that the classifier is good at classifying.

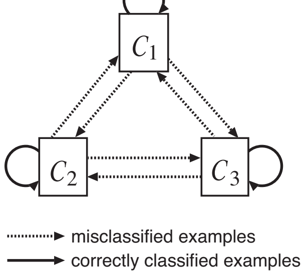


Fig. 9 Illustration of collaborative ensemble learning.

The formal procedure of the collaborative ensemble learning is presented below.

1. Set the number of rounds r to 1. Given a labeled data set D, set the weight $w_{i,j}$ of each labeled instance d_j, $j \in \{1, \ldots, |D|\}$, for each classifier $\mathcal{C}_i$, $i \in \{1, 2, \ldots, N\}$, to the same value, $1/|D|$, where N is the number of classifiers. The weight $w_{i,j}$ indicates the probability that d_j is drawn for $\mathcal{C}_i$ from the whole training data.
2. For each classifier $\mathcal{C}_i$, construct a training data set D_i^r of size $|D|$ by sampling D according to the weights $w_{i,j}$.
3. Train each classifier $\mathcal{C}_i$ using the training set D_i^r.
4. If r exceeds the predefined number of iterations provided by a user, then finish the learning process.
5. Classify the entire training data by each classifier $\mathcal{C}_i$.
6. For each instance d_j and each classifier $\mathcal{C}_i$, update the weights w as follows:
   - If a classifier $\mathcal{C}_i$ misclassified a training instance d_j, the weights $w_{k,j}$, $k \neq i$, for the other classifiers $\mathcal{C}_k$ are increased according to the error rate of the classifier $\mathcal{C}_i$ [13].
   - Likewise, if $\mathcal{C}_i$ correctly classified d_j, the weights $w_{k,j}$ are decreased according to the error rate of $\mathcal{C}_i$.
7. Increment r and repeat from the step 2 above.

Then, given an unseen test instance, all the $i \times r$ classifiers learned through the procedure are applied and their outputs are combined by considering their error rates and class separabilities so as to predict a single most likely class for the instance. See the original work [23] for more details.

Using this collaborative ensemble learning framework, different smoothing algorithms can be combined as follows to tackle the potential problem of AMSS caused by short-term oscillations. First, the data set D is preprocessed by N different smoothing algorithms $\mathcal{S}_i$, $i = 1, 2, \ldots, N$, separately, resulting in N different data sets D_i. Then, in the above procedure 2, samples (training data D_i^r) for each classifier $\mathcal{C}_i$ are drawn from the corresponding data set D_i. In this setting, different smoothing algorithms with the same type of classifier (e.g., 1-nearest neighbor (1NN)) are used to produce different outputs. That is, classifiers are the same but the training data for each classifier is differently smoothed.

5 Evaluation

5.1 Overview

This section investigates the effectiveness of the proposed similarity measure, AMSS through various experiments on the data sets obtained from the UCR Time Series Classification/Clustering page [18]. Using the UCR data sets, AMSS is compared with existing (dis)similarity measures in classification performance. Then, AMSS in the ensemble framework is evaluated. All the experiments reported in the evaluation were executed on a PC with a 2.4GHz Intel Core 2 Duo processor and 2GB of RAM. The source codes for the experiments were written in Java.

5.2 Initial results

The section studied the properties of AMSS in the context of classification in comparison with other (dis)similarity measures, specifically, Euclidean distance, DTW, DDTW, EDR,[5] and SpADe. A simple 1NN approach was adopted to allow fair comparison with less parameter to tune. By a 1NN classifier, the class label for a given sequence is predicted as the class label of its nearest neighbor in the training set. The classification performance is reported in error rate, which indicates a proportion of misclassified sequences in the test set. Table 1 summarizes the experimental results for each (dis)similarity measure on different data sets, where boldface indicates the best performance across different measures. The results for Euclidean distance and DTW were taken from the UCR Time Series Classification/Clustering page [18] and those for EDR and SpADe were from Chen et al.'s paper [6]. According to Keogh et al. [18], the results of DTW were obtained with the best width of Sakoe-Chiba band by a search over the training set. Lastly, the results for DDTW are based on our own implementation with no warping constraints. Appendix B will report the performance of AMSS in comparison with a wider set of (dis)similarity measures based on the methodology described in Ding et al. [9] .

As shown, DTW achieved the lowest error rates for six data sets; EDR, SpADe, and AMSS for five data sets; and Euclid and DDTW for two data sets. This result indicates that no single measure was effective for all the data sets with different characteristics. In the following, AMSS is compared with DTW as a representative, relatively effective data point-based measure to discuss the properties of AMSS.

First, let us discuss the data on which AMSS performed better. Figure 10, for example, shows an input sequence as solid line from the "OSU Leaf" data set and other sequences as dotted line found as the most similar

[5] EDR was chosen because it is reported to be the most accurate measure among the edit-distance family, including LCSS and ERP [9].

Table 1 Classification performance in error rates of different (dis)similarity measures using 1NN.

Data set	Euclid	DTW	DDTW	EDR	SpADe	AMSS
50Words	0.369	0.242	0.308	**0.198**	0.215	0.242
Adiac	0.389	0.391	0.414	0.384	**0.319**	0.345
Beef	0.467	0.467	0.467	—	—	**0.433**
CBF	0.148	**0.004**	0.408	0.011	0.020	0.522
Coffee	0.250	0.179	0.179	—	—	**0.143**
Gun-Point	0.087	0.087	0.007	0.020	0.007	**0.000**
ECG	0.120	0.120	0.170	**0.100**	0.130	0.170
Face (all)	0.286	0.192	**0.127**	0.194	0.214	0.265
Face (four)	0.216	0.114	0.375	**0.034**	**0.034**	0.261
Fish	0.217	0.160	0.103	0.080	**0.017**	0.040
Lightning-2	0.246	**0.131**	0.328	0.148	0.278	0.180
Lightning-7	0.425	**0.288**	0.425	0.301	0.315	0.301
OliveOil	**0.133**	0.167	0.200	—	—	0.200
OSU Leaf	0.483	0.384	0.120	0.215	0.132	**0.103**
Swedish Leaf	0.213	0.157	0.115	**0.096**	0.125	0.104
Syn. Control	0.120	**0.017**	0.433	0.040	0.080	0.523
Trace	0.240	0.010	**0.000**	0.040	**0.000**	**0.000**
Two Patterns	0.090	**0.002**	0.003	**0.002**	0.005	0.092
Wafer	**0.005**	**0.005**	0.022	0.007	0.012	0.011
Yoga	0.170	0.155	0.180	0.194	**0.123**	0.158

sequences (i.e., 1NNs) by AMSS (top figure) and DTW (bottom figure), where AMSS assigned the right class and DTW did not.

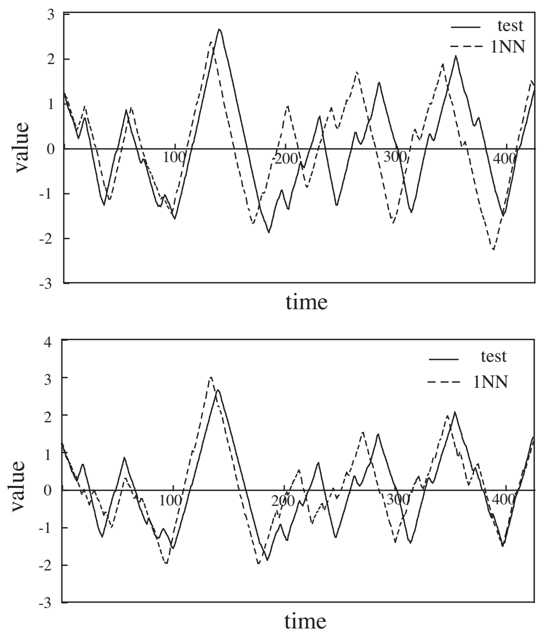


Fig. 10 1NNs shown as dotted lines found by AMSS (top) and DTW (bottom) for the same "OSU Leaf" test sequence shown by solid lines. AMSS's 1NN has the same class label as the test sequence and DTW's one does not.

On a cursory look, one may think that DTW found a good sequence more closely overlapping with the input sequence than that found by AMSS. However, carefully looking at the shapes of sequences, DTW's 1NN involves some fluctuation that does not have a good match in the input sequence. In contrast, AMSS's 1NN has relatively different amplitude and phase from those of the input sequence but more similar trajectory. This example indicates that AMSS is more robust for data involving amplitude and time shifting by focusing on vector directions or shapes.

On the other hand, Figure 11 shows the case from the CBF data where AMSS failed to find the correct 1NN and DTW succeeded. As can be seen, the CBF data demonstrate short-term oscillations. Because DTW looks at the locations of the data points, the influence of the small fluctuations was limited. On the other hand, AMSS focuses on directions of vectors and thus was heavily affected by each fluctuation, leading to the 1NN with a different (wrong) class from the test instance. Although the details are omitted here, one can make similar observations on the Synthetic Control data, where AMSS often fails due to short term oscillations.

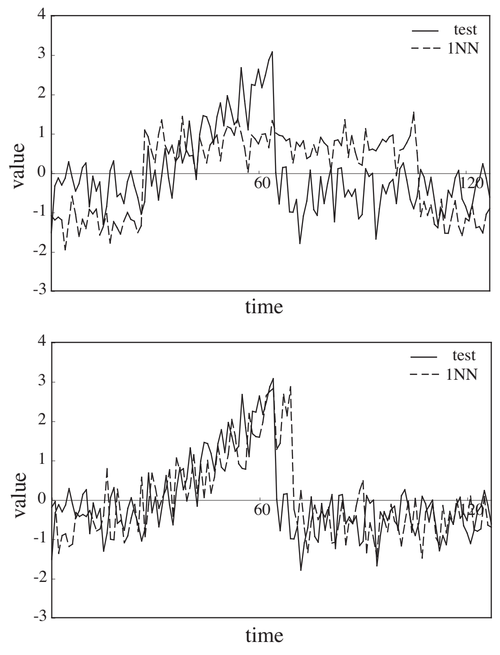


Fig. 11 1NNs shown as dotted lines found by AMSS (top) and DTW (bottom) for the same "CBF" test sequence shown as solid lines. DTW's 1NN has the same class label as the test sequence and AMSS's one does not.

Here, it is interesting to see that the other (partly) shape-based measures, DDTW and SpADe, exhibit somewhat similar behaviors to AMSS. Looking at the CBF and Synthetic Control data, DDTW also appears to suffer from short-term oscillations; the error rate was 0.408 and 0.433 as compared with 0.004 and 0.017 by DTW, respectively. Also, DDTW achieved good performance close to AMSS on the OSU Leaf data (the error rate was 0.116). On the other hand, SpADe showed moderately good performance on CBF, Synthetic Control, and OSU Leaf for which the error rates were found to be 0.020, 0.080, and 0.132, respectively. SpADe performed relatively well on those data sets due to the fact that it considers both data points and shapes in the form of their weighted sum. Note that, although SpADe could be fine-tuned for a given data set, those weights as well as many other parameters need to be manually set by trial and error. In contrast, AMSS and DDTW do not require such tweaking.

Although similar in the idea, an important distinction between AMSS and DDTW is that the latter uses the difference of the approximated local derivatives as the distance between two data points, which can be quite large when there is a sudden increase/decrease (e.g., outlier) in one time series. For AMSS, on the other hand, the influence of such an outlier would be limited due to the use of cosine similarity.

To contrast the difference between AMSS and DDTW, a time series Q was randomly selected from the test data for each UCR data set and a hypothetical sequence Q' was created by replacing a value of Q with an outlier, which was arbitrarily set to 3.0. Then, their (dis)similarity was computed by AMSS and DDTW, i.e., AMSS(Q, Q') and DDTW(Q, Q'), and self-(dis)similarity of the original data, i.e., AMSS(Q, Q) and DDTW(Q, Q).[6] Their absolute differences, Δ_{AMSS} and Δ_{DDTW}, defined respectively as $|\text{AMSS}(Q, Q') - \text{AMSS}(Q, Q)|$ and $|\text{DDTW}(Q, Q') - \text{DDTW}(Q, Q)|$, are attributed solely to the inserted outlier, and thus, indicate the susceptibility to it. To directly compare the differences, Δ_{AMSS} and Δ_{DDTW} were normalized by the within-class standard deviation SD (for the class to which Q belongs) of AMSS and DDTW, respectively. Figure 12 shows the bar plots of the normalized absolute difference for DDTW (left figure) and AMSS (right figure), where the figures on the plots are exact values for those exceeding the plot range. The order of the data sets is the same as Table 2. That is, the leftmost and rightmost are 50Words and Yoga, respectively. As can be seen, DDTW's absolute difference way exceeded $2 \times SD$ for all but two data sets, whereas AMSS's one is generally smaller. These results indicate that AMSS is more robust than DDTW to data containing outliers.

[6] DDTW(Q, Q) always becomes 0, whereas AMSS(Q, Q) linearly increases with the length of Q because AMSS is defined as a sum of similarities between matched subsequences.

5.3 Efficiency in Similarity Computation

Using the upper bound function UB(Q', C') in Equation (13), the execution time of classification was recorded for each UCR data set. Fig. 13 shows the bar plot summarizing how much speed-up was observed, where the results are shown in the same order as Table 1. The "Wafer" class, the tallest bar, most benefited from the upper bound; the execution time decreased by 94% as compared with AMSS with no upper bound (16.7 times faster). On the other hand, the execution time for classes with short-term oscillations, such as CBF and Synthetic Control, slightly increased by up to 2%. Overall, execution time decreased by 32% on average.

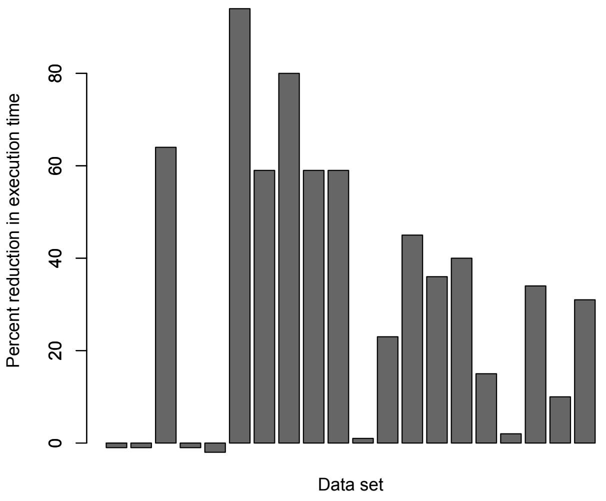


Fig. 13 Percent reduction in execution time for the UCR data sets.

It should be reminded that the upper bound function and the procedure to create resultant vectors described in Section 3.4 are solely for efficiency and do not affect the classification accuracy. In other words, whether they are used or not, the classification performance of AMSS in Table 1 does not change.

5.4 AMSS in ensemble

This section examined AMSS in the ensemble framework, which incorporates smoothing algorithms or different (dis)similarity measures for handling different types of data, for example, with/without short-term oscillations. First, for smoothing, two algorithms, moving average and low-path filter, were experimentally chosen. In essence, moving average takes an average of the

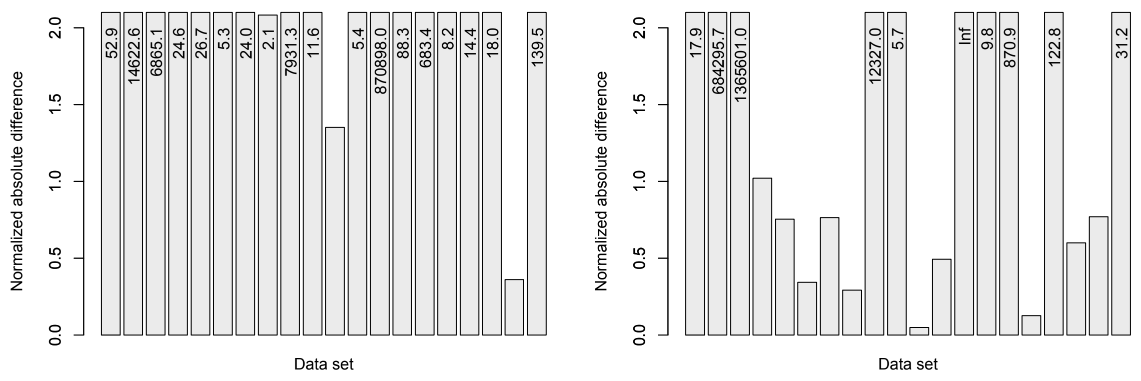


Fig. 12 Absolute differences, Δ_{DDTW} and Δ_{AMSS}, normalized by within-class standard deviation for DDTW (left) and AMSS (right).

data points in a sliding window to replace the value of the middle data point in the window, and low-path filter reduces the amplitude of fluctuations with high frequency.

Following the procedure outlined in Section 4, an experiment was carried out on the UCR data sets, where a window size of 7 was used for moving average and 10 elements with highest frequency after the discrete Fourier transform were discarded for low-path filter. These parameters, among the values tested, gave the lowest error rates for AMSS without ensemble on the training data. The tested values were all the odd numbers between 1–41 for the window size and all the numbers between 1–21 for low-path filter. Then, using the best parameters identified above, the appropriate number of iterations in ensemble learning was explored on the training data. There was no significant difference in performance among the tested values, 20, 30, 40, and 50, and thus the smallest (i.e., 20) was chosen considering the computation time. Note that, although not attempted here, they could be optimized for individual data sets to further improve the classification performance.

Because the collaborative ensemble learning involves a random factor in constructing training data sets, the experiment was repeated 10 times for each data set and the mean error rate was calculated. The results are presented in Table 2. The columns "none", "MA" (moving average), and "LPF" (low-path filter) show the classification error rates when they are used for smoothing, and the column "Ensemble" shows those for the collaborative ensemble learning. A plus symbol (+) indicates improvement from "none". Note that "none" is the same as AMSS without smoothing and the error rates are exactly the same as those presented in Table 1.

As can be seen, applying either moving average or low-path filter reduced the average error rates as compared to "none". Also, using the multiple classifiers in the ensemble framework further improved the performance. The improvement from the original AMSS ("none") to "Ensemble" was statistically significant at the 5% level based on a pairwise Wilcoxon signed rank tests ($p = 0.02$). Incidentally, the difference between the average error rate of 0.132 by the ensemble and that of 0.164 by DTW (calculated from Table 1) was not statistically significant ($p = 0.19$).

Then, another strategy to combine different (dis)similarity measures, instead of different smoothing algorithms, was tested using the ensemble framework. One can expect the greatest effect when diverse and high-performing measures are combined. To quantify the similar/different behaviors between these mea-

Table 2 Classification error rates for the individual classifiers and the collaborative ensemble method.

Data set	Individual classifier			Ensemble
	None	MA	LPF	
50Words	0.242	0.224$^+$	0.231$^+$	**0.219**$^+$
Adiac	0.345	0.358	0.307$^+$	**0.276**$^+$
Beef	**0.433**	0.467	0.467	0.450
CBF	0.522	0.160$^+$	**0.043**$^+$	0.046$^+$
Coffee	0.143	0.107$^+$	0.036$^+$	**0.014**$^+$
ECG	0.170	**0.120**$^+$	**0.120**$^+$	**0.120**$^+$
Face (all)	**0.265**	0.273	0.300	**0.265**
Face (four)	0.261	0.227	**0.125**$^+$	0.231$^+$
Fish	**0.040**	0.051	0.046	0.053
Gun-Point	**0.000**	0.013	0.013	0.012
Lightning-2	0.180	0.148$^+$	0.180$^+$	**0.146**$^+$
Lightning-7	0.301	0.247$^+$	0.384	**0.244**$^+$
OliveOil	0.200	**0.167**$^+$	**0.167**$^+$	0.180$^+$
OSU Leaf	**0.103**	0.149	0.136	0.130
Swedish Leaf	0.104	0.115	0.120	**0.076**$^+$
Syn. Control	0.523	0.180$^+$	0.073$^+$	**0.045**$^+$
Trace	**0.000**	**0.000**	**0.000**	**0.000**
Two Patterns	0.092	**0.000**$^+$	0.003$^+$	0.001$^+$
Wafer	0.011	0.020	0.010$^+$	**0.006**$^+$
Yoga	0.158	0.143$^+$	0.155$^+$	**0.132**$^+$
Average	0.205	0.158$^+$	0.146$^+$	**0.132**$^+$

Table 3 Pearson's correlation coefficients of error rates on the UCR data sets between different (dis)similarity measures.

	Euclid	DTW	DDTW	EDR	SpADe	AMSS
Euclid	1.000	0.875	0.368	0.807	0.689	0.169
DTW	0.875	1.000	0.292	0.902	0.736	0.070
DDTW	0.368	0.292	1.000	0.420	0.546	0.871
EDR	0.807	0.902	0.420	1.000	0.893	0.223
SpADe	0.689	0.736	0.546	0.893	1.000	0.337
AMSS	0.169	0.070	0.871	0.223	0.337	1.000

sures, Table 3 shows the correlation coefficients of the error rates on the UCR data sets. It can be observed that AMSS's correlations with other measures except for DDTW are generally low. This result suggests the distinctive properties of AMSS, coming from the unique representation of the data by vector sequences.

Considering the results in Table 3, AMSS and DTW were chosen as shape- and data point-based measures, respectively, and they were combined in the ensemble framework. Briefly, this combination yielded an average error rate of 0.142 on the UCR data sets. It is indeed lower than AMSS or DTW alone (their average error rates are 0.205 and 0.164, respectively) but slightly behind 0.132 where AMSS, MA, and LPF were combined. It may be possible that one could achieve greater performance by exhaustive search for the best combination, which is left for future work.

Lastly, whether or not other (dis)similarity measures could benefit from smoothing as well was investigated. Another experiment was carried out on the UCR data sets, where Euclidean distance and DTW were applied after smoothing by moving average. The window size was set to 7 as in the previous experiment. For Euclidean distance, the average error rate over the data sets slightly decreased (improved) from 0.234 to 0.211 and that of DTW without warping bands rather deteriorated from 0.164 to 0.226. This is due to the fact that these measures are based on individual values of data in contrast to AMSS which focuses on vectors spanning between two successive data points. AMSS is, by design, sensitive to fluctuations and can be improved by smoothing out short-term oscillations with long-term trend being left. On the other hand, those data point-based measures are by nature relatively robust to short-term oscillations with low amplitude as they do not greatly change data points. Thus, smoothing brings a less or even harmful effect. Taken together, these results indicate the potential of AMSS when short-term oscillations are properly handled within the ensemble framework.

6 Conclusion and future work

This paper introduced a shape-based similarity measure, AMSS, for time series data. Unlike other existing measures, AMSS first converts a sequence of data points to a sequence of vectors spanning between two consecutive data points, which enables AMSS to properly handle spatial variation. Also, by employing a variant of cosine similarity between two vectors, AMSS is less affected by outliers. The experiments proved that AMSS was especially effective for data involving amplitude and time shifting as designed. At the same time, the initial observations confirmed that AMSS was susceptible to short-term oscillation. To overcome the problem, a framework of ensemble learning was applied, where two different smoothing techniques, moving average and low-path filter, were incorporated. The experiments showed an improvement in error rates for most classes and that average error rate over the 20 UCR data sets decreased by around 36% and 20% from AMSS and DTW alone, respectively.

Future work would include studying the optimal data representation for AMSS. As discussed above, AMSS is more sensitive to short-term oscillations, which require preprocessing (e.g., smoothing). As demonstrated, ensemble approaches are a possible solution, which, however, are computationally costly. It would be desirable to treat all types of data with/without oscillations in the same single procedure through improved data representation.

Acknowledgements The authors would like to thank Takashi Okamura for his help with implementations and experiments.

References

1. J. Aßfalg, H.-P. Kriegel, P. Kröger, P. Kunath, A. Pryakhin, and M. Renz. Similarity search on time series based on threshold queries. In *Proc. of the 10th international Conference on Extending Database Technology*, pages 276–294, 2006.
2. D. J. Berndt and J. Clifford. Finding patterns in time series: A dynamic programming approach. In U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 229–248. AAAI, Menlo Park, CA, USA, 1996.
3. K.-P. Chan and A. W.-C. Fu. Efficient time series matching by wavelets. In *Proceedings of the 15th International Conference on Data Engineering*, pages 126–133, 1999.
4. L. Chen and R. Ng. On the marriage of lp-norms and edit distance. In *Proc. of the 30th International Conference on Very Large Data Bases*, pages 792–803, 2004.
5. L. Chen, M. T. Özsu, and V. Oria. Robust and fast similarity search for moving object trajectories. In *Proc.*

of the 2005 ACM SIGMOD International Conference on Management of Data, pages 491–502, 2005.
6. Y. Chen, M. A. Nascimento, B. C. Ooi, and A. K. H. Tung. Spade: On shape-based pattern detection in streaming time series. In *Proc. of the IEEE 23rd International Conference on Data Engineering*, pages 786–795, 2007.
7. G. Das, D. Gunopulos, and H. Mannila. Finding similar time series. In *Proc. of the First European Symposium on Principles of Data Mining and Knowledge Discovery*, pages 88–100, 1997.
8. T. G. Dietterich. Ensemble methods in machine learning. In *Proc. of the 1st International Workshop on Multiple Classifier Systems*, pages 1–15, 2000.
9. H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh. Querying and mining of time series data: Experimental comparison of representations and distance measure. *Proceedings of the VLDB Endowment*, 1:1542–1552, 2008.
10. C. Faloutsos, M. Ranganathan, and Y. Manolopoulos. Fast subsequence matching in time-series databases. In *Proc. of the 1994 ACM SIGMOD International Conference on Management of Data*, pages 419–429, 1994.
11. E. Frentzos, K. Gratsias, and Y. Theodoridis. Index-based most similar trajectory search. In *Proc. of the IEEE 23rd International Conference on Data Engineering*, pages 816–825, 2007.
12. Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of the Second European Conference on Computational Learning Theory*, pages 23–37, 1995.
13. Y. Freund and R. E. Schapire. A decision theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.
14. P. Geurts and L. Wehenkel. Segment and combine approach for non-parametric time-series classification. In *Proc. of the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 478–485, 2005.
15. D. Gunopulos and G. Das. Time series similarity measures. In *Tutorial notes of the sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 243–307, 2000.
16. F. Itakura. Minimum prediction residual principle applied to speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 23(1):67–72, 1975.
17. E. Keogh. Exact indexing of dynamic time warping. In *Proc. of the 28th International Conference on Very Large Data Bases*, pages 406–417, 2002.
18. E. Keogh, X. Xi, L. Wei, and C. A. Ratanamahatana. The UCR Time Series Classification/Clustering Homepage, 2006. Available at http://www.cs.ucr.edu/~eamonn/time_series_data/.
19. E. J. Keogh and M. J. Pazzani. Derivative dynamic time warping. In *Proc. of the 1st SIAM International Conference on Data Mining*, pages 1–11, 2001.
20. J. Lin, E. Keogh, S. Lonardi, and B. Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, pages 2–11, 2003.
21. Y. Liu and X. Yao. Ensemble learning via negative correlation. *Neural Network*, 12(10):1399–1404, 1999.
22. M. D. Morse and J. M. Patel. An efficient and accurate method for evaluating time series similarity. In *Proc. of the 2007 ACM SIGMOD International Conference on Management of Data*, pages 569–580, 2007.
23. H. Nomiya and K. Uehara. Multistrategical approach in visual learning. In *Proc. of the 8th Asian Conference on Computer Vision*, pages 502–511, 2007.
24. T. Pavlidis and S. L. Horowitz. Segmentation of plane curves. *IEEE Transactions on Computers*, 23:860–870, 1974.
25. D. Rafiei and A. Mendelzon. Similarity-based queries for time series data. In *Proceedings of the 1997 ACM SIGMOD international conference on Management of data*, pages 13–25, 1997.
26. J. J. Rodríguez and L. I. Kuncheva. Time series classification: Decision forests and SVM on interval and DTW features. In *Proc. of the Workshop and Challenge on Time Series Classification*, 2007.
27. H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(1):46–49, 1978.

Appendix

AMSS was compared with other (dis)similarity measures: Euclidean distance, DTW, DDTW, EDR, and SpADe as shown in Table 1. However, there is more comprehensive comparison reported by Ding et al. [9], who used k-fold cross validation for parameter tuning and evaluation for 13 different (dis)similarity measures. The value of k was individually set for each UCR data set (or class).

This paper did not take Ding et al.'s results for the comparison in Table 1 because of the significant amount of computation needed for the ensemble experiments with the way they tuned parameters through cross validation. For completeness, however, AMSS (without the ensemble framework) is compared with those reported by Ding et al. with their evaluation scheme. The results are summarized in Table 4, where boldface indicates the best performance (lowest error rates) across different measures.[7] "DTW (w)" and "LCSS (w)" indicate DTW and LCSS with warping constraint, respectively. All the results but DDTW were taken from Ding et al.'s paper [9], and DDTW's results are based on our own implementation with no warping window. For this experiments, the same number of splits k as Ding et al. was used so that all the results are directly comparable to theirs. Among the 20 data sets, AMSS performed the best for 10 data sets including one tie.

In terms of the number of data sets for which error rates are lowest, AMSS was found to be the best (dis)similarity measure in this particular setting. As discussed in Section 5.2, however, no single measure works for every type of data. For example, simple Euclidean distance performed better than all the other measures, including AMSS, for the "Beef" data set. More work is needed to understand the interaction between the properties of similarity measures and the characteristics of the target data types.

[7] Note that these results are different from Table 1 due to the different treatment of the training data.

Table 4 Classification performance in error rates of different (dis)similarity measures using 1NN.

Data set	Euclid	L_1-norm	L_∞-norm	DISSIM	TQuEST	DTW	DTW (w)	EDR	ERP	LCSS	LCSS (w)	Swale	SpADe	DDTW	AMSS
50Words	0.407	0.379	0.555	0.378	0.526	0.375	0.291	0.271	0.341	0.298	0.279	0.500	0.341	0.291	**0.187**
Adiac	0.464	0.495	0.428	0.497	0.718	0.465	0.446	0.457	0.436	0.434	0.418	0.961	0.438	0.380	**0.343**
Beef	**0.400**	0.550	0.583	0.550	0.683	0.433	0.583	**0.400**	0.567	0.402	0.517	0.617	0.500	0.585	0.680
CBF	0.087	0.041	0.534	0.049	0.171	0.003	0.006	0.013	**0.000**	0.017	0.015	0.040	0.044	0.377	0.526
Coffee	0.193	0.246	**0.087**	0.196	0.427	0.191	0.252	0.160	0.213	0.213	0.237	0.446	0.185	0.177	0.154
ECG200	0.162	0.182	0.175	0.160	0.266	0.221	0.153	0.211	0.213	0.171	**0.126**	0.539	0.256	0.167	0.147
FaceFour	0.149	0.144	0.421	0.172	0.144	0.064	0.164	0.045	0.042	0.144	0.046	0.368	0.250	0.048	**0.031**
FacesUCR	0.225	0.192	0.401	0.205	0.289	0.060	0.079	0.050	0.028	0.046	0.046	0.127	0.315	0.125	**0.017**
fish	0.319	0.293	0.314	0.311	0.496	0.329	0.261	0.107	0.216	0.067	0.160	0.857	0.150	0.090	**0.056**
Gun_Point	0.146	0.093	0.186	0.084	0.175	0.140	0.055	0.079	0.161	0.098	0.065	0.346	0.007	0.006	**0.002**
Lighting2	0.341	0.251	0.389	0.261	0.444	0.204	0.320	**0.088**	0.190	0.199	0.108	0.288	0.272	0.273	0.185
Lighting7	0.378	0.286	0.566	0.300	0.503	0.252	0.202	**0.093**	0.287	0.282	0.116	0.545	0.557	0.431	0.327
OliveOil	0.150	0.236	0.167	0.216	0.298	0.100	0.118	0.062	0.132	0.135	**0.055**	0.717	0.207	0.262	0.750
OSULeaf	0.448	0.488	0.520	0.474	0.571	0.401	0.424	0.115	0.365	0.359	0.281	0.405	0.212	0.082	**0.065**
Swedish Leaf	0.295	0.286	0.357	0.299	0.347	0.256	0.221	0.145	0.164	0.147	0.148	0.667	0.254	0.088	**0.079**
Syn. Control	0.143	0.146	0.227	0.158	0.640	0.019	**0.014**	0.118	0.035	0.060	0.075	0.146	0.150	0.430	0.515
Trace	0.368	0.279	0.445	0.286	0.158	0.016	0.075	0.150	0.084	0.118	0.142	0.273	**0.000**	**0.000**	**0.000**
Two Patterns	0.095	0.039	0.797	0.036	0.747	**0.000**	**0.000**	0.001	0.010	**0.000**	**0.000**	0.003	0.052	0.000	0.034
Wafer	0.005	0.004	0.021	0.005	0.014	0.015	0.005	**0.002**	0.006	0.004	0.004	0.106	0.018	0.010	0.003
Yoga	0.160	0.161	0.181	0.167	0.216	0.151	0.151	0.112	0.133	0.109	0.134	0.430	0.130	0.078	**0.056**