EAI Endorsed Transactions
on Scalable Information Systems
Research Article **EAI.EU**

Deep spectral network for time series clustering

Duc-Trung Hoang[1,*], Mahdi Achache[2], Vinay Kumar Jain[3]

[1]Institut d'Informatique d'Auvergne (ISIMA), Clermont Auvergne University, Clermont-Ferrand, France.
Email: duc_trung.hoang@uca.fr
[2]Institut de Mathématiques (I2M), Aix Marseille University, French National Centre for Scientific Research (CNRS), Marseille, France.
Email: achachemahdi@hotmail.fr
[3]Department of Numerical Modelling, French Institute of Petroleum (IFPEN), 69360, Solaize, France.
Email: vinay.jain@ifpen.com

Abstract

Deep clustering is an approach that uses deep learning to cluster data, since it involves training a neural network model to become familiar with a data representation that is suitable for clustering. Deep clustering has been applied to a wide range of data types, including images, texts, time series and has the advantage of being able to automatically learn features from the data, which can be more effective than using hand-crafted features. It is also able to handle high-dimensional data, such as time series with many variables, which can be challenging for traditional clustering techniques. In this paper, we introduce a novel deep neural network type to improve the performance of the auto-encoder part by ignoring the unnecessary extra-noises and labelling the input data. Our approach is helpful when just a limited amount of labelled data is available, but labelling a big amount of data would be costly or time-consuming. It also applies for the data in high-dimensional and difficult to define a good set of features for clustering.

Received on 10 July 2023; accepted on 22 September 2023; published on 25 September 2023

Keywords: Deep clustering, neural network, time series, spectral clustering, mutual information, multi label classification

doi:10.4108/eetsis.3907

1. Introduction

Clustering for time series is always an interesting and challenging problem for several main reasons [2]. While the most popular time series data types are univariate, multivariate, or tensor fields, the time sequence data's high dimensionality and irregular lengths present numerous difficulties for the conventional clustering techniques in actual use [1]. For conventional approaches, the partition-based methods or density-based methods are often used in many fields for their simpleness, but they are also appeared the weaknesses [15]. For example, the hierarchical agglomerate clustering (HAC) method often raise drawbacks such as the ambiguity in determining the final cluster number and the difficulty in selecting a suitable distance metric when merging two clusters. On the other hand, using the clustering for the time series with k-means methods also raise some disadvantages, such as the imprecision of silhouette analysis score or the difficulty to determine the values of k. Furthermore, the "mean" of time series with different length could not be well-defined, and the assumptions on k-means often require the following conditions:

- Considering the dataset's balanced cluster size.
- Taking into account the fact that features within a cluster are independent and have equal variance.
- Hypothesizing on the similar density of available clusters.

Instead of using the central grouping methods, the graph-based clustering method such as the spectral clustering is introduced here. A first approach

*This research is done by a finanical aid from Analgesia Institut. Address: 28 Place Henri Dunant, 63000 Clermont-Ferrand, France.
†Corresponding author. *duc_trung.hoang@uca.fr*

for clustering the time series, based on spectral decomposition of the affinity matrix and then a low-dimensional space k-means clustering, could be found in the works of [3, 35]. The spectral clustering is not only be able to dual with the high dimensionality of arbitrary length time series, but also can determine automatically the optimal cluster number and self-tune the variance of the Gaussian kernel. Moreover, Wang et al.'s work [35] provides theoretical support for the feasibility of our approach. For a large-scale and high-dimensional data, we refer the spectral clustering based on iterative optimization method of Zhao et al [34]. A set of clustering techniques known as "deep clustering" uses deep neural networks to develop representations that are conducive to clustering. The last decade observes the use of deep learning for the clustering problem as a potentially new research trends in this domain because of its ability to capture the non-linearity in the data representations. The most well-known neural network architectures are considered as the fully-connected neural network (FCN), the convolutional neural network (CNN), the deep belief network (DBN), the autoencoder (AE), and the generative adversarial net work (GAN) [27]. Without using neural network, the classical representations methods such as principal component analysis (PCA), multidimensional scaling (MDS), discrete Fourier transform (DFT), or discrete wavelet transform (DWT) often reduce the form of dimension to get better loss function and achieve the higher accuracy of clustering. In [31], the authors investigate on a deep learning approach for approximate spectral clustering called "SpectralNet". This is a very first beginning neural network model to linearize the input data, then applying directly the spectral clustering for the encoding part. However, this model does not introduce better methods to improve the data representations and enhance the overall loss functions. Other several deep clustering models could be found in [1, 27]. The deep clustering convolution neural network for image data such as the DAE or DCAE model could be found in the work of Huang et al [20] and Guo et al [17]. Recently, there are some interesting methods to enhance the representation methods in the encoding parts such as the self-organizing map (SOM), the growing neural gas (GNG), and the generative adversarial networks (GANs). In the case of missing data, a CRLI model is proposed by Ma et el with jointly optimization methods on two parallel branches [24]. There are some deep clustering model proposed in the specific context of time series, such as the deep temporal clustering (DTC) [26] and the deep temporal clustering representation (DTCR) [23].

Motivated by the existing approaches, we propose an improved spectral clustering based on bidirectional dilated recurrent neural networks, namely DSTR, for large-scale and high-dimensional time series data. Even though an outcome variable exists or some basic information regarding the clusters is available in many circumstances, one may still wish to run a cluster analysis. Using another approach from [23], we employ a novel model that extends conventional cross entropy minimization to an ideal transport problem by maximizing the information between labels and input data indices. In reality, clustering after using latent representation via the neural network is an unsupervised task that may lead to enhance the model's complexity and may give the trivial solutions [32]. To increase the information between data indices and labels, we add the restriction that the labels must induce an equi-partition of the data. This appears to transform the task into a semi-unsupervised problem.

2. State of the art

Problem statement : Considering the time series $\mathbf{X} = \{x_i\}_{i=1}^M = \{x_1, x_2, ..., x_M\}$. Each time series has the same lengths d, i.e: $x_i = [x_{i,1}, x_{i,2}, ..., x_{i,d}]$ for all $i \in [1, M]$, and that leads to a data matrix $\mathcal{A} = [x_1, x_2, ..., x_M]$. The goal of our problem is to cluster the set $\mathbf{X}$ into a finite partition $\mathcal{C} = \{\mathcal{C}_1, ..., \mathcal{C}_k\}$ of k clusters in a way that maximizes the similarity between objects within a cluster while minimizing the dissimilarity between objects in different clusters. Then $\mathcal{C}_i$ is called a cluster where $\mathcal{A} = \cup_{i=1}^M \mathcal{C}_i$ and $\mathcal{C}_i \cap \mathcal{C}_j = \emptyset$ for all $i \neq j$. Lets us denote $\mathcal{X}$ be the pairwise constraint matrix and F be the function that map the set $\mathbf{X}$ into k clusters, then the map result is resulted by $\mathcal{C}$. The most representative approaches for the classical problem of time series clustering could be classified into the following main categories: hierarchical based method that agglomerate each item and merge the clusters from the bottom-up, partitioning based method that decompose a data set into a set of disjoint clusters from unlabeled objects such that each cluster contain at least one element, model-based method that recover the original model from a set of data and could fit the data well, density-based method, grid-based method, and multistep method.

Deep clustering is a machine learning technique that combines unsupervised learning (e.g., clustering) with deep learning. It is often used to learn meaningful representations of data in an unsupervised manner, and it can be implemented using any of the aforementioned architectures. In practice, the taxonomy function $\mathcal{R}$ contains two modules: ***the representation module*** and ***the clustering module***, which play central roles in modern deep clustering. Normally, we have the following four kinds of deep clustering problem:

- The one view deep clustering problem: $\mathcal{R}(\mathbf{X}) \rightarrow \mathcal{C}$.

- The partially supervised deep clustering problem: $\mathcal{R}(\mathbf{X}, \mathcal{X}) \rightarrow \mathcal{C}$.

- The multi view deep clustering problem: $\mathcal{R}(\mathbf{X}_1, ..., \mathbf{X}_n) \rightarrow \mathcal{C}$ where $\mathbf{X}_i$ be the i^{th} view of $\mathbf{X}$.

- The domain adaptation deep clustering problem: $\mathcal{R}(\mathbf{X}_i, \mathcal{X}_i, \mathbf{X}_j) \rightarrow \mathcal{C}$ where $(\mathbf{X}_i, \mathcal{X}_i)$ be the identified source and $\mathbf{X}_j$ be the target domain without labels.

When performing traditional clustering tasks, it is frequently believed that all of the data are one view or simple modal, which have the same shape and structure. There are five different types of one view deep clustering algorithms: the DAE, the DNN, the VAE, the GAN, and the GNN types [42]. When the data to be processed only include a tiny amount of prior restrictions, standard clustering techniques cannot effectively use the prior information [42]. The multi-view deep clustering uses the consistent and supplementary information present in multi view data, incorporating these frameworks with DEC type, subspace clustering-based, and GNN type, to enhance clustering performance [42]. The approaches for deep clustering based on transfer learning such as DNN type or GAN type aim to improve the effectiveness of existing clustering tasks using data from pertinent tasks [42].

Model-based approaches: The most well-known architectures for the deep clustering are Fully Connected Neural Network (FCNN), Convolution Neural Network (CNN), and Recurrent Neural Network (RNN) [1]:

- FCNNs are a type of feedforward neural network that consists of layers of interconnected "neurons," where every neuron in one layer is coupled to every neuron in the following layer. FCNNs are generally used for tasks that involve input data with a fixed-size feature vector, such as image classification or regression.

- CNNs, a particular type of neural network, are particularly well suited for the tasks requiring data with a spatial structure, such those involving images. CNNs are composed of layers of convolutional filters, which are used to learn local patterns in the data, as well as pooling layers, which are used to reduce the spatial resolution of the data and increase the network's ability to generalize.

- RNNs are a particular kind of neural network that function well for tasks that need sequential data, including time series forecasting or natural language processing. RNNs are composed of "recurrent" units, which allow the network to process data with an arbitrary length by maintaining a hidden state that is updated at each time step.

The FCNN architectures could be considered as Deep Embedded Clustering (DEC) (see Xie et al. [32]) and Improved Deep Embedded Clustering (IDEC) (see Guo et al. [18]). Meanwhile, the loss function is referred to the deep clustering model's optimization objective. While the DEC model propose a unique clustering loss, namely Kullback-Leibler (KL) divergence, the IDEC generalize it by adding a reconstruction in the total loss function $\mathfrak{L} = \lambda\mathfrak{L}_{\text{network}} + (1-\lambda)\mathfrak{L}_{\text{clustering}}$ where $\lambda \in [0, 1]$. The total loss function for the IDEC model is a weighted sum of the network loss and the clustering loss, where the weighting factor λ determines the relative importance of each term. When $\lambda = 0$, the model is equivalent to the DEC model, and when $\lambda = 1$, the model is equivalent to an autoencoder. By varying λ, it is possible to trade off the reconstruction error for the clustering quality, or vice versa. There are several variations of the IDEC model that have been proposed in the literature. Some examples include:

- The Structural Deep Clustering Network (SDCN), which incorporates structural information into the clustering process by adding a graph convolutional layer to the IDEC model (see Bo et al [6]).

- The Deep Embedded Regularized Clustering (DEPICT) model, which adds a regularization term to the overall loss function of the IDEC model to encourage the latent representation to be smooth and continuous (see Dizaji et al [13]).

- The Deep Clustering Network (DCN), which extends the IDEC model by adding a deep autoencoder to the network architecture and using a new clustering loss function based on the minimum entropy principle (see Yang et al [38]).

- The Deep Adaptive Image Clustering (DAIC) model, which combines the IDEC model with an adversarial learning framework to improve the robustness and generalization of the learned representations (see Chang et al [8]).

- The Deep Clustering with Convolutional Autoencoder (DCCAE) model, which uses a convolutional autoencoder as the encoder-decoder component of the IDEC model, and a new clustering loss function based on the maximum mean discrepancy (MMD) criterion (see Guo et al [18]).

There are many well-known configurations of convolutional neural networks (CNNs) that have been proposed in the literature. Some examples include:

- LeNet: a classic CNN architecture that includes a number of convolutional and pooling layers, followed by a small number of fully linked layers. LeNet was among the first CNNs to be utilized successfully for tasks like handwritten digit recognition (see Lecun et al [11]).
- AlexNet: a CNN architecture with three fully linked layers and five convolutional layers, and it was the first time that CNN has triumphed in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) (see Krizhevsky et al [21]).
- VGG: A number of convolutional layers precede a few fully linked layers in the architecture, and it is known for its good performance on image classification tasks (see Simonyan et al [29]).
- ResNet: a CNN architecture that was introduced by Wang et al. in their paper "Deep Residual Learning for Image Recognition." It consists of a series of residual blocks, which are composed of convolutional layers and shortcut connections, and it is known for its ability to train very deep networks without suffering from the vanishing gradient problem. (see Wang et al [36]).
- Dilated convolutional neural networks (DCNN): a type of CNN that use "dilated" convolutions, which have a larger effective receptive field than standard convolutions. Dilated convolutions can be used to increase the context captured by the network without increasing the number of parameters or the computational cost, and they have been shown to be effective in tasks such as semantic segmentation and image restoration (see Franceschi et al [16]).

For the recurrent neural network, we have more configurations such as the Deep Temporal Clustering (DTC) (see Madiraju et al [26]), the Bidirectional Long Short Term Memory (BLSTM), the Bidirectional Gate Recurrent Unit (BGRU), the Dilated Recurrent Neural Network (see Ma et al. [23]).

There are many different configurations of recurrent neural networks (RNNs) that have been proposed in the literature. Some examples include:

- Deep Temporal Clustering (DTC): a RNN architecture that was introduced by Madiraju et al. in their paper "Deep Temporal Clustering: Unsupervised Learning of Temporal Patterns." It is designed for unsupervised learning of temporal patterns in data, and it consists of a RNN encoder and a clustering layer (see Madiraju et al [26]).
- Bidirectional Long Short Term Memory (BLSTM): an RNN design made up of long short term memory (LSTM) units coupled in a "bidirectional" fashion. This means that the network processes the data in both forward and backward directions, which allows it to capture contextual information from both the past and the future. BLSTMs are often used for tasks involving sequential data, such as natural language processing (see Yulita et al [39]).
- Bidirectional Gated Recurrent Unit (BGRU): a RNN architecture that is similar to BLSTM, but it uses gated recurrent units (GRUs) instead of LSTM units. GRUs are a simpler variant of LSTMs that have fewer parameters and are easier to train. BGRUs are often used for tasks involving sequential data, such as natural language processing (see Sammani et al [28]).
- Dilated Recurrent Neural Network (DRNN): a type of RNN architecture, used as "dilated" recurrent units, which have a larger effective receptive field than standard recurrent units. Dilated recurrent units could be used to increase the context captured by the network without increasing the number of parameters or the computational cost, and they have been shown to be effective in tasks such as natural language processing and machine translation (see Chang et al. [8]).
- Variational recurrent neural networks (VRNNs): a type of recurrent neural network (RNN) that combines an RNN architecture with a probabilistic model. Indeed, it could be used to learn latent representations of a sequential data such as the natural language or time series data (see Chien et al [9]).

The representation module: Recently, the use of deep learning technique for representation learning problems, especially the unsupervised ones, raise the new trends for this research area. However, the majority of approaches now in use are not specifically intended for clustering tasks, and they are therefore unable to incorporate possible clustering information to learn improved representations [20]. There are several unsupervised techniques in representative theory such as the clustering friendly, auto encoder, subspace, deep generative, mutual information maximization, and the contractive representation learning [42]. The data used for representation learning could be various types of the aforementioned architectures such as images, texts, videos, or graphs [42].

Auto-Encoder based: The representation of time series obtained from deep neural network is called an encoder. For different data formats, such as the vectors, the images, the graphs, or the videos, the general structure of the auto-encoder can be changed.

Generative model based: The generative model is a further branch of deep unsupervised representation learning, which seek to provide fresh data samples like a training dataset. By reversibly deducing the posterior of the representation $p(h|x)$ from the data, generative approaches presume that the data x were produced from a latent representation h. There are many different types of generative models, ranging from simple models such as the Gaussian mixture model to more complex models such as generative adversarial networks (GANs) [22]. Deep generative models, which are implemented using deep neural networks, have gained a lot of popularity in recent years due to their ability to learn complex, high-dimensional distributions and generate high-quality synthetic data [14]. Among these, variational auto-encoder (VAE) is the most used technique, and the VAE-based deep clustering algorithms aim at solving an optimization problem about evidence lower bound (ELBO) on the data likelihood $\mathfrak{L}_{\mathrm{ELBO}} = \mathbb{E}_{\mathrm{p(h|x)}}\left[\log \frac{\mathrm{p(x,h)}}{\mathrm{q(h|x)}}\right]$. In this expression, $p(x,h)$ is the true joint distribution over observed variables x and latent variables h, and $q(h|x)$ is the approximating distribution, also known as the variational distribution. The term $\mathbb{E}_{p(h|x)}$ represents the expectation with respect to the posterior distribution $p(h|x)$.

Mutual Information: Adopting in unsupervised learning, the mutual information is used for the discrete variables or known probability distribution to measure of the amount of information that one random variable contains about another. It is defined as the expected value of the logarithm of the ratio of the joint probability distribution of the two variables to the product of their individual probability distributions. The mutual information $\mathcal{I}(U,V)$ is introduced to measure the dependence between two random variables U and V.

$$\mathcal{I}(U,V) = \int \log \frac{d\mathbf{P}_{UV}}{d\mathbf{P}_U \otimes d\mathbf{P}_V} d\mathbf{P}_{UV}$$

where $\mathbf{P}_{UV}$ be the joint distribution, $\mathbf{P}_U = \int_V d\mathbf{P}_{UV}$, $\mathbf{P}_Y = \int_U d\mathbf{P}_{UV}$, and $\mathbf{P}_U \otimes \mathbf{P}_V$ are the individual probability distributions of U and V, respectively. To remind, the work of Hjelm et al. [19] introduced the efficiency of representation learning through deep neural network encoders that maximize mutual information between input and output. Considering a discriminator function D modeled by a neural network, the Jensen-Shannon divergence is a popular mutual information estimate method:

$$\mathcal{I}_{JSD}(X,H) = \mathbb{E}_{\mathbf{P}_{XH}}[-\log(1+\exp(-D(x,h))] \\ -\mathbb{E}_{\mathbf{P}_X \times \mathbf{P}_X}[\log(1+\exp(D(x,h))]$$

In [37], the authors introduce the mutual information $\mathcal{I}(X,H,\theta)$ between the input X and the features H with an encoder parameter θ to provide more discriminative information from the inputs such that the learned latent representations could be more robust to noise. Using the Jensen–Shannon divergence estimation, the authors maximize the term $\mathcal{I}(X,H,\theta)$ as much as possible when training the encoder network.

Similarity measure: The choice of distance measure directly affect the clustering performance. Let's consider $\mathrm{dist}(x_i,x_j)$ be the sum of distance between two time series x_i and x_j. i.e: $\mathrm{dist}(x_i,x_j) = \sum_{t=1}^{d} \mathrm{dist}(x_{i,t},x_{i,t})$. Considering the Steiner sequence of time series R_j that minimize the distance sum between time series observation and cluster prototype and the average distance:

$$E(C_i,R_j) = \frac{1}{n}\sum_{m=1}^{n} \mathrm{dist}(F_m,R_j), \quad C_i = \{F_1,F_2,..,F_n\}$$

This expression is a formula for the average distance between a set of feature vectors F_1, F_2, ..., F_n and a reference vector R_j. Here, C_i represents the set of feature vectors, and $E(C_i,R_j)$ is the average distance between the feature vectors and the reference vector. In a clustering context, the reference vector might represent the centroid of a cluster, and the average distance could be used as a measure of the compactness of the cluster. In practice, the most familiar metrics are the Euclidean distance, the dynamic time warping (DTW), the Hausdorff distance and the shape-based distance.

3. Model architecture

In this paper, we introduce a new kind of deep neural network that consist of three kinds of loss consisting the principal network loss, the classification loss and the auxiliary clustering loss. The deep embedding clustering (DEC) [32] is considered as the first model to propose the clustering loss, namely Kullback-Leibler (KL) divergence. After that, the improved deep embedded clustering (IDEC) [18] generalize the DEC model by adding a reconstruction in the total loss function:

$$\mathfrak{L} = \lambda\mathfrak{L}_{\mathrm{network}} + (1-\lambda)\mathfrak{L}_{\mathrm{clustering}}$$

In the paper of [23], the authors introduce a binary classification loss that distinguish the real and fake time series data. In details, this model consist around 20% extra-data from the input as the noises, then try to use a soft-max formula to classify the data after encoding. However, the authors do not model well the noise for

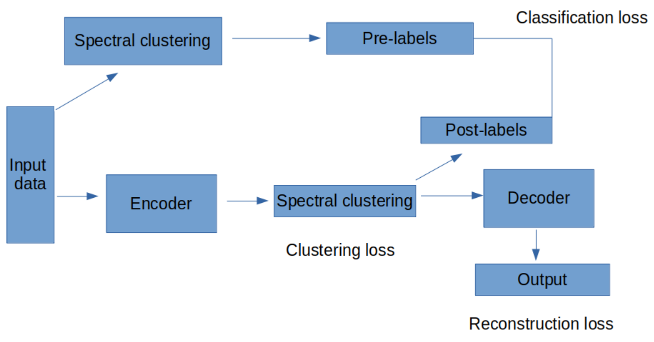


Figure 1. The general architecture design for deep clustering network.

the information of its probability distributions in deep neural network, and that could be make the model loss the regularization. Also, this mutual information is not well estimated with current loss function. Our plan is to employ some semi-supervised clustering algorithms (or occasionally supervised clustering methods) that can be used with partially labelled data or data that contains other kinds of outcome metrics. By proposing the new architecture when eliminating the excess noise that is superfluous and labeling the input data, we increase the performance of the auto-encoder component, as well as giving the new the classification losses. In practice, learning from labelled data could give easier implements and dramatically reduce the cost of deploying algorithms. In fact, we require the labels the data to train the network, and we require the network to predict the labels of the outcome. We use the dilated recurrent neural network (DRNN) proposed by Chang et al [8] that could reduce the number of network parameters and improve the training efficiency. Overall, the total loss of deep neural network model could be considered as:

$$\mathfrak{L}_{\text{overall}} = \lambda\mathfrak{L}_{\text{network}} + \gamma\mathfrak{L}_{\text{classification}} + (1-\lambda-\gamma)\mathfrak{L}_{\text{clustering}}$$

In the case of $\gamma = 0$, our model is collapse to the IDEC architecture [18].

3.1. Labelling data via spectral clustering

At the beginning, we try to assign pseudo-labels for each element in the original data set A. By using the spectral clustering, one could categorize the data into the eigenspace associated to the Laplacian matrix. In fact, the whole data set could be viewed as a graph with each node as an observation and edges as the similarity. The goal of the clustering problem is to divide the graph into smaller parts so that edges within each subcomponent have a high degree of similarity and edges across subcomponents have a low degree of similarity. Such partitions can be obtained by solving the mincut problem. Let's define the weight function $w : \mathbb{R}^d \times \mathbb{R}^d \to (0, +\infty)$. The loss function of spectral clustering is defined as:

$$\mathfrak{L}_{\text{spectral}} = \frac{1}{m^2}\sum_{i,j=1} e^{-\frac{d(x_i,x_j)}{2\sigma^2}}\|x_i - x_j\|^2 \tag{1}$$

where $d(x,y)$ be a Gaussian type of the distance metric between two vectors x and y, σ be the scaling parameter, and m be the sample mini-batch of at each iteration. In our case, we choose the dynamic time warping distance (DTW) as the metric. According to [3], the processes of spectral clustering are given as:

- Construct an affinity matrix $\mathcal{W} \in \mathbb{R}^{M\times M}$ in which its diagonal elements are set to be zero while others elements $\mathcal{W}_{ij} = e^{-\frac{d(x_i,x_j)}{2\sigma^2}}$.
- Construct a diagonal matrix $\mathcal{K}$ of size $m \times m$ such that the i^{th} row's diagonal element contains $\mathcal{K}_{i,i} = \sum_j \mathcal{W}_{i,j}$, then a normalized Laplacian matrix could be used as $\mathcal{L} = \mathcal{K}^{-1/2}(\mathcal{K} - \mathcal{W})\mathcal{K}^{-1/2}$.
- Creating an orthogonal matrix $\mathcal{O}$ consisting the k largest eigenvectors of $\mathcal{L}$, then normalizing $\mathcal{O}$ to have unit length for each row.
- Clustering every row of $\mathcal{O}$ into k clusters, then labeling the point x_i into cluster j if and only if the row i of $\mathcal{O}$ is belong to cluster j.

3.2. Clustering via latent representation

We define a non-linear mapping $\Phi_{\text{enc}}(\theta) : x_i \to h_i$ that define the latent representation for encoding (AE-based) i.e: $f(x_i, \theta) = h_i \in \mathbb{R}^h$ where $h < d$. The new space created by the map $\Phi_{\text{enc}}(\theta)$ is called the latent space, in opposition to the original data space. Let's us denote the latent representation as $H(\theta) = [h_1, h_2, ..., h_M] = [\Phi(x_1, \theta), \Phi(x_2, \theta), ..., \Phi(x_M, \theta)]$ and E is a permutation matrix of size k, then $HE = [H_1, H_2, ..., H_k]$ represent a clustering process into k different groups where the i-th cluster is defined as:

$$H_i(\theta) = [\Phi(x_{i1}, \theta), \Phi(x_{i2}, \theta), ..., \Phi(x_{is_i}, \theta)] \qquad \text{and}$$

$$\text{Card}(H_i(\theta)) = s_i.$$

We denote m as the sample means of the latent representation i.e $m(\theta) = \frac{1}{M}\sum_{i=1}^{M}\Phi(x_i, \theta)$, and for each $i \in [1, k]$, denoting the mean vector of the i-th cluster as $m_i(\theta) = \frac{1}{s_i}\sum_{j=1}^{s_i}\Phi(x_{ij}, \theta)$. The total within-cluster scatter matrix (Total_{wsm}) and the total between-cluster scatter matrix (Total_{bsm}) for the encoding part could be considered as:

$$\text{Total}_{wsm}(\theta) = \sum_{i=1}^{k}\sum_{s_j \in i}\|\Phi(x_{s_j}, \theta) - m_i(\theta)\|^2 \qquad \text{and}$$

$$\text{Total}_{bsm}(\theta) = \sum_{i=1}^{k} s_k \|m_i(\theta) - m(\theta)\|^2$$

According to [35], the total-data scatter matrix (Total_{dsm}) is not depended on the number of the cluster. In fact:

$$\begin{aligned}\text{Total}_{dsm}(\theta) &= \text{Total}_{wsm}(\theta) + \text{Total}_{bsm}(\theta) \\ &= \sum_{i=1}^{M} \|\Phi(x_i, \theta) - m_i(\theta)\|^2\end{aligned} \tag{2}$$

To obtain high and low between cluster similarity, we should decrease trace ($\text{Total}_{wsm}(\theta)$) and inscrease trace ($\text{Total}_{bsm}(\theta)$).. Noting that the minimization of trace (Total_{wsm}) and maximization of trace (Total_{bsm}) are two equivalent problems. We have:

$$\begin{aligned}&\text{Trace}\Big(\text{Total}_{wsm}(\theta)\Big) \\ &= \text{Trace}\Big(\textstyle\sum_{i=1}^{k} \sum_{s_j \in i}\Big(\|\Phi(x_{s_j}, \theta)\|^2 + \|m_i(\theta)\|^2 - \\ &2\Phi(x_{s_j}, \theta) m_i(\theta)^T\Big)\Big) \\ &= \text{Trace}\Big(\textstyle\sum_{i=1}^{k} (H_i(\theta) H_i^T(\theta) - \frac{1}{s_i} H_i(\theta) e_i e_i^T H_i(\theta)^T)\Big) \\ &= \text{Trace}\Big((H(\theta) H^T(\theta)\Big) - \text{Trace}\Big(Q^T H^T(\theta) H(\theta) Q\Big)\end{aligned} \tag{3}$$

where $e_i \in \mathbb{R}^{s_i}$ be the vector of values 1 for $\forall i$, and Q be the block-diagonal orthogonal matrix i.e $Q^T Q = I$. The solution of the problem $\min_{\theta} \text{Trace}\Big(\text{Total}_{wsm}(\theta)\Big)$ could be found in the theorem of KyFan [33].

3.3. Reconstruct loss

If we define the non-linear decoding mapping $\Psi_{dec}(\theta') : h_i \to \hat{x}_i \in \mathbb{R}^M$ i.e $\Psi_{dec}(h_i, \theta') = \hat{x}_i$, then the loss function in the reconstructing phase should be considered in L^2-norm, i.e the mean square error.

$$\begin{aligned}\mathfrak{L}_{\text{reconstruction}} &= \frac{1}{M} \sum_{i=1}^{M} \|x_i - \hat{x}_i\|_2^2 \\ &= \frac{1}{M} \sum_{i=1}^{M} \|\Phi_{\text{enc}}^{-1}(h_i, \theta) - \Psi_{\text{dec}}(h_i, \theta)\|_2^2\end{aligned} \tag{4}$$

The optimization problem for the reconstruction loss could be viewed as $\min_{\theta, \theta'} \mathfrak{L}_{\text{reconstruction}}$.

3.4. Multi-class classification loss

In this section, we want to utilize an appropriate classification model to reduce the cost between calculating assignments and labels learned by deep neural networks. Defining a classification head $h : \mathbb{R}^d \to \mathbb{R}^k$, which transform the feature vector into a class score vector. Denoting the set $\{\ell_1, \ell_2, ..., \ell_M\} \in \{1, 2, ..., k\}$ be the pre-assigned labels. The class score is given to class probabilities via softmax operator as $p(\ell = .|) = \text{softmax}(h \circ \Phi(x_i, \theta))$. therefore, the loss for minimizing the average cross entropy is given as:

$$\mathbb{E}(p|x_1, x_2, ..., x_M) = -\frac{1}{M} \sum_{i=1}^{M} \log \text{p}(x_i|\Phi(x_i, \theta)) \tag{5}$$

If the posterior distribution $q(x|\Phi(x_i, \theta))$ is set to be deterministic, then another way to express this equation is:

$$\begin{aligned}\mathfrak{L}_{\text{classification}} &= E(p(\theta), q(\theta)) \\ &= -\frac{1}{M} \sum_{i=1}^{M} \sum_{j=1}^{k} q(x|\Phi(x_i, \theta)) \log \text{p}(x|\Phi(x_i, \theta))\end{aligned} \tag{6}$$

If we assume more that each label is assigned uniformly, and each data point can take only one label, then the optimization problem for the above formula with constraints becomes:
$\theta E(p(\theta), q(\theta)) q(x|\Phi(x_i, \theta) \in \{0, 1\} \sum_{i=1}^{M} q(x|\Phi(x_i, \theta) = \frac{M}{k}$.
The above problem could be viewed as the form of optimal transport problem with binary constraints. The first constraint specifies that the function $q(x|\Phi(x_i, \theta))$, often used in optimization problems with binary variables, must take on only the values 0 and 1. The second constraint specifies that the sum of the function $q(x|\Phi(x_i, \theta))$ over all M data points must equal $\frac{M}{k}$. This constraint is often used in optimization problems with a balance condition, where the sum of the variables must be a certain value. There are several optimization strategies that can be used to tackle this problem, including gradient descent and the primal-dual interior point method. The specific algorithm used will depend on the characteristics of the objective function and the constraints, as well as the desired accuracy and computational efficiency. The above problem that could be solved in linear time by using a variety of optimization algorithms, such as gradient descent or a primal-dual interior point method [4].

4. Experiment Results

We consider around 10 sample data sets that extracted from UCR time series database [12]. There is a standard train/test split for each data set. The statistics of the benchmark time series datasets could be found in the appendix section. In fact, we will compare our results to those of DTCR model and the K means methods. The hidden size of the encoder part is $[100, 50, 50]$, while the regularization coefficient λ is chosen in the set $\{1, 1e-1, 1e-2, 1e-3\}$. The following table indicates the improving performance during the learning process of our deep spectral clustering model, and we also compare it to the DTCR model in [23]. The recorded results are extracted from the epochs 0, 30 and 50 respectively. The experiment is taken 5 times consecutively, and we just consider the average values overall.

Model evaluation: Metrics for assessing clustering output may be external or internal. For the external measure, we refer the the Rand Index (RI), Adjusted Rand Index (ARI), Adjusted Mutual Information (AMI), Fowlkes Mallows index (FMS), Homogeneity, and Completeness.

- **Rand Index:** The RI index refers to the similarities between the partitions of the clustering algorithms and the data set underlying structure.

$$RI = \frac{\text{TP} + \text{TN}}{n(n-1)/2}$$

- **Adjusted Rand Index:** The ARI index is a variant of the Rand Index (RI), which count the number of point pairings that are either in the same cluster or different clusters in both clusterings.

$$ARI = \frac{\text{RI} + \text{Expected(RI)}}{\text{Max(RI)} - \text{Expected(RI)}}$$

- **Normalized Mutual Information :** The NMI index measure of the mutual dependence between two random variables. It is a standardized version of the mutual information, which measures how much knowledge one random variable has about another.

$$NMI = \frac{\sum_{i=1}\sum_{j=1} N_{ij}\log(\frac{N.N_{ij}}{|G_i||A_j|})}{\sqrt{(\sum_{i=1}|G_i|\log\frac{|G_i|}{N})(\sum_{j=1}^{M}|A_j|\log\frac{A_j}{N})}}$$

For the internal measure, we could consider the Davies–Bouldin index, Calinski–Harabasz index, Silhouette score, the I-index, and sum of square errors (SSE).

- **Calinski–Harabasz index:** The CH index measures how compact the clusters are by calculating the distances between the points and centroids of each cluster.

Let's denote $\mu_k = \frac{1}{|I_k|}\sum_{i\in I_k} x^i$ be the average point of the group k, and $\mu = \frac{1}{N}\sum_{i=1}^{N} x^i$ be the center of the data set. The Calinski-Harabasz Index is introduced as

$$CHI = \frac{(N-K)B}{(K-1)\sum_{k=1}^{K} W_k}$$

- **Davies–Bouldin index:** The DB index measures the average similarity between any two clusters and their nearest neighbors.

where B be the intergroup variance of formula $B = \sum_{i=1}^{M}|I_i|\|\mu_i - mu\|^2$ and $W_i = \sum_{k\in I_i}$ be the intragroup variance. Moreover, let us defining $\bar{\gamma}_i = \frac{1}{|I|_i}\sum_{k\in I_i} d(x^k, \mu_i)$ be the average distance between each point of the set I_i to its groups center, the Davies-Bouldin Index is defined as:

$$\text{DBI} = \frac{1}{M}\sum_{i=1}^{M}\max_{i'\neq i}\left(\frac{\bar{\gamma}_i + \bar{\gamma}_{i'}}{d(\mu_i, mu_{i'})}\right)$$

We list all the RI and NMI comparing results for all 36 times series data sets between DSTR and DTCR model as the following:

Table 1. The RI comparisons during the epochs 0, 30 and 50.

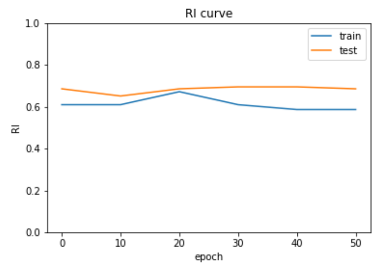
Dataset	DSTR	DTCR
Arrow	0.5429 /0.5698 /0.5698	0.4126 /0.4952/ 0.5717
Beef	0.5816 /0.6299 /0.6299	0.4828 /0.5942 /0.6778
Beetle.Fly	0.5579 /0.5579 /0.5589	0.5403 /0.6639/ 0.7640
Bird.Chicken	0.4789 /0.4789 /0.4789	0.4876 /0.5980 /0.6870
Car	0.6487 /0.6627 /0.6362	0.4503 /0.5532 /0.6235
ChlorineConcentration	0.5284 /0.5324 /0.5313	0.3216 /0.3996 /0.4632
Coffee	0.4814 /0.5238 /0.5476	0.5600 /0.6605 /0.7665
DiatomsizeReduction	0.7000 /0.7250 /0.7833	0.5837 /0.7059 /0.8235
Distphaloutl.agegroup	0.7579 /0.7642 /0.7589	0.4701 /0.5801 /0.6614
Distphaloutl.correct	0.5076 /0.5014 /0.5068	0.3658 /0.4526 /0.5118
ECG 200	0.5404/ 0.5604/ 0.5604	0.3987/ 0.4962/ 0.5557
ECGFiveDays	0.5968/ 0.5257/ 0.5573	0.5783/ 0.7029/ 0.8035
GunPoint	0.4931/ 0.4931/ 0.6278	0.3839/ 0.4712/ 0.5342
Ham	0.5005/ 0.4964/ 0.4964	0.3222/ 0.3961/ 0.4582
Herring	0.5000/ 0.5417/ 0.5417	0.3459/ 0.4191/ 0.4869
Lighting2	0.5277/ 0.6023/ 0.5729	0.3555/ 0.4284/ 0.4900
Meat	0.7644/ 0.6740/ 0.6870	0.5863/ 0.7254/ 0.8190
Mid.phal.outl.agegroup	0.7862/ 0.5882/ 0.6014	0.4792/ 0.5763/ 0.6508
Mid.phal.outl.correct	0.4999/ 0.5090/ 0.5095	0.3369/ 0.4110/ 0.4837
Mid.phal.TW	0.7742/ 0.8037/ 0.8507	0.5183/ 0.6316/ 0.7195
Mote.Strain	0.4737/ 0.4947/ 0.4947	0.4611/ 0.5605/ 0.6492
OSU.Leaf	0.5634/ 0.5878/ 0.6779	0.4645/ 0.5747/ 0.6628
Plane	0.8982/ 0.8668/ 0.8985	0.5752/ 0.7034/ 0.8019
Proxphaloutl.ageGroup	0.7659/ 0.7473 / 0.7441	0.4895/ 0.5895/ 0.6641
ProxphalTW	0.7980/ 0.8046/ 0.8457	0.5416/ 0.6594/ 0.7607
SonyAIBORobotSurface	0.6053/ 0.6053/ 0.6053	0.5264/ 0.6391/ 0.7238
SonyAIBORobotSurfaceII	0.4872/ 0.5385/ 0.6011	0.5015/ 0.6269/ 0.7025
SwedishLeaf	0.5933/ 0.7279/ 0.8892	0.5570/ 0.6731/ 0.7699
Symbols	0.8400/ 0.8100/ 0.8433	0.5520/ 0.6723/ 0.7808
TwoSegmentation1	0.4923/ 0.4923/ 0.5192	0.3396/ 0.4145/ 0.4736
TwoSegmentation2	0.5111/ 0.4921/ 0.4921	0.4988/ 0.6084/ 0.7065
TwoPatterns	0.5534/ 0.6423/ 0.6772	0.4210/ 0.5099/ 0.5987
TwoLead.ECG	0.5573/ 0.4783/ 0.5573	0.4279/ 0.5059/ 0.5814
Wafer	0.4899/ 0.5677/ 0.6881	0.4420/ 0.5314/ 0.6085
Wine	0.5038/ 0.5038/ 0.5038	0.3779/ 0.4589/ 0.5330
WordsSynonyms	0.5561/ 0.6672/ 0.7772	0.5406/ 0.6582/ 0.7572
Overall Average	**0.5740 /0.6046 /0.6339**	**0.4638 /0.5652/ 0.6080**

5. Conclusions

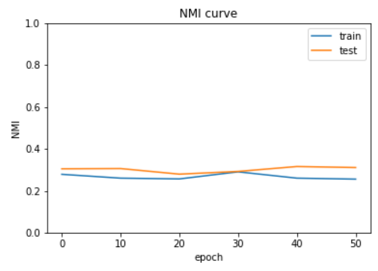
In this paper, we introduce a new method for spectral clustering data using deep learning techniques. It

Table 2. The NMI comparisons during the epochs 0, 30, and 50.

Dataset	DSTR	DTCR
Arrow	0.2865 /0.1804 /0.1804	0.3317 /0.4119 /0.4687
Beef	0.2001 /0.3060 /0.4933	0.3371 /0.4025 /0.4650
Beetle.Fly	0.1263 /0.1263 /0.1263	0.4566 /0.5640 /0.6436
Bird.Chicken	0.0107 /0.0107 /0.0371	0.3189 /0.3911 /0.4483
Car	0.1723 /0.1419 /0.1776	0.3023 /0.3676 /0.4244
ChlorineConcentration	0.5284 /0.5324 /0.5313	0.0279 /0.0333 /0.0385
Coffee	0.5283 /0.5283 /0.5476	0.4892 /0.6055 /0.7098
DiatomsizeReduction	0.7000 /0.7250 /0.7833	0.5666 /0.6864/ 0.7810
Distphaloutl.agegroup	0.7579 /0.7642 /0.7589	0.2730 /0.3379 /0.3870
Distphaloutl.correct	0.5076 /0.5014 /0.5068	0.0707 /0.0851 /0.0973
ECG 200	0.6564/ 0.7545/ 0.7922	0.2218/ 0.2679/ 0.3121
ECGFiveDays	0.7742/ 0.7778/ 0.8024	0.4856/ 0.5874/ 0.6802
GunPoint	0.0633/ 0.0642/ 0.0645	0.2526/ 0.3137/ 0.3562
Ham	0.0664/ 0.0792/ 0.0912	0.0592/ 0.0730/ 0.0831
Herring	0.1134/ 0.1783/ 0.1965	0.1354/ 0.1690/ 0.1965
Lighting2	0.1267/ 0.1545/ 0.1667	0.1375/ 0.1640/ 0.1923
Meat	0.0576/ 0.0698/ 0.0789	0.5797/ 0.7145/ 0.8185
Mid.phal.outl.agegroup	0.3231/ 0.3756/ 0.3878	0.2799/ 0.3474/ 0.4034
Mid.phal.outl.correct	0.0770/ 0.0923/ 0.0943	0.0689/ 0.0841/ 0.0954
Mid.phal.TW	0.0243/ 0.0345/ 0.0377	0.3303/ 0.3965/ 0.4597
Mote.Strain	0.4467/ 0.4998/ 0.5192	0.2459/ 0.3009/ 0.3458
OSU.Leaf	0.1334/ 0.1855/ 0.2550	0.1563/ 0.1969/ 0.2235
Plane	0.7664/ 0.8110/ 0.9354	0.5585/ 0.7019/ 0.8039
Proxphaloutl.ageGroup	0.0442/ 0.0534/ 0.0632	0.3450/ 0.4256/ 0.4753
ProxphalTW	0.4536/ 0.6192/ 0.7453	0.4186/ 0.5151/ 0.5885
SonyAIBORobotSurface	0.4738/ 0.5534/ 0.6635	0.4597/ 0.5627/ 0.6562
SonyAIBORobotSurfaceII	0.4175/ 0.4999/ 0.5634	0.3699/ 0.4574/ 0.5243
SwedishLeaf	0.4673/ 0.5362/ 0.6183	0.4017/ 0.5034/ 0.5753
Symbols	0.0652/ 0.6789/ 0.7144	0.5395/ 0.6656/ 0.7470
TwoSegmentation1	0.2032/ 0.2654/ 0.3183	0.1869/ 0.2338/ 0.2633
TwoSegmentation2	0.2012/ 0.2486/ 0.3143	0.2340/ 0.2841/ 0.3243
TwoPatterns	0.2165/ 0.2976/ 0.3232	0.2836/ 0.3470/ 0.3992
TwoLead.ECG	0.3562/ 0.3981/ 0.4682	0.2770/ 0.3419/ 0.3883
Wafer	0.7680/ 0.8611/ 0.8785	0.0135/ 0.0163/ 0.0186
Wine	0.1895/ 0.2461/ 0.2879	0.1732/ 0.2199/ 0.2458
WordsSynonyms	0.3162/ 0.3782/ 0.4527	0.3271/ 0.3953/ 0.4593
Overall Average	**0.3212 /0.3786 /0.4348**	**0.2977 /0.3659/ 0.4194**



[]



[]

Figure 2. The model evaluation for beef data through the epochs: (a) Rand Index curve, (b) Normalized Mutual Information curve.

entails utilizing spectral clustering on the learnt representation to produce the final clusters after training a neural network model to learn a representation of the data that is appropriate for clustering. Deep spectral clustering combines the ability of deep learning to learn powerful feature representations from data with the ability of spectral clustering to handle nonlinear relationships in the data. It has been used with a variety of data types, including texts, time series, and images. Deep spectral clustering has the benefit of being able to automatically extract features from the data, which can be more efficient than doing it manually. Additionally, it can handle high-dimensional data, which might be difficult for conventional clustering approaches to manage, such as photos with lots of pixels or texts with lengthy word sequences. However, deep spectral clustering also has some challenges, such as the need for large amounts of labeled data for training and the risk of overfitting if the model is too complex. It is important to carefully select the model architecture and the training procedures in order to achieve good performance on the clustering task.

Acknowledgement

This research is done by a finanical aid from Analgesia Institut. Address: 28 Place Henri Dunant, 63000 Clermont-Ferrand, France. The authors also would like to thank professor Violaine Antoine and professor Engelbert Mephu Nguifo for useful discussions.

References

[1] A. Alqahtani, M. Ali, X. Xie, and M. Jones "Deep Time-Series Clustering: A Review: a review", *Electronics 2*, vol. 10, no.23, 2021.

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

[3] A. Ng, M. Jordan, and Y. Weiss "On spectral clustering: analysis and an algorithm", *Proceedings of the 14th International Conference on Neural Information Processing Systems:*, pp. 849-856, Jan. 2001.

[4] Y. Asano, C. Rupprecht, and A. Vedaldi, "Self-labelling via simultaneous clustering and representation learning", *ICLR Conference Paper*, 2020.

[5] F. Bach and M. Jordan "Learning Spectral Clustering", *Proceedings of the 16th International Conference on Neural Information Processing Systems:*, 2003.

[6] D. Bo, X. Wang, C. Shi, M. Zhu, E. Lu, and P. Cui, "Structural deep clustering network", *Proceedings of The Web Conference*, pp 1400–1410, 2020.

[7] S. Chang, Y. Zhang, W. Han, M. Yu, X. Guo, W. Tan, X. Cui, M. Witbrock, M. Johnson, and T. Huang, "Dilated recurrent neural networks", *Advances in Neural Information Processing Systems*, pp. 77–87, 2017.

[8] J. Chang, L. Wang, G. Meng, S. Xiang, and C. Pan "Deep Adaptive Image Clustering", *ICCV*, 2017.

[9] J. Chien, K. Kuo, "Variational Recurrent Neural Networks for Speech Separation", *Interspeech*, 2017.

[10] M. Corduas and D. Piccolo, "Time series clustering and classification by the autoregressive metric", *Computational*

Statistics Data Analysis, vol. 52, no.4, pp. 1860-1872, Jan. 2008.

[11] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition", *IEEE*, 1998.

[12] H. Dau, A. Bagnall, K. Kamgar, C. Yeh, S. Gharghabi, C. Ratanamahatana, and E. Keogh "The UCR Time Series Archive", *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 6, pp. 1293-1305, Nov. 2019.

[13] K. Dizaji, A. Herandi, C. Deng, W. Cai, and H. Huang H, "Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization", *Proceedings of the IEEE international conference on computer vision*, pp 5736–5745, 2017.

[14] H. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P. Muller, "Deep learning for time series classification: a review", *Data Mining and Knowledge Discovery volume*, vol. 33, pp. 917-963, 2019.

[15] N.El Malki, "New partition-based and density-based approaches for improving clustering", thèse de doctorat, *https://theses.hal.science/tel-03716266/document*, 2021.

[16] J. Franceschi, A. Dieuleveut, and M. Jaggi, "Unsupervised scalable representation learning for multivariate time series", *Advances in Neural Information Processing Systems*, pp 4652–4663, 2019.

[17] X. Guo, X. Liu, E. Zhu, and J. Yin, "Deep Clustering with Convolutional Autoencoders" *In Proceedings of the International Conference on Neural Information Processing*, pp. 373–382, 2017.

[18] X. Guo, L. Gao, X. Liu and J. Yin, "Improved deep embedded clustering with local structure preservation", *IJCAI*, pp 1753–1759, 2017.

[19] R. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, P. Bachman, A. Trischler, and Y. Bengio, "Learning deep representations by mutual information estimation and maximization", *ICLR*, 2019.

[20] P. Huang, Y. Huang, W. Wang, L. Wang, "Deep embedding network for clustering", *In Proceedings of the International Conference on Pattern Recognition*, pp. 1532–1537, 2014.

[21] A. Krizhevsky, I. Sutskever, G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks", *Advances in Neural Information Processing Systems 25 (NIPS)*, 2012.

[22] B. Lafabregue, J. Weber, P. Gancarski, and G. Forestier, "End-to-end deep representation learning for time series clustering: a comparative study", *Data Mining and Knowledge Discovery volume*, vol. 36, pp. 29-81, 2022.

[23] Q. Ma and J. Zheng and S. Li and G. Cottrel, "Learning representations for time series clustering", *Advances in neural information processing systems*, page 3781-3791, 2019.

[24] Q. Ma, C. Chen, S. Li, and G. Cottrell, "Learning Representations for Incomplete Time Series Clustering" *AAAI Technical Track on Machine Learning*, Vol. 35, No. 10, 2021.

[25] M. Law, R. Urtasun and R. Zemel, "Deep spectral clustering learning," *Proceedings of the 34th International Conference on Machine Learning*, vol 70, pp. 1985–1994, 2017.

[26] N. Madiraju, S. Sadat, D. Fisher, and J. Karimabadi, "Deep Temporal Clustering: Fully Unsupervised Learning of Time-Domain Features", *arXiv 2018*, *arXiv:1802.01059*, 2018.

[27] E. Min, X. Guo, Q. Liu, G. Zhang, J. Cui and J. Long, "A Survey of Clustering With Deep Learning: From the Perspective of Network Architecture," *IEEE Access*, vol. 6, pp. 39501-39514, 2018

[28] A. Sammani, A. Bagheri, and W. Asselbergs, "Automatic multilabel detection of ICD10 codes in Dutch cardiology discharge letters using neural networks," *nature portfolio*, 2021.

[29] K. Simonyan, A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *ICLR*, 2015.

[30] P. Tzirakis, M. Nicolaou, B. Schuller, and S. Zafeiriou, "Time-series Clustering with Jointly Learning Deep Representations, Clusters and Temporal Boundaries", *14th IEEE International Conference on Automatic Face Gesture Recognition* , 2019.

[31] U. Shaham, K. Stanton, H. Li, B. Nadler, R. Basri and Y. Kluger, "SpectralNet: Spectral clustering using deep neural networks," *6th International Conference on Learning Representations*, 2018.

[32] J. Xie, R. Girshick and A. Farhadi, "Unsupervised deep embedding for clustering analysis", *International conference on machine learning*, pp. 478–487, 2016.

[33] H. Zha, X. He, C. Ding, H. Simon, and M. Gu "Spectral relaxation for K-means clustering", *Proceedings of the 14th International Conference on Neural Information Processing Systems:*, pp. 1057-1064, Jan. 2001.

[34] Y. Zhao, Y. Yuan, F. Nie, and Q. Wang, "Spectral clustering based on iterative optimization for large-scale and high-dimensional data", *Neurocomputing*, vol. 318, pp. 227-235, Jan. 2018.

[35] F. Wang and C. Zhang, "Spectral Clustering for Time Series", *Pattern Recognition and Data Mining*, pp. 345-354, 2005.

[36] Z. Wang Z, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline", *International joint conference on neural networks (IJCNN)*, IEEE, pp 1578–1585, 2017.

[37] X. Yang, C. Deng, F. Zheng, J. Yan, and W. Liu, "Deep Spectral Clustering Using Dual Autoencoder Network", *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[38] B. Yang, X. Fu, N. Sidiropoulos, M. Hong "Towards K-means-friendly Spaces: Simultaneous Deep Learning and Clustering", *Proceedings of the 34th International Conference on Machine Learning PMLR*, vol 70 pp. 3861-3870, 2017.

[39] N. Yulita, M. Ivan Fanany, A. Murni Arymuthya, "Bi-directional Long Short-Term Memory using Quantized data of Deep Belief Networks for Sleep Stage Classification", *Procedia Computer Science*, vol 116 pp. 530-538, 2017.

[40] S. Zhang, H. Tong, J. Xu, and R. Maciejewski, "A Comprehensive Survey on Deep Clustering: Taxonomy, Challenges, and Future Directions", *Computational social network*, pp 06–11, 2019.

[41] S. Zhou, H. Xu, and Z. Zheng, "Graph convolutional networks: a comprehensive review", *ArXiv*, https://arxiv.org/abs/2206.07579, 2022.

[42] Y. Ren, J. Pu, Z. Zhang, J. Xu, X. Pu, S. Yu, and L. He, "Deep Clustering: A Comprehensive Survey", *ArXiv*, https://arxiv.org/abs/2210.04142, 2022.