IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 26, 2020, accepted December 7, 2020, date of publication December 10, 2020,
date of current version December 24, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3043839

Fast Constrained Dynamic Time Warping for Similarity Measure of Time Series Data

WONYOUNG CHOI[1], (Graduate Student Member, IEEE), JAECHAN CHO[1], (Graduate Student Member, IEEE), SEONGJOO LEE[2], (Senior Member, IEEE), AND YUNHO JUNG[1], (Senior Member, IEEE)

[1]School of Electronics and Information Engineering, Korea Aerospace University, Goyang-si 10540, South Korea
[2]Department of Information and Communication Engineering, Sejong University, Seoul 05006, South Korea

Corresponding author: Yunho Jung (yjung@kau.ac.kr)

This work was supported by the Institute of Information and Communications Technology Planning and Evaluation (IITP) Grant through Ministry of Science and ICT (MSIT), Korean Government, under Grant 2017-0-00528 and Grant 2020-0-00201.

ABSTRACT In this paper, we propose an efficient algorithm for reducing the computational complexity of dynamic time warping (DTW) for obtaining similarity measures between time series. The DTW technique exhibits superior classification accuracy compared to other algorithms but has a limitation of high computational complexity. To reduce the computational complexity of standard DTW, constrained DTW and fast DTW techniques have been proposed. The constrained DTW technique reduces the computational complexity of standard DTW by only considering limited alignments and prevent excessive alignments between two time series, which can reduce the overall classification accuracy. However, since the searching window for limited alignments is fixed, the computational complexity is still high when the length of time series is long. In contrast, fast DTW has a lower classification accuracy than the constrained DTW technique. However, fast DTW estimates the optimal alignment while considering only the alignments within an adaptive window; as two time series increase in length, the fast DTW technique more strongly reduces the computational complexity. Therefore, we propose a fast constrained DTW approach that applies the optimal alignment estimation of fast DTW within the limited alignments of constrained DTW. As the proposed fast constrained DTW operates within a fixed window area of the constrained DTW, which prevents excessive alignments, it has a classification accuracy similar to that of the constrained DTW. Also, in the fast constrained DTW, when the length of the time series is long, a low computational complexity is maintained by the influence of the adaptive window of the fast DTW. Experimental results on 19 UCR time series datasets show that the proposed fast constrained DTW method achieves computational complexity reductions of approximately 52.2% and 22.3% compared to the existing fast DTW and constrained DTW, while maintaining almost the same classification accuracy as the constrained DTW.

INDEX TERMS Computational complexity, dynamic time warping, similarity measure, time series data.

I. INTRODUCTION

Time series data mining is utilized in numerous applications such as clustering, classification [1]–[3] fault detection [4], [5], pattern recognition [6], and prediction [7]. In these applications, measuring the similarity between two time series is a frequent and important task [8]. Multilayer perceptrons (MLPs) [9], convolutional neural networks (CNNs) [10], hidden Markov model (HMM) [11], recurrent neural networks (RNNs) [12], and dynamic time warping (DTW) [38] have been proposed for similarity measure of time series. In time series mining, MLPs and CNNs show high accuracy and have been used in some applications [13], [14], but they require a large number of training data for good performance and have a complex network including various learning parameters [15]. Also, it cannot be applied to time series datasets with non-uniform lengths [16]. HMM has few parameters and is suitable for time series datasets with few training data [17]. But, HMM is complex and has poor performance [18], [19]. As RNNs and more specifically long short-term memory RNNs (LSTM) [20] can store internal states within cyclic recurrent nodes to use temporal information,

The associate editor coordinating the review of this manuscript and approving it for publication was Keli Xiao.

it has been successful in traffic prediction [21], speech recognition [22], [23], human trajectory prediction [24] and natural language processing [25]. But, RNN has very different structural properties and is hard to train and time-consuming [26]. On the other hand, DTW can show high accuracy even with a small train dataset, has a simple structure, is not dependent on parameters, and has a relatively low computational complexity compared to other algorithms [27]. Accordingly, DTW is most frequently used in time series data mining.

To accurately measure the similarity between two time series, the time series must be aligned on the time axis. Dynamic time warping (DTW) measures the similarity by warping two time series on the time axis to find the optimal alignment among all possible alignments [28]. Accordingly, DTW has been widely used in many fields such as gesture, image, and speech recognition [29]–[32], intrusion detection [33], financial analysis [34], biometrics [35], and medical diagnosis [36]. However, standard DTW, which considers all possible alignments, requires a quadratic time complexity with the lengths of two time series [43].

To overcome the high complexity limitation of standard DTW, two major approaches have been reported. The indexing [37] technique reduces the calling times of standard DTW algorithm, while constrained DTW [38]–[40] and data abstraction [41]–[43] techniques reduce the calculations of standard DTW. By using a lower bounding function, the indexing technique reduces the calling times of standard DTW algorithm and performs DTW operation only for the remaining time series. Therefore, it has a low time complexity.

Two algorithms have been developed as constrained DTW techniques that consider only limited alignments. The Sakoe–Chiba DTW (SC-DTW) [38] performs equal limited alignments on all data points, while incremental DTW (I-DTW) [40] performs more alignments on recent data points than that on previous data points. These constrained DTW techniques can reduce the number of alignments to be considered and prevent the pathological alignment problem, which can reduce the overall classification accuracy due to excessive alignment between time series. Therefore, the constrained DTW technique has a higher classification accuracy than the standard DTW, which suffers from the pathological alignment problem. However, with these constrained DTW techniques, the searching window for limited alignments is fixed. In addition, the calculations in the constrained DTW techniques increase quadratically with the length of the time series. Accordingly, high computational complexity is required for the constrained DTW techniques, when the length of time series is long [43].

The data abstraction technique decreases the calculations by using a data representation that has a reduced dimensionality of time series. Piecewise aggregate DTW [41] reduces the dimensionality by taking the average of equally sized segments of the time series, which also causes the loss of some important features [44]. In contrast, blocked DTW [42] reduces the dimensionality by finding points with consecutive values in the time series and then reducing them. However, as the level of dimensionality decreases, the similarity measure results become increasingly inaccurate [45]. In the fast DTW technique [43], the resolution of two time series is hierarchically reduced to create multiple resolutions; then, the optimal alignment is identified from the lowest layer and used to estimate the approximate optimal alignment of the next higher layer. In the resolutions of fast DTW, only the alignments near the estimated optimal alignment from the previous lower layer are considered. This approach utilizes an adaptive window for the alignments and reduces the number of alignments to be considered. The fast DTW algorithm, which has a linear computational complexity, more accurately finds the optimal alignment as the value of the *radius* parameter increases, but this leads to an increase in the calculations. Accordingly, the calculations of the fast DTW decrease only when the increase in calculations due to the data length is greater than the increase in calculations due to the *radius* parameter. Moreover, the fast DTW suffers from the pathological alignment problem and thus has a lower classification accuracy than the constrained DTW.

Motivated by these observations, we propose a fast constrained DTW that applies the optimal alignment estimation technique of the fast DTW algorithm within the constraints of the constrained DTW technique. In this manner, as the proposed fast constrained DTW operates within a fixed window area of the constrained DTW, which prevents pathological alignment problem, it has a classification accuracy similar to that of the constrained DTW. Also, when the increase in calculations due to the *radius* parameter is greater than the increase in calculations due to the data length, the computational complexity of the proposed fast constrained DTW, which operates within a fixed window area of the constrained DTW, is lower than or similar to that of the constrained DTW. In the opposite situation, the computational complexity of the proposed fast constrained DTW, which applies the optimal alignment estimation technique of the fast DTW, is lower than or similar to that of the fast DTW. The remainder of this paper is organized as follows. Section II describes related work on standard DTW, constrained DTW, and fast DTW. In Section III, we propose the fast constrained DTW approach. Section IV presents experimental results comparing the proposed fast constrained DTW with constrained DTW and fast DTW. Lastly, conclusions are given in Section V.

II. RELATED WORK

A. DYNAMIC TIME WARPING

As shown in Fig. 1, the DTW technique measures the similarity between two time series by warping them in a nonlinear fashion. The DTW method shows superior performance over the Euclidean distance (ED) approach, which measures the similarity at the exact same position on the time axis.

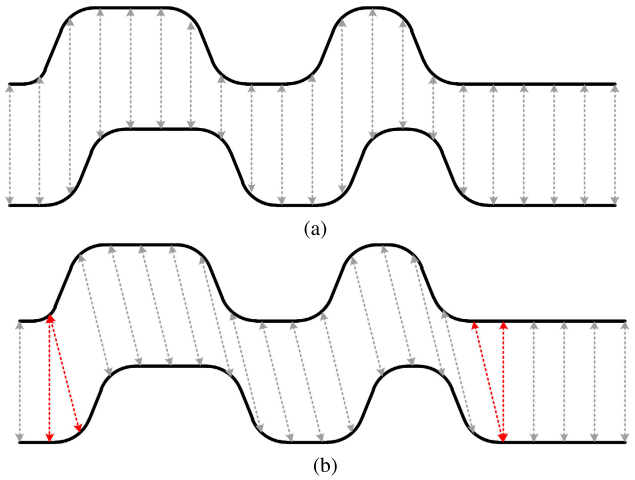

FIGURE 1. Alignment of two time series (the arrows represent aligned points). (a) Euclidean distance, (b) DTW distance.

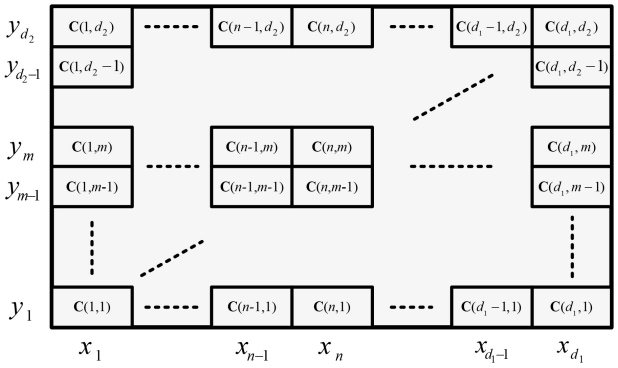

FIGURE 2. Cost matrix.

DTW finds the optimal alignment to accurately measure the similarity between two time series. To find the optimal alignment, one must first calculate all distances between two points in the two time series. In addition, the distances are accumulated while satisfying the conditions for finding the optimal alignment, and the similarity is then measured by using the accumulated distances. The distances between two points in the two time series and their accumulation are expressed as a cost matrix and an accumulated cost matrix, respectively. The cost matrix $\mathbf{C}(n,m) \in R^{d_1 \times d_2}$, $1 \le n \le d_1$, $1 \le m \le d_2$ for the distances between two time series $\mathbf{x}$ and $\mathbf{y}$ is given as follows:

$$\begin{aligned} x &= (x_1, x_2, \ldots, x_{d_1-1}, x_{d_1}), \\ y &= (y_1, y_2, \ldots, y_{d_2-1}, y_{d_2}) \qquad (1) \\ C(n,m) &= (x_n - y_m)^2 \qquad (2) \end{aligned}$$

Here, d_1 and d_2 are the lengths of time series $\mathbf{x}$ and $\mathbf{y}$, respectively, and the cost matrix shown in Fig. 2 is used to calculate the accumulated cost matrix.

In calculating the accumulated cost matrix, the optimal warping path $\mathbf{p}$, a set of index points that directly influence the optimal alignment, is defined as follows:

$$p = (p_1, p_2, \ldots, p_l, \ldots, p_{L-1}, p_L) \qquad (3)$$

$$p_l = (n_l, m_l) \in C(n,m) \quad \text{for } l \in [1, L] \qquad (4)$$

where L is the length of the optimal warping path. The optimal warping path of the accumulated cost matrix is subject to a boundary condition, monotonicity condition, and step size condition.

1) Boundary condition
The starting and ending points of the optimal warping path are as follows:

$$p_1 = (1,1) \text{ and } p_L = (d_1, d_2) \qquad (5)$$

2) Monotonicity condition
The subsequent index value of the optimal warping path must be greater than or equal to the current index value.

$$\begin{aligned} &n_1 \le n_2 \le \cdots \le n_{L-1} \le n_L \text{ and} \\ &m_1 \le m_2 \le \cdots \le m_{L-1} \le m_L \qquad (6) \end{aligned}$$

3) Step size condition
The difference between neighboring values in the optimal warping path has a step size defined as follows:

$$p_{l+1} - p_l \in \{(1,0), (0,1), (1,1)\}, \quad \text{for } l \in [1 : L-1] \qquad (7)$$

Accumulated cost matrix $\mathbf{A}(n,m) \in R^{d_1 \times d_2}$, $1 \le n \le d_1$, $1 \le m \le d_2$ is calculated using the following formula:

$$A(n,m) = \begin{cases} C(n,m) & \text{if } n = 1 \text{ and } m = 1 \\ C(n,m) + A(n-1,m) & \text{if } n \ge 2 \text{ and } m = 1 \\ C(n,m) + A(n,m-1) & \text{if } n = 1 \text{ and } m \ge 2 \\ C(n,m) + \begin{cases} A(n-1,m-1) \\ A(n-1,m) \\ A(n,m-1) \end{cases} & \text{if } n \ge 2 \text{ and } m \ge 2 \end{cases} \qquad (8)$$

After the accumulated cost matrix has been calculated, the optimal warping path can be calculated by following the smallest value of elements with the step size of the current index from $\mathbf{A}(d_1, d_2)$ to $\mathbf{A}(1,1)$, as shown in Fig. 3. In this manner, DTW can measure the similarity between two time series, and the DTW distance, which represents the similarity, is expressed as

$$DTW(x,y) = A(d_1, d_2) \qquad (9)$$

B. CONSTRAINED DTW

The red portions in Fig. 4 show the computational components of the ED method and standard DTW. The computational complexity of the standard DTW is $O(N^2)$ for two time series of length N, which is much higher than the $O(N)$

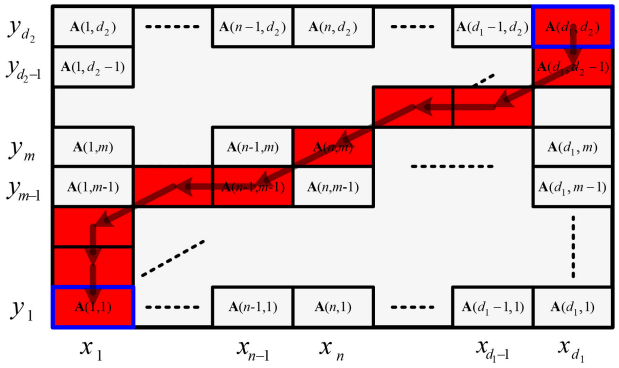

FIGURE 3. Accumulated cost matrix with an optimal warping path.

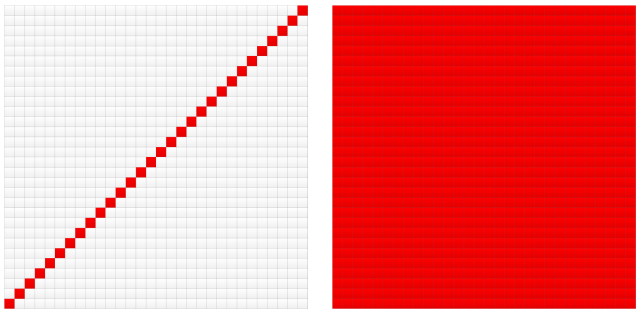

FIGURE 4. A comparison of computational complexity. (a) ED method, (b) DTW distance.

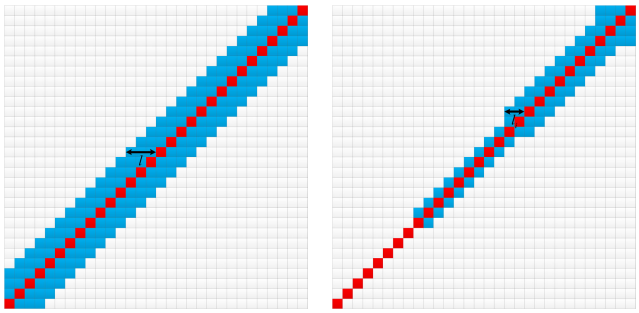

FIGURE 5. Accumulated cost matrix of constrained DTW. (a) SC-DTW, (b) I-DTW.

obtained for ED. However, it is not necessary to consider all alignments to perform DTW.

Fig. 5 shows the accumulated cost matrix for the most representative SC-DTW and for a recently published I-DTW based on the constrained DTW. The red portions indicate index points that match on the time axis between the two time series, and the blue portions indicate the window area; here, only the alignments of a specific area are considered, rather than all alignments. By considering only the alignments included in this window area, the constrained DTW technique can reduce the computational complexity and effectively prevent the pathological alignment problem, in which the classification accuracy is reduced due to excessive alignments far from the red portions.

The window of the SC-DTW has the same length along the horizontal and vertical axes. In this case, the window length is determined by the window percentage value r', $0 \leq r' \leq 1$. The window length l can be expressed as

$$l = [r' \times \min(d_1, d_2)] \tag{10}$$

Here, d_1 and d_2 refer to the lengths of each time series. The window percentage value determines the number of searching cells of the accumulated cost matrix, which affects the computational complexity. Also, if the window percentage value is too large, the accuracy may be reduced due to a pathological alignment problem. Conversely, if the window percentage value is too small, the accuracy may be reduced because the optimal warping path cannot be accurately found. Therefore, the window percentage value must be set appropriately. In Fig. 5(a), the most common window percentage value, $r' = 0.1$ [46], is used, and calculations can be reduced by approximately 80% compared to standard DTW.

For I-DTW, in which recent data points are more important than earlier data points, the window length is gradually increased as the index values of the two time series increase, as shown in Fig. 5(b). The window length l_i of I-DTW is as follows:

$$l_i = [r' \times \min(n_i, m_i)] \tag{11}$$

The lengths of each time series are n_i and m_i for the corresponding index i.

When the length of the two time series is N, SC-DTW and I-DTW require $N(2N \cdot r' + 1)$ and $N(N \cdot r' + 1)$ calculations, respectively; thus, their computational complexity is lower than that of the standard DTW, which has N^2 calculations. However, the constrained DTW uses a fixed window and exhibits a quadratic computational complexity; consequently, for long time series, the computational complexity is still very high.

C. FAST DTW

Fast DTW, which is a representative data abstraction algorithm, improves the computational complexity by using multiple resolutions. As shown in Fig. 6, fast DTW generates multiple resolutions and considers alignments within dark and light gray window area, which changes adaptively to estimate optimal alignments. This fast DTW can be implemented through the following sequence.

1) Coarsening
   Create hierarchical multiple resolutions by repeatedly averaging two adjacent points in the time series.
2) Projection
   Find the optimal warping path from the lower resolution and use it to estimate the approximate optimal warping path of the next higher resolution.
3) Refinement
   Refine the optimal warping path of the current resolution by considering both the approximate optimal

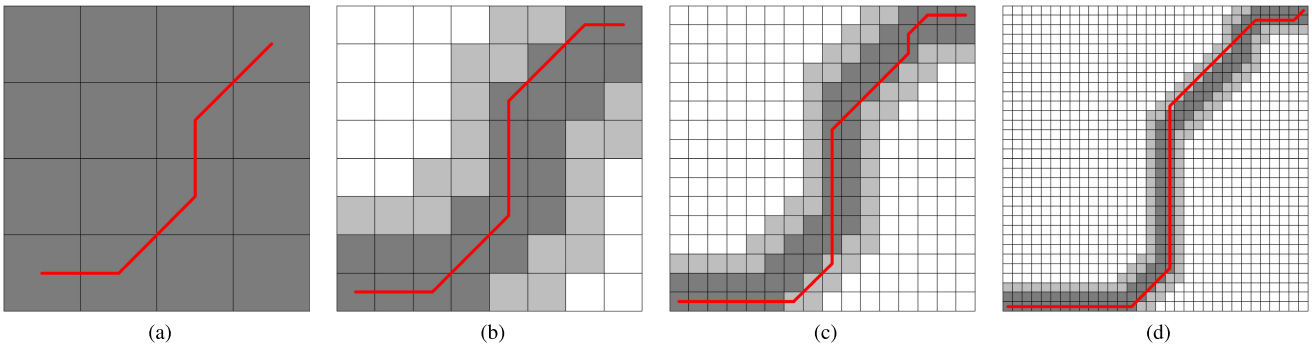

FIGURE 6. Accumulated cost matrix of four resolutions for the fast DTW algorithm: (a) the lowest resolution with a data length reduced by 1/8 compared to the original, (b) the resolution with a data length reduced by 1/4, (c) the resolution with a data length reduced by 1/2, (d) the original resolution.

warping path estimated from the previous lower resolution and additional surrounding regions using the *radius* parameter.

Here, the *radius* parameter determines additional regions around the estimated optimal warping path. As the *radius* parameter value increases, the optimal warping path can be found more accurately, but this increases the number of searching cells. Accordingly, fast DTW is more efficient for applications with long time series [43]. Fig. 6 illustrates the implementation process of the fast DTW algorithm. Three additional resolutions are created from the original resolution, which has a data length of 32 points on the horizontal and vertical axes. In the lowest resolution, the optimal warping path obtained by considering all alignments is shown as the red line in Fig. 6(a). This path is used to estimate the approximate optimal warping path in Fig. 6(b), which has a resolution with a data length of 1/4 compared to the original resolution. In the resolution with a data length of 1/4, the estimated optimal warping path and the additional components within the *radius* parameter are shown as dark gray and light gray, respectively. The optimal warping path, which is shown as the red line in Fig. 6(b), can be refined by performing alignments within this window area. In Fig. 6, the *radius* (r) parameter is set to 1. The projection and refinement processes are repeated until the optimal warping path and DTW distance in Fig. 6(d), which has the original resolution, are calculated through Fig. 6(c), which has a resolution with a data length of 1/2 compared to the original.

The outputs of the DTW algorithm are the DTW distance and optimal warping path. Assuming that only the DTW distance is calculated without finding the optimal warping path, the computational complexity of the fast DTW and standard DTW is determined as follows. For the fast DTW, calculations were performed for a total of 369 cells of 16, 44, 97, and 212 for the accumulated cost matrix from the lowest resolution to the original resolution. A total of $2 \times (16 + 8 + 4) = 56$ calculations were performed to create multiple resolutions with data lengths of 16, 8, and 4 points from the original resolution. Lastly, the fast DTW performed 5, 11, 24, and 49 calculations, each corresponding to the

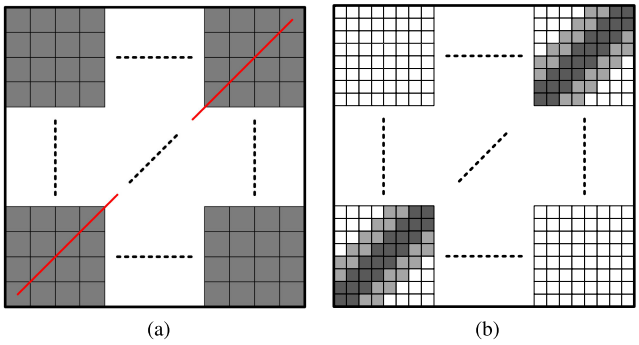

FIGURE 7. Accumulated cost matrix of two resolutions for the fast DTW algorithm in the worst-case scenario.

length of the optimal warping paths from the lowest resolution to the original resolution indicated by the red lines in Fig. 6. Because we have assumed that only the DTW distance is calculated, calculations to find the optimal warping path of the original resolution must be excluded; thus, $5 + 11 + 24 = 40$ calculations were performed, with a total of $369 + 56 + 40 = 465$ calculations for the fast DTW. If the standard DTW calculates only the DTW distance, it performs $32 \times 32 = 1,024$ calculations. Therefore, the fast DTW can reduce the calculations by approximately 45.41% compared to the standard DTW.

The fast DTW requires calculations for three components: accumulated cost matrix, multi-resolution, and optimal warping path. The calculations of the fast DTW can be theoretically calculated as follows. To simplify the calculations in this analysis, the lengths of the two time series are assumed to be N, and all analyses are performed for the worst-case scenario. Fig. 7(a) presents the accumulated cost matrix for the resolution whose data length is $N/2$, and the accumulated cost matrix for the original resolution with a data length of N is shown in Fig. 7(b). As indicated by the red line in Fig. 7(a), when the optimal warping path lies in the regions in which the index points of the two time series match in the lower resolution,the alignments within the dark and light window regions in Fig. 7(b) are performed for the next higher

resolution. In this case, we have the worst-case scenario for performing calculations for the largest number of cells. The number of cells to be calculated in Fig. 7(b) is as follows. The dark gray regions in Fig. 7(b) estimated by the optimal warping path in Fig. 7(a) include 3 cells in a line, and a total of $3N$ cells are formed by multiplying the data length N by the 3 cells. In addition, the light gray regions within the *radius* parameter, which is set to 1, include $4r$ cells in a line and a total of $4Nr$ cells. Accordingly, the total number of cells to be calculated for the original resolution whose data length is N is as follows:

$$3N + 4Nr = N(4r + 3) \tag{12}$$

The data length of each resolution is as follows:

$$N, \frac{N}{2}, \frac{N}{4}, \frac{N}{8}, \cdots \tag{13}$$

The sum of the number of cells to be calculated in the accumulated cost matrix of all resolutions is as follows:

$$N(4r + 3) + \frac{N}{2}(4r + 3) + \frac{N}{4}(4r + 3) + \frac{N}{8}(4r + 3) + \cdots \tag{14}$$

Based on the infinite geometrical series, (14) is summarized as follows:

$$\begin{aligned} &N(4r + 3)(1 + \frac{1}{2} + \frac{1}{2^2} + \frac{1}{2^3} + \cdots) \\ &= N(4r + 3)(1/(1 - \frac{1}{2})) \\ &= 2N(4r + 3) \end{aligned} \tag{15}$$

To create two time series with length $\frac{N}{2}$ that form the previous lower resolution from two time series of length N that form the original resolution, $\frac{N}{2}$ calculations are required for each time series. Accordingly, the calculations required to form all resolutions are as follows:

$$\begin{aligned} &2(\frac{N}{2} + \frac{N}{4} + \frac{N}{8} + \frac{N}{16} + \cdots) \\ &= 2N(\frac{1}{2} + \frac{1}{2^2} + \frac{1}{2^3} + \frac{1}{2^4} + \cdots) \\ &= 2N(\frac{1}{2}/(1 - \frac{1}{2})) \\ &= 2N \end{aligned} \tag{16}$$

Lastly, the calculations required to find the optimal warping path are $2N$, which is the sum of the lengths of two time series in the worst-case scenario. The calculations required to find the optimal warping paths of all resolutions are as follows:

$$\begin{aligned} &2N + N + \frac{N}{2} + \frac{N}{4} + \cdots \\ &= 2N(1 + \frac{1}{2} + \frac{1}{2^2} + \frac{1}{2^3} + \cdots) \\ &= 2N(1/(1 - \frac{1}{2})) \\ &= 4N \end{aligned} \tag{17}$$

The sum of (15), (16), and (17) represents the computational complexity of fast DTW.

$$\textit{Fast DTW computational complexity} = N(8r + 12) \tag{18}$$

Based on this theoretical analysis of computational complexity, in the worst-case situation, the fast DTW has a linear computational complexity based on the data length N, which can greatly reduce the calculations if the length of time series is sufficiently long compared to $8r + 12$. However, when the data length is smaller than $8r + 12$, the fast DTW is unlikely to achieve a reduction in calculations. In addition, fast DTW suffers from the pathological alignment problem and thus has a lower classification accuracy than the constrained DTW.

III. PROPOSED FAST CONSTRAINED DTW

The constrained DTW reduces calculations by performing alignments only within a limited window area with a specific shape and can increase the classification accuracy by preventing the pathological alignment problem; however, this approach utilizes a fixed window area and exhibits quadratic computational complexity. In contrast, the fast DTW uses multiple resolutions to estimate the approximate optimal warping path. This fast DTW consider only the alignments within an adaptive window and has a linear computational complexity. However, a decrease in calculations can be expected only if the increase in calculations due to the length of the two time series is greater than the increase in calculations due to the parameter r. Moreover, the fast DTW has a pathological alignment problem and thus the classification accuracy of the fast DTW is lower than that of the constrained DTW. Accordingly, in this paper, we propose a fast constrained DTW method in which the optimal warping path estimation technique of fast DTW is applied within the limited window area of the constrained DTW.

A. FAST CONSTRAINED DTW

The proposed fast constrained DTW is implemented through coarsening, projection, and refinement of the fast DTW within the window area of the applied constrained DTW. The implementation process of fast SC-DTW, which is applied to the most representative SC-DTW algorithm of the constrained DTW techniques, is as follows. In Fig. 8, the window percentage value r' of SC-DTW is set as 0.2, and the parameter r, indicating the *radius* of the additional region around the estimated optimal warping path, is set to 1. In Fig. 8, multiple resolutions with data lengths of 1/4 (a) and 1/2 (b) compared to the original resolution (c) are created by coarsening from the original resolution (c). For the lower resolution shown in Fig. 8(a), the optimal warping path is found after alignments are performed within the limited window area of SC-DTW. This path is used to estimate the approximate optimal warping path of the next higher resolution with a data length of 1/2 compared to the original. For the resolution with a data length of 1/2 shown in Fig. 8(b), the area used for the alignments is shown in the dark and light gray regions corresponding to the approximate optimal warping path and

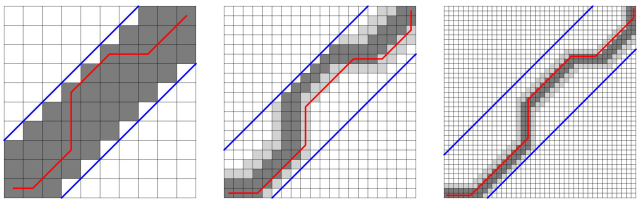

FIGURE 8. Accumulated cost matrix of three resolutions for the fast SC-DTW algorithm: (a) the lowest resolution with a data length of 1/4 compared to the original, (b) the resolution with a data length of 1/2 compared to the original, (c) the original resolution.

the additional areas, respectively. After alignments are performed in this area, the optimal warping path of Fig. 8(b) is found. Finally, the optimal warping path and DTW distance of the original resolution are calculated using the estimated optimal warping path of the resolution with a data length of 1/2 compared to the original.

For the proposed fast SC-DTW and the existing SC-DTW, the calculations for only the DTW distance are as follows. In the fast SC-DTW, calculations are performed for a total of $44 + 122 + 264 = 430$ cells in the accumulated cost matrix from the lowest resolution shown in Fig. 8(a) to the original resolution in Fig. 8(c). In addition, a total of $2 \times (10 + 20) = 60$ calculations are performed to create multiple resolutions with data lengths of 10 and 20 points. Lastly, the lengths of the optimal warping paths indicated by the red lines in Fig. 8(a) and (b) are 13 and 26, respectively. Therefore, a total of 39 calculations are computed for the optimal warping paths, with a total calculation number of $430 + 60 + 39 = 529$ for the proposed fast SC-DTW. Meanwhile, the total number of calculations in the SC-DTW window area which is shown in blue lines of Fig. 8(c) is 608. As a result, the fast SC-DTW can reduce the calculations by approximately 12.99% compared to the SC-DTW in the situation for Fig. 8.

Unlike the existing fast DTW, which finds the optimal warping path while considering all areas of the accumulated cost matrix, the proposed fast SC-DTW finds the optimal warping path only within the window area of the SC-DTW. Despite this difference, when the optimal warping path is located near index points that match on the time axis between the two time series, the fast SC-DTW has a computational complexity similar to that of the fast DTW. However, when the optimal warping path is located far from index points that match on the time axis between the two time series, the fast SC-DTW forms the optimal warping path at the boundary regions of the SC-DTW window. In this case, the fast SC-DTW has a lower computational complexity because the optimal warping path is shorter than that of the fast DTW. In addition, fast DTW may cause pathological alignment problems due to excessive alignment operations, which may reduce the overall classification accuracy. In contrast, the fast SC-DTW forms the optimal warping path within the window area of SC-DTW, thus preventing the pathological alignment problem and increasing the overall classification accuracy.

TABLE 1. Information for UCR time series datasets.

ID	Name	Class Number	Train Size	Test Size	Length
1	Synthetic-Control	6	300	300	60
2	ECG200	2	100	100	96
3	CBF	3	30	900	128
4	SwedishLeaf	15	500	625	128
5	Two-Patterns	4	1,000	4,000	128
6	FaceAll	14	560	1690	131
7	Gun-Point	2	50	150	150
8	Wafer	2	1,000	6,174	152
9	Adiac	37	390	391	176
10	Trace	4	100	100	275
11	Coffee	2	28	28	286
12	Lighting7	7	70	73	319
13	FaceFour	4	24	88	350
14	Yoga	2	300	3,000	426
15	OSULeaf	6	200	242	427
16	FISH	7	175	175	463
17	Beef	5	30	30	470
18	OliveOil	4	30	30	570
19	Lighting2	2	60	61	637

The pseudo-code for the fast SC-DTW is shown in Algorithm 1. The coarsening process corresponds to lines 3–15. Lines 4–13 dictate the process of determining whether lower resolutions should be generated by considering the additional calculations. Lines 14 and 15 generate an additional resolution by averaging two adjacent points in the two time series at the current lowest resolution. In lines 16–19, projection and refinement processes are repeated to calculate from the lowest resolution to the original resolution. The both lines 17 and 19 measure the similarity and find the optimal warping path. The lowest resolution that considers all of the accumulated cost matrix cells within the window area of the SC-DTW is calculated by line 17. In line 19, The remaining resolutions are calculated by considering the accumulated cost matrix cells of the estimated approximate optimal warping path and the additional surrounding areas. The pseudo-code of another fast constrained DTW to which other constrained DTW techniques are applied can be implemented by modifying lines 4, 7, 8, 17 and 19, in Algorithm 1.

IV. PERFORMANCE EVALUATION RESULTS

In this section, we compare the classification accuracy and time complexity of the proposed fast constrained DTW algorithm with those of the existing constrained DTW and fast DTW using various time series datasets provided by UCR [47] and synthetic datasets comprising a single period of a sine wave, as in [43].

A. CLASSIFICATION ACCURACY COMPARISON

For the experiment, the 19 time series datasets presented in Table 1 were used. The datasets consist of 2–50 class numbers, 24–1,000 train sizes, 28–6,174 test sizes, and lengths of 60–637 and include domains such as medicine, robotics, and handwriting recognition. The 19 datasets are ordered according to length for analysis of the experimental results.

Time series datasets were classified using the nearest neighbor technique, which is primarily used in similarity

Algorithm 1 [*Distance*, *Optimal_Warping_Path*] = Fast_SC_DTW(x, y, r, r')

Input:
x, y - Time series data
r(*radius*) - Surrounding radius of estimated optimal warping path
r'(*radius*) - Window percentage value
Output:
Distance - the result of similarity measure between x and y
Optimal_Warping_Path - optimal warping path between x and y

1: $Length\_x = |x|$; $Length\_y = |y|$;
2: $Shrunk\_x = x$; $Shrunk\_y = y$;
3: **while** TRUE
   // For searched cells of accumulated cost matrix
4: Cal_{CLR}=$(2 * Length\_x * r' + 1) * Length\_y$;
   // The length of the additional lower resolution
5: $Length\_x = Length\_x$ **div** 2;
6: $Length\_y = Length\_y$ **div** 2;
   // For searched cells of accumulated cost matrix
7: Cal_{ALR1}=$(2 * Length\_x * r' + 1) * Length\_y$;
   // For shrinking resolutions
8: Cal_{ALR2}=$Length\_x + Length\_y$;
   // For finding optimal warping path at the worst-case
9: Cal_{ALR3}=$Length\_x + Length\_y$;
   // For searched cells of accumulated cost matrix at the worst-case
10: Cal_{ALR4}=$(4 * r + 3) * Length\_y$;
11: Cal_{ALR}=Cal_{ALR1}+Cal_{ALR2}+Cal_{ALR3}+Cal_{ALR4};
   // Decide whether to make additional lower resolution or not.
12: **if** $Cal_{ALR} > Cal_{CLR}$
13: **break**;
   // Create an additional resolution
14: $Shrunk\_x = [Shrink\_Half(); Shrunk\_x]$;
15: $Shrunk\_y = [Shrink\_Half(); Shrunk\_y]$;
16: $[n, none] = |Shrunk\_x|$;
   // Measure the similarity and find the optimal warping path
   // For lowest resolution
17: [*Optimal_Warping_Path*, *Distance*] = $Fast\_SC\_DTW_{Full}(Shrunk\_x, Shrunk\_y)$;
   // For other resolutions except lowest resolution
18: **for** $i = 2$ **to** n **do**
19: [*Optimal_Warping_Path*, *Distance*] = $Fast\_SC\_DTW_{Path}(Shrunk\_x, Shrunk\_y)$;

measurements, including the DTW algorithm [48]. The test and train datasets of the 19 time series datasets were used as query sequences and sub-sequences, respectively. When a query sequence is utilized as the input, the similarities between the two time series of a query sequence and all sub-sequences are measured using algorithms given in the experiment, and the class information of the sub-sequence with the smallest similarity measure value is output as the classification result for each algorithm.

TABLE 2. Classification accuracy of ED, standard DTW, SC-DTW and I-DTW.

Dataset	ED	Std DTW	SC-DTW	I-DTW
Synthetic-Control	88.0	99.3	99.3	97.7
ECG200	88.0	77.0	82.0	89.0
CBF	85.2	99.7	99.6	94.4
SwedishLeaf	78.9	79.2	80.5	87.2
Two-Patterns	90.7	100.0	100.0	99.5
FaceAll	71.4	80.8	78.6	84.4
Gun-Point	91.3	90.7	94.0	95.3
Wafer	99.5	98.0	98.3	99.4
Adiac	61.1	60.4	60.4	59.8
Trace	76.0	100.0	100.0	100.0
Coffee	100.0	100.0	100.0	100.0
Lighting7	57.5	72.6	74.0	71.2
FaceFour	78.4	83.0	83.0	88.6
Yoga	83.0	83.6	84.2	85.5
OSULeaf	52.1	59.1	59.1	60.7
FISH	78.3	82.3	82.9	80.0
Beer	66.7	63.3	63.3	73.3
OliveOil	86.7	83.3	83.3	83.3
Lighting2	75.4	86.9	86.9	86.9
Avg	79.4	84.2	84.7	86.1

SC-DTW, I-DTW, and fast DTW, which all reduce the computational complexity of the standard DTW, were implemented for classification performance comparison, and the standard DTW was also implemented. The parameter r', indicating the window percentage value of SC-DTW and I-DTW, was set as 0.1, which is a commonly used value [46], and the parameter b, indicating the starting length of the I-DTW window, was set to 0 as in [40]. In addition, {0, 1, . . . , 9, 10} was used as the parameter r of the fast DTW, indicating the additional regions around the estimated optimal warping path. The proposed fast constrained DTW was implemented as fast SC-DTW and fast I-DTW by applying SC-DTW and I-DTW as the constrained DTW techniques, with the parameters r' and b of the constrained DTW techniques and the parameter r of the fast DTW set to the same values used in the previous algorithms. Additionally, the ED method, which is the traditional technique for obtaining similarity measures, was also implemented. If the indexing technique is applied to all the DTW algorithms used in this experiment, computational complexity will be further reduced. However, since the indexing technique reduces the calling times of the DTW algorithm, the amount of reduction is the same for all the DTW algorithms. Accordingly, the indexing technique is not applied for a more clear comparison in our experiment.

In Tables 2 and 3, the results of the classification accuracy produced by the seven algorithms are shown, where *Avg* indicates the average of the classification accuracy results for all datasets. As shown in Table 2, all implemented DTW algorithms that measure the similarity using an alignment process between the time series achieved a higher overall classification accuracy than the traditional ED method. As two constrained DTW techniques that effectively prevent the pathological alignment problem, the I-DTW and SC-DTW methods have a higher overall classification accuracy

than the standard DTW. As shown in Table 3, the average classification accuracies of the fast DTW, fast SC-DTW, and fast I-DTW approach those of the standard DTW, SC-DTW, and I-DTW, respectively, as the parameter r increases.

To analyze the experimental results, the classification accuracy of the proposed fast SC-DTW is compared with those of the existing fast DTW and SC-DTW in Figs. 9 and 10. Similarly, the classification accuracy of the proposed fast I-DTW is compared with those of the fast DTW and I-DTW in Figs. 11 and 12. The red dots in Fig. 9 represent the classification accuracies of the two algorithms for 19 datasets. Two regions are formed based on the blue line, indicating where the horizontal and vertical axes match. If there are more dots in a particular region, the algorithm has a higher overall classification accuracy for the 19 datasets. Comparing the classification accuracy of the fast SC-DTW and fast DTW, Fig. 9 shows that the red dots in the region of the proposed fast SC-DTW are denser than in the region of the fast DTW for all *radius* values. This trend indicates that the proposed fast SC-DTW for these 19 datasets has a higher overall classification accuracy than the fast DTW. Fig. 10 compares the classification accuracy of the fast SC-DTW and SC-DTW. For a *radius* of 0, 1, or 2, there are differences in the overall classification accuracy between the two algorithms; however, when the *radius* is 3 or more, the red dots are primarily located above the blue line, implying that the overall classification accuracy of the two algorithms is similar for the 19 datasets. From these experimental results, it is confirmed that the fast SC-DTW operating within the window area of the SC-DTW effectively prevents the pathological alignment problem, similar to the SC-DTW. In addition, as shown in Figs. 11 and 12, the fast I-DTW has higher overall classification accuracy than the fast DTW for all *radius* values, and when the *radius* is 3 or more, this method has an overall classification accuracy similar to that of the I-DTW.

B. COMPUTATIONAL COMPLEXITY COMPARISON

In this Section, we analyze the computational complexity of the proposed and existing DTW algorithms in the same experimental environment as Section IV-A. Tables 4 and 6 shows the total number of calculations used to classify all test datasets for each of the 19 datasets in Table 1. The number of calculations includes three components: accumulated cost matrix, optimal warping path, and multi-resolutions. Table 4 presents results for ED, standard DTW, SC-DTW, and I-DTW, while Table 6 shows results for the fast DTW, fast SC-DTW, and fast I-DTW algorithms. Tables 5 and 7 show the comparison results for the rate (R_{NC}) of the number of calculations between each algorithm and standard DTW as described in (19), where *Avg* indicates the average of R_{NC} for each algorithm in the 19 datasets.

$$R_{NC} = \frac{NC_{anothrer\ algorithm}}{NC_{standard\ DTW}} \times 100 \tag{19}$$

In (19), $NC_{standard\ DTW}$ represents the total number of calculations for the standard DTW, and $NC_{another\ algorithm}$ denotes the total number of calculations for another implemented algorithms. As shown by the classification accuracy comparison, the proposed fast SC-DTW and fast I-DTW have classification accuracies similar to those of SC-DTW and I-DTW when the parameter r is 3 or more. Therefore, when the parameter r is 3, the proposed fast SC-DTW, which has an average R_{NC} of 11.4%, reduces the calculations by approximately 30.1% and 40.3% compared with the fast DTW and SC-DTW, which have average R_{NC} values of 16.3% and 19.1%. In addition, compared to the fast DTW and I-DTW, which have average R_{NC} values of 16.3% and 10.1%, the fast I-DTW, with an average R_{NC} of 7.8%, reduces the calculations by approximately 52.2% and 22.9%, respectively.

In Fig. 13, the R_{NC} of the proposed fast SC-DTW for 19 datasets is compared with those of the existing fast DTW and SC-DTW according to the *radius*; in addition, the proposed fast I-DTW is also compared with the existing fast DTW and I-DTW. The SC-DTW and I-DTW, which are not influenced by the parameter r, have a constant R_{NC} for all *radius* values. The longer the data length and the smaller the *radius*, the lower the R_{NC} of the fast DTW. Accordingly, for the fast DTW shown in Fig. 13, the R_{NC} of the 19th dataset is lower than that of the first dataset, and (a) has a lower overall R_{NC} than that of (f). The first dataset in Fig. 13(f), which is the shortest among the 19 datasets, has an R_{NC} of 100%, indicating that when the *radius* is 10, the calculations in the first dataset are the same as those in the standard DTW. Therefore, when the *radius* is 10 and the data length is less than 60, which is the length of the first dataset, or when the data length is 60 and the *radius* is 10 or more, the fast DTW does not reduce the computational complexity. In contrast, for all datasets and *radius* values, the proposed fast SC-DTW has an R_{NC} value similar to or lower than that of the algorithm (fast DTW or SC-DTW) with the lower R_{NC}. Similarly, the R_{NC} of the fast I-DTW is similar to or lower than that of the algorithm (fast DTW or I-DTW) with the lower R_{NC}.

C. TIME COMPLEXITY COMPARISON

In this experiment, we compared the time complexity by measuring the execution time of the implemented algorithms, which is weakly affected by the data shape but is greatly affected by the data length. Accordingly, synthetic datasets comprising a single period of a sine wave with Gaussian noise were used as the experimental datasets [43]. The data lengths vary from 10 to 20,000. The standard deviation of the Gaussian noise is 0.01, and the execution time is measured in milliseconds.

In this experiment, the standard DTW and the existing fast DTW, SC-DTW, and I-DTW were used. The proposed fast constrained DTW algorithm uses the fast SC-DTW and fast I-DTW applying the SC-DTW and I-DTW. In the implemented algorithms, all parameters are the same as those for the experiment in Section IV-A except for the parameter r, which was set to $\{0, 5, 10, 100\}$. This experiment was

TABLE 3. Classification accuracy of fast DTW, fast SC-DTW and fast I-DTW according to *radius*.

Dataset	DTW	0	1	2	3	4	5	6	7	8	9	10
Synthetic-Control	Fast	89.0	92.3	97.0	97.3	99.0	98.7	98.7	99.0	99.3	99.3	99.3
	Fast SC	90.0	95.7	99.3	99.3	99.3	99.3	99.3	99.3	99.3	99.3	99.3
	Fast I	97.7	97.7	97.7	97.7	97.7	97.7	97.7	97.7	97.7	97.7	97.7
ECG200	Fast	80.0	77.0	77.0	79.0	77.0	77.0	77.0	77.0	77.0	77.0	77.0
	Fast SC	84.0	85.0	85.0	82.0	82.0	82.0	82.0	82.0	82.0	82.0	82.0
	Fast I	89.0	89.0	89.0	89.0	89.0	89.0	89.0	89.0	89.0	89.0	89.0
CBF	Fast	99.4	99.4	99.4	99.6	99.7	99.9	99.8	99.8	99.9	99.9	99.8
	Fast SC	99.0	99.2	99.7	99.7	99.6	99.6	99.6	99.6	99.6	99.6	99.6
	Fast I	93.2	94.0	94.4	94.4	94.4	94.4	94.4	94.4	94.4	94.4	94.4
SwedishLeaf	Fast	78.9	78.6	79.2	79.2	79.4	79.4	79.2	79.2	79.2	79.2	79.2
	Fast SC	79.2	80.2	80.2	80.3	80.5	80.5	80.5	80.5	80.5	80.5	80.5
	Fast I	86.1	86.6	87.2	87.2	87.2	87.2	87.2	87.2	87.2	87.2	87.2
Two-Patterns	Fast	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	Fast SC	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	Fast I	99.5	99.5	99.5	99.5	99.5	99.5	99.5	99.5	99.5	99.5	99.5
FaceAll	Fast	72.2	80.8	79.1	79.6	79.9	79.9	79.8	79.6	80.1	80.3	80.4
	Fast SC	76.9	78.2	78.4	78.3	78.6	78.6	78.6	78.6	78.6	78.6	78.6
	Fast I	82.7	84.1	84.4	84.4	84.4	84.4	84.4	84.4	84.4	84.4	84.4
Gun-Point	Fast	93.3	92.0	91.3	91.3	91.3	91.3	91.3	90.7	90.7	90.7	90.7
	Fast SC	94.7	94.0	94.0	94.0	94.0	94.0	94.0	94.0	94.0	94.0	94.0
	Fast I	96.0	95.3	95.3	95.3	95.3	95.3	95.3	95.3	95.3	95.3	95.3
Wafer	Fast	97.7	97.8	97.9	97.9	98.0	97.9	97.9	97.9	97.9	97.9	97.9
	Fast SC	98.3	98.3	98.3	98.3	98.3	98.3	98.3	98.3	98.3	98.3	98.3
	Fast I	99.4	99.5	99.5	99.4	99.4	99.4	99.4	99.4	99.4	99.4	99.4
Adiac	Fast	60.6	60.9	60.9	60.9	60.6	60.6	60.6	60.6	60.4	60.4	60.4
	Fast SC	60.6	60.9	60.9	60.9	60.6	60.6	60.4	60.4	60.4	60.4	60.4
	Fast I	59.3	59.8	59.8	59.8	59.8	59.8	59.8	59.8	59.8	59.8	59.8
Trace	Fast	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	Fast SC	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	Fast I	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
Coffee	Fast	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	Fast SC	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	Fast I	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
Lighting7	Fast	69.9	69.9	65.8	68.5	72.6	71.2	71.2	71.2	71.2	71.2	68.5
	Fast SC	78.1	75.3	74.0	74.0	74.0	75.3	75.3	75.3	75.3	75.3	75.3
	Fast I	68.5	71.2	67.1	71.2	72.6	72.6	71.2	71.2	71.2	71.2	71.2
FaceFour	Fast	63.6	84.1	83.0	84.1	83.0	81.8	83.0	83.0	83.0	83.0	83.0
	Fast SC	79.5	83.0	85.2	84.1	83.0	83.0	83.0	83.0	83.0	83.0	83.0
	Fast I	89.8	88.6	88.6	88.6	88.6	88.6	88.6	88.6	88.6	88.6	88.6
Yoga	Fast	83.4	83.7	83.8	83.6	83.7	83.6	83.7	83.7	83.6	83.7	83.7
	Fast SC	83.9	84.2	84.5	84.2	84.1	84.1	84.1	84.2	84.2	84.2	84.2
	Fast I	85.1	85.3	85.5	85.5	85.4	85.5	85.5	85.5	85.5	85.5	85.5
OSULeaf	Fast	50.4	57.9	59.1	59.5	59.1	58.7	59.1	58.7	58.7	58.7	59.1
	Fast SC	61.2	59.9	58.7	58.7	58.7	58.7	59.1	59.1	59.1	59.1	59.1
	Fast I	59.5	60.3	60.3	60.3	60.7	60.7	60.7	60.7	60.7	60.7	60.7
FISH	Fast	83.4	82.9	82.9	82.9	82.9	82.3	82.3	82.3	82.3	82.3	82.3
	Fast SC	83.4	82.9	82.9	82.9	82.9	82.9	82.9	82.9	82.9	82.9	82.9
	Fast I	81.7	80.0	80.0	80.0	80.0	80.0	80.0	80.0	80.0	80.0	80.0
Beer	Fast	66.7	66.7	63.3	63.3	63.3	63.3	63.3	63.3	63.3	63.3	63.3
	Fast SC	66.7	66.7	63.3	63.3	63.3	63.3	63.3	63.3	63.3	63.3	63.3
	Fast I	70.0	66.7	73.3	73.3	73.3	73.3	73.3	73.3	73.3	73.3	73.3
OliveOil	Fast	86.7	83.3	83.3	83.3	83.3	83.3	83.3	83.3	83.3	83.3	83.3
	Fast SC	86.7	83.3	83.3	83.3	83.3	83.3	83.3	83.3	83.3	83.3	83.3
	Fast I	86.7	83.3	83.3	83.3	83.3	83.3	83.3	83.3	83.3	83.3	83.3
Lighting2	Fast	91.8	88.5	86.9	91.8	91.8	88.5	86.9	86.9	86.9	86.9	86.9
	Fast SC	88.5	88.5	88.5	90.2	88.5	88.5	90.2	88.5	88.5	88.5	88.5
	Fast I	90.2	86.9	86.9	88.5	88.5	88.5	88.5	88.5	86.9	86.9	86.9
Avg	Fast	82.5	84.0	83.7	84.3	84.4	84.1	84.1	84.0	84.0	84.1	83.9
	Fast SC	84.8	85.0	85.1	84.9	84.8	84.8	84.9	84.9	84.9	84.9	84.9
	Fast I	86.0	85.7	85.9	86.2	86.3	86.3	86.2	86.2	86.1	86.1	86.1

implemented using MATLAB R2019b using a 3.8-GHz Intel Core i7-10700K CPU with RAM of 32 GB and 64-bit Windows 10 Pro.

Fig. 14 shows the execution time of each algorithm for a data length of 10–1,000. The execution time of the standard DTW increases quadratically according to the data length. The SC-DTW, I-DTW, and fast DTW with *radius* of 0, 5, and 10 are faster than the standard DTW. However, for a data length below 500, the fast DTW with a radius of 100 has an execution time similar to that of the standard DTW. In contrast, the proposed fast SC-DTW has a lower or similar execution time compared with the fast DTW and SC-DTW,

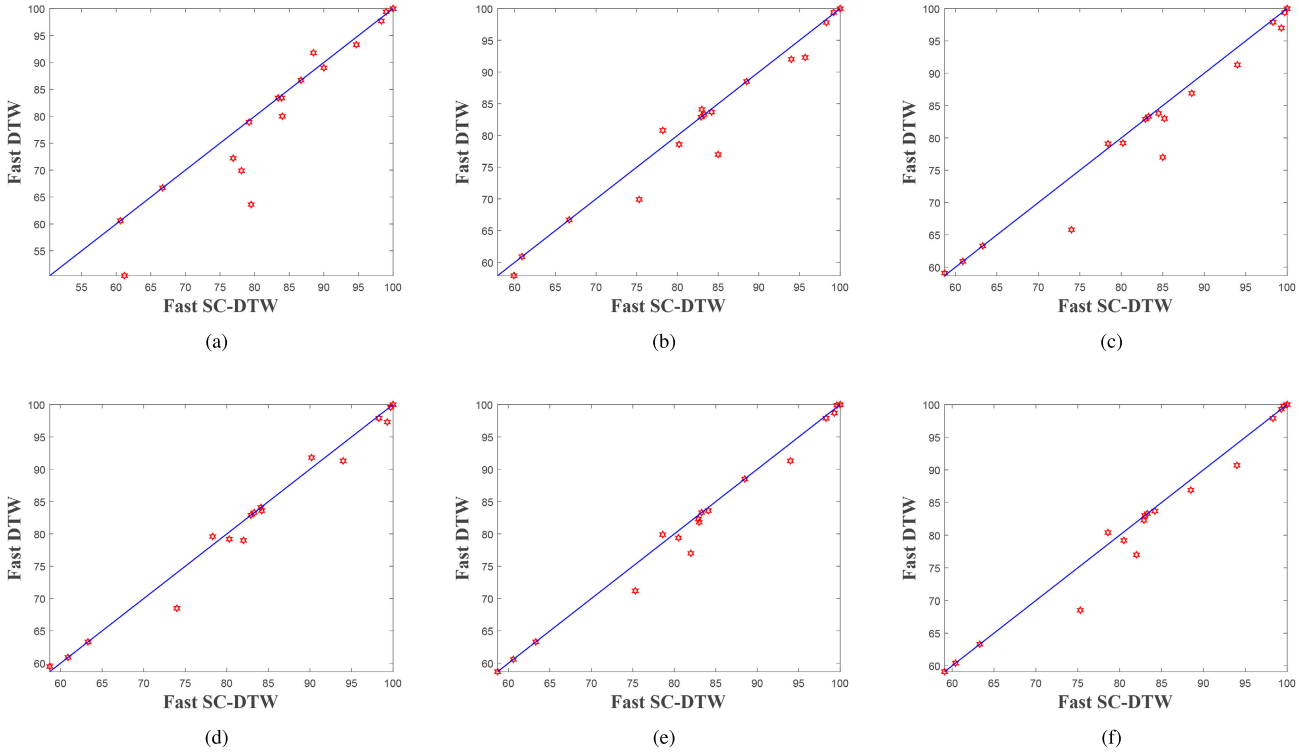

FIGURE 9. Comparison of classification accuracy according to *radius* between fast SC-DTW and fast DTW. (*radius*∈{0,1,2,3,5,10} = {(a),(b), . . .,(f)}).

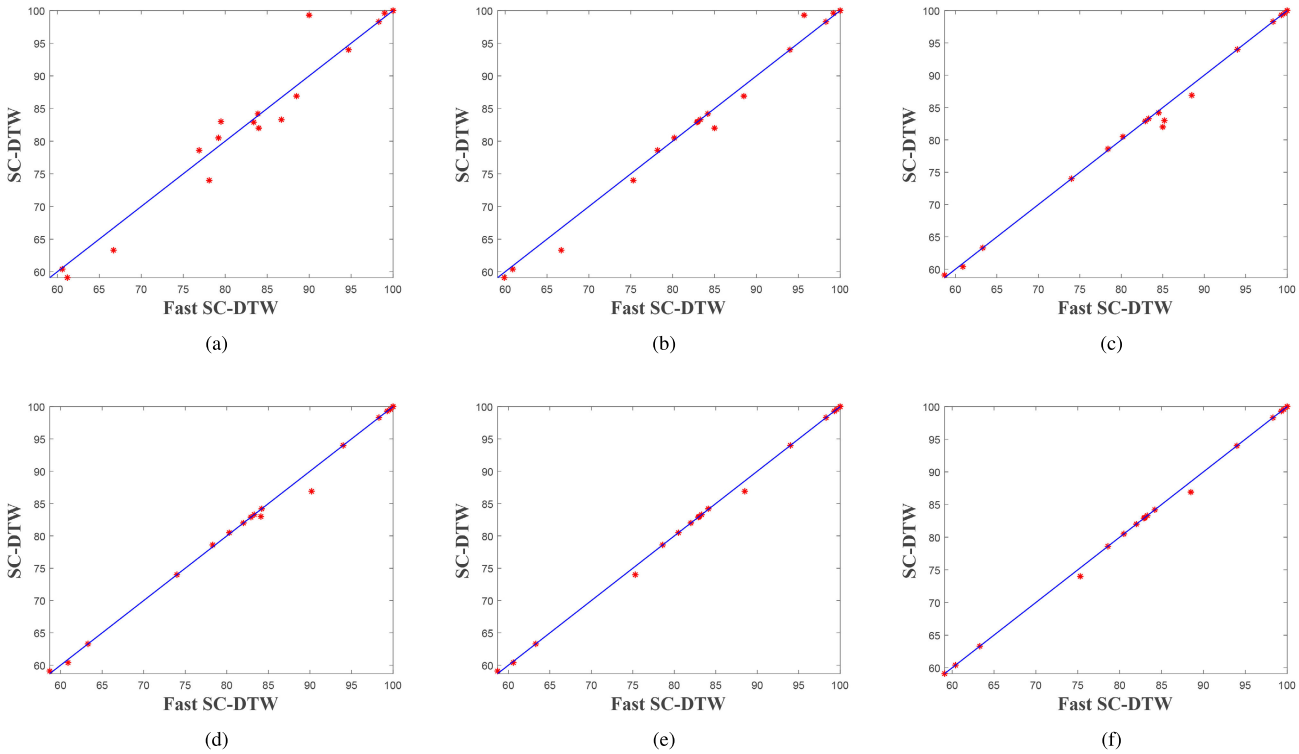

FIGURE 10. Comparison of classification accuracy according to *radius* between fast SC-DTW and SC-DTW. (*radius*∈{0,1,2,3,5,10} = {(a),(b), . . .,(f)}).

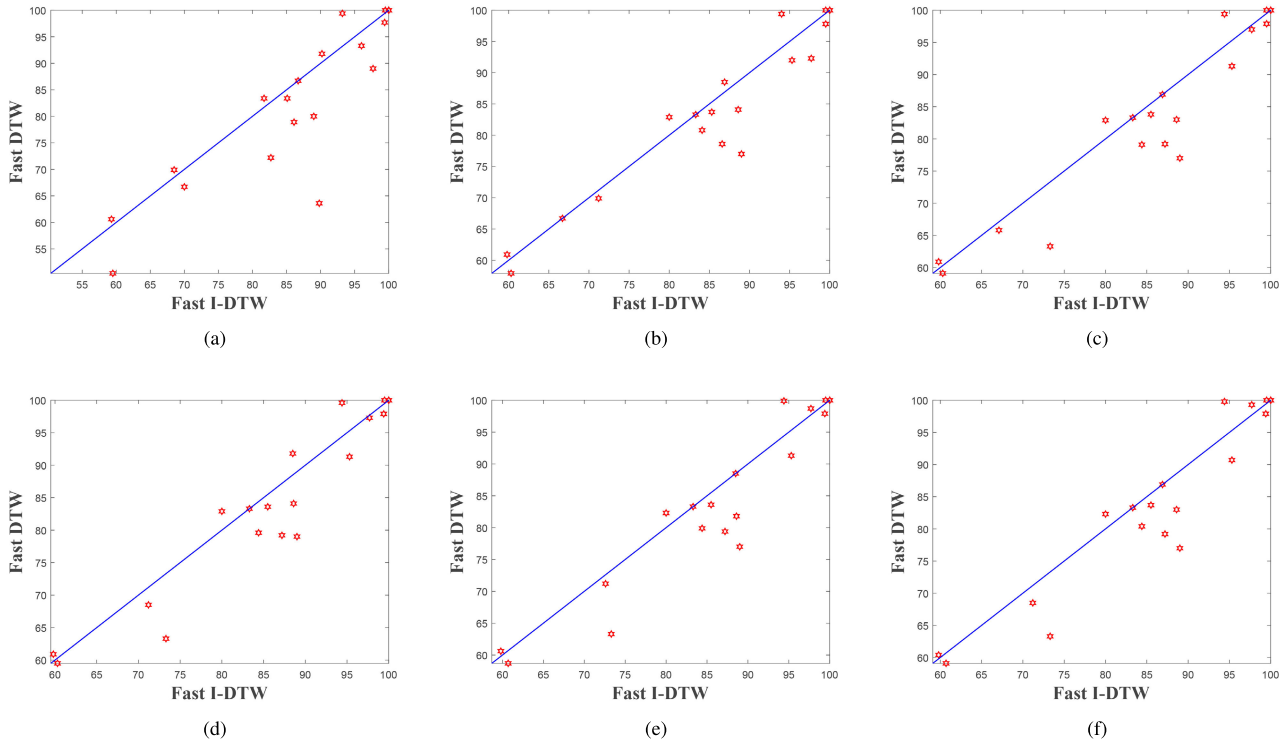

FIGURE 11. Comparison of classification accuracy according to *radius* between fast I-DTW and fast DTW. (*radius*∈{0,1,2,3,5,10} = {(a),(b), . . .,(f)}).

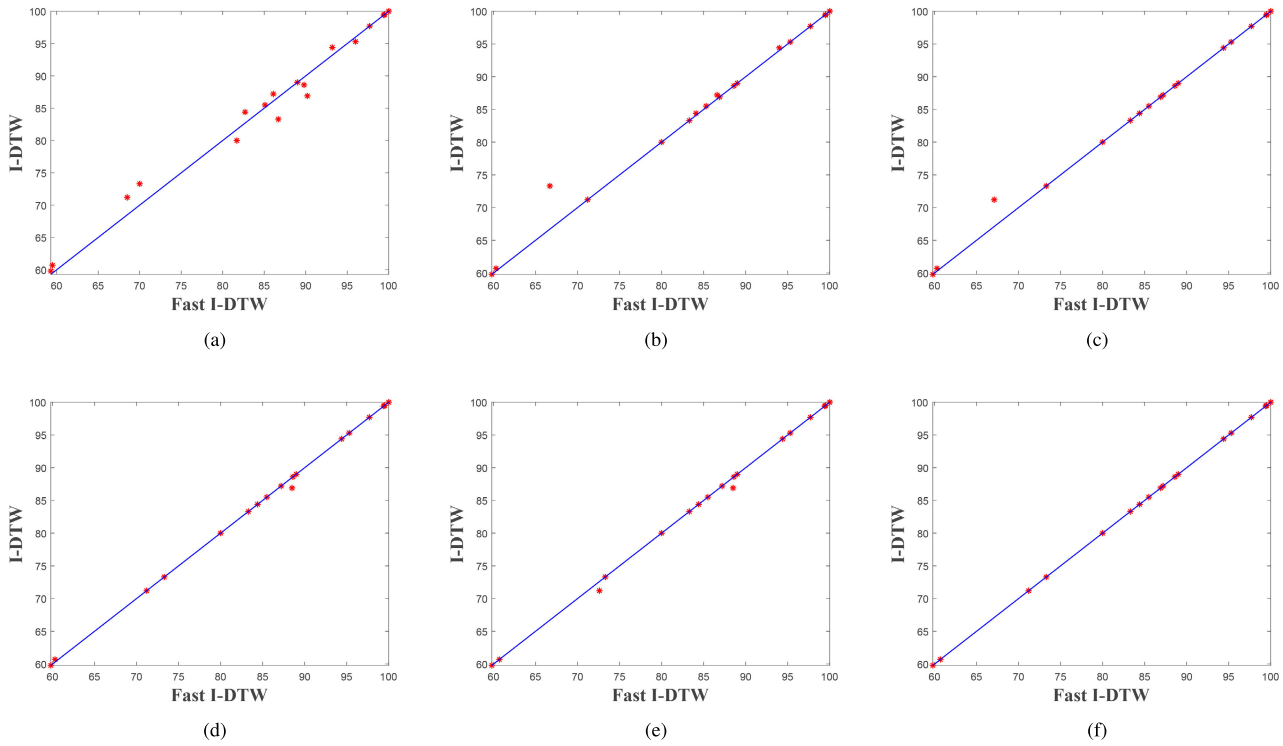

FIGURE 12. Comparison of classification accuracy according to *radius* between fast I-DTW and I-DTW. (*radius*∈{0,1,2,3,5,10} = {(a),(b), . . .,(f)}).

and the proposed fast I-DTW is faster than or similar to the fast DTW and I-DTW.

For dataset with short data lengths, as the *radius* increases, the execution time of the existing fast DTW approaches that

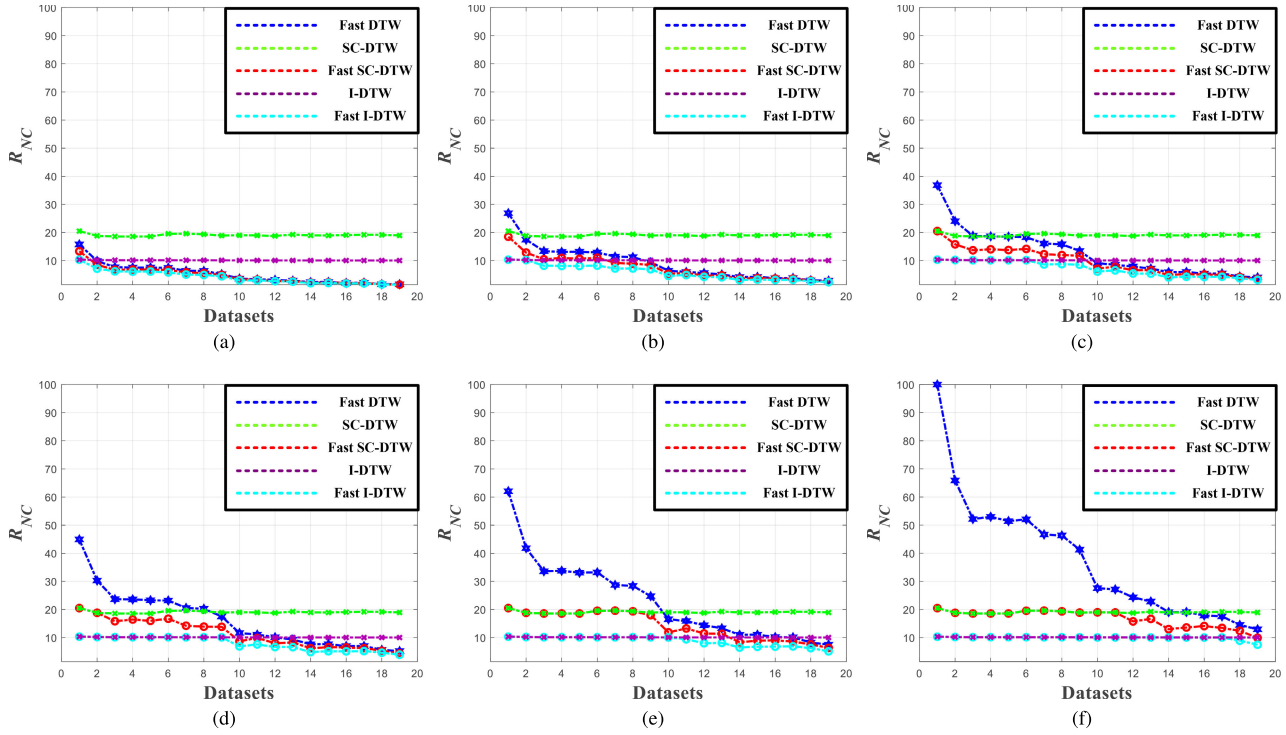

FIGURE 13. Comparison of R_{NC} of fast DTW, SC-DTW, fast SC-DTW, I-DTW and fast I-DTW according to *radius*. (*radius*∈{0,1,2,3,5,10} = {(a),(b), . . .,(f)}).

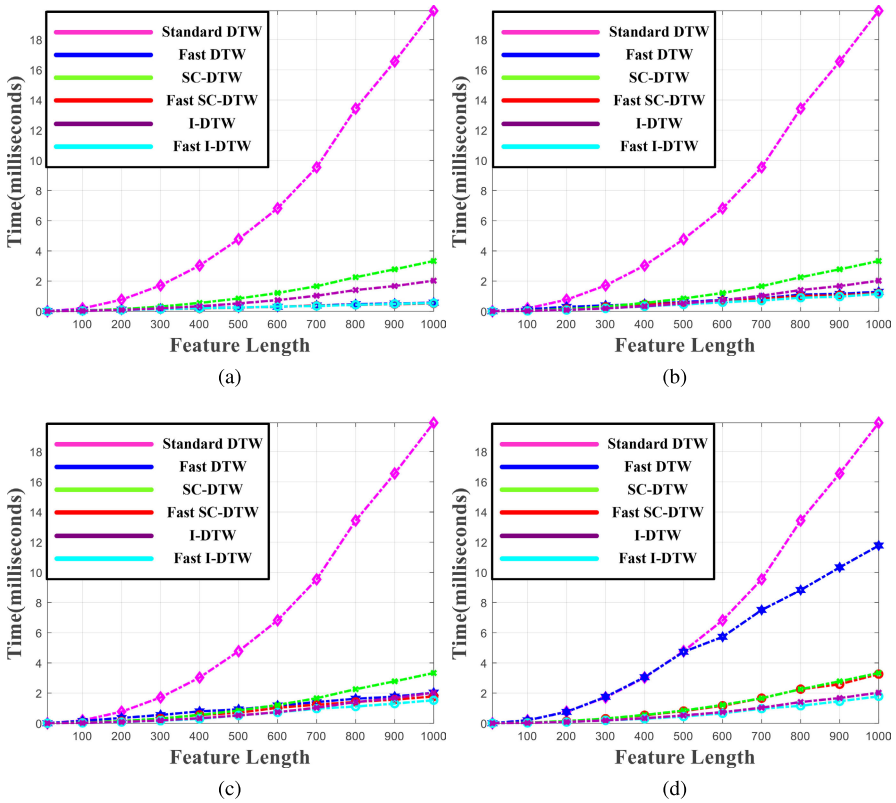

FIGURE 14. Execution time for standard DTW, fast DTW, SC-DTW, fast SC-DTW, I-DTW and fast I-DTW for short time series. (*radius*∈{0,5,10,100} = {(a),(b),(c),(d)}).

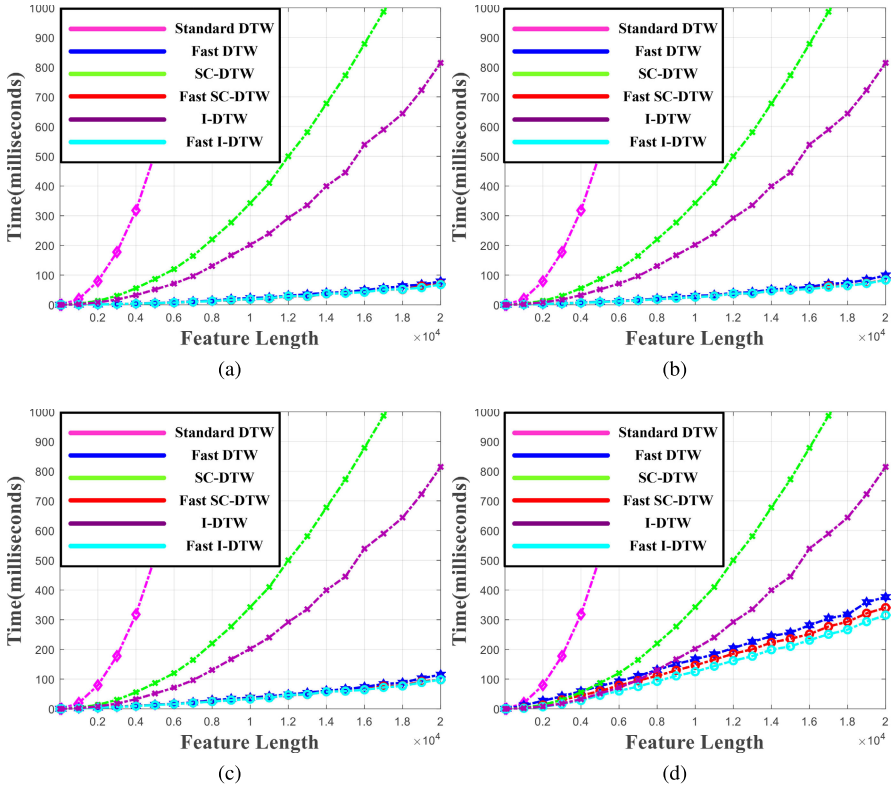

FIGURE 15. Execution time for standard DTW, fast DTW, SC-DTW, fast SC-DTW, I-DTW and fast I-DTW for long time series. (*radius*∈{0,5,10,100} = {(a),(b),(c),(d)}).

TABLE 4. Number of calculations of ED, standard DTW, SC-DTW and I-DTW.

Dataset	ED	Std DTW	SC-DTW	I-DTW
Synthetic-Control	5.4E+06	3.2E+08	6.6E+07	3.3E+07
ECG200	9.6E+05	9.2E+07	1.7E+07	9.4E+06
CBF	3.5E+06	4.4E+08	8.2E+07	4.5E+07
SwedishLeaf	4.0E+07	5.1E+09	9.5E+08	5.2E+08
Two-Patterns	5.1E+08	6.6E+10	1.2E+10	6.7E+09
FaceAll	1.2E+08	1.6E+10	3.2E+09	1.6E+09
Gun-Point	1.1E+06	1.7E+08	3.3E+07	1.7E+07
Wafer	9.4E+08	1.4E+11	2.8E+10	1.4E+10
Adiac	2.7E+07	4.7E+09	8.9E+08	4.8E+08
Trace	2.8E+06	7.6E+08	1.4E+08	7.6E+07
Coffee	2.2E+05	6.4E+07	1.2E+07	6.5E+06
Lighting7	1.6E+06	5.2E+08	9.8E+07	5.2E+07
FaceFour	7.4E+05	2.6E+08	5.0E+07	2.6E+07
Yoga	3.8E+08	1.6E+11	3.1E+10	1.6E+10
OSULeaf	2.1E+07	8.8E+09	1.7E+09	8.9E+08
FISH	1.4E+07	6.6E+09	1.3E+09	6.6E+08
Beer	4.2E+05	2.0E+08	3.8E+07	2.0E+07
OliveOil	5.1E+05	2.9E+08	5.6E+07	2.9E+07
Lighting2	2.3E+06	1.5E+09	2.8E+08	1.5E+08

TABLE 5. Rate of the number of calculations of ED, standard DTW, SC-DTW and I-DTW compared to standard DTW.

Dataset	ED	Std DTW	SC-DTW	I-DTW
Synthetic-Control	1.7	100.0	20.5	10.3
ECG200	1.0	100.0	18.8	10.2
CBF	0.8	100.0	18.6	10.2
SwedishLeaf	0.8	100.0	18.6	10.2
Two-Patterns	0.8	100.0	18.6	10.2
FaceAll	0.8	100.0	19.6	10.2
Gun-Point	0.7	100.0	19.6	10.1
Wafer	0.7	100.0	19.4	10.1
Adiac	0.6	100.0	18.9	10.1
Trace	0.4	100.0	19.0	10.1
Coffee	0.3	100.0	18.9	10.1
Lighting7	0.3	100.0	18.8	10.1
FaceFour	0.3	100.0	19.3	10.1
Yoga	0.2	100.0	19.0	10.0
OSULeaf	0.2	100.0	18.9	10.0
FISH	0.2	100.0	19.1	10.0
Beer	0.2	100.0	19.2	10.0
OliveOil	0.2	100.0	19.2	10.0
Lighting2	0.2	100.0	18.9	10.0
Avg	0.5	100.0	19.1	10.1

of the standard DTW while the execution time of the proposed fast SC-DTW and fast I-DTW approaches those of the SC-DTW and I-DTW, respectively. In other words, when the increase in execution time due to the data length is less than the increase in execution time due to the *radius*, such as when the *radius* is 100 and the data length is 500 or less, the existing fast DTW does not improve the execution time over the standard DTW as shown in Fig. 14(d). However, the proposed fast SC-DTW and fast I-DTW improve the execution time, similar to the SC-DTW and I-DTW, respectively.

For data lengths of 10–20,000, the execution time of each algorithm is shown in Fig. 15. The standard DTW shows an exponential response to the data length, and SC-DTW and I-DTW show flatter exponential curves compared to that of the standard DTW. In contrast, the fast DTW, fast SC-DTW, and fast I-DTW show straight lines for all *radius* values.

In this experiment on dataset with long data lengths, the existing fast DTW shows a linear trend in execution time,

TABLE 6. Number of calculations of fast DTW, fast SC-DTW and fast I-DTW according to *radius*.

Dataset	DTW	0	1	2	3	4	5	6	7	8	9	10
Synthetic-Control	Fast	5.1E+07	8.7E+07	1.2E+08	1.5E+08	1.7E+08	2.0E+08	2.2E+08	2.3E+08	2.5E+08	2.6E+08	3.2E+08
	Fast SC	4.3E+07	6.0E+07	6.6E+07	6.6E+07	6.6E+07	6.6E+07	6.6E+07	6.6E+07	6.6E+07	6.6E+07	6.6E+07
	Fast I	3.3E+07	3.3E+07	3.3E+07	3.3E+07	3.3E+07	3.3E+07	3.3E+07	3.3E+07	3.3E+07	3.3E+07	3.3E+07
ECG200	Fast	9.1E+06	1.6E+07	2.2E+07	2.8E+07	3.4E+07	3.9E+07	4.3E+07	4.8E+07	5.5E+07	5.8E+07	6.1E+07
	Fast SC	7.8E+06	1.2E+07	1.5E+07	1.7E+07	1.7E+07	1.7E+07	1.7E+07	1.7E+07	1.7E+07	1.7E+07	1.7E+07
	Fast I	6.6E+06	9.4E+06	9.4E+06	9.4E+06	9.4E+06	9.4E+06	9.4E+06	9.4E+06	9.4E+06	9.4E+06	9.4E+06
CBF	Fast	3.4E+07	5.9E+07	8.3E+07	1.0E+08	1.3E+08	1.5E+08	1.7E+08	1.8E+08	2.0E+08	2.2E+08	2.3E+08
	Fast SC	3.0E+07	4.6E+07	6.1E+07	7.0E+07	8.2E+07	8.2E+07	8.2E+07	8.2E+07	8.2E+07	8.2E+07	8.2E+07
	Fast I	2.7E+07	3.6E+07	4.5E+07	4.5E+07	4.5E+07	4.5E+07	4.5E+07	4.5E+07	4.5E+07	4.5E+07	4.5E+07
SwedishLeaf	Fast	3.9E+08	6.7E+08	9.5E+08	1.2E+09	1.5E+09	1.7E+09	1.9E+09	2.1E+09	2.3E+09	2.5E+09	2.7E+09
	Fast SC	3.5E+08	5.6E+08	7.2E+08	8.4E+08	9.5E+08	9.5E+08	9.5E+08	9.5E+08	9.5E+08	9.5E+08	9.5E+08
	Fast I	3.1E+08	4.2E+08	5.2E+08	5.2E+08	5.2E+08	5.2E+08	5.2E+08	5.2E+08	5.2E+08	5.2E+08	5.2E+08
Two-Patterns	Fast	4.9E+09	8.6E+09	1.2E+10	1.5E+10	1.8E+10	2.2E+10	2.4E+10	2.7E+10	2.9E+10	3.1E+10	3.4E+10
	Fast SC	4.5E+09	7.0E+09	9.0E+09	1.0E+10	1.2E+10	1.2E+10	1.2E+10	1.2E+10	1.2E+10	1.2E+10	1.2E+10
	Fast I	4.0E+09	5.3E+09	6.7E+09	6.7E+09	6.7E+09	6.7E+09	6.7E+09	6.7E+09	6.7E+09	6.7E+09	6.7E+09
FaceAll	Fast	1.2E+09	2.1E+09	3.0E+09	3.8E+09	4.5E+09	5.4E+09	6.0E+09	6.7E+09	7.3E+09	7.9E+09	8.5E+09
	Fast SC	1.1E+09	1.8E+09	2.3E+09	2.7E+09	3.2E+09	3.2E+09	3.2E+09	3.2E+09	3.2E+09	3.2E+09	3.2E+09
	Fast I	9.4E+08	1.3E+09	1.6E+09	1.6E+09	1.6E+09	1.6E+09	1.6E+09	1.6E+09	1.6E+09	1.6E+09	1.6E+09
Gun-Point	Fast	1.1E+07	1.9E+07	2.7E+07	3.5E+07	4.2E+07	4.8E+07	5.6E+07	6.2E+07	6.8E+07	7.3E+07	7.9E+07
	Fast SC	9.8E+06	1.6E+07	2.1E+07	2.4E+07	2.7E+07	3.3E+07	3.3E+07	3.3E+07	3.3E+07	3.3E+07	3.3E+07
	Fast I	8.5E+06	1.2E+07	1.5E+07	1.7E+07	1.7E+07	1.7E+07	1.7E+07	1.7E+07	1.7E+07	1.7E+07	1.7E+07
Wafer	Fast	9.0E+09	1.6E+10	2.3E+10	2.9E+10	3.5E+10	4.0E+10	4.7E+10	5.2E+10	5.7E+10	6.1E+10	6.6E+10
	Fast SC	7.9E+09	1.3E+10	1.7E+10	2.0E+10	2.2E+10	2.8E+10	2.8E+10	2.8E+10	2.8E+10	2.8E+10	2.8E+10
	Fast I	7.0E+09	1.0E+10	1.2E+10	1.4E+10	1.4E+10	1.4E+10	1.4E+10	1.4E+10	1.4E+10	1.4E+10	1.4E+10
Adiac	Fast	2.4E+08	4.4E+08	6.3E+08	8.3E+08	1.0E+09	1.2E+09	1.3E+09	1.5E+09	1.7E+09	1.8E+09	1.9E+09
	Fast SC	2.3E+08	4.0E+08	5.5E+08	6.5E+08	7.5E+08	8.5E+08	8.9E+08	8.9E+08	8.9E+08	8.9E+08	8.9E+08
	Fast I	2.1E+08	3.3E+08	4.0E+08	4.8E+08	4.8E+08	4.8E+08	4.8E+08	4.8E+08	4.8E+08	4.8E+08	4.8E+08
Trace	Fast	2.7E+07	4.9E+07	6.9E+07	8.8E+07	1.1E+08	1.2E+08	1.4E+08	1.6E+08	1.8E+08	1.9E+08	2.1E+08
	Fast SC	2.5E+07	4.0E+07	5.4E+07	6.6E+07	7.7E+07	9.0E+07	9.7E+07	1.0E+08	1.1E+08	1.4E+08	1.4E+08
	Fast I	2.3E+07	3.5E+07	4.6E+07	5.2E+07	5.8E+07	7.6E+07	7.6E+07	7.6E+07	7.6E+07	7.6E+07	7.6E+07
Coffee	Fast	2.1E+06	3.8E+06	5.5E+06	7.1E+06	8.7E+06	1.0E+07	1.2E+07	1.3E+07	1.5E+07	1.6E+07	1.7E+07
	Fast SC	2.0E+06	3.6E+06	5.0E+06	6.3E+06	7.6E+06	8.5E+06	9.3E+06	1.0E+07	1.1E+07	1.2E+07	1.2E+07
	Fast I	1.9E+06	3.1E+06	4.2E+06	4.9E+06	5.5E+06	6.0E+06	6.5E+06	6.5E+06	6.5E+06	6.5E+06	6.5E+06
Lighting7	Fast	1.6E+07	2.9E+07	4.1E+07	5.3E+07	6.4E+07	7.5E+07	8.6E+07	9.7E+07	1.1E+08	1.2E+08	1.3E+08
	Fast SC	1.4E+07	2.4E+07	3.4E+07	4.2E+07	4.9E+07	6.0E+07	6.4E+07	6.9E+07	7.4E+07	7.8E+07	8.2E+07
	Fast I	1.4E+07	2.2E+07	2.8E+07	3.5E+07	3.8E+07	4.2E+07	5.2E+07	5.2E+07	5.2E+07	5.2E+07	5.2E+07
FaceFour	Fast	7.4E+06	1.3E+07	1.9E+07	2.4E+07	2.9E+07	3.4E+07	4.0E+07	4.5E+07	5.0E+07	5.4E+07	5.9E+07
	Fast SC	7.0E+06	1.2E+07	1.7E+07	2.1E+07	2.5E+07	2.9E+07	3.3E+07	3.6E+07	3.8E+07	4.1E+07	4.3E+07
	Fast I	6.5E+06	1.1E+07	1.4E+07	1.7E+07	1.9E+07	2.1E+07	2.3E+07	2.6E+07	2.6E+07	2.6E+07	2.6E+07
Yoga	Fast	3.8E+09	6.8E+09	9.7E+09	1.2E+10	1.5E+10	1.8E+10	2.1E+10	2.3E+10	2.6E+10	2.9E+10	3.1E+10
	Fast SC	3.5E+09	5.8E+09	8.0E+09	1.0E+10	1.2E+10	1.4E+10	1.5E+10	1.8E+10	1.9E+10	2.0E+10	2.1E+10
	Fast I	3.3E+09	5.0E+09	6.6E+09	8.0E+09	9.8E+09	1.1E+10	1.1E+10	1.2E+10	1.3E+10	1.6E+10	1.6E+10
OSULeaf	Fast	2.1E+08	3.7E+08	5.3E+08	6.8E+08	8.3E+08	9.7E+08	1.1E+09	1.3E+09	1.4E+09	1.6E+09	1.7E+09
	Fast SC	2.0E+08	3.4E+08	4.6E+08	5.8E+08	6.9E+08	7.9E+08	8.8E+08	1.0E+09	1.1E+09	1.1E+09	1.2E+09
	Fast I	1.8E+08	2.9E+08	3.8E+08	4.5E+08	5.5E+08	6.0E+08	6.4E+08	6.9E+08	7.3E+08	8.9E+08	8.9E+08
FISH	Fast	1.4E+08	2.5E+08	3.5E+08	4.6E+08	5.6E+08	6.7E+08	7.7E+08	8.7E+08	9.7E+08	1.1E+09	1.2E+09
	Fast SC	1.3E+08	2.4E+08	3.3E+08	4.3E+08	5.1E+08	5.9E+08	6.7E+08	7.5E+08	8.2E+08	8.8E+08	9.3E+08
	Fast I	1.2E+08	2.0E+08	2.8E+08	3.4E+08	4.1E+08	4.5E+08	4.8E+08	5.2E+08	5.5E+08	5.9E+08	6.6E+08
Beer	Fast	4.3E+06	7.6E+06	1.1E+07	1.4E+07	1.7E+07	2.0E+07	2.3E+07	2.6E+07	2.9E+07	3.2E+07	3.5E+07
	Fast SC	4.1E+06	7.1E+06	9.8E+06	1.2E+07	1.5E+07	1.7E+07	1.9E+07	2.1E+07	2.4E+07	2.5E+07	2.7E+07
	Fast I	3.8E+06	6.3E+06	8.5E+06	1.0E+07	1.2E+07	1.4E+07	1.5E+07	1.6E+07	1.7E+07	1.8E+07	2.0E+07
OliveOil	Fast	4.7E+06	8.7E+06	1.3E+07	1.7E+07	2.0E+07	2.4E+07	2.8E+07	3.2E+07	3.5E+07	3.9E+07	4.2E+07
	Fast SC	4.6E+06	8.4E+06	1.2E+07	1.6E+07	1.9E+07	2.2E+07	2.5E+07	2.8E+07	3.1E+07	3.4E+07	3.6E+07
	Fast I	4.5E+06	7.8E+06	1.1E+07	1.4E+07	1.6E+07	1.8E+07	2.0E+07	2.2E+07	2.3E+07	2.5E+07	2.6E+07
Lighting2	Fast	2.4E+07	4.3E+07	6.0E+07	7.8E+07	9.5E+07	1.1E+08	1.3E+08	1.4E+08	1.6E+08	1.8E+08	1.9E+08
	Fast SC	2.2E+07	3.7E+07	5.2E+07	6.6E+07	7.9E+07	9.4E+07	1.1E+08	1.2E+08	1.3E+08	1.4E+08	1.5E+08
	Fast I	2.1E+07	3.5E+07	4.7E+07	5.8E+07	6.7E+07	7.6E+07	8.4E+07	9.6E+07	1.0E+08	1.1E+08	1.1E+08

with the slope of the term including the *radius* of (18), and the execution time of the proposed fast SC-DTW and fast I-DTW is lower than or similar to that of the fast DTW. In contrast, the execution time of the existing SC-DTW and I-DTW increases quadratically according to the data length. For cases in which the increase in execution time due to the data length is significantly greater than the increase in execution time due to the *radius* parameter, the proposed fast SC-DTW and fast I-DTW improve the execution time, similar to the fast DTW; however, the SC-DTW and I-DTW require higher execution times.

For dataset with short data lengths, when the increase in execution time due to the dataset length is smaller than the increase in execution time due to the *radius* parameter, the execution time of the fast DTW is not superior to that of the standard DTW. For a dataset with long data lengths, the SC-DTW and I-DTW show a quadratic execution time. For the both datasets with short and long data lengths, the exe-

TABLE 7. Rate of the number of calculations of fast DTW, fast SC-DTW and fast I-DTW compared to standard DTW according to *radius*.

Dataset	DTW	0	1	2	3	4	5	6	7	8	9	10
Synthetic-Control	Fast	15.7	26.9	36.8	44.9	52.6	62.1	67.1	71.8	76.3	80.6	100.0
	Fast SC	13.3	18.5	20.5	20.5	20.5	20.5	20.5	20.5	20.5	20.5	20.5
	Fast I	10.3	10.3	10.3	10.3	10.3	10.3	10.3	10.3	10.3	10.3	10.3
ECG200	Fast	9.9	17.4	24.0	30.3	36.7	41.8	46.8	51.6	59.5	62.7	65.9
	Fast SC	8.5	12.8	15.8	18.8	18.8	18.8	18.8	18.8	18.8	18.8	18.8
	Fast I	7.1	10.2	10.2	10.2	10.2	10.2	10.2	10.2	10.2	10.2	10.2
CBF	Fast	7.7	13.4	18.8	23.6	28.3	33.6	37.5	41.3	45.0	48.6	52.2
	Fast SC	6.7	10.5	13.7	15.8	18.6	18.6	18.6	18.6	18.6	18.6	18.6
	Fast I	6.2	8.2	10.2	10.2	10.2	10.2	10.2	10.2	10.2	10.2	10.2
SwedishLeaf	Fast	7.5	13.1	18.6	23.6	28.4	33.8	37.8	41.8	45.6	49.3	53.0
	Fast SC	6.8	11.0	14.0	16.4	18.6	18.6	18.6	18.6	18.6	18.6	18.6
	Fast I	6.1	8.1	10.2	10.2	10.2	10.2	10.2	10.2	10.2	10.2	10.2
Two-Patterns	Fast	7.6	13.1	18.5	23.3	27.9	33.1	37.0	40.7	44.4	48.0	51.4
	Fast SC	6.8	10.7	13.8	16.0	18.6	18.6	18.6	18.6	18.6	18.6	18.6
	Fast I	6.1	8.1	10.2	10.2	10.2	10.2	10.2	10.2	10.2	10.2	10.2
FaceAll	Fast	7.5	12.9	18.3	23.2	27.9	33.2	37.1	41.0	44.8	48.5	52.1
	Fast SC	6.7	11.0	14.2	16.7	19.6	19.6	19.6	19.6	19.6	19.6	19.6
	Fast I	5.8	8.2	10.2	10.2	10.2	10.2	10.2	10.2	10.2	10.2	10.2
Gun-Point	Fast	6.4	11.4	16.1	20.6	24.7	28.7	33.4	36.9	40.2	43.5	46.7
	Fast SC	5.8	9.2	12.3	14.2	16.1	19.6	19.6	19.6	19.6	19.6	19.6
	Fast I	5.0	7.2	8.6	10.1	10.1	10.1	10.1	10.1	10.1	10.1	10.1
Wafer	Fast	6.3	11.2	15.8	20.3	24.4	28.4	33.1	36.5	39.9	43.1	46.3
	Fast SC	5.6	8.9	12.0	13.9	15.7	19.4	19.4	19.4	19.4	19.4	19.4
	Fast I	4.9	7.2	8.7	10.1	10.1	10.1	10.1	10.1	10.1	10.1	10.1
Adiac	Fast	5.2	9.4	13.4	17.5	21.2	24.7	28.2	32.5	35.5	38.4	41.3
	Fast SC	4.9	8.5	11.7	13.8	16.0	18.0	18.9	18.9	18.9	18.9	18.9
	Fast I	4.4	6.9	8.6	10.1	10.1	10.1	10.1	10.1	10.1	10.1	10.1
Trace	Fast	3.6	6.4	9.1	11.7	14.1	16.5	19.0	21.2	23.4	25.5	27.6
	Fast SC	3.3	5.3	7.1	8.7	10.2	11.9	12.9	13.8	14.7	19.0	19.0
	Fast I	3.0	4.6	6.1	6.9	7.6	10.1	10.1	10.1	10.1	10.1	10.1
Coffee	Fast	3.3	6.0	8.5	11.1	13.5	15.9	18.4	20.6	22.8	25.0	27.2
	Fast SC	3.2	5.6	7.9	9.9	11.9	13.3	14.6	15.9	17.2	18.4	18.9
	Fast I	3.0	4.9	6.6	7.6	8.6	9.4	10.1	10.1	10.1	10.1	10.1
Lighting7	Fast	3.2	5.6	7.9	10.1	12.3	14.4	16.5	18.7	20.6	22.5	24.3
	Fast SC	2.8	4.7	6.5	8.0	9.4	11.5	12.4	13.3	14.1	15.0	15.8
	Fast I	2.7	4.2	5.4	6.7	7.4	8.0	10.1	10.1	10.1	10.1	10.1
FaceFour	Fast	2.9	5.1	7.2	9.3	11.3	13.3	15.3	17.4	19.3	21.1	22.8
	Fast SC	2.7	4.7	6.6	8.3	9.8	11.4	12.7	13.8	14.8	15.7	16.7
	Fast I	2.5	4.1	5.4	6.6	7.4	8.1	8.8	10.1	10.1	10.1	10.1
Yoga	Fast	2.4	4.2	5.9	7.6	9.4	11.0	12.6	14.2	15.8	17.6	19.1
	Fast SC	2.1	3.6	4.9	6.2	7.3	8.4	9.4	11.0	11.7	12.4	13.0
	Fast I	2.0	3.1	4.1	4.9	6.0	6.5	7.0	7.5	7.9	10.0	10.0
OSULeaf	Fast	2.4	4.2	6.0	7.7	9.4	11.0	12.7	14.3	15.8	17.6	19.1
	Fast SC	2.3	3.8	5.2	6.6	7.8	8.9	10.0	11.5	12.2	12.9	13.6
	Fast I	2.1	3.2	4.3	5.1	6.2	6.8	7.3	7.8	8.2	10.0	10.0
FISH	Fast	2.1	3.8	5.4	7.0	8.6	10.2	11.7	13.2	14.7	16.2	17.9
	Fast SC	2.0	3.6	5.1	6.5	7.8	9.0	10.2	11.4	12.5	13.3	14.1
	Fast I	1.9	3.1	4.2	5.1	6.2	6.8	7.4	7.9	8.4	8.9	10.0
Beer	Fast	2.1	3.8	5.4	7.0	8.6	10.1	11.6	13.1	14.5	15.9	17.6
	Fast SC	2.1	3.6	4.9	6.3	7.5	8.6	9.7	10.8	12.0	12.7	13.5
	Fast I	1.9	3.2	4.3	5.2	6.1	6.9	7.5	8.0	8.5	9.0	10.0
OliveOil	Fast	1.6	3.0	4.3	5.7	7.0	8.3	9.6	10.8	12.1	13.3	14.5
	Fast SC	1.6	2.9	4.2	5.4	6.5	7.6	8.6	9.6	10.6	11.6	12.4
	Fast I	1.5	2.7	3.7	4.7	5.5	6.3	7.0	7.5	8.0	8.5	8.9
Lighting2	Fast	1.6	2.9	4.0	5.2	6.4	7.5	8.6	9.8	10.8	11.9	13.0
	Fast SC	1.5	2.5	3.5	4.4	5.3	6.3	7.1	7.8	8.5	9.3	9.9
	Fast I	1.4	2.3	3.1	3.9	4.5	5.1	5.7	6.4	6.8	7.2	7.5
Avg	Fast	5.2	9.1	12.9	16.3	19.6	23.0	26.0	28.8	31.6	34.2	37.5
	Fast SC	4.7	7.4	9.7	11.4	12.9	14.1	14.7	15.3	15.8	16.5	16.8
	Fast I	4.1	5.8	7.1	7.8	8.3	8.7	9.1	9.3	9.5	9.8	9.9

cution time of the proposed fast SC-DTW is always lower than or similar to that of the algorithm (fast DTW or SC-DTW) with the lower execution time. Similarly, the execution time of the fast I-DTW is lower than or similar to that of the algorithm (fast DTW or I-DTW) with the lower execution time.

V. CONCLUSION

For obtaining a similarity measure between two time series, one of the most efficient approaches is DTW, which measures the similarity by finding the optimal warping path of all alignments for the two time series. However, DTW exhibits a quadratic computational complexity depending on

the length of the two time series. Aiming to reduce the computational complexity of DTW, the constrained DTW technique introduces global constraints using a window with a specific shape, while the data abstraction technique uses a data representation with a reduced dimensionality for the two time series.

Constrained DTW techniques include SC-DTW, which always forms a window of the same length based on index points that match on the time axis between the two time series, and I-DTW, which gradually increases the window length as the index value increases for the two time series. These constrained DTW techniques reduce the number of searched cells and achieve a higher classification accuracy than the standard DTW by preventing the pathological alignment problem. However, the constrained DTW uses a fixed window and presents a quadratic computational complexity.

The fast DTW technique utilizes multiple resolutions to estimate the approximate optimal warping path and then reduces the computational complexity by calculating only the area near the estimated optimal warping path. This fast DTW technique considers only the alignments within an adaptive window and has a linear computational complexity. However, if the increase in calculations due to the *radius* parameter is greater than the increase in calculations due to the data length, the fast DTW technique is not expected to reduce the calculations and will also have a lower overall classification accuracy than the constrained DTW techniques.

Accordingly, in this paper, we proposed a fast constrained DTW that applies an optimal warping path estimation method using the multi-resolution of the fast DTW within a window with a specific shape based on the constrained DTW technique. The time complexity of the proposed fast constrained DTW is linear and lower than or similar to those of the fast DTW and constrained DTW techniques, while achieving a classification accuracy similar to that of the constrained DTW technique.

We conducted experiments using 19 UCR time series datasets and synthetic datasets comprising a single period of a sine wave with various data lengths. These experiments demonstrated that the proposed fast I-DTW, in which the I-DTW window of constrained DTW techniques is applied, reduces the calculations by approximately 52.2% and 22.3% compared with the I-DTW and fast DTW in the UCR datasets. In addition, the classification accuracy of the proposed fast I-DTW is similar to that of the I-DTW, and its execution time exhibits linear behavior and is lower than or similar to those of the I-DTW and fast DTW for all sine wave datasets.

REFERENCES

[1] A. Fakhrazari and H. Vakilzadian, "A survey on time series data mining," in *Proc. IEEE Int. Conf. Electro Inf. Technol. (EIT)*, Lincoln, NE, USA, May 2017, pp. 476–481.

[2] T. Lampert, T.-B.-H. Dao, B. Lafabregue, N. Serrette, G. Forestier, B. Crémilleux, C. Vrain, and P. Gançarski, "Constrained distance based clustering for time-series: A comparative and experimental study," *Data Mining Knowl. Discovery*, vol. 32, no. 6, pp. 1663–1707, Nov. 2018.

[3] S. Aghabozorgi, A. Seyed Shirkhorshidi, and T. Ying Wah, "Time-series clustering—A decade review," *Inf. Syst.*, vol. 53, pp. 16–38, Oct. 2015.

[4] C. H. Fontes and O. Pereira, "Pattern recognition in multivariate time series—A case study applied to fault detection in a gas turbine," *Eng. Appl. Artif. Intell.*, vol. 49, pp. 10–18, Mar. 2016.

[5] T. Han, X. Liu, and A. C. C. Tan, "Fault diagnosis of rolling element bearings based on multiscale dynamic time warping," *Measurement*, vol. 95, pp. 355–366, Jan. 2017.

[6] G. Plouffe and A.-M. Cretu, "Static and dynamic hand gesture recognition in depth data using dynamic time warping," *IEEE Trans. Instrum. Meas.*, vol. 65, no. 2, pp. 305–316, Feb. 2016.

[7] Z. Shi, M. Xu, Q. Pan, B. Yan, and H. Zhang, "LSTM-based flight trajectory prediction," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2018, pp. 1–8.

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

[9] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychol. Rev.*, vol. 65, no. 6, p. 386, 1958.

[10] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[11] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.

[12] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986.

[13] R. Xie and J. Cao, "Accelerometer-based hand gesture recognition by neural network and similarity matching," *IEEE Sensors J.*, vol. 16, no. 11, pp. 4537–4545, Jun. 2016.

[14] M. C. Kwon, G. Park, and S. Choi, "Smartwatch user interface implementation using CNN-based gesture pattern recognition," *Sensors*, vol. 18, no. 9, pp. 1–12, 2018.

[15] M. Kim, J. Cho, S. Lee, and Y. Jung, "IMU sensor-based hand gesture recognition for human-machine interfaces," *Sensors*, vol. 19, no. 18, p. 3827, Sep. 2019.

[16] B. K. Iwana, V. Frinken, and S. Uchida, "DTW-NN: A novel neural network for time series recognition using dynamic alignment between inputs and weights," *Knowl.-Based Syst.*, vol. 188, Jan. 2020, Art. no. 104971.

[17] S. S. Yulin and I. N. Palamar, "Probability model based on cluster analysis to classify sequences of observations for small training sets," *Statist., Optim. Inf. Comput.*, vol. 8, no. 1, pp. 296–303, Feb. 2020.

[18] V. Niennattrakul and C. A. Ratanamahatana, "On clustering multimedia time series data using K-Means and dynamic time warping," in *Proc. Int. Conf. Multimedia Ubiquitous Eng. (MUE)*, 2007, pp. 733–738.

[19] H. Li, J. Liu, Z. Yang, R. W. Liu, K. Wu, and Y. Wan, "Adaptively constrained dynamic time warping for time series classification and clustering," *Inf. Sci.*, vol. 534, pp. 97–116, Sep. 2020.

[20] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[21] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 865–873, Apr. 2015.

[22] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 6645–6649.

[23] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proc. ICML*, 2014, pp. 1764–1772.

[24] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human trajectory prediction in crowded spaces," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 961–971.

[25] M. Sundermeyer, R. Schlüter, and H. Ney, "LSTM neural networks for language modeling," in *Proc. Interspeech*, 2012, pp. 194–197.

[26] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 157–166, Mar. 1994.

[27] Y. Wang, P. Lei, H. Zhou, X. Wang, M. Ma, and X. Chen, "Using DTW to measure trajectory distance in grid space," in *Proc. 4th IEEE Int. Conf. Inf. Sci. Technol.*, Apr. 2014, pp. 152–155.

[28] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh, "Querying and mining of time series data: Experimental comparison of representations and distance measures," in *Proc. VLDB Endowment*, 2008, vol. 1, no. 2, pp. 1542–1552.
[29] M. Zadghorban and M. Nahvi, "An algorithm on sign words extraction and recognition of continuous Persian sign language based on motion and shape features of hands," *Pattern Anal. Appl.*, vol. 21, no. 1, pp. 323–335, 2016.
[30] G. Nagendar and C. V. Jawahar "Efficient word image retrieval using fast DTW distance," in *Proc. 13th Int. Conf. Document Anal. Recognit. (ICDAR)*, 2015, pp. 876–880.
[31] N. T. Hai, N. Van Thuyen, T. T. Mai, and V. Van Toi, "MFCC-DTW algorithm for speech recognition in an intelligent wheelchair," in *Proc. 5th Int. Conf. Biomed. Eng. Vietnam*. Cham, Switzerland: Springer, Jan. 2015, pp. 417–421.
[32] I.-J. Ding and J. Y. Shi, "Kinect microphone array-based speech and speaker recognition for the exhibition control of humanoid robots," *Comput. Electr. Eng.*, vol. 62, pp. 719–729, Jul. 2017.
[33] A. Azmoodeh, A. Dehghantanha, M. Conti, and K.-K. R. Choo, "Detecting crypto-ransomware in iot networks based on energy consumption footprint," *J. Ambient Intell. Humanized Comput.*, vol. 9, no. 4, pp. 1–12, 2017.
[34] G. Cormode, "Fundamentals of analyzing and mining data streams," in *Proc. Tutorial Workshop Data Stream Anal.*, Caserta, Italy, 2007, pp. 1–5.
[35] L. Gupta, D. L. Molfese, R. Tammana, and P. G. Simos, "Nonlinear alignment and averaging for estimating the evoked potential," *IEEE Trans. Biomed. Eng.*, vol. 43, no. 4, pp. 348–356, Apr. 1996.
[36] Y. Gao, L. Xia, Y.-L. Gong, and D.-C. Zheng, "Electrocardiogram (ECG) patterns of left anterior fascicular block and conduction impairment in ventricular myocardium: A whole-heart model-based simulation study," *J. Zhejiang Univ.-Sci. B*, vol. 19, no. 1, pp. 49–56, Jan. 2018.
[37] E. Keogh, "Exact indexing of dynamic time warping," in *Proc. 28th Int. Conf. Very Large Data Bases*, 2002, pp. 406–417.
[38] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-26, no. 1, pp. 43–49, Feb. 1978.
[39] F. Itakura, "Minimum prediction residual principle applied to speech recognition," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-23, no. 1, pp. 67–72, Feb. 1975.
[40] H. Li and C. Wang, "Similarity measure based on incremental warping window for time series data mining," *IEEE Access*, vol. 7, pp. 3909–3917, 2019.
[41] E. J. Keogh and M. J. Pazzani, "Scaling up dynamic time warping for datamining applications," in *Proc. 6th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2000, pp. 285–289.
[42] A. Sharabiani, H. Darabi, S. Harford, E. Douzali, F. Karim, H. Johnson, and S. Chen, "Asymptotic Dynamic Time Warping calculation with utilizing value repetition," *Knowledge Inf. Syst.*, vol. 57, no. 2, pp. 1–30, 2018.
[43] S. Salvador and P. Chan, "FastDTW: Toward accurate dynamic time warping in linear time and space," in *Proc. KDD Workshop Mining Temporal Sequential Data*, 2004, pp. 70–80.
[44] C. Guo, H. Li, and D. Pan, "An improved piecewise aggregate approximation based on statistical features for time series mining," in *Proc. 4th Int. Conf. Knowl. Sci., Eng. Manage. (KSEM)*. Berlin, Germany: Springer-Verlag, 2010, pp. 234–244.
[45] L. Ge and S. Chen, "Exact dynamic time warping calculation for weak sparse time series," *Appl. Soft Comput.*, vol. 96, Nov. 2020, Art. no. 106631.
[46] C. A. Ratanamahatana and E. Keogh, "Three myths about dynamic time warping data mining," in *Proc. SIAM Int. Conf. Data Mining*, Apr. 2005, pp. 506–510.
[47] Y. Chen. (Jul. 2015). *The UCR Time Series Classification Archive*. [Online]. Available: http://www.cs.ucr.edu/~eamonn/time_series_data/
[48] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification and Scene Analysis*, vol. 3. New York, NY, USA: Wiley, 1973.

WONYOUNG CHOI (Graduate Student Member, IEEE) received the B.S. degree in electronics and information engineering from Korea Aerospace University, in 2019, where he is currently pursuing the M.S. degree in electronics and information engineering. His research interests include data mining, network on chip (NoC), restricted coulomb energy-neural network (RCE-NN), and system-on-chip (SoC) design.

JAECHAN CHO (Graduate Student Member, IEEE) received the B.S. and M.S. degrees from the School of Electronics and Information Engineering, Korea Aerospace University, Goyang, South Korea, in 2015 and 2017, respectively, where he is currently pursuing the Ph.D. degree. His research interests include the signal processing algorithm and VLSI implementation for an edge-AI and image processing system.

SEONGJOO LEE (Senior Member, IEEE) received the B.S., M.S., and Ph.D. degrees in electrical and electronic engineering from Yonsei University, Seoul, South Korea, in 1993, 1998, and 2002, respectively. From 2002 to 2003, he was a Senior Research Engineer with the IT SOC Research Center and the ASIC Research Center, Yonsei University. From 2003 to 2005, he was a Senior Engineer with the Core Tech Sector, Visual-Display Division, Samsung Electronics Company Ltd., Suwon, South Korea. He was a Research Professor with the IT Center and the IT SoC Research Center, Yonsei University, from 2005 to 2006. He is currently a Professor with the Department of Information and Communication Engineering, Sejong University, Seoul. His current research interests include system-on-chip (SoC) design for wireless communication systems, RADAR and LiDAR signal processing, and SoC design for image signal processing.

YUNHO JUNG (Senior Member, IEEE) received the B.S., M.S., and Ph.D. degrees in electrical and electronic engineering from Yonsei University, Seoul, South Korea, in 1998, 2000, and 2005, respectively. From 2005 to 2007, he was a Senior Engineer with the Communication Research Center, Wireless Device Solution Team, Telecommunication Network Division, Samsung Electronics Company Ltd., Suwon, South Korea. From 2007 to 2008, he was a Research Professor with the Institute of TMS Information Technology, Yonsei University. He is currently a Professor with the School of Electronics and Information Engineering, Korea Aerospace University, Goyang, South Korea. His current research interests include signal processing algorithm and system-on-chip (SoC) implementation for radar, wireless communication, and image processing systems.