TS-MIoU: A Time Series Similarity Metric Without Mapping

Azim Ahmadzadeh[1]✉, Yang Chen[1], Krishna Rukmini Puthucode[1], Ruizhe Ma[2], and Rafal A. Angryk[1]

[1] Department of Computer Science, Georgia State University
[2] Department of Computer Science, University of Massachusetts Lowell
✉ aahmadzadeh1@gsu.edu

Abstract. Quantifying the similarity or distance between time series, processes, signals, and trajectories is a task-specific problem and remains a challenge for many applications. The simplest measure, meaning the Euclidean distance, is often dismissed because of its sensitivity to noise and the curse of dimensionality. Therefore, elastic mappings (such as DTW, LCSS, ED) are often utilized instead. However, these measures are not metric functions, and more importantly, they must deal with the challenges intrinsic to point-to-point mappings, such as pathological alignment. In this paper, we adopt an object-similarity measure, namely Multiscale Intersection over Union (MIoU), for measuring the distance/similarity between time series. We call the new measure TS-MIoU. Unlike the most popular time series similarity measures, TS-MIoU does not rely on a point-to-point mapping, and therefore, circumvents all respective challenges. We show that TS-MIoU is indeed a metric function, especially that it holds the triangle inequality axiom, and therefore can take advantage of indexing algorithms without a lower bounding. We further show that its sensitivity to noise is adjustable, which makes it a strong alternative to the Euclidean distance while not suffering from the curse of dimensionality. Our proof-of-concept experiments on over 100 UCR datasets show that TS-MIoU can fill the gap between the unforgiving strictness of the ℓ_p-norm measures, and the mapping challenges of elastic measures.

Keywords: time series · similarity · distance.

1 Introduction

Signals, processes, time series, and trajectories are data types which despite their differences have a lot in common. They all are ordered—and often equally spaced in time—values of a random variable recorded in time. With the advances in machine learning algorithms and computational power at our disposal, these high-dimensional data types have become ubiquitous. Since our primary focus in this study is on their spatiotemporal similarities, we use the name "time series" as an umbrella term for all such data types.

One of the primary challenges in dealing with these data types is in the way the similarity notion is defined. Similarity is a subjective concept, so much so that often different applications require very different or even contradictory criteria to define similarity. For example, two trajectories sampled from the Australian Sign Language dataset [15] may be considered 'similar' despite the absence of any spatial alignment, whereas in the Taxi Service Trajectory dataset [25], a significant spatial alignment is essential for trips to be considered 'similar'. Therefore, the *no-free-lunch* theorem applies; there is no general-purpose, universal similarity measure that outperforms all others in all applications. This subjectivity has given rise to the invention of an array of effective and elegant measures, each with its own strengths and shortcomings. Because of this diversity, a degree of expertise is almost always expected for the user to be able to appropriately utilize them and achieve the optimal gain.

In this paper, we entertain a new idea that claims, in the context of similarity measures, time series can be treated as *objects* with unique shapes and structures, as opposed to spatiotemporal data types. Time series have been treated as objects before (e.g., as fractals [27]), primarily to extract their complex features. But to the best of our knowledge, the similarity notion has almost always been tied to a mapping function of some sort, which requires time series to be considered what they actually are; sequences of points.

Although the mapping-based strategies are ideal for a large pool of time series applications, it should not be generalized as the *only* way of defining and quantifying their similarity. Borrowing from the computer vision domain, this is analogous to the pixel-to-pixel similarity measures (e.g., Mean Square Error), as opposed to other strategies such as a patch-to-patch comparison or Histogram of Oriented Gradients (HOG) [21]. We bring this up because it was shown that the pixel-to-pixel similarity measures, despite their popularity, are not the best choices for, for example, an objective similarity evaluation [26]. Compared to images, time series are much less intuitive data points, and therefore, we might not be able to always perceive the inadequacy of some measures' mapping strategy.

Our non-mapping perspective has two fundamental advantages. Firstly, it circumvents the real challenges caused by different point-to-point mapping strategies. Secondly, it builds (adjustable) resistance against noise. The latter will be more clear after we define the measure. Regarding the former, the mapping strategy generally divides the similarity measures into two main groups: those which do not allow local time shifting, such as the ℓ_p norm family, and those which do, such as the Edit Distance on Real Sequence (EDR) [6], Longest Common Subsequence (LCSS) [34], and Dynamic Time Warping (DTW) [4,30,17], and their many variants. The first group of measures are very sensitive to noise, and moreover, they require time series to be of the same length. Among the measures in the second group, one of the most prominent issues is the occurrence of pathological alignments, i.e., when a single point is mapped onto a large subsection of another time series—a known issue of DTW and some of its variants [30]. Several constraints have been proposed to control for such undesirable mappings, such as *windowing*, *slope weighting*, and *step patterns* (see [29] and references therein).

In addition, methods such as feature mapping/segmentation have also been proposed to avoid pathological warpings [22,13]. Another complication that comes with this approach lies in the mapping of some key points (e.g., peaks or dips of time series). In addition to the possibility of pathological mappings of these key points, the detection of peaks and dips is often a challenge of its own, as peaks and dips are also subjective and task-specific concepts [22]. There are also assumptions in the mapping functions that may or may not be desirable in some applications. For example, DTW is restricted by assumptions such as *continuity* and/or *monotonicity*. The continuity assumption forbids 'jumps', meaning that every point on each time series must be mapped onto at least one other point on the other time series. The monotonicity assumption forbids going back in time, i.e., connecting a point of one time series to a processed point in another time series. Because of these restrictions, an optimal mapping may exist well outside such a confined space. A good example in which discontinuity is allowed is LCSS, however, its mapping is still monotonic. A similarity measure that does not require a mapping is entirely free of such challenges—while of course subject to some other challenges.

2 Background

The literature on the similarity measures for time series is as rich and diverse as the time series application itself. A thorough review of these measures, even when limited to a particular application, requires a dedicated study on its own. Therefore, in the present work, which should not be seen as anything other than a proof of concept, we do not go beyond a quick review of the most popular similarity measures (in Section 2.1). Instead, we review a few other measures which, at some level, bear some resemblance to our proposed measure (in Section 2.2).

2.1 Popular Distance/Similarity Measures

The simplest approach for measuring the distance between two time series is by seeing them as high-dimensional data points in the Euclidean space, and measuring their distance with the Euclidean distance (a.k.a. ℓ_2 norm). This measure, as well as the Manhattan distance (ℓ_1 norm) and Chebyshev distance (ℓ_∞ norm), are special cases of a more generalized distance function called the Minkowski distance (ℓ_p norm). Euclidean distance, the most popular of the ℓ_p norm family, is very sensitive to noise. Meaning small variations on the time axis or any spatial misalignment may significantly impact the distance. Moreover, it cannot be used for time series of different lengths and sampling rates due to its static pairwise mapping between the time series elements. For applications such as GPS tracking, where the spherical coordinates system might be preferred over a Cartesian coordinate system, Haversine distance can be used instead. Haversine distance is the angular distance between two points on a surface of a sphere. Haversine distance is also sensitive to noise, but what makes these measures popular, in addition to their simplicity and cheap computation (i.e.,

a linear time complexity), is the fact that they are metric functions. A metric function holds the triangle inequality axiom, which makes it a natural choice for indexing and tree-based search algorithms (as discussed in Section 3.5).

The fact that ℓ_p norm does not allow local time shifting—it restricts the mapping of the i-th element of one time series to only the i-th element of the other time series—gave rise to a number of other distance/similarity measures of which we only review some of the most popular. Inspired by the Edit Distance (ED) measure used for string comparisons [19], the ED with Real Penalty (ERP) was introduced for quantifying the time series similarity with local time shifting [5]. Although ERP is a metric, it was shown that (like DTW) it is sensitive to noise [6]. As a remedy, a modified version of it, namely the ED on Real Sequence (EDR), was proposed [6]. EDR defines the distance between two time series in terms of the number of modifications (i.e., insertion, deletion, and replacement) one time series may need to change into the other. EDR reduces the impact of noise the same way LCSS does; by quantizing pairwise distances to either 0 or 1. This advantage comes at the cost of violating the triangle inequality which makes EDR a non-metric [5]. Among several ED-based measures, the Time Warp ED with Stiffness Adjustment (TWED) was tailored to (1) hold the triangle inequality while being an elastic metric and (2) provide a parameter to control the elasticity of the mapping function [24]. All these measures have a quadratic time complexity.

The LCSS measure can also be seen as a special case of ED. The key difference for LCSS is that unlike DTW and ED-based measures (which require all elements of time series to be matched), it allows partial comparisons, i.e., parts of time series can be left unmatched. This is advantageous because it allows tolerance of some noise. This unique feature, from a different angle, limits the application of LCSS, since the unmatched elements are entirely ignored and do not contribute to the final value of the distance. LCSS is not a metric as it violates the triangle inequality. It is also worth mentioning that LCSS, as well as DTW and ED-based measures, cannot be *directly* used for 2D time series (with time as the third dimension). Interested readers in multidimensional time series can read about some proposed approaches in [9].

DTW searches through a 2D space to find an optimal mapping between the two given time series and then defines the distance as the sum of the Euclidean distance between all matched elements. In principle, DTW requires quadratic computation time, it is not a metric [5], and it remains sensitive to noise. That said, DTW seems to have become the most popular elastic measure for time series data mining community thanks to the several lower bounding methods (including the lower bounding based on warping constraints, i.e., 4S), which significantly sped up its computation time [28].

2.2 Measures with Comparable Ideas

A grid-based approach for measuring the distance between two trajectories was introduced by Lin et al. [20]. The authors talked about the applications such as animal migration patterns and city traffic monitoring, in which the similarity

of interest is primarily determined by the trajectories' spatial patterns, and the temporal aspect (e.g., timestamp and velocity) is not seen as critical. Given a query trajectory Q and an arbitrary trajectory T, their method superimposes a grid over T and Q, and then computes the one-way distance (OWD) of Q from T. The OWD is computed by first identifying the so-called "local min points" on T (one local min point relative to each point on Q) and then summing up the Euclidean distances between each point on Q and its corresponding local min point on T. The main purpose of their grid representation is to build an efficient indexing method that speeds up the similarity search. The authors recommended building multiple grid representations of trajectories with different granularity levels to confine the search space. This is done by starting the search algorithm from the coarsest representation of trajectories and iteratively passing forward the k-most similar trajectories, as the granularity of the grid increases. This approach is similar to ours in that they both use a 2-dimensional grid-based segmentation. However, it is only in our approach that the hierarchical representation of time series directly contributes to the measure of similarity. In the method proposed by Lin et al., the hierarchical representation is part of their retrieval algorithm and not the similarity measure.

The Complexity-Invariant Distance (CID) is a method for adding a complexity sensitivity to distance measures [3]. CID has a somewhat similar motivation as ours; without taking into account this new invariance property, similarity search algorithms may not be able to differentiate between 'complex' and 'simple' time series because of their overall similarities, hence ignoring their difference in 'complexity'. Although the authors did not define 'complexity', they explained it intuitively, that the complexity of a time series is proportional to the total sum of its line segments' length. This is generally how the *correction factor* is computed. Their notion of complexity is identical to what is known in fractal geometry as *fractal dimension* [23]. Our measure of similarity takes full advantage of the definition of fractal dimension to account for time series' complexity (see Section 3).

The Hausdorff distance [2] is another measure that—if carefully examined—is somewhat similar to our approach. To quantify the distance between two time series (originally between two shapes), Hausdorff distance finds the smallest radius of the disk needed that if each point of either of the time series is replaced with that disk, the union of those disks contains all points of the other time series. This 'thickening' process is in principle similar to the change of resolution in our proposed measure, as we explain in Section 3.2. Hausdorff distance is a metric, but unlike our measure, it is sensitive to noise. This is because the computed distance is always determined by the single farthest point from the other time series, and therefore, a single outlier can heavily impact the distance.

Although not a distance measure, the indexing technique used in [17] bears some resemblance to our proposed measure, in that they both treat time series as shapes. The authors introduced LB_PAA—a modified version of the Piecewise Aggregate Approximation [35]—for reducing the dimensionality of time series (from n to 16) and therefore speeding up the indexing process for DTW. Instead

of comparing a candidate time series T to a query time series Q, they compare a 16-dimensional version of T, denoted as $\bar{T}$, to the 16-dimensional lower and upper bounds of Q. Then, when building the tree structure for indexing, they compute the distance of Q from a minimum bounding rectangle (MBR) of $\bar{T}$ instead of T itself. This is where a time series is treated as a shape and estimated by an MBR. Our proposed measure uses a similar spatial estimate of time series to compare them without a mapping.

3 Multiscale IoU (MIoU) for Time Series

Since we are borrowing a region-based similarity measure, namely MIoU [1], and re-purposing it for time series analysis, we first review the original idea, and then discuss the specific modifications needed for this adoption [3] [4].

3.1 MIoU Recap

Intuitively, MIoU [1] is the marriage of two concepts: *Intersection over Union* (IoU) [10] and the *fractal dimension* [23]. IoU (a.k.a. the Jaccard Index [14]) is a widely used object-similarity measure that quantifies the degree of which a ground-truth object is detected (i.e., intersection) relative to the area occupied by both of the ground-truth and detected objects (i.e., union). The fractal dimension was originally proposed to quantify the complexity of self-similar objects, called *fractals*. Among several methods that compute the fractal dimension the *box-counting* method utilizes grids of varying cell sizes in order to capture the complexity of fractals' geometry (see [33] and the references therein). Fig. 1 illustrates the general idea behind MIoU for two different proposed objects representing a solar filament.

The motivation behind introducing MIoU for measuring objects' similarity is to compensate for the limited sensitivity of IoU (as well as other area-based measures such as f_1 score, precision, and recall) to the fine details visible in objects' structures—a highly informative feature present in many scientific computer vision applications (see examples in [1]). MIoU achieves this through a multiscale approach: for noticeably misaligned objects, either spatially or structurally, MIoU is able to capture major misalignments early on, at lower resolution levels, whereas for well-aligned objects, MIoU can identify the subtle misalignments at higher resolution levels.

Formally, the MIoU measure is formulated using three functions. Let O denote a set of all valid objects (represented as binary masks of regions), and $\Delta \subset \mathbb{N}$ denote a finite set of box sizes. The first function, $s : O \times \Delta \longrightarrow O$, performs a grid-based segmentation on the object $o \in O$, with a given box size $\delta_i \in \Delta$, and downsamples the region. The second function, $|\cdot| : O \longrightarrow \mathbb{N}$, computes the area of a given region by counting the number of pixels (or grid

[3] MIoU repository: https://bitbucket.org/gsudmlab/multiscale_iou/
[4] TS-MIoU repository: https://bitbucket.org/gsudmlab/ts_miou_ecmlpkdd22/

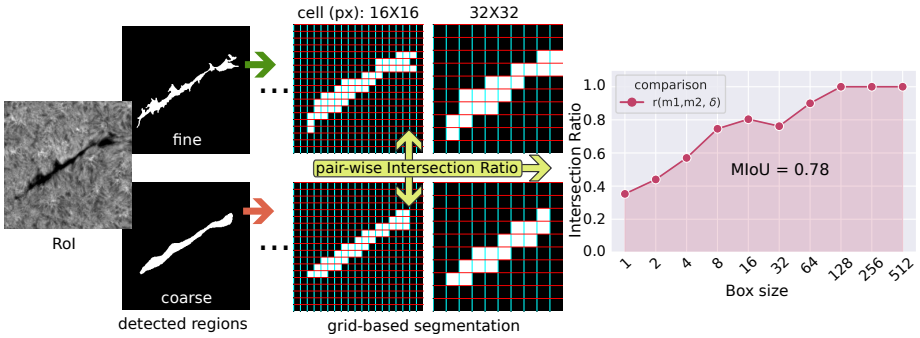

Fig. 1: The graphic reviews how MIoU measures the similarity between a fine (top) and coarse (bottom) region (each of size 512×512 pixels). The box sizes of the grid varies from 1 pixel (highest resolution) to 512 pixels (lowest resolution). The line plot shows the intersection ratios, and the area under its curve (0.78) which defines the similarity between two regions.

cells after it was downsampled by s) o's boundary spans over. The last function, $r : O^2 \times \Delta \longrightarrow [0, 1]$, called *intersection ratio*, computes the intersection between the boundaries of a region o and its estimate $\tilde{o}$, in terms of the number of $\delta_i \times \delta_i$ boxes they share, and normalizes it by the number of boxes o's boundary spans over. More precisely, $r(o, \tilde{o}, \delta_i) = \frac{|s(o,\delta_i) \cap s(\tilde{o},\delta_i)|}{|s(o,\delta_i)|}$.
Having r calculated for all different box sizes in Δ, MIoU is then computed by measuring the area under the curve formed by r. This curve is shown in Fig. 1, and can be formulated as follows: $\text{MIoU}(o, \tilde{o}) = \int_0^1 r(o, \tilde{o}, \delta)\, d\delta$, which is the total area of $|\Delta| - 1$ trapezoids. MIoU varies within the interval $[0, 1]$ if $d\delta = \frac{1}{|\Delta|-1}$. As a similarity measure, greater values indicate greater similarity, 1 means a perfect alignment, and no overlap is represented by 0. It is worth noting that although $1 - \text{MIoU}$ can be considered a distance (dissimilarity) measure, it is not a *metric*. We will discuss this shortly in Section 3.5.

3.2 TS-MIoU: MIoU for Time Series

We claim that MIoU is an effective measure for quantifying time series similarity. In many applications, the implicit or explicit definition of similarity for time series is indistinguishable from that for objects. In such cases, while we want similar time series to have similar shapes and patterns, we expect them to generally stay within a close distance along the time dimension, regardless of their individual sampling rate. Trajectory of moving objects [20] is just one of such cases. As we discuss in this section, with some minor changes and appropriate (task-specific) segmentation strategies, time series can be treated as objects and shapes. Needless to say that neither this nor any other realization of similarity can be used universally for all applications.

To justify our main modifications of MIoU, we first need to highlight the fact that MIoU is defined under a specific assumption; the given regions of interest are categorized into either *ground truth* (annotated by human) or *detected* (annotated by an algorithm) regions. This subtle assumption gives away a priori knowledge about the intended comparison, and to take advantage of that, when defining intersection ratio, the authors replaced the *area of the union*—which is in the definition of IoU—with the *area of the ground-truth*. In the absence of this a priori knowledge (e.g., in unsupervised approaches and information retrieval systems), it is highly advantageous for us to revert to the original definition of IoU. Not only is this supported by our empirical study, but also, and perhaps more importantly, it rewards us with the triangle inequality condition (see Section 3.5). The reverted definition of the intersection ratio that we used for TS-MIoU is given in Eq. 1.

$$r(T_1, T_2, (\delta_{x,i}, \delta_{y,i})) = \frac{|s(T_1, (\delta_{x,i}, \delta_{y,i})) \cap s(T_2, (\delta_{x,i}, \delta_{y,i}))|}{|s(T_1, (\delta_{x,i}, \delta_{y,i})) \cup s(T_2, (\delta_{x,i}, \delta_{y,i}))|} \tag{1}$$

Another minor change is that compared to the intersection ratio in MIoU, TS-MIoU's intersection ratio benefits from a tuple, i.e., $(\delta_{x,i}, \delta_{y,i})$, determining the box sizes, instead of a single value of δ. Consequently, TS-MIoU$(T_1, T_2) = \int_0^1 r(T_1, T_2, (\delta_x, \delta_y))\, d\delta_x \delta_y$. This allows us to build segmentations grids with non-squared cells in order to tackle the ill-definedness of the time series space. We further discuss these topics in Sections 3.3 and 3.4.

3.3 Ill-definedness of Space

TS-MIoU transforms time series into binary masks. This procedure requires quantisation (binning) on both of the dimensions. In doing so caution must be taken in using shape-based similarity metrics on time series. For a time series to be considered a 2-dimensional object with a well-defined geometry, both dimensions must be of the same unit. This incommensurability of the axes of time series plots makes the geometrical aspect ratio of this artificially-made space ill-defined. One direct consequence of this ill-definedness is the arbitrariness of measures such as the fractal dimension. Such measures depend on the arbitrariness of the aspect ratios of time series plots. In fact, time series' patterns and motifs might be partially or completely obscured by choosing an inappropriate aspect ratio of the axes in the binning process.

This is certainly a concern for the box-counting method incorporated in TS-MIoU. Theoretically, the $\delta \times \delta$ boxes used by the box-counting method do not represent any meaningful geometric area on time series space. However, similarity is a *relative* concept. For example, a retrieval algorithm looks for the most similar instances relative to a query instance. Therefore, this ill-defined space can still be explored as long as (1) the sides of the boxes used for segmentation can be adjusted independently to account for the different resolutions needed for each dimension of the time series space, and (2) the conditions determining the space remain constant for all comparisons. We elaborate in these conditions in the following sections.

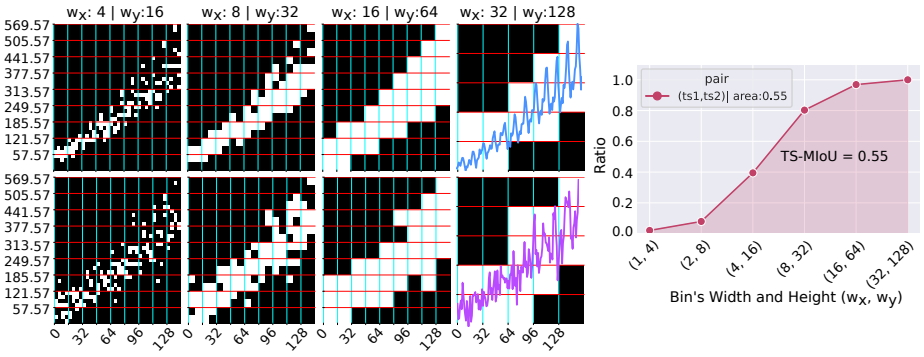

Fig. 2: Segmentation with proportional binning is illustrated for computing TS-MIoU on two time series (blue and purple). For visibility purposes, only four of the six representations of time series are shown, corresponding to the bin sizes in $\Delta_{x,y} = \{(1,4),(2,8),(4,16),(8,32),(16,64),(32,128)\}$.

3.4 Segmentation with Proportional Binning

As mentioned before, TS-MIoU treats time series like objects through the use of grid-based segmentation while remaining completely agnostic to the segmentation method used. The simplest approach for segmentation is to set an upper bound for the number of cells, k, and then carry out hierarchical segmentation with all integers from 1 to k. The upper bound can be axis-specific as well. A slightly more dynamic strategy is to define non-square boxes with widths and heights proportional to the ranges of values on the x and y axes, respectively. These boxes will then be used to form the segmentation grids representing varying resolutions. The latter is the strategy we use in our experiments, and we call it *segmentation with proportional binning*.

Suppose we have a dataset of N time series, $\mathcal{T} = \{T\}_{i=1}^{N}$, each of length n. Note that this assumption of fixed length is not a requirement for TS-MIoU as it does not rely on any types of mapping between time series. Let m and M denote the global minimum and maximum values in $\mathcal{T}$ (per variate, for multivariate time series), respectively. The width and height of the largest $(\delta_x \times \delta_y)$-box used for segmentation can then be determined by $\delta_x = \frac{n}{c_x}$ and $\delta_y = \frac{(M-m)}{c_y}$, respectively, where $c_x, c_y \in (0, 1]$ are user-defined parameters. Appropriate choices of c_x and c_y can guarantee that TS-MIoU does not overlook the interesting structures of the time series. Each of the upper bounds δ_x and δ_y determine the rest of the corresponding box sizes using a linear or logarithmic function, starting from 1, representing the original time series. In Section 4, we use powers of two, inspired by the original definition of fractal dimension [23]. An example of such a segmentation strategy is illustrated in Fig. 2.

In this paper, we will not address an optimum binning strategies as it heavily depends on the data, the time series' patterns, and structures. An efficient,

data-driven methodology for determining the optimal set $\Delta_{x,y}$ seeks a thorough investigation of its own, which belongs to our future work.

3.5 TS-MIoU as a Metric

MIoU's corresponding distance function, $d_{\text{MIoU}} = 1 - \text{MIoU}$, is not a metric. But we will show that $d_{\text{TS-MIoU}} = 1 - \text{TS-MIoU}$ satisfies all conditions of a metric function. Let us start by reviewing these conditions.

Given a distance function $d : \mathcal{X} \times \mathcal{X} \longrightarrow \mathbb{R}$ where $\mathcal{X}$ is the universe of all valid objects (time series in our case), d is called a *metric* if the following properties hold for all $x, y, z \in \mathcal{X}$: (1) *positiveness*, $d(x, y) \geq 0$; (2) *strict positiveness*, $x \neq y \Rightarrow d(x, y) > 0$; (3) *symmetry*, $d(x, y) = d(y, x)$; (4) *reflexivity*, $d(x, x) = 0$; and (5) *triangle inequality*, $d(x, z) \leq d(x, y) + d(y, z)$.

d_{MIoU} is by definition asymmetric. Moreover, it does not hold the triangle inequality. A simple counterexample can be made with any three objects A, B, and $|A| = |C|$, $A \cap C = \emptyset$, and $A \cup C = B$. This gives us $\text{MIoU}(A, C) = 0$, $\text{MIoU}(A, B) = 1$, and $\text{MIoU}(B, C) = 0.5$. Consequently, d_{MIoU} of those pairs are 1, 0, and 0.5, respectively. The triangle inequality $d_{\text{MIoU}}(A, C) \leq d_{\text{MIoU}}(A, B) + d_{\text{MIoU}}(B, C)$ yields the contradiction $1 \leq 0+0.5$. Therefore, d_{MIoU} is not a metric function.

It is easy to see that $d_{\text{TS-MIoU}}$ satisfies the positiveness condition as the area under the intersection-ratio curve is non-negative and less than or equal to 1. The strict positiveness also holds as long as $(1, 1) \in \Delta$. This guarantees the inclusion of the original time series in all comparisons, i.e., by computing $r(T_1, T_2, (1, 1))$; If $T_1 \neq T_2$, no matter how subtle their differences might be, $r(T_1, T_2, (1, 1)) > 0$, and hence $d_{\text{TS-MIoU}} > 0$. The generalization discussed in Section 3.2 propagates the symmetry property of IoU to $d_{\text{TS-MIoU}}$. The reflexivity condition is trivial. And lastly, $d_{\text{TS-MIoU}}$'s triangle inequality is inherited from the triangle inequality of d_{IoU} (see the proofs in [11,18]); since TS-MIoU is the sum of a finite number of IoUs, it therefore preserves the IoU's triangle inequality condition. Therefore, $d_{\text{TS-MIoU}}$ is indeed a metric function.

3.6 Time Complexity of TS-MIoU

A pseudo-code of TS-MIoU is given in Algorithm 1. Outside the loop, the `area` can be computed in linear time ($O(n)$, n being the number of iterations) using the Riemann sum. Inside the loop, the `segmentize` method is responsible for binning on the axes of a given time series. One possible implementation of this (e.g., see the `digitize` method in the *NumPy* package [12]) can be achieved through a binary search (with $O(log(n))$, where n is the number of bins). The logical `and` and `or` operations on 2D arrays require $O(r \times c)$, where r and c are the number of rows and columns of the binary matrices m_1 and m_2. This is the bottleneck of TS-MIoU's time complexity. If the proportional binning strategy discussed in Section 3.4 is adopted, the number of `and` (or `or`) operations at the i-th iteration will be $(r \cdot c)/4^i$. The overall complexity of the algorithm is then equivalent to the sum of the geometric series $\sum_{i=0}^{\infty} (r \cdot c)/4^i$ which converges to $4/3(r \cdot$

Algorithm 1: TS-MIoU Distance Metric

Input : T_1, T_2, Δ
output: TS-MIoU
function *ts_ miou(t_1: array, t_2: array, Δ: array) : float*
  ratios = [ ]
  for $\delta_x, \delta_y \in \Delta$ **:**
    m_1 = segmentize(T_1, δ_x, δ_y)
    m_2 = segmentize(T_2, δ_x, δ_y)
    union = $m_1 \vee m_2$ /* logical or */
    inters = $m_1 \wedge m_2$ /* logical and */
    ratios.append(sum(*inters*) / sum(*union*))
  tsmiou = 1 - area_under_curve(*ratios*)
  return *tsmiou*
end

c). Therefore, the current implementation of TS-MIoU has a quadratic time complexity. This is similar to the time complexity of the most popular similarity measures such as DTW, EDR, and LCSS without any additional constraints.

4 Experiments and Results

4.1 Experimental Settings

We conducted our experiments on 105 (out of 128) datasets of the UCR Time Series Archive [7]. We excluded the 11 datasets with varying lengths of time series because each similarity measure handles the varying length (if at all) differently, and this would have introduced a confounding factor to our experiments. We excluded another set of 12 datasets which contain very lengthy time series (> 900 elements). This decision was made primarily because for such long time series, a full-size comparison of time series has limited application in the real world, and failure or success of a similarity measure on such time series does not reveal much about its weaknesses or strengths.

As a proof of concept, we compared TS-MIoU with Euclidean distance (EuD), DTW, and LCSS. For the time series segmentation of TS-MIoU, we determined the bin sizes (i.e., δ_x and δ_y) as discussed in Section 3.4. Except for 16 datasets, for all other datasets, we defined the upper bounds of the bin sizes on the x and y axes by setting $c_x = 2.5$ and $c_y = 0.025$, respectively. This difference compensates for the difference in the range of the values on the two axes. For the 16 datasets in which the ratio $n/(M-m)$ was lower than 10, we noticed that setting $c_y = 0.025$ would make our segmentation method generate significantly imbalanced bin-size sets (i.e., $|\delta_y| - |\delta_x| \geq 4$). For these datasets, we increased c_y to 0.25 to reduce the difference. In all cases, we handled the imbalanced bin-size sets (of x and y axes) by clipping the head of the longer sets.

Regarding DTW, for the results shown in Figs. 3 and 4, we used the best values reported in the UCR official web site [8] (under the column `DTW (learned_w)`). For LCSS, we used the *tslearn* Python package [32] and we set the *maximum matching distance* threshold ϵ to one, i.e., the default value in the package.

4.2 Accuracy Gain of TS-MIoU

To fairly assess the overall quality of TS-MIoU, we use the so-called "Texas sharpshooter plot", as suggested in [7,3]. We compute the accuracy of the 1-NN classifier, using leave-one-out cross-validation, on the training set D^{train} of each UCR dataset, using an arbitrary distance function μ. We repeat the experiment but this time we classify time series of the respective test set D^{test}. The former gives us the *expected* performance, denoted as $\widehat{\text{acc}}_\mu$, and the latter gives us the *actual* performance, denoted as acc_μ. Using these quantities, we calculate the *expected gain* ($\widehat{g}_{\mu,\text{ref}}$) and the *actual gain* ($g_{\mu,\text{ref}}$) of using the distance function μ (i.e., TS-MIoU) over another distance function μ_{ref} (i.e., DTW or EuD), for each dataset. Precisely, $\widehat{g}_{\mu,\text{ref}} = \dfrac{\widehat{\text{acc}}_\mu(D^{\text{train}})}{\widehat{\text{acc}}_{\mu_{\text{ref}}}(D^{\text{train}})}$ and $g_{\mu,\text{ref}} = \dfrac{\text{acc}_\mu(D^{\text{test}})}{\text{acc}_{\mu_{\text{ref}}}(D^{\text{test}})}$.

The aggregated results illustrated in Fig. 3 show that TS-MIoU (with $\text{acc}_{\text{TS-MIoU}} = 0.72 \pm 0.19$, $\widehat{\text{acc}}_{\text{TS-MIoU}} = 0.73 \pm 0.20$) can indeed make 1-NN classifier to achieve a performance similar to that of DTW (with $\text{acc}_{\text{DTW}} = 0.76 \pm 0.19$, $\widehat{\text{acc}}_{\text{DTW}} = 0.75 \pm 0.21$) and EuD (with $\text{acc}_{\text{EuD}} = 0.71 \pm 0.20$, $\widehat{\text{acc}}_{\text{EuD}} = 0.73 \pm 0.22$). Note that in this comparison, DTW's parameter, the *warping window size*, was already optimized for each dataset, whereas for TS-MIoU a generic set of bin sizes were used. 1-NN with LCSS performed significantly worse than the others, which shows its sensitive dependence on its parameter ϵ. Because of LCSS's significantly lower accuracy values which resulted in outside-the-range accuracy gain of TS-MIoU, we had to remove its corresponding points from the Texas sharpshooter plot for better visibility.

There are four partitions in Fig. 4, namely true positive (TP), true negative (TN), false positive (FP), and false negative (FN). The TP region represents datasets on which TS-MIoU claimed to make improvements, and it did. In the TN region, TS-MIoU made no such claims and no improvements were made either. In the FN region, TS-MIoU improved 1-NN's performance despite making no claim about it. The worst cases lie in the FP region, where TS-MIoU falsely claimed to achieve an improvement. Overall, TS-MIoU wins on 42 datasets (41% of all datasets) over EuD, on 87 datasets (83%) over LCSS, and on 14 datasets (13%) over DTW with learned window sizes. Although these numbers paint a convincing picture of TS-MIoU's effectiveness in many applications, this is not the most informative way to analyze the outcome; even a handful of datasets could be enough to show the unique value of a measure.

Looking at the distribution of the red and blue points in these regions, we are interested in a few different angles. One is the magnitude of accuracy gain 1-NN benefits from. By recognizing all the points in the TP region, which are not close to the center, it is evident that the improvement is not just marginal.

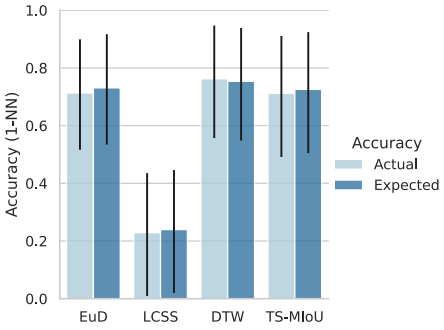

Fig. 3: The average expected and actual accuracy values of 1-NN on the UCR datasets, using four different distance functions.

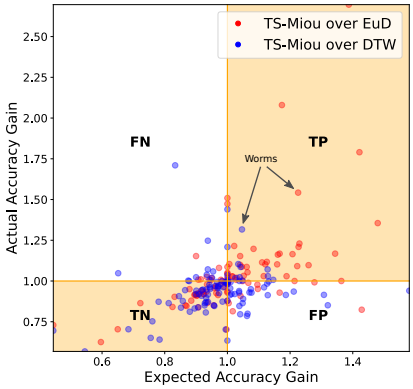

Fig. 4: The Texas sharpshooter plot showing the accuracy gain of 1-NN on UCR datasets, using TS-MIoU as the distance function over Euclidean distance and DTW.

For example see the `Worms` dataset (marked in Fig. 4) on which TS-MIoU improved 1-NN's performance significantly compared to both EuD and DTW, in both the expected and actual cases. This shows TS-MIoU can in fact make a significant contribution to the similarity search and retrieval applications. Another interesting angle to consider is the reason as to why TS-MIoU significantly under-performs in several cases, precisely in 51% of datasets compared to DTW and 40% compared to EuD. Our analysis reveals that in most of these cases our generic binning strategy resulted in too large or too small bin sizes which could not capture the discriminatory characteristics of time series. For example, time series in `TwoPatterns` dataset have two distinct patterns; a white noise pattern and a clear min-max binary pattern. Based on the class labels, it seems that only the second pattern has discriminatory power. DTW did an excellent job on this dataset ($\text{acc}_{\text{DTW}} \simeq 1.00$) which makes it perhaps the most difficult dataset to compete against DTW on. This nearly perfect performance is owed to DTW's learned window size and its success in the mapping of the binary peaks and dips. The added distance corresponding to the (correct or incorrect) mapping of the noisy patterns is almost constant across all pairwise comparisons and does not have a significant impact on the 1-NN's performance. Regarding TS-MIoU, our generic binning strategy on the y-axis returns the bin size set $\{0.01, 0.02, 0.04, 0.08, 0.16, 0.32\}$ (since we set c_y to 0.025). These disproportionately small bin sizes results in $\text{acc}_{\text{TS-MIoU}} = 0.57$. Upon a quick grid-search on c_y, we found that changing the bin sizes to a much smaller but more effective set, i.e., $\{0.5, 1.0\}$ (by setting c_y to 1.25), significantly boosts the performance of 1-NN to 0.94. It might be interesting to note that the range of the noisy pattern of `TwoPatterns`'s time series lies almost always within the range of 0.5 to

1.0, the bin sizes corresponding to $c_y = 1.25$. This example is a strong evidence supporting the power and flexibility of TS-MIoU when appropriate bin sizes are used.

5 Discussion, Conclusion, and Future Work

We introduced a similarity measure for time series, called *Time Series Multiscale Intersection over Union* (TS-MIoU). The novelty of TS-MIoU lies in the fact that unlike most of the other similarity measures, it does not require a point-to-point mapping of time series. We discussed that this approach circumvents many challenges such as pathological warping. TS-MIoU, however, is not intended to be used for capturing non-co-occurring trends between time series. DTW measure (with a large window size) and LCSS are more suitable for these applications where the temporal alignment is of little or no importance. We also showed that TS-MIoU is a metric function which makes it an excellent choice for indexing algorithms.

In this proof-of-concept study, we only focused on the feasibility and applicability of TS-MIoU in time series similarity analysis. One avenue that we wish to explore is its pruning and further indexing potential. Sequential scan algorithms can be used to illustrate the pruning power, independent of the actual indexing structure [5]. The next natural step is to investigate the indexing of the TS-MIoU algorithm in detail. Other than taking advantage of the triangle inequality condition, another method for indexing is to apply the GEMINI framework where lower-bounding is used. Since the TS-MIoU is based on box-counting, we could adapt it for lower-bounding, similar to the Piecewise Aggregate Approximation used in LB_Keogh [16]. Another avenue of our future work is to adapt TS-MIoU for streaming time series data. We will investigate the various dynamic normalization methods, such as the ones based on z-normalization and min-max normalization (e.g., [31] among many). Real-time processing combined with early abandoning can greatly enhance TS-MIoU in real-world applications.

Acknowledgment This project has been supported in part by funding from CISE, MPS and GEO Directorates under NSF award #1931555, and by funding from the LWS Program, under NASA award #80NSSC20K1352.

References

1. Ahmadzadeh, A., Kempton, D.J., Chen, Y., Angryk, R.A.: Multiscale iou: A metric for evaluation of salient object detection with fine structures. In: 2021 IEEE International Conference on Image Processing (ICIP). pp. 684–688 (2021). https://doi.org/10.1109/ICIP42928.2021.9506337
2. Alt, H.: The computational geometry of comparing shapes. In: Efficient Algorithms, Essays Dedicated to Kurt Mehlhorn on the Occasion of His 60th Birthday. Lecture Notes in Computer Science, vol. 5760, pp. 235–248. Springer (2009). https://doi.org/10.1007/978-3-642-03456-5_16, https://doi.org/10.1007/978-3-642-03456-5_16

3. Batista, G.E.A.P.A., Wang, X., Keogh, E.J.: A complexity-invariant distance measure for time series. In: Proceedings of the Eleventh SIAM International Conference on Data Mining, SDM 2011, April 28-30, 2011, Mesa, Arizona, USA. pp. 699–710. SIAM / Omnipress (2011). https://doi.org/10.1137/1.9781611972818.60, https://doi.org/10.1137/1.9781611972818.60
4. Berndt, D.J., Clifford, J.: Using dynamic time warping to find patterns in time series. In: Knowledge Discovery in Databases: Papers from the 1994 AAAI Workshop, Seattle, Washington, USA, July 1994. Technical Report WS-94-03. pp. 359–370. AAAI Press (1994)
5. Chen, L., Ng, R.T.: On the marriage of lp-norms and edit distance. In: (e)Proceedings of the Thirtieth International Conference on Very Large Data Bases, VLDB 2004, Toronto, Canada, August 31 - September 3 2004. pp. 792–803. Morgan Kaufmann (2004). https://doi.org/10.1016/B978-012088469-8.50070-X, http://www.vldb.org/conf/2004/RS21P2.PDF
6. Chen, L., Özsu, M.T., Oria, V.: Robust and fast similarity search for moving object trajectories. In: Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data. p. 491–502. SIGMOD '05, Association for Computing Machinery, New York, NY, USA (2005). https://doi.org/10.1145/1066157.1066213, https://doi.org/10.1145/1066157.1066213
7. Dau, H.A., Bagnall, A.J., Kamgar, K., Yeh, C.M., Zhu, Y., Gharghabi, S., Ratanamahatana, C.A., Keogh, E.J.: The UCR time series archive. IEEE CAA J. Autom. Sinica **6**(6), 1293–1305 (2019). https://doi.org/10.1109/jas.2019.1911747, https://doi.org/10.1109/jas.2019.1911747
8. Dau, H.A., Keogh, E., Kamgar, K., Yeh, C.C.M., Zhu, Y., Gharghabi, S., Ratanamahatana, C.A., Yanping, Hu, B., Begum, N., Bagnall, A., Mueen, A., Batista, G.: The ucr time series classification archive (October 2018), https://www.cs.ucr.edu/~eamonn/time_series_data_2018/
9. Frentzos, E., Gratsias, K., Theodoridis, Y.: Index-based most similar trajectory search. In: Proceedings of the 23rd International Conference on Data Engineering, ICDE 2007, The Marmara Hotel, Istanbul, Turkey, April 15-20, 2007. pp. 816–825. IEEE Computer Society (2007). https://doi.org/10.1109/ICDE.2007.367927, https://doi.org/10.1109/ICDE.2007.367927
10. Ge, F., Wang, S., Liu, T.: Image-segmentation evaluation from the perspective of salient object extraction. In: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06). vol. 1, pp. 1146–1153. IEEE (2006)
11. Gilbert, G.: Distance between sets. Nature **239**, 174–174 (1972)
12. Harris, C.R., Millman, K.J., et al.: Array programming with NumPy. Nature **585**(7825), 357–362 (Sep 2020). https://doi.org/10.1038/s41586-020-2649-2, https://doi.org/10.1038/s41586-020-2649-2
13. Hong, J.Y., Park, S.H., Baek, J.G.: Ssdtw: Shape segment dynamic time warping. Expert Systems with Applications **150**, 113291 (2020)
14. Jaccard, P.: The distribution of the flora in the alpine zone. New Phytologist **11**, 37–50 (1912). https://doi.org/10.1111/j.1469-8137.1912.tb05611.x, https://doi.org/10.1111/j.1469-8137.1912.tb05611.x
15. Kadous, W., Taylor, S.A.: Grasp: Recognition of australian sign language using instrumented gloves (1995)
16. Keogh, E.J.: Exact indexing of dynamic time warping. In: Proceedings of 28th International Conference on Very Large Data Bases, VLDB 2002, Hong Kong, August 20-23, 2002. pp. 406–417. Morgan Kaufmann (2002). https://doi.org/10.1016/B978-155860869-6/50043-3, http://www.vldb.org/conf/2002/S12P01.pdf

17. Keogh, E.J., Ratanamahatana, C.A.: Exact indexing of dynamic time warping. Knowl. Inf. Syst. **7**(3), 358–386 (2005). https://doi.org/10.1007/s10115-004-0154-9, https://doi.org/10.1007/s10115-004-0154-9
18. Kosub, S.: A note on the triangle inequality for the jaccard distance. Pattern Recognit. Lett. **120**, 36–38 (2019). https://doi.org/10.1016/j.patrec.2018.12.007, https://doi.org/10.1016/j.patrec.2018.12.007
19. Levenshtein, V.I., et al.: Binary codes capable of correcting deletions, insertions, and reversals. In: Soviet physics doklady. vol. 10, pp. 707–710. Soviet Union (1966)
20. Lin, B., Su, J.: Shapes based trajectory queries for moving objects. In: 13th ACM International Workshop on Geographic Information Systems, ACM-GIS 2005, November 4-5, 2005, Bremen, Germany, Proceedings. pp. 21–30. ACM (2005). https://doi.org/10.1145/1097064.1097069, https://doi.org/10.1145/1097064.1097069
21. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. **60**(2), 91–110 (2004). https://doi.org/10.1023/B:VISI.0000029664.99615.94, https://doi.org/10.1023/B:VISI.0000029664.99615.94
22. Ma, R., Ahmadzadeh, A., Boubrahimi, S.F., Angryk, R.A.: Segmentation of time series in improving dynamic time warping. In: 2018 IEEE International Conference on Big Data (Big Data). pp. 3756–3761 (2018). https://doi.org/10.1109/BigData.2018.8622554
23. Mandelbrot, B.B.: The fractal geometry of nature. 1982. San Francisco, CA (1982)
24. Marteau, P.: Time warp edit distance with stiffness adjustment for time series matching. IEEE Trans. Pattern Anal. Mach. Intell. **31**(2), 306–318 (2009). https://doi.org/10.1109/TPAMI.2008.76, https://doi.org/10.1109/TPAMI.2008.76
25. Moreira-Matias, L., Gama, J., Ferreira, M., Mendes-Moreira, J., Damas, L.: Predicting taxi-passenger demand using streaming data. IEEE Trans. Intell. Transp. Syst. **14**(3), 1393–1402 (2013). https://doi.org/10.1109/TITS.2013.2262376, https://doi.org/10.1109/TITS.2013.2262376
26. Mrak, S., et al.: Reliability of objective picture quality measures. Journal of Electrical Engineering **55**(1-2), 3–10 (2004)
27. Pilgrim, I., Taylor, R.P.: Fractal analysis of time-series data sets: Methods and challenges (2019). https://doi.org/10.5772/intechopen.81958, https://doi.org/10.5772/intechopen.81958
28. Ratanamahatana, C.A., Keogh, E.: Everything you know about dynamic time warping is wrong. In: Third workshop on mining temporal and sequential data. vol. 32. Citeseer (2004)
29. Ratanamahatana, C.A., Keogh, E.J.: Making time-series classification more accurate using learned constraints pp. 11–22 (2004). https://doi.org/10.1137/1.9781611972740.2, https://doi.org/10.1137/1.9781611972740.2
30. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. IEEE Transactions on Acoustics, Speech, and Signal Processing **26**(1), 43–49 (1978). https://doi.org/10.1109/TASSP.1978.1163055
31. Sukhanov, S., Wu, R., Debes, C., Zoubir, A.M.: Dynamic pattern matching with multiple queries on large scale data streams. Signal Processing **171**, 107402 (2020)
32. Tavenard, R., Faouzi, J., Vandewiele, G., Divo, F., Androz, G., Holtz, C., Payne, M., Yurchak, R., Rußwurm, M., Kolar, K., Woods, E.: Tslearn, a machine learning toolkit for time series data. J. Mach. Learn. Res. **21**, 118:1–118:6 (2020)
33. Theiler, J.: Estimating fractal dimension. Journal of The Optical Society of America A-optics Image Science and Vision **7**(6), 1055–1073 (1990). https://doi.org/10.1364/JOSAA.7.001055, https://doi.org/10.1364/JOSAA.7.001055

34. Vlachos, M., Kollios, G., Gunopulos, D.: Discovering similar multidimensional trajectories. In: Proceedings 18th international conference on data engineering. pp. 673–684. IEEE (2002)
35. Yi, B., Faloutsos, C.: Fast time sequence indexing for arbitrary lp norms pp. 385–394 (2000), http://www.vldb.org/conf/2000/P385.pdf