Zhou J, Zhu SF, Huang X *et al.* Enhancing time series clustering by incorporating multiple distance measures with semi-supervised learning. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 30(4): 859–873 July 2015. DOI 10.1007/s11390-015-1565-7

Enhancing Time Series Clustering by Incorporating Multiple Distance Measures with Semi-Supervised Learning

Jing Zhou [1,2] (周 竞), Shan-Feng Zhu [1,2] (朱山风), *Member, CCF, ACM*
Xiaodi Huang [3] (黄晓地), *Member, ACM, IEEE*, and Yanchun Zhang [1,4,5] (张彦春), *Member, CCF*

[1] *School of Computer Science, Fudan University, Shanghai 200433, China*
[2] *Shanghai Key Laboratory of Intelligent Information Processing, Fudan University, Shanghai 200433, China*
[3] *School of Computing and Mathematics, Charles Sturt University, Albury, NSW 2640, Australia*
[4] *School of Engineering and Science, Victoria University, Melbourne, Victoria 8001, Australia*
[5] *Shanghai Key Laboratory of Data Science, Fudan University, Shanghai 201203, China*

E-mail: {jingzhou12, zhusf}@fudan.edu.cn; xhuang@csu.edu.au; Yanchun.Zhang@vu.edu.au

Received February 1, 2015; revised March 25, 2015.

Abstract Time series clustering is widely applied in various areas. Existing researches focus mainly on distance measures between two time series, such as dynamic time warping (DTW) based methods, edit-distance based methods, and shapelets-based methods. In this work, we experimentally demonstrate, for the first time, that no single distance measure performs significantly better than others on clustering datasets of time series where spectral clustering is used. As such, a question arises as to how to choose an appropriate measure for a given dataset of time series. To answer this question, we propose an integration scheme that incorporates multiple distance measures using semi-supervised clustering. Our approach is able to integrate all the measures by extracting valuable underlying information for the clustering. To the best of our knowledge, this work demonstrates for the first time that the semi-supervised clustering method based on constraints is able to enhance time series clustering by combining multiple distance measures. Having tested on clustering various time series datasets, we show that our method outperforms individual measures, as well as typical integration approaches.

Keywords time series analysis, clustering, dynamic programming, information search and retrieval

1 Introduction

Analysis of time series data plays a critical role in many domains such as medical science, finance, meteorology, space science, oceanography, motion capture, and environmental science[1-4]. Time series clustering is one of the most popular tasks in time series data mining[3,5-8]. Time series data differ from others in that all of their data points have a time property. This fact leads to increasing the complexity and difficulty in clustering time series data. Based on different similarity measures[9] used, there are three types of time series clustering.

The first type is based on the similarities in time, which tends to cluster temporal sequences varying similarly in time. A simple way of measuring similarities is to use Euclidean distance or other L_p norms. Such poor similarity measures have been reported[10], since two similar series may start at different time points or move at different speeds. This means that there may be a shift, stretching, or shrinking in the time dimension.

The second type of clustering takes into account the similarities in shapes. It groups together time series with common shapes or sub-shapes. Two popular approaches to measuring distances based on shapes include sequence transformation using warping tech-

Regular Paper
Special Section on Data Management and Data Mining
The work was partially supported by the National Natural Science Foundation of China under Grant Nos. 61332013, 61272110, and 61370229, and the National Key Technology Research and Development Program of China under Grant No. 2013BAH72B01.

niques and algorithms that find common features among sub-patterns. As an alternative, elastic distance measures, such as dynamic time warping (DTW), calculate an optimal match between two given sequences. For instance, similarities in walking patterns could be detected with DTW, even if people walk at different speeds. Edit-based measures, like the longest common subsequence (LCSS)[11], and shapelets-based technique[12], are two common approaches to finding similar sub-shapes. Time series shapelets are small, local subsequences that are representatives of a class. For example, shapelets could help distinguish the abnormal heartbeat of a sudden for cardiologists.

The last type is on the basis of similarities in change. Such a type of clustering attempts to cluster time series by the similarity in the way of how they vary along the time dimension. Among two data points with the same value from two sequences, for instance, one is in a rising trend, while the other is in a falling trend. We would prefer not to align a rising trend to a falling trend. However, elastic distance measures, like DTW, as mentioned previously, are not able to handle this situation. The broad approach to solving this problem is called derivative dynamic time warping (DDTW)[13], which is based on first-order differences of time series.

From the above three types of distance measures on time series, our initial goal of this work is to find a measure that outperforms the others in terms of spectral clustering. Through extensive experiments on various datasets of time series, we show that, however, there is no single winner. Each measure has its own strength and weakness.

This fact motivates us to combine multiple distance measures by taking advantage of their strengths. In addition, for time series clustering, we normally do not know which measure is the best for a given data due to the lack of labeled data. It may be effective to combine different measures.

The further question is how to make full use of the strengths of different measures. Instead of simply weighting different measures, we utilize them to generate prior knowledge as constraints for semi-supervised clustering. In particular, these constraints are generated in the form of pairwise constraints indicating the relationship between two data points; that is, whether these samples belong to the same cluster or not, denoted as must-link and cannot-link constraints, respectively. As we know, the prior knowledge for a domain is not always available in reality. Our solution also provides a new way of tackling this problem.

In summary, we propose a new framework based on semi-supervised clustering that integrates multiple distance measures. It is widely believed that the prior knowledge used by semi-supervised clustering can improve the performance. Our experiments have also demonstrated that for clustering time series data, our integrated method is better than not only any single typical distance measure, but also other rival well-known integration schemes. To the best of our knowledge, this is the first time that semi-supervised clustering based on links is used in the time series domain.

The rest of this paper is organized as follows. In Section 2, we review related work on time series clustering. Section 3 briefly introduces seven distance measures we use in our method, and presents our approach that effectively incorporates multiple distance measures. Section 4 describes our experimental settings, reports and analyzes experimental results. Section 5 concludes this paper.

2 Related Work

One key step of time series clustering is to measure the distances between compared sequences. Actually, many measures are available for time series clustering, including variant of dynamic time warping, such as weighted dynamic time warping[14], kernel dynamic time warping[15], derivative dynamic time warping[13], and edit distance based methods, like longest common subsequence[11], time warping edit distance[16]. All of these measures are common in that they try to find an optimal alignment between two time series along the time dimension so as to handle outliers, as outliers are quite common in the time series domain. A distribution-based distance measure was proposed by shao *et al.*[17] to deal with multi-dimension sequences. Recently, a new concept, called shapelets, was introduced[12]. It is believed that time series data may contain a lot of useless data, while some common sub-patterns, shapelets, are highly discriminative. In addition, an improved symbolic aggregate approximation (SAX) distance measure was proposed by Sun *et al.*[18], which integrates a weighted trend distance. In [19], an adaptive approximation was reported to create compact approximations of time series, and then an efficient method based on indexing was used to compare time series.

For clustering time series data, various clustering algorithms have been developed. In [20], expectation maximization (EM) was adopted to cluster time

series according to an initial approximation of raw data by wavelets. A hierarchy clustering method using DTW distance was proposed[21]. Hidden Markov models (HMMs) was used to cluster multivariate time series[22-23]. Three fuzzy clustering-based methods using DTW were reported[24]. In [25], a Bayesian method was presented to cluster sequences by modeling temporal sequences as Markov chains. The shapelets-based clustering was introduced by [8] as well.

Clustering time series has been receiving great attention. Nevertheless, most of studies focus on clustering time series using an individual measure. How to integrate multiple measures is not well resolved. The two widely referenced studies are [26] and [27]. Their approaches basically transform time series sequences into different representations. This idea differs from our combination of multiple distance measures. Moreover, an ensemble classification method that combines several elastic distance measures[28] demonstrated promising performance on extensive time series data. It has also been shown that there is no significant difference between elastic distance measures for classification where 1NN (one nearest neighbor) is applied.

Semi-supervised learning algorithms have emerged these years which are considered as a great alternative to traditional supervised and unsupervised learning algorithms. To deal with prior knowledge, semi-supervised kernel K-means (SSKK) transforms the clustering distance measure by weighted kernel K-means with reward and penalty terms[29]. In [30], complex structure data, such as graphs, are taken into account in SSKK. In [31], constraints are incorporated into semi-supervised non-negative matrix factorization by refactoring the former affinity matrix in non-negative matrix factorization. A time-efficient semi-supervised learning using multiple graphs that combines soft spectral clustering with label propagation was reported[32].

3 Weighted Semi-Supervised Normalized Cut

In this section, we first describe common distance measures and then present our approach. The notations used are listed in Table 1.

3.1 Distance Measures

To measure the distance between temporal sequences from multiple aspects, we choose seven popular measures, which range from shape-based measures to change-based ones.

3.1.1 Dynamic Time Warping

Dynamic time warping[33] is based on the edit distance that looks for the optimal alignment between two sequences in the time dimension. Given two time series, $X = \{x_1, x_2, ..., x_n\}$ and $Y = \{y_1, y_2, ..., y_m\}$, we can obtain an $n \times m$ distances matrix $\boldsymbol{D}$ where the (i, j) element of $\boldsymbol{D}$ is distance $d(x_i, y_j)$ between data points x_i and y_j. Euclidean distance is used to measure the distance between two points. A warping path, $P = < (e_1, f_1), (e_2, f_2), ..., (e_s, f_s) >$ where $\max(n, m) \leqslant s \leqslant n + m - 1$, is a set of points (i.e., pairs of the index) that satisfy the following three constraints: 1) boundary condition: $(e_1, f_1) = (1, 1)$ and $(e_s, f_s) = (n, m)$, which means the warping path must start and end in diagonally opposite corner cells of

Table 1. Notations in Distance Measures

Notation	Description
X, Y	Time series containing an order set of real values, such as $X = \{x_1, x_2, ..., x_n\}$
x_i, y_i	Data points in time series at index i
X_i, Y_i	Subsequences with time index varying from 1 to i, $X_i = \{x_1, x_2, ..., x_i\}, Y_i = \{y_1, y_2, ..., y_i\}$
n, m	Length of time series
$d(x_i, y_j)$	Distance between two data points x_i and y_j
$d(X_i, Y_j)$	Distance between two subsequences X_i and Y_j
$\boldsymbol{D}$	Distance matrix between two time series, where $D(i, j)$ is the distance $d(x_i, y_j)$
w	Window parameter for elastic distance measure
P	Warping path found by elastic distance measure, like $P = < (e_1, f_1), (e_2, f_2), ..., (e_s, f_s) >$
p_i	Path point in P at index i, $p_i = (e_i, f_i)$, where e_i and f_i are indexes in two time series
$\boldsymbol{S}$	Similarity matrix over time series dataset, where $S(i, j)$ is the similarity between two time series
M	Must-link set, like $M = < (a_1, b_1), ..., (a_s, b_s) >$, where (a_i, b_i) is a pair of time series index in data
C	Cannot-link set defined similar to M
M_{weight}	Weighted must-link set, $M_{\text{weight}} = < (a_1, b_1, w_1), ..., (a_s, b_s, w_s) >$, where w_i is the related frequency
C_{weight}	Weighted cannot-link set

$\boldsymbol{D}$; 2) continuity condition: $0 \leqslant e_{i+1} - e_i \leqslant 1$ and $0 \leqslant f_{i+1} - f_i \leqslant 1$ for all $i \leqslant s$, which means adjacent cells are the allowable steps; and 3) monotonic condition, $e_i \leqslant e_{i+1}$ and $f_i \leqslant f_{i+1}$, which restricts the directions of the path.

The DTW distance between two temporal sequences is a warping path that minimizes the total distance.

$$d_{\mathrm{DTW}} = \min\left(\sum_{i=1}^{s} p_i\right),$$

where $p_i = D(e_i, f_i)$. Dynamic programming can be utilized to find the minimized path by

$$\begin{aligned} & d_{\mathrm{DTW}}(X_i, Y_j) \\ = \ & d(x_i, y_j) + \min\Big\{d_{\mathrm{DTW}}(X_i, Y_{j-1}), \\ & d_{\mathrm{DTW}}(X_{i-1}, Y_{j-1}), d_{\mathrm{DTW}}(X_{i-1}, Y_j)\Big\}, \end{aligned} \quad (1)$$

where X_i denotes a sequence with the time index varying from 1 to i. The time complexity of calculating DTW distances is $O(nm)$. The constrained version of DTW[34] can reduce the time complexity by adding a window parameter that restricts the maximum allowable distance between any pairs of indexes in a warping path. If w is the warping window length, then the optimal path is restricted so that we have $|e_i - f_i| \leqslant w$. Fig.1 shows an optimal warping path and alignment between two sequences by DTW. A window is used to constrain the warping operation.

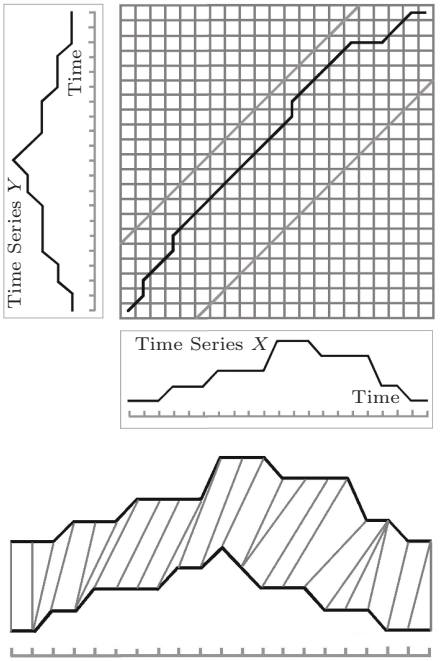


Fig.1. Optimal warping by DTW.

3.1.2 Weighted Dynamic Time Warping

Weighted dynamic time warping (WDTW) is a penalty-based DTW[14]. According to the phase difference between two points, it penalizes points to avoid the minimum distance distortion by outliers. It is believed that neighbor points are valuable. Thus a larger weight penalty is imposed if the phase difference is higher, and the smaller weight one, otherwise. WDTW is an alternative to the constrained form of DTW using a warping window[28]. When computing the distance matrix $\boldsymbol{D}$, we calculate the distance between two points x_i and y_j as $d(x_i, y_j) = ||w_{|i-j|}(x_i - y_j)||$, where $w_{|i-j|}$ is the weight penalty between x_i and y_j.

In addition, a new modified logistic weight function[14] assigns weights to WDTW according to the phase difference between two points:

$$w_{|i-j|} = \frac{w_{\max}}{1 + \exp(-g(|i-j| - m/2))}, \quad (2)$$

where $w_{\max}$ is the desired upper bound for a weight that is set to 1 usually, m the sequence length, and g a parameter to control the level of penalization. The smaller g, the less penalty imposed. Four kinds of weights are obtained by setting different values to g including constant weight, linear weight, sigmoid weight, and two distinct weights. In a case of two distinct weights, the first one-half and the second one-half would be assigned to different weights.

3.1.3 Kernel Dynamic Time Warping

A regularized form of DTW, called kernel dynamic time warping (KDTW)[15], constructs positive definite kernels from DTW. DTW searches only for one of optimal warping paths, while KDTW uses the kernel trick to examine all matching paths of existing subsequences that can help align points better and penalize misalignments. Hence, the min operation in (1) is replaced by a sum operation in KDTW. A symmetric corridor function is introduced to limit summation and computational costs.

$$d_{\mathrm{KDTW}}(X_i, Y_j) = d^{xy}_{\mathrm{KDTW}}(X_i, Y_j) + d^{xx}_{\mathrm{KDTW}}(X_i, Y_j),$$

where

$$\begin{aligned} & d^{xy}_{\mathrm{KDTW}}(X_i, Y_j) \\ = \ & \frac{1}{3} e^{\frac{-d(x_i, y_j)}{\sigma}} \Big\{\mathrm{h}(i-1, j) d^{xy}_{\mathrm{KDTW}}(X_{i-1}, Y_j) + \\ & \mathrm{h}(i-1, j-1) d^{xy}_{\mathrm{KDTW}}(X_{i-1}, Y_{j-1}) + \\ & \mathrm{h}(i, j-1) d^{xy}_{\mathrm{KDTW}}(X_i, Y_{j-1})\Big\}, \end{aligned}$$

and

$$\begin{aligned} & d^{xx}_{\mathrm{KDTW}}(X_i, Y_j) \\ = & \frac{1}{3}\Big\{\mathrm{h}(i-1,j)d^{xx}_{\mathrm{KDTW}}(X_{i-1}, Y_j)e^{\frac{-d(x_i,y_j)}{\sigma}} + \\ & \Delta_{i,j}\mathrm{h}(i,j)d^{xx}_{\mathrm{KDTW}}(X_{i-1}, Y_{j-1})e^{\frac{-d(x_i,y_j)}{\sigma}} + \\ & \mathrm{h}(i,j-1)d^{xx}_{\mathrm{KDTW}}(X_i, Y_{j-1})e^{\frac{-d(x_i,y_j)}{\sigma}}\Big\}, \end{aligned}$$

where $\Delta_{i,j}$ is the Kronecker's symbol, h(.,.) symmetric positive function that controls symmetric corridor, σ controls contributions of local elementary costs, and $d(.,.)$ the Euclidean distance defined on $\mathbb{R}^k$.

3.1.4 Derivative Dynamic Time Warping

Time series may have local differences in data point values as well. For instance, a local peak in one series may be higher than that in the other sequence. The elastic distance measures, like DTW, are not able to deal with this case, leading to aligning a single point on one temporal sequence to a large subsection of the other sequence. A modified DTW, called derivative dynamic time warping (DDTW)[13], takes the first derivative of time series into account. Considering a sequence $X = \{x_1, x_2, ..., x_n\}$, we estimate the derivative sequence as follows,

$$x'_i = \frac{(x_i - x_{i-1} + (x_{i+1} - x_{i-1})/2)}{2}, \quad 1 < i < n.$$

Note that the first and the last elements are not given in the above formula, which are replaced by the second one and the penultimate one respectively. Besides, window parameter w in DTW can be applied in DDTW as well in order to limit the computational cost in dynamic programming.

In addition, a weighted-based DDTW (WDDTW)[14] weights distances by using the logistic function defined in (2).

3.1.5 Longest Common Subsequence

The longest common subsequence (LCSS) was first applied to find the longest common subsequence between two strings on the basis of edit distance. It was extended in [11] for its use in numeric time series by introducing a threshold ϵ to define the maximum difference between a pair of data points that can be considered to be equal. The main idea of LCSS is to calculate the greatest number of matching pairs in two sequences by inserting or deleting mismatched elements. It can be solved by using the following recursion,

$$LCSS(X_i, Y_j) = \begin{cases} 1 + LCSS(X_{i-1}, Y_{j-1}), \\ \quad \text{if } d(x_i, y_j) < \epsilon \text{ and } |i-j| < w, \\ \max\Big\{LCSS(X_i, Y_{j-1}), \\ \quad LCSS(X_{i-1}, Y_j)\Big\}, \\ \quad \text{otherwise}, \end{cases}$$

where w is a window parameter like the one in DTW. Then the normalized distance based on LCSS can be transformed as follows,

$$d_{\mathrm{LCSS}}(X_i, Y_i) = 1 - \frac{LCSS(X_i, Y_i)}{\min(n, m)}.$$

3.1.6 Time Warp Edit Distance

The time warp edit distance (TWED)[16] incorporates the properties of LCSS and DTW not only to allow time warping, but also to use the edit distance with the L_p norm. Unlike the normal edit distance, TWED replaces the *insert* operation with the *delete* operation. It again contains three operations of $delete_x$, $delete_y$ and *match*, where $delete_x$ stands for deleting an element from time series X to map series Y. The final formulation is given as follows:

$$\begin{aligned} d_{\mathrm{TWED}}(X_i, Y_j) = \min\Big\{ & d_{\mathrm{TWED}}(X_{i-1}, Y_j) + \Gamma(x'_i \to \Lambda), \\ & d_{\mathrm{TWED}}(X_{i-1}, Y_{j-1}) + \Gamma(x'_i \to y'_j), \\ & d_{\mathrm{TWED}}(X_i, Y_{j-1}) + \Gamma(\Lambda \to y'_j)\Big\}, \end{aligned}$$

with

$$\begin{aligned} \Gamma(x'_i \to \Lambda) &= d(x_i, x_{i-1}) + \upsilon(t_{x_i} - t_{x_{i-1}}) + \lambda, \\ \Gamma(x'_i \to y'_j) &= d(x_i, y_j) + d(x_{i-1}, y_{j-1}) + \\ &\qquad \upsilon(|t_{x_i} - t_{y-j}| + |t_{x_{i-1}} - t_{y_{j-1}}|), \\ \Gamma(\Lambda \to y'_j) &= d(y_i, y_{i-1}) + \upsilon(t_{y_i} - t_{y_{i-1}}) + \lambda, \end{aligned}$$

where x'_i is the i-th sample of time series X. It considers $x'_i \in S \times T$, where $S \subset \mathbb{R}^d$ with $d \geqslant 1$ embeds the multidimensional space variables, and $T \subset \mathbb{R}$ embeds the time-stamp variable, then we have $x'_i = (x_i, t_{x_i})$ where $x_i \in S$ and $t_{x_i} \in T$.

The parameter υ, called stiffness, controls the level of warping. When $\upsilon = 0$, TWED is equivalent to DTW, and when $\upsilon = \infty$, TWED gives Euclidean distance. λ is a constant parameter used to penalize misalignment.

3.1.7 Unsupervised-Shapelets

Sometimes, time series contain a lot of significant noises or extraneous data that limit the performance

of time series clustering. In this case, the clustering accuracy can be improved by using only some common sub-shape features and ignoring the rest of data. Time series shapelets, introduced by Ye and Keogh[12], are small and local subsequences that are representatives of a class. Most of researches on discovering shapelets require labeled training data to guide the algorithm. Unsupervised-shapelets (u-shapelets)[8], however, is able to obtain shapelets without any knowledge of class labels.

The basic idea of u-shapelets is to search greedily for u-shapelets, which divides and discards a sub-collection of sequences from the rest of data until none of data is retained to be divided. A dissociation measure, called *gap*, is defined as,

$$gap = \mu_B - \sigma_B - (\mu_A + \sigma_A),$$

where A is the subset of time series similar to the shapeplet, B the rest of data, μ_A the mean distance between the shapelet and the subset A, and σ_A the standard deviation between the shapeplet and the subset A. The greedy search is aimed to find shapelets that maximize the separation gap. In order to avoid unbalance between A and B, for instance, either A or B contains a single time series only. The ratio of A and B has to meet the condition of $(\frac{1}{k}) < |D_A|/|D_B| < (1 - \frac{1}{k})$, where k is the number of clusters.

With shapelets, a distance matrix $\boldsymbol{DIS}$ between shapelets and time series is able to be obtained by the Euclidean distance. The distances of a time series with all the shapelets can be considered as the new features of the given sequence. The shapelets-based distances between temporal sequences are then calculated on the new feature space using the Euclidean distance.

3.2 Normalized Cut

We use normalized cut (NCut)[35] to group time series, since it is able to handle any kinds of distance measures. NCut is the spectral clustering method based on graph-partitioning. The similarity matrix over data in NCut is represented as a graph where the similarity between two instances is the weight of an edge connecting two samples. The key idea of spectral clustering is to minimize the number of cuts so that the sum of inter-cluster similarities is minimized.

NCut has been validated to overwhelm other existing measures due to the avoidance of the unnatural bias of separating small groups of points. The NCut is defined as follows:

$$NCut(A, B) = \frac{cut(A, B)}{assoc(A, V)} + \frac{cut(A, B)}{assoc(B, V)},$$

where $cut(A, B) = \sum_{u \in A, t \in B} w(u, t)$ is the total connections from the samples in A to the ones in B, $assoc(A, V) = \sum_{u \in A, t \in V} w(u, t)$ is the total connections from instances in A to all the nodes in a graph, and so is $assoc(B, V)$. It can be seen that the original problem of minimizing NCut can be cast as one of minimizing the matrix trace. Although minimizing normalized cuts has been proved to be NP-hard, it can be computed efficiently by solving a generalized eigenvalue problem with some proper relaxation. For the generalized eigenvalue problem, it typically attempts to project each sample into the eigen-space spanned by K eigenvectors, and then to apply the K-means algorithm in the eigen-space. The objective function of NCut is given below,

$$\begin{aligned} F_{\mathrm{NCut}} &= \sum_{k=1}^{K} \frac{\boldsymbol{X}_k^{\mathrm{T}}(\boldsymbol{G} - \boldsymbol{W})\boldsymbol{X}_k}{\boldsymbol{X}_k^{\mathrm{T}}\boldsymbol{G}\boldsymbol{X}_k} \\ &= K - \mathrm{Tr}(\boldsymbol{Y}^{\mathrm{T}}\boldsymbol{G}^{-\frac{1}{2}}\boldsymbol{W}\boldsymbol{G}^{-\frac{1}{2}}\boldsymbol{Y}), \end{aligned}$$

where $\boldsymbol{X}_k = (x_{1k}, x_{2k}, ..., x_{Nk})^{\mathrm{T}}$ is the binary indicator vector of cluster k, and $\boldsymbol{G}$ is the diagonal matrix such that we have $\boldsymbol{G} \cdot \boldsymbol{e} = \boldsymbol{W} \cdot \boldsymbol{e}$, $\boldsymbol{e} = (1, 1, ..., 1)^{\mathrm{T}}$, and $\boldsymbol{Y} = (\boldsymbol{Y}_1, \boldsymbol{Y}_2, ..., \boldsymbol{Y}_K)$, $\boldsymbol{Y}_k = \boldsymbol{G}^{\frac{1}{2}}\boldsymbol{X}_k / ||\boldsymbol{G}^{\frac{1}{2}}\boldsymbol{X}_k||$.

3.3 Linear Combination Method

A common method to integrate different types of similarity matrixes is a linear combination method (LCM)[36]. We first transfer all the distance matrixes into similarity matrixes and normalize them. Then we combine them with equal weights linearly as follows:

$$\boldsymbol{S} = \frac{1}{p} \sum^{ms \in MS} \boldsymbol{S}_{ms},$$

where MS is the set of names of distance measures, which includes DTW, WDTW, KDTW, DDTW, WDDTW, LCSS, and U-shapelets in our studies. $\boldsymbol{S}_{ms}$ is the similarity matrix based on the related distance measure, $\boldsymbol{S}$ is the final integrated similarity matrix, and p is the amount of measures used.

3.4 Hybrid Bipartite Graph Formation

The LCM, described previously, is an input-based (i.e., similarity matrix) integration method. Another

typical way of combining different measures is to ensemble predicted labels by different individual measures. It is in fact an output-based approach as it relies on clustering results on individual measures. As a well-accepted graph ensemble clustering scheme, hybrid bipartite graph formation (HBGF)[37] has been well validated[38-39]. It generates a "meta-graph" (in which both input instances and input clusters are nodes) as a bipartite graph, in which nodes are clustered. This is a one-step procedure which partitions given clusters into "meta-clusters" and assigns each instance to one meta-cluster simultaneously.

3.5 Semi-Supervised Normalized Cut

Although LCM and HBGF are able to integrate multiple distance measures, both of them have the underlying weaknesses. In LCM, on the one hand, choosing a set of suitable weights for different types of similarities is a complicated task, especially when training data are unavailable. On the other hand, the contributions of similarity values in a specific similarity matrix could be different, for instance, similarity values within the top and bottom ones of a similarity matrix usually provide more useful information for the boundaries of such true clusters which should be paid more attention. HBGF integrates clustering results based on individual similarity matrixes. Sometimes, some distance measures, however, cannot capture the real relationship among data points, which leads to very poor clustering results. As such, the performance of HBGF would be significantly deteriorated by those poor clustering results as a result of ineffective distance measures. Nevertheless, we use semi-supervised clustering to combine multiple distance measures. It makes use of the most valuable and reliable information from different types of distance measures to generate constraints as prior knowledge. Such knowledge is able to guide the semi-supervised clustering process efficiently and to obtain the better performance.

We use semi-supervised normalized cut (SSNCut) to cluster time series over the integrated must-link set and and cannot-link set. As a stable method, SSNCut has been proved to outperform multiple well-known semi-supervised clustering methods[40]. Another reason why we choose SSNCut is that it can be conducted over any distance measures. In contrast, many other algorithms on semi-supervised clustering have a strong restriction on distance measures for the basic distance/similarity matrix over instances, such as the use of Euclidean distance in semi-supervised kernel K-means[29], and the inner product in semi-supervised non-negative matrix factorization[31]. This restriction on the distance measure limits their usage in the time series domain.

One advantage of the constrained Normalized Cut (CNC) algorithm[41] is able to incorporate must-link constraints for NCut. To enforce prior knowledge, a penalty term is added as must-link constraints to the original NCut cost function. A parameter α is also introduced into the penalty term to control the degree of enforcement of positive constraints. The larger the parameter, the stronger the enforcement of prior knowledge.

Nevertheless, CNC incorporates only must-link constraints into NCut without cannot-link constraints, which also play a critical role in improving clustering performance. Therefore, it has been extended[40] to accommodate cannot-link constraints by developing a new distinguished scheme called Semi-supervised Normalized Cut (SSNCut). Similar to CNC, another penalty item based on cannot-link constraints is added with a parameter β to adjust the effect of negative constraints on the objection function. It has been proved that SSNCut is able to reveal remarkable performance improvements with very limited constraints. The objective function of SSNCut is described as follows.

$$\begin{aligned} F_{\mathrm{SSNC}} &= \sum_{k=1}^{K} \frac{\boldsymbol{X}_k^{\mathrm{T}}(\boldsymbol{G}-\boldsymbol{W}^l)\boldsymbol{X}_k}{\boldsymbol{X}_k^{\mathrm{T}}\boldsymbol{G}\boldsymbol{X}_k} + \\ &\quad \alpha||\boldsymbol{U}\boldsymbol{X}||^2 + \beta\mathrm{Tr}(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{Z}\boldsymbol{X}) \\ &= K - \mathrm{Tr}(\boldsymbol{Y}^{\mathrm{T}}\boldsymbol{G}^{-\frac{1}{2}}\boldsymbol{W}^l\boldsymbol{G}^{-\frac{1}{2}}\boldsymbol{Y}) + \\ &\quad \alpha\mathrm{Tr}(\boldsymbol{Y}^{\mathrm{T}}\boldsymbol{G}^{-\frac{1}{2}}\boldsymbol{U}^{\mathrm{T}}\boldsymbol{U}\boldsymbol{G}^{-\frac{1}{2}}\boldsymbol{Y}) + \\ &\quad \beta\mathrm{Tr}(\boldsymbol{Y}^{\mathrm{T}}\boldsymbol{G}^{-\frac{1}{2}}\boldsymbol{Z}\boldsymbol{G}^{-\frac{1}{2}}\boldsymbol{Y}) \\ &= K - \mathrm{Tr}(\boldsymbol{Y}^{\mathrm{T}}\boldsymbol{G}^{-\frac{1}{2}}(\boldsymbol{W}^l - \\ &\quad \alpha\boldsymbol{U}^{\mathrm{T}}\boldsymbol{U} - \beta\boldsymbol{Z})\boldsymbol{G}^{-\frac{1}{2}}\boldsymbol{Y}), \end{aligned}$$

where $\boldsymbol{U} = (\boldsymbol{U}_1, \boldsymbol{U}_2, .., \boldsymbol{U}_s)^{\mathrm{T}}$, s is the number of must-links, and each $\boldsymbol{U}_p = (u_{1p}, u_{2p}, ..., u_{np})$ encodes a constraint as we have $u_{ip} = 1$ and $u_{jp} = -1$ and the rest of all are 0. $\boldsymbol{Z} = \sum_{i,j}(\boldsymbol{L}_i\boldsymbol{L}_j^{\mathrm{T}} + \boldsymbol{L}_j\boldsymbol{L}_i^{\mathrm{T}})$ encodes all the cannot-links. Both $\boldsymbol{L}_i^{\mathrm{T}}$ and $\boldsymbol{L}_j^{\mathrm{T}}$ are selection vectors, for example, $\boldsymbol{L}_i^{\mathrm{T}} = (...0, 1, 0, ...)$, where the element of 1 represents a cluster to be chosen.

To incorporate weights of constraints, we multiply each must-link vector $\boldsymbol{U}_p$ with its weight as $\boldsymbol{U}_q = w_{i,j}(u_{1q}, u_{2q}, ..., u_{nq})$ where i and j are the related instances constrained by must-link q. For cannot-link constraints, we modify the cannot-link matrix as $\boldsymbol{Z} = \sum_{i,j} w_{i,j}(\boldsymbol{L}_i\boldsymbol{L}_j^{\mathrm{T}} + \boldsymbol{L}_j\boldsymbol{L}_i^{\mathrm{T}})$. In order to differentiate our modified SSNCut from the original SSNCut, we name

our approach as "wSSNCut" standing for weighted semi-supervised normalized cut.

3.6 Constraint Sets

At first, we generate initial constraints from each distance measure. A typical approach to obtaining constraints is to generate must-links if two time series have a relatively small distance, and cannot-links, otherwise. Thus, a cut-off trick is used to generate must-link and cannot-link constraints from distance matrixes. We first extract the top p percentage and the bottom p percentage of elements from every distance matrix calculated by measures described in Subsection 3.1 (except for TWED, we use it as the basic distance matrix for the clustering process), and then regard pairs of sequences within top $p\%$ distances as cannot-link constraints, and pairs of time series within bottom $p\%$ distances as must-link constraints. p is a parameter that controls the number of constraints. Thus, based on different measures, we can get multiple constraint sets. Denote MS as the set of names of distance measures including DTW, WDTW, KDTW, DDTW, WDDTW, LCSS, and U-shapelets. Then, we combine all the constraints sets using union operation to calculate the whole possible constraints,

$$M_{\text{union}} = \bigcup^{ms\in MS} M_{ms}, \qquad C_{\text{union}} = \bigcup^{ms\in MS} C_{ms},$$

where $M_{\text{union}} = \{(a_1, b_1), (a_2, b_2), ..., (a_s, b_s)\}$ and $C_{\text{union}} = \{(c_1, d_1), (c_2, d_2), ..., (c_t, d_t)\}$ are two sets of points (i.e., pairs of index) representing locations of must-link and cannot-link, respectively.

To generate weighted constraints, we calculate the frequencies of occurrence of constraints and assign them to related constraints. Then, we get the weighted must-link set $M_{\text{weight}} = \{(a_1, b_1, w_1), (a_2, b_2, w_2), ..., (a_s, b_s, w_s)\}$ and the weight cannot-link set $C_{\text{weight}} = \{(c_1, d_1, w_1), (c_2, d_2, w_2), ..., (c_t, d_t, w_t)\}$, where w_i is the weight of related constraint. Obviously, the w of a constraint could be considered as the confidence of the link constraint.

Sometimes some distance measures may not capture the real relationship among data points. This fact leads to many noises in the constraints. Therefore, we discard those 1-vote constraints whose $w = 1$ to reduce the noises in constraints.

In order to test performance of the link confidence, we design another scheme to integrate constraint sets only using union operation and removing 1-vote constraints without using constraint weights.

3.7 Overview of Our Approach wSSNCut

Our method consists of the following four steps.

1) Compute the distances for all pairs of time series in a dataset using different distance measures. For simplicity, we denote a distance matrix as $\boldsymbol{D}_{ms}$, where ms stands for a distance measure, such as DTW and LCSS. $D_{ms}(i, j)$ is the distance between the i-th and the j-th time series based on the relevant measure. The higher $D_{ms}(i, j)$ is, the less similar they are.

2) Generate must-link and cannot-link constraints according to $\boldsymbol{D}_{ms}$, for each distance measure ms. For simplicity, we denote the must-link set by $M_{ms} = \{(a_1, b_1), (a_2, b_2), ..., (a_s, b_s)\}$ and the cannot-link set by $C_{ms} = \{(c_1, d_1), (c_2, d_2), ..., (c_t, d_t)\}$, where two sets of points (i.e., pairs of index) represent locations of must-link and cannot-link, respectively.

3) Integrate M_{ms} and C_{ms} into M_{weight} and C_{weight}, respectively, for each measure ms. We have $M_{\text{weight}} = \{(a_1, b_1, w_1), (a_2, b_2, w_2), ..., (a_s, b_s, w_s)\}$ and $C_{\text{weight}} = \{(c_1, d_1, w_1), (c_2, d_2, w_2), ..., (c_t, d_t, w_t)\}$, which are the final integrated must-link and cannot-link sets as inputs to semi-supervised clustering. w_i is the frequency of the occurrence of a constraint. The value of w_i could be considered as the confidence of a link constraint.

4) Perform semi-supervised clustering over M_{weight} and C_{weight}. Particularly, we preserve the similarity matrix based on time warping edit distance (TWED) from generating constraints in 2) and using it as the fundamental similarity matrix for the semi-supervised clustering algorithm.

4 Experiments

4.1 Datasets

To demonstrate the performance of our approach, we have conducted experiments on different time series datasets. We have compared our approach on 12 UCR (University of California, Riverside) dataset[①]. In our experiments, we combine the training dataset and the test dataset together in UCR datasets, which are all split into training and testing data for classification. The statistics of these datasets are given in Table 2.

[①]Keogh E, Zhu Q, Hu B *et al.* The UCR time series classification/clustering homepage, 2011. www.cs.ucr.edu/∼eamonn/time_series_data/, Dec. 2014.

As usual, we use the number of clusters as a known parameter.

Table 2. Characteristics of the Benchmark Datasets

Name	Length	Size	Number of Clusters
50Words	270	905	50
CBF	128	930	3
Adiac	176	781	37
Beef	470	120	4
ECGFiveDays	136	884	2
Lighting2	637	121	2
Lighting7	319	143	7
MedicalImages	99	1 141	10
WordSynonyms	270	905	25
Synthetic control	60	600	6
TwoLeadECG	82	1 162	25
Trace	275	200	4

4.2 Evaluation Criteria

We use normalized mutual information (NMI) for evaluating clustering performance, which is widely used in information retrieval and feature selection. NMI quantifies the amount of statistical information shared by the random variables representing cluster assignments and user-labeled class assignments of points. It has been shown in [42] that mutual information excels in other criteria. There are several normalized versions of mutual information. We use the squared version[43] in our experiments, which is defined as:

$$NMI(E;Q) = \frac{I(E;Q)}{\sqrt{H(E) \times \mathrm{H}(Q)}},$$

where E and Q are predicted clusters and correct class labels, respectively, $I(E;Q) = H(E) - H(E|Q)$ is the mutual information between E and Q, $H(E)$ and $H(Q)$ are the entropies of E and Q, respectively, and $H(E|Q)$ the conditional entropy of E given Q.

4.3 Experimental Procedures

To examine the performance of our method, we first explore the parameters of distance measures by searching in a range of empirical values. Then, we select a fixed setting for each measure that is able to obtain good clustering performance on all the datasets generally. Particularly, the range of the window parameters used in DTW, LCSS, and DDTW is selected from $\{\frac{1}{20}m, \frac{1}{10}m, \frac{1}{5}m\}$, where m is the length of time series, and $\frac{1}{5}m$ shows typically good performance in the experiments. Besides, we traverse all the possible lengths of shapelets in the u-shapelets method. Finally, we generate integrated must-link and cannot-link constraint sets by incorporating multiple distance sets. To reduce the noises in constraints, we remove the 1-vote constraints; that is, constraints whose $w = 1$ are discarded. We then conduct semi-supervised clustering on constraint sets and the similarity matrix based on TWED. As we know, the values of α and β in wSSNCut rely on the accuracy of constraints. If we obtain a higher rate of the correct must-link set, for example, we may assign a larger α to amplify the power of must-link constraints; otherwise, a smaller α is set. In the experiments, we set $\alpha = 64$ and $\beta = 8$ for the high accuracy of must-link and cannot-link constraints, respectively, and $\alpha = 0.3$ and $\beta = 0.1$ for the low accuracy of must-link and cannot-link, respectively. Actually, the accuracy can be improved a lot by deleting 1-vote constraints. We find that the mean accuracy is 84.5% for must-link, and 83.5% for cannot-link in our datasets.

4.4 Results

4.4.1 Comparisons with Single Distance Measure

Table 3 reports the NMI scores on 12 datasets by eight different distance measures under NCut. The last column lists the NMI scores, which are obtained by our weighted SSNCut method using top and bottom 1% elements as cannot-link and must-link constraints, described in Subsection 3.6. For each dataset, the highest NMI value is highlighted in boldface. In the following, we first compare the performance of seven existing distance measures. Except for wSSNCut, there is no single distance measure that is able to archive the highest overall NMI scores, even though TWED performs better than the others on six datasets. Generally, TWED performs better than the other seven measures. It outperforms LCSS on all the datasets, and DTW and u-shapelets on 11 datasets, except for only one dataset. That is Lighting7 for DTW and TwoLeadECG for u-shapelets, respectively. Besides, TWED outperforms WDDTW on 10 out of 12 datasets, WDTW and KDTW on 9 out of 12 datasets, and DDTW on 7. It is obvious that no measure is better than the others overall. Each method has its strength. This validates the fact that each measure takes different aspects into account to avoid mismatching. By combining multiple distance measures evaluating from different aspects, we believe this way is able to improve the performance.

Table 3. Comparison with Single Distance Measure

Name	DTW	WDTW	KDTW	DDTW	WDDTW	LCSS	TWED	U-Shapelets	wSSNCut
50Words	0.671 0	0.690 8	0.629 7	0.746 7	0.708 9	0.653 5	0.684 9	0.604 6	**0.811 4**
CBF	0.752 2	0.534 1	0.273 6	0.750 2	0.332 4	0.677 9	0.785 4	0.562 8	**0.959 1**
Adiac	0.549 8	0.567 1	0.658 1	0.671 7	0.667 8	0.643 3	0.649 4	0.518 3	**0.694 8**
Beef	0.319 8	0.321 1	0.366 5	0.258 0	0.331 6	0.329 6	0.366 5	0.220 9	**0.383 4**
ECGFiveDays	0.008 5	0.008 3	0.172 3	0.009 7	0.008 9	0.003 2	0.168 4	0.004 9	**0.213 5**
Lighting2	0.035 6	0.006 7	0.065 2	0.167 0	0.020 7	0.002 7	**0.192 8**	0.064 1	**0.192 8**
Lighting7	0.604 9	0.535 2	0.428 8	0.528 0	0.460 3	0.396 3	0.512 9	0.273 7	**0.615 5**
MedicalImages	0.281 3	0.280 4	0.207 4	0.312 7	0.286 6	0.190 1	0.332 8	0.237 6	**0.369 2**
WordSynonyms	0.534 3	0.548 5	0.488 1	0.566 8	0.508 6	0.493 0	0.538 5	0.301 0	**0.627 4**
Synthetic control	0.804 8	0.772 6	0.049 0	0.678 4	0.409 6	0.562 1	0.905 4	0.790 3	**0.962 7**
TwoLeadECG	0.066 8	0.071 0	0.008 0	0.000 3	0.000 1	0.141 7	0.241 6	0.314 8	**0.837 5**
Trace	0.750 8	0.751 0	0.501 3	**0.799 3**	0.728 7	0.530 2	0.752 7	0.739 9	0.752 7

We now compare our approach with others. As shown in Table 3, it can easily be seen that wSSNCut outperforms other clustering with a single type of distance measure. Our method outperforms on 11 out of 12 datasets. In particular, the highest NMI score by a single measure for the CBF dataset is only around 0.78 (TWED), while the corresponding NMI score is increased to 0.959 1 by our method. In addition, our method performs better than NCut with TWED on the 10 datasets, and achieves equal NMI scores on the other two datasets. This result demonstrates that constraints generated from other distance measures can help improve clustering performance generally. This is because we use the TWED-based similarity matrix in wSSNCut as well.

Additionally, another interesting result is that the majority of measures perform poorly on some datasets in Table 3, such as TwoLeadECG. In contrast, our weighted SSNCut is able to improve performance efficiently. The reason for this is that although various distance measure matrixes cannot distinguish true clusters sometimes, the distance values within the top and the bottom ones usually provide useful information for the boundaries of such true clusters. Our approach uses semi-supervised clustering with the constraints[40] that makes use of these information.

4.4.2 Comparisons with Integrated Methods

In the following, we compare our method with the other two integrated methods. Table 4 compares the NMI scores of rival combination methods with those of SSNCut with various percentages of constraints and different types of integration schemes. The SSNC stands for original SSNCut that uses union operation to combine constraint sets. The percentage implies the percentage of constraints. For example, the $\text{wSSNC}_{1\%}$ column is the results obtained by weighted SSCNut (wSSNCut) under 1% constraints, while $\text{SSNCut}_{2\%}$ col-

Table 4. Comparison with Rival Integration Method and Different Percentages of Constraints

Name	LCM	HBGF	$\text{wSSNC}_{1\%}$	$\text{wSSNC}_{0.5\%}$	$\text{wSSNC}_{2\%}$	$\text{SSNC}_{1\%}$	$\text{SSNC}_{0.5\%}$	$\text{SSNC}_{2\%}$
50Words	0.746 3	0.757 1	**0.811 4**	0.792 8	0.792 7	0.810 3	0.791 2	0.804 6
CBF	0.784 3	0.782 1	0.959 1	0.965 0	0.808 1	**0.976 3**	0.941 6	0.971 6
Adiac	0.670 0	0.670 7	**0.694 8**	0.689 7	0.674 2	0.688 1	0.691 5	0.677 8
Beef	0.283 3	0.318 9	0.383 4	**0.392 5**	0.374 7	0.386 1	0.386 1	0.374 7
ECGFiveDays	0.018 3	0.013 7	0.213 5	0.192 0	**0.235 9**	0.204 9	0.191 2	0.205 0
Lighting2	0.064 3	0.023 0	**0.192 8**	**0.192 8**	0.173 1	**0.192 8**	**0.192 8**	0.173 1
Lighting7	0.561 0	0.550 8	0.615 5	0.534 2	**0.616 0**	0.554 8	0.519 2	0.595 5
MedicalImages	0.340 1	0.325 7	0.369 2	0.359 4	**0.387 8**	0.362 0	0.335 7	0.384 0
WordSynonyms	0.586 2	0.588 1	0.627 4	0.606 4	**0.631 6**	0.601 5	0.596 9	0.624 4
Synthetic control	0.799 5	0.791 3	0.962 7	**0.970 0**	0.820 8	0.966 1	**0.970 0**	0.959 5
TwoLeadECG	0.285 9	0.094 5	0.837 5	0.540 6	**0.966 8**	0.676 0	0.488 8	0.904 0
Trace	0.751 0	0.751 0	0.752 7	0.752 7	**0.813 6**	0.752 7	0.752 7	**0.813 6**

umn lists results achieved by SSNCut under 2% constraints. From the results, we can clearly see that our method outperforms LCM and HBGF against all the datasets. As an example, SSNCut on the Beef dataset, among all the six settings, achieves the lowest NMI score of 0.374 7 with 2% constraints by using the weighted combination. This score is much higher than the best one of the other two rival methods, i.e., 0.318 9 by HBGF. Actually, almost all of NMI scores by wSSNCut are higher than those by LCM and HBGF, except for only one case which is under 0.5% constraints on the Lighting7 dataset. Besides, we find that neither LCM nor HBGF is able to handle the case in which most of distance measures fail in differentiating clusters mentioned in Subsection 4.4.1. For example, in a case of TwoLeadECG, NMI by LCM is 0.285 9, and NMI by HBGF is 0.094 5, while NMI by wSSNCut is improved to 0.540 6 at least and 0.966 8 at most. When the most distance measures perform poorly, the input information, neither the integrated similarity matrix, nor cluster labels by NCut with different measures, for LCM and HBGF is inconclusive. However, the cut-off method used in the generation of constraints can filter out most of invaluable information to prevent misguiding in the clustering process.

As for the percentage of constraints, we find that semi-supervised clustering shows superior performance at $p = 2\%$ under the weighted integration, and $p = 1\%$ under the union form. Specifically, wSSNCut wins 6 out of 12 datasets at $p = 2\%$, while only wining 3 at both $p = 0.5\%$ and $p = 1\%$. This is because the number of constraints is relatively small. Only around 300 constraints, for instance, are created for the Trace dataset at $p = 0.5\%$, while this number is almost tripled at $p = 2\%$.

In addition, we can clearly see that the weighted scheme outperforms the union method. For example, on TwoLeadECG dataset, NMI by the weighted form at $p = 2\%$ is 0.966 8, while the corresponding NMI score by the union form is only 0.900 4. It demonstrates that by summing up all constraint sets generated by different measures, we are able to obtain the valuable confidence constraint sets. Each element of the sets indicates not only whether a constraint exists in the related pair of time series or not, but also how reliable this particular constraint is.

4.4.3 Accuracy of Constraint Sets

As we know, different measures quantify the distances between time series from different aspects. Among the constraints generated by a measure, some may be the same as those by other measures, while the others are totally new. As such, constraints generated by a measure could be validated by the degree of their overlapping with those by other measures. On the other hand, the accuracy of constraints can be measured by the true labels of data points. If two instances connected by a must-link constraint have different true labels, we believe this must-link is incorrect, and an accurate constraint, otherwise. For an accurate cannot-link constraint, two instances should have different true labels. To show the accuracy of constraints, we choose two typical datasets of CBF and MedicalImages, and plot the results in Fig.2 and Fig.3. The reason for selecting these two datasets is that the performance of our approach is greatly improved on CBF and shows a relatively small improvement on MedicalImages. Fig.2 and Fig.3 show the constraint by adding constraints generated from the relevant measures step by step. The X-axis denotes the different distance measures. The Y-axes of Figs.2(a), 2(c), 3(a) and 3(c) are the total number of accurate constraints, and the Y-axes of Figs.2(b), 2(d), 3(b) and 3(d) are the percentage of accurate constraints.

From these two figures, we can easily see that the number of accurate constraints increases clearly with that of distance measures combined. This fact validates our original idea of integrating different measures to avoid mismatching. Especially, the percentage of accurate must-links in Figs.2(a) and 2(b) is close to 100%. Furthermore, the number of accurate must-links keeps increasing. Another observation is that although there are almost 30% inaccurate cannot-link constraints on CBF, our method divides data into groups almost correctly. This indicates the good robustness of wSSNCut.

For the MedicalImages dataset, we can see that the relatively large percentage of inaccurate constraints is contained in both must-links and cannot-links. But the weighted scheme shows better performance than the union form all the time when p varies from 0.5% to 2%. This explains the efficiency of weighed constraint sets in another way. The almost completely accurate positive constraints on the CBF dataset also lead to the fact that our method archives 0.959 1 NMI score at $p = 1\%$.

For all the datasets at $p = 1\%$, the accuracies of constraints are improved generally by deleting 1-vote constraints. In particular, the mean accuracy is boosted from 74.8% to 84.5% for must-links, and from 82.0% to 83.5% for cannot-links. For instance, the accuracy

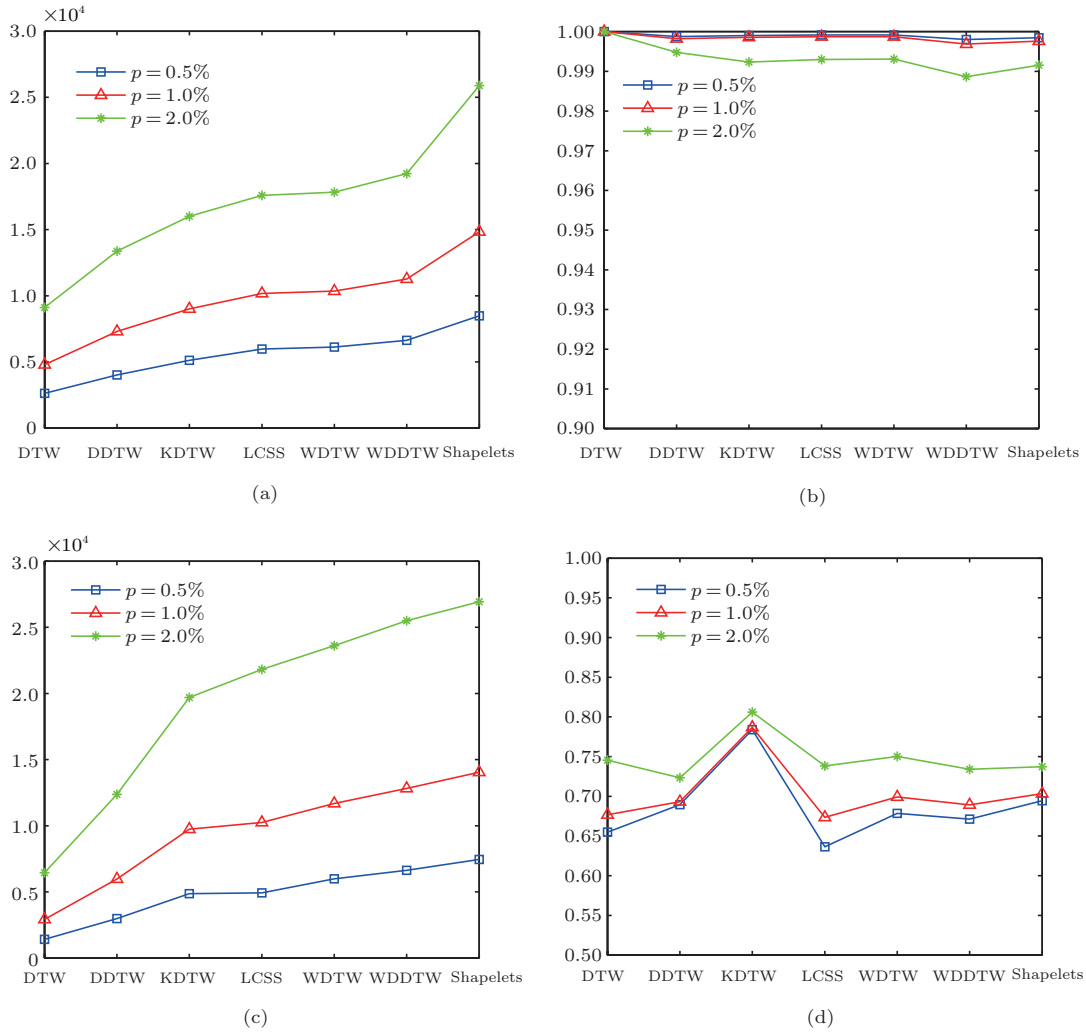


Fig.2. (a) Number of accurate must-links on CBF. (b) Percentage of accurate must-links on CBF. (c) Number of accurate cannot-links on CBF. (d) Percentage of accurate cannot-links on CBF.

is improved from 68.1% to 85.5% for must-links, and from 96.6% to 98.2% for cannot-links on the 50Words dataset; the accuracy is increased from 62.2% to 78.6% for must-links, and from 97.2% to 98.6% for cannot-links on the Lighting7 dataset.

5 Conclusions

A number of distance measures on time series have been proposed and evaluated on clustering datasets of time series in the past decade. A good measure may, however, rely on a specific domain of time series data. This fact raises a question of how to choose an appropriate measure for a given time series data where labeled data are unavailable. After finding no winner of a single measure from the experiments, we presented a method that is able to integrate multiple distance measures by using the semi-supervised clustering. Extensively experimental evaluations on various datasets have shown that our method outperforms not only individual measures, but also the typical integration approaches for clustering. Furthermore, we demonstrated the efficiency of confidence constraint sets weighted by multiple distance measures from different perspectives. In addition, our method has the good robustness in that it is able to tolerate the high level of inaccurate constraints.

As performance is improved greatly under the large percentage of accurate constraints, how to develop a method that distinguishes incorrect constraints and improves the clustering performance further becomes our future work.

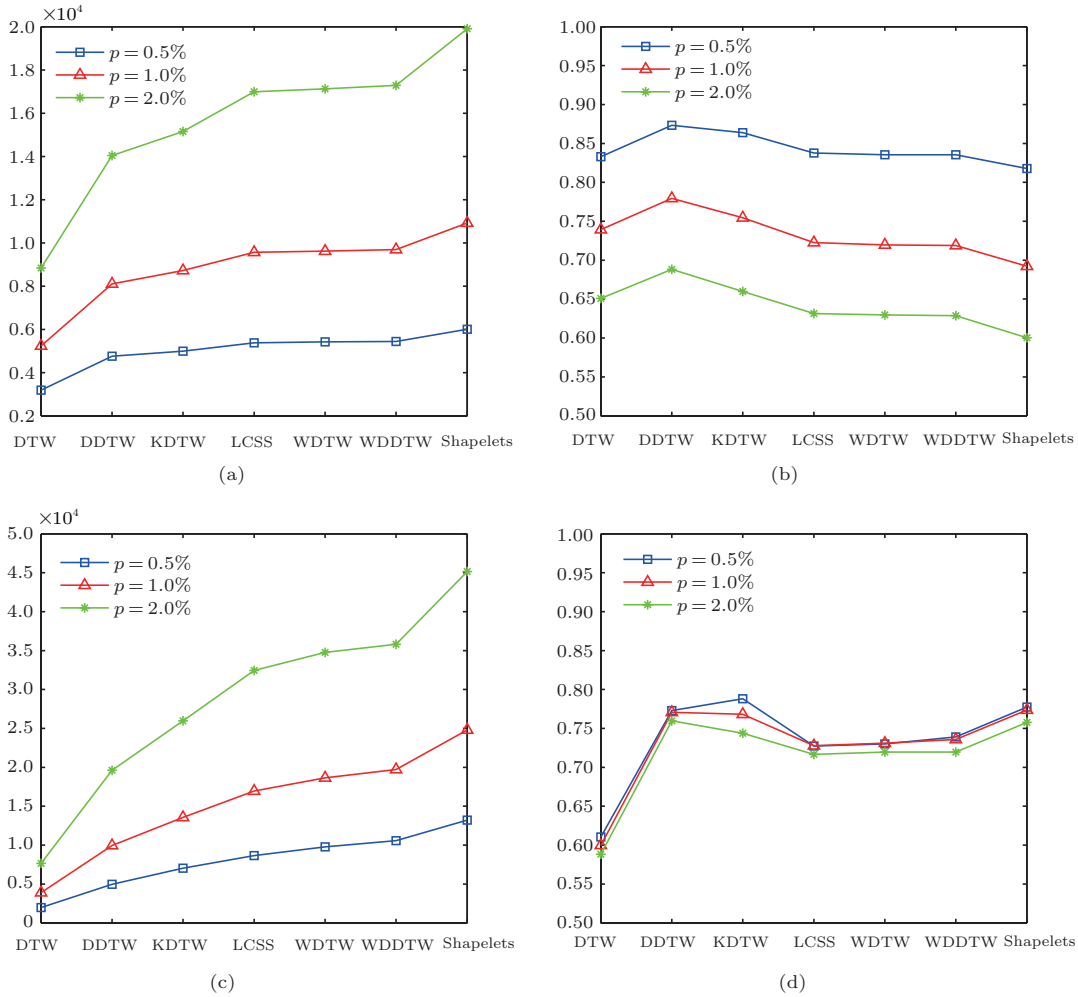


Fig.3. (a) Number of accurate must-links on MedicalImages. (b) Percentage of accurate must-links on MedicalImages. (c) Number of accurate cannot-links on MedicalImages. (d) Percentage of accurate cannot-links on MedicalImages.

References

[1] Hirano S, Tsumoto S. Cluster analysis of time-series medical data based on the trajectory representation and multiscale comparison techniques. In *Proc. the 6th International Conference on Data Mining*, December 2006, pp.896-901.

[2] Ruiz E J, Hristidis V, Castillo C, Gionis A, Jaimes A. Correlating financial time series with micro-blogging activity. In *Proc. the 5th ACM International Conference on Web Search and Data Mining*, February 2012, pp.513-522.

[3] Tan S C, San L J P. Time series clustering: A superior alternative for market basket analysis. In *Proc. the 1st International Conference on Advanced Data and Information Engineering*, January 2013, pp.241-248.

[4] Mackas D L, Greve W, Edwards M *et al.* Changing zooplankton seasonality in a changing ocean: Comparing time series of zooplankton phenology. *Progress in Oceanography*, 2012, 97/98/99/100: 31-62.

[5] Lai C P, Chung P C, Tseng V S. A novel two-level clustering method for time series data analysis. *Expert Systems with Applications*, 2010, 37(9): 6319-6326.

[6] Wang X, Smith K, Hyndman R. Characteristic-based clustering for time series data. *Data Mining and Knowledge Discovery*, 2006, 13(3): 335-364.

[7] Zhang X, Liu J, Du Y, Lv T. A novel clustering method on time series data. *Expert Systems with Applications*, 2011, 38(9): 11891-11900.

[8] Zakaria J, Mueen A, Keogh E J. Clustering time series using unsupervised-shapelets. In *Proc. the 12th IEEE International Conference on Data Mining*, December 2012, pp.785-794.

[9] Bagnall A, Janacek G. Clustering time series with clipped data. *Machine Learning*, 2005, 58(2/3): 151-178.

[10] Ding H, Trajcevski G, Scheuermann P, Wang X, Keogh E J. Querying and mining of time series data: Experimental comparison of representations and distance measures. *Proc. the VLDB Endowment*, 2008, 1(2): 1542-1552.

[11] Vlachos M, Hadjieleftheriou M, Gunopulos D, Keogh E J. Indexing multi-dimensional time-series with support for multiple distance measures. In *Proc. the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, August 2003, pp.216-225.

[12] Ye L, Keogh E J. Time series shapelets: A new primitive for data mining. In *Proc. the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, June 28-July 1, 2009, pp.947-956.

[13] Keogh E J, Pazzani M J. Derivative dynamic time warping. In *Proc. the 1st SIAM International Conference on Data Mining*, April 2001, pp.1:1-1:11.

[14] Jeong Y S, Jeong M K, Omitaomu O A. Weighted dynamic time warping for time series classification. *Pattern Recognition*, 2011, 44(9): 2231-2240.

[15] Marteau P F, Gibet S. On recursive edit distance kernels with application to time series classification. *IEEE Transactions on Neural Networks and Learning Systems*, 2015, 26(6): 1121-1133.

[16] Marteau P F. Time warp edit distance with stiffness adjustment for time series matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2009, 31(2): 306-318.

[17] Shao J, Huang Z, Shen H T, Shen J, Zhou X. Distribution-based similarity measures for multi-dimensional point set retrieval applications. In *Proc. the 16th ACM International Conference on Multimedia*, October 2008, pp.429-438.

[18] Sun Y, Li J, Liu J, Sun B, Chow C. An improvement of symbolic aggregate approximation distance measure for time series. *Neurocomputing*, 2014, 138: 189-198.

[19] Qi J, Zhang R, Ramamohanarao K, Wang H, Wen Z, Wu D. Indexable online time series segmentation with error bound guarantee. *World Wide Web*, 2015, 18(2): 359-401.

[20] Lin J, Vlachos M, Keogh E J, Gunopulos D. Iterative incremental clustering of time series. In *Proc. the 9th International Conference on Extending Database Technology*, March 2004, pp.106-122.

[21] Hautamaki V, Nykanen P, Franti P. Time-series clustering by approximate prototypes. In *Proc. the 19th International Conference Pattern Recognition*, December 2008.

[22] Oates T, Firoiu L, Cohen P R. Clustering time series with hidden Markov models and dynamic time warping. In *Proc. the IJCAI-99 Workshop on Neural, Symbolic and Reinforcement Learning Methods for Sequence Learning*, August 1999, pp.17-21.

[23] Ghassempour S, Girosi F, Maeder A. Clustering multivariate time series using hidden Markov models. *International Journal of Environmental Research and Public Health*, 2014, 11(3): 2741-2763.

[24] Izakian H, Pedrycz W, Jamal I. Fuzzy clustering of time series data using dynamic time warping distance. *Engineering Applications of Artificial Intelligence*, 2015, 39: 235-244.

[25] Ramoni M, Sebastiani P, Cohen P. Bayesian clustering by dynamics. *Machine Learning*, 2002, 47(1): 91-121.

[26] Yang Y, Chen K. Temporal data clustering via weighted clustering ensemble with different representations. *IEEE Transactions on Knowledge and Data Engineering*, 2011, 23(2): 307-320.

[27] Yang Y, Chen K. Time series clustering via RPCL network ensemble with different representations. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 2011, 41(2): 190-199.

[28] Lines J, Bagnall A. Ensembles of elastic distance measures for time series classification. In *Proc. the 14th SIAM International Conference on Data Mining*, April 2014, pp.524-532.

[29] Kulis B, Basu S, Dhillon I, Mooney R. Semi-supervised graph clustering: A kernel approach. *Machine Learning*, 2009, 74(1): 1-22.

[30] Huang X, Cheng H, Yang J, Yu J X, Fei H, Huan J. Semi-supervised clustering of graph objects: A subgraph mining approach. In *Proc. the 17th International Conference on Database Systems for Advanced Applications — Volume Part I*, April 2012, pp.197-212.

[31] Chen Y, Rege M, Dong M, Hua J. Non-negative matrix factorization for semi-supervised data clustering. *Knowledge and Information Systems*, 2008, 17(3): 355-379.

[32] Shiga M, Mamitsuka H. Efficient semi-supervised learning on locally informative multiple graphs. *Pattern Recognition*, 2012, 45(3): 1035-1049.

[33] Sakoe H, Chiba S. A dynamic programming approach to continuous speech recognition. In *Proc. the 7th International Congress on Acoustics*, August 1971, pp.65-69.

[34] Sakoe H, Chiba S. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 1978, 26(1): 43-49.

[35] Shi J, Malik J. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000, 22(8): 888-905.

[36] Zhu S, Zeng J, Mamitsuka H. Enhancing MEDLINE document clustering by incorporating MeSH semantic similarity. *Bioinformatics*, 2009, 25(15): 1944-1951.

[37] Fern X Z, Brodley C E. Solving cluster ensemble problems by bipartite graph partitioning. In *Proc. the 21st International Conference on Machine Learning*, July 2004, Article No. 36.

[38] Ghaemi R, Sulaiman M N, Ibrahim H, Mustapha N. A survey: Clustering ensembles techniques. *World Academy of Science, Engineering and Technology*, 2009, 3(2): 477-486.

[39] Huang X, Zheng X, Yuan W, Wang F, Zhu S. Enhanced clustering of biomedical documents using ensemble non-negative matrix factorization. *Information Sciences*, 2011, 181(11): 2293-2302.

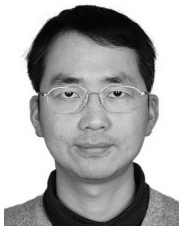
[40] Gu J, Feng W, Zeng J, Mamitsuka H, Zhu S. Efficient semisupervised MEDLINE document clustering with MeSH-semantic and global-content constraints. *IEEE Transactions on Cybernetics*, 2013, 43(4): 1265-1276.

[41] Ji X, Xu W. Document clustering with prior knowledge. In *Proc. the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, August 2006, pp.405-412.

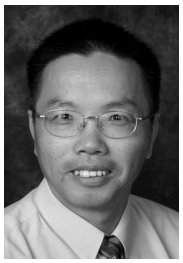
[42] Ghosh J. Scalable clustering. In *Handbook of Data Mining*, Ye N (ed.), CRC Press, 2003, pp.247-277.

[43] Strehl A, Ghosh J. Cluster ensembles — A knowledge reuse framework for combining multiple partitions. *The Journal of Machine Learning Research*, 2003, 3: 583-617.

Jing Zhou received his B.S. degree in computer science from Donghua University, Shanghai, in 2012. He is currently a graduate student of the Shanghai Key Laboratory of Intelligent Information Processing, Fudan University, Shanghai. His current research interests include time series analysis, data mining, and bioinformatics.



Shan-Feng Zhu received his B.S. and M.S. degrees in computer science from Wuhan University, Wuhan, in 1996 and 1999, respectively, and his Ph.D. degree in computer science from the City University of Hong Kong in 2003. He is currently an associate professor with School of Computer Science and Shanghai Key Laboratory of Intelligent Information Processing, Fudan University, Shanghai. Before joining Fudan University in July 2008, he was a postdoctoral fellow at Kyoto University, Japan. His research focuses on developing and applying machine learning, data mining and algorithmic methods for information retrieval, algorithmic trading, and bioinformatics. He is a member of CCF and ACM.



Xiaodi Huang is a senior lecturer in the School of Computing and Mathematics at Charles Sturt University, Australia. He received his Ph.D. degree in computer science in 2004. His research areas include visualization, data mining, and Web services. He has published over 100 scholar papers in international journals and conferences. Dr. Huang is a regular reviewer for several international journals, and serves as the committee member of international conferences. He is a member of ACM and IEEE Computer Society.

Yanchun Zhang obtained his Ph.D. degree in computer science from The University of Queensland in 1991. He has been a professor and the director of Centre for Applied Informatics at Victoria University since 2004. He is one of the National "Thousand Talents Program" Professors in China (since 2010), and currently with Fudan University. His research interests include databases, data mining, Web services and e-health. He has published over 260 research papers in international journals and conference proceedings including top journals such as ACM Transactions on Computer and Human Interaction (TOCHI), IEEE Transactions on Knowledge and Data Engineering (TKDE), and a dozen of books and journal special issues in the related areas. Dr. Zhang is a founding editor and editor-in-chief of World Wide Web Journal (Springer) and Health Information Science and Systems Journal (BioMed Central), and also the founding editor of Web Information Systems Engineering Book Series and Health Information Science Book Series. He is Chairman of International Web Information Systems Engineering Society (WISE). He was a member of Australian Research Council's College of Experts (2008∼2010), and serves as expert panel member at various funding agencies such as the Royal Society of New Zealand Marsden Fund, the National Natural Science Foundation of China (NSFC), and so on.