Fast Subsequence Query Algorithm Based on Short Sequence Grouping and Concatenation Strategy (Postprint)

Authors: Fan Chunlong, Jingyun Wang, Teng Yiping, Ding Guohui

Date: 2019-05-10T00:00:00+00:00

Abstract

Subsequence query technology finds important applications in finance, commerce, healthcare, and other domains. However, the high time complexity of similarity comparison algorithms such as DTW (dynamic time warping) means that subsequence length exerts a substantial influence on retrieval time, thereby limiting the efficiency of long subsequence retrieval on datasets. To address this issue, we propose a fast subsequence query algorithm. The algorithm first groups all subsequences of a specific length within the dataset and identifies representative subsequences; subsequently, during query processing, it partitions the query sequence into fixed-length small segments and employs the DTW algorithm to determine a candidate set of representative subsequences similar to these segments; finally, it performs sequence concatenation on the candidate set to obtain the query result sequence. Experimental results demonstrate that the proposed algorithm achieves approximately a tenfold improvement in efficiency compared to typical algorithms.

Full Text

Preamble

Vol. 37 No. 6
Application Research of Computers
ChinaXiv Cooperative Journal

Fast Subsequence Query Algorithm Based on Short Sequence Grouping and Assembling Strategy

Fan Chunlong[1,2], Wang Jingyun[1,2], Teng Yiping[1,2], Ding Guohui[1,2]

[1]School of Computer, Shenyang Aerospace University, Shenyang 110136, China;
[2]Large-scale Distributed System Laboratory in Liaoning Province, Shenyang 110136, China

Abstract: Subsequence query technology has important applications in finance, commerce, healthcare, and other fields. However, due to the high time complexity of similarity comparison algorithms such as Dynamic Time Warping (DTW), subsequence length significantly impacts retrieval time, limiting the efficiency of long subsequence retrieval on datasets. To address this problem, we propose a fast subsequence query algorithm. The algorithm first groups all subsequences of a specific length in the dataset and marks representative subsequences for each group. During query processing, the query sequence is split into fixed-length small segments, and DTW is used to identify candidate sets of representative subsequences similar to these segments. Finally, sequence assembly is performed on the candidate sets to obtain the query result sequences. Experiments demonstrate that the new algorithm achieves approximately 10× efficiency improvement compared to typical algorithms.

Keywords: sequential data query; dynamic time warping; subsequence

0 Introduction

Time series data refers to ordered data lists formed over time, commonly called time series. Such data reflects the state of objects or events as they evolve, with each time point represented by numerical or symbolic values. Time series data widely exists in finance, commerce, medicine, meteorology, and other application domains. Data mining on time series has extensive applications in stock price prediction, genetic material analysis, weather forecasting, and related areas. Sequence data query is a fundamental and important problem in time series mining, and this paper focuses on querying subsequences of varying lengths in large-scale time series data.

Distance measurement forms the foundation of sequence query technology. Euclidean Distance (ED) is one of the most commonly used methods due to its computational simplicity and efficiency. Other mainstream distance metrics include Dynamic Time Warping (DTW), Longest Common Subsequence (LCSS), Edit Distance on Real sequence (EDR), and Edit distance with Real Penalty (ERP). These methods exhibit strong robustness, supporting temporal shifting and calculations on unequal-length sequences.

Time series representation is the primary approach for improving query efficiency. Time series data is typically high-dimensional and large-volume. For a dataset containing N time series, each of length n, the total number of subsequences to compute is $N \times n \times (n-1)/2$. For example, the StarLightCurves benchmark dataset from the UCR time series collection contains 9,236 sequences, each of length 1,024, resulting in 4.83×10 subsequences. Real-world datasets are of-

ten three orders of magnitude larger, making similarity comparisons across all subsequences impractical. To avoid performance degradation from excessive data volume, dimensionality reduction is necessary. Common representation methods include Discrete Fourier Transform (DFT), Discrete Wavelet Transform (DWT), Singular Value Decomposition (SVD), and Piecewise Aggregate Approximation (PAA). While these preserve most feature information from original sequences, they do not prioritize efficiency and thus are unsuitable for large datasets.

Range search and nearest neighbor search are the main objectives of sequence query. Clustering methods identify representatives for each class and utilize them for querying, including approaches such as Nearest Centroid Classifier and K-means clustering. Reference [14] employs data editing to maintain original data for accurate similarity queries while improving classification speed. Reference [15] introduces DTW-based clustering and improved density peak algorithms to enhance clustering performance. However, these methods target clustering rather than similarity query, which does not align with our current work.

State-of-the-art query algorithms such as Online Exploration of Time Series (ONEX) also face a trade-off between accuracy and query efficiency, particularly struggling to return results quickly for long subsequence queries. Some systems sacrifice time for improved accuracy, limiting their practicality. References [18] and [19] convert subsequence queries to vector queries using DTW, but this approach requires numerous parameters, constraining query efficiency.

This paper improves existing query algorithms by proposing MONEX (Modified Online Exploration of Time Series), a method for fast long subsequence querying. MONEX divides candidate sequences into subsequences and encodes similarity relationships among them to support rapid queries. Since encoding similarity relationships across all possible subsequence lengths is infeasible, we select a specific length for subsequence segmentation and employ lower-complexity point-to-point distances (e.g., Euclidean distance) for similarity encoding. During the query phase, the method splits the query sequence to ensure fast long subsequence retrieval, employs an improved DTW algorithm for similarity computation, and finally assembles similar sequence segments into complete sequences.

MONEX improves upon the state-of-the-art ONEX algorithm, achieving nearly 10× query efficiency improvement and 0.03% accuracy improvement. While both algorithms exhibit similar accuracy, MONEX demonstrates significant improvements in both accuracy and efficiency for long subsequence queries.

---

1 Basic Definitions and Framework

1.1 Fundamental Definitions

Definition 1 (Time Series). A time series is denoted as X = (x , x , ···, x), representing a sequence of n real values. D denotes a time series dataset, representing a collection of N time series.

Given the high dimensionality and large volume of real-world datasets, direct analysis is often impractical. Instead, time series are typically split into shorter subsequences for analysis.

Definition 2 (Time Series Subsequence). Given a time series X = (x , x , ···, x), a subsequence of length i starting at position j is denoted as X[i,j] = (x , x , ···, x), where 1 i n and 0 j n−i.

During queries, similarity between two sequences is generally determined using a predefined similarity threshold.

Definition 3 (Similarity Threshold). Two time series X and Y are considered similar if their Euclidean or DTW distance is less than a given similarity threshold ST, expressed as Dist(X,Y) ST, where Dist {ED, DTW}.

Query algorithms require appropriate distance metrics. This paper employs both Euclidean distance and DTW distance. However, due to ubiquitous noise in time series collection, normalized distances are necessary.

Definition 4 (Normalized Euclidean Distance). For two sequences X and Y, the normalized Euclidean distance is defined as:

$$ED(X,Y) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - y_i)^2}$$

Definition 5 (Normalized Dynamic Time Warping Distance). For two time series X and Y of length m and n respectively, the normalized DTW distance is defined as:

$$DTW(X,Y) = \frac{1}{2n}\sum_{i=1}^{2n}|x_i - y_i|$$

Lemma 1. For two subsequences X and Y in the same query space similarity group, their normalized Euclidean distance is always less than or equal to the threshold, i.e., $ED(X,Y) \leq ST$, where $X, Y \in G_k$.

Proof. Given two equal-length subsequences X = (x , ···, x) and Y = (y , ··· , y) from the same query space similarity group, and the group representative R = (r , ···, r), Definition 7 implies $ED(X,R) \leq \frac{ST}{2}$ and $ED(Y,R) \leq \frac{ST}{2}$. Substituting into Definition 4, we need to prove:

$$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - y_i)^2} \leq ST$$

Squaring both sides yields:

$$\frac{1}{n}\sum_{i=1}^{n}(x_i - y_i)^2 \leq ST^2$$

Using the Cauchy-Schwarz inequality, we obtain:

$$|x_i - y_i| \leq |x_i - r_i| + |r_i - y_i|$$

Squaring and summing gives:

$$\sum_{i=1}^{n}(x_i - y_i)^2 \leq \sum_{i=1}^{n}(x_i - r_i)^2 + \sum_{i=1}^{n}(r_i - y_i)^2 + 2\sum_{i=1}^{n}|x_i - r_i||r_i - y_i|$$

Combining with Definition 4 and the earlier inequalities, we get:

$$\frac{1}{n}\sum_{i=1}^{n}(x_i - y_i)^2 \leq \frac{ST^2}{4} + \frac{ST^2}{4} + \frac{ST^2}{2} = ST^2$$

Thus, the inequality holds, proving $ED(X, Y) \leq ST$.

Definition 6 (Representative Sequence). Given a query space similarity group S, the pointwise average of all sequences in S is defined as the representative of S, denoted as R, where for all $X_{i,j}^p \in S$, $R = \text{avg}(X_{i,j}^p)$.

Definition 7 (Query Space Similarity Group). For a dataset D, all subsequences of specific length length are placed in set T. Suppose these subsequences are distributed into similarity groups, each with its own representative R_k, such that all subsequences belong to exactly one query space similarity group, denoted as G_k. Query space similarity groups must satisfy two conditions:

a) For any subsequence $X_{i,j}^p$ and its representative R_k, the Euclidean distance is less than or equal to half the similarity threshold:

$$ED(X_{i,j}^p, R_k) \leq \frac{ST}{2}$$

b) For any subsequence $X_{i,j}^p$ and any other representative R_l (where $l \neq k$), the Euclidean distance to its own representative is minimal:

$$ED(X_{i,j}^p, R_k) \leq ED(X_{i,j}^p, R_l)$$

1.2 Time Series Online Query Space Similarity Groups

The core of MONEX is constructing time series online query space similarity groups. First, we establish the triangle inequality between ED and DTW to enable efficient querying.

Definition 8 (Representative Space). Given a dataset D, the space composed of representatives R_k from each query space similarity group G_k and their associated subsequences is called the representative space, denoted as R-space.

Lemma 2. Given $X_{i,j}^p$ as any subsequence in similarity group G_k with representative R_k, and Q as a query sequence, if $DTW(Q, R_k) \leq \frac{ST}{2}$, then $DTW(Q, X_{i,j}^p) \leq ST$.

Proof. From Definition 3 and Lemma 1, we have:

$$DTW(Q, X_{i,j}^p) \leq DTW(Q, R_k) + DTW(R_k, X_{i,j}^p) \leq \frac{ST}{2} + \frac{ST}{2} = ST$$

Squaring both sides yields:

$$DTW^2(Q, X_{i,j}^p) \leq ST^2$$

Since DTW computation requires matrix construction, we consider matrices M(Q,R_k) and M(Q,X_{i,j}^p). In matrix M(Q,R_k), the warping path P runs from (1,1) to (n,n), so the DTW weight is at most $2n \cdot ST$. We must show that in matrix M(Q,X_{i,j}^p), any warping path from (1,1) to (n,n) has weight at most $2n \cdot ST$.

Given a warping path $P = (p_1, p_2, ..., p_T)$ in M(Q,R_k) where $p_t = (i_t, j_t)$, Lemma 2' s condition implies:

$$\sum_{t=1}^{T} |q_{i_t} - r_{j_t}| \leq 2n \cdot ST$$

Squaring and applying the inequality yields:

$$\sum_{t=1}^{T} (q_{i_t} - x_{i_t})^2 \leq 25n \cdot ST^2$$

Since $25n \cdot ST^2$ is always less than or equal to $4n^2 \cdot ST^2$, the inequality holds, proving $DTW(Q, X_{i,j}^p) \leq ST$.

1.3 Time Warping Retrieval Based on ED-DTW Triangle Inequality

Our time series online query framework is built upon the triangle inequality relationship between ED and DTW distances. This relationship ensures that if a query sequence is similar to a representative of a query space similarity group, it is similar to all subsequences in that group. This enables querying in the compressed R-space rather than comparing against the entire dataset, significantly improving efficiency. Specifically, if the normalized DTW distance between a query sequence and a representative is less than or equal to half the similarity threshold, then all subsequences in that similarity group are similar to the query sequence, with normalized DTW distances less than or equal to ST.

---

2 MONEX Algorithm

2.1 Algorithm Overview

MONEX performs a complete query through three steps: constructing query space similarity groups, computing representative distances, and assembling subsequences:

a) **Preprocessing:** Candidate sequences are split into all possible subsequences of length length. Based on similarity relationships among these subsequences, query space similarity groups are formed, and each group' s representative is computed and stored in the representative space R-space. The query sequence is also split into small segments of length length.

b) **Candidate Retrieval:** Each small query segment is compared against group representatives using DTW to identify the most similar representative and its corresponding query space similarity group for each segment.

c) **Sequence Assembly:** The identified similarity groups are arranged according to their corresponding query segment order. Subsequences from these groups that can be concatenated into continuous sequences are assembled.

2.2 Query Space Similarity Group Construction Algorithm

This algorithm identifies similarity groups for subsequences of length length and obtains each group' s representative.

a) Decompose all time series in dataset D into subsequences of length length. To eliminate bias from continuous data, use RANDOMIZE-IN-PLACE to randomly permute the subsequences.

b) Designate the first subsequence in the permuted set as the first group member and its representative. Random permutation ensures groups are independent of the input order.

c) Compare remaining subsequences against existing representatives by computing Euclidean distances. Assign each subsequence to the group with the minimum distance if $ED \leq \frac{ST}{2}$; otherwise, create a new group with the subsequence as its representative. Repeat until all subsequences are assigned.

Algorithm 1: Query Space Similarity Group Construction
Input: Dataset D, similarity threshold ST, specific length i
Output: Representatives {R}, similarity groups {G}

```
Begin
    Randomize subsequences {X} = all subsequences of length i
    minSM = 0, mink = 0, members[] = 0
    if (G =  ) then
        G  ← {X_{i,j}^p}
```

```
        members[1] ← {index of subsequence in group}
        R ← X_{i,j}^p
    for each remaining subsequence X_{i,j}^p do
        for k = 1 to Representative.count do
            minSM = ED(R_k, X_{i,j}^p)
            mink = Representative index
        if (minSM   ST/2) then
            update G_mink ← G_mink   {X_{i,j}^p}
            members[mink] ← members[mink]   {index}
        else
            K ← K + 1
            G_K ← {X_{i,j}^p}
            R_K ← X_{i,j}^p
```

Complexity Analysis: The complexity of constructing query space similarity groups is $O(ngl)$, where n is the number of time series in the dataset, g is the number of query space similarity groups, and l represents the number of subsequences. Typically, initial time series have fixed lengths, and during grouping, these sequences are segmented, so the length cannot be considered infinite. The number of sequences n varies with dataset size and is usually much larger than l. Thus, l can be treated as a constant relative to n, reducing complexity to $O(ng)$.

The value of g can be analyzed probabilistically. Assuming k existing similarity groups and a new time series X, there are (k+1) possibilities: X joins one of the k existing groups or forms a new (k+1)th group. With equal probability for each event, the probability of X forming a new group is $\frac{1}{k+1}$. The expected number of new groups follows a geometric distribution with value $\sum_{k=1}^{g-1} \frac{1}{k+1} = O(\log g)$. Therefore, the complexity becomes $O(n \log g)$, and the overall algorithm complexity is $O(n \log n)$.

2.3 Multi-Similarity Parameter Configuration

As described in Section 2.2, the representative space is built using specific thresholds and subsequence lengths. Since analysts may have different similarity requirements, threshold ST and length length must be adjustable.

Definition 9 (Representative Distance). In R-space, the Euclidean distance between any two representatives R_i and R_k is defined as the representative distance: $D_c(R_i, R_k) = ED(R_i, R_k)$.

Definition 10 (Similarity Space). SP-Space is a conceptual space where subsequences have specific length length and similarity threshold ST.

Based on similarity thresholds, corresponding similarity spaces can be established for each specific length. Generally, if $ST \geq D_c + ST'$, two similarity groups merge into a new group. We define two metrics: ST_{half} and ST_{final}. For length length, when half of the similarity groups merge, the threshold is

ST_{half}; when all groups merge, the threshold is ST_{final}. For example, with specific length i, if $ST_{half} = 0.5$ and $ST_{final} = 0.78$, then when the new threshold equals 0.5, half the groups merge; when it reaches 0.78 or higher, all groups merge.

For long subsequence queries, decomposing candidate sequences into all possible lengths is impractical. We set specific length length to five values: 10, 20, 50, 80, and 100. For each length, compute corresponding ST_{half} and ST_{final}. Based on these parameters, similarity levels are classified as:

a) **Strict level:** $ST < ST_{half}$

b) **Intermediate level:** $ST_{half} \leq ST \leq ST_{final}$

c) **Fuzzy level:** $ST > ST_{final}$

For example, if an analyst requests a "strict" level similarity threshold range for subsequence length i, the recommended value falls within [0, 0.6]. Any value selected in this interval returns "strict" similarity results, with lower values producing stricter similarity within each query space similarity group.

2.4 Query Process Based on Query Space Similarity Groups

This paper studies user-driven queries where analysts provide target query sequences. Two query types exist:

a) The system returns time series most similar to the user-provided query sequence with matching length. For example, querying stocks with fluctuations similar to Apple stock during a specific period, where the query sequence exists in the dataset.

b) Users design desired patterns and query for the best match in the dataset, which may not exist exactly but returns the closest sequence.

For long subsequence queries, MONEX improves efficiency by splitting the query sequence into equal-length segments, querying each short segment, then assembling the results into sequences matching the original length.

Definition 11 (Time Series Segment). A query sequence Q of length n is split into m segments of length i: $Q = \{Q_1, Q_2, ..., Q_m\}$, where m is a positive integer and $m = \lfloor n/i \rfloor$.

As introduced in Section 1.2, query space similarity groups form compact clusters based on ED, and the query system applies time warping strategies on these groups:

a) Split query sequence Q into segments of length length, ignoring any final segment shorter than length.

b) Compute DTW distance between each segment Q_q and each group representative R_k, storing the minimum distance index in array numbest[q]

and retrieving group indices.

c) Determine if subsequences in group G_{num_k} can be concatenated with subsequences in group $G_{num_{k+1}}$, finding candidate sequences that can form continuous sequences of the query length.

d) For each assembled sequence, retrieve the precomputed Euclidean distances between its subsegments and their group representatives (calculated during group construction), selecting the sequence with minimum total distance as the final result.

Algorithm 2: Query Processor
Input: Query sequence Q, specific length length, array members
Output: Assembled sequences {X}

```
Begin
    m = n / i, numk = 0
    /* Split Q into m segments of length i */
    for q = 1 to m do
        /* Compute DTW distance between Q_q and each representative */
        numk = index of best distance
        numbest[q] = numk
        R_numk = representative of group G_numk
    for k = 1 to m-1 do
        /* Check if subsequences in G_numk and G_numk+1 can be concatenated */
        if (members[numk][end] + 1 == members[numk+1][start]) then
            candidate_sequences ← concatenate matching subsequences
    Select sequence with minimum total ED to representatives as result
```

3 Experimental Results and Analysis

We evaluate MONEX on real datasets, comparing query time and accuracy against two methods: (a) Standard DTW, which computes distances between all subsequences and the query to guarantee optimal results; (b) ONEX, which splits sequences into all possible lengths, applies clustering for dimensionality reduction, and queries the reduced data.

The experimental environment is Visual Studio 2010 with C++, running on Windows 7 with Intel Core i5-3470 3.2 GHz and 4 GB RAM. We select datasets from the largest public time series collection UCR: Face(four), Symbols, MALLAT, ECG5000, and HandOutlines, containing 8,400, 39,800, 56,320, 70,000, and 1,002,330 data points respectively. For accurate results, each sequence is normalized by scaling to [0,1] using $x_i' = (x_i - \min)/(\max - \min)$.

3.1 Query Time Comparison

We compare MONEX against standard DTW and ONEX across different dataset sizes and query sequence lengths. For dataset size tests, we use average response time over 20 queries. To test both in-dataset and out-of-dataset queries, we extract 10 subsequences of varying lengths from the candidate dataset and 10 from other datasets, forming a query set of 20 sequences. Each subsequence is queried 5 times, with the average taken as its query time. The overall query time is the average across all 20 sequences.

MONEX significantly outperforms standard DTW, so Figure 1 [Figure 1: see original paper] uses logarithmic scale for time (ms). MONEX consistently outperforms standard DTW by several orders of magnitude and achieves shorter response times than ONEX. For small datasets, ONEX and MONEX show comparable performance, but as dataset size increases, the difference becomes pronounced, with MONEX achieving approximately 10× speedup.

To evaluate query sequence length impact, we extract 20 subsequences from HandOutlines with lengths ranging from 20 to 400. As shown in Figure 2 [Figure 2: see original paper], query time differences become more apparent with increasing length, but MONEX consistently outperforms ONEX, with greater advantages for longer sequences.

3.2 Query Accuracy Comparison

Since DTW computes the minimum distance between sequences, we use standard DTW as the evaluation benchmark. MONEX accuracy is assessed by how closely its returned distances approximate standard DTW distances. Following reference [21] on path accuracy and reference [7] on query accuracy, we define:

$$\text{Accuracy} = \left(1 - \frac{1}{n}\sum_{i=1}^{n}\frac{\text{apprDist}_i - \text{optimDist}_i}{\text{optimDist}_i}\right) \times 100\%$$

where optimDist is the distance returned by standard DTW (optimal), and apprDist is the distance from MONEX. Since optimDist is the minimum possible distance, apprDist optimDist, making accuracy 0.

Tables 1 and 2 show query accuracy across datasets and sequence lengths. MONEX achieves 97.36% average accuracy, outperforming ONEX by 0.03% on average. While both show similar precision, MONEX demonstrates significantly improved accuracy for long subsequence queries.

3.3 Preprocessing Evaluation

We evaluate preprocessing time and space overhead under varying similarity thresholds. Figure 3 [Figure 3: see original paper] shows offline construction

time for query space similarity groups. Lower thresholds require longer construction times due to more groups. As ST increases, construction time decreases until reaching a stable point where all subsequences merge.

Figure 4 [Figure 4: see original paper] shows that larger similarity thresholds produce fewer representatives in R-space. Table 3 details the number of representatives, subsequences, and space overhead for threshold 0.2 across datasets.

---

4 Conclusion

This paper proposes MONEX, an improved time series online query algorithm based on ONEX. Experiments demonstrate that as dataset size increases, MONEX achieves 10× speedup over ONEX. With increasing query sequence length, MONEX' s performance advantage becomes more pronounced while maintaining 0.03% higher accuracy.

Currently, the most similar subsequence must match the query sequence length, which may limit accuracy in practice where optimal matches might have different lengths. Future work will explore alternative segmentation strategies to handle variable-length queries.

References

[1] Hinich, Melvin J. Time series analysis by state space methods [J]. Technometrics, 2005, 47 (3): 373-373.

[2] Eduardo J R, Vagelis H, Carlos C, et al. Correlating financial time series with micro-blogging activity [C]// Proc of the 5th ACM International Conference on Web Search and Data Mining. California: ACM Press, 2012: 513-522.

[3] 范剑青, 姚琦伟. 非线性时间序列: 建模、预报及应用 [M]. 北京: 高等教育出版社, 2005: 1-7. (Fan Jianqing, Yao qiwei. Nonlinear time series: modeling, forecasting and application [M]. Beijing: Higher Education Press, 2005: 1-7.)

[4] Stefan A, Athitsos V, Das G. The move-split-merge metric for time series [J]. IEEE Trans on Knowledge and Data Engineering, 2013, 25 (6): 1425-1438.

[5] Khatua M, Safavi S H, Cheung N M. Sparse laplacian component analysis for Internet traffic anomalies detection [J]. IEEE Trans on Signal and Information Processing Over Networks, 2018, PP (99): 1-1.

[6] Fu W C, Keogh E, Lau L Y, et al. Scaling and time warping in time series querying [J]. Vldb Journal, 2008, 17 (4): 899-921.

[7] Dau H A, Keogh E. Matrix profile V: a generic technique to incorporate domain knowledge into motif discovery [C]// Proc of the 23rd ACM SIGKDD

International Conference on Knowledge Discovery and Data Mining. New York: ACM Press, 2017: 125-134.

[8] Sharif M, Lujo B, Michael K. R. On the suitability of Lp-Norms for creating and preventing adversarial examples [C]// Proc of IEEE CVPRW. Salt Lake City: IEEE Press, 2018: 1686-16888.

[9] Hui Y, Yufang W, Yuan G. Research on similarity mining for flight data [C]// Proc of the 2nd International Workshop on Education Technology and Computer Science. [S. l.]: IEEE Press, 2010: 150-153.

[10] Khandelwal I, Satija U, Adhikari R. Efficient financial time series forecasting model using DWT decomposition [C]// Proc of IEEE CONECCT. [S. l.]: IEEE Press, 2015: 1-5.

[11] Wang Y, Wang P, Pei J, et al. A data-adaptive and dynamic segmentation index for whole matching on time series [J]. Very Large Database Endowment, 2013, 6 (10): 793-804.

[12] Cunningham J P, Yu B M. Dimensionality reduction for large-scale neural recordings [J]. Nature Neuroscience, 2014, 17 (11): 1500-9.

[13] Petitjean F, Forestier G, Webb G I, et al. Faster and more accurate classification of time series by exploiting a novel dynamic time warping averaging algorithm [J]. Knowledge and Information Systems, 2016, 47 (1): 1-26.

[14] Begum N, Ulanova L, Wang J, et al. Accelerating dynamic time warping clustering with a novel admissible pruning strategy [C]// Proc of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. [S. l.]: ACM Press, 2015: 49-58.

[15] Neamtu R, Ahsan R, Rundensteiner E, et al. Interactive time series exploration powered by the marriage of similarity distances [J]. Very Large Database Endowment, 2016, 10 (3): 169-180.

[16] François P, Forestier G, Webb G I, et al. Faster and more accurate classification of time series by exploiting a novel dynamic time warping averaging algorithm [J]. Knowledge and Information Systems, 2016, 47 (1): 1-26.

[17] Athitsos V, Papapetrou P, Potamias M, et al. Approximate embedding-based subsequence matching of time series [C]// Proc of ACM SIGMOD. Athens: ACM Press, 2008: 365-378.

[18] Neamtu R, Ahsan R, Rundensteiner E, et al. Interactive time series exploration powered by the marriage of similarity distances [J]. Very Large Database Endowment, 2016, 10 (3): 169-180.

[19] Rakthanmanon T, Campana B, Mueen A, et al. Searching and mining trillions of time series subsequences under dynamic time warping [C]// Proc of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. California: ACM Press, 2012: 262-270.

[20] Nagendar G, Jawahar C V. Efficient word image retrieval using fast DTW distance [C]// Proc of the 13th International Conference on Document Analysis and Recognition. Hyderabad: IEEE Press, 2015: 1-5.

[21] Moradi H R, Furuichi S, Minculete N. Estimates for tsallis relative operator entropy [J]. Mathematical Inequalities and Applications, 2017, 20 (4): 1079-1088.

Note: Figure translations are in progress. See original paper for figures.