Deep informative representation learning for time series clustering

Rui Ye, Qun Dai*

College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China

Abstract. As an important component of unsupervised learning, time series clustering has been widely used in many research fields. It aims to draw inferences from data without class information. The specific properties of time series, such as temporal scales, dimensionality, etc., inevitably pose great challenges to the clustering task. To address this problem, in this paper, a deep informative representation learning (DIRL) method is proposed with the purpose of capturing effective cluster-specific features. Due to the considerable performances of convolutional neural network (CNN), the encoding layer, as a key component of DIRL, is constructed based on CNN. Since the results of this layer is crucial to the downstream tasks, four auxiliary parts, respectively named future steps prediction part, sequence order part, paired classification part, and Jensen-Shannon (JS) divergence part, are proposed to enhance the ability of this encoding layer. These four parts cooperate and complement with each other toward the goal of obtaining important cluster-specific features. Effectiveness of the proposed method is demonstrated by experiments conducted on sufficient time series datasets.

Keywords: Time Series Clustering; Deep Learning; Convolutional Neural Network (CNN).

1 Introduction

As an important branch of dynamic data analysis, time series clustering has attracted broad

* Corresponding author: Tel: +86-25-84593038; Fax: +86-25-84892848; Email: daiqun@nuaa.edu.cn (Q. Dai).

interest and attentions in many fields. It aims to draw inferences from time series without known labels, and has been an active research area in scientific and engineering disciplines, encompassing anomaly detection [1], financial trading [2], signal processing [3, 4] and genome analysis [5], etc.

As a commonly used technique of unsupervised learning, clustering approach has maintained good monument of development in static data analysis. However, obstacles still exist for extending those traditional clustering techniques to dynamic time series data analysis. Causes of this phenomenon can be briefly summarized as follows. Firstly, most traditional approaches on the clustering task are closely related to the formulation of a meaningful distance measure, which is the cornerstone to classifying similar data into the same group and separating data with few resemblances. However, distinction between the closest and the farthest neighbor gradually shrinks with the increase of dimension [6, 7]. Hence, the performance of traditional clustering algorithms is normally restricted to high data dimensionality. Secondly, considerable variabilities may occur among time series from different domains, making the corresponding characteristics and temporal scales exhibit obvious differences [8]. These two aspects leave a gap in efficiently employing standard clustering techniques for time series.

Aimed at the characters of time series, many creative clustering algorithms have been proposed in recent years. Most of these algorithms are feature-based, and can be subdivided into shapelet-based methods and other feature-based methods. The former one emphasizes the exploration of salient subsequence features. For example, in [9], a new form of unsupervised shapelet (u-shapelet) learned from unlabeled time series was proposed and applied to time series clustering. It only utilizes some local patterns and deliberately neglects the rest of the

data. In [10], a scalable u-shapelet algorithm was proposed. It only considers relevant subsequences of time series. However, to find effective shapelets, many existing unsupervised shapelet selection models need to exhaustively traverse the candidate segments of the original time series. Other feature-based methods hammer at extracting informative and low-dimensional feature representation of the time series. Thus, the design of an effective dimensionality reduction procedure usually plays an important role in this kind of methods. In [11], Yang et al. proposed a novel feature selection approach, which jointly takes the discriminative representations and the feature correlations into consideration. In [12], a robust spectral learning framework was created for unsupervised feature selection. In [13], a fully unsupervised algorithm called Deep Temporal Clustering (DTC) was proposed. It jointly reduces the data dimensionality and optimizes the clustering objective. Though such feature-based approaches own robust performance and redundancy removal ability, they still have some limitations. For example, pseudo-labels learned by local learning are utilized in many of these methods, which may predispose the model to select linear features. In DTC, the encoder ability profoundly affects the performance of the model, since the predicted and target distributions of KL divergence objective are obtained based on the learned representation. This makes it difficult to redress the negative effects if large deviation occurs in the representation.

To alleviate the limitations of the techniques mentioned above, in this paper, we propose a novel deep informative representation learning (DIRL) method for time series clustering. It thinks through the specific properties of time series which can further facilitate the clustering performance. This proposed method mainly consists of the following part: the encoding layer part, the future steps prediction part, the sequence order part, the paired classification part and

the Jensen-Shannon (JS) divergence part.

Among these parts, as a key component, the encoding layer is designed to explore informative representation by casting the time series data to a latent space. Since the result of this layer is represented as the input of downstream tasks, capability of this layer is of great influence. The better the encoder layer is trained, the more important representation can be achieved. Hence, to further enhance the ability of this layer, it is learned with the help of the latter four auxiliary objectives. Motivated by the broad applications and great success of convolutional neural network (CNN) [14-16], CNN layers are adopted to implement the encoding layer. Profit from the specific layered structure, CNN can commonly help recognize the meaningful characteristics, which allows the CNN-based encoding layer to explore and process informative features efficiently.

To help the learning of encoding layer, the above mentioned four parts with different objectives are incorporated. Next, we respectively give descriptions to them.

The future steps prediction part is designed with support of predictive coding [17, 18], which is successfully used for unsupervised learning. Compared with the traditional generative techniques, future steps prediction assigns attentions to the contextual information rather than concentrate on reconstructing details of the original time series data. When predicting closer steps, the local smoothing of a series can be achieved. The shared information between future steps and current context indeed gradually dwindles with the increase of time span, which may urge the model to explore more global features [19]. Thus, this part is constructed with the purpose of capturing effective features of the series. As an important property of temporal data, sequence order occupies a significant place in time series, which can reflect a degree of inherent

characteristics of a series. On this basis, the sequence order part is designed to explore the potential vital features by considering the temporal sequence of the data.

As mentioned above, the last two parts give priority to discovering informative representation of a series self, while relatively leave out the discriminative features which stimulate the distinction between different series. To complement this, the paired classification part and JS divergence part are incorporated. Among these two parts, to guarantee the discrimination, the former one explores specific features beneficial to the discrimination of different series by hypothesizing that different series come from different clusters. While the latter part takes the commonness of series belonging to the same cluster into consideration. Fig. 1 shows the block diagram of the DIRL. From this figure, we can know that in our work, the encoding layer is composed of three 1D convolutional layers and a global averaging pooling layer. Fig. 1(a) presents the entire framework of the proposed DIRL. Fig. 1(b)-Fig. 1(e) respectively show the structures of the four auxiliary tasks.

The main contributions of this work can be summarized as follows:

Firstly, in allusion to the problem of time series clustering, we propose a novel deep informative representation learning method. It aims to explore effective cluster-specific features. As a crucial component of DIRL, inspired by the wide-spread use of convolutional neural network, the encoding layer is constructed based on CNN. Four auxiliary tasks are designed to help the learning of this encoding layer, enabling the learned representation better reflect the cluster structure.

Secondly, instead of dwelling on reconstructing details of the series, future steps prediction part puts emphasis on contextual information, aiming to explore more effective local and global

features.

Thirdly, as a significant attribute of temporal data, sequence order plays an important role in time series. It can, to some extent, reveal some vital and inherent characteristics of a series. This property is considered in the sequence order part to help grasp the potential representative features.

Fourthly, to explore more distinguishable features of different series, the paired classification part and JS divergence part are incorporated. The former one focuses on extracting specific representation facilitating to the distinction of different series. The latter takes the commonness of series coming from the same cluster into consideration. All these parts collaborate and complement with each other toward the goal of learning informative and cluster-specific representation.

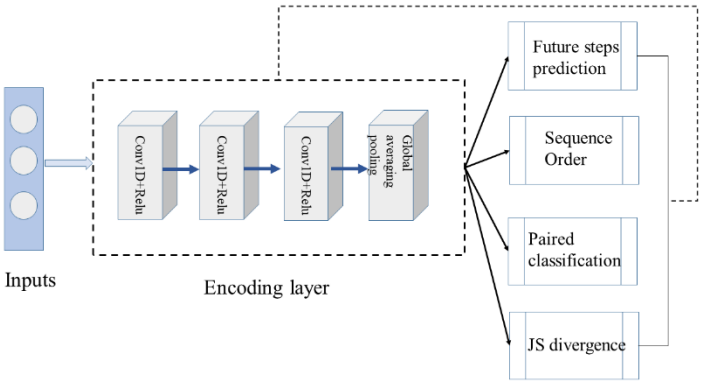

Fig. 1(a) overview of the whole procedure of DIRL

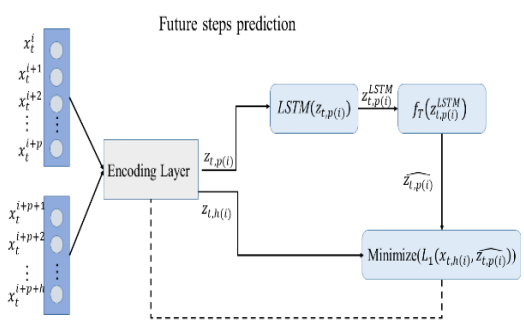

Fig. 1(b) Future steps prediction part

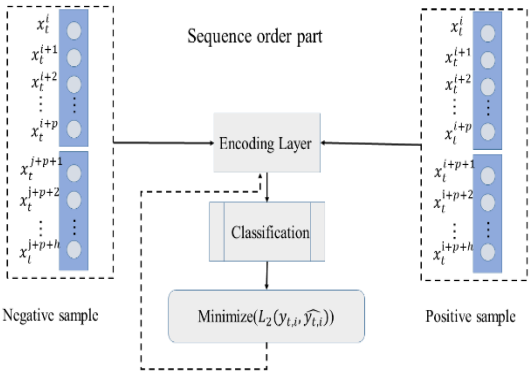

Fig. 1(c) Sequence order part

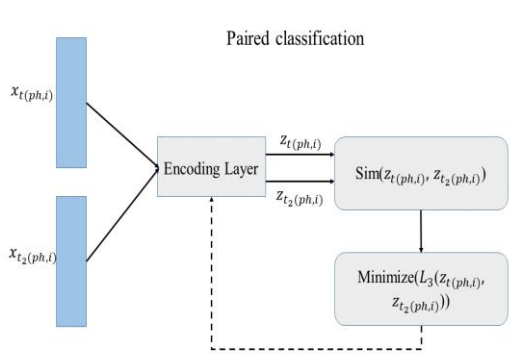

Fig. 1(d) Paired classification part

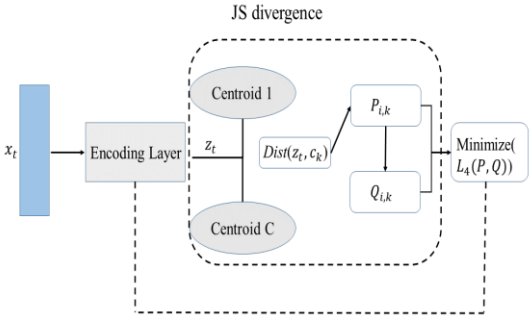

Fig. 1(e) JS divergence part

Fig. 1 Block diagram of the proposed DIRL. Fig. 1(a) shows the overview of the whole procedure of DIRL, and the other four figures respectively present the four parts of DIRL, i.e., Fig. 1(b) presents the structure of Future steps prediction part, Fig. 1(c) shows the structure of Sequence order part. Similarly, Fig. 1(d) displays the structure of Paired classification part and Fig. 1(e) presents the structure of JS divergence part.

2 Related work

Time series clustering has been widely applied to many practical fields [20, 21]. Directing at this issue, a series of valuable methods have been proposed recently. These techniques can be roughly categorized as: shape-based methods and feature-based methods. The shape-based ones concentrate on measuring distances between series and view the design of similarity evaluations as a significant point. While the feature-based ones expect to learn low-dimensional and representative representation of original series.

2.1 Shape-based methods

In time series clustering, shape-based methods work toward a suitable similarity measure between sequences. In [22], a K-Spectral Centroid (KSC) clustering algorithm was proposed. The similarity method designed in this technique is invariant to scaling and shifting. In [23],

authors put forward a novel method called DTW Barycenter Averaging (DBA). It realizes the temporal alignment by making use of K-means and dynamic time warping. The alignment skill is finally utilized for clustering. In [24], Paparrizos et al. presented a k-shape algorithm for time series clustering, where the shapes of time series are considered by the cross-correlation measure. Though the shape-based techniques exhibit effective performance in time series clustering, there are challenges as they are prone to be disturbed by outliers and noise. To alleviate this problem, feature-based methods are proposed by researchers.

2.2 Feature-based methods

Compared to shape-based methods, featured-based algorithms concentrate on capturing informative representation with low dimensionality of the time series. How to design an effective dimensionality reduction is a research priority in this kind of techniques. In [25], to obtain discriminative features, Li et al. executed the spectral clustering and the feature selection jointly. In [26], to extract effective features, authors mapped the original data into a low dimensional space by using independent component analysis. In [27], a seminal representation learning method called Deep Temporal Clustering Representation (DTCR) was designed, which incorporates the K-means objective and the auxiliary classification objective to help explore important features of the series. In broad terms, the shapelet-based techniques can also be deemed as feature-based methods. In [28], a creative model, incorporating shapelet learning, spectral technique and pseudo-label method, was constructed to learn representative shapelets of a series without known labels.

3 Proposed method

In this paper, a novel deep informative representation learning (DIRL) method is proposed for time series clustering. It aims to capture cluster-specific representation to improve the distinguish degree between different clusters. A detailed description of the architecture of our method is presented in this section.

Consider a set of N unlabeled time series $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N\}$, where $\boldsymbol{x_i} = (x_i^1, x_i^2, \dots, x_i^D)$ consists of D sequential values, our method is designed to capture distinguished representation of each series to better cluster the N unlabeled sequences into K groups. As shown in fig.1, DIRL can be roughly divided into four parts: the future steps prediction part, the paired classification part, the sequence order part and the JS divergence part. The first three parts share the same input, denoted as $\left[\boldsymbol{x}_{t,p(i)}, \boldsymbol{x}_{t,h(i)}\right], t = 1, 2, \dots, N$, where $\boldsymbol{x}_{t,p(i)} = \left(x_t^i, x_t^{i+1}, \dots, x_t^{i+p}\right)$, $\boldsymbol{x}_{t,h(i)} = \left(x_t^{i+p+1}, x_t^{i+p+2}, \dots, x_t^{i+p+h}\right)$, $i = 1,2, \dots, D-p-h$. The input of the fourth part is $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N\}$. To capture informative representation, the inputs of the four parts are then fed to an encoding layer, which can also compress the time series data into a more compact embedding space. Then the totally unsupervised clustering problem can be turned into formulating four auxiliary supervised tasks. We respectively detail all the parts in the upcoming chapter.

3.1 Encoding layer

As a crucial part of DIRL, the encoding layer casts the original input $\boldsymbol{x}$ into a latent space $\boldsymbol{z} = f_L(\boldsymbol{x})$, which is used as the representation for downstream tasks. Hence, the major purpose of this layer is to learn representation appropriate for the clustering task. To achieve this objective, as shown in Fig. 1, the encoding layer is utilized. To capture pivotal short-term

features, three 1D convolution layers with the Rectified Linear Unit (Relu) as activation function are contained. Following these 1D convolution layers, a global averaging pooling layer is designed to further reduce the number of parameters. The Encoding layer allows to remain most of the relevant information and map the input sequences into a smaller latent space, in which subsequent processing is potentially easier to operate. Training the encoding layer is driven by alternatively minimizing the loss functions of the four parts, which are explained in detail in the next several sections.

3.2 Future steps prediction part

Motivated by the wide-spread use of predictive coding in unsupervised learning field, for example, in Word2Vec tasks [29, 30], word representations are learned by forecasting context words, future steps prediction is utilized in our method. The fruitfulness of predicting next steps is partly based on the hypothesis that the future steps and the context used for predicting share similar high-level latent information. Exploration of the local smoothing of a series can be achieved by predicting closer future steps. With the increase of time span, the shared information between future steps and current context is expected to taper off, this prompts the model to explore more global features. Compared to traditional generative techniques which take much effort on reconstructing details of the series, future steps prediction allocates attentions to contextual information and aims to explore more local and global potential characteristics. Hence, in this work, we attempt to capture more informative representation by making use of future steps prediction.

As mentioned above, suppose there are N unlabeled time series $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N\}$, each component is denoted as $\boldsymbol{x}_i = (x_i^1, x_i^2, \dots, x_i^D)$. Input of the future steps prediction task is

represented as $[\boldsymbol{x}_{t,p(i)}, \boldsymbol{x}_{t,h(i)}], t = 1,2,\dots,N, i = 1,2,\dots,D-p-h$, where $\boldsymbol{x}_{t,p(i)} = (x_t^i, x_t^{i+1}, \dots, x_t^{i+p})$, $\boldsymbol{x}_{t,h(i)} = (x_t^{i+p+1}, x_t^{i+p+2}, \dots, x_t^{i+p+h})$, $i = 1,2,\dots,D-p-h$, p is the window size, h is the amount of future steps, *i* is the start position of $\boldsymbol{x}_{t,p(i)}, \boldsymbol{x}_{t,h(i)}$. To explore the latent information, we first cast the original input $\boldsymbol{x}_{t,p(i)}, \boldsymbol{x}_{t,h(i)}$ into a compact embedding space by feeding $\boldsymbol{x}_{t,p(i)}, \boldsymbol{x}_{t,h(i)}$ to the encoding layer, whose output is denoted as $\boldsymbol{z}_{t,p(i)} = f_L(\boldsymbol{x}_{t,p(i)})$, $\boldsymbol{z}_{t,h(i)} = f_L(\boldsymbol{x}_{t,h(i)})$. To get contextual information, $\boldsymbol{z}_{t,p(i)}$ is then fed to a LSTM layer. Output of this layer is denoted as $\boldsymbol{z}_{t,p(i)}^{LSTM} = LSTM(\boldsymbol{z}_{t,p(i)})$. Next a linear transformation $f_T(\boldsymbol{x}) = W_T\boldsymbol{x}$ is applied to $\boldsymbol{z}_{t,p(i)}^{LSTM}$ for the prediction of future steps. The model is trained with the purpose of maximizing the agreement between $\widehat{\boldsymbol{z}_{t,p(i)}} = f_T(\boldsymbol{z}_{t,p(i)}^{LSTM})$ and $\boldsymbol{z}_{t,h(i)}$.Here we use the following expression as the loss function.

$$L_1 = -\frac{1}{N\cdot(D-p-h)}\sum_{t=1}^{N}\sum_{i=1}^{D-p-h}\log P_{sim}(\widehat{\boldsymbol{z}_{t,p(i)}}, \boldsymbol{z}_{t,h(i)}) \tag{1}$$

where P_{sim} is the probability of $\widehat{\boldsymbol{z}_{t,p(i)}}$ similar to $\boldsymbol{z}_{t,h(i)}$.

$$P_{sim}(\widehat{\boldsymbol{z}_{t,p(i)}}, \boldsymbol{z}_{t,h(i)}) = \frac{\exp(sim(\widehat{\boldsymbol{z}_{t,p(i)}}, \boldsymbol{z}_{t,h(i)}))}{\sum_{k=1}^{D-p-h} I(k\neq i)\exp(sim(\widehat{\boldsymbol{z}_{t,p(i)}}, \boldsymbol{z}_{t,h(k)}))} \tag{2}$$

$$I(g) = \begin{cases} 1 & if(g = True) \\ 0 & if(g = False) \end{cases} \tag{3}$$

In Eq. (2), *Sim* is defined to measure the similarity between two compared vectors. Its mathematical formulation is defined as follows:

$$sim(\widehat{\boldsymbol{z}_{t,p(i)}}, \boldsymbol{z}_{t,h(i)}) = \frac{\widehat{\boldsymbol{z}_{t,p(i)}}^T\cdot\boldsymbol{z}_{t,h(i)}}{\|\widehat{\boldsymbol{z}_{t,p(i)}}\|\cdot\|\boldsymbol{z}_{t,h(i)}\|} \tag{4}$$

3.3 Sequence order part

As a significant property, temporal characteristic occupies an important position in the resolution of time series. It can the certain degree reflect some inherent features of each series. The last future steps prediction part mainly considers the consistency between the contextual information and the information of future steps, yet it relatively weakens the exploration of the

sequence order of a series. To complement this, in this part, we take the sequence order into consideration.

Given input $\boldsymbol{x}_{t,p(i)}$ and $\boldsymbol{x}_{t,h(i)}$, $[\boldsymbol{x}_{t,p(i)}, \boldsymbol{x}_{t,h(i)}]$ is a positive sample with correct sequence order, while other formulations, such as $[\boldsymbol{x}_{t,h(i)}, \boldsymbol{x}_{t,p(i)}]$, $[\boldsymbol{x}_{t,p(i)}, \boldsymbol{x}_{t,h(j)}]$ (the correct match of $\boldsymbol{x}_{t,h(j)}$ is $\boldsymbol{x}_{t,p(j)}$), etc., are negative ones. We generate these negative samples by randomly shuffling the sequence of $x_{t,p(i)}$ and $\boldsymbol{x}_{t,h(j)}$. For each $\boldsymbol{x}_{t,p(i)}$, a corresponding negative sample is generated. Thus the negative and positive samples take equal share of the total. For simplicity, in sequence order part, each sample is represented as $[\boldsymbol{x}_{t(i)}]$ =$[\boldsymbol{x}_{t,p(i)}, \boldsymbol{x}_{t,h(j)}]$. If the combination of $\boldsymbol{x}_{t,p(i)}$ and $\boldsymbol{x}_{t,h(j)}$ is in the right order, the sample is subsumed into the positive class, otherwise not. Thus this sequence order part can be reducible to a problem of binary classification. The model is trained to detect whether a given sample is positive or negative. The loss function can be formally defined as follows:

$$L_2 = -\frac{1}{2N\cdot(D-p-h)}\sum_{t=1}^{N}\sum_{i=1}^{D-p-h}\sum_{m=1}^{2} I(y_{t(i)}^{m}=1)\log\frac{\exp(\widehat{y_{t(i)}^{m}})}{\sum_{m=1}^{2}\exp(\widehat{y_{t(i)}^{m}})} \tag{5}$$

where y is the real label of each sample, it is a two dimensional one-hot vector demonstrating whether the sample is positive or not. $y_{t(i)}^{m}$ is the real label of sample $[\boldsymbol{x}_{t(i)}]$, while $\widehat{y_{t(i)}^{m}}$ is the corresponding classification result. For simplicity, the classification result is calculated through a fully connected layer.

$$\widehat{y_{t(i)}^{m}} = \boldsymbol{W}_M(f_L([\boldsymbol{x}_{t(i)}])) \tag{6}$$

where $\boldsymbol{W}_M$ is the parameter of the fully connected layer. In this part, by considering the sequence order, we aim to extract more informative features which can reflect the sequential characteristic of each time series.

3.4 Paired classification part

As introduced in section 3.2 and 3.3, future steps prediction part and sequence order part are conductive to the exploration of useful information of the original sequences. However, they mainly pay attention to the specific representation of each series self, while ignoring to extract the features with high differentiation which can help distinguish different series. Hence only utilizing the features obtained by future steps prediction and sequence order part may not necessarily satisfy the clustering task. To better capture the information beneficial to the differentiation of different time series, an auxiliary paired classification task is designed.

Since in this classification part, we put emphasis on extracting representative features beneficial to the differentiation of different series, prediction and sequence relations between $\boldsymbol{x}_{t,p(i)}$ and $\boldsymbol{x}_{t,h(i)}$ can temporarily be ignored. Hence, we first concatenate $\boldsymbol{x}_{t,p(i)}$ and $\boldsymbol{x}_{t,h(i)}$ to form a whole, denoted as $\boldsymbol{x}_{t(phi)} = concatenate(\boldsymbol{x}_{t,p(i)}, \boldsymbol{x}_{t,h(i)})$, then $\boldsymbol{x}_{t(phi)}$ is fed to the encoding layer with the output $\boldsymbol{z}_{t(phi)} = f_L(\boldsymbol{x}_{t(phi)})$. For each $\boldsymbol{x}_{t(phi)}$, we randomly select another $\boldsymbol{x}_{t_2(phj)}$ from all the samples to form a pair. If $\boldsymbol{x}_{t(phi)}$ and $\boldsymbol{x}_{t_2(phj)}$ arise from the same series, they are subsumed under the same category, otherwise not. This can be reductive to a problem of maximizing the similarity probability of two samples coming from the same series. The loss function form has some resemblances with that of future steps prediction part. Specifically, it can be expressed as follows:

$$L_3 = -\frac{1}{N\cdot(D-p-h)}\sum_{t=1}^{N}\sum_{i=1}^{D-p-h}\log\frac{\exp(sim(\boldsymbol{z}_{t(phi)},\boldsymbol{z}_{t_2(phi)}))}{\sum_{k=1}^{N}\sum_{j=1}^{D-p-h}I(not(k,t))\exp(sim(\boldsymbol{z}_{t(phi)},\boldsymbol{z}_{k(phj)}))} \quad (7)$$

where function *sim* is the cosine similarity, with the same formulation as Eq. (4). It aims to evaluate the distance between the two vectors. If $\boldsymbol{z}_{t(phi)}$ and $\boldsymbol{z}_{k(phi)}$ come from different series, $I\big(not(k,t)\big) = 1$, else $I\big(not(k,t)\big) = 0$. At first glance, the formulation of Eq. (7) is similar to that of Eq. (1), there are indeed some significant differences between them. In Eq.

(1), the inputs of *sim* are the predicted values $\widehat{\boldsymbol{z}_{t,p(i)}}$ based on contextual information, and the true latent values $\boldsymbol{z}_{t,h(i)}$ of the future steps. It is designed to minimize the prediction loss. Based on this loss function, valid information of each series is captured. However, it puts emphasis on the features of the series itself, while ignoring the representation which can help distinguish different series. This problem is, to some extent, remedied by the paired classification part. From Eq. (7), we find that it mainly focuses on differentiating samples from different series. This can, to a certain extent, help explore the distinguishable features.

3.5 JS divergence part

As mentioned in section 3.4, representations learned by future steps prediction part and sequence order part can preserve the informative features of original series. These representations emphasize the characteristics of each series itself, yet relatively pass over the distinguishable features of different series. Paired classification part's accession to some extent alleviates this problem. It focuses on exploring specific features beneficial to the differentiation of different series. However, in paired classification part, each series is deemed as a separate class. Though distinguishable features can be captured based on this part, commonness of series belonging to the same cluster is not considered. To enable the learned representation better reflect the cluster structure, inspired by the idea proposed in DTC [13], JS divergence is incorporated.

Specifically, the original input of this part is $\boldsymbol{x}_i, i = 1,2, \dots, N$. Then it is fed to the encoding layer with the output $\boldsymbol{z}_i = f_L(\boldsymbol{x}_i)$. Guided by the general steps of clustering methods, C centroids $\{\boldsymbol{c}_k, k = 1,2, \dots, C\}$ are first generated on the latent representation $\{\boldsymbol{z}_i, i = 1,2, \dots N\}$. Then according to the probability P_{ik}, i.e. the probability of $\boldsymbol{z}_i$ belonging to $\boldsymbol{c}_k$,

each representation $\boldsymbol{z}_i$ is assigned to the closest cluster with the highest P_{ik}. The centroids, during the training phase, are updated by the network. Eq. (8) shows the mathematical formulation of probability P_{ik}.

$$P_{ik} = \frac{(1+dist(\boldsymbol{z}_i,\boldsymbol{c}_k))^{-1}}{\sum_{j=1}^{C}(1+dist(\boldsymbol{z}_i,\boldsymbol{c}_k))^{-1}} \tag{8}$$

where *dist* is the method used to measure the distance between $\boldsymbol{z}_i$ and $\boldsymbol{c}_k$, C is the number of centroids. Here we define *dist* based on correlation coefficient similarity, which is defined as follows:

$$dist(\boldsymbol{z},\boldsymbol{c}) = 1 - \rho_{\boldsymbol{z},\boldsymbol{c}} \tag{9}$$

$$\rho_{\boldsymbol{z},\boldsymbol{c}} = \frac{Cov(\boldsymbol{z},\boldsymbol{c})}{\sigma_{\boldsymbol{z}}\sigma_{\boldsymbol{c}}} \tag{10}$$

where Cov represents covariance, σ represents standard deviation. For each P_{ik}, we need to construct the corresponding target distribution $Q_{i,k}$. Minimizing the JS divergence between these two distributions can potentially be a precaution to prevent representation distortions. Therefore, how to design the target distribution is of great significance. Referring to DTC, $Q_{i,k}$ is designed by the following expression.

$$Q_{i,k} = \frac{\frac{P_{ik}^2}{\sum_i^N P_{ik}}}{\sum_{k=1}^{C}\frac{P_{ik}^2}{\sum_i^N P_{ik}}} \tag{11}$$

Since JS divergence is symmetrical and can evaluate the similarity between two distributions, loss function is defined upon JS divergence.

$$L_4 = \frac{1}{2}KL\left(\boldsymbol{P}\left\|\frac{\boldsymbol{P}+\boldsymbol{Q}}{2}\right.\right) + \frac{1}{2}KL\left(\boldsymbol{Q}\left\|\frac{\boldsymbol{P}+\boldsymbol{Q}}{2}\right.\right) \tag{12}$$

$$KL\left(\boldsymbol{P}\left\|\frac{\boldsymbol{P}+\boldsymbol{Q}}{2}\right.\right) = \sum_{i=1}^{N}\sum_{k=1}^{C} P_{ik}\log\frac{P_{ik}}{(Q_{ik}+P_{ik})/2} \tag{13}$$

where N is the sample number.

Finally, the overall loss function is defined as follows:

$$L = \lambda_1 L_1 + \lambda_2 L_2 + \lambda_3 L_3 + \lambda_4 L_4 \tag{14}$$

where L_1 aims to extract local and global features of a series based on future steps prediction, L_2 emphasizes the sequence order characteristic of time series. Both of them mainly focus attention on each series self. Complements are offered by L_3 and L_4. L_3 devotes to exploring the distinguishable representation facilitating to separate out different series, while L_4 is expected to explore the similarity between series in the same cluster. After training, the learned representation is then transmitted to K-means to achieve the final cluster result.

4 Experiments

In this work, we analyze the performance of our DIRL algorithm on the 52 well-known UCR Time series Classification Archive datasets [31]. Each dataset is, by default, divided into two parts: training dataset and test dataset. Referring to the protocol in USSL [28], we train the model on the training dataset and evaluate its performance on the test dataset.

As mentioned in Section 3.2, the encoding layer is composed of convolutional layers and a global averaging pooling layer. We fix the number of convolution layers to 3. Filter number and filter size are respectively selected from [64, 128, 256] and [1, 2, 3]. In future steps prediction part, sequence order part and paired classification part, values of window size p and number of future steps h are respectively chosen from [10, 20, 30, 40, 50] and [10, 20, 30]. In the expression of overall loss function, we fix $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = 1$.

To better analyze the performance of our proposed method, several state-of-the-art benchmarks, including K-means [32], Unsupervised Discriminative Feature Selection (UDFS) [11], Nonnegative Discriminative Feature Selection (NDFS) [25], Robust Unsupervised Feature Selection (RUFS) [33], Robust Spectral learning framework for unsupervised Feature Selection (RSFS) [12], K-Spectral Centroid (KSC) [22], DTW Barycenter Averaging (DBA) [23], K-

shape [24], Deep Embedded Clustering (DEC) [34], Improved Deep Embedded Clustering (IDEC) [35], Deep Temporal Clustering (DTC) [13], Unsupervised Salient Subsequence Learning (USSL) [28], Deep Temporal Clustering Representation (DTCR) [27], are adopted as the comparative methods. For K-means, we implement it based on the wrapper KMeans function in sklearn package [36]. And it acts on the entire time series. For other baseline methods, we realize them by using the codes provided by authors on github. Corresponding parameters of each method are tuned according to the descriptions in the originals.

4.1 Evaluation metrics

As two commonly used metrics for clustering performance evaluation, Rand Index (RI) [37] and Normalized Mutual Information (NMI) [38] are utilized in our paper to measure the performance of our method. Mathematical formulations of these two evaluation methods are expressed as follows:

$$RI = \frac{TP+TN}{TP+FP+FN+TN} \tag{15}$$

Supposing there are N' data and $\frac{N'(N'-1)}{2}$ data pairs, *TP* represents the number of series pairs which actually belong to the same class and are assigned to the same cluster. *TN* represents the number of series pairs which come from different classes and are assigned to different clusters. *FP* represents the number of series pairs belonging to different classes but assigned to the same cluster. Analogously, *FN* represents the number of series pairs coming from the same class but assigned to different clusters.

$$NMI = \frac{MI(\boldsymbol{\Omega},\boldsymbol{G})}{\sqrt{H(\boldsymbol{\Omega})H(\boldsymbol{G})}} \tag{16}$$

$$MI(\boldsymbol{\Omega},\mathbf{G}) = \sum_{i=1}^{C}\sum_{j=1}^{C}\frac{|\boldsymbol{\Omega}_i \cap \boldsymbol{G}_j|}{N}\log\frac{N\cdot|\boldsymbol{\Omega}_i \cap \boldsymbol{G}_j|}{|\boldsymbol{\Omega}_i||\mathbf{G}_j|} \tag{17}$$

$$H(\boldsymbol{\Omega}) = -\sum_{i=1}^{C}\frac{|\boldsymbol{\Omega}_i|}{N}\log\frac{|\boldsymbol{\Omega}_i|}{N} \qquad H(\boldsymbol{G}) = -\sum_{j=1}^{C}\frac{|\mathbf{G}_j|}{N}\log\frac{|\mathbf{G}_j|}{N} \tag{18}$$

where *MI* represents the mutual information. *H* is the entropy, *C* is the number of clusters, *N* is the number of series. $|\boldsymbol{\Omega}_i|$ and $|\boldsymbol{G}_j|$ respectively represent the number of series belonging to cluster $\boldsymbol{\Omega}_i$ and class $\boldsymbol{G}_j$. $|\boldsymbol{\Omega}_i \cap \boldsymbol{G}_j|$ represents the number of series in the intersection of $\boldsymbol{\Omega}_i$ and $\boldsymbol{G}_j$.

4.2 Experimental results

As introduced in 4.1, RI and NMI are adopted in this paper to help compare the performance of the proposed DIRL method and the selected benchmark methods. Table 1 and Table 2 respectively show the results of NMI and RI of different methods on all the 52 UCR datasets.

Table 1 NMI values of different methods on the UCR datasets

Dataset	Kmeans	UDFS	NDFS	RUFS	RSFS	KSC	KDBA	Kshape	DEC	IDEC	DTC	USSL	DTCR	DIRL
ArrowHead	0.4816	0.5240	0.4997	0.5975	0.5104	0.5240	0.4816	0.5240	0.3100	0.2949	0.5000	**0.6322**	0.5513	0.4112
Beef	0.2925	0.2718	0.3647	0.3799	0.3597	0.3828	0.3340	0.3338	0.2463	0.2463	0.2751	0.3338	0.5473	**0.5601**
BME	0.3866	0.1642	0.1547	0.1669	0.5424	0.0169	0.1637	0.3866	0.1843	0.1672	0.1773	0.3639	**0.5424**	0.4797
BeetleFly	0.0073	0.0371	0.1264	0.1919	0.2795	0.2215	0.2783	0.3456	0.0308	0.0082	0.3456	0.5310	**0.7610**	**0.7610**
BirdChicken	0.0371	0.0371	0.3988	0.1187	0.3002	0.3988	0.2167	0.3456	0.0160	0.0082	0.0073	0.6190	0.5310	**1.0000**
Car	0.2540	0.2319	0.2361	0.2511	0.2920	0.2719	0.2691	0.3771	0.2766	0.2972	0.1892	0.4650	0.5021	**0.5632**
ChlorineConcentration	0.0129	0.0138	0.0075	0.0254	0.0159	0.0147	0.0164	0.0000	0.0009	0.0008	0.0013	0.0133	0.0195	**0.0463**
Coffee	0.5246	0.6945	**1.0000**	0.2513	**1.0000**	**1.0000**	0.0778	**1.0000**	0.0120	0.1431	0.5523	**1.0000**	0.6277	**1.0000**
Computers	0.0369	0.0310	0.0118	0.0118	0.0555	0.0008	0.0059	0.0369	0.0661	0.0318	0.0367	0.0192	0.0123	**0.1346**
DiatomSizeReduction	0.9300	0.9300	0.9300	0.8734	0.8761	**1.0000**	0.9300	**1.0000**	0.8030	0.5140	0.6863	**1.0000**	0.9418	0.8543
Dist.phal.outl.agegroup	0.1880	0.3262	0.1943	0.2762	0.3548	0.3331	0.4261	0.2911	0.4405	0.4400	0.3406	0.3846	0.4553	**0.4577**
Dist.phal.outl.correct	0.0278	0.0473	0.0567	0.1071	0.0782	0.0261	0.0199	0.0527	0.0011	0.0150	0.0115	0.1026	0.1180	**0.2112**
Earthquakes	0.0171	0.0406	0.0021	0.0084	0.0701	0.0031	0.0073	0.0171	0.0031	0.0117	0.0609	0.0361	0.0727	**0.1135**
ECG200	0.1403	0.1854	0.1403	0.2668	0.2918	0.1403	0.1886	0.3682	0.1885	0.2225	0.0918	0.3776	0.3691	**0.3927**
FaceFour	0.6632	0.4095	0.5068	0.3708	0.4191	0.0632	0.4543	0.6633	0.3630	0.3439	0.3876	0.4605	0.2618	**0.7967**
Fish	0.2357	0.1942	0.4376	0.2687	0.2864	0.1050	0.3850	0.2357	0.3630	0.2015	0.3876	0.2198	0.2236	**0.5210**
GunPoint	0.0126	0.0220	0.0334	0.2405	0.0152	0.0126	0.1288	0.3653	0.0020	0.0031	0.0194	**0.4878**	0.4200	0.3672
GunPointAgeSpan	0.2250	0.2002	0.2992	0.0041	0.0519	0.0102	0.3039	0.2250	0.4428	0.3201	0.3332	0.2194	0.3548	**0.4525**
Gunpoint.Male.Ver.Female	0.1779	0.1683	0.007	0.1247	0.3354	0.0308	0.0035	0.1779	0.0893	0.0305	0.1498	0.6538	0.5832	**0.7562**
Gunpoint.Old.Ver.Young	0.1003	0.1867	0.2577	0.1120	0.3511	0.0866	0.2494	0.1003	0.0489	0.0368	0.0961	0.2942	0.2898	**0.3599**
Ham	0.0093	0.0389	0.0595	0.0980	0.0256	0.0595	0.0265	0.0517	0.1508	0.1285	0.1016	**0.3411**	0.0989	0.1291
Herring	0.0013	0.0253	0.0225	0.0518	0.0236	0.0027	0.0000	0.0027	0.0306	0.0207	0.0143	0.1718	0.2248	**0.2279**
Lightning2	0.0038	0.0047	0.0851	0.1426	0.0326	0.1979	0.0850	0.2670	0.0600	0.1248	0.1435	**0.3727**	0.2289	0.3185

Lightning7	0.4389	0.3448	0.5078	0.4316	0.4912	0.1491	0.4611	0.4389	0.4826	0.4615	0.4695	0.4549	0.5554	**0.5582**
Meat	0.2510	0.2832	0.2416	0.1943	0.3016	0.2846	0.3661	0.2254	0.5176	0.2250	0.2250	0.9085	**0.9653**	0.9394
Mid.phal.outl.agegroup	0.0219	0.1105	0.0416	0.1595	0.0968	0.1061	0.1148	0.0722	0.2686	0.2199	0.1390	0.2780	**0.4661**	0.1196
Mid.phal.outl.correct	0.0024	0.0713	0.0150	0.0443	0.0321	0.0053	0.0760	0.0349	0.1005	0.0083	0.0079	**0.2503**	0.1150	0.1256
Mid.phal.TW	0.4134	0.4267	0.4149	0.5366	0.4219	0.4486	0.4497	0.5229	0.4509	0.3444	0.1156	**0.9202**	0.5503	0.4899
MoteStrain	0.0051	0.1187	0.1919	0.1264	0.2373	0.3002	0.0970	0.2215	0.3867	0.3821	0.0094	**0.5310**	0.4094	0.3541
OSULeaf	0.0208	0.0200	0.0352	0.0246	0.0463	0.0421	0.0327	0.0126	0.2141	0.2412	0.2201	0.3353	0.2599	**0.5372**
Plane	0.8598	0.8046	0.8414	0.8675	0.8736	0.9218	0.8784	0.9642	0.8947	0.8947	0.8678	**1.0000**	0.9296	**1.0000**
Prox.phal.outl.agegroup	0.0635	0.0182	0.0830	0.0726	0.0938	0.0682	0.0377	0.0110	0.2500	0.5396	0.4153	**0.6813**	0.5581	0.5822
Prox.phal.TW	0.0082	0.0308	0.2215	0.1187	0.0809	0.1919	0.2167	0.1577	0.5864	0.3289	0.6199	**1.0000**	0.6539	0.6181
RefrigerationDevices	0.0051	0.0309	0.0092	0.0504	0.0134	0.0272	0.0053	0.0051	0.0134	0.0229	0.0069	0.0381	0.0111	**0.1768**
ScreenType	0.0108	0.0063	0.0305	0.0176	0.0192	0.0036	0.0207	0.0108	0.0271	0.0441	0.0245	0.0175	0.0345	**0.1046**
ShapeletSim	0.4281	0.0078	0.0259	0.0001	0.0014	0.0450	0.0008	0.4281	0.0355	0.0072	0.0250	0.0074	0.0034	**0.7610**
SmallKitchenAppliances	0.0271	0.0334	0.0396	0.0593	0.0667	0.0020	0.0110	0.0271	0.0101	0.0094	0.2606	0.0522	0.1320	**0.3539**
SonyAIBORobotSurface	0.6112	0.6122	0.6112	0.6278	0.6368	0.6129	0.5516	**0.7107**	0.2773	0.4451	0.2559	0.5597	0.6634	0.6575
SonyAIBORobotSurface2	0.5444	0.4803	0.5413	0.5107	0.5406	0.5619	0.5481	0.0110	0.2214	0.2327	0.4257	**0.6858**	0.6121	0.5125
SwedishLeaf	0.0168	0.0082	0.0934	0.0457	0.0269	0.0073	0.1277	0.1041	0.5569	0.5573	0.6187	**0.9186**	0.6663	0.7882
Symbols	0.7780	0.7277	0.7593	0.7174	0.8027	0.8264	0.9388	0.6366	0.7421	0.7419	0.7995	0.8821	0.8989	**0.9683**
ToeSegmentation1	0.0022	0.0089	0.2141	0.0880	0.0174	0.0202	0.2712	0.3073	0.0010	0.0010	0.0188	0.3351	0.3115	**0.3476**
ToeSegmentation2	0.0863	0.0727	0.1713	0.1713	0.1625	0.0863	0.2627	0.0863	0.0065	0.0118	0.0096	0.4308	0.3249	**0.4400**
Trace	0.5094	0.6814	0.5084	0.5369	0.5104	0.0538	0.5280	0.5094	0.5081	0.5051	0.5107	0.6072	0.7735	**0.9336**
TwoPatterms	0.4649	0.3393	0.4351	0.4678	0.4608	0.4705	0.4419	0.3949	0.0195	0.0142	0.0119	**0.4911**	0.4713	0.4318
TwoLeadECG	0.0000	0.0004	0.1353	0.1238	0.0829	0.0011	0.0103	0.0000	0.0017	0.0010	0.0036	0.5471	0.4614	**0.9544**
UMD	0.2614	0.2157	0.1871	0.2063	0.3139	0.0061	0.1793	0.2614	0.2132	0.2074	0.1795	0.3139	0.3008	**0.4375**
Wafer	0.0010	0.0010	0.0546	0.0746	0.0194	0.0010	0.0000	0.0010	0.0165	0.0188	0.0008	0.0492	0.0228	**0.2212**
Wine	0.0031	0.0045	0.0259	0.0065	0.0096	0.0094	0.0211	0.0119	0.0018	0.1708	0.0000	**0.7511**	0.2580	0.2714
WordsSynonyms	0.5435	0.4745	0.5396	**0.5623**	0.5462	0.4874	0.4527	0.4154	0.4134	0.4387	0.3498	0.4984	0.5448	0.3821
Worms	0.1325	0.0555	0.1542	0.1170	0.0897	0.1379	0.0351	0.1325	0.1170	0.0607	0.2002	0.2607	0.1314	**0.4043**
WormsTwoClass	0.0240	0.0069	0.0069	0.0098	0.0155	0.0108	0.0002	0.0240	0.0072	0.0119	0.0033	0.1054	0.0059	**0.2364**
Average NMI	0.2173	0.2073	0.2494	0.2290	0.2611	0.2075	0.2344	0.2674	0.2206	0.2069	0.2251	0.4425	0.4005	**0.4927**

Table 2 RI values of different methods on the UCR datasets

Dataset	Kmeans	UDFS	NDFS	RUFS	RSFS	KSC	KDBA	Kshape	DEC	IDEC	DTC	USSL	DTCR	DIRL
ArrowHead	0.6905	0.7254	0.7381	**0.7476**	0.7108	0.7254	0.7222	0.7254	0.5817	0.6210	0.6692	0.7159	0.6868	0.7442
Beef	0.6713	0.6759	0.7034	0.7149	0.6975	0.7057	0.6713	0.5402	0.5954	0.6276	0.6345	0.6966	0.8046	**0.8092**
BME	0.7026	0.6017	0.5827	0.5991	0.7365	0.5582	0.6081	0.7026	0.6160	0.6112	0.5370	0.6965	0.7365	**0.7744**
BeetleFly	0.4789	0.4949	0.5579	0.6053	0.6516	0.6053	0.6052	0.6053	0.4947	0.6053	0.5211	0.8105	**0.9000**	**0.9000**
BirdChicken	0.4947	0.4947	0.7316	0.5579	0.6632	0.7316	0.6053	0.6632	0.4737	0.4789	0.4947	0.8105	0.8105	**1.0000**
Car	0.6345	0.6757	0.6260	0.6667	0.6708	0.6898	0.6254	0.7028	0.6859	0.6870	0.6695	0.7345	0.7501	**0.8057**
ChlorineConcentration	0.5241	0.5282	0.5225	0.5330	0.5316	0.5256	0.5300	0.4111	0.5348	0.5350	0.5353	0.4997	**0.5357**	0.5132
Coffee	0.7460	0.8624	**1.0000**	0.5476	**1.0000**	**1.0000**	0.4851	**1.0000**	0.4921	0.5767	0.4841	**1.0000**	0.9286	**1.0000**
Computers	0.5231	0.4981	0.5062	0.5062	0.5231	0.4985	0.5018	0.5232	0.5419	0.5197	0.5232	0.5072	0.5012	**0.5783**

DiatomSizeReduction	0.9583	0.9583	0.9583	0.9333	0.9137	**1.0000**	0.9583	**1.0000**	0.9294	0.7347	0.8792	**1.0000**	0.9682	0.9402
Dist.phal.outl.agegroup	0.6171	0.6531	0.6239	0.6252	0.6539	0.6535	0.6750	0.6020	0.7785	0.7786	0.7812	0.6650	**0.7825**	0.7207
Dist.phal.outl.correct	0.5252	0.5362	0.5362	0.5252	0.5327	0.5235	0.5203	0.5252	0.5029	0.5330	0.5010	0.5962	0.6075	**0.6095**
Earthquakes	0.5007	0.5810	0.5183	0.5247	0.6204	0.5007	0.5007	0.5008	0.5007	0.5057	0.5539	0.5871	0.6277	**0.6591**
ECG200	0.6315	0.6533	0.6315	0.7018	0.6916	0.6315	0.6018	0.7018	0.6422	0.6233	0.6018	**0.7285**	0.6648	**0.7285**
FaceFour	0.8367	0.7264	0.7588	0.7382	0.7301	0.6311	0.7620	0.8367	0.7160	0.7134	0.6883	0.7210	0.6758	**0.9172**
Fish	0.7844	0.7354	0.8081	0.7707	0.6993	0.7564	0.7912	0.7844	0.7160	0.7831	0.6883	0.7500	0.7401	**0.8475**
GunPoint	0.4971	0.5029	0.5102	0.6498	0.4994	0.4971	0.5420	0.6278	0.4981	0.4974	0.5400	**0.7257**	0.6398	0.6943
GunPointAgeSpan	0.6435	0.5373	0.6335	0.5013	0.5321	0.5042	0.6368	0.6435	0.7415	0.7033	0.6575	0.5466	0.6470	**0.7735**
Gunpoint.Male.Ver.Female	0.6175	0.5165	0.4997	0.5833	0.6114	0.5120	0.5018	0.6175	0.5591	0.5202	0.5997	0.8432	0.7974	**0.9209**
Gunpoint.Old.Ver.Young	0.5652	0.5238	0.6046	0.5652	0.6230	0.5562	0.5989	0.5652	0.5283	0.5224	0.5520	0.6167	0.6136	**0.6294**
Ham	0.5025	0.5219	0.5362	0.5107	0.5127	0.5362	0.5141	0.5311	0.5963	0.4956	0.5648	**0.6393**	0.5362	0.5799
Herring	0.4965	0.5099	0.5164	0.5238	0.5151	0.4940	0.5164	0.4965	0.5099	0.5099	0.5045	0.6190	0.5759	**0.6528**
Lightning2	0.4966	0.5119	0.5373	0.5729	0.5269	0.6263	0.5119	0.6548	0.5311	0.5519	0.5770	0.6955	0.5913	**0.6995**
Lightning7	0.7823	0.7233	0.8283	0.8036	0.8078	0.7378	0.8013	0.7823	0.8120	0.8146	0.8070	0.8013	0.7964	**0.8535**
Meat	0.6595	0.6483	0.6635	0.6578	0.6657	0.6723	0.6816	0.6575	0.6475	0.6220	0.3220	0.7740	0.9763	**0.9780**
Mid.phal.outl.agegroup	0.5351	0.5269	0.5350	0.5315	0.5473	0.5364	0.5513	0.5105	0.7059	0.6800	0.5757	0.5807	**0.7982**	0.5483
Mid.phal.outl.correct	0.5000	0.5431	0.5047	0.5114	0.5149	0.5014	0.5563	0.5114	0.5423	0.5423	0.5272	**0.6635**	0.5617	0.5794
Mid.phal.TW	0.0983	0.1125	0.1919	0.7920	0.8062	0.8187	0.8046	0.6213	0.8590	0.8626	0.7115	0.7920	**0.8638**	0.8373
MoteStrain	0.4947	0.5579	0.6053	0.5579	0.6168	0.6632	0.4789	0.6053	0.7435	0.7324	0.5062	**0.8105**	0.7686	0.7249
OSULeaf	0.5615	0.5372	0.5622	0.5497	0.5665	0.5714	0.5541	0.5538	0.7484	0.7607	0.7329	0.6551	0.7739	**0.8168**
Plane	0.9081	0.8949	0.8954	0.9220	0.9314	0.9603	0.9225	0.9901	0.9447	0.9447	0.9040	**1.0000**	0.9549	**1.0000**
Prox.phal.outl.agegroup	0.5288	0.4997	0.5463	0.5780	0.5384	0.5305	0.5192	0.5617	0.4263	0.8091	0.7430	0.7939	0.8091	**0.8093**
Prox.phal.TW	0.4789	0.4947	0.6053	0.5579	0.5211	0.6053	0.5211	0.5211	0.8189	**0.9030**	0.8380	0.7282	0.9023	0.8572
RefrigerationDevices	0.5554	0.4931	0.5585	0.3561	0.5606	0.5553	0.5565	0.5554	0.5279	0.5653	0.4522	0.5628	0.5571	**0.5925**
ScreenType	0.5555	0.5464	0.5578	0.5613	0.5596	0.5340	0.5594	0.5555	0.5648	0.5641	0.5663	0.5072	0.4790	**0.5959**
ShapeletSim	0.7206	0.5002	0.5151	0.4972	0.4982	0.5272	0.4977	0.7206	0.5002	0.5022	0.4972	0.5022	0.4994	**0.9045**
SmallKitchenAppliances	0.5445	0.3599	0.4283	0.4038	0.4398	0.5459	0.4531	0.5445	0.5492	0.5585	0.5938	0.3482	0.5698	**0.6921**
SonyAIBORobotSurface	0.7721	0.7695	0.7721	0.7787	0.7928	0.7726	0.7988	0.8084	0.5732	0.6900	0.5563	0.8105	0.8769	**0.8813**
SonyAIBORobotSurface2	0.8697	0.8745	0.8865	0.8756	0.8948	**0.9039**	0.8684	0.5617	0.6514	0.6572	0.7012	0.8575	0.8534	0.8054
SwedishLeaf	0.4987	0.4923	0.5500	0.5192	0.5038	0.4923	0.5500	0.5333	0.8837	0.8893	0.8871	0.8547	0.9223	**0.9504**
Symbols	0.8810	0.8548	0.8562	0.8525	0.9060	0.8982	0.9774	0.8373	0.8841	0.8857	0.9053	0.9200	0.9168	**0.9920**
ToeSegmentation1	0.4873	0.4921	0.5873	0.5429	0.4968	0.5000	0.6143	0.6143	0.4984	0.5017	0.5077	0.6718	0.5659	**0.7094**
ToeSegmentation2	0.5257	0.5257	0.5968	0.5968	0.5826	0.5257	0.5573	0.5257	0.4991	0.4991	0.5348	0.6778	0.8286	**0.8438**
Trace	0.7503	0.7507	0.7511	0.7563	0.7531	0.6242	0.7511	0.7503	0.7503	0.7478	0.7531	0.7773	0.8925	**0.9733**
TwoPatterms	0.8529	0.8259	0.8530	0.8385	**0.8588**	0.8585	0.8446	0.8046	0.6293	0.6338	0.6251	0.8318	0.6984	0.6826
TwoLeadECG	0.5476	0.5495	0.6328	0.8246	0.5635	0.5464	0.5476	0.8246	0.5007	0.5016	0.5116	0.8628	0.7114	**0.9895**
UMD	0.6318	0.5913	0.5982	0.6251	0.6223	0.5515	0.5994	0.6318	0.6305	0.6255	0.5859	0.6223	0.6658	**0.7328**
Wafer	0.4925	0.4925	0.5263	0.5263	0.4925	0.4925	0.4925	0.4925	0.5679	0.5597	0.5324	**0.8246**	0.7338	0.5798
Wine	0.4984	0.4987	0.5123	0.5021	0.5033	0.5006	0.5064	0.5001	0.4913	0.5157	0.4906	**0.8985**	0.6271	0.6695
WordsSynonyms	0.8775	0.8697	0.8760	0.8861	0.8817	0.8727	0.8159	0.7844	0.8893	0.8947	0.8855	0.8540	**0.8984**	0.8889
Worms	0.6213	0.5283	0.6667	0.6514	0.6363	0.6305	0.6370	0.6213	0.6363	0.6432	0.6336	0.6298	0.6568	**0.7601**
WormsTwoClass	0.5126	0.4976	0.4976	0.5037	0.5078	0.5038	0.4941	0.5126	0.5037	0.5003	0.5003	0.5126	0.4976	**0.6514**

Average RI	0.6131	0.5976	0.6308	0.6301	0.6427	0.6326	0.6250	0.6434	0.6297	0.6412	0.6182	0.7179	0.7265	**0.7769**

Remark: In Table 1 and Table 2, among all the values, the best RI and NMI values are respectively marked in bold fonts.

Table 1 and Table 2 summarize the performance of different methods on the experimental datasets. From Table 1 can we find that in the 52 datasets, DIRL gets the highest NMI value for 33 times. USSL performs the second best with respect to NMI value on 16 datasets. Consistently, DIRL gets the highest average NMI value on all the datasets, and USSL is the second. By analyzing Table 2, in terms of RI value, DIRL reaches the best on 36 datasets. USSL and DTCR are the second and the third best methods. USSL gets the highest RI value for 10 times, and DTCR gets the highest RI value for 6 times. Average RI values of the methods agree approximately with the above result. DIRL achieves the best average RI value. DTCR and USSL perform the second and the third best. By analyzing the experimental results, we can roughly conclude that in most cases, the proposed DIRL owns superior performance than other counterparts.

Note that in all the comparative benchmarks, USSL and DTCR have relatively favorable performances on several datasets. However, there are still a few defections of these two methods. In USSL, pseudo-labels are utilized to direct the learning of the model. This may cause the clustering result deviating from the correct entry because no measures can be taken to limit the negative effects when mistakes occur in the pseudo-labels. While in DIRL, the future steps prediction part is able to capture the informative information of the original series, which can, to some extent, provide corrections for the mistakes. As an inherent characteristic of temporal data, sequence order has an import place in time series. It can partly reflect some specific attributes of a series. In DIRL, the sequence order is considered while DTCR takes no

consideration of this aspect.

The following figures present the comparison between the original data and the representations learned by DIRL. For ease of viewing, we use t-SNE [39] to project the high-dimension data to the plane. For space limitation, we only show the results on four datasets, i.e. ShapeletSim, SmallKitchenAppliances, TwoLeadECG and GunPointMaleVersusFemale datasets.

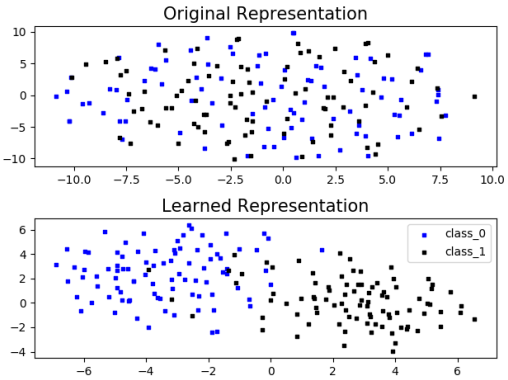

Fig. 2 original and the learned representations on ShapeletSim

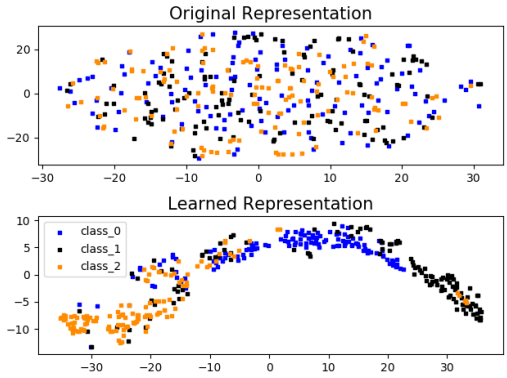

Fig. 3 original and the learned representations on SmallKitchenAppliances

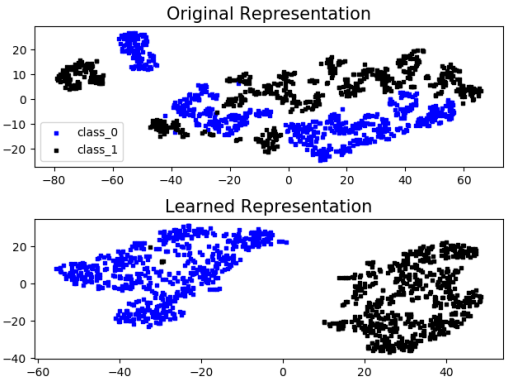

Fig. 4 original and the learned representations on TwoLeadECG

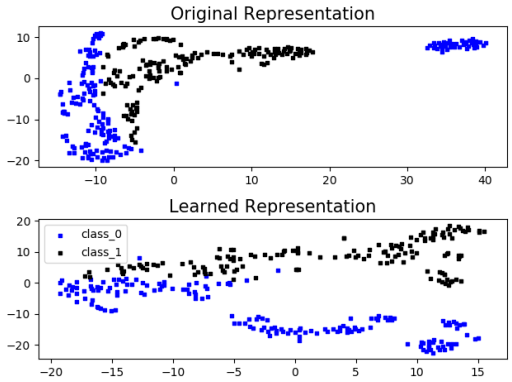

Fig. 5 original and the learned representations on GunPointMaleVersusFemale

In the figures above, data from different classes are marked in different colors. By analyzing the four figures, we can find that in the original representation, data from different classes are confused with each other, which poses great challenges to the clustering task. While

representation learned by DIRL are comparatively easier to be distinguished, which further verifies the effectiveness and the feasibility of our proposed DIRL on clustering tasks.

5 Conclusions and future works

In this paper, we propose a deep novel informative representation learning method (DIRL) for the time series clustering task. It mainly contains four parts, which interwork and interconnect with each other to guarantee the effectiveness of learning cluster-specific representations.

In the future steps prediction step, instead of expanding much efforts on reconstructing details of the series, it allocates attentions to contextual information with the purpose of exploring more local and global potential characteristics. As an important attribute, temporal characteristic holds a significant role in the time series. It can, to some extent, reveal some inherent properties of a series. Hence, in the sequence order part, the sequence order of a series is considered. Though the last two parts can properly extract valid information of a series, they lay emphasis on each series self, which may neglect some distinguishable features. To complement this, the paired classification part and the JS divergence part are incorporated. The former part focuses on the excavation of the representative features which facilitate to the distinction of different series. While the latter part takes the similarity between different series but belonging to the same cluster into consideration.

These four parts complement with each other for assisting the learning of the encoding layer, whose output is the final representation we obtained. During the training phase, post-hoc labeling and comparison with real labels are not involved, which ensures that the model is constructed without any labeled information. Experimental results also verify the effectiveness

and feasibility of the proposed DIRL in time series clustering.

However, there is also some room to improve. In this work, we mainly consider the clustering of univariate time series. But in many practical scenarios, multivariate time series are also widely involved. Hence, in the future work, how to expand our model to process multivariate series will be considered. Besides, in this work, we only consider the time dimension of time series, while ignore its spatial dimension. However, spatio-temporal data are also of great utility in many engineering tasks. Therefore, in the future work, we will further consider how to modify our model for dealing with spatio-temporal data.

Acknowledgements

This work is supported by the National Key R&D Program of China (Grant Nos. 2018YFC2001600, 2018YFC2001602), and the National Natural Science Foundation of China under Grant No. 61473150.

References

[1] P. K. Chan and M. V. Mahoney, "Modeling multiple time series for anomaly detection," in *Fifth IEEE International Conference on Data Mining (ICDM'05)*, 2005, pp. 90-97.

[2] S. Aghabozorgi, A. Seyed Shirkhorshidi, and T. Ying Wah, "Time-series clustering - A decade review," *Information Systems,* vol. 53, pp. 16-38, 2015.

[3] R. K. H. Galvao and T. Yoneyama, "A competitive wavelet network for signal clustering," *IEEE Transactions on Systems Man & Cybernetics Part B Cybernetics A Publication of the IEEE Systems Man & Cybernetics Society,* vol. 34, pp. 1282-1288, 2004.

[4] S. Güne, K. Polat, and E. Yosunkaya, "Efficient sleep stage recognition system based on EEG

signal using k-means clustering based feature weighting," *Expert Systems with Applications,* vol. 37, pp. 7922-7928, 2010.

[5] A. Fujita, P. Severino, K. Kojima, J. R. Sato, A. G. Patriota, and S. Miyano, "Functional clustering of time series gene expression data by Granger causality," *BioMed Central (BMC) Systems Biology,* vol. 6, pp. 1-12, 2012.

[6] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, "When is "nearest neighbor" meaningful?," in *International Concerence on Database Theory (ICDT'99)*, 1999, pp. 217-235.

[7] R. J. Durrant and A. Kaban, "When is 'nearest neighbour' meaningful: A converse theorem and implications," *Journal of Complexity,* vol. 25, pp. 385-397, Aug 2009.

[8] M. Shahnawaz, A. Ranjan, and M. Danish, "Temporal Data Mining: An Overview," *International Journal of Engineering and Advanced Technology,* pp. 9-18, 2011.

[9] J. Zakaria, A. Mueen, and E. Keogh, "Clustering Time Series Using Unsupervised-Shapelets," in *2012 IEEE 12th International Conference on Data Mining (ICDM)*, Belgium, 2012, pp. 785-794.

[10] L. Ulanova, N. Begum, and E. Keogh, "Scalable Clustering of Time Series with U-Shapelets," in *2015 SIAM International Conference on Data Mining (SDM 15)*, Canada, 2015, pp. 900-908.

[11] Y. Yang, H. T. Shen, Z. Ma, Z. Huang, and X. Zhou, "L21-norm regularized discriminative feature selection for unsupervised learning," in *the Twenty-Second international joint conference on Artificial Intelligence (IJCAI)*, 2011, pp. 1589-1594.

[12] L. D. Lei Shi, and Yi-Dong Shen, "Robust spectral learning for unsupervised feature selection," in *2014 IEEE International Conference on Data Mining (ICDM)*, China, 2014, pp. 977-982.

[13] N. S. Madiraju, S. M. Sadat, D. Fisher, and H. Karimabadi, "Deep Temporal Clustering : Fully

Unsupervised Learning of Time-Domain Features," *arXiv preprint arXiv:1802.01059*, 2018.

[14] C.-L. Zhang and J. Wu, "Improving CNN linear layers with power mean non-linearity," *Pattern Recognition,* vol. 89, pp. 12-21, 2018.

[15] J. Zhang and C. Zong, "Deep Neural Networks in Machine Translation: An Overview," *IEEE Intelligent Systems,* vol. 30, pp. 16-25, 2015.

[16] A. V. D. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves*, et al.*, "WaveNet: A Generative Model for Raw Audio," *arXiv:1609.03499v1,* Sep 2016.

[17] G. Hickok, "The cortical organization of speech processing: Feedback control and predictive coding the context of a dual-stream model," *Journal of Communication Disorders,* vol. 45, pp. 393-402, 2012.

[18] R. Srinivasan and K. Rao, "Predictive Coding Based on Efficient Motion Estimation," *IEEE Transactions on Communications,* vol. 33, pp. 888-896, 2003.

[19] Y. L. Aaron van den Oord, Oriol Vinyals, "Representation Learning with Contrastive Predictive Coding," *arXiv:1807.03748v2.,* 2019.

[20] A. M. Denton and B. D. H. Dorr, "Pattern-based time-series subsequence clustering using radial distribution functions," *Knowledge&Information Systems,* vol. 18, pp. 1-27*,* 2009.

[21] J. A. Vilar, A. M. Alonso, and J. M. Vilar, "Non-linear time series clustering based on non-parametric forecast densities," *Computational Statistics & Data Analysis,* vol. 54, pp. 2850-2865, 2010.

[22] J. Yang and J. Leskovec, "Patterns of Temporal Variation in Online Media," in *the Forth International Conference on Web Search and Web Data Mining (WSDM)* Hong Kong, China, 2011, pp. 177-186.

[23] F. O. Petitjean, A. Ketterlin, and P. Gancarski, "A global averaging method for dynamic time warping, with applications to clustering," *Pattern Recognition*, vol. 44, pp. 678-693, 2011.

[24] J. Paparrizos and L. Gravano, "k-Shape: Efficient and Accurate Clustering of Time Series," *Special Interest Group on Management of Data (SIGMOD) of the Association for Computing Machinery (ACM) Record,* vol. 45, pp. 69-76, 2016.

[25] Y. Y. Zechao Li, Jing Liu, Xiaofang Zhou, and Hanqing Lu., "Unsupervised feature selection using nonnegative spectral analysis.," in *the Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012, pp. 1-4.

[26] H. J. Chonghui Guo, and Na Zhang., "Time series clustering based on ica for stock data analysis.," *in 2008 4-th International Conference on Wireless Communications, Networking and Mobile Computing,* pp. 1-4, 2008.

[27] Q. Ma, J. Zheng, S. Li, and G. W. Cottrell, "Learning Representations for Time Series Clustering," in *the Thirty-third Conference on Neural Information Processing Systems (NeurIPS 2019)*, Canada, 2019, pp. 1-11.

[28] Q. Zhang, J. Wu, P. Zhang, G. Long, and C. Zhang, "Salient Subsequence Learning for Time Series Clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 41, pp. 2193-2207, 2019.

[29] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," *Computer Science,* pp. 1-12, 2013.

[30] T. Mikolov, I. Sutskever, C. Kai, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," *Advances in Neural Information Processing Systems,* vol. 26, pp. 1-9, 2013.

[31] E. Keogh, "Welcome to the UCR time-series Classification/Clustering Page," http://www.cs.ucr.edu/~eamonn/time_series_data/.

[32] J. A. Hartigan and M. A. Wong, "A K-Means Clustering Algorithm," *Applied Statistics,* vol. 28, pp. 100-108, 1979.

[33] M. Qian and C. Zhai, "Robust Unsupervised Feature Selection," in *the Twenty-Third international joint conference on Artificial Intelligence (IJCAI)*, 2013, pp. 1621-1627.

[34] R. G. Junyuan Xie, and Ali Farhadi., "Unsupervised deep embedding for clustering analysis.," in *International conference on machine learning (ICML)*, USA, 2016, pp. 478-487.

[35] X. Guo, L. Gao, X. Liu, and J. Yin, "Improved Deep Embedded Clustering with Local Structure Preservation," in *the 26-th International Joint Conference on Artificial Intelligence (IJCAI)*, Australia, 2017, pp. 1753-1759.

[36] Sklearn Package, *Available: https://pypi.org/project/sklearn/*.

[37] William, M., and Rand, "Objective Criteria for the Evaluation of Clustering Methods," *Journal of the American Statistical Association,* vol. 66, pp. 846-850, 1971.

[38] H. Zhang, T. B. Ho, Y. Zhang, and M. S. Lin, "Unsupervised Feature Extraction for Time Series Clustering Using Orthogonal Wavelet Transform," *Informatica,* vol. 30, pp. 305-319, 2006.

[39] V. D. M. Laurens and G. Hinton, "Visualizing Data using t-SNE," *Journal of Machine Learning Research,* vol. 9, pp. 2579-2605, 2008.