This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3466175

Feature Weighting-Based Deep Fuzzy C-Means for Clustering Incomplete Time Series

Yurui Li, Mingjing Du, *Member, IEEE,* Wenbin Zhang, *Member, IEEE*, Xiang Jiang and Yongquan Dong

***Abstract*—Time series clustering is a crucial unsupervised technique for analyzing data, commonly used in various fields, including medicine and stock analysis. However, in real-world scenarios, time series data inevitably contain missing values, consequently reducing the efficiency of traditional clustering methods. In incomplete time series, existing clustering methods typically adopt a two-stage strategy, i.e., initially imputing missing values followed by clustering. However, this approach of separating imputation from clustering may lead to inconsistencies in the optimization objectives and increase the complexity of parameter tuning, potentially resulting in unsatisfactory clustering results. This paper proposes an end-to-end deep fuzzy clustering model for incomplete time series (EEDFC), which jointly optimizes imputation and clustering within a unified framework by integrating multiple losses. In the imputation part, an attention mechanism is integrated to tackle challenges associated with dependencies in extended sequences. In addition, an adversarial strategy is introduced to enhance the encoder's imputation and feature representation learning capability, thus reducing the error propagation from imputation to clustering. In the clustering part, EEDFC combines a feature weighting-based fuzzy clustering (FWFC), which considers intra-cluster compactness and inter-cluster separateness. Furthermore, exponential distance is adopted, and feature and cluster weighting are also integrated into the Kullback-Leibler (KL) divergence loss to improve clustering performance. We conduct extensive experiments comparing our proposed model with eleven other methods across ten benchmark datasets. The experimental results demonstrate that our proposed model performs better than eleven comparative methods.**

***Index Terms*—Time series; fuzzy clustering; missing values imputation; deep learning; end-to-end.**

I. Introduction

TIME series data refers to a sequence of data points collected or recorded at regular time intervals. It is commonly used in various fields, such as financial markets and industrial engineering, and is characterized by temporal continuity and high dimensionality. In real-world production scenarios, various digital devices generate vast amounts of daily time series data. Due to equipment failure, harsh environments, or operational errors, time series data is inevitably missing [1], violating the assumption of data integrity in most traditional clustering techniques [2]. Incomplete data complicates the clustering process and may degrade its performance [3] [4]. Unfortunately, incomplete time series are prevalent in certain fields, and re-recording them is costly and unfeasible.

The research on time series clustering techniques has developed rapidly in recent years [5]–[8]. However, some existing work may lack consideration of objectives consistency between imputation and clustering and unclear memberships in time series when dealing with incomplete time series [9]. Therefore, clustering incomplete time series is challenging. A natural solution to the problem of clustering incomplete time series involves initially employing an imputation [10]–[12] algorithm to impute missing values, followed by the application of well-established clustering techniques to the resulting complete dataset [13]. The two-stage methods exhibit considerable system complexity and are not automated [14]. As this approach consists of two entirely independent components, it is susceptible to error accumulation, which potentially impacts its overall performance. Recently, several studies have explored end-to-end models for clustering incomplete time series, which can optimize the entire pipeline [15] [16]. In comparison to the work in this paper, these studies do not effectively address the issue of unclear membership in time series. The reason for this is that they all use a hard clustering approach, which assigns each data point to a single cluster with a fixed membership. However, time series data frequently exhibits intricate patterns with unclear memberships [17] [18]. Hard clustering partitions restrict their ability to capture the nuanced temporal dynamics inherent in time series data, resulting in suboptimal performance when confronted with complex data.

To solve the above problems, this paper proposes a novel end-to-end deep fuzzy clustering model for incomplete time series (EEDFC), which integrates both imputation and clustering work into an unified deep framework. The structure of EEDFC consists of two parts: imputation and clustering. The imputation module imputes missing values using the bidirectional gated recurrent unit (BIGRU)-based autoencoder structure incorporating an attention mechanism that prioritizes contributions from significant parts of the data. Moreover, an adversarial strategy is introduced to enhance the encoder's imputation and feature representation learning capability, thus mitigating errors from imputation to clustering tasks. The clustering module integrates an improved fuzzy c-means (FCM) [19] clustering into the autoencoder framework to generate cluster-friendly representations. We define a single objective function that jointly optimizes both imputation and clustering.

This work was supported in part by the National Natural Science Foundation of China under Grant 62006104, and Grant 61872168. *(Corresponding author: Mingjing Du.)*

Y. Li, M. Du, X. Jiang and Y. Dong are with the Jiangsu Key Laboratory of Educational Intelligent Technology, School of Computer Science and Technology, Jiangsu Normal University, Xuzhou 221116, China (e-mail: lyrchris417@gmail.com; dumj@jsnu.edu.cn; xjiang@jsnu.edu.cn; tomdyq@163.com).

W. Zhang is with the Jiangsu Key Laboratory of Educational Intelligent Technology, School of Computer Science and Technology, Jiangsu Normal University, Xuzhou 221116, China, and also with the College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China (e-mail: zwbwen@jsnu.edu.cn).

Authorized licensed use limited to: XIDIAN UNIVERSITY. Downloaded on September 24,2024 at 00:09:50 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3466175

We train our model using an alternating optimization strategy to make the generated imputation values as close to the actual value distribution as possible. This training strategy reduces errors caused by imputation and enhances the feature representation for the clustering task, thus improving the overall clustering performance.

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

- We propose a novel end-to-end deep fuzzy clustering (EEDFC) model designed for handling time series with missing values. The model combines imputation and clustering within a unified framework, implemented via a single objective function, ultimately improving accuracy.
- We develop an improved BIGRU-based autoencoder structure, which integrates attention mechanisms and an adversarial training strategy into the encoder to enhance its imputation and feature representation learning capability.
- We present the feature weighting-based fuzzy clustering (FWFC) algorithm, which adopts the exponential distance and incorporates feature and cluster weighting into the Kullback-Leibler (KL) divergence loss, thereby improving its performance and robustness to noise.

The rest of the paper is structured as follows: Sec. II mainly outlines existing time series imputation and clustering methods. The relevant fundamental concepts and definitions are introduced in Sec. III. Sec. IV presents the proposed EEDFC model. Sec. V analyzes the experimental data and experimental results. Finally, the conclusion and future work are given in Sec. VI.

II. Related Work

In this section, we briefly review the related work about time series imputation and time series clustering.

A. Time Series Imputation

Time-series imputation is a long-standing research topic. Recently, deep learning techniques have gained increased popularity for this task [20] [21]. Luo et al. [22] propose a novel method for time-series imputation based on the generative adversarial network [23] [12], which takes random noise as the input. This method requires substantial training time and appears unstable with random noise inputs. To solve this problem, Luo et al. further propose an improved version named E^2GAN [24], which adopts an autoencoder structure based on a GRU for data imputation. E^2GAN takes the original time series as its input, tackling the difficulty of training the model and the accuracy.

Cao et al. propose BRITS [11], a bidirectional LSTM-based model that imputes missing values without assuming any specific data-generating process. Zhou et al. [25] introduce a similar approach, employing a bidirectional LSTM network with an autoencoder structure for imputing missing values in time series data. However, a simple LSTM structure struggles to capture long-term dependencies in time series data. To overcome this limitation, Suo et al. propose GLIMA [26], which integrates a multi-dimensional attention mechanism to capture distant correlations as well as local and global dependencies across time series. Combined with attention, GLIMA can capture critical features in the data, overcoming the limitations of traditional neural networks. Similar to GLIMA, Du et al. propose SAITS [27], which also employs an attention mechanism [28] [29]. However, unlike some existing transformer-based models, SAITS learns missing values from a weighted combination of two diagonally-masked self-attention blocks.

B. Time Series Clustering

Time series data is typically characterized by high dimensionality and significant noise. If conventional mining methods are directly applied to the raw data, the algorithm's time and space complexity will be increased, impacting the final clustering outcomes. Deep learning-based time series clustering involves reducing the raw data to low-dimensional feature representations and clustering them with suitable algorithms [30]. Sai et al. [31] propose DTC, which uses an autoencoder to transform raw data into low-dimensional feature representations for subsequent clustering. It naturally integrates feature representation learning and time series clustering into a single, fully unsupervised end-to-end learning framework. However, DTC leads to potential instability during training since each iteration's target distribution is updated based on the predicted distribution. Furthermore, the performance of DTC depends heavily on the ability of the encoder to learn feature representations since the prediction distribution depends on the learned feature representations. In response to these limitations, Ma et al. propose DTCR [32], which introduces a dummy sample generation strategy and a classifier to improve the encoder's capabilities. DTCR incorporates the reconstruction loss and K-means objective into a sequence-to-sequence (seq2seq) model, improving the overall performance and stability of the original DTC model.

Dino et al. propose conDetSEC [33], an approach for handling variable-length multivariate time series across diverse fields. conDetSEC trains an autoencoder to achieve data embedding representation and then implements a clustering enhancement phase to assign them to their categories. De Jong et al. propose VaDER [15] to cluster multivariate time series containing missing values. However, VaDER does not impose constraints on missing values, so encoding incomplete sequences as hidden representations may compromise the quality of feature representations, leading to clustering errors. As a result, VaDER inevitably experiences the adverse effects of missing values on the overall clustering performance. Ma et al. propose CRLI [16], similar to VADER, a model designed for clustering time series with missing values. The critical distinction is that CRLI applies constraints on the imputed values it generates. However, it still performs poorly in some cases because it does not consider the long-term dependence of the time series and the correlation between the variables. The above clustering algorithms incorporate k-means in their clustering layers, which partition each data point into a specific cluster. However, a significant concern in real-world time series datasets is that the boundaries between clusters are

Authorized licensed use limited to: XIDIAN UNIVERSITY. Downloaded on September 24,2024 at 00:09:50 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3466175

often unclear. These algorithms' performance degrades when the data boundaries become fuzzy, producing unsatisfactory results.

Different from existing studies on end-to-end models for clustering incomplete time series, considering the complexity of time series and unclear memberships, our method incorporates soft clustering in the clustering layer. In addition, we employ attention mechanisms and weighting techniques to emphasize critical information in time series by highlighting its correlations with variables.

III. Fundamental Concepts

This section introduces two deep clustering models based on autoencoder, DEC [34] and IDEC [35], and reviews two traditional fuzzy clustering algorithms, FCM [19] and FCS [36]. They are the fundamental basis for our proposed deep fuzzy clustering model. The key symbols used in this article are outlined in Table I.

A. Multivariate Time Series

We denote a multivariate time series (MTS) dataset as $\boldsymbol{\chi} = \{\boldsymbol{X}_i\}_{i=1}^{n}$, where n is the total number of samples. Each $\boldsymbol{X}_i \in \mathbb{R}^{l\times d} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t, \ldots, \boldsymbol{x}_l)$ is a time series, where d is the number of variables (dimensions), and l is the sequence length. $\boldsymbol{x}_t \in \mathbb{R}^d$ is the multidimensional vector of the time series $\boldsymbol{X}_i$ at the t-th time step, and $1 \leq t \leq l$.

B. Deep Autoencoder Clustering based on Student's-t Distribution

The autoencoder is a model designed to extract valuable low-dimensional latent representations from its hidden layers. It is composed of an encoder and a decoder. The encoder compresses data into low-dimensional nonlinear features, and the decoder reconstructs the original data from the encoded features. Two primary deep clustering models, DEC [34] and IDEC [35], utilize deep autoencoder to cluster data by optimizing the autoencoder networks and cluster centers. DEC employs a t-SNE-based centroid probability distribution for cluster assignment and feature representations. To measure the similarity between feature representations and cluster centers, the cluster layer output can be expressed as follows:

$$q_{ij} = \frac{(\mathbf{1} + \left\|\boldsymbol{z}_i - \boldsymbol{c}_j\right\|^2/\delta)^{-\frac{\delta+1}{2}}}{\sum_{v=1}^{k}(\mathbf{1} + \left\|\boldsymbol{z}_i - \boldsymbol{c}_v\right\|^2/\delta)^{-\frac{\delta+1}{2}}}, s.t. \sum_{j=1}^{k} q_{ij} = 1 \quad (1)$$

where $\boldsymbol{Z} = (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_i, \ldots, \boldsymbol{z}_n)$ and $\boldsymbol{z}_i \in \mathbb{R}^{d'}$ for $i = 1, \ldots, n$. Each $\boldsymbol{z}_i$ represents the feature representations in hidden layer corresponding to input data $\boldsymbol{X}_i \in \boldsymbol{\chi}$ with a lower dimension d'. $\boldsymbol{z}_i = f(\boldsymbol{X}_i)$, where $f(\cdot)$ represents the nonlinear mapping function of the deep autoencoder from the input layer to the hidden layer. Let $\boldsymbol{Z}$ be assigned to k cluster centers $\boldsymbol{C} = (\boldsymbol{c}_1, \ldots, \boldsymbol{c}_j, \ldots, \boldsymbol{c}_k)$, where $\boldsymbol{c}_j \in \mathbb{R}^{d'}$ for $j = 1, \ldots, k$ is the center of the j-th cluster. Let q_{ij} be the degree of membership for $\boldsymbol{z}_i$ in the j-th cluster, and let δ be a hyperparameter typically set to 1.

TABLE I: MAJOR SYMBOLS USED IN THIS PAPER

Notation	Description
$\boldsymbol{\chi}$	Time-series dataset
$\boldsymbol{X}_i$	The i_{th} time series
$\boldsymbol{x}_t$	The time series at time step t
n	Number of time series samples
d	Dimension of time series
l	Maximum time series length
$\boldsymbol{\Lambda}$	Missing values indicator
$\boldsymbol{Z}$	The feature representation
$\boldsymbol{Q}$	Cluster partition matrix
$\boldsymbol{C}$	Cluster center matrix
k	Number of clusters
d'	Dimension of feature representation
$\boldsymbol{\Phi}$	Feature weight matrix
$\boldsymbol{\sigma}$	Cluster weight vector
α	Fuzzification coefficient
r	Sensitivity coefficient of feature weight
s	Sensitivity coefficient of cluster weight
η	Balance coefficient between compactness and separation
γ	Reciprocal of the sample variance
λ, λ_0, λ_1, λ_2	Weight coefficients

The Student's-t based membership q_{ij} defines the desired target p_{ij} by:

$$p_{ij} = \frac{q_{ij}^2/\sum_{i=1}^{n} q_{ij}}{\sum_{v=1}^{k}(q_{iv}^2/\sum_{i=1}^{n} q_{iv})}, s.t. \sum_{j=1}^{k} p_{ij} = 1 \quad (2)$$

Finally, as shown in Eq. (3), DEC's objective function minimizes the KL-divergence between p_{ij} and q_{ij}:

$$\mathcal{L}_{DEC} = \mathcal{L}_{KL} = KL(\boldsymbol{P}||\boldsymbol{Q}) = \sum_{i=1}^{n}\sum_{j=1}^{k} p_{ij} log\frac{p_{ij}}{q_{ij}} \quad (3)$$

IDEC extends DEC by incorporating reconstruction loss to preserve the local structure of the data. The reconstruction loss is calculated using the Mean Squared Error (MSE). The objective function can be expressed as:

$$\begin{aligned}\mathcal{L}_{IDEC} &= \lambda\mathcal{L}_{KL} + \mathcal{L}_R \\ &= \lambda\sum_{i=1}^{n}\sum_{j=1}^{k} p_{ij} log\frac{p_{ij}}{q_{ij}} + \sum_{i=1}^{n} ||\,\boldsymbol{X}_i - g(\boldsymbol{z}_i)\,||_2^2\end{aligned} \quad (4)$$

where $\mathcal{L}_{KL}$ is the clustering loss, and $\mathcal{L}_R$ is the reconstruction loss. $g(\cdot)$ denotes the nonlinear mapping from the hidden layer to output of the decoder, and the coefficient $\lambda > 0$ is used to balance $\mathcal{L}_{KL}$ and $\mathcal{L}_R$.

C. Fuzzy Clustering

Fuzzy c-means (FCM) [19] is a fundamental clustering algorithm in the field of fuzzy theory [37]–[39]. By introducing fuzzy membership, each sample is assigned to a specific cluster based on the degree to which it belongs, providing a soft and flexible method of assigning samples to clusters [40].

Authorized licensed use limited to: XIDIAN UNIVERSITY. Downloaded on September 24,2024 at 00:09:50 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3466175

The update rules for the membership u_{ij} and the cluster center $\boldsymbol{c}_j$ are given by:

$$u_{ij} = \frac{(\|\boldsymbol{z}_i - \boldsymbol{c}_j\|^2)^{\frac{1}{1-\alpha}}}{\sum_{v=1}^{k}(\|\boldsymbol{z}_i - \boldsymbol{c}_v\|^2)^{\frac{1}{1-\alpha}}} \tag{5}$$

$$\boldsymbol{c}_j = \frac{\sum_{i=1}^{n} u_{ij}^{\alpha}\boldsymbol{z}_i}{\sum_{i=1}^{n} u_{ij}^{\alpha}} \tag{6}$$

where u_{ij} denotes the membership degree of the sample $\boldsymbol{z}_i$ belonging to the cluster center $\boldsymbol{c}_j$. α is the fuzzifier in fuzzy clustering algorithms that determines the flexibility of fuzzy memberships and how fuzzy the resultant clusters are. Typically, α is set to be greater than 1. For each iteration of FCM, the fuzzy memberships and cluster centers are updated until convergence is achieved.

Moreover, Fuzzy Compactness & Separation (FCS) [36] is proposed as an algorithm to improve FCM by optimizing the fuzzy between-cluster scatter matrix and maximizing the separation measure. By incorporating both hard c-means and fuzzy c-means [41], FCS is updated as follows:

$$u_{ij} = \frac{(\|\boldsymbol{z}_i - \boldsymbol{c}_j\|^2 - \eta_j\|\boldsymbol{c}_j - \bar{\boldsymbol{z}}\|^2)^{\frac{1}{1-\alpha}}}{\sum_{v=1}^{k}(\|\boldsymbol{z}_i - \boldsymbol{c}_v\|^2 - \eta_v\|\boldsymbol{c}_v - \bar{\boldsymbol{z}}\|^2)^{\frac{1}{1-\alpha}}} \tag{7}$$

$$\boldsymbol{c}_j = \frac{\sum_{i=1}^{n} u_{ij}^{\alpha}\boldsymbol{z}_i - \eta_j\sum_{i=1}^{n} u_{ij}^{\alpha}\bar{\boldsymbol{z}}}{\sum_{i=1}^{n} u_{ij}^{\alpha} - \eta_j\sum_{i=1}^{n} u_{ij}^{\alpha}} \tag{8}$$

where η_j is a parameter, and $\bar{\boldsymbol{z}}$ represents the mean of samples.

IV. Proposed EEDFC Model

The proposed end-to-end deep fuzzy clustering (EEDFC) model for incomplete time series is shown in Fig. 1. It consists of two main parts: the missing value imputation module and the clustering module. In the missing value imputation module, an attention mechanism is integrated into the encoder structure to enhance the model's ability to generate imputed values. In addition, an adversarial strategy is introduced to enhance the encoder's feature learning capabilities and mitigate error propagation from imputation to clustering, resulting in high-quality clustering results. The encoder extracts the feature representation $\boldsymbol{Z}$ for subsequent clustering module. In the clustering module, we integrate a fuzzy clustering objective into the autoencoder network to concurrently acquire cluster-friendly representations and identify clusters. The whole training process of EEDFC is trained end-to-end, optimizing the imputation and clustering simultaneously throughout its final objective function.

A. Overview of EEDFC

Our goal is to generate imputed values close to the actual data distribution and obtain feature representations suitable for clustering. The overall objective function of the proposed method EEDFC is defined as follows:

$$\mathcal{L}_{EEDFC} = \mathcal{L}_{REC} + \lambda_0\mathcal{L}_{PRE} + \lambda_1\mathcal{L}_{ADV} + \lambda_2\mathcal{L}_{KL} \tag{9}$$

where $\mathcal{L}_{REC}$, $\mathcal{L}_{PRE}$, and $\mathcal{L}_{ADV}$ are imputation-related losses (detailed in Sec. IV-B), and $\mathcal{L}_{KL}$ is a clustering-related loss (detailed in Sec. IV-C). λ_0, λ_1, and λ_2 represent coefficients, which we set to 1, 1, and 0.1, respectively. In particular, $\mathcal{L}_{REC}$ represents the reconstructed loss, aiming to preserve the structural characteristics of the original data. $\mathcal{L}_{PRE}$ represents the predicted loss, capturing the temporal dynamics of the incomplete time series. $\mathcal{L}_{ADV}$ represents the adversarial loss, motivating the generated imputed values to match the actual data distribution as closely as possible. The imputation module, which uses the three losses mentioned above, effectively performs the imputation task while extracting representations of the original time series. However, these representations may not be suitable for the clustering task. To this end, the Kullback-Leibler divergence loss ($\mathcal{L}_{KL}$), tailored explicitly for clustering, is combined with the other three objective functions for joint training [42] [43]. This integration enables the model to learn clustering-friendly representations, enhancing clustering performance.

B. Learning Representations on Incomplete Time Series

In the context of imputing missing values of time series, a missing values indicator set, denoted as $\boldsymbol{\Lambda} = \{\boldsymbol{M}_i\}_{i=1}^{n}$, is employed to indicate whether a certain element of the time series is missing. Each indicator $\boldsymbol{M}_i = (\boldsymbol{m}_1, \ldots, \boldsymbol{m}_t, \ldots, \boldsymbol{m}_l)$, where $\boldsymbol{m}_t \in \{0,1\}^d$. If the o-th variable of $\boldsymbol{x}_t$ is missing, we set the corresponding component of $\boldsymbol{m}_t$ to 0; otherwise, $\boldsymbol{m}_t$ is set to 1.

The encoder uses a bidirectional gated recurrent unit (BIGRU) to impute missing values. BIGRU consists of two GRUs: a forward and a backward GRU. They have a similar network structure, but the inputs are in opposite directions. BIGRU outperforms GRU by incorporating information from past, present, and future time steps, leading to a better understanding of data structure and features.

We define $\overrightarrow{\hat{\boldsymbol{X}}}_i = (\overrightarrow{\hat{\boldsymbol{x}}}_1, \ldots, \overrightarrow{\hat{\boldsymbol{x}}}_t, \ldots, \overrightarrow{\hat{\boldsymbol{x}}}_l)$ as the predicted data generated by the forward GRU and $\overleftarrow{\hat{\boldsymbol{X}}}_i = (\overleftarrow{\hat{\boldsymbol{x}}}_1, \ldots, \overleftarrow{\hat{\boldsymbol{x}}}_t, \ldots, \overleftarrow{\hat{\boldsymbol{x}}}_l)$ as the predicted data generated by the backward GRU. The multidimensional vectors of the predicted sequences at each time step are represented by:

$$\overrightarrow{\hat{\boldsymbol{x}}}_t = \boldsymbol{W}_x\overrightarrow{\boldsymbol{h}}_{t-1} + \boldsymbol{b}_x \tag{10}$$

$$\overleftarrow{\hat{\boldsymbol{x}}}_t = \boldsymbol{W}_x\overleftarrow{\boldsymbol{h}}_{t-1} + \boldsymbol{b}_x \tag{11}$$

where $\overrightarrow{\hat{\boldsymbol{x}}}_t$ and $\overleftarrow{\hat{\boldsymbol{x}}}_t$ are multidimensional vectors of the forward and backward GRU predicted sequences at the t-th time step. $\boldsymbol{W}_x$ and $\boldsymbol{b}_x$ are parameters, and $\overrightarrow{\boldsymbol{h}}_{t-1}$ and $\overleftarrow{\boldsymbol{h}}_{t-1}$ are the hidden states of previous time step.

1) Predicted Loss: In prediction tasks, the predicted loss is usually calculated by comparing the predicted value with the actual value. However, in an unsupervised scenario, the actual value of the missing part is unknown. To address this, we employ an approximation strategy. Specifically, our goal is to ensure that the values of the non-missing parts generated by the model are close to their corresponding actual values. If they are highly matched, then the predicted values of the missing parts are also likely to be highly accurate. Based on this idea, we define the prediction loss as follows:

$$\mathcal{L}_{PRE} = \frac{1}{n}\sum_{i=1}^{n}\left\|(\boldsymbol{X}_i - \overrightarrow{\hat{\boldsymbol{X}}}_i)\odot\boldsymbol{M}_i\right\|_2^2 + \frac{1}{n}\sum_{i=1}^{n}\left\|(\overleftarrow{\boldsymbol{X}}_i - \overleftarrow{\hat{\boldsymbol{X}}}_i)\odot\overleftarrow{\boldsymbol{M}}_i\right\|_2^2 \tag{12}$$

Authorized licensed use limited to: XIDIAN UNIVERSITY. Downloaded on September 24,2024 at 00:09:50 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3466175

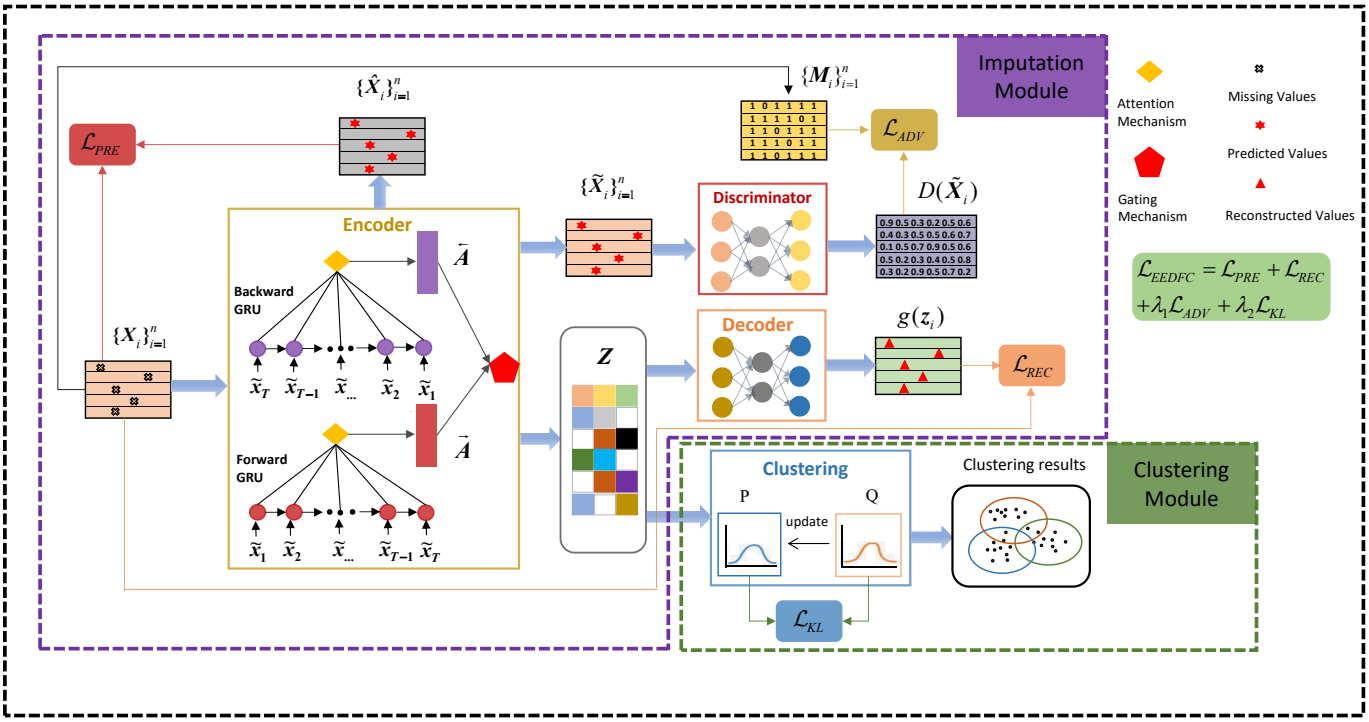


Fig. 1: The architecture of EEDFC. We take the time series data $\{\boldsymbol{X}_i\}_{i=1}^n$ as the input to the encoder, which is processed by the encoder to obtain the imputed sequence $\{\widetilde{\boldsymbol{X}}_i\}_{i=1}^n$, and the feature representation $\boldsymbol{Z}$. The role of the discriminator is to enhance the learning capability of the encoder. Fuzzy clustering is done on the extracted feature representation $\boldsymbol{Z}$ to get the clustering results. The imputation and clustering is done in a unified framework.

where n represents the number of samples, and $\overleftarrow{\boldsymbol{X}}_i = (\overleftarrow{\boldsymbol{x}}_1, \ldots, \overleftarrow{\boldsymbol{x}}_t, \ldots, \overleftarrow{\boldsymbol{x}}_l)$ (i.e., the input to the backward GRU) is the reverse of $\boldsymbol{X}_i = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t, \ldots, \boldsymbol{x}_l)$. $\boldsymbol{M}_i = (\boldsymbol{m}_1, \ldots, \boldsymbol{m}_t, \ldots, \boldsymbol{m}_l)$ and $\overleftarrow{\boldsymbol{M}}_i = (\overleftarrow{\boldsymbol{m}}_1, \ldots, \overleftarrow{\boldsymbol{m}}_t, \ldots, \overleftarrow{\boldsymbol{m}}_l)$ represent the missing value indicator corresponding to $\boldsymbol{X}_i$ and $\overleftarrow{\boldsymbol{X}}_i$.

During training, the above objective function may lead to overfitting issues. Specifically, Eq. (12) represents the training error between the true values of the non-missing portion (regarded as labels) and the corresponding generated values (considered as training data). However, owing to the unavailability of true values for the missing portion (similar to unknown labels in test data), computing the test error, representing the difference between the true values of the missing data and their generated estimates, is not feasible. Therefore, we train the model to learn data structures and patterns from the non-missing data by minimizing the training error, and apply the learned patterns to the entire time series. Nevertheless, when the training error (i.e., Eq. (12)) becomes too small, overfitting may occur. To accurately generate predicted values for missing data, we introduce a dropout mechanism to mitigate overfitting.

Let $\widetilde{\boldsymbol{\chi}} = \{\widetilde{\boldsymbol{X}}_i\}_{i=1}^n$, where each $\widetilde{\boldsymbol{X}}_i = (\widetilde{\boldsymbol{x}}_1, \ldots, \widetilde{\boldsymbol{x}}_t, \ldots, \widetilde{\boldsymbol{x}}_l)$ denote the imputed sequence. The multidimensional vectors of the imputed sequence at each time step can be computed by combining the information of the original and predicted sequences as follows:

$$\overrightarrow{\widetilde{\boldsymbol{x}}}_t = \boldsymbol{m}_t \odot \boldsymbol{x}_t + (\boldsymbol{1} - \boldsymbol{m}_t) \odot \overrightarrow{\hat{\boldsymbol{x}}}_t \tag{13}$$

$$\overleftarrow{\widetilde{\boldsymbol{x}}}_t = \overleftarrow{\boldsymbol{m}}_t \odot \overleftarrow{\boldsymbol{x}}_t + (\boldsymbol{1} - \overleftarrow{\boldsymbol{m}}_t) \odot \overleftarrow{\hat{\boldsymbol{x}}}_t \tag{14}$$

$$\widetilde{\boldsymbol{x}}_t = (\overrightarrow{\widetilde{\boldsymbol{x}}}_t + \overleftarrow{\widetilde{\boldsymbol{x}}}_t)/2 \tag{15}$$

where $\overrightarrow{\widetilde{\boldsymbol{x}}}_t$ and $\overleftarrow{\widetilde{\boldsymbol{x}}}_t$ are multidimensional vectors of the forward and backward GRU imputed sequences at the t-th time step.

According to the properties of the BIGRU model, the hidden states at the current time step are derived from the hidden states at the previous time step and the multidimensional vectors of the imputed sequences at the current time step. The updating equation is expressed as follows:

$$\overrightarrow{\boldsymbol{h}}_t = tanh(\boldsymbol{W}_h \overrightarrow{\boldsymbol{h}}_{t-1} + \boldsymbol{U}_h \overrightarrow{\widetilde{\boldsymbol{x}}}_t + \boldsymbol{b}_h) \tag{16}$$

$$\overleftarrow{\boldsymbol{h}}_t = tanh(\boldsymbol{W}_h \overleftarrow{\boldsymbol{h}}_{t-1} + \boldsymbol{U}_h \overleftarrow{\widetilde{\boldsymbol{x}}}_t + \boldsymbol{b}_h) \tag{17}$$

where $tanh(\cdot)$ is the activation function, $\boldsymbol{W}_h$, $\boldsymbol{U}_h$, and $\boldsymbol{b}_h$ are parameters, and $\overrightarrow{\boldsymbol{h}}_t$ and $\overleftarrow{\boldsymbol{h}}_t$ are the hidden states at the t-th time step.

2) Reconstructed Loss: Time series data often has specific structures that are critical for understanding its content. During autoencoder training, the model endeavors to capture these structural characteristics by minimizing the reconstruction loss (i.e., the difference between the original and reconstructed

Authorized licensed use limited to: XIDIAN UNIVERSITY. Downloaded on September 24,2024 at 00:09:50 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3466175

sequences). However, traditional reconstruction losses cannot directly be applied to time series data with missing values. To address this issue, we use masked mean squared error (masked MSE) as the reconstruction loss, where "masked" denotes that the model excludes missing values during loss computation. The formula for this loss is as follows:

$$\mathcal{L}_{REC} = \frac{1}{n}\sum_{i=1}^{n} \|(\boldsymbol{X}_i - g(\boldsymbol{z}_i)) \odot \boldsymbol{M}_i\|_2^2 \tag{18}$$

where $g(\cdot)$ denotes the nonlinear mapping from the hidden layer to the output of the decoder. $\boldsymbol{z}_i$ represents the learned feature representation, which is fed into the decoder to reconstruct the original time series. The feature representation $\boldsymbol{Z} = (\boldsymbol{z}_1, ..., \boldsymbol{z}_i, ..., \boldsymbol{z}_n)$ can be computed by Eqs. (19)-(24).

An attention mechanism is incorporated into the encoder structure to focus on information relevant to the missing data. This enhances the model's ability to generate imputed values and obtain more representative feature representations. The attention scores are calculated based on the hidden states at the t-th time step and the BIGRU outputs as follows:

$$\overrightarrow{\boldsymbol{\Delta}} = softmax(\overrightarrow{\boldsymbol{h}}_t \odot \overrightarrow{\boldsymbol{\Gamma}}) \tag{19}$$

$$\overleftarrow{\boldsymbol{\Delta}} = softmax(\overleftarrow{\boldsymbol{h}}_t \odot \overleftarrow{\boldsymbol{\Gamma}}) \tag{20}$$

where $\overrightarrow{\boldsymbol{\Delta}}$ and $\overleftarrow{\boldsymbol{\Delta}}$ represent the attention scores, indicating the learned contribution and relevance strength of the data at each time step in the sequence. $\overrightarrow{\boldsymbol{\Gamma}}$ and $\overleftarrow{\boldsymbol{\Gamma}}$ represent the forward and backward GRU outputs, respectively. The softmax activation function normalizes the weight scores, ensuring their sum equals 1.

The attention mechanism results can be calculated as follows:

$$\overrightarrow{\boldsymbol{A}} = \overrightarrow{\boldsymbol{\Delta}} \odot \overrightarrow{\boldsymbol{\Gamma}} \tag{21}$$

$$\overleftarrow{\boldsymbol{A}} = \overleftarrow{\boldsymbol{\Delta}} \odot \overleftarrow{\boldsymbol{\Gamma}} \tag{22}$$

where $\overrightarrow{\boldsymbol{A}}$ and $\overleftarrow{\boldsymbol{A}}$ represent the results of the attention mechanism for forward and backward GRU, respectively.

The gating mechanism is introduced to enhance information fusion from the attention mechanism, thereby augmenting the system's decision-making capabilities. The intermediate feature representation $\boldsymbol{Z}$ can be obtained by:

$$\boldsymbol{Z} = concat(gate(\overrightarrow{\boldsymbol{A}}) \odot \overrightarrow{\boldsymbol{A}}, gate(\overleftarrow{\boldsymbol{A}}) \odot \overleftarrow{\boldsymbol{A}}) \tag{23}$$

$$gate(\overrightarrow{\boldsymbol{A}}) = sigmoid(\boldsymbol{W}_A \overrightarrow{\boldsymbol{A}} + \boldsymbol{b}_A) \tag{24}$$

where $gate(\cdot)$ represents the gating mechanism, $\boldsymbol{W}_A$ and $\boldsymbol{b}_A$ are parameters, and $sigmoid(\cdot)$ is activation function. It is worth noting that the output of the $gate(\cdot)$ ranges in the interval $[0, 1]$, allowing it to adjust the information obtained from the attention operation. The gating mechanism can select or retain valuable features to support the task at hand, leading to more efficient and effective utilization of relevant information and improving overall task performance.

3) Adversarial Loss: We integrate a discriminator into our architecture and utilize adversarial training to enhance the generator's (encoder's) data generation capability (specifically, time series imputation capability). In our method, the generator aims to impute missing values accurately, while the discriminator tries to differentiate between the observed and imputed values in an imputed sequence. Formally, the discriminator is trained by minimizing the following objective function:

$$\mathcal{L}_{DIS} = -\frac{1}{n}\sum_{i=1}^{n}[\boldsymbol{M}_i \odot log(D(\widetilde{\boldsymbol{X}}_i)) + (\boldsymbol{1} - \boldsymbol{M}_i) \odot log(\boldsymbol{1} - D(\widetilde{\boldsymbol{X}}_i))] \tag{25}$$

where $D(\cdot)$ denotes the indicator assigned by the discriminator, which represents the probability that each element in $\widetilde{\boldsymbol{X}}_i$ is an actual observed value. We aim for the discriminator to assign a high probability $D(\widetilde{\boldsymbol{X}}_i)$ to actual observed values and a low probability to generated imputed values. In this manner, we train the discriminator to distinguish between the actual observed values and the generated imputed values by minimizing Eq. (25).

The generator is designed to generate an imputed sequence that is as realistic as possible, making it challenging for the discriminator to differentiate between actual observed values and imputed values within the sequence. Therefore, for generated imputed values, the generator wants the discriminator's output probability $D(\widetilde{\boldsymbol{X}}_i)$ close to 1, i.e., the discriminator considers the generated imputed values to be actual observed values. Therefore, the generator is trained to maximize the discriminator's misclassification rate by minimizing Eq. (26). The adversarial objective function of the model is defined as follows:

$$\mathcal{L}_{ADV} = \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{1} - \boldsymbol{M}_i) \odot log(\boldsymbol{1} - D(\widetilde{\boldsymbol{X}}_i)) \tag{26}$$

The detailed process of feature representation learning is described in Algorithm 1.

Algorithm 1: The Feature Representation Learning Algorithm.

Input: The input data $\boldsymbol{\chi} = \{\boldsymbol{X}_i\}_{i=1}^n$; The maximum iteration: $Maxepoch$.
Output: Feature representation $\boldsymbol{Z}$.

```
Initialize h;
for epoch = 1 to Maxepoch do
    Compute the imputed data χ̃ with (10)-(17);
    Feed the imputed data χ̃ into the discriminator;
    Compute the attention score with (19) and (20), the
    attention result with (21) and (22);
    Obtain the feature representation Z with (23);
    Feed Z into the decoder to reconstruct the original
    time series, and employ Z for the subsequent
    clustering stage.
end
return Feature representation Z.
```

Authorized licensed use limited to: XIDIAN UNIVERSITY. Downloaded on September 24,2024 at 00:09:50 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3466175

C. Fuzzy Clustering with KL-Divergence Loss

Traditional FCM and FCS clustering algorithms face several challenges, such as sensitivity to noise and outliers and dependence on initial cluster centers. To overcome these challenges, we propose the feature weighting-based fuzzy clustering (FWFC) algorithm incorporating localized feature weighting to enhance clustering accuracy. We also introduce a cluster weighting technique to reduce the influence of poorly initialized cluster centers [44]. Furthermore, we define a novel objective function that uses a non-Euclidean distance metric, which is robust to noise and outliers, as suggested by [45].

The proposed FWFC has the following objective function:

$$\mathcal{L}_{KL} = KL(\boldsymbol{P} \,||\, \boldsymbol{Q}) = \sum_{i=1}^{n}\sum_{j=1}^{k} p_{ij} log \frac{p_{ij}}{q_{ij}} \tag{27}$$

where n refers to the sample number, and k denotes the cluster number. $\boldsymbol{Q} \in \mathbb{R}^{n\times k}$ is a matrix, where each element q_{ij} represents the membership degree of the i-th sample to the j-th cluster. The derivation of the update strategy for q_{ij} is detailed in section A of the supplementary material. $\boldsymbol{P} \in \mathbb{R}^{n\times k}$ is also a matrix, where each element p_{ij} is the target distribution of the Student's-t based membership q_{ij}, denoted by Eq. (2).

The iterative optimization of the algorithm starts with initializing the cluster center $\boldsymbol{C}$, feature weight $\boldsymbol{\Phi}$, and cluster weight $\boldsymbol{\sigma}$.

1) Update $\boldsymbol{Q}$ by Fixing $\boldsymbol{C}$, $\boldsymbol{\Phi}$, and $\boldsymbol{\sigma}$:

$$q_{ij} = \frac{(\sigma_j^s \sum_{a=1}^{d'} \varphi_{ja}^r dist(z_{ia}, c_{ja}) - \eta_j \sigma_j^s \sum_{a=1}^{d'} \varphi_{ja}^r dist(c_{ja}, \bar{z}_a))^{\frac{1}{1-\alpha}}}{\sum_{v=1}^{k} (\sigma_v^s \sum_{a=1}^{d'} \varphi_{va}^r dist(z_{ia}, c_{va}) - \eta_v \sigma_v^s \sum_{a=1}^{d'} \varphi_{va}^r dist(c_{va}, \bar{z}_a))^{\frac{1}{1-\alpha}}} \tag{28}$$

where $\alpha > 1$ is the fuzzifier, d' is the dimension of feature representations in hidden layer, r represents the sensitivity coefficient of feature weight, and s represents the sensitivity coefficient of cluster weight. z_{ia} indicates the a-th feature in the i-th sample, $\bar{z}_a$ represents the mean of the a-th feature of the sample, and c_{ja} is the center of the j-th cluster denoted by Eq. (31). φ_{ja} refers to the weight of the a-th feature in the j-th cluster denoted by Eq. (32), and σ_j represents the weight of the j-th cluster denoted by Eq. (34).

The term $dist(\cdot,\cdot)$ indicates a exponential distance metric, and is defined as follows:

$$dist(z_{ia}, c_{ja}) = 1 - E(z_{ia}, c_{ja}) \tag{29}$$

where $E(z_{ia}, c_{ja}) = exp(-\gamma_a(\|z_{ia} - c_{ja}\|^2))$. $\|\cdot\|$ is the L_2 norm, and γ_a represents the reciprocal of the variance of the a-th feature in $\boldsymbol{Z}$.

The algorithm restricts q_{ij} to a specific range interval $[0, 1]$. However, the value of q_{ij} in the update Eq. (28) may be negative for certain data points $\boldsymbol{z}_i$, which requires imposing some restrictions. For a given data point $\boldsymbol{z}_i$, if $\sigma_j^s \sum_{a=1}^{d'} \varphi_{ja}^r dist(z_{ia}, c_{ja}) \leq \eta_j \sigma_j^s \sum_{a=1}^{d'} \varphi_{ja}^r dist(c_{ja}, \bar{z}_a)$ then q_{ij} is set to 1, and $q_{ij'}$ is set to 0 for all $j' \neq j$. That is, if the exponential-distance between the data point and the j-th cluster center is smaller than $\eta_j \sigma_j^s \sum_{a=1}^{d'} \varphi_{ja}^r dist(c_{ja}, \bar{z}_a)$, these data points will then belong exactly to the j-th cluster with membership value of 1. Each cluster in algorithm will have a crisp boundary such that all data points inside this boundary will have a crisp membership value $q_{ij} \in \{0, 1\}$ and other data points outside this boundary will have fuzzy membership value $q_{ij} \in [0, 1]$.

η_j is a parameter designed to ensure non-overlap among the k clusters, and it is defined as follows:

$$\eta_j = \frac{(\beta/4) min_{j'\neq j}(\mathbf{1} - exp(-\boldsymbol{\gamma} \,||\, \boldsymbol{c}_j - \boldsymbol{c}_{j'} \,||^2))}{max_v(\mathbf{1} - exp(-\boldsymbol{\gamma} \,||\, \boldsymbol{c}_v - \bar{\boldsymbol{z}} \,||^2))} \tag{30}$$

where $0 \leq \beta \leq 1$ is a coefficient to balance within-cluster and between-cluster distance in the cluster space. $\boldsymbol{\gamma}$ represents the reciprocal of the variance of samples, and $\bar{\boldsymbol{z}}$ represents the mean of samples.

2) Update $\boldsymbol{C}$ by Fixing $\boldsymbol{Q}$, $\boldsymbol{\Phi}$, and $\boldsymbol{\sigma}$:

$$c_{ja} = \frac{\sum_{i=1}^{n} q_{ij}^{\alpha} E(z_{ia}, c_{ja}) z_{ia} - \eta_j \sum_{i=1}^{n} q_{ij}^{\alpha} E(c_{ja}, \bar{z}_a) \bar{z}_a}{\sum_{i=1}^{n} q_{ij}^{\alpha} E(z_{ia}, c_{ja}) - \eta_j \sum_{i=1}^{n} q_{ij}^{\alpha} E(c_{ja}, \bar{z}_a)} \tag{31}$$

$\boldsymbol{C} \in \mathbb{R}^{k\times d'}$ is a matrix, where each element c_{ja} denotes the center of the j-th cluster, and q_{ij} is the fuzzy membership value calculated by Eq. (28).

3) Update $\boldsymbol{\Phi}$ by Fixing $\boldsymbol{Q}$, $\boldsymbol{C}$, and $\boldsymbol{\sigma}$:

$$\varphi_{ja} = \begin{cases} \dfrac{1}{\kappa_a} & \text{if } B\varphi_{ja} = 0 \text{ and } \kappa_a = |\{e : B\varphi_{je} = 0\}| \\ 0 & \text{if } B\varphi_{ja} \neq 0 \text{ and } \exists e\, B\varphi_{je} = 0 \\ \dfrac{1}{\sum_{e=1}^{d'} \left[\frac{B\varphi_{ja}}{B\varphi_{je}}\right]^{\frac{1}{r-1}}} & \text{if } B\varphi_{je} \neq 0\,, \forall 1 \leq e \leq d' \end{cases} \tag{32}$$

where

$$B\varphi_{ja} = \sum_{i=1}^{n} q_{ij}^{\alpha} dist\,(z_{ia}, c_{ja}) - \sum_{i=1}^{n} \eta_j q_{ij}^{\alpha} dist\,(c_{ja}, \bar{z}_a) \tag{33}$$

$\boldsymbol{\Phi} \in \mathbb{R}^{k\times d'}$ is a matrix, where each element φ_{ja} refers to the weight of the a-th feature in the j-th cluster. The theoretical derivation for formula φ_{ja} is provided in section A of the supplementary material. $r > 1$ or $r < 0$ is set manually.

4) Update $\boldsymbol{\sigma}$ by Fixing $\boldsymbol{Q}$, $\boldsymbol{C}$, and $\boldsymbol{\Phi}$:

$$\sigma_j = \begin{cases} \dfrac{1}{\varrho_j} & \text{if } B\sigma_j = 0 \text{ and } \varrho_j = |\{v : B\sigma_v = 0\}| \\ 0 & \text{if } B\sigma_j \neq 0 \text{ and } \exists v\, B\sigma_v = 0 \\ \dfrac{1}{\sum_{v=1}^{k} \left[\frac{B\sigma_j}{B\sigma_v}\right]^{\frac{1}{s-1}}} & \text{if } B\sigma_v \neq 0\,, \forall 1 \leq v \leq k \end{cases} \tag{34}$$

where

$$\begin{aligned} B\sigma_j = &\sum_{i=1}^{n}\sum_{a=1}^{d'} q_{ij}^{\alpha} \varphi_{ja}^r dist\,(z_{ia}, c_{ja}) \\ &- \sum_{i=1}^{n}\sum_{a=1}^{d'} \eta_j q_{ij}^{\alpha} \varphi_{ja}^r dist\,(c_{ja}, \bar{z}_a) \end{aligned} \tag{35}$$

$\boldsymbol{\sigma} \in \mathbb{R}^k$ is a vector, where each element σ_j represents the weight of the j-th cluster. Section A of the supplementary material provides the theoretical derivation for σ_j. $0 \leq s < 1$

Authorized licensed use limited to: XIDIAN UNIVERSITY. Downloaded on September 24,2024 at 00:09:50 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3466175

is determined automatically during iteration. Related explanations regarding r and s can be found in section B of the supplementary material.

The FWFC algorithm is presented in Algorithm 2.

Algorithm 2: The FWFC Algorithm.

Input: The feature representation $\boldsymbol{Z} = \{\boldsymbol{z}_i\}_{i=1}^n$; Number of clusters k; Number of variates d'; Parameters s_{max}, s_{init}, s_{step}, ε, β; Sensitivity coefficient of feature weight r; Fuzzification coefficient α; Maximum iteration: $Maxiter$.

Output: Membership matrix $\boldsymbol{Q}$; Clustering loss $\mathcal{L}_{KL}$.

Initialize $\mathcal{L}_{KL}^{(0)}$=INF; $s_{init} = 0$; $\varphi_{ja}^{(0)} = \frac{1}{d'}, \forall j = 1, ..., k, \forall a = 1, ..., d'$; $\sigma_j^{(0)} = \frac{1}{k}, \forall j = 1, ..., k$;
Initialize cluster centers $\boldsymbol{C}^{(0)}$ with random values;
set empty=FALSE, $s = s_{init}$;
for $iter = 1$ *to* $Maxiter$ **do**
  Update $\boldsymbol{Q}^{(iter)}$ with (28) by fixing $\boldsymbol{C}^{(iter-1)}$, $\boldsymbol{\Phi}^{(iter-1)}$, and $\boldsymbol{\sigma}^{(iter-1)}$;
  Update $\boldsymbol{C}^{(iter)}$ with (31) by fixing $\boldsymbol{Q}^{(iter)}$, $\boldsymbol{\Phi}^{(iter-1)}$, and $\boldsymbol{\sigma}^{(iter-1)}$;
  if empty or singleton clusters have occurred **then**
    empty=TRUE;
    $s = s - s_{step}$;
    if $s < s_{init}$ **then**
      return NULL;
    else
      $\boldsymbol{\Phi}^{(iter)} = \boldsymbol{\Phi}^{(iter-1)}$;
      $\boldsymbol{\sigma}^{(iter)} = \boldsymbol{\sigma}^{(iter-1)}$;
    end if
  else if $s < s_{max}$ **and** empty=FALSE **then**
    $s = s + s_{step}$;
    Update $\boldsymbol{\Phi}^{(iter)}$ with (32) by fixing $\boldsymbol{Q}^{(iter)}$, $\boldsymbol{C}^{(iter)}$, and $\boldsymbol{\sigma}^{(iter-1)}$;
    Update $\boldsymbol{\sigma}^{(iter)}$ with (34) by fixing $\boldsymbol{Q}^{(iter)}$, $\boldsymbol{C}^{(iter)}$, and $\boldsymbol{\Phi}^{(iter)}$;
  end if
  Compute the clustering loss $\mathcal{L}_{KL}^{(iter)}$ with (27);
  until $|\mathcal{L}_{KL}^{(iter)} - \mathcal{L}_{KL}^{(iter-1)}| < \varepsilon$ **or** $iter \geq Maxiter$
end
return Membership matrix $\boldsymbol{Q}$; Clustering loss $\mathcal{L}_{KL}$.

D. The Training Algorithm

We adopt an alternating optimization strategy to train our model, aiming to make the generated imputed values as close to the actual data distribution as possible and achieve optimal clustering results. The algorithm of the proposed EEDFC model is summarized in Algorithm 3.

Algorithm 3: The EEDFC Algorithm.

Input: The input data $\boldsymbol{\chi} = \{\boldsymbol{X}_i\}_{i=1}^n$; The number of clusters k; The coefficients λ_0, λ_1, and λ_2; The maximum iteration: $Maxepoch$.

Output: Clustering results $\boldsymbol{y}$.

for $epoch = 1$ *to* $Maxepoch$ **do**
  Invoke Algorithm 1 to obtain the feature representation $\boldsymbol{Z}$;
  Compute the prediction loss $\mathcal{L}_{PRE}$ with (12), the reconstruction loss $\mathcal{L}_{REC}$ with (18), the discriminator loss $\mathcal{L}_{DIS}$ with (25), the adversarial loss $\mathcal{L}_{ADV}$ with (26);
  Input $\boldsymbol{Z}$ into the clustering module and apply Algorithm 2 to obtain the membership degree $\boldsymbol{Q}$ and clustering loss $\mathcal{L}_{KL}$;
  Compute the final loss $\mathcal{L}_{EEDFC}$ with (9);
  `// Discriminator optimization`
  Adjust discriminator with SGD by (25);
  `// EEDFC optimization`
  Fine feature representation $\boldsymbol{Z}$ with SGD by (9).
end
return Clustering results $\boldsymbol{y}$.

V. Experimental Studies

In this section, we conduct comparative experiments across multiple time series datasets to evaluate the proposed EEDFC's performance. We also conduct ablation studies, parameter sensitivity analyses, robustness analyses at different missing rates, and noise robustness experiments. We present all experimental results, analysis outcomes, and conclusions.

A. Datasets and Experimental Settings

To validate the performance of our proposed model, we apply it to benchmark datasets extracted from the UCR and UCI databases. Detailed statistics for these datasets can be found in section C of the supplementary material. In our work, we configure the autoencoder structure in the model to be a common architecture: $d - 200 - 200 - 500 - d' - 500 - 200 - 200 - d$. Here, d represents original data's dimension, and d' represents the dimension of feature representations, which we set to 10. Both the encoder and decoder networks employ GRU. The batch size is set to 32, and the learning rate is 0.001. We prevent overfitting by applying dropout to hidden layers during model training. Experiments are run on a server with the environment of Intel Core i9-12900H, 2.50-GHz CPU, 16-GB RAM and a GeForce GTX 3060-Ti 8G GPU.

B. Algorithms in Comparison

We select eleven benchmark algorithms to compare with the proposed EEDFC[1]. These algorithms can be divided into two categories: two-stage methods (imputing missing values followed by clustering) and one-stage methods (joint optimization). Two-stage methods include SAITS [27] + Times-C [46], SAITS + conDetSEC [33], SAITS + T-GMRF [47], TIDER [48] + Times-C, TIDER + conDetSEC, TIDER + T-GMRF, BRITS [11] + Times-C, BRITS + conDetSEC, and BRITS + T-GMRF. One-stage methods include VaDER [15] and CRLI [16]. In section D of the supplementary material,

[1]Code available: https://github.com/Du-Team/EEDFC

Authorized licensed use limited to: XIDIAN UNIVERSITY. Downloaded on September 24,2024 at 00:09:50 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3466175

TABLE II: RI COMPARISON OF CLUSTERING PERFORMANCE ON TEN DATASETS

Imputer	Clustering	SC	Meat	Coffee	Ham	Wine	Libras	Epilepsy	RS	UG	EC	**Best**	**avg Rank**
SAITS	Times-C	0.8489	0.7804	0.7311	0.5098	0.4987	0.8951	0.5867	0.507	0.7949	0.5933	0	6
	conDetSEC	0.8785	0.7128	0.5053	0.4955	0.4913	**0.9416**	0.6848	0.7213	0.8425	0.593	1	5.8
	T-GMRF	0.8106	0.7028	0.7169	0.5042	0.5107	0.8771	0.7258	0.5676	0.7327	0.5285	0	7.2
TIDER	Times-C	0.8012	0.6405	0.4894	0.5305	0.4932	0.8916	0.6206	0.6211	0.8287	0.5976	0	7.5
	conDetSEC	0.8987	0.7410	0.4921	0.4986	0.4934	0.9271	0.6528	0.7402	0.8497	0.5769	0	5.3
	T-GMRF	0.8192	0.4554	0.5159	0.528	0.4927	0.8821	0.7314	0.6325	0.7177	0.5189	0	8.1
BRITS	Times-C	0.8647	0.7250	0.6261	0.5340	0.4973	0.8905	0.5541	0.6354	0.8531	0.5989	0	5.2
	conDetSEC	0.8731	0.7968	0.4841	0.4956	0.4969	0.9361	0.6616	0.7212	**0.857**	0.5646	1	5.3
	T-GMRF	0.8316	0.6876	0.6984	0.5161	0.5206	0.8251	0.7318	0.6799	0.726	0.5355	0	6.5
VaDER		0.1639	0.3220	0.4841	0.4956	0.4906	0.0615	0.2463	0.2474	0.1223	0.2471	0	11.6
CRLI		0.8389	0.6947	0.4841	0.5041	0.4969	0.9037	0.5875	0.5035	0.7956	0.6151	0	7.3
EEDFC		**0.9013**	**0.8819**	**1**	**0.781**	**0.8288**	0.9164	**0.8217**	**0.7805**	0.8398	**0.7255**	**8**	**1.7**

TABLE III: NMI COMPARISON OF CLUSTERING PERFORMANCE ON TEN DATASETS

Imputer	Clustering	SC	Meat	Coffee	Ham	Wine	Libras	Epilepsy	RS	UG	EC	**Best**	**avg Rank**
SAITS	Times-C	0.6267	0.6438	0.5100	0.0201	0.0094	0.5361	0.1076	0.0838	0.4332	0.012	0	7
	conDetSEC	0.729	0.5947	0.0336	0.0007	0.001	**0.7442**	0.2565	0.3864	0.539	0.0592	1	4.9
	T-GMRF	0.4826	0.5089	0.493	0.1579	0.111	0.4525	0.3227	0.106	0.234	0.025	0	6
TIDER	Times-C	0.601	0.2827	0.0087	0.0519	0.0045	0.5313	0.1117	0.2312	0.5275	0.0096	0	8.1
	conDetSEC	**0.7454**	0.541	0.0351	0.0053	0.0051	0.6580	0.1852	0.4287	0.535	0.0546	1	5
	T-GMRF	0.4153	0.1112	0.1844	0.2103	0.0103	0.4175	0.3102	0.1099	0.2014	0.013	0	7.6
BRITS	Times-C	0.6953	0.4419	0.2154	0.0568	0.0096	0.5367	0.1215	0.2338	0.5434	0.0072	0	6.3
	conDetSEC	0.7137	0.6058	0.0012	0.0007	0.0197	0.7014	0.1774	0.3611	0.5195	0.04	0	5.4
	T-GMRF	0.4055	0.4992	0.5328	0.1798	0.1065	0.3701	0.3215	0.1635	0.2236	0.018	0	6.4
VaDER		0.0503	0	0	0	0	0	0.0217	0	0	0	0	12.00
CRLI		0.5954	0.3610	0.0012	0.013	0.0283	0.5562	0.1613	0.0516	0.3044	0.0305	0	7.6
EEDFC		0.6955	**0.7594**	**1**	**0.5656**	**0.6374**	0.6702	**0.5983**	**0.5718**	**0.5579**	**0.4385**	**8**	**1.5**

these algorithms are described in detail. For the comparative methods, we use open-source code implementations, adhering to the parameter configurations as specified in the respective papers.

C. Evaluation Metrics

We use four evaluation metrics to assess the clustering performance of the proposed EEDFC and the comparison algorithms. These metrics include rand index (RI), normalized mutual information (NMI), accuracy (ACC), and purity (PUR). The range of values for four metrics is $[0, 1]$. The closer the value of these four metrics is to 1, the better the clustering effect of the model. The details of the metrics are provided in section E of the supplementary material.

D. Experimental Results

In this section, several experiments are conducted to evaluate the performance of the proposed EEDFC in this paper. Tables II - III show the clustering performance of the proposed EEDFC and the comparison methods evaluated by RI and NMI. In section F of the supplementary material, we further compare the performance of other methods and ours on ACC and ARI, and present an experimental analysis. Overall, EEDFC exhibits superior performance, achieving the highest scores on eight out of ten datasets for RI and NMI metrics. The average ranking for the two metrics is 1.7 and 1.5, respectively. A detailed analysis of each metric's performance is given below.

On the RI metric, EEDFC achieves first place on eight datasets, with an average metric ranking of 1.7 on ten datasets. On the Coffee, Ham, and Wine datasets, EEDFC attains the top position with RI values of 1, 0.781, and 0.8288, respectively. These values surpass the second-place scores by 0.2689, 0.247, and 0.3082 for each dataset, where the second-place scores are 0.7311, 0.5340, and 0.5206, respectively. EEDFC's RI values on the Libras and UG datasets are only 0.0252 and 0.0172 lower than the first-place scores, respectively.

Regarding the NMI metric, EEDFC secures the first position on eight datasets, with an average metric ranking of 1.5 on ten datasets. On the Meat, Coffee, Ham, Wine, Epilepsy, RS, UG, and EC datasets, EEDFC secures the top position with performance scores of 0.7594, 1, 0.5656, 0.6374, 0.5983, 0.5718, 0.5579, and 0.4385, respectively. On the Wine and EC datasets, EEDFC achieves NMI values of 0.6374 and 0.4385, respectively, while most competitors perform below 0.1. Moreover, on the Coffee, Ham, and Wine datasets, EEDFC surpasses the second-best method by 0.4672, 0.3553, and 0.5264, respectively, with the latter achieving scores of 0.5328, 0.2103, and 0.111.

In terms of the current one-stage methods, VaDER ranks the lowest on four metrics on average. We suspect that this phenomenon arises from its generative framework's performance, which is more suited to larger datasets but performs inadequately on smaller ones. While the same one-stage method, CRLI, is a discriminative framework that optimizes imputation and clustering simultaneously. This improves the clustering performance for incomplete time series. For two-stage methods, Times-C and T-GMRF perform worse than conDetSEC on average. We speculate that this may be because they only utilize the traditional dimensionality reduction approach,

Authorized licensed use limited to: XIDIAN UNIVERSITY. Downloaded on September 24,2024 at 00:09:50 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3466175

TABLE IV: ABLATION EXPERIMENTS OF MODULE COMPONENTS

Module components	Meat		Ham		Epilepsy		RS	
	RI	NMI	RI	NMI	RI	NMI	RI	NMI
w/o attention mechanism	0.6006	0.2786	0.5055	0.0639	0.5754	0.194	0.6687	0.2234
w/o adversarial strategy	0.7271	0.4917	0.5007	0.0405	0.7728	0.4585	0.6908	0.3571
w/o exponential distance	0.7158	0.5093	0.4956	0.0011	0.4548	0.1704	0.5824	0.1415
w/o feature and cluster weighting	0.6881	0.3633	0.6769	0.3111	0.7032	0.336	0.6481	0.1291
EEDFC	**0.8819**	**0.7594**	**0.781**	**0.5656**	**0.8217**	**0.5983**	**0.7805**	**0.5718**

which ignores the more complex nonlinear relationships in time series data. Unlike Times-C and T-GMRF, conDetSEC employs an autoencoder to automatically learn features based on the task requirements, significantly improving clustering performance. Based on the experimental results, it appears that one-stage methods CRLI and VaDER are not as effective as the two-stage methods in some situations. We speculate that this may be because of the following reasons: (1) Two-stage methods produce better imputation results, so the error introduced in clustering is small enough. (2) Despite the fact that the two-stage methods may introduce some errors in the imputation stage, they are corrected to some extent in the clustering stage by selecting appropriate algorithms and parameters. The one-stage methods CRLI and VaDER have difficulty in controlling the parameters and thus perform less well than the two-stage methods in some cases. Table III indicates that VaDER's NMI score is zero on most datasets because it assigns all samples to a single cluster in these cases. EEDFC outperforms eleven benchmark algorithms, which is attributed to its deep clustering architecture and unified training approach that optimizes imputation and clustering concurrently. Specifically, an attention mechanism is integrated into the encoder to enhance the model's imputed capabilities, which pays more attention to the information associated with missing data. EEDFC incorporates a fuzzy clustering method with weights to align learned features more effectively with the clustering task. Furthermore, an adversarial strategy is employed to mitigate error propagation from imputation to clustering, significantly enhancing EEDFC's performance.

E. Ablation Studies

To verify the effectiveness of each component proposed in EEDFC, we conduct comparison experiments between EEDFC and its four variants. These variants are specifically designed to evaluate the impact of the attention mechanism and adversarial strategy in the imputation module, as well as the exponential distance and weighting mechanism in the clustering module. Specifically, **w/o attention mechanism** removes the attention mechanism, **w/o adversarial strategy** removes the adversarial strategy, **w/o exponential distance** replaces the exponential distance with the euclidean distance, and **w/o feature and cluster weighting** removes the weighting mechanism, setting r and s to fixed values. Table IV presents the comparative experimental results of EEDFC and its four variants on four datasets in terms of RI and NMI metrics. The best performance values for each dataset are highlighted in bold. Fig. 2 presents a visual comparison of clustering performance.

Overall, each module plays an essential role in EEDFC. It is evident from the results that the variant without exponential distance performs the worst, which confirms our motivation for integrating exponential distance into the clustering process is effective. In addition, it can be observed that the attention mechanism plays a more critical role than feature and cluster weighting, which may be attributed to the good performance of the attention mechanism in focusing on key features during representation learning. Furthermore, **w/o attention mechanism** results in an average decrease of 69% in the NMI metric across all datasets, while **w/o adversarial strategy**, which is also part of the imputation module, experiences a decrease of 45%. This indicates that attention mechanism plays a more significant role than adversarial strategy in our proposed method. Based on the results, it appears that all components of EEDFC are indispensable and can significantly improve the overall clustering performance of the proposed method.

To further analyze the significance of each loss component in EEDFC, we conduct an ablation study of loss components on four datasets. We provide detailed analyses and visual results in section G of the supplementary material.

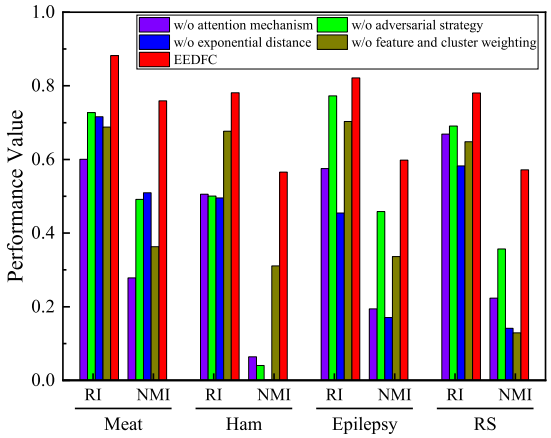


Fig. 2: Histogram visualising of ablation experiments results.

F. Parameter Sensitivity Analysis

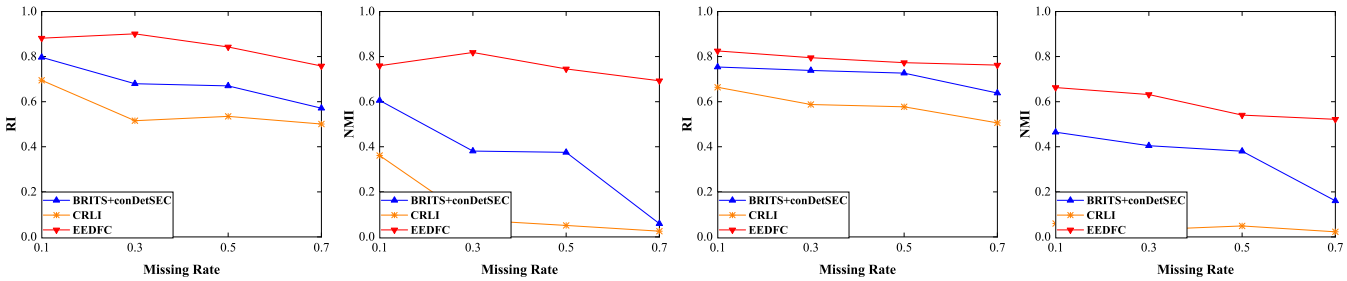
In the proposed EEDFC, hyper-parameters are of critical importance. We use the grid search method to find sensitive regions of hyper-parameters, focusing specifically on the sensitivity analysis of five selected parameters, β, α, λ_0, λ_1, and λ_2. The detailed experimental analysis can be found in section G of the supplementary material.

G. Robustness Analysis

To further investigate the robustness and effectiveness of our method, we conduct robustness experiments on Meat and RS,

Authorized licensed use limited to: XIDIAN UNIVERSITY. Downloaded on September 24,2024 at 00:09:50 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3466175



(a) RI on Meat (b) NMI on Meat (c) RI on RS (d) NMI on RS

Fig. 3: The changing trend of clustering performance with different missing rates on Meat and RS.

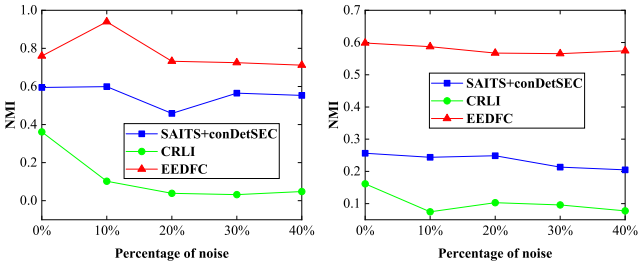
varying the missing rate from 0.1 to 0.7 with an interval of 0.2. We utilize RI and NMI metrics to compare the performance of our method, EEDFC, with two selected comparative algorithms: the one-stage method CRLI and the two-stage method BRITS + conDetSEC. Fig. 3 presents the trend in clustering performance of our method, EEDFC, compared to CRLI, and BRITS + conDetSEC, as the missing rate varies from 0.1 to 0.7. We can observe that the RI and NMI values decrease as the missing rate increases. As can be observed from Fig. 3, with the missing rate increasing, our method's performance decreases slowly in a certain region. However, the performance of the comparative algorithms BRITS + conDetSEC and CRLI significantly decreases. Especially prominent in Fig. 3(b), as the missing rate varies from 0.1 to 0.7, CRLI's performance drops from 0.4 to 0, and BRITS + conDetSEC's degrades from 0.6 to 0. We speculate that our EEDFC performs imputation and clustering simultaneously, and employs an attentional mechanism and an adversarial strategy to reduce the error introduced from imputation to soft clustering, resulting in significant robustness. To further analyze how EEDFC's loss components collaborate to maintain clustering performance when handling datasets with high missing rates, we conduct an ablation study on several datasets with a missing rate of 0.7. Detailed analyses and visual results are presented in section H of the supplementary material.

H. Noise Experiments

To assess the robustness of EEDFC against noise, we introduce a specific ratio of noisy data to the Meat and Epilepsy datasets. Subsequently, we utilize the NMI metric to compare the performance of EEDFC with two alternative methods: the one-stage method CRLI and the two-stage method SAITS + conDetSEC.

Fig. 4(a) presents the experimental results for the Meat dataset with noisy data. Interestingly, EEDFC's performance demonstrates enhancement by introducing approximately 10% noise into the dataset. We speculate that introducing a certain level of noise may act as some regularisation, preventing the model from overfitting and thus improving its generalization performance. In addition, even if the percentage of noise in the data increases, there is no deterioration compared to the results without adding noise. However, with the noise ratio increasing, CRLI's performance drops significantly, and SAITS + conDetSEC's performance also fluctuates. As can be seen in Fig. 4(b), experimental results are presented for the Epilepsy dataset with noisy data. With the addition of noise, the performance of EEDFC is relatively smooth, while CRLI and SAITS + conDetSEC show a gradual decline.

From the above experimental results, it can be concluded that EEDFC is more effective in handling noisy and abnormal data, showing high robustness.



(a) Meat (b) Epilepsy

Fig. 4: Noise robustness analysis on Meat and Epilepsy.

I. Running Time Comparison

To evaluate the computational complexity of EEDFC, we conduct runtime comparison experiments with two one-stage methods across all datasets. We set some key parameters of the two one-stage comparison methods to be the same as those of EEDFC to ensure that the three models operate under the same conditions as much as possible. These parameters include learning rate, batch size, and intermediate representation dimension.

Fig. 5 visualizes the runtime results. It can be observed that VaDER is frequently the slowest in many instances. EEDFC is slightly disadvantaged compared to the other two comparison algorithms when dealing with Coffee, which has a small sample size, and Libras, which has short time steps. However, EEDFC exhibits a clear advantage when processing UG, which has a large sample size, and EC, which has long time steps. It can be concluded that our proposed method demonstrates better scalability on time series data with large sample sizes and long time steps, compared to the other two one-stage methods.

Authorized licensed use limited to: XIDIAN UNIVERSITY. Downloaded on September 24,2024 at 00:09:50 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3466175

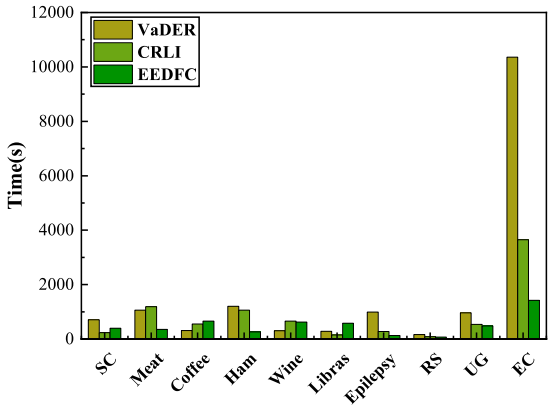
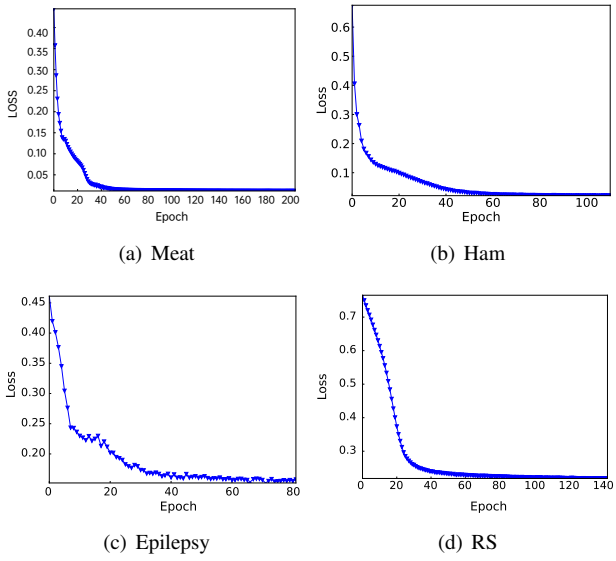


Fig. 5: Running time comparison of one-stage methods.

J. Convergence Analysis

Fig. 6 presents curves of loss values versus epochs for the datasets Meat, Ham, Epilepsy, and RS to illustrate the convergence of EEDFC. As shown in Fig. 6, the loss value decreases rapidly at the beginning and gradually reaches a stable state, indicating that the proposed EEDFC converges well.



(a) Meat (b) Ham

(c) Epilepsy (d) RS

Fig. 6: Relationship curve between loss values and epochs.

VI. Conclusion

This paper proposes a novel end-to-end deep fuzzy clustering model designed for incomplete time series, aiming to concurrently acquire feature representations and identify clusters. Within EEDFC, missing values imputation and clustering are performed simultaneously, improving clustering performance while preserving the original data structure. This approach yields feature representations that are better suited to clustering tasks. In the missing value imputation stage, the attention mechanism is used to retain the most informative and task-relevant parts of the time series, thus maximizing the representativeness of the representation. In the clustering stage, the proposed fuzzy clustering algorithm considers intra-cluster compactness and inter-cluster separability to generate more meaningful clusters. The clustering approach employs local feature weighting so that different features have different weights within each cluster, and the cluster weights are calculated dynamically. In addition, an adversarial strategy is used to mitigate error propagation from imputation to clustering, resulting in high-quality clustering results.

In future work, we plan to investigate the time series multimodal clustering problem more deeply. We will focus on integrating multi-source time series datasets from different domains into a unified clustering model.

References

[1] R. C. Pereira, M. S. Santos, P. P. Rodrigues, and P. H. Abreu, "Reviewing autoencoders for missing data imputation: Technical trends, applications and outcomes," *Journal of Artificial Intelligence Research*, vol. 69, pp. 1255–1285, 2020.

[2] G. Chao, S. Sun, and J. Bi, "A survey on multiview clustering," *IEEE Transactions on Artificial Intelligence*, vol. 2, no. 2, pp. 146–168, 2021.

[3] Y. Zhao, "Statistical inference for missing data mechanisms," *Statistics in Medicine*, vol. 39, no. 28, pp. 4325–4333, 2020.

[4] G. Chao, J. Sun, J. Lu, A. Wang, D. D. Langleben, C. R. Li, and J. Bi, "Multi-view cluster analysis with incomplete data to understand treatment effects," *Information Sciences*, vol. 494, pp. 278–293, 2019.

[5] H. Li, T. Du, and X. Wan, "Time series clustering based on relationship network and community detection," *Expert Systems with Applications*, vol. 216, p. 119481, 2023.

[6] C. Pealat, G. Bouleux, and V. Cheutet, "Improved time-series clustering with umap dimension reduction method," in *Proceedings of the 25th International Conference on Pattern Recognition*, 2021, pp. 5658–5665.

[7] D. Ienco and R. Interdonato, "Deep multivariate time series embedding clustering via attentive-gated autoencoder," in *Advances in the 24th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2020, pp. 318–329.

[8] N. Zhang and S. Sun, "Multiview unsupervised shapelet learning for multivariate time series clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 4, pp. 4981–4996, 2023.

[9] G. Chao, S. Wang, S. Yang, C. Li, and D. Chu, "Incomplete multi-view clustering with multiple imputation and ensemble clustering," *Applied Intelligence*, vol. 52, no. 13, pp. 14 811–14 821, 2022.

[10] C. Fang and C. Wang, "Time series data imputation: A survey on deep learning approaches," *arXiv preprint arXiv:2011.11347*, 2020.

[11] W. Cao, D. Wang, J. Li, H. Zhou, L. Li, and Y. Li, "Brits: Bidirectional recurrent imputation for time series," in *Advances in the 32nd Annual Conference on Neural Information Processing Systems*, 2018, pp. 6776–6786.

[12] J. Yoon, J. Jordon, and M. Schaar, "Gain: Missing data imputation using generative adversarial nets," in *Proceedings of the 35th International Conference on Machine Learning*, 2018, pp. 5675–5684.

[13] G. Chao, Y. Jiang, and D. Chu, "Incomplete contrastive multi-view clustering with high-confidence guiding," in *Proceedings of the 38th AAAI Conference on Artificial Intelligence*, 2024, pp. 11 221–11 229.

[14] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu, "Recurrent neural networks for multivariate time series with missing values," *arXiv preprint arXiv:1606.01865*, 2016.

[15] J. de Jong, M. A. Emon, P. Wu, R. Karki, M. Sood, P. Godard, A. Ahmad, H. Vrooman, M. Hofmann-Apitius, and H. Fröhlich, "Deep learning for clustering of multivariate clinical patient trajectories with missing values," *GigaScience*, vol. 8, no. 11, p. giz134, 2019.

[16] Q. Ma, C. Chen, S. Li, and G. W. Cottrell, "Learning representations for incomplete time series clustering," in *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, 2021, pp. 8837–8846.

[17] H. Guo, M. Wan, L. Wang, X. Liu, and W. Pedrycz, "Weighted fuzzy clustering for time series with trend-based information granulation," *IEEE Transactions on Cybernetics*, vol. 54, no. 2, pp. 903–914, 2024.

[18] Z. Yang, S. Jiang, F. Yu, W. Pedrycz, H. Yang, and Y. Hao, "Linear fuzzy information-granule-based fuzzy c -means algorithm for clustering time series," *IEEE Transactions on Cybernetics*, vol. 53, no. 12, pp. 7622–7634, 2022.

Authorized licensed use limited to: XIDIAN UNIVERSITY. Downloaded on September 24,2024 at 00:09:50 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Fuzzy Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TFUZZ.2024.3466175

[19] J. C. Bezdek, R. Ehrlich, and W. Full, "Fcm: The fuzzy c-means clustering algorithm," *Computers & Geosciences*, vol. 10, no. 2-3, pp. 191–203, 1984.

[20] C. Zhu, X. Ma, W. Ding, and J. Zhan, "Long-term time series forecasting with multi-linear trend fuzzy information granules for LSTM in a periodic framework," *IEEE Transactions on Fuzzy Systems*, vol. 32, no. 1, pp. 322–336, 2023.

[21] L. Shen, Q. Ma, and S. Li, "End-to-end time series imputation via residual short paths," in *Proceedings of the 10th Asian Conference on Machine Learning*, 2018, pp. 248–263.

[22] Y. Luo, X. Cai, Y. Zhang, J. Xu, and X. Yuan, "Multivariate time series imputation with generative adversarial networks," in *Proceedings of the 32nd Annual Conference on Neural Information Processing Systems*, 2018, pp. 1603–1614.

[23] J. Kim, D. Tae, and J. Seok, "A survey of missing data imputation using generative adversarial networks," in *Proceedings of the 9th International Conference on Artificial Intelligence in Information and Communication*, 2020, pp. 454–456.

[24] Y. Luo, Y. Zhang, X. Cai, and X. Yuan, "E^{2}gan: End-to-end generative adversarial network for multivariate time series imputation," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 2019, pp. 3094–3100.

[25] Y. Zhou, S. Wang, T. Wu, L. Feng, W. Wu, J. Luo, X. Zhang, and N. Yan, "For-backward lstm-based missing data reconstruction for time-series landsat images," *GIScience & Remote Sensing*, vol. 59, no. 1, pp. 410–430, 2022.

[26] Q. Suo, W. Zhong, G. Xun, J. Sun, C. Chen, and A. Zhang, "Glima: Global and local time series imputation with multi-directional attention learning," in *Proceedings of the 8th International Conference on Big Data*, 2020, pp. 798–807.

[27] W. Du, D. Côté, and Y. Liu, "SAITS: Self-attention-based imputation for time series," *Expert Systems with Applications*, vol. 219, p. 119619, 2023.

[28] W. Ding, H. Wang, J. Huang, H. Ju, Y. Geng, C.-T. Lin, and W. Pedrycz, "Ftranscnn: Fusing transformer and a cnn based on fuzzy logic for uncertain medical image segmentation," *Information Fusion*, vol. 99, p. 101880, 2023.

[29] T. Xie, W. Ding, J. Zhang, X. Wan, and J. Wang, "Bi-ls-attm: A bidirectional lstm and attention mechanism model for improving image captioning," *Applied Sciences*, vol. 13, no. 13, p. 7916, 2023.

[30] P. Ji, T. Zhang, H. Li, M. Salzmann, and I. Reid, "Deep subspace clustering networks," in *Proceedings of the 31st Annual Conference on Neural Information Processing Systems*, 2017, pp. 24–33.

[31] N. Sai 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.

[32] Q. Ma, J. Zheng, S. Li, and G. W. Cottrell, "Learning representations for time series clustering," in *Proceedings of the 33rd Annual Conference on Neural Information Processing Systems*, 2019, pp. 3776–3786.

[33] D. Ienco and R. Interdonato, "Deep semi-supervised clustering for multi-variate time-series," *Neurocomputing*, vol. 516, pp. 36–47, 2023.

[34] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *Proceedings of the 33rd International Conference on Machine Learning*, 2016, pp. 478–487.

[35] X. Guo, L. Gao, X. Liu, and J. Yin, "Improved deep embedded clustering with local structure preservation." in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 2017, pp. 1753–1759.

[36] K.-L. Wu, J. Yu, and M.-S. Yang, "A novel fuzzy clustering algorithm based on a fuzzy scatter matrix with optimality tests," *Pattern Recognition Letters*, vol. 26, no. 5, pp. 639–652, 2005.

[37] Y. Tang, J. Huang, W. Pedrycz, B. Li, and F. Ren, "A fuzzy clustering validity index induced by triple center relation," *IEEE Transactions on Cybernetics*, vol. 53, no. 8, pp. 5024–5036, 2023.

[38] F. Nie, C. Liu, R. Wang, Z. Wang, and X. Li, "Fast fuzzy clustering based on anchor graph," *IEEE Transactions on Fuzzy Systems*, vol. 30, no. 7, pp. 2375–2387, 2022.

[39] C. Liu, F. Nie, R. Wang, and X. Li, "Graph-based soft-balanced fuzzy clustering," *IEEE Transactions on Fuzzy Systems*, vol. 31, no. 6, pp. 2044–2055, 2023.

[40] M. Chen, M. Gong, and X. Li, "Feature weighted non-negative matrix factorization," *IEEE Transactions on Cybernetics*, vol. 53, no. 2, pp. 1093–1105, 2023.

[41] F. Nie, X. Zhao, R. Wang, X. Li, and Z. Li, "Fuzzy k-means clustering with discriminative embedding," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 3, pp. 1221–1230, 2022.

[42] Q. Feng, L. Chen, C. P. Chen, and L. Guo, "Deep fuzzy clustering-A representation learning approach," *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 7, pp. 1420–1433, 2020.

[43] M. Gong, Y. Zhao, H. Li, A. K. Qin, L. Xing, J. Li, Y. Liu, and Y. Liu, "Deep fuzzy variable c-means clustering incorporated with curriculum learning," *IEEE Transactions on Fuzzy Systems*, vol. 31, no. 12, pp. 4321–4335, 2023.

[44] M. Chen, Q. Wang, and X. Li, "Adaptive projected matrix factorization method for data clustering," *Neurocomputing*, vol. 306, pp. 182–188, 2018.

[45] K.-L. Wu and M.-S. Yang, "Alternative c-means clustering algorithms," *Pattern recognition*, vol. 35, no. 10, pp. 2267–2278, 2002.

[46] X. Wang, R. Song, J. Xiao, T. Li, and X. Li, "Accelerating k-shape time series clustering algorithm using GPU," *IEEE Transactions on Parallel and Distributed Systems*, vol. 34, no. 10, pp. 2718–2734, 2023.

[47] W. Ding, W. Li, Z. Zhang, C. Wan, J. Duan, and S. Lu, "Time-varying gaussian markov random fields learning for multivariate time series clustering," *IEEE Transactions on Knowledge and Data Engineering*, no. 11, pp. 11 950–11 966, 2022.

[48] S. Liu, X. Li, G. Cong, Y. Chen, and Y. Jiang, "Multivariate time-series imputation with disentangled temporal representations," in *Proceedings of the 11th International Conference on Learning Representations*, 2023, pp. 1–19.

Yurui Li received the bachelor's degree from North China University of Science and Technology, Tangshan, China, in 2020.

She is currently pursuing the master's degree with the School of Computer Science and Technology, Jiangsu Normal University, Xuzhou, China. Her research interests cover deep learning, clustering and time-series analysis.

Mingjing Du (Member, IEEE) received the Ph.D. degree in computer science from the China University of Mining and Technology, Xuzhou, China, in 2018.

He is currently an associate professor with the School of Computer Science and Technology, Jiangsu Normal University, Xuzhou, China. His current research interests include cluster analysis and three-way decisions. For more information, see https://dumingjing.github.io/

Wenbin Zhang (Member, IEEE) received the M.S. degree from Soochow University, Suzhou, China, in 2010.

He is currently pursuing the Ph.D. degree at Shandong University of Science and Technology, Qingdao, China. He is currently an assistant professor in the School of Computer Science and Technology, Jiangsu Normal University, Xuzhou, China. His research interests include machine learning, time-series analysis and deep learning.

Xiang Jiang received the Ph.D. degree in signal and information processing from Beijing JiaoTong University, Beijing, China, in 2022.

He is currently a assistant professor with the School of Computer Science and Technology, Jiangsu Normal University, Xuzhou, China. His current research interests include image forensic and computer vision.

Yongquan Dong received the Ph.D. degree in computer science from Shandong University, Jinan, China, in 2010.

He is currently a professor with the School of Computer Science and Technology, Jiangsu Normal University, Xuzhou, China. His research interests include web information integration and web data management.

Authorized licensed use limited to: XIDIAN UNIVERSITY. Downloaded on September 24,2024 at 00:09:50 UTC from IEEE Xplore. Restrictions apply.
© 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.