A Fast Semi-Supervised Clustering Framework for Large-Scale Time Series Data

Guoliang He, Yanzhou Pan, Xuewen Xia, Jinrong He, Rong Peng, and Neal N. Xiong

***Abstract*—Semi-supervised clustering algorithms have several limitations: 1) the computation complexity of them is very high, because calculating the similarity distances of pairs of examples is time-consuming; 2) traditional semi-supervised clustering methods have not considered how to make full use of must-link and cannot-link constraints. In the clustering, the contribution of a few pairwise constraints to the clustering performance is very limited, and some may negatively affect the outcome; and 3) these methods are not effective to handle high dimensional data, especially for time series data. Up to now, few work touched semi-supervised clustering on time series data. To efficiently cluster large-scale time series data, we first tackle contract time series clustering to produce the most accurate clustering results under a contracted time. We propose a semi-supervised time series clustering framework (STSC), which integrates a fast similarity measure and a constraint propagation approach. Based on the proposed framework, two valid semi-supervised clustering algorithms including fssK-means and fssDBSCAN are designed. Experiments on 11 datasets show that our proposed method is efficient and effective for clustering large-scale time series data.**

***Index Terms*—Constraint propagation, semi-supervised learning, similarity measure, time series clustering.**

I. Introduction

LARGE volumes of time series data ubiquitously exist in a very broad range of fields, such as biology [1], geology [2], medicine [3], finance [4], engineering [5], and so on. In recent years, as developments in computer technology have provided the basic infrastructure for fast access to vast amounts of time series data, time series data mining has become one of the most popular branches [6]–[8]. It can discover useful implied information and knowledge of the data, which can provide reasonable theoretical basis and technical support for decision analysis [9]–[11].

In the data mining communities of time series, clustering is an important issue to divide a given data into a set of nonoverlapping clusters in some natural way without prior knowledge. Due to its applicability, clustering has been widely studied in the past two decades and a number of approaches have been proposed, such as unsupervised subsequence learning method [12]–[14], model-based clustering [15], shape-based algorithms [16], fuzzy algorithms [17], and consistent algorithms [18]. To improve the accuracy of clustering, by converting pairwise constraints provided by experts into prior information, semi-supervised clustering is a popular way and many methods have been advanced [19]–[23]. However, due to the high-dimension and complexity of time series, few work touched semi-supervised clustering on time series data. Moreover, effectively comparing the similarity distances of pairs of examples is critical, which could greatly affect the clustering results. Up to now, lots of effective similarity approaches have been advanced for time series data, such as Euclidean distance (ED) [24], dynamic time warping (DTW) [25], edit distance [26], area-based shape distance measure [27], set-based similarity measure [28], and adaptive similarity measure selection strategy [29]. Among all similarity measures, DTW is a competitive and popular method for time series data.

Although most of the state-of-the-art clustering algorithms obtained good results on lots of datasets based on DTW, they still have several limitations.

1) The computation complexity of these approaches is very high. One reason is that calculating the similarity distances of pairs of examples in a large-scale data is time-consuming.
2) Conventional semi-supervised clustering methods did not consider how to make full use of must-link and cannot-link constraints. Limited by human resources, the amount of pairwise constraints provided by experts for semi-supervised clustering is usually small. In the clustering, the contribution of a few pairwise constraints to the performance is very limited, and some may negatively affect the outcome.
3) These methods are not effective to handle high dimensional data, especially for time series data.

In order to address the limitations of conventional clustering algorithms, in this article, we first tackle contract time series clustering, where we would like to produce the most accurate clustering results under a contracted time. We propose

Manuscript received May 14, 2019; accepted July 18, 2019. This work was supported by the National Natural Science Foundation of China under Grant 61876136 and Grant 61663009. This article was recommended by Associate Editor G. Nicosia. *(Corresponding authors: Guoliang He; Xuewen Xia.)*

G. He and R. Peng are with the School of Computer Science, Wuhan University, Wuhan 430079, China (e-mail: glhe@whu.edu.cn; 2015301500167@whu.edu.cn; rongpeng@whu.edu.cn).

Y. Pan is with the Engineering Department, Rice University, Houston, TX 77005 USA (e-mail: 2015301500167@whu.edu.cn).

X. Xia is with the College of Physics and Information Engineering, Minnan Normal University, Zhangzhou 363000, China (e-mail: xwxia@whu.edu.cn).

J. He is with the College of Mathematics and Computer Science, Yan'an University, Yan'an 716000, China (e-mail: hejinrong@nwafu.edu.cn).

N. N. Xiong is with the Department of Mathematics and Computer Science, Northeastern State University, Tahlequah, OK 74464 USA (e-mail: xiongnaixue@gmail.com).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMC.2019.2931731

a semi-supervised time series clustering framework (STSC), which integrates a fast similarity measure and a constraint propagation approach (CPA). Compared with traditional semi-supervised clustering methods, STSC has following properties.

1) An approximate similarity measure is introduced to improve the efficiency of clustering without sacrificing the accuracy. In the clustering, we only need to determine which one of the time series examples is more similar to a given example instead of accurately calculating the similarity distances between them.
2) To make best use of the pairwise constraints provided by experts, we advance a valid CPA to disseminate the pairwise constraints. Next, based on STSC, we design two semi-supervised clustering algorithms including fast semi-supervised *K*-means algorithm (fssK-means) and fast semi-supervised DBSCAN algorithm (fssDBSCAN).
3) We evaluate proposed algorithms on eleven time series datasets. Experiments show that our methods can drastically reduce the computation time of clustering with the good clustering results. To the best of our knowledge, the proposed clustering method is the first one to combine these two issues simultaneously.

The contribution of this article is fourfold. First, a fast similarity measure strategy is introduced to evaluate the approximate similarity between pairs of time series examples without deteriorating the clustering results. Second, a CPA is advanced to make full use of pairwise constraints provided by experts. Third, we first tackle contract time series clustering and propose a semi-supervised clustering framework to reduce the time complexity of clustering and improve the clustering accuracy. Fourth, based on the proposed clustering framework, we design two clustering algorithms fssK-means and fssDBSCAN.

The remainder of this article is organized as follows. Section II introduces the related work. Section III advances our proposed fast semi-supervised clustering framework with a fast similarity measure and a pairwise CPA. In Section IV, fssK-means clustering and fssDBSCAN clustering are designed based on our framework. Section V is about experiments and Section VI provides the conclusion and future work. Table I clarifies the abbreviations and symbols in this article, helping keep track of symbols' meaning.

TABLE I
SYMBOL TABLE

DTW	dynamic time warping similarity measure
ED	Euclidean Distance similarity measure
fDTW	a fast DTW (fDTW) similarity measure
aDTW	an approximate DTW between two samples
CPA	a constraint propagation approach
STSC	a semi-supervised time series clustering framework
fssK-means	a fast semi-supervised K-means algorithm with fDTW and CPA
fssK-means (DTW)	fssK-means with DTW instead of fDTW
K-means (ED)	traditional K-means with ED
fssK-means (withoutCPA)	fssK-means without CPA
fssDBSCAN	a fast semi-supervised DBSCAN algorithm with fDTW and CPA
fssDBSCAN (DTW)	fssDBSCAN with DTW instead of fDTW
DBSCAN(ED)	traditional DBSCAN with ED
fssDBSCAN (withoutCPA)	fssDBSCAN without CPA

II. RELATED WORK

Semi-supervised clustering is a hot topic in machine learning research. Compared with traditional clustering methods, semi-supervised clustering tries to make use of side-information provided by experts, and transforms these knowledge into pairwise constraints to improve the clustering performance. In the past few years, scientists have studied semi-supervised clustering from various perspectives. For instance, Yu *et al.* proposed an incremental ensemble framework for semi-supervised clustering in high dimensional feature spaces. They made use of the advantage of the random subspace technique and incremental ensemble member selection process to perform high dimensional data [19]. Anand *et al.* [30] introduced a semi-supervised framework for kernel mean shift clustering that uses only pairwise constraints to guide the clustering procedure. Liu *et al.* [31] proposed a semi-supervised linear discriminant clustering, which considers clustering and dimensionality reduction simultaneously by connecting *K*-means and linear discriminant analysis. Xiong *et al.* [32] studied an iterative active learning framework to select pairwise constraints for semi-supervised clustering and proposed a novel method for selecting the most informative queries. Wang *et al.* [33] proposed a constraint neighborhood projection method to conduct semi-supervised clustering with fewer labeled data points and deal with constraint conflicts. Huang *et al.* introduced a technique named constraint co-projections for semi-supervised co-clustering, which makes use of the pair-wise constraints and constraint projections. Their method can perform the object constraint projections and feature constraint projections simultaneously [34]. However, as human resources are costly and the side-information provided by experts is relatively limited, performance improvement of clustering is undesirable. To make best use of pairwise constraints provided by experts, Yu *et al.* [35] proposed a transitive closure-based CPA to improve the clustering results.

Since time series data has high dimension and much noise, up to now just a few works touched semi-supervised clustering on time series data. For instance, Zhou *et al.* [36] advanced a weighted semi-supervised normalized cut algorithm (wSSNCut), which incorporates multiple distance measures and constraints. To improve time series clustering, Dau *et al.* [37] introduced a novel semi-supervised technique to set the best warping window width ω of DTW similarity measure. However, few work studied the neighbor space of time series examples in pairwise constraints to make full use of the prior knowledge provided by experts.

Meanwhile, to enhance clustering results, effectively evaluating the similarity between pairs of examples is critical. As time series data being high-dimensional, it is difficult to effectively measure the similarity between two examples due to the curse of dimensionality. Time series clustering critically depends on the choice of similarity measure and it is generally believed that the choice of distance measure is even more important than the clustering algorithm itself [8], [38]. Recently, a number of researchers have explored and introduced many similarity measures for time series data, such as ED, DTW, or its variations (e.g., DDTW, WDTW, and cDTW) [39]–[41], ERP [26], PCA [42], move-split-merge (MSM) [43], and shape-based distance [16], etc. Among all similarity measures, ED and DTW are competitive and popular methods. However, the drawback of ED is that it needs the compared time series with the same length. DTW could measure two examples with different length based on dynamic programming because it could wrap the time to minimize the distance of two examples.

Although DTW is widely used in measuring similarity of time series and can produce good results in many datasets, its prominent shortage is high computational cost [2]. In order to accelerate the calculation of DTW, a few strategies have been advanced. For example, LB_Kim [44] and LB_Keogh [25] were proposed as two lower bounds (LBs) of DTW to speed up the sequential scan search for a query. Salvador and Chan [45] proposed a method called "FastDTW" to avoid the brute-force dynamic programming approach in the calculation process of DTW. When using their method, DTW can be calculated within $O(n)$ but with a high constant factor. To speed up clustering process, Zhu *et al.* [46] combined approximate DTW (aDTW) with anytime algorithm and proposed anytime clustering algorithm (ACA). In ACA users may interact with the clustering and examine an answer at any time. As time goes, the similarity distance between any two examples will be updated by the ground-truth DTW in the end if ACA is not interrupted. However, the aDTW in the ACA clustering process could sacrifice the clustering performance to some degree.

In addition, many approaches have been advanced to speed up the clustering process. For instance, Mai *et al.* [47] proposed an active density-based clustering algorithm, which works under a restricted number of used pairwise similarities. It adaptively selects the most informative pairwise lower-bounding similarities to update with real ones to reconstruct the result until the budget limitation is reached. This method also sacrifices the clustering performance to some degree. Later, they advanced an efficient anytime density-based clustering algorithm (AnyDBC) for very large complex datasets by reducing both the range query and the label propagation time of DBSCAN [48]. AnyDBC not only allows user interaction but also can be used to obtain good approximations under arbitrary time constraints. Based on sampling, Ding *et al.* [49] proposed a novel end-to-end time series clustering algorithm (YADING) for large-scale time series data with fast performance.

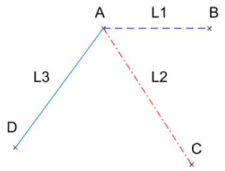


Fig. 1. Example of the fast similarity measure, where L1, L2, and L3 are the similarity distances between A and B, A and C, and A and D, respectively.

III. Proposed Semi-Supervised Clustering Framework

As we mentioned in the above, few existing semi-supervised clustering algorithms consider how to efficiently and effectively handle time series datasets. To address these two issues simultaneously, we make full use of prior knowledge provided by experts as well as reducing the computation time of clustering process to enhance the clustering results on time series data.

A. Fast Similarity Measure

We notice that the ultimate goal of clustering is dividing the time series data into several groups. To speed up the clustering process, the similarity of two individuals does not always have to be measured accurately. For instance, A, B, C, and D points and their locations are given in Fig. 1. We see that an approximate estimation is enough to figure out which one of B and C is closer to A because distance(A, B) differs greatly from distance(A, C). Only when two distances are almost the same, such as distance(A, C) and distance(A, D), do we need to calculate two distances accurately to make the decision about whether C or D is closer to A.

Based on this idea, the approximate similarity measure (ASD) between any two examples could be fast obtained by estimation. Specifically, $\mathrm{ASD}(X, Y)$ between two examples X and Y could be evaluated by its upper bound (UB) and LB of the ground-truth similarity distance (TSD) $\mathrm{TSD}(X, Y)$ as follows:

$$\mathrm{TSD}(X, Y) \approx \mathrm{ASD}(X, Y) = f(\mathrm{LB}(X, Y), \mathrm{UB}(X, Y)) \quad (1)$$

where $f(\bullet)$ is a function. In the process of clustering, we generally use ASD to fast evaluate the similarity between any two examples instead of its ground-TSD. When the difference between two ASDs is too small as shown in (2), it is difficult to distinguish which one (X or Y) is closer to the given example Z [similar to ASD(A, C) and ASD(A, D) in Fig. 1]

$$|\mathrm{ASD}(X, Z) - \mathrm{ASD}(Y, Z)| < \delta. \quad (2)$$

In this case, both $\mathrm{ASD}(X, Z)$ and $\mathrm{ASD}(Y, Z)$ should be updated to the ground-TSD.

Based on this idea, we can apply it to many similarity measures for clustering. Since DTW is difficult to beat among lots of similarity measures for time series data, its prominent shortage is time-consuming. To speed up the clustering without scarifying its performance, here we present a fast DTW (fDTW) similarity measure. It includes two stages: first, an aDTW between two samples is estimated based on its upper

Algorithm 1 Calculation Approach of fDTW

```
1  Require
2    Input:
3    The dataset to be clustered
4    β: the parameter used to estimate the DTW
5    δ: the threshold parameter
6  Ensure
7      fDTW(xi,xj) = Computing_aDTW(xi,xj)
8    If fDTW(xi,xj) satisfy the condition (2)
       then fDTW(xi,xj) = DTW(xi,xj)
9  Output: fDTW(xi,xj)
```

and LBs in the clustering process. Next, an updating principle is adopted to update the approximate similarity distance between two examples with ground-TSD when the approximate similarity distance cannot determine which one of two is much closer to another example. The basic process of our fDTW strategy is described in Algorithm 1.

In Algorithm 1, fDTW first estimates the aDTW distance between any two examples in the dataset (line 7). If aDTW distance does not satisfy the need in the process of clustering, it is updated by the ground-truth DTW distance (line 8).

To fast evaluate the aDTW similarity between two examples, we adopt the upper and LBs of DTW [46]. Its definition is as follows:

$$\mathrm{aDTW}(X,Y) = \mathrm{LB}_{\mathrm{DTW}}(X,Y) + \beta \times (\mathrm{UB}_{\mathrm{DTW}}(X,Y) - \mathrm{LB}_{\mathrm{DTW}}(X,Y)) \quad (3)$$

where X and Y are two examples of length n, fDTW stands for fDTW, $\mathrm{UB}_{\mathrm{DTW}}$ stands for the UB of DTW, and $\mathrm{LB}_{\mathrm{DTW}}$ stands the LB of DTW. They are defined as following:

$$\mathrm{UB}_{\mathrm{DTW}}(X,Y) = \mathrm{ED}(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad (4)$$

$$\mathrm{LB}_{\mathrm{DTW}}(X,Y) = \sqrt{\sum_{i=1}^{n}\begin{cases}(x_i - U_i)^2, & \text{if } x_i > U_i\\ (x_i - L_i)^2, & \text{if } x_i < L_i\\ 0, & \text{otherwise}\end{cases}} \quad (5)$$

where U_i and L_i, respectively, represent the maximum value and minimum value of time series Y in the warping range r, they form a bounding envelope that encloses Y from above and below

$$\begin{cases} U_i = \max\left(Y_{(i-r):(i+r)}\right) \\ L_i = \min\left(Y_{(i-r):(i+r)}\right). \end{cases} \quad (6)$$

The time complexity required to calculate the UB and LB are both $O(n)$. Hence, the time complexity of fDTW is also $O(n)$.

B. Constraint Propagation Approach

In semi-supervised clustering, it is expected that the clustering result could be improved as pairwise constraints are added. Theoretically, with the addition of human-provided side-information, the rand index (RI) of the clustering result should grow monotonously, but the actual situation is not as expected. Actually, the actual improvement of clustering results cannot meet the desired effect. With a few pairwise constraints, the improvement of the clustering result is not obvious and there are some fluctuations.

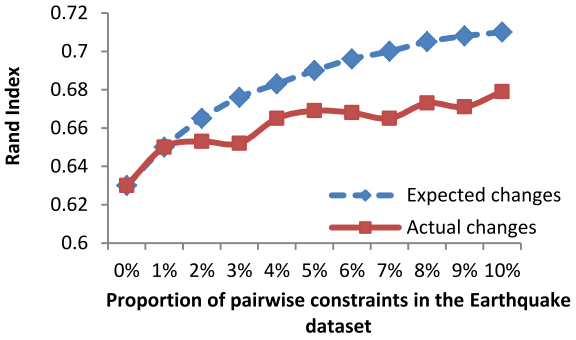


Fig. 2. Trend of clustering performance in Earthquake dataset as the number of pairwise constraints increases gradually.

To illustrate this phenomenon, an experiment is carried out on the Earthquake dataset. The experimental result is shown in Fig. 2. Ideally, it is expected that the RI of the clustering result would grow smoothly as more and more pairwise constraints are gradually added. However, this is not the case in fact. Sometimes there is even a reverse effect that causes RI to go down. We think the main reason of the fluctuation in performance is that pairwise constraints provided by experts are not full used, since implicit information of these constraints cannot be partitioned into the clustering.

To make full use of pairwise constraints provided by experts, we propose a CPA to disseminate the pairwise constraints. In fact, the examples in the pairwise constraints provided by experts and the examples closely related to them can be considered together. Before introducing CPA, we first give some concepts.

Definition 1: Given the dataset D, for a time series example q, its reverse nearest neighbors (RNNs), i.e., $\mathrm{RNN}(q)$ is a subset of D. Each one in $\mathrm{RNN}(q)$ treat q as its nearest neighbor [50]. The formal formula of $\mathrm{RNN}(q)$ is as follows:

$$\mathrm{RNN}(q) = \{p \in D | \forall r \in D : \mathrm{Distance}(p,q) < \mathrm{Distance}(p,r)\}. \quad (7)$$

According to the RNNs of an example, pairwise constraints have following transitivity properties.

Lemma 1: Consider an example q and its $\mathrm{RNN}(q)$, we could generate some pairwise constraints must-link(q, r), where $r \in \mathrm{RNN}(q)$.

Lemma 2: Given two pairwise constraints must-link(p, q) and must-link(q, r), based on the transitivity, we can generate a pairwise constraint must-link(p, r).

Lemma 3: Given a pairwise constraint must-link(p, q), we can generate pairwise constraints must-link(c, q) and must-link(p, r), where $c \in \mathrm{RNN}(p)$ and $r \in \mathrm{RNN}(q)$.

Lemma 4: Given a pairwise constraint cannot-link(p, q), we can generate pairwise constraints cannot-link(p, r) and cannot-link(c, q), where $c \in \mathrm{RNN}(p)$ and $r \in \mathrm{RNN}(q)$.

Solution 1: Integration. Given the dataset D and pairwise constraints P_0, based on CPA described in Lemmas 1–3, we can build a few core groups, and in each group D_i, any pair of two examples is a musk-link constraint. That is to say,

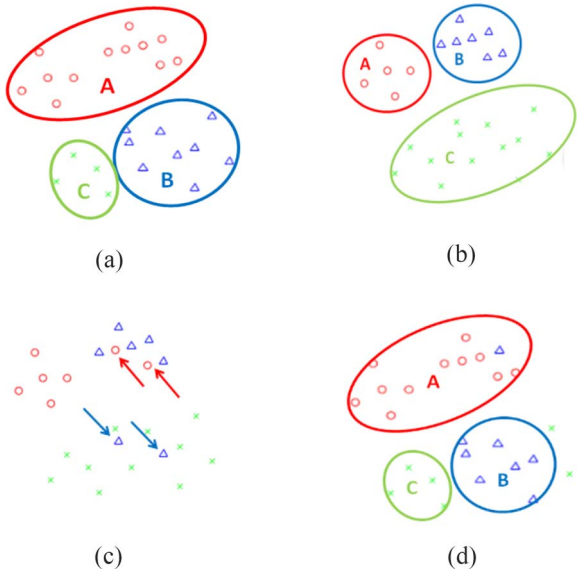


Fig. 3. Effect of the pairwise constraints for clustering on a simple dataset. (a) Real distribution of a simple dataset. (b) Clustering result with k-means. (c) Clustering result with k-means-based pairwise constraints provided by experts. (d) Clustering result with k-means-based pairwise constraints provided by experts and CPA.

examples in each group have the same label, and they should be divided into a cluster.

Solution 2: Segmentation. Given the dataset D and pairwise constraints P_0, based on CPA described in Lemmas 1 and 4, we can build a few core groups. In any two different group D_i and D_j, where $a \in D_i$ and $b \in D_j$, there is a pairwise constraint cannot-link (a, b). That is to say, all these core groups should be divided into different clusters.

To illustrate the effect of pairwise constraints on datasets, we give an example to cluster a simple dataset as shown in Fig. 3. Since time series data is high dimension, we adopt principal component analysis (PCA) to reduce it into two dimensions for better presentation.

Fig. 3(a) is the real distribution of the simple dataset, there is one cluster on the upper part and two clusters on the lower part. However, in Fig. 3(b), the traditional clustering algorithm divides the data on the upper part into two clusters and the data on the lower part into one cluster. When a few pairwise constraints are provided, some data are correctly clustered but the improvement is not obvious, and the clustering result is shown in Fig. 3(c). From this figure, we notice that some data samples around pairwise constraints should also be reclustered. Based on this idea, we apply CPA to the clustering algorithm. And the clustering result is shown in Fig. 3(d), which is almost the same as the real distribution of the data in Fig. 3(a).

The ultimate goal of CPA is to improve the clustering result while saving human resources. Based on CPA, we can supply additional pairwise constraints. It could use limited cost to improve the clustering performance by make full use of the information provided by experts. The process of CPA is shown in Algorithm 2. Specifically, for every must-link constraint, it applies Lemmas 1–3 to disseminate the must-link

Algorithm 2 CPA

```
1  Require:
2    Input:
3    Dataset D
4    pairwise constraints P0
5  Ensure:
6      P1 = ϕ
7    for every pairwise constraint must-link(xi,xj)∈ P0
8        Obtain their reverse nearest neighbors RNN(xi) and
   RNN(xj)
9        Apply Lemma 1 to obtain pairwise constraints must-link(xi,
   RNN(xi)) and must-link(xj, RNN(xj)), and add them to P1
10       Apply Lemma 2 to obtain pairwise constraints must-link(xi,
   RNN(xj)) and must-link(xj, RNN(xi)), and add them to P1
11   for every pairwise constraint cannot-link(xi,xj)∈ P0
12       Obtain their reverse nearest neighbors RNN(xi) and
   RNN(xj), and add them to P1
13       Apply Lemma 1 to obtain pairwise constraints must-link(xi,
   RNN(xi)) and must-link(xj, RNN(xj)), and add them to P1
14       Apply Lemma 3 to obtain pairwise constraints cannot-
   link(xi, RNN(xj)) and cannot-link(xj, RNN(xi)), and add them
   to P1
15 Output: pairwise constraints P = P0 ∪ P1
```

pairwise constraints based on CPA (lines 7–10); For each cannot-link constraint, Lemmas 1, 2, and 4 are used to generate new must-link and cannot-link constraints (lines 11–14). Finally, it outputs all pairwise constraints including original and generated pairwise constraints (line 15).

By this way, based on a few pairwise constraints to be provided by experts, we could supply enough implicit information to further improve the clustering result.

Occasionally when there are contradictions among these generated pairwise constraints, they can be handled according to the following principles.

1) When there is a contradiction between the pair-wise constraints provided by human experts and those generated by CPA algorithm, the pair-wise constraints provided by experts have the priority, and they are adopted to improve the clustering performance. At the same time, those conflicting pairwise constraints provided by CPA algorithm are overlooked, because they are not as accurate and credible as human experts.
2) When there is a contradiction between two pair-wise constraints generated by CPA, the first come first serve (FCFS) strategy is adopted. With FCFS, the first traversed pair-wise constraints will participate in the clustering process and the late generated conflicting constraints will be ignored.

C. Semi-Supervised Clustering Framework

Based on fDTW and CPA techniques, we first tackle contract time series clustering, and propose a semi-supervised clustering framework (STSC). The overview of the framework is shown in Algorithm 3.

This framework takes the original dataset D with pairwise constraints P_0 as the input. First, STSC applies a CPA to disseminate the pairwise constraints provided by experts (lines 6 and 7). Then, a clustering algorithm is executed to obtain the

Algorithm 3 Framework of Semi-Supervised Clustering Approach

1 Require:
2 Input:
3 The dataset D
4 The pairwise constraints P_0
5 Ensure:
6 Apply constraint propagation approach in D based on pairwise constraints P_0 to obtain new pairwise constraints P_1
7 $P = P_0 \cup P_1$
8 Apply a clustering algorithm to obtain the result with fast similarity measure and P
9 Output: the clustering results

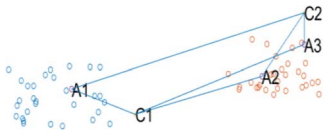


Fig. 4. Example of calculating fDTW in fssK-means.

clustering result based on fast similarity measure and extended pairwise constraints (lines 8 and 9).

The clustering is to minimize interdistances between any pair of examples in each cluster and maximize the intradistances between any two clusters while satisfying many pairwise constraints as much as possible. So, the global evaluation function of clustering is as follows:

$$Q = \underset{Q}{\operatorname{argmin}} \sum_{i=1}^{M} \sum_{j=1}^{N} q_{ij} * \mathrm{fDTW}\big(C_i, x_j\big) \tag{8}$$

where

$$q_{ij} = \begin{cases} 1, & \text{if } \mathrm{fDTW}\big(x_j, C_i\big) \le \mathrm{fDTW}\big(x_j, C_b\big),\ 1 \le \forall b \le M \\ -1, & \text{otherwise} \end{cases}$$

M and N are the number of clusters and the number of examples of the dataset D, respectively.

IV. Two Fast Semi-Supervised Clustering Approaches

In this section, we apply the proposed framework to K-means and DBSCAN [49], and two clustering algorithms including fssK-means and fssDBSCAN are designed. Moreover, this framework is not limited to K-means and DBSCAN algorithms, it can be applied to any distance-based clustering algorithm with little or no modification.

A. fssK-Means Clustering

In a nutshell, K-means is a clustering algorithm that attempts to find K nonoverlapping clusters. Based on the proposed semi-supervised clustering framework, we extend K-means to fssK-means to make it faster and more accurate. A quick example is used to explain how to update fDTW matrix in fssK-means. Fig. 4 represents a dataset with two clusters. A1, A2, and A3 are three data samples, C1 and C2 are two cluster centers. Since the difference between aDTW(A1, C1) and aDTW(A1, C2) is large, we can easily tell that A1 belongs to class C1. In this case, we need not update the distances aDTW(A1, C1) and aDTW(A1, C2). Similarly, A3 could be divided into C2 with aDTW(A3, C1) and aDTW(A3, C2) without updating its ground-truth DTW. However, aDTW(A2, C1) and aDTW(A2, C2) are almost the same. In this case, to judge which center is closer to A2, we should further calculate their ground-truth distance DTW(A2, C1) and aDTW(A2, C2) to update their aDTW distances.

Algorithm 4 fssK-Means Clustering Approach

1 Require
2 Input
3 Dataset D
4 The parameter threshold δ
5 Pairwise constraints P_0
6 Ensure
7 Apply constraint propagation approach based on P_0 to obtain new pairwise constraints P_1
8 $P = P_0 \cup P_1$
9 S $\leftarrow$ Build a few core groups with pairwise constraints P
10 D_0 $\leftarrow$ Construct an updated dataset D_0 based on solution 1 and 2
11 Do
12 Generate K cluster centers
13 Assign each example xi in the dataset D_0 to one of centers based on fDTW
{
14 if exists a cannot-link(xi,xj) in P
then assign xj to the another center different from the center of xi based on solution 2
}
15 Until (stopping condition is satisfied)
16 Output: the clustering result

Moreover, we adopt CPA to disseminate the pairwise constraints provided by experts. Algorithm 4 presents a procedure of fssK-means. In this algorithm, it first generates an extended pairwise constraints based on CPA(lines 7–10), then according to K cluster centers the data is divided into k groups based on fDTW, solutions 1 and 2 (lines 11–14). The algorithm iterates until it converges or satisfies the stopping condition (line 15). Finally, it outputs the clustering result (line 16).

B. fssDBSCAN Clustering

Similarly, DBSCAN is extended to fssDBSCAN based on our proposed semi-supervised clustering framework. The example in Fig. 5 describes the fDTW in fssDBSCAN. Since fDTW(A, B1) and fDTW(A, B2) are quite different from the radius, we can easily tell that B1 is in the circle, and B2 is outside. On the contrary, fDTW(A, B3) and fDTW(A, B4) are close to the radius. To identify whether B3 or B4 is in the area of A's neighbor space, we should update their ground-truth DTW distances instead of aDTW distances.

Based on the CPA, we augment pairwise constrains by making full use of side-information provided by experts. The pseudocode of the fssDBSCAN is described in Algorithm 5.

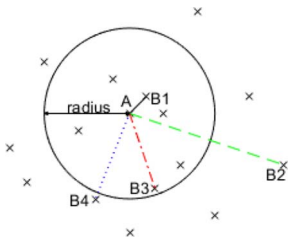


Fig. 5. Example of calculating fDTW in fssDBSCAN.

Algorithm 5 fss DBSCAN Clustering Approach

Require
  Input
  Dataset D
  The parameter threshold δ
  Pairwise constraints P_0
Ensure
  Apply constraint propagation approach based on P_0 to obtain new pairwise constraints P_1
  $P = P_0 \cup P_1$
  S $\leftarrow$ Merging into super-examples with must-link pairwise constraints of P
  D_0 $\leftarrow$ Constructing an updated dataset D_0 based on sup-examples S
  R $\leftarrow$ DBSCAN(D_0, ε, MinPts) based on fDTW
  R1 $\leftarrow$ Adjusting clustering results R based on cannot-link pairwise constraints in P
Output: the clustering results R1

In this algorithm, it first generates an extended pairwise constraints based on CPA. After that, it applies DBSCAN to the dataset based on fDTW, and solution 1. Finally, the clustering result is adjusted based on the cannot-link pairwise constraints.

V. Experiments

To illustrate the performance of our proposed approaches, in experiments all methods are evaluated using 11 real-world datasets shown in Table II. These datasets come from the open time series database UCR [51]. They are Adiac, ECG5000, FaceALL, wafer, MALLAT, MedicalImages (MI), Earthquakes, PhalangesOutlinesCorrect (POC), Strawberry, wafer, yoga, and StarLightCurves (SLC). The length of the time series is between 80 and 1024, the size of the datasets is between 55 and 9236, and the number of classes of the datasets is between 2 and 37.

In our experiments, the parameter k of K-Means algorithm is set to the number of class labels of each dataset. For the DBSCAN algorithm, the parameters *minPts* is set to 5 referring to the parameter setting for high-dimensional data in [48], and the optimal parameter ε is determined based on k-dist function proposed in [52]. In our semi-supervised learning experiments, the number of data examples involved in the pairwise constraints is set to 10% of the size of the datasets if there is no special description.

In the validation step, we employ three widely used criteria to measure performance of clustering results. We evaluate each method by calculating these criteria's mean value after ten runs. These criteria are briefly introduced below.

TABLE II
Summary of 11 Datasets for Our Experiments

Name	Length	Size	Classes
Adiac	176	390	37
ECG5000	140	500	5
Earthquakes	512	139	2
FaceALL	131	560	14
MALLAT	1024	55	8
MI	99	381	10
POC	80	1800	2
Strawberry	235	370	2
wafer	152	1000	2
yoga	426	300	2
SLC	1024	9236	3

A. Rand Index

RI is commonly used to evaluate clustering results. Let us say TP is the number of time series pairs that is divided into the same cluster correctly, TN is the number of time series pairs that is not and should not be divided into the same cluster, FP is the number of time series pairs that is divided into the same cluster but should not be divided into the same cluster, and FN is the number of time series pairs that is not assigned to the same cluster and should not be divided into the same cluster. RI is defined as follows:

$$\mathrm{RI} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}. \tag{9}$$

RI lies between 0 and 1, and higher RI value represents better clustering performance.

B. Purity

The purity of the cluster is defined as a ratio between total rough entropy and maximum total rough entropy of the cluster where total rough entropy is sum of rough entropies on all the attributes of the cluster. The purity of clustering result can be calculated like this

$$\mathrm{Purity} = \sum_{i=1}^{K} \frac{m_i}{m} \tag{10}$$

where K is the number of the classes, m_i is the largest group of data in the current class that really belong to the same class, and m in the denominator is the total amount of data in the current class. Purity lies between 0 and 1, and higher purity value represents better clustering performance.

C. Normalized Mutual Information

The normalized mutual information (NMI) is one of the most popular information theoretic measures for community detection methods. Given the ground-truth result X with n clusters $X = \{X_1, X_2, \ldots, X_n\}$, and the result Y obtained by the clustering algorithm with m clusters $Y = \{Y_1, Y_2, \ldots, Y_n\}$, the NMI value is calculated as follows:

$$\mathrm{NMI}(X, Y) = 2R = 2\frac{I(X; Y)}{H(X) + H(Y)} \tag{11}$$

where

$$I(X, Y) = \sum_{x,y} p(x, y) \log \frac{p(x, y)}{p(x)p(y)} \tag{12}$$

$$H(X) = \sum_{i-1}^{n} p(x_i)I(x_i) = \sum_{i-1}^{n} p(x_i)\log_b \frac{1}{p(x_i)}$$
$$= -\sum_{i-1}^{n} p(x_i)\log_b p(x_i). \quad (13)$$

The process of normalization is to use entropy as the denominator to adjust the value of mutual information between 0 and 1, and higher NMI also represents better clustering performance.

D. Effect of Parameters in fDTW

In Section III, the process of computing fDTW includes two main steps, in which two important parameters are involved. In the first step, parameter β is adopted to obtain the estimated distance measure according to the upper and LBs. In the second step, parameter d is used to determine whether the approximate similarity distance should be updated instead of ground-truth DTW or not.

First, we analyze the threshold parameter δ used in (6). Generally, if δ is too large, most of the estimated distance measure will not be updated, including the wrong estimates, which will lead to the decrease of the accuracy. On the other hand, if δ is too small, there will be too many updating operations. It leads to too much time of computation, thus fDTW strategy is meaningless.

In our experiment, δ is defined as a sigmoid function as follows:

$$\delta = \frac{1}{1 + e^{-\gamma}} \quad (14)$$

where γ represents the standard deviation of all aDTWs between all pairs of examples.

Next, in order to specify the effect of the parameter β on the estimating result, in-depth analysis of β is conducted with experiments on 11 datasets. We use different β values to complete the estimating operation, and compare the aDTW results with the ground-truth DTW value to calculate the error rate. The estimation error rate for a given dataset D is defined as follows:

$$\mathrm{err}_\beta(D) = \frac{1}{2n} * \sum_{x_i, x_j \in D} \frac{\mathrm{abs}\big(\mathrm{DTW}(x_i, x_j) - \mathrm{aDTW}(x_i, x_j)\big)}{\mathrm{DTW}(x_i, x_j)}. \quad (15)$$

According to the definition of the estimation error rate, an appropriate β value for a given dataset D is obtained by minimizing the estimation error rate $\mathrm{err}_\beta(D)$. β can be described as follows:

$$\beta = \mathrm{argmin}_\beta\big(\mathrm{err}_\beta(D)\big). \quad (16)$$

In the experiments, we increase β from 0.05 to 0.95 with an increment step of 0.05. The estimating error is calculated on 11 datasets for every given β. The results are shown in Fig. 6. It shows that the estimate error of most datasets can reach the bottom when β is in the range [0.2, 0.4]. For instance, the lowest estimate error values of 0.285, 0.154, 0.203, and 0.150 on Adiac, Earthquake, ECG5000, and FaceAll datasets are obtained when β is equal to 0.2, 0.3, 0.3, and 0.35.

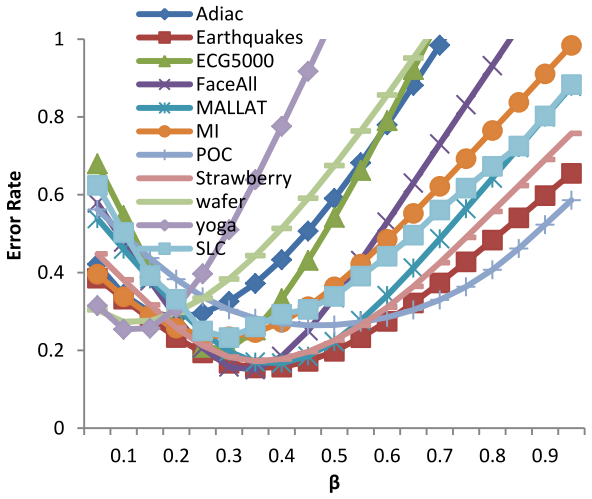


Fig. 6. Relationship between β and the estimating error on 11 datasets.

TABLE III
TIME CONSUMPTION (IN SECONDS) OF FSSK-MEANS, FSSK-MEANS(DTW), AND K-MEANS(ED)

	fssK-means	fssK-means(DTW)	K-means(ED)
Adiac	92.3	233.7	122.1
Earthquakes	27.6	76.7	19.3
ECG5000	46.5	197.6	82.1
FaceALL	70.0	96.7	57.2
MALLAT	44.1	62.3	30.8
MI	51.7	228.2	92.7
POC	60.1	171.9	37.1
Strawberry	84.3	207.3	57.5
wafer	42.3	89.7	30.8
yoga	90.3	165.0	81.9
SLC	1105.6	16387.9	856.4

A possible reason is that when β is between 0.2 and 0.4, the relationship between DTW and its two bounds (UB and LBs) can be adequately captured. Thus, β is set to 0.3 for the following experiments.

E. Performance of fssK-Means and fssDBSCAN

To investigate the efficiency and effectiveness of fDTW, we use the ground-truth DTW distance to replace the fDTW in fssK-means and fssDBSCAN and denote them fssK-means (DTW) and fssDBSCAN (DTW). Then, we compare the time consumption and performance of fssK-means and fssK-means (DTW), fssDBSCAN, and fssDBSCAN (DTW) on 11 datasets. In addition, they are compared with baselines methods K-means and DBSCAN using ED as the similarity measurement. For simplification, they are denoted as K-means (ED) and DBSCAN (ED), respectively.

First, we analyze the computation time of these algorithms on 11 datasets, and the results are shown in Tables III and IV. From Table III, we notice that the time consumption of fssK-means is much less than fssK-means (DTW). For instance, fssK-means (DTW) spent 233.7 s, 197.6 s, 228.2 s, and 171.9 s on Adiac, ECG5000, MI, and POC datasets, respectively. They cost more 141.4 s, 151.1 s, 176.5 s, and 111.8 s than the time consumption of fssK-means on the same dataset, respectively.

TABLE IV
TIME CONSUMPTION (IN SECONDS) OF FSSDBSCAN, FSSDBSCAN(DTW), AND DBSCAN(ED)

	fssDBSCAN	fssDBSCAN(DTW)	DBSCAN(ED)
Adiac	97.4	165.9	85.1
Earthquakes	42.2	88.7	41.7
ECG5000	31.6	83.5	36.1
FaceALL	66.1	182.4	66.0
MALLAT	113.3	237.3	117.2
MI	31.7	77.3	31.7
POC	134.5	316.7	65.9
Strawberry	90.8	167.8	100.4
wafer	118.2	304.5	53.7
yoga	67.4	244.0	93.1
SLC	1232.2	13622.6	913.3

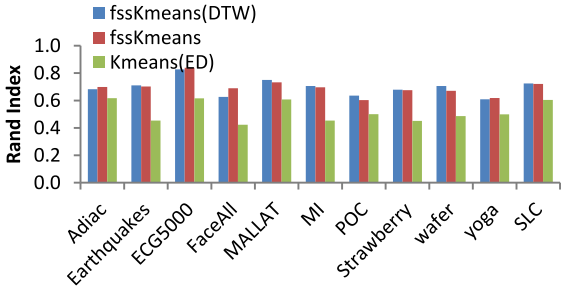


Fig. 7. RI of clustering results of *K*-means(ED), fssK-means(DTW), and fssK-means.

On the other hand, the computation time of fssK-means is almost the same as that of *K*-means (ED) on most of the datasets. For instance, the time cost for *K*-means (ED) on Adiac, ECG5000, MI, and POC datasets are 122.1 s, 82.1 s, 92.7 s, and 37.1 s, respectively. The time cost differences between fssK-means and *K*-means (ED) are just 29.8 s, 35.6 s, 41.0 s, and 23 s, respectively.

Similarly, Table IV shows that fssDBSCAN cost much less time than fssDBSCAN (DTW) and is almost the same as DBSCAN (ED). For example, fssDBSCAN cost less 68.5 s, 51.9 s, 45.6 s, and 182.2 s than fssDBSCAN (DTW) on Adiac, ECG5000, MI, and POC datasets, respectively. The difference between fssDBSCAN and DBSCAN (ED) are only 12.6 s, 4.5 s, 0.06 s, and 68.6 s on the same dataset, respectively. Therefore, It shows that the computation complexity of fDTW as same as that of ED in the clustering, and fDTW could greatly reduce the computation time compared with DTW.

Meanwhile, the quality of the clustering results using *K*-means(ED), fssK-means, and fssK-means(DTW), DBSCAN(ED), fssDBSCAN, and fssDBSCAN(DTW) on 11 datasets are shown in Figs. 7–12, respectively, including RI, purity, and NMI.

From Figs. 7 to 12, we see that the quality of fssK-means and fssDBSCAN is as well as fssK-means (DTW) and fssDBSCAN (DTW) on almost all datasets. For example, RI of fssK-means (DTW) and fssDBSCAN (DTW) on Adiac dataset is 0.682 and 0.633, respectively. They are just larger than those of fssK-means and fssDBSCAN by 0.017 and 0.004, respectively. The purity of fssK-means (DTW) and fssDBSCAN (DTW) on Adiac dataset is 0.789 and 0.582,

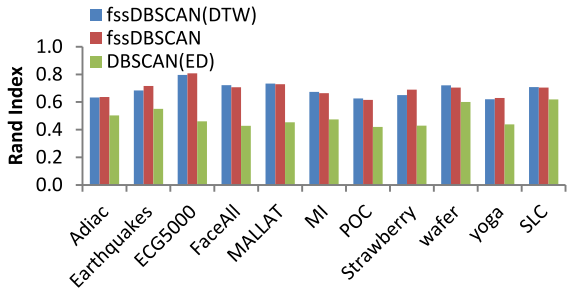


Fig. 8. RI of clustering results of DBSCAN(ED), fssDBSCAN (tDTW), and fssDBSCAN.

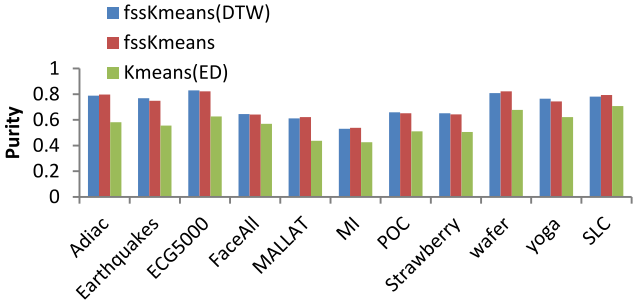


Fig. 9. Purity of clustering results of *K*-means(ED), fssK-means(DTW), and fssK-means.

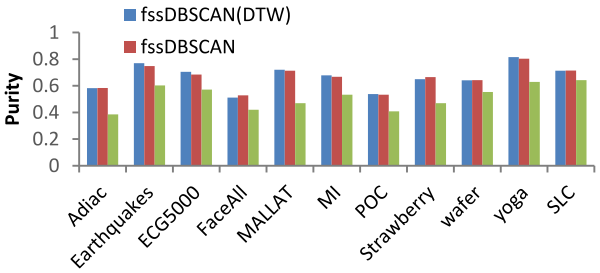


Fig. 10. Purity of clustering results of DBSCAN(ED), fssDBSCAN (DTW), and fssDBSCAN.

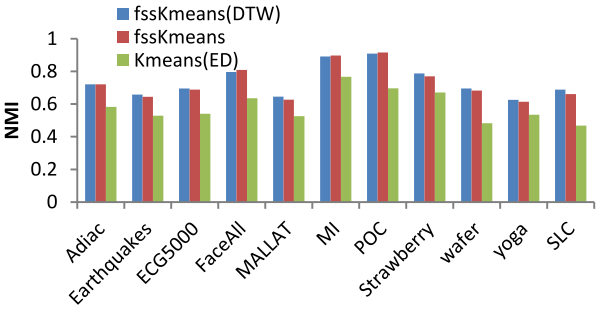


Fig. 11. NMI of clustering results of *K*-means(ED), fssK-means(DTW), and fssK-means.

respectively. Compared with those of fssK-means and fssDBSCAN, the differences are just 0.008 and 0.002, respectively. Similarly, the difference of NMI between fssK-means (DTW) and fssK-means, fssDBSCAN (DTW) and fssDBSCAN on MI dataset are just 0.006 and 0.027, respectively.

At the same time, the performances of fssK-means and fssDBSCAN outperform those of K-means (ED) and

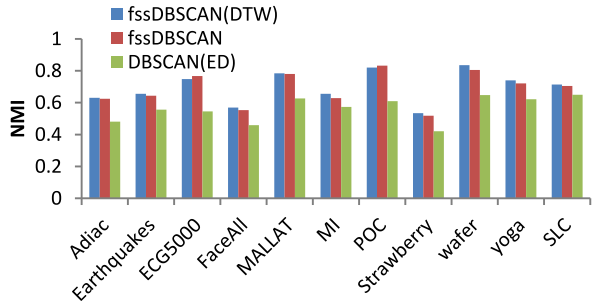


Fig. 12. NMI of clustering results of DBSCAN(ED), fssDBSCAN (DTW), and fssDBSCAN.

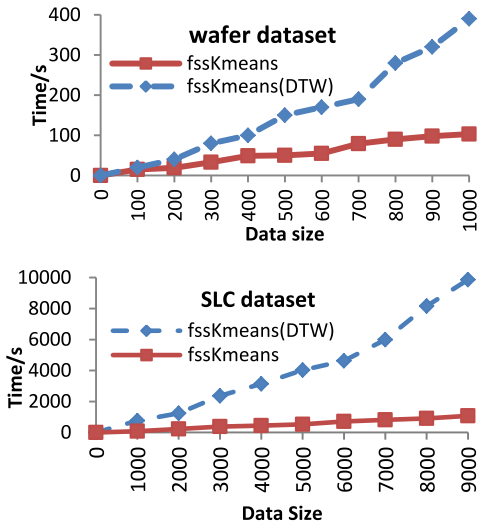


Fig. 13. Time consumption of two clustering algorithms on two datasets with different scales.

DBSCAN (ED) on all datasets. For instance, the RI of fssK-means and fssDBSCAN on Adiac dataset are 0.699 and 0.637, respectively. They are 0.081 and 0.133 higher than those of *K*-means (ED) and DBSCAN (ED), respectively. The purity of fssK-means and fssDBSCAN on Adiac dataset are 0.797 and 0.584, they are 0.215 and 0.199 higher than those of *K*-means (ED) and DBSCAN (ED), respectively. Similarly, the NMI of fssK-means (DTW) and fssDBSCAN (DTW) on Adiac dataset is 0.138 and 0.143 higher than those of *K*-means (ED) and DBSCAN (ED), respectively.

To sum up, the experimental results show that fDTW can largely accelerate the clustering process compared with DTW. At the same time, fDTW does not reduce the clustering quality. Hence, it is a remarkable fact that the fDTW can dramatically reduce the time cost without bringing down the accuracy of clustering.

To further investigate the time complexity of fDTW on large-scale time series data, fssK-means (DTW), and fssK-means are tested as the size of data increases. We conduct experiments on 11 datasets and the results are all similar. For brevity, we only present the results of two datasets (wafer and SLC) here. The results are shown in Fig. 13, which shows that as the data size increase, the time consumption of fssK-means (DTW) increases exponentially but the time consumption of fssK-means increases linearly. Specifically, when

TABLE V
COMPARISON BETWEEN ACA AND FSSK-MEANS

	ACA					fssK-means	
	NUDs					NUDs	RI
	0	40	80	200	500		
Adiac	0.575	0.622	0.650	0.708	0.714	85	0.705
Earthquakes	0.502	0.515	0.550	0.615	0.613	98	0.626
ECG5000	0.543	0.583	0.638	0.743	0.761	89	0.775
FaceAll	0.524	0.523	0.537	0.648	0.672	132	0.668
MALLAT	0.616	0.640	0.716	0.719	0.716	33	0.691
MI	0.452	0.505	0.583	0.644	0.645	151	0.647
POC	0.507	0.548	0.575	0.583	0.584	324	0.592
Strawberry	0.401	0.457	0.562	0.636	0.648	97	0.626
wafer	0.419	0.449	0.525	0.547	0.544	212	0.559
yoga	0.489	0.586	0.664	0.691	0.690	109	0.643
SLC	0.533	0.596	0.657	0.692	0.708	342	0.721

the size of wafer dataset goes up to 1000 gradually, the time consumption of fssK-means (DTW) increases from 0 to 390 s while fssK-means increases from 0 to 103 s. Similarly, when the size of SLC dataset increases to 9000, the growth rate of the time consumption of fssK-means (DTW) is much higher than that of fssK-means.

In addition, we compare fDTW with an approximation to DTW, which could reduce the computation time of clustering. Based on the approximation to DTW, Zhu *et al.* [46] introduced ACA. To be fair, we compare fssK-means with ACA, which also adopts *k*-means clustering algorithm. Since ACA is an anytime algorithm while our algorithm is a contract algorithm, here we record the number of pairs needed to update their ground-truth DTW in ACA in different stages. The experimental results are shown in Table V (NUD stands for the number of pairs needed to update their ground-truth DTW (lower NUD means lower time consumption).

We can see that fssK-means can reach almost the same RI as ACA with a much lower NUD on most of the datasets. For example, RI of ACA on Adiac, Earthquakes, ECG5000, and FaceAll datasets with 500 NUDs is 0.71, 0.61, 0.76, and 0.67, respectively. For fssK-means, RI of clustering results on these datasets is 0.70, 0.62, 0.77, and 0.66, respectively, which are almost the same as ACA. And the NUDs of fssK-means is 85, 98, 89, and 132, respectively. Compared with ACA, the NUDs of fssK-means is much lower, which means the computation time of fssK-means is much less than that of ACA when they obtain the same clustering quality. So, fDTW could further enhance the efficiency of clustering compared with an approximate similarity measure in ACA. The reason is that fssK-means captures the relationship between DTW and its upper and LBs fully while adopting an effectively updating strategy to guarantee the accuracy. Thus, it can achieve good performance with less cost of computation time. Moreover, when using ACA, it is hard for users to stop the program with a good clustering result at the right time. On the contrary, fssK-means could efficiently obtain a good clustering result without users' interruption in the process of clustering.

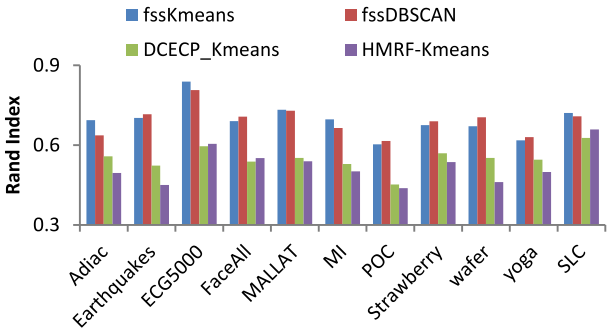


Fig. 14. Clustering results of fssKmeans, fssDBSCAN, DCECP_Kmeans, and HMRF_Kmeans on 11 datasets.

TABLE VI
TIME CONSUMPTION (IN SECONDS) OF FSSKMEANS, FSSDBSCAN, DCECP_KMEANS, AND HMRF_KMEANS ON 11 DATASETS

	fssKmeans	FssDBSCAN	DCECP_ Kmeans	HMRF_ Kmeans
Adiac	92.3	97.4	125.3	130.6
Earthquakes	27.6	42.2	89.0	71.0
ECG5000	46.5	31.6	77.8	83.2
FaceALL	70.0	66.1	110.4	111.6
MALLAT	44.1	113.3	136.4	150.4
MI	51.7	31.7	73.4	60.2
POC	60.1	134.5	118.4	109.8
Strawberry	84.3	90.8	130.1	135.6
wafer	42.3	118.2	138.0	131.5
yoga	90.3	67.4	100.6	122.2
SLC	1105.6	1232.2	2393.8	2147.3

In order to prove that our framework can achieve better clustering results over traditional methods on high-dimensional time series datasets, we compare fssK-means and fssDBSCAN with two state-of-the-art semi-supervised clustering methods HMRF_Kmeans [20] and DCECP [23]. For DCECP being a semi-supervised ensemble clustering method, to be fair, we apply its semi-supervised clustering framework on K-means clustering algorithm in our experiments and denote it as DCECP_Kmeans. To test the clustering performance of the algorithms when experts can only provide a small amount of pair-constraints, the proportion of data examples involved in the constraints are set to 5% of the size of the dataset for this experiment. The results are shown in Fig. 14 and Table VI.

From Fig. 14, we see that the performances of fssKmeans and fssDBSCAN algorithms obviously outperform those of DCECP_Kmeans and HMRF_Kmeans on high-dimensional time series datasets. For example, RI of fssK-means is 0.694, 0.702, and 0.823 on Adiac, Earthquake, and ECG5000 dataset, respectively. They are 0.136, 0.179, and 0.243 higher than those of DCECP_Kmeans, and 0.199, 0.252, and 0.234 higher than those of HMRF_Kmeans, respectively; RI of fssDBSCAN is 0.637, 0.716, and 0.807 on Adiac, Earthquake, and ECG5000 dataset, respectively. They are 0.079, 0.193, and 0.211 higher than those of DCECP_Kmeans, and 0.142, 0.266, and 0.202 higher than those of HMRF_Kmeans, respectively. When comparing fssK-means and fssDBSCAN, the results show that these two methods have their own merits and shortcomings on different datasets. For example, RI of fssK-means is 0.057 and 0.032 higher than those of fssDBSCAN on Adiac and ECG5000, respectively. However, RI of fssDBSCAN is 0.014 and 0.017 higher than those of fssK-means on Earthquakes and FaceAll, respectively. However, both of them outperform the traditional methods.

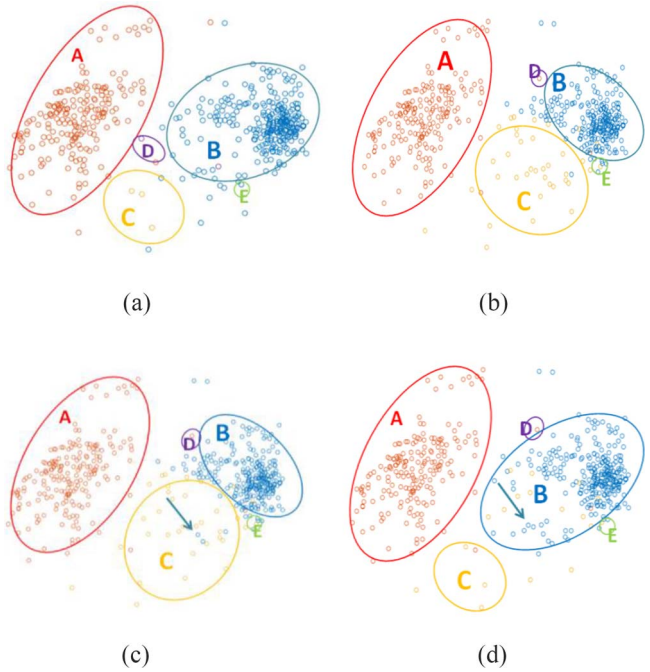


Fig. 15. Effect of pairwise constraints provided by experts and CPA on ECG5000 dataset. (a) Real distribution. (b) Clustering result with K-means. (c) Clustering result with K-means based on pairwise constraints provided by experts. (d) Clustering result with K-means based on pairwise constraints provided by experts and CPA.

Moreover, Table VI shows that the time consumption of our proposed fssKmeans and fssDBSCAN are much smaller than that of DCECP_Kmeans and HMRF_Kmeans. For example, the time consumption of fssKmeans, fssDBSCAN, DCECP_Kmeans, and HMRF_Kmeans on SLC dataset is 1105.6 s, 1232.2 s, 2393.8 s, and 2147.3 s, respectively. The time of proposed two methods is just a half time of DCECP_Kmeans and HMRF_Kmeans. Therefore, the fast clustering framework STSC proposed in this article can be applied to various clustering algorithms to improve the performance and speed of clustering.

F. Analysis of Constraint Propagation Approach

1) Visualization of the Effect of CPA: First, we visualize the effect of CPA for the clustering. Since dimensions of time series data is very high, we adopt principle component analysis (PCA) to transform each time series example into a two-dimensional (2-D) one and plot it in a 2-D plane. For the sake of brevity, only ECG5000 dataset and wafer dataset are visualized in this section, and CPA works on other datasets is in a similar way.

According to the real distribution shown in Fig. 15(a), ECG5000 has five clusters and most of them are divided into three clusters (cluster A, B, and C). Cluster D and E just contains a few examples, so we mainly focus on cluster A, B,

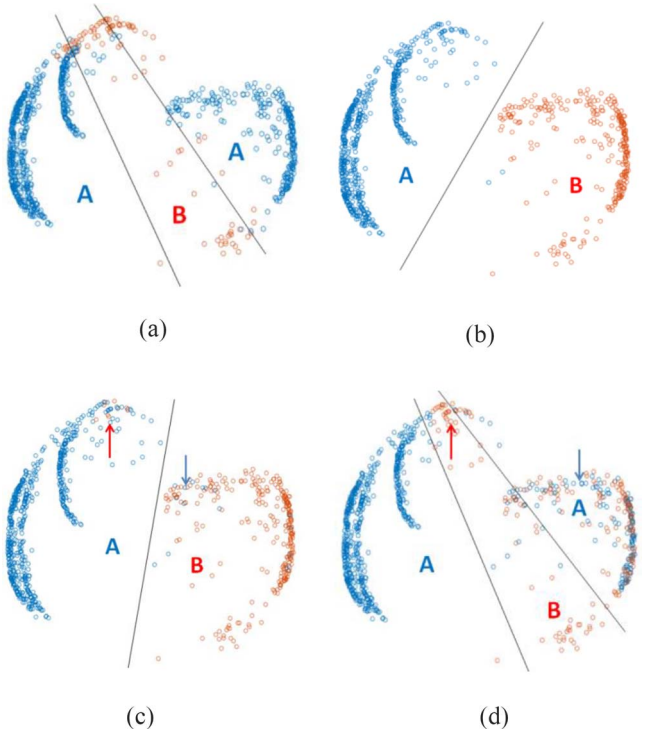


Fig. 16. Effect of pairwise constraints provided by experts and CPA on wafer dataset. (a) Real distribution. (b) Clustering result with *K*-means. (c) Clustering result with *K*-means based on pairwise constraints provided by experts. (d) Clustering result with *K*-means based on pairwise constraints provided by experts and CPA.

TABLE VII
PROPORTION OF NUMBER OF RNNS IN DATASETS

	0 RNN	1 RNN	2 RNNs	>=3 RNNs
Adiac	36.7%	36.4%	19.0%	7.9%
Earthquakes	33.1%	39.6%	18.7%	8.6%
ECG5000	37.8%	34.8%	19.0%	8.4%
FaceAll	37.0%	37.5%	16.8%	8.8%
MALLAT	32.7%	40.0%	23.6%	3.6%
MI	32.3%	42.0%	19.9%	5.8%
POC	42.2%	32.0%	15.6%	10.3%
Strawberry	32.4%	41.6%	20.3%	5.7%
wafer	30.8%	44.1%	19.6%	5.5%
yoga	34.9%	35.4%	24.7%	5.0%
SLC	31.2%	36.4%	25.1%	7.3%

TABLE VIII
TIME COST (IN SECONDS) OF CLUSTERING WITH DIFFERENT AMOUNT OF RNNS IN CPA

	KM	KM-PC	KM-CPA1	KM-CPA2	KM-CPA3
Adiac	139.5	115.2	97.9	92.3	90.4
Earthquakes	72.7	65.4	42.8	27.6	27.0
ECG5000	99.6	78.3	54.6	46.5	43.2
FaceALL	109.4	98.2	82.6	70.0	65.5
MALLAT	66.9	62.0	55.3	44.1	42.6
MI	103.7	87.9	59.8	51.7	47.4
POC	89.5	83.1	63.8	60.1	60.0
Strawberry	116.0	108.5	89.7	84.3	77.4
wafer	68.4	56.3	49.5	42.3	40.0
yoga	136.8	124.9	103.0	90.3	81.5
SLC	1539.3	1441.5	1226.8	1105.6	1058.1

and C. According to the clustering result obtained by *K*-means algorithm shown in Fig. 15(b), the main problem is that cluster B is too small and cluster C is too large. Fig. 15(c) shows the clustering result of ECG5000 after pairwise constraints provided by experts are added. Several points in class C are forced into class B with pairwise constraints. However, the effect is not obvious and the clustering results show little improvement. To further mine the "representative information" contained in pairwise constraints, we advance CPA to generate more pairwise constraints based on original constraints provided by experts. In Fig. 15(d), a chunk of data on class has been correctly clustered with CPA, and the clustering result is competitive. It indicates CPA is effectively to improve the clustering performance. Similarly, Fig. 16(a)–(d) are the visualization of the wafer data set. According to the real distribution of wafer on Fig. 16(a), the data on both side are cluster A and the data on middle part is cluster B. Fig. 16(b) is the clustering result obtained by *K*-means algorithm, the data on the left side is cluster A and the data on the right side is cluster B. Similar to ECG5000 dataset, the semi-supervised clustering result with pairwise constraints provided by experts shows little improvement [Fig. 16(c)]. Adopting CPA, the clustering result is much closer to the real distribution [Fig. 16(d)]. To sum up, just a few pairwise constraints provided by experts cannot greatly enhance the clustering result. On the other hand, CPA could effectively disseminate the pairwise constraints, which can largely improve the clustering result.

2) Analysis of RNN in CPA: In CPA, RNN technique is adopted to generate more pairwise constraints. To investigate the effect of CPA with different size of RNNs, in this section, we do experiments on 11 datasets. Moreover, we also execute experiments to compare traditional clustering algorithms without pairwise constraints, with just pairwise constraints provided by experts, and with our proposed clustering algorithms with CPA. The experimental results of fssK-means and fssDBSCAN are similar. For the limited space, here we just show the clustering results with fssK-means in Tables VII and VIII, and Fig. 17.

As we known, not all examples have equal number of RNNs in a dataset. To determine the size of RNNs in CPA, we first analyze the number of RNNS of examples in all datasets. The proportion of examples with different amounts of RNN in all datasets is shown in Table VII. It shows that most data examples have 0 RNN or 1 RNN. For example, 73.1% of the data in Adiac, 72.7% of the data in Earthquakes, 72.6% of the data in ECG5000, and 74.5% of the data in FaceAll have 0 or 1 RNN. Few data examples have 2 or more RNNs, for example, 26.9% in Adiac dataset, 27.3% in Earthquakes dataset, and 25.6% in FaceAll dataset.

At the same time, we compare the clustering results with different amount of RNNs in CPA. For brevity, clustering algorithms with different amount of RNNs in CPA is s abbreviated as following.

1) *KM-PC: K*-means with fDTW and pairwise constraints provided by experts.
2) *KM-CPA1: K*-means with fDTW and CPA, in which size of RNN is set to 1.

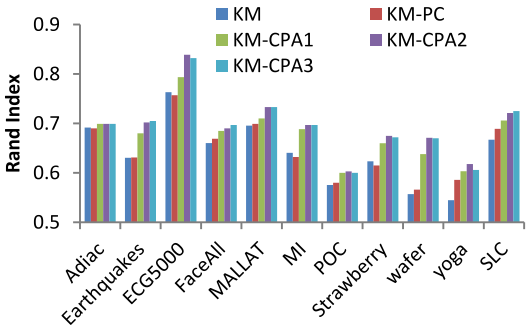


Fig. 17. Clustering result with different amount of RNNs in CPA.

TABLE IX
RI OF CLUSTERING RESULTS WITH FSSK-MEANS

	Proportion of pairwise constraints provided by experts					
	2%	4%	6%	8%	10%	12%
Adiac	0.690	0.693	0.694	0.694	0.699	0.694
Earthquakes	0.671	0.697	0.709	0.709	0.702	0.712
ECG5000	0.796	0.812	0.823	0.831	0.839	0.840
FaceAll	0.677	0.681	0.686	0.689	0.690	0.690
MALLAT	0.725	0.729	0.730	0.730	0.733	0.733
MI	0.667	0.677	0.692	0.696	0.697	0.697
POC	0.589	0.592	0.594	0.602	0.603	0.603
Strawberry	0.640	0.655	0.664	0.675	0.675	0.676
wafer	0.601	0.631	0.656	0.664	0.671	0.671
yoga	0.559	0.572	0.578	0.580	0.618	0.618
SLC	0.661	0.674	0.687	0.695	0.721	0.721

3) *KM-CPA2: K*-means with fDTW and CPA, in which size of RNN is set to 2.
4) *KM-CPA3: K*-means with fDTW and CPA, in which size of RNN is equal to of more than 3.

In addition, we also do *K*-means with fDTW and no pairwise constraints, which as noted as KM for short.

The comparative clustering results of these algorithms are shown in Fig. 17. According to Fig. 17, three things can be observed.

1) Only using pairwise constraints provided by experts has little positive effect on improving the accuracy of the clustering results, sometimes even has the opposite effect.
2) With CPA, clustering performance becomes better on most of the datasets. The reason is that CPA can capture the useful information contained in the side-information provided by experts and the structural characteristics of the clusters. Thus, it can improve the clustering results effectively.
3) There is little difference of the clustering performance with different amounts of RNNs in CPA, and it is competitive when the size of RNN is equal to 2. To sum up, the result above shows that adding only few of pairwise constraints provided by experts has not a substantial improvement in the results and our proposal CPA can greatly improve the clustering performance.

Moreover, the time consumption of clustering with different amounts of RNNs in CPA are compared in 11 datasets, and the results are shown in Table VIII. We observe that adopting pairwise constraints can slightly reduce the time consumption

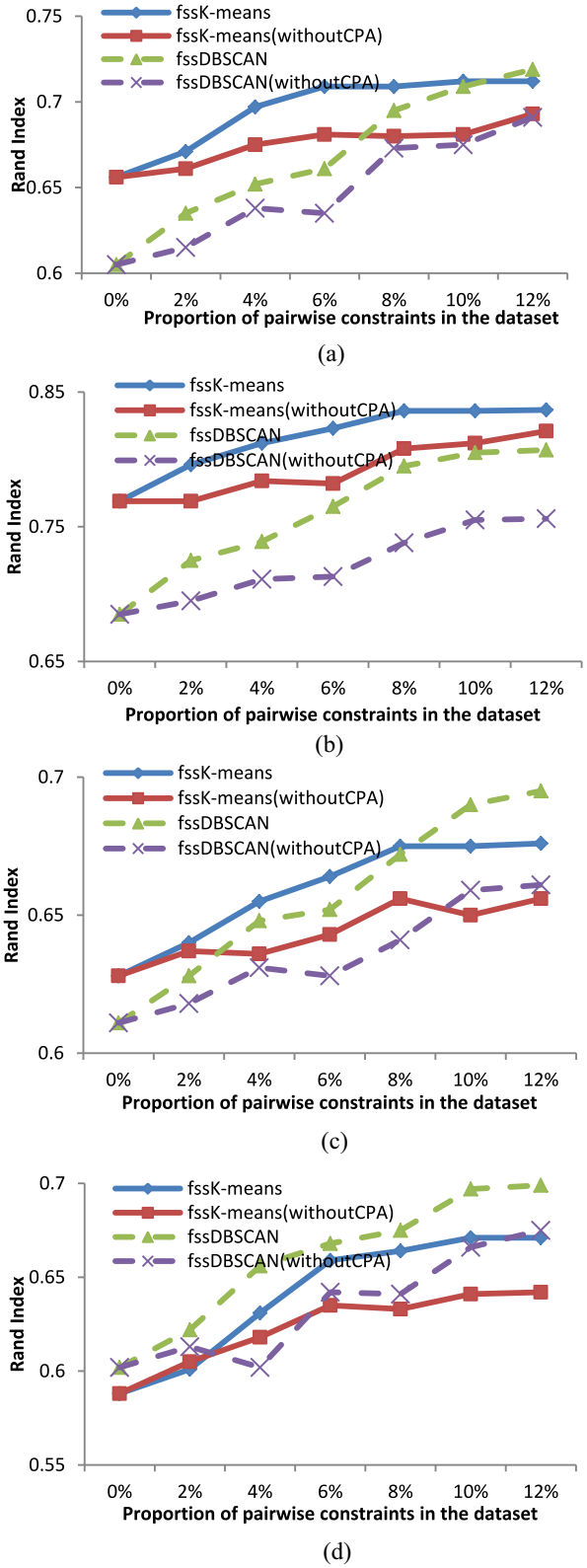


Fig. 18. Clustering results of fssK-means, fssK-means (withoutCPA), fssDBSCAN, and fssDBSCAN (withoutCPA) on (a) Earthquakes, (b) ECG5000, (c) Strawberry, and (d) wafer datasets with different proportion of pair-wise constraints.

of the clustering process on all of the datasets. For example, the time consumption of KM, KM-PC, KM-CPA1, KM-CPA2, and KM-CPA3 on Adiac dataset is 139.5 s, 115.2 s, 97.9 s,

TABLE X
RI OF CLUSTERING RESULTS WITH WSNNCUT

	Proportion of pairwise constraints provided by experts					
	2%	4%	6%	8%	10%	12%
Adiac	0.682	0.684	0.690	0.690	0.690	0.690
Earthquakes	0.655	0.669	0.706	0.703	0.708	0.708
ECG5000	0.778	0.796	0.821	0.832	0.840	0.840
FaceAll	0.664	0.668	0.680	0.687	0.687	0.687
MALLAT	0.699	0.707	0.718	0.727	0.730	0.730
MI	0.652	0.664	0.684	0.696	0.697	0.697
POC	0.577	0.584	0.597	0.602	0.606	0.606
Strawberry	0.627	0.645	0.655	0.661	0.667	0.667
wafer	0.57	0.60	0.64	0.67	0.67	0.67
yoga	0.653	0.662	0.676	0.681	0.681	0.681
SLC	0.596	0.618	0.634	0.667	0.688	0.710

TABLE XI
RI OF CLUSTERING RESULTS WITH DCECP

	Proportion of pairwise constraints provided by experts					
	2%	4%	6%	8%	10%	12%
Adiac	0.679	0.682	0.69	0.692	0.692	0.692
Earthquakes	0.648	0.663	0.707	0.707	0.708	0.708
ECG5000	0.77	0.793	0.813	0.83	0.837	0.84
FaceAll	0.658	0.665	0.677	0.684	0.688	0.688
MALLAT	0.695	0.702	0.719	0.73	0.73	0.733
MI	0.65	0.661	0.686	0.697	0.697	0.697
POC	0.579	0.585	0.598	0.601	0.606	0.606
Strawberry	0.623	0.643	0.655	0.66	0.665	0.667
wafer	0.56	0.59	0.61	0.65	0.675	0.677
yoga	0.649	0.66	0.676	0.678	0.683	0.683
SLC	0.611	0.627	0.653	0.689	0.702	0.714

92.3 s, and 90.4 s, respectively. In a word, CPA can not only reduce the time complexity of the algorithm, but also improve the clustering results.

3) Analysis of the Effect of CPA: Here, we analyze the effect of CPA in detail. First, fssK-means and fssDBSCAN are compared with fssK-means and fssDBSCAN without CPA. The latter two algorithms are denoted as fssK-means (withoutCPA) and fssDBSCAN (withoutCPA). We conduct experiments on all datasets with different proportion of pairwise constraints to analyze these algorithms, and the results are all similar. For brevity, we only present the clustering results of four datasets (Earthquakes, ECG5000, Strawberry, and wafer) in Fig. 18.

From Fig. 18, we see that fssK-means and fssDBSCAN outperform fssK-means (withoutCPA) and fssDBSCAN (withoutCPA). For example, when the proportion of pairwise constraints is 2%, 4%, and 6% on Earthquake dataset, RI of fssK-means are 0.671, 0.697, and 0.709, respectively. They are 0.010, 0.022, and 0.023 higher than those of fssK-means (withoutCPA), respectively; when the proportion of pairwise constraints is 2%, 4%, and 6% on Earthquake dataset, RI of fssDBSCAN are 0.635, 0.652, and 0.661, respectively. They are 0.020, 0.014, and 0.026 higher than those of fssDBSCAN (withoutCPA), respectively. Moreover, we notice that although pair-wise constraints can improve the clustering results without using CPA, the effect is not obvious. Sometimes there is even a reverse effect that causes RI value to decrease, and the algorithms using CPA do not have such problem. For instance, when the proportion of pairwise constraints is 2%, 4%, and 6% on ECG5000 dataset, RI value of fssK-means increases steadily, it is 0.796, 0.821, and 0.823, respectively. However, RI value of fssK-means (withoutCPA) has a fluctuation; it first increases by 0.015 and then decreases by 0.002. The clustering results on other datasets are similar. In a word, fssK-means and fssDBSCAN perform better than fssK-means (withoutCPA) and fssDBSCAN (withoutCPA) and they are more stable. The reason is that CPA strategy could disseminate the pairwise constraints provided by experts, which effectively improves the clustering performance.

Next, CPA is compared to two state-of-the-art semi-supervised methods in wSNNCut framework [36] and DCECP [23], respectively. In wSNNCut,

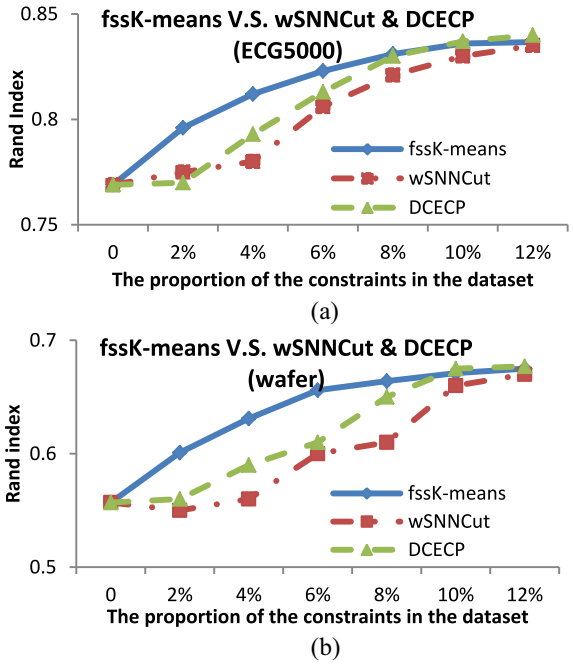


Fig. 19. RI of clustering results with fssK-means, wSNNCut, and DCECP on (a) ECG5000 and (b) wafer datasets with different proportion of pair-wise constraints.

it took a variety of similarity measures into account and users can add some constraints to the clustering algorithm based on the result of similarity measures. And in DCECP, a constraint weighting process is adopted to select the most useful pair-wise constraints provided by users. To be fair, we compare fssK-means with wSNNCut and DCECP, which adapt k-means clustering algorithm, and the RI of clustering results are shown in Tables IX–XI.

From Tables IX to XI, we see that when the amount of pairwise constraints provided by experts is small, fssK-means outperforms wSNNCut and DECEP on most of the datasets. For example, when the proportion of pairwise constraints is 2%, 4%, and 6% on MI dataset, RI of the clustering results with fssK-means are 0.667, 0.677, and 0.692, respectively. They are better than those obtained by wSNNCut and

DCECP. When more and more pairwise constraints are provided by the experts, RIs of fssK-means, wSNNCut, and DCECP tend to be the same. For instance, when the proportion is 12%, fssK-means, wSNNCut, and DCECP obtained the same RI value (0.697).

In order to better present this result, we select two datasets ECG5000 and wafer to draw the curve plot, as shown in Fig. 19. We see that fssK-means outperforms wSNNCut and DCECP when the proportion of the pairwise constraints provided by experts is low. When more and more manually side-information is provided, the advantage of fssK-means gradually weakens. The reason is that CPA could supply lots of pairwise constraints to improve the clustering results when there is only a small amount of pairwise constraints provided by experts. As more and more pairwise constraints provided by experts are added, the clustering becomes easier and easier, and the difference between two methods is very small. At this time, the effect of CPA becomes less and less obvious, but many pair-wise constraints need much cost. In general, we try to improve the clustering result with little cost. So, the results confirm that CPA is more effective than wSNNCut and DCECP with a few of pairwise constraints provided by experts.

VI. Conclusion and Future Work

In this article, based on pairwise constraints provided by experts, we first tackle contract time series clustering, and the task is to produce the most accurate clustering results under a contracted time. We propose an STSC framework, which integrates a fast similarity measure and a CPA. The proposed method has two advantages compared with conventional semi-supervised clustering methods: 1) the time consumption of our proposed algorithm is relatively small, and it does not increase rapidly as the data size goes up and 2) based on CPA, the accuracy of the clustering result can be greatly improved with just a few pairwise constraints provided by experts. Last, our framework is applied to *K*-means clustering algorithm and DBSCAN clustering algorithm. The experimental results clearly show that our proposed method can achieve efficient and effective clustering results on time series data.

So far, constraints provided by experts mainly including must-link and cannot-link pairwise constraints, which are instance-level constraints. However, in general experts often have global properties on some domain data, which is useful for clustering. In future, we will research semi-supervised clustering by incorporating expert guidance in the form of usefulness properties instead of pairwise constraints provided by experts.

References

[1] A. Gharehbaghi and M. Lindén, "A deep machine learning method for classifying cyclic time series of biological signals using time-growing neural network," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 9, pp. 4102–4115, Sep. 2018.

[2] J. D. Ouellette *et al.*, "A time-series approach to estimating soil moisture from vegetated surfaces using L-band radar backscatter," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 6, pp. 3186–3193, Jun. 2017.

[3] J. K. Pant and S. Krishnan, "Compressive sensing of foot-gait signals by enhancing group block-sparse structure on the first-order difference," in *Proc. 38th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC)*, Orlando, FL, USA, Oct. 2016, pp. 3700–3703.

[4] J.-S. Chou and T.-K. Nguyen, "Forward forecast of stock price using sliding-window metaheuristic-optimized machine learning regression," *IEEE Trans. Ind. Informat.*, vol. 14, no. 7, pp. 3132–3142, Jul. 2018.

[5] M. Han, S. Zhang, M. Xu, T. Qiu, and N. Wang, "Multivariate chaotic time series online prediction based on improved kernel recursive least squares algorithm," *IEEE Trans. Cybern.*, vol. 49, no. 4, pp. 1160–1172, Apr. 2019.

[6] K. George and P. Mutalik, "A multiple model approach to time-series prediction using an online sequential learning algorithm," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 49, no. 5, pp. 976–990, May 2019.

[7] H. He and Y. Tan, "Pattern clustering of hysteresis time series with multivalued mapping using tensor decomposition," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 6, pp. 993–1004, Jun. 2018.

[8] A. Sharabiani, H. Darabi, A. Rezaei, S. Harford, H. Johnson, and F. Karim, "Efficient classification of long time series by 3-D dynamic time warping, " *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 47, no. 10, pp. 2688–2703, Oct. 2017.

[9] M. Han, R. Zhang, T. Qiu, M. Xu, and W. Ren, "Multivariate chaotic time series prediction based on improved grey relational analysis," *IEEE Trans. Syst., Man, Cybern., Syst.*, to be published. doi: 10.1109/TSMC.2017.2758579.

[10] G. He, Y. Duan, R. Peng, X. Jing, T. Qian, and L. Wang, "Early classification on multivariate time series," *Neurocomputing*, vol. 149, pp. 777–787, Feb. 2015.

[11] G. He, L. Chen, C. Zeng, Q. Zheng, and G. Zhou, "Probabilistic skyline queries on uncertain time series," *Neurocomputing*, vol. 191, pp. 224–237, Mar. 2016.

[12] Q. Zhang, J. Wu, P. Zhang, G. Long, and C. Zhang, "Salient subsequence learning for time series clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, to be published. doi: 10.1109/TPAMI.2018.2847699.

[13] J. Zakaria, A. Mueen, and E. Keogh, "Clustering time series using unsupervised-shapelets," in *Proc. IEEE Int. Conf. Data Min.*, Brussels, Belgium, Dec. 2012, pp. 785–794.

[14] R. Randel, D. Aloise, N. Mladenović, and P. Hansen, "On the *k*-medoids model for semi-supervised clustering," in *Proc. 6th Int. Conf. Variable Neighborhood Search*, Sithonia, Greece, Oct. 2018, pp. 13–27.

[15] N. Asadi, A. Mirzaei, and E. Haghshenas, "Creating discriminative models for time series classification and clustering by HMM ensembles," *IEEE Trans. Cybern.*, vol. 46, no. 12, pp. 2899–2910, Dec. 2016.

[16] J. Paparrizos and L. Gravano, "Fast and accurate time-series clustering," *ACM Trans. Database Syst.*, vol. 42, no. 2, pp. 1–49, 2017.

[17] C. M. Salgado, M. C. Ferreira, and S. M. Vieira, "Mixed fuzzy clustering for misaligned time series," *IEEE Trans. Fuzzy Syst.*, vol. 25, no. 6, pp. 1777–1794, Dec. 2017.

[18] A. Khaleghi, D. Ryabko, J. Mary, and P. Preux, "Consistent algorithms for clustering time series," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 94–125, Jan. 2016.

[19] Z. Yu *et al.*, "Incremental semi-supervised clustering ensemble for high dimensional data clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 2, pp. 701–714, Mar. 2016.

[20] Z. Yu *et al.*, "Semi-supervised ensemble clustering based on selected constraint projection," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 12, pp. 2394–2407, Dec. 2018.

[21] K. Kamnitsas *et al.*, "Semi-supervised learning via compact latent space clustering," in *Proc. 35th Int. Conf. Mach. Learn. (ICML)*, Stockholm, Sweden, Jun. 2018, pp. 2459–2468.

[22] F. Nie, G. Cai, and X. Li, "Multi-view clustering and semi-supervised classification with adaptive neighbours," in *Proc. 31st AAAI Conf. Artif. Intell.*, San Francisco, CA, USA, Feb. 2017, pp. 2408–2414.

[23] S. Ding, H. Jia, M. Du, and Y. Xue, "A semi-supervised approximate spectral clustering algorithm based on HMRF model," *Inf. Sci.*, vol. 429, pp. 215–228, Mar. 2018.

[24] T. Rakthanmanon and E. Keogh, "Fast shapelets: A scalable algorithm for discovering time series shapelets," in *Proc. 2013 SIAM Int. Conf. Data Min.*, Austin, TX, USA, 2013, pp. 668–676.

[25] E. Keogh and C. A. Ratanamahatana, "Exact indexing of dynamic time warping," in *Proc. 28th Int. Conf. Very Large Data Bases*, Hong Kong, 2002, pp. 406–417.

[26] L. Chen and R. Ng, "On the marriage of Lp-norms and edit distance," in *Proc. 30th Int. Conf. Very Large Data Bases*, Toronto, ON, Canada, 2004, pp. 792–803.

[27] X. Wang, F. Yu, and W. Pedrycz, "An area-based shape distance measure of time series," *Appl. Soft Comput.*, vol. 48, pp. 650–659, Nov. 2016.

[28] J. Peng, H. Wang, J. Li, and H. Gao, "Set-based similarity search for time series," in *Proc. Int. Conf. Manag. Data*, San Francisco, CA, USA, 2016, pp. 2039–2052.

[29] U. Mori, A. Mendiburu, and J. A. Lozano, "Similarity measure selection for clustering time series databases," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 1, pp. 181–195, Jan. 2016.

[30] S. Anand, S. Mittal, O. Tuzel, and P. Meer, "Semi-supervised kernel mean shift clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 6, pp. 1201–1215, Jun. 2014.

[31] C.-L. Liu, W.-H. Hsaio, C.-H. Lee, and F.-S. Gou, "Semi-supervised linear discriminant clustering," *IEEE Trans. Cybern.*, vol. 44, no. 7, pp. 989–1000, Jul. 2014.

[32] S. Xiong, J. Azimi, and X. Z. Fern, "Active learning of constraints for semi-supervised clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 1, pp. 43–54, Jan. 2014.

[33] H. Wang, T. Li, T. Li, and Y. Yang, "Constraint neighborhood projections for semi-supervised clustering," *IEEE Trans. Cybern.*, vol. 44, no. 5, pp. 636–643, May 2014.

[34] S. Huang, H. Wang, T. Li, Y. Yang, and T. Li, "Constraint co-projections for semi-supervised co-clustering," *IEEE Trans. Cybern.*, vol. 46, no. 2, pp. 3047–3058, Dec. 2016.

[35] Z. Yu *et al.*, "Adaptive ensembling of semi-supervisered clustering solutions," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 8, pp. 1577–1590, Aug. 2017.

[36] J. Zhou, S.-F. Zhu, X. Huang, and Y. Zhang, "Enhancing time series clustering by incorporating multiple distance measures with semi-supervised learning," *J. Comput. Sci. Techol.*, vol. 30, no. 4, pp. 859–873, Jul. 2015.

[37] H. A. Dau, N. Begum, and E. Keogh, "Semi-supervision dramatically improves time series clustering under dynamic time warping," in *Proc. ACM Int. Conf. Inf. Knowl. Manag.*, Indianapolis, IN, USA, Oct. 2016, pp. 999–1008.

[38] G. E. A. P. A. Batista, E. J. Keogh, O. M. Tataw, and V. M. de Souza, "CID: An efficient complexity-invariant distance for time series," *Data Min. Knowl. Disc.*, vol. 28, no. 3, pp. 634–669, 2014.

[39] J. Mei, M. Liu, Y.-F. Wang, and H. Gao, "Learning a mahalanobis distance-based dynamic time warping measure for multivariate time series classification," *IEEE Trans. Cybern.*, vol. 46, no. 6, pp. 1363–1374, Jun. 2016.

[40] Y.-S. Jeong, M. K. Jeong, and O. A. Omitaomu, "Weighted dynamic time warping for time series classification," *Pattern Recognit.*, vol. 44, no. 9, pp. 2231–2240, Sep. 2011.

[41] T. Górecki and M. Łuczak, "Using derivatives in time series classification," *Data Min. Knowl. Disc.*, vol. 26, no. 2, pp. 310–331, 2013.

[42] K. Yang and C. Shahabi, "A PCA-based similarity measure for multivariate time series," in *Proc. 2nd ACM Int. Workshop Multimedia Databases*, Washington, DC, USA, Nov. 2004, pp. 65–74.

[43] A. Stefan, V. Athitsos, and G. Das, "The move-split-merge metric for time series," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 6, pp. 1425–1438, Jun. 2013.

[44] B.-K. Yi, H. V. Jagadish, and C. Faloutsos, "Efficient retrieval of similar time sequences under time warping," in *Proc. 14th Int. Conf. Data Eng.*, Washington, DC, USA, Feb. 1998, pp. 23–27.

[45] S. Salvador and P. Chan, "Toward accurate dynamic time warping in linear time and space," *Intell. Data Anal.*, vol. 11, no. 5, pp. 561–580, Oct. 2007.

[46] Q. Zhu, G. Batista, T. Rakthanmanon, and E. Keogh, "A novel approximation to dynamic time warping allows anytime clustering of massive time series datasets," in *Proc. SIAM Int. Conf. Data Min.*, Anaheim, CA, USA, 2012, pp. 999–1010.

[47] S. T. Mai, X. He, N. Hubig, C. Plant, and C. Bohm, "Active density-based clustering," in *Proc. IEEE 13th Int. Conf. Data Min.*, Dallas, TX, USA, Dec. 2013, pp. 508–517.

[48] S. T. Mai, I. Assent, and M. Storgaard, "AnyDBC: An efficient anytime density-based clustering algorithm for very large complex datasets," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, San Francisco, CA, USA, Aug. 2016, pp. 1025–1034.

[49] R. Ding, Q. Wang, Y. Dang, Q. Fu, H. Zhang, and D. Zhang, "YADING: Fast clustering of large-scale time series data," in *Proc. 41st Int. Conf. Very Large Data Bases (VLDB)*, Aug. 2015, pp. 473–484.

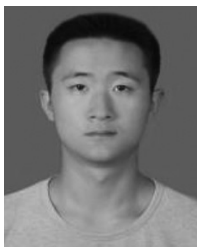
[50] F. Korn and S. Muthukrishnan, "Influence sets based on reverse nearest neighbor queries," in *Proc. ACM SIGMOD Int. Conf. Manag. Data*, Dallas, TX, USA, May 2000, pp. 201–212.

[51] Y. Chen *et al.* (2015). *The UCR Time Series Classification Archive*. [Online]. Available: ww.cs.ucr.edu/∼eamonn/time_series_data/

[52] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. 2nd Int. Conf. Knowl. Disc. Data Min. (KDD)*, Portland, OR, USA, Aug. 1996, pp. 226–231.

Guoliang He received the Ph.D. degree in computer software and theory from Wuhan University, Wuhan, China, in 2007.

He is currently an Associate Professor with the School of Computer Science, Wuhan University. His current research interests include data mining, machine learning, and intelligent algorithms.



Yanzhou Pan received the B.S. degree in computer science from Wuhan University, Wuhan, China, in 2019. He is currently pursuing the M.C.S. degree in engineering department with Rice University, Houston, TX, USA.

His current research interests include data mining and artificial intelligence.

Xuewen Xia received the Ph.D. degree in computer science from Wuhan University, Wuhan, China.

He is currently a Professor with the College of Physics and Information Engineering, Minnan Normal University, Zhangzhou, China. His current research interest includes computational intelligence techniques and their applications.

Jinrong He received the Ph.D. degree in computer software from Wuhan University, Wuhan, China.

He is currently a Lecturer with the College of Mathematics and Computer Science, Yan'an University, Yan'an, China. His current research interests include computer vision, dimensionality reduction, and feature extraction.

Rong Peng received the Ph.D. degree in computer software and theory from Wuhan University, Wuhan, China.

She is a Professor with the School of Computer Science, Wuhan University. Her current research interests include requirements engineering, software engineering, and mobile computing.

Neal N. Xiong received the Ph.D. degree in computer science from Wuhan University, Wuhan, China.

He is currently an Associate Professor with the Department of Mathematics and Computer Science, Northeastern State University, Tahlequah, OK, USA. His current research interests include cloud computing, security and dependability, and parallel and distributed computing.