Accurate Subsequence Matching on Data Stream under Time Warping Distance

Vit Niennattrakul, Dechawut Wanichsan, and Chotirat Ann Ratanamahatana

Department of Computer Engineering, Chulalongkorn University
Phayathai Rd., Pathumwan, Bangkok 10330 Thailand
{g49vnn,g49dwn,ann}@cp.eng.chula.ac.th

Abstract. Dynamic Time Warping (DTW) distance has been proven to work exceptionally well, but with higher time and space complexities. Particularly for time series data, subsequence matching under DTW distance poses a much challenging problem to work on streaming data. Recent work, SPRING, has introduced a solution to this problem with only linear time and space which makes subsequence matching on data stream become more and more practical. However, we will demonstrate that it may still give inaccurate results, and then propose a novel Accurate Subsequence Matching (ASM) algorithm that eliminates this discrepancy by using a global constraint and a scaling factor. We further demonstrate utilities of our work on a comprehensive set of experiments that guarantees an improvement in accuracy while maintaining the same time and space complexities.

Keywords: Subsequence Matching, Data Stream, Dynamic Time Warping Distance.

1 Introduction

Dynamic Time Warping (DTW) distance measure, the distance that is widely used in various time series mining tasks, especially in classification [1], is largely established as one of the most accurate methods in finding similarity between two time series data. Past research work [2] has confirmed that the DTW distance measure dominantly outperforms the traditional Euclidean distance metric in terms of accuracy since the DTW distance measure exploits dynamic programming that allows more flexibility in sequence alignments. However, it relatively requires much higher time and space complexities.

Especially in working on data streams, DTW at first seems unattainable. In subsequence matching problems, we try to find most similar subsequences to the query on a much longer candidate sequence, i.e., a data stream. A candidate sequence can either be a fixed-length sequence or an infinite streaming sequence, producing a much more challenging problem. A brute-force method for subsequence matching simply extracts every subsequence from the candidate sequence, and then these sequences are all compared with the query sequence in a similar fashion to the whole sequence matching approach. This clearly is impractical for large or streaming data.

SPRING [3], a recently proposed subsequence matching algorithm on a data stream under DTW distance, has been introduced to provide solutions for the best-matched query and range query. SPRING algorithm obtains the same retrieval results as the brute-force subsequence matching approach with an impressively low complexity. Unfortunately, we have discovered that SPRING may give undesired results. To illustrate our point, Figure 1 shows an example of this discrepancy in the SPRING algorithm that fails to accurately retrieve the best-matched subsequence.

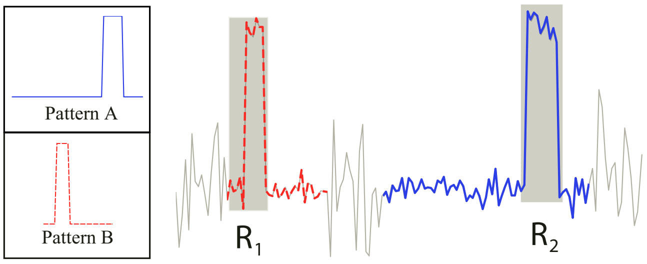

Fig. 1. Illustration of failure in retrieving accureate subsequences of SPRING

Given two pattern sequences of interest, A and B, which have similar shape, but with different lengths, onsets, and offset positions, we would like to find the best-matched subsequences to each pattern on a streaming sequence S. If we use pattern A as a query, R_1 (in left highlighted box) will be incorrectly retrieved as the best match, and both R_1 and R_2 (in right highlighted box) will be incorrectly detected in the range query. In fact, we expect that if we use pattern A as a query, only a blue solid sequence will be an answer for both range query and the best-matched query. On the other hand, if we use pattern B as a query, though R_1 will be incorrectly retrieved as the best match due to its shorter length, both R_1 and R_2 will again be incorrectly detected for the range query. The correct result is expected to be a red dashed-dot sequence. Although this example is a bit contrived, it clearly demonstrates undesired results from SPRING algorithm which is critical to achieving an accurate subsequence matching algorithm.

In this work, we introduce ASM – Accurate Subsequence Matching – which is fast and accurate. We extend an idea of linear-time subsequence matching from SPRING, and we generalize the subsequence matching to support a global constraint and uniform scaling, where SPRING algorithm is ASM's special case. Our algorithm is comprehensively examined over 10 datasets to demonstrate effectiveness of our proposed work comparing with the best existing method, SPRING.

The remainder of this paper is organized as follows. In Section 2, we state subsequence matching problems and provide essential background knowledge. We describe our proposed work, ASM, in Section 3, and then report experimental results and give discussion in Section 4, before concluding our work in Section 5.

2 Problem Definition and Background

In this section, we review related work, define problems for subsequence matching, and provide background knowledge of Dynamic Time Warping distance measure, global constraints, and SPRING algorithm.

2.1 Related Work

With proven superiority of DTW distance measure, many subsequent works have been proposed to speed up its calculations by exploiting various indexing techniques through the use of lower bounds [4,5,6,7,8]. However, all of these techniques are designed for non-streaming data. Therefore, the advent of research on data stream has triggered a great number of works [9,10]. But, only until recently, SPRING algorithm [3] has been introduced to solve the subsequence matching problem on streaming data under DTW distance.

After the introduction of SPRING, many extensions and its applications [11,12,13] have been proposed, including Fast Subsequence Matching (FSM) [13] and Embedded Subsequence Matching (EBSM) [11]. More specifically, FSM extends SPRING to further reduce unnecessary distance calculations, and EBSM computes an approximate distance for subsequence matching by modifying query data and the data stream in vectors before the actual DTW calculations. However, all these works mainly focus on speed of the calculation, but not the retrieval accuracy. Therefore, this work attempts to improve retrieval accuracy without affecting the time and space complexities of the algorithm. We will first start by familiarizing the readers with formal problem definitions and essential background.

2.2 Problem Definition

In this section, we formalize and define two fundamental types of query – non-overlapping range query and non-overlapping top-k query – that are essential for the subsequence matching problem and the rest of this work. Let Q be a fixed-length query sequence with length n, $S[t_s : t_e]$ be a subsequence of data stream S from time t_s to t_e, R be a global constraint, and $[n_{min}, n_{max}]$ be the sequence length ranging from a scaling range of $[f_{min} : f_{max}]$. This scaling range is a user-defined parameter that indicates possible lengths of a candidate subsequence, where $n_{min} = f_{min} \times n$, $n_{max} = f_{max} \times n$, and n is the query sequence's length. When no global constraint and scaling are applied, non-overlapping range query is equivalent to SPRING algorithm's, and non-overlapping top-k query is equivalent to the best-matched query in SPRING when $k = 1$. In this work, we define $D_R(X, Y)$ as the DTW distance with a global constraint R of data sequences X and Y. In typical subsequence matching problem, a subsequence from non-overlapping range query is reported when a local minimum-distance subsequence is found, and subsequences from non-overlapping top-k query are reported when a set of top-k subsequences is changed.

Definition 1 (Non-Overlapping Range Query). Non-overlapping range query returns a set $\Omega_{NORange}$ of non-overlapping subsequences $S[t_s : t_e]$ whose distance to a query sequence Q is less than a specific threshold ϵ under DTW with a global constraint R, and the length of a subsequence is between n_{min} and n_{max}.

Definition 2 (Non-Overlapping Top-k Query). Non-overlapping top-k query returns a set Ω_{NOTopK} of first k non-overlapping subsequences $S[t_s : t_e]$ with smallest distances resulted from non-overlapping range query with constrained DTW measure. The lengths of subsequences are also between n_{min} and n_{max}.

2.3 Dynamic Time Warping Distance Measure

Dynamic Time Warping (DTW) distance measure [14,15] is a well-known shape-based similarity measurement. It uses a dynamic programming technique to find an optimal warping path between two time series sequences. Suppose we have two time series, a sequence $X = \langle x_1, x_2, \ldots, x_i, \ldots x_n \rangle$ and a sequence $Y = \langle y_1, y_2, \ldots, y_j, \ldots y_m \rangle$. The distance is calculated by following equations.

$$D(X_{1\ldots n}, Y_{1\ldots m}) = d(x_n, y_m) + \min \begin{cases} D(X_{1\ldots n-1}, Y_{j\ldots m-1}) \\ D(X_{1\ldots n}, Y_{1\ldots m-1}) \\ D(X_{1\ldots n-1}, Y_{1\ldots m}) \end{cases} \tag{1}$$

where $D(\emptyset, \emptyset) = 0$, $D(X_{i\ldots n}, \emptyset) = D(\emptyset, Y_{j\ldots m}) = \infty$, and $\emptyset$ is an empty sequence. Any distance metrics can be used for $d(x_i, y_j)$, including L_1-norm, $d(x_i, y_j) = |x_i - y_j|$, and L_2-norm, $d(x_i, y_j) = (x_i - y_j)^2$. For simplicity, we use L_1-norm to describe our proposed method, but L_2-norm is used in experimental evaluation to achieve better accuracy.

However, in reality, DTW measure may not give the best alignment that fits our need as it tries its best to find a minimum distance, it may generate an unwanted path. Without a global constraint, DTW measure will find its optimal mapping between the two time series data. We can resolve this problem by simply limiting the permissible warping paths using a global constraint.

2.4 Global Constraints

Although unconstrained DTW distance measure gives as optimal distance between two time series data, an unwanted warping path may be generated. The global constraint [1,16,17] efficiently limits the optimal path to give a more suitable alignment. Recently, Ratanamahatana-Keogh band (R-K band), a general model of global constraints, has been proposed, as shown in Figure 2. It can be specified by a one-dimensional array R, i.e., $R = \langle r_1, r_2, \ldots, r_i, \ldots, r_n \rangle$, where n is the length of time series, and r_i is the height above the diagonal in y direction and the width to the right of the diagonal in x direction. Each r_i value is arbitrary; therefore, R-K band is also an arbitrary-shaped global constraint. Note that when $r_i = 0$, where $1 \leq i \leq n$, this R-K band represents the classic Euclidean distance, and when $r_i = n$, $1 \leq i \leq n$, this R-K band represents the original DTW distance without constraint.

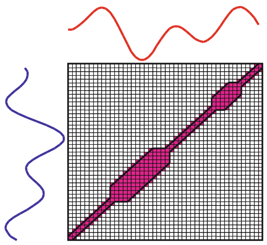

Fig. 2. An arbitrary-shaped global constraint, R-K band

2.5 SPRING Algorithm

SPRING algorithm [3], the first-proposed subsequence matching on data stream under DTW measure, can calculate optimal distance among subsequences in the data stream requiring $O(n)$ in both time and space complexities, where n is the length of a query sequence. SPRING is implemented based on two main ideas of Star-padding technique and STWM (Subsequence Time Warping Matrix). Star-padding is used to separate the overlapped subsequences, and STWM is a data structure that stores a minimum distance $d_{t,i}$ and a starting position $sp_{t,i}$. Suppose we have streaming sequence $S = \langle s_1, s_2, \ldots, s_t, \ldots \rangle$ and a query sequence $Q = \langle q_1, q_2, \ldots, q_i, \ldots, q_n \rangle$. At each time slice, new elements are calculated by following Equation 2 and Equation 3.

$$d_{t,i} = \|s_t - q_i\| + d_{best} \tag{2}$$

$$sp_{t,i} = \begin{cases} sp_{t-1,i-1} & \text{if } d_{best} = d_{t-1,i-1} \\ sp_{t,i-1} & \text{if } d_{best} = d_{t,i-1} \\ sp_{t-1,i} & \text{if } d_{best} = d_{t-1,i} \end{cases} \tag{3}$$

where $d_{t,0} = 0$, $d_{0,i} = \infty$, and $d_{best} = \min\{d_{t-1,i-1}, d_{t,i-1}, d_{t-1,i}\}$.

For more detail, a complete description of SPRING algorithm can be found in [3].

3 Accurate Subsequence Matching Algorithm

In this section, ASM (Accurate Subsequence Matching) algorithm and two important ideas, i.e., Scaled-Array (S-A) band and Modified Subsequence Matrix (MSM), are proposed. We describe S-A band and MSM in Sections 3.1 and 3.2, respectively. In Section 3.3, we introduce our novel subsequence matching algorithm, ASM, which supports two important features, i.e., a global constraint and scaling range.

3.1 Scaled-Array Band

The current global constraint representations are applicable only for a squared calculation matrix. Thus, we propose a Scaled-Array (S-A) band which can represent a global constraint as a single vector array A. More specifically, suppose we have an n-by-m path matrix, S-A band is defined as $A = \langle a_1, a_2, \ldots, a_i, \ldots, a_n \rangle = \langle (\alpha_1, \beta_1), (\alpha_2, \beta_2), \ldots, (\alpha_i, \beta_i), \ldots, (\alpha_n, \beta_n) \rangle$, where $1 \leq \alpha_i \leq \beta_i \leq m$ for all $1 \leq i \leq n$. Each element a_i in A collects a tuple of vector, i.e., a valid starting position α_i and a valid ending position β_i. A valid starting position is the smallest j that a cell (i, j) is a valid position within a global constraint, and a valid ending position is the largest j that a cell (i, j) is a valid position within a global constraint, where $1 \leq j \leq m$.

Since R-K band cannot be used as a global constraint for comparing time series data with different lengths, S-A band is proposed to represent global constraint. Suppose we have a query sequence with length n and a candidate sequence with length is a scaling range from f_{min} to f_{max}. Valid starting and ending positions (α_i, β_i) are defined in Equations 4 and 5.

$$\alpha_i = \arg\min \{k | k + r_k \geq i\} \times f_{min} \tag{4}$$

$$\beta_i = \begin{cases} (i + r_i) \times f_{max} & ; \text{if } i + r_i \leq n \\ (n - (i + r_i)) \times f_{max} & ; \text{otherwise} \end{cases} \tag{5}$$

where $0 < f_{min} \leq f_{max}$, $r_i \leq n$, $1 \leq i \leq n$, and $1 \leq \alpha_i \leq \beta_i \leq m$.

Note that a full global constraint and Euclidean distance are also defined when $\alpha_i = 1, \beta_i = \infty, 1 \leq i \leq n$, and $\alpha_i = i, \beta_i = i, 1 \leq i \leq n$, respectively.

3.2 Modified Subsequence Matrix

Modified Subsequence Matrix (MSM) is a data structure, derived from Subsequence Time Warping Matrix (STWM) in SPRING, where each element consists of four values, i.e., distance $d_{t,i}$, starting position $sp_{t,i}$, and positions x and y ($x_{t,i}$, $y_{t,i}$) on a global constraint S-A band. Each element (t, i) containing $d_{t,i}$, $sp_{t,i}$, $x_{t,i}$, and $y_{t,i}$ means that at this point, we have a valid subsequence from $sp_{t,i}$ that give the optimal distance $d_{t,i}$ at its coordinate ($x_{t,i}$,$y_{t,i}$) on a global constraint. A "valid" subsequence is defined as the subsequence that all coordinates ($x_{t,i}$, $y_{t,i}$) in its warping path from $sp_{t,i}$ to t are valid within the S-A band. Updating algorithm for an element in MSM is described in the next section.

3.3 Accurate Subsequence Matching

The basic idea behind ASM is the validation before an update of $d_{t,i}$ from $d_{t-1,i-1}$, $d_{t-1,i}$, or $d_{t,i-1}$ in MSM. We check the validity of global constraint position $(x_{t,i}, y_{t,i})$ from position $(x_{t-1,i-1} + 1, y_{t-1,i-1} + 1)$, $(x_{t-1,i} + 1, y_{t-1,i})$, and $(x_{t,i-1}, y_{t,i-1} + 1)$. If some positions $(x_{t-1,i-1}, y_{t-1,i-1})$, $(x_{t-1,i}, y_{t-1,i})$, or $(x_{t,i-1}, y_{t,i-1})$ make $(x_{t,i}, y_{t,i})$ invalid on the S-A band, these positions will not be selected in the calculation for d_{best}. Let $Q = \langle q_1, q_2, \ldots, q_i, \ldots, q_n \rangle$ be a query sequence, $S = \langle s_1, s_2, \ldots, s_t, \ldots \rangle$ be a streaming sequence, and $A = \langle (\alpha_1, \beta_1), (\alpha_2, \beta_2), \ldots, (\alpha_i, \beta_i), \ldots, (\alpha_n, \beta_n) \rangle$ be an S-A band with scaling ranges from f_{min} to f_{max}. We define function $v(x, y)$ to validate position (x, y) in the S-A band. It returns value 1 if (x, y) lies within the global constraint or $\alpha_y \leq x \leq \beta_y$; otherwise, it returns positive infinity. Four values at time t are updated according to the following equations.

$$d_{t,i} = \|q_i - s_t\| + d_{best} \tag{6}$$

$$sp_{t,i} = \begin{cases} sp_{t-1,i-1} & \text{if } d_{best} = d_{t-1,i-1} \\ sp_{t,i-1} & \text{if } d_{best} = d_{t,i-1} \\ sp_{t-1,i} & \text{if } d_{best} = d_{t-1,i} \end{cases} \tag{7}$$

$$x_{t,i} = \begin{cases} x_{t-1,i-1} + 1 & \text{if } d_{best} = d_{t-1,i-1} \\ x_{t,i-1} & \text{if } d_{best} = d_{t,i-1} \\ x_{t-1,i} + 1 & \text{if } d_{best} = d_{t-1,i} \end{cases} \tag{8}$$

$$y_{t,i} = \begin{cases} y_{t-1,i-1} + 1 & \text{if } d_{best} = d_{t-1,i-1} \\ y_{t,i-1} + 1 & \text{if } d_{best} = d_{t,i-1} \\ y_{t-1,i} & \text{if } d_{best} = d_{t-1,i} \end{cases} \tag{9}$$

where $d_{t,0}$ =0, $d_{0,i}$= ∞, and d_{best}=min $\begin{cases} d_{t-1,i-1} \times v(x_{t-1,i-1}+1, y_{t-1,i-1}+1) \\ d_{t-1,i} \times v(x_{t-1,i}+1, y_{t-1,i}) \\ d_{t,i-1} \times v(x_{t,i-1}, y_{t,i-1}+1) \end{cases}$.

As we can see, values of each element at time t depends only on the previous element values at time $t-1$. Therefore, in practice, only two arrays are required. We denote d_i, sp_i, x_i, and y_i as distance, starting point, x position, and y position at current time t and at the query sequence position i, and d'_i, sp'_i, x'_i, and y'_i as distance, starting point, x position, and y position at previous time $t-1$.

For non-overlapping range query, a subsequence $S[t_s : t_e]$ is considered a result when minimum DTW distance with S-A band between $S[t_s : t_e]$ and Q is less than a threshold ϵ, and the subsequence has a qualified length. In addition, ASM will report this subsequence when $\forall i, d_i \geq d_{min} \vee sp_i > t_e$, as shown in Table 1. It is important to note that SPRING algorithm is a special case of our ASM when $(\alpha_i, \beta_i) = (1, \infty)$ for all i in the S-A band and a scaling length is $[1, \infty]$, so the same results are returned when non-overlapping range query is issued.

Non-overlapping top-k query is used to monitor a set of k subsequences which has minimum distance among overlapped subsequences. Generally, non-overlapping top-k query is implemented on non-overlapping range query whose initial threshold ϵ be positive infinity. When optimal range subsequence is found, we push this subsequence into a distance-priority queue. If size of the queue exceeds k, we pop the maximum-distance subsequence, and reset threshold ϵ to be a maximum distance of the queue. ASM algorithm for top-k query is shown in Table 2.

We would like to emphasize that ASM always achieves higher accuracy (as will be shown in our experimental section) while maintaining the same time and space complexities as SPRING's, i.e., $O(n)$ for both space and time complexities at each time slice, where n is the length of a query sequence. Since ASM keeps a single matrix and a single array which are both length n, and updates $O(n)$ numbers every time slice, ASM requires only $O(n)$ both in space and time.

Table 1. ASM algorithm for optimal range query

```
Algorithm ASMOptimalRange
1  Input: new streaming data point $s_t$
2  Output: optimal subsequence $S[t_s : t_e]$, if any
3  Let $n$ be the length of a query sequence
4  $n_{min} = n \times f_{min}$, $n_{max} = n \times f_{max}$
5  for $i = 1$ to $n$ do
6      Compute $d_i$, $sp_i$, $x_i$, and $y_i$
7  endfor
8  if $d_{min} \leq \epsilon$
9     if $\forall i, d_i \geq d_{min} \vee sp_i > t_e$ then
10         Report($d_{min}$, $t_s$, $t_e$)
11         $d_{min} = \infty$
12         for $i = 1$ to $m$ do
13             if $sp_i \leq t_e$ then
14                 $d_i = \infty$
15     endif
16 endif
17 if $[d_m \leq \epsilon] \wedge [d_m < d_{min}] \wedge [n_{min} \leq x_m \leq n_{max}]$ then
19     $d_{min} = d_m$; $t_s = s_m$; $t_e = t$
20 endif
21 Substitute $d'_i$ for $d_i$, $sp'_i$ for $sp_i$, $x'_i$ for $x_i$, $y'_i$ for $y$;
```

Table 2. ASM algorithm for optimal top-k query

```
Algorithm ASMOptimalTopK
1  Input: new streaming data point $s_t$
2  Output: updated set $P$ of top $k$
3  $S[t_s : t_e]$= ASMOptimalRange($s_t$, $\epsilon$)
4  if ($S[t_s : t_e] \neq NULL$)
5      $P.push(S[t_s : t_e])$
6      if ($size(P) > k$)
7          $P.pop()$
8          $e = P.peek().distance$
9      endif
10     Report($P$)
11 endif
```

4 Experimental Evaluation

To evaluate the performance of our proposed method, we measure an accuracy of our algorithm comparing with the best existing algorithm, SPRING, using two evaluation metrics, i.e., Accuracy-on-Retrieval (AoR) and Accuracy-on-Detection (AoD).

4.1 Evaluation Metrics

We use two metrics, Accuracy-on-Retrieval (AoR) defined in Definition 3 and Accuracy-on-Detection (AoD) defined in Definition 4, to measure the quality of retrieval results on streaming sequences. Suppose we have a streaming sequence S, a set of expected pattern sequences E, and a set of retrieved sequences R. We first define an overlapping subsequence. Let $S[t_s : t_e]$ be the subsequence starting at t_s and ending at t_e. Overlapping subsequence $O_{X,Y}$ and overlap percentage $P_{X,Y}$, where $X = S[a : b]$ and $Y = S[c : d]$, are defined as follows.

$$O_{X,Y} = S[\max\{a,c\} : \min\{b,d\}] \tag{10}$$

$$P_{X,Y} = \frac{|O_{X,Y}|}{\max\{b,d\} - \min\{a,c\} + 1} \tag{11}$$

For instance, if we have subsequence $X = S[2 : 5]$ and $Y = S[3 : 7]$, $O_{X,Y} = S[3 : 5]$ and $P_{X,Y} = \frac{|S[3:5]|}{\max\{5,7\}-\min\{2,3\}+1} = \frac{3}{6} = 0.5$.

Definition 3 Accuracy-on-Retrieval. This evaluation measures how well an algorithm has *found* a set of expected subsequences while definition of *found* depends on overlapped percentage p. An extremely optimistic case is when p is 0, i.e., even subsequences are only one single data point overlapped, subsequence is marked as *found*. AoR is defined in Equation 12. Note that the higher the AoR, the better the result.

$$AoR = \frac{|\{O_{X,Y} | P_{X,Y} > p, X \in R, Y \in E\}|}{|E|} \tag{12}$$

Definition 4 Accuracy-on-Detection. This evaluation measures that once expected subsequences are *found*, as described in Definition 3, with overlapped percentage p, how well an algorithm can recognize these expected subsequences; in the other words, AoD is an average overlapping percentage $(P_{X,Y})$ of found sequences. AoD is defined in Equation 13. Again, the higher the AoD, the better the result.

$$AoD = \frac{\sum \{P_{X,Y} | P_{X,Y} > p, X \in R, Y \in E\}}{|\{O_{X,Y} | P_{X,Y} > p, X \in R, Y \in E\}|} \tag{13}$$

4.2 Datasets

To test the accuracy of stream monitoring, we assemble synthetic datasets based on UCR time series archive [18]. We use this archive because these datasets are labeled, and we can precisely calculate the accuracy. More specifically, we concatenate each time series sequence from a training dataset together, but each time series pattern is separated by normalized random walk data with twice the pattern's length to simulate real-world applications. For example, in CBF dataset [18], each training data has 128 data points. We concatenate random

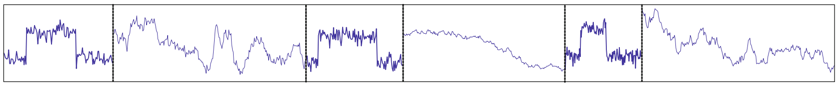

Fig. 3. Example of the data stream built from the CBF classification dataset

walk sequence of 256 data points to connect each and every sequence in the training data, as shown in Figure 3.

4.3 Experiments

All experiments are implemented in Java, and run on Linux Redhat 6.2 with Intel Xeon 3.2 GHz and 2 GB main memory. We test our ASM algorithm using non-overlapping top-k query on ten classification datasets from the UCR time series data mining archive. The k value is set to be the number of patterns within the data stream. We use a concatenated training dataset as a data stream, and use a test dataset as query sequences for evaluation. We then report AoR and AoD of our proposed method comparing with SPRING algorithm, as shown in Table 3.

Table 3. Our experiment result outperforms SPRING in both AoR and AoD

Dataset	AoR		AoD		ASM Parameters	
	SPRING	ASM	SPRING	ASM	$[f_{min} : f_{max}]$	Global Constraint
Synthetic Control	73.82%	74.64%	58.33%	76.85%	[0.6:1.4]	12%
Gun Point	44.31%	53.21%	50.90%	80.11%	[0.7:1.3]	0%
CBF	74.03%	78.10%	60.65%	74.76%	[0.7:1.3]	14%
Trace	72.84%	77.99%	33.21%	76.41%	[0.7:1.3]	50%
Face Four	57.23%	63.15%	55.77%	83.81%	[0.8:1.2]	0%
Lighting 7	47.57%	52.16%	35.41%	89.61%	[0.8:1.2]	0%
ECG 200	60.66%	64.94%	62.87%	88.26%	[0.9:1.1]	0%
Beef	33.89%	38.33%	36.11%	88.94%	[0.9:1.1]	24%
Coffee	79.59%	80.10%	49.23%	94.51%	[0.9:1.1]	2%
Olive Oil	69.76%	72.59%	72.59%	89.69%	[1:1]	0%

4.4 Discussion

From our experiment, we can see that ASM achieves higher accuracies both in terms of Accuracy-on-Retrieval (AoR) and Accuracy-on-Detection (AoD) comparing with the best similarity subsequence matching on data stream under DTW distance, SPRING, as expected. Since SPRING finds minimum distance from all possible subsequences of a data stream, it tries to match the query with a subsequence that gives minimum distance. Therefore, shorter subsequences are preferred. If we have different classes of two patterns as shown in Figure 1, SPRING cannot distinguish. It still reports wrong subsequences, although the size of a query sequence and a retrieved subsequence greatly differ. ASM has great flexibility on limiting the unwanted warping path. Scaling range is also needed since patterns in streaming data are unpredictable. Additionally, S-A band makes a global constraint support sequences of different length.

5 Conclusion

In this work, we have illustrated that the current subsequence matching algorithm (SPRING) on data stream under time warping distance is somewhat inaccurate, and may give undesired results. We then introduce a novel Accurate Subsequence Matching algorithm that has SPRING algorithm as its special case. With the use of a global constraint and scaling range, our experiments have shown to improve the retrieval accuracy on every dataset by a wide margin, while being able to maintain both time and space complexities of $O(n)$.

Acknowledgement

This research is partially supported by the Thailand Research Fund given through the Royal Golden Jubilee Ph.D. Program (PHD/0141/2549 to V. Niennattrakul).

References

1. Ratanamahatana, C.A., Keogh, E.J.: Making time-series classification more accurate using learned constraints. In:Proceedings of 4th SIAM International Conference on Data Mining (SDM 2004), Lake Buena Vista, Florida, USA, April 22-24, pp. 11–22 (2004)
2. Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., Keogh, E.: Querying and mining of time series data: Experimental comparison of representations and distance measures. In: Proceedings of 34th International Conference on Very Large Data Bases (VLDB 2008), Auckland, New Zealand, August 23 - 28 (2008)
3. Sakurai, Y., Faloutsos, C., Yamamuro, M.: Stream monitoring under the time warping distance. In: Proceedings of IEEE 23rd International Conference on Data Engineering (ICDE 2007), Istanbul, Turkey, April 15-20, pp. 1046–1055 (2007)
4. Keogh, E., Ratanamahatana, C.A.: Exact indexing of dynamic time warping. Knowledge and Information Systems 7(3), 358–386 (2005)
5. Kim, S.W., Park, S., Chu, W.W.: An index-based approach for similarity search supporting time warping in large sequence databases. In: Proceedings of the 17th International Conference on Data Engineering (ICDE 2001), Heidelberg, Germany, April 2-6, pp. 607–614 (2001)
6. Yi, B.K., Jagadish, H.V., Faloutsos, C.: Efficient retrieval of similar time sequences under time warping. In: Proceedings of 14th International Conference on Data Engineering (ICDE 1998), Orlando, FL, USA, February 23-27, pp. 201–208 (1998)
7. Zhu, Y., Shasha, D.: Warping indexes with envelope transforms for query by humming. In: Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data (SIGMOD 2003), San Diego, CA, USA, June 9-12, pp. 181–192 (2003)

8. Sakurai, Y., Yoshikawa, M., Faloutsos, C.: FTW: Fast similarity search under the time warping distance. In: Proceedings of 24th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, Baltimore, ML, USA, June 13-15, pp. 326–337 (2005)
9. Zhu, Y., Shasha, D.: Statstream: Statistical monitoring of thousands of data streams in real time. In: Proceedings of 28th International Conference on Very Large Data Bases (VLDB 2002), Hong Kong, China, August 20-23, pp. 358–369 (2002)
10. Wei, L., Keogh, E.J., Herle, H.V., Mafra-Neto, A.: Atomic wedgie: Efficient query filtering for streaming times series. In: Proceedings of the 5th IEEE International Conference on Data Mining (ICDM 2005), Houston, TX, USA, November 27-30, pp. 490–497 (2005)
11. Athitsos, V., Papapetrou, P., Potamias, M., Kollios, G., Gunopulos, D.: Approximate embedding-based subsequence matching of time series. In: Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD 2008), Vancouver, BC, Canada, June 10-12, pp. 365–378 (2008)
12. Yueguo, C., Shouxu, J., Beng Chin, O., Tung, A.K.H.: Querying complex spatio-temporal sequences in human motion databases. In: Proceedings of IEEE 24th International Conference on Data Engineering (ICDE 2008), Cancún, México, April 7-12, pp. 90–99 (2008)
13. Zou, P., Su, L., Jia, Y., Han, W., Yang, S.: Fast similarity matching on data stream with noise. In: Proceedings of the 24th International Conference on Data Engineering Workshops (ICDEW 2008), Cancún, México, April 7-12, pp. 194–199 (2008)
14. Berndt, D.J., Clifford, J.: Using dynamic time warping to find patterns in time series. In: The 1994 AAAI Workshop on Knowledge Discovery in Databases, Seattle, Washington, July 1994, pp. 359–370 (1994)
15. Ratanamahatana, C.A., Keogh, E.J.: Three myths about dynamic time warping data mining. In: Proceedings of 2005 SIAM International Data Mining Conference (SDM 2005), Newport Beach, CL, USA, April 21-23, pp. 506–510 (2005)
16. 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)
17. Itakura, F.: Minimum prediction residual principle applied to speech recognition. IEEE Transactions on Acoustics, Speech, and Signal Processing 23(1), 67–72 (1975)
18. Keogh, E., Xi, X., Wei, L., Ratanamahatana, C.A.: UCR time series classification/clustering page, `http://www.cs.ucr.edu/~eamonn/time_series_data`