Information Sciences 639 (2023) 119009

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Multi-feature based network for multivariate time series classification

Mingsen Du [a], Yanxuan Wei [a], Xiangwei Zheng [a,b], Cun Ji [a,b,*]

[a] *School of Information Science and Engineering, Shandong Normal University, Jinan, China*
[b] *Shandong Provincial Key Laboratory for Distributed Computer Software Novel Technology, Jinan, China*

ARTICLE INFO

Keywords:
Multivariate time series classification
Multi-feature
Graph neural networks

ABSTRACT

Multivariate time series classification is widely available in several areas of real life and has attracted the attention of many researchers. In recent years, many multivariate time series classification methods have been proposed. However, existing multivariate time series classification methods focus only on local or global features and usually ignore the spatial dependency features among multiple variables. For this, we propose a multi-feature based network (MF-Net). First, MF-Net uses the global-local block to acquire local features through the attention-based mechanism. Next, the sparse self-attention mechanism captures global features. Finally, MF-Net integrates the local features and global features to capture the spatial dependency features using the spatial-local block. Therefore, we can mine the spatial dependency features of multivariate time series while incorporating both local and global features. We conducted experiments on UEA datasets and the experimental results showed that our method achieved performance competitive with that of state-of-the-art methods.

1. Introduction

Time series classification aims to assign labels to time series through supervised learning [1]. Time series classification is usually categorized into two types: univariate time series classification and multivariate time series (MTS) classification. In real life, multiple sensors often work together. Therefore, the MTS is widely used daily. For example, wearable mobile devices collect various activity data of the human body during movement to improve people's lives [2]. In this paper, we focus on MTS classification.

In MTS classification, it is crucial to fully use the extracted features. The commonly used features include local, global, and spatial dependency features. Local features are the local temporal features [1] (i.e., shapelet [3]) of univariate time series, as shown in Fig. 1. Global features can be regarded as the relationships among local features [4], as shown in Fig. 1. Additionally, spatial dependency features are the complex dependencies among the dimensions of MTS, which can be obtained using quantitative criteria, for example, similarity (e.g., distance [5]), correlation (e.g., Pearson correlation [6]), and causality relationship [7], as shown in Fig. 1. For example, in traffic flow, traffic on one street is easily influenced by neighboring streets; thus, the interactions among streets develop the spatial dependency. Extracting effective features from MTS has become a key challenge in MTS classification [8].

[*] Corresponding author at: School of Information Science and Engineering, Shandong Normal University, Jinan, China.
E-mail addresses: MingsenDu@163.com (M. Du), wyx_314159@qq.com (Y. Wei), xwzhengcn@163.com (X. Zheng), jicun@sdnu.edu.cn (C. Ji).

https://doi.org/10.1016/j.ins.2023.119009
Received 12 January 2023; Received in revised form 20 April 2023; Accepted 21 April 2023

Available online 26 April 2023

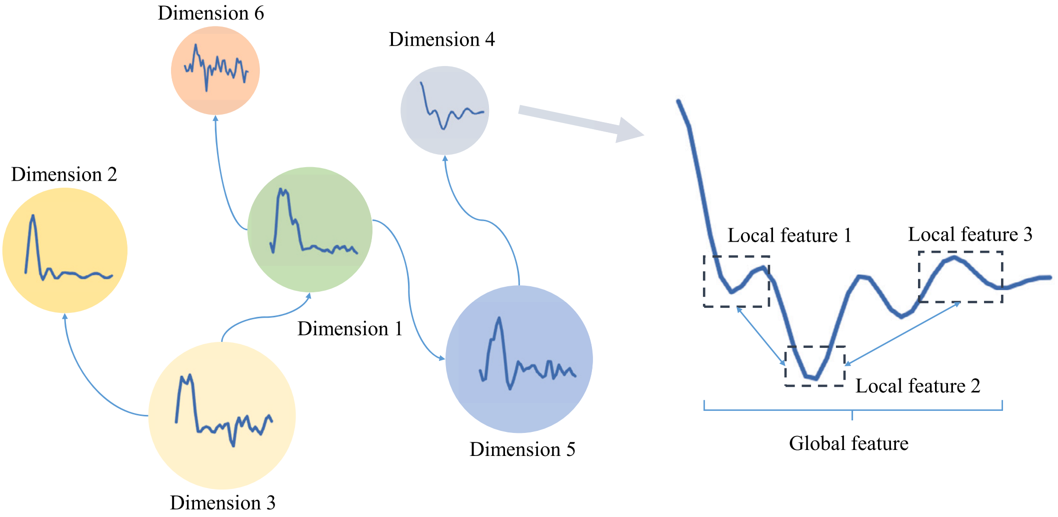


Fig. 1. Local, global, and spatial dependency features of MTS. Lines with arrows represent the spatial dependencies among MTS. The local features contained within the dashed box and the interrelationship among local features are the global features.

In response to the challenge, many MTS classification methods have been proposed recently. These methods can be broadly classified into traditional machine learning-based methods and deep learning-based methods. The former, such as the hidden state conditional random field [9] and hidden unit logistic model [10], mainly exploits the need for tedious feature engineering or data pre-processing efforts. The latter aims to unfold the internal representational hierarchy of the time series, which is beneficial for mining the intrinsic connections of representations [11]. Compared with traditional machine learning-based classification methods, deep learning-based methods require less domain expertise and achieve higher accuracy.

In recent years, many MTS classification methods based on deep learning have been proposed. However, they still encounter the following three challenges: (1) Effective local feature selection: Local features, such as shapelets [3], can represent a class in some sense maximally. Ye and Keogh [3] demonstrated that the nearest neighbor approach based on shapelets can provide an effective and interpretable classification decision [12]. How to obtain the distinguishing local features is a fundamental challenge for MTS classification. (2) Effective and efficient global feature extraction: In contrast to local feature extraction, global feature extraction is designed to capture the relationships among previously extracted local features. This means that relationship extraction compensates for the loss of representation that is neglected by the corresponding local feature network. Recently, in several studies, researchers have demonstrated the significance of global features for MTS classification [4,13,14]. (3) Spatial dependency features: The majority of existing MTS classification methods directly use the features of MTS. However, they usually ignore the complex hidden dependency among the dimensions of MTS. How to obtain the spatial dependency features is another key challenge for MTS classification.

To address the several challenges mentioned above, we propose a multi-feature based network (MF-Net). MF-Net consists of two parts: the global-local block and spatial-local block. First, the global-local block uses the convolutional block attention module (CBAM) block with an attention mechanism to focus on more differentiated information and the sparse self-attention (SSA) block with the SSA mechanism to mine the global features while reducing computational complexity. Second, the spatial-local block successfully extracts spatial dependency features using a mix-hop block, while also taking into account the refinement of local features. Therefore, MF-Net successfully models the local, global, and spatial dependency feature correlations.

The following are the main contributions of this paper:

1. We design a novel MTS classification network architecture to exploit local features, global features, and spatial dependency features, which corresponds to various blocks of MF-Net. First, the local feature-based module uses an attention mechanism to focus on more distinguishing information. Second, the global feature-based module uses the SSA mechanism to mine global information while reducing computational complexity. Finally, the spatial feature-based module leverages the graph convolution network to extract spatial information.
2. We apply visualization methods to investigate the interpretability of the features learned by MF-Net on various datasets. We use a Grad-CAM-based heat map and graph structure visualization method to visualize local, global, and spatial dependency features, and study their joint contributions to MTS classification.
3. We conducted extensive experiments to demonstrate the effects of our method on a variety of datasets from UEA datasets.[1] The experiments included comparative and sensitive experiments, which verified the superiority of our method compared with other methods and the effectiveness of the blocks of MF-Net. The experimental results proved that MF-Net improved MTS classification accuracy.

[1] The datasets are available at http://timeseriesclassification.com.

The remainder of this paper consists of the following sections: Section 2 covers related work regarding multiple leading-edge methodologies. In Section 3, we introduce the details of the proposed methodology. In Section 4, we report and discuss the experimental results. Finally, in Section 5, we conclude this paper.

2. Related work

MTS classification methods can be divided into traditional machine learning-based methods and deep learning-based methods.

2.1. Traditional machine learning-based methods

After several years of development, traditional machine learning-based MTS classification methods can be mainly categorized into distance-based and feature-based methods.

Distance-based methods use distance functions to calculate the similarity among time series and are combined with the 1-nearest neighbor (1NN) method to predict class labels. In a large number of studies, researchers have viewed dynamic time warping (DTW) [15] as the most outstanding distance-based method. Shokoohi-Yekta et al. [16] adopted two strategies for using DTW for multivariate problems. They discussed the idea of selecting independent and dependent DTW. Additionally, they proposed an adaptive program in which the decision regarding which distance to choose depends on a threshold mined from the training data and the decision is made depending on the score function.

Feature-based methods use various features to classify time series. Ye and Keogh [17] pioneered the concept of shapelets. Shapelets, as one of the most popular feature-based methods, are discriminatory sub-series that are easily interpretable. Karlsson et al. [18] proposed a generalized random shapelet forest, which uses randomly selected shapelets within a forest of decision trees. Schäfer and Leser proposed WEASEL+MUSE [19], which extracts and filters multivariate features per window and dimension of MTS by encoding a context message into each small and discriminative feature for MTS classification. Li and Tang [20] proposed a new semi-supervised local feature selection method, which selects different feature subsets for different classes. Using these methods, class-specific feature candidates are selected by learning the importance of features using different methods for each class separately.

2.2. Deep learning-based methods

Despite the effectiveness of traditional machine learning-based methods, they can only extract some shallow features using heuristic and hand-crafted methods [21]. By contrast, deep learning-based approaches tend to overcome these limitations and extract deep high-level representations.

2.2.1. Deep learning-based methods based on local features

Deep learning-based methods classify MTS based on local features. Chen et al. [22] proposed an end-to-end network designed with convolutional neural network (CNN)-based multi-scale convolutional modules to enhance the robustness of deep learning models. The network stacks multiple modules that can generate different ranges of receptive domains to obtain features at different scales. Fawaz et al. [23] proposed InceptionTime, which achieves high accuracy by building on residual networks (ResNet) to incorporate inception. Xiao et al. [4] presented RTFN, which contains a temporal feature network as a local feature extraction network to capture adequate local features and a long short-term memory (LSTM)-based attention network as a relation extraction network to determine intrinsic relationships. Chen et al. proposed TFDM [24], which adopts multilevel discrete wavelet decomposition to learn the nonlinear features of MTS. Zheng et al. proposed MCDCNN [25], which uses CNN to learn the features for each dimension of MTS in parallel. Zhang et al. proposed TapNet [26] to learn low-dimensional features extracted from MTS data using a multilayer CNN. However, these methods ignore the global features of time series.

2.2.2. Deep learning-based methods based on global features

Some deep learning-based methods classify MTS based on the global features of time series. Liu et al. [13] proposed GTN, which classifies MTS through channel-wise and step-wise correlations. Karim et al. [27] proposed MLSTM-FCN, which uses LSTM to capture global features and fully convolutional networks (FCN) to extract local features. Chen et al. [14] proposed DA-Net, which uses the SSA layer to mine local-global features.

2.2.3. Deep learning-based method based on spatial features

In addition to local and global features, spatial dependency features can also be used for MTS classification. Graph neural networks (GNN) [28] are often used for mining spatial dependency features. As a unique type of data, graphs describe the relationships among different entities. In recent years, researchers have found that it is promising to model MTS using GNN [29]. Each univariate time series of MTS can be considered as nodes of a graph, whereas the interrelationships among MTS can be regarded as edges. The hidden information in MTS data is stored in the graph structure, which can then be processed using a GNN [6]. TEGNN [29] regards each variable as a graph node and represents the casual relationship among variables as edges. Wu et al. [30] proposed MTGNN to capture spatial and temporal dependencies. MTPool [5] combines a GNN, encoder, and decoder-based variational graph pooling block to create adaptive centroids for graph coarsening. MTHetGNN [6] constructs a relation embedding module to explore the relations in both dynamic and static approaches for MTS. All the above methods can effectively model the MTS spatial dependency with the help of a GNN.

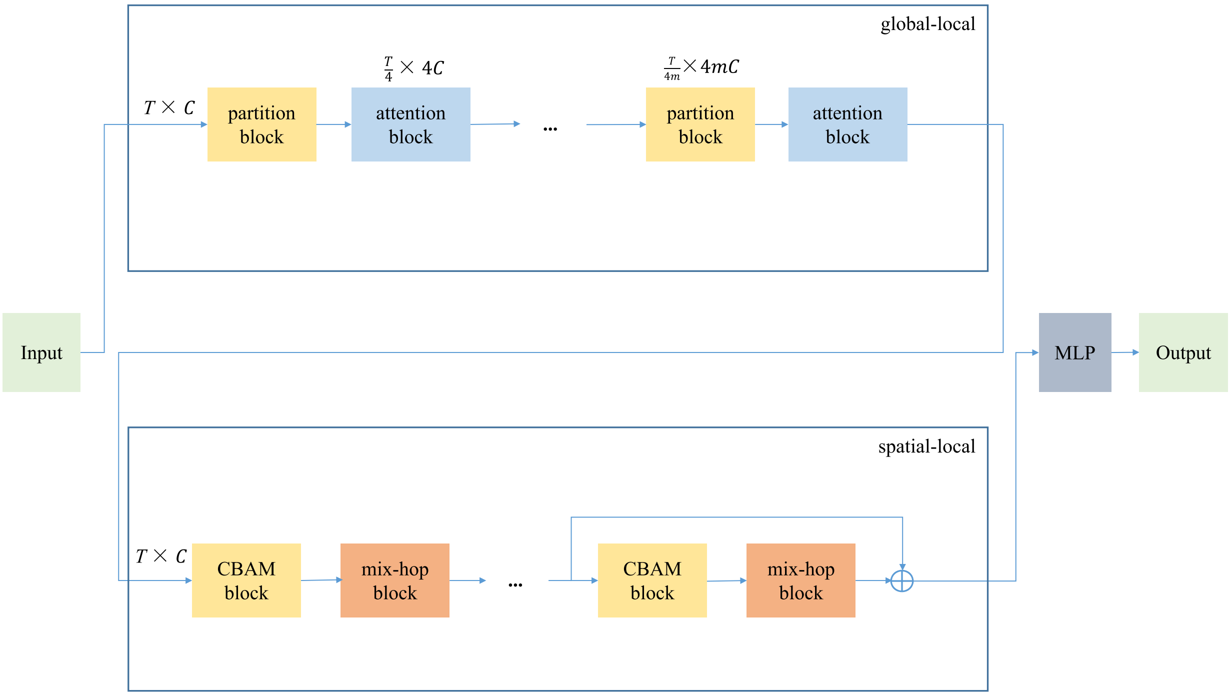


Fig. 2. Overall structure of MF-Net.

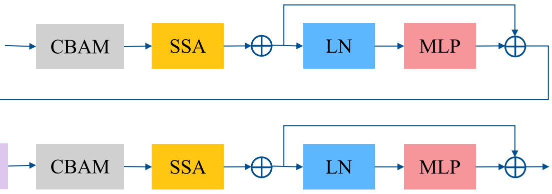


Fig. 3. Attention block.

The above methods achieve satisfying results using a combination of local features and global features, or a combination of local features and spatial features, but they neglect the combination of the three features. Our study bridges the gap as the first comprehensive study in which local, global, and spatial dependency features are used for MTS classification.

3. Proposed method

In this section, first, we briefly introduce the overall MF-Net framework. Then we provide details of the attention block and mix-hop block, and the specifics of the completed classification task.

3.1. Overview

The overall structure of MF-Net is shown in Fig. 2. MF-Net contains two major parts: the global-local block and spatial-local block.

The first part of MF-Net is the global-local block, which is used to extract global and local features. The first stage of the global-local block is the time partition block, which is used to obtain the time series embedding. As shown in Fig. 3, the attention block is the second stage of the global-local block and is used for extracting local features using the CBAM block and further capturing global features using the SSA block.

The second part of MF-Net is the spatial-local block, which is used to extract spatial dependency features. The first stage of the spatial-local block is the CBAM block, which performs local feature refinement. As shown in Fig. 7, the mix-hop block is the second stage of the spatial-local block. The mix-hop block consists of a graph learning layer used to adaptively learn the graph adjacency matrix and a graph convolution layer that aims to capture spatial dependency features. Finally, MTS classification is implemented using a multilayer perceptron (MLP).

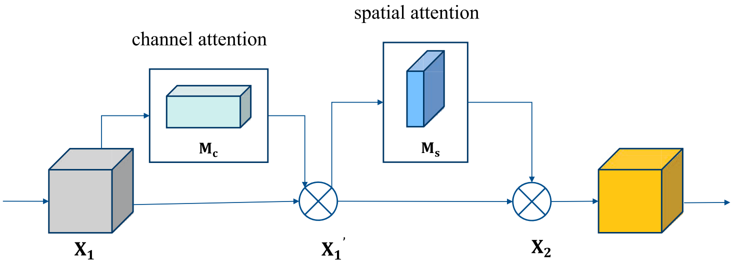


Fig. 4. CBAM block.

3.2. Global-local block

As shown in Fig. 2, the global-local block is used to extract global and local features. The time partition block, as the first stage of the global-local block, is used to obtain time series embedding. The attention block, as the second stage of the global-local block, aims to capture local distinguishing features and global features, as shown in Fig. 3.

3.2.1. Partition block

Partition blocks are used to split time series. First, four non-overlapping neighboring time stamps are used as time chunks to obtain $\frac{T}{4}$ time chunks, and then each time chunk is flattened and projected into a 4C-dimensional embedding. So, the output $\mathbf{X_1}$ of i-th partition block ($1 \le i \le m$, and m is the total number of partition blocks) is a tensor with dimensionality $\frac{T}{4i} \times 4iC$ (T is the length of the initial MTS, and C is the dimension of the initial MTS).

3.2.2. Attention block

As shown in Fig. 3, the attention block first goes through the CBAM block, which uses a spatial and channel attention mechanism to mine more distinguishing local features. Then it enters the SSA, which introduces sparse bias to capture global features efficiently. Finally, it enters a layer normalization (LN) layer in addition to an MLP layer. The above process is repeated twice in each attention block, with the difference that, the second time, a shifted layer [31] is added.

Some classical global feature extraction models, such as the recurrent neural network (RNN), are frequently used in MTS classification. However, the original RNN has the problem that it adopts a linear sequence structure to continuously gather input information from head to tail. Thus, the RNN is relatively inept at capturing the global features of time series, mainly because the backpropagation path is too long and thus prone to severe gradient disappearance, in addition to gradient explosion problems.

Thus, we use the transformer structure to extract global features. We use the transformer instead of the RNN or others because the transformer does not depend on past hidden states to mine the global features of previous tokens. Instead, it processes time series as a whole to allow parallel computation, decrease training time, and diminish performance degradation caused by global features. The transformer [32], as a typical feature extraction structure, achieves good performance in capturing global features with its overall time series processing capability and multi-head attention mechanism. Among transformer-based methods, the Swin transformer has the most typical structure. It introduces a hierarchical feature map and applies a window partitioning strategy to decrease the time complexity. It also adopts a shifted window mechanism that restricts the self-attention computation to non-overlapping local windows to enhance classification ability while allowing connectivity in the same manner as a cross window. Therefore, we use the Swin transformer structure to handle the costly computational complexity, feature interactions, and most importantly, the global features of MTS.

CBAM block Capturing effective local features is one of the fundamental steps in the MTS classification task. Some traditional local feature extraction methods enhance interpretability by selecting local candidate features to obtain local temporal features [1]. The CBAM block [33] obtains the attention map along double independent dimensions, that is, the channel dimension and spatial dimension. Additionally, it then multiplies the attention map by the input feature for adaptive feature refinement. $\mathbf{X_1}$ goes through the CBAM block, which is a module that aims to exploit the spatial and channel attention mechanisms and can focus more on distinguishing local features, as shown in Fig. 4. Additionally, we obtain the channel attention $\mathbf{X_2}$. The channel attention $\mathbf{X_2}$ of i-th attention block ($1 \le i \le m$, and m is the total number of attention blocks) is a tensor with dimensionality $\frac{T}{4i} \times 4iC$ (T is the length of the initial MTS, and C is the dimension of the initial MTS).

Step 1: Channel attention. There are three main processes to obtain the channel attention. First, information in the feature graph is aggregated using the average pooling and maximum pooling operations to generate two spatial context descriptors: $\mathbf{X_1}^c_{avg}$ and $\mathbf{X_1}^c_{max}$, which represent the average pooling features and maximum pooling features, respectively. Second, double descriptors are forwarded to a parameter sharing network that consists of an MLP. Finally, the output feature vectors are merged using the element-wise sum.

In sum, the channel attention $\mathbf{X}'_1$ in i-th attention block can be calculated by Eq. (1). In Eq. (1), $\mathbf{X_1}$ is the output of the i-th partition block (refer to Section 3.2.1). $\mathbf{X_1}^c_{avg}$ and $\mathbf{X_1}^c_{max}$ are results processed by $Avgpool$ and $Maxpool$, respectively. $\mathbf{W_1}$ and $\mathbf{W_0}$ are

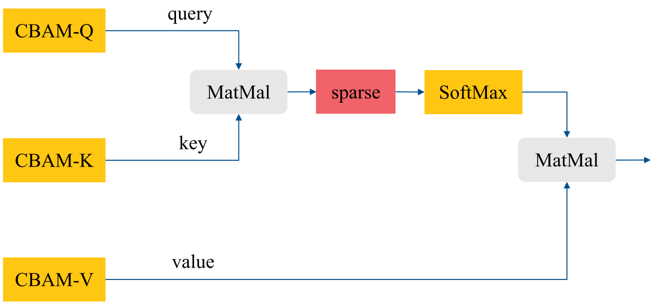


Fig. 5. SSA self-attention mechanism.

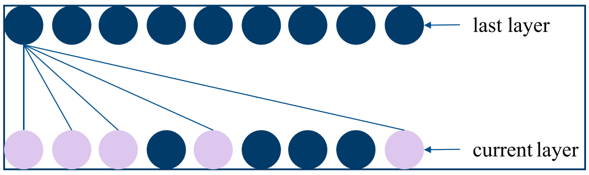


Fig. 6. Overview of the sparse self-attention mechanism. Sparse self-attention by allowing each cell of the last layer to only attend to the cells of the current layer with an exponential step size. Dark circles in the current layer do not participate in dot product operations, whereas bright circles participate in dot product operations.

the shared parameters of MLP operation. $\mathbf{X_1}_{avg}^{c}$, $\mathbf{X_1}_{max}^{c}$, $\mathbf{W_1}$, $\mathbf{W_0}$, and $\mathbf{X}'_1$ are all tensors with dimensionality $\frac{T}{4i} \times 4iC$ (T is the length of the initial MTS, and C is the dimension of the initial MTS).

$$\begin{aligned} \mathbf{X}'_1 &= sigmoid(MLP(AvgPool(\mathbf{X_1})) + MLP(MaxPool(\mathbf{X_1}))) \\ &= sigmoid(\mathbf{W_1}(\mathbf{W_0}(\mathbf{X_1}_{avg}^{c})) + \mathbf{W_1}(\mathbf{W_0}(\mathbf{X_1}_{max}^{c}))) \end{aligned} \tag{1}$$

Step 2: Spatial attention. The relationship in the feature space is used to generate a spatial attention map. Spatial attention is concerned with an information component that is complementary to channel attention. The average pooling and maximum pooling operations are first applied along the channel axis and then stitched together to generate a feature descriptor. Then a spatial attention feature map is generated using a one-dimensional standard CNN. The spatial attention in i-th attention block is computed as

$$\begin{aligned} \mathbf{X_2} &= sigmoid(f^a([AvgPool(\mathbf{X}'_1); MaxPool(\mathbf{X}'_1)])) \\ &= sigmoid(f^a([\mathbf{X}'^{s}_{\mathbf{1}avg}; \mathbf{X}'^{s}_{\mathbf{1}max}])), \end{aligned} \tag{2}$$

where f^a denotes a one-dimensional convolution operation with filter size a, $\mathbf{X}'_1$ is the result of Eq. (1), $\mathbf{X}'^{s}_{\mathbf{1}avg}$ and $\mathbf{X}'^{s}_{\mathbf{1}max}$ are results processed by $Avgpool$ and $Maxpool$. $\mathbf{X}'^{s}_{\mathbf{1}avg}$, $\mathbf{X}'^{s}_{\mathbf{1}max}$, and $\mathbf{X_2}$ are tensors with dimensionality $\frac{T}{4i} \times 4iC$.

SSA block Efficient global feature extraction is equally critical. After local features $\mathbf{X_2}$ are captured by the CBAM block, $\mathbf{X_2}$ is embedded in the SSA block to capture global features. However, the time complexity of the Swin transformer [31] for computing dot product pairs is $O(L^2)$; thus, the computational effort is extremely huge. The procedure for calculating the dot product is shown in Fig. 5.

The space complexities of the normal transformer and Swin transformer grow quadratically with length L of the time series. Moreover, time complexity $O(L^2)$ makes memory consumption huge [34]. Therefore, to compute the self-attention dot product more efficiently, we adopt a sparse strategy [35] and design an SSA block, which introduces sparse bias matrix M in the self-attention model, and a log-sparse mask, which only needs to compute the $O(logL)$ dot product of each cell in each layer, thus reducing the computational complexity from $O(L^2)$ to $O(logL)$, as shown in Fig. 6.

When the transformer layers are stacked, the heads of each layer may focus on features that correspond to different types [35]. For example, for time series generated each month, the heads of each layer focus on the features of a particular week. Therefore, we restrict the cells between two adjacent layers, that is, we allow each layer cell to participate in the dot product operation only for the cells with exponential step size before it. Thus, the SSA block only needs to compute the $O(logL)$ dot products in each layer.

The query (Q), key (K), and value (V) are the matrices acquired by extracting features from the CBAM block, specifically, embedding $\mathbf{X_2}$ as matrices Q, K, and V, respectively:

$$\begin{aligned} Q &= CBAM_Q(\mathbf{X_2}) \\ K &= CBAM_K(\mathbf{X_2}) \\ V &= CBAM_V(\mathbf{X_2}), \end{aligned} \tag{3}$$

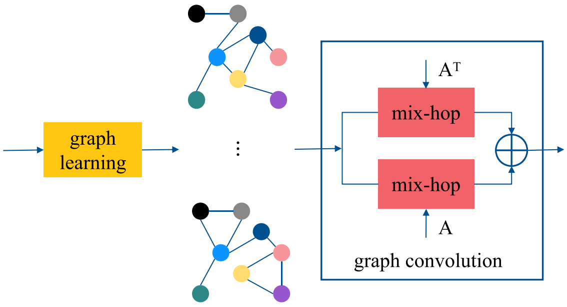


Fig. 7. Mix-hop block.

where $CBAM_Q$,$CBAM_K$ and $CBAM_V$ are the CBAM block functions for obtaining the Q, K and V, $\mathbf{X_2}$ is the result of Eq. (2). The dimensionalities of Q, K, and V are $\frac{T}{4i} \times 4iC$.

The dot product of the self-attention mechanism is then calculated by

$$\mathbf{F} = SoftMax(\frac{QK^T}{\sqrt{d_k}} + M)V, \tag{4}$$

as shown in Fig. 5, where $SoftMax$ is used to compute the possibility of a certain matrix as a commonly used function, K^T is the transpose of K, M is a sparse bias matrix with dimensionality $\frac{T}{4i} \times \frac{T}{4i}$, and $\sqrt{d_k}$ is the scaling factor with the value $4iC$ to avoid the result from being too large or too small.

Finally, $\mathbf{F}$ goes through an LN and MLP. The above process is repeated twice in the attention block, with the difference that a shifted layer [31] is added the second time, which moves the temporal blocks within the window to solve the problem of global features being restricted to the local window partition. $\mathbf{F}$ also goes through CBAM, SSA, LN, and MLP to obtain the result of the i-th attention block $\mathbf{X_3}$, which is a tensor with dimensionality $\frac{T}{4i} \times 4iC$.

3.3. *Spatial-local block*

The spatial-local block is used to capture spatial dependency features, as shown in Fig. 7, which consists of a CBAM block and mix-hop block.

3.3.1. *CBAM block*

The CBAM block in this section has the same structure as the block in Section 3.2.2. The result of the i-th attention block $\mathbf{X_3}$ is processed by the CBAM block for feature refinement to obtain the result of CBAM block $\mathbf{X_4}$, which is a tensor with dimensionality $T \times C$ (T is the length of the initial MTS, and C is the dimension of the initial MTS).

3.3.2. *Mix-hop block*

The mix-hop block consists of two parts: a graph learning layer is used to adaptively learn the graph adjacency matrix to mine the unidirective pairwise hidden relationships among MTS, and a graph convolution layer is used to integrate the information of nodes with that of their neighbors to handle the spatial dependency in the graph.

The multidimensional space of MTS is an essential spatial dependency feature compared with univariate time series. The GNN has achieved great success in processing spatial dependency among entities. Therefore we use a GNN to obtain spatial dependency features.

Graph learning layer We use an adaptive relational embedding strategy that adaptively learns the adjacency matrix $A \in R^{N \times N}$ (N is the number of MTS sample dimensions), which is a metric-based approach. It first requires the initialization of an adjacency matrix that is independent of the features of each dimension. Then the model optimizes the adjacency matrix during training and uses a metric function to build an adjacency matrix. The method requires few parameters. The purpose is to model the complex hidden dependency of MTS global features $\mathbf{X_4}$ from the last layer to capture the pairwise hidden relationships among MTS data as Eq. (5)-Eq. (12).

$$E_1 = Embed_1(\mathbf{X_4}) \tag{5}$$

$$E_2 = Embed_2(\mathbf{X_4}) \tag{6}$$

$$M_1 = tanh(E_1\mathbf{W_1}) \tag{7}$$

$$M_2 = tanh(E_2\mathbf{W_2}) \tag{8}$$

$$A = relu(tanh(M_1 M_2^T - M_2 M_1^T)) \tag{9}$$

$$idx = topk(A[i,:]) \tag{10}$$

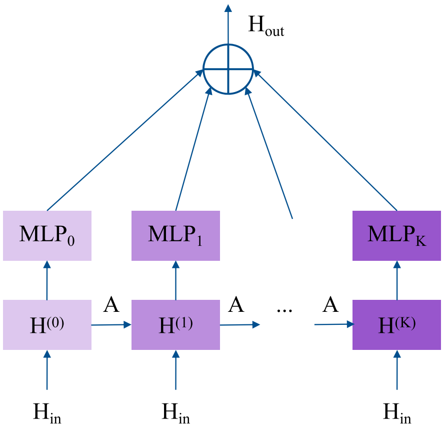


Fig. 8. Mix-hop layer. K denotes the propagation depth, H_{in} denotes the input hidden states that its previous layer output, and H_{out} denotes the hidden states output by the current layer.

$$A[i, idx] = 1 \tag{11}$$

$$A[i, -idx] = 0. \tag{12}$$

In Eq. (5)-Eq. (6), $\mathbf{X_4}$ is the result processed by CBAM block (refer to Section 3.3.1). E_1 and E_2 are matrix with dimensionality $C \times C$ (C is the dimension of the initial MTS), which used to represent for the relationships among the dimensions of MTS. E_1 and E_2 are initialized through $Embed_1$ and $Embed_2$ operation, respectively. In Eq. (5)-Eq. (6), $\mathbf{W_1}$ and $\mathbf{W_2}$ are learned parameters with dimensionality $T \times C$, M_1 and M_2 are learned matrix with dimensionality $C \times C$. In Eq. (7)-Eq. (9), $tanh$ and $relu$ are the activation function. In Eq. (9)-Eq. (12), A is the learned adjacency matrix with dimensionality $C \times C$, i ($i \in 1, ..., N$) are adjacency matrix nodes, $topk$ returns the first k maximum indexes for each node. Finally, the weights of the connected nodes are set to 1 through Eq. (11) and the weights of the non-connected nodes are set to 0 through Eq. (12).

Graph convolution layer The graph convolution layer includes double mix-hop propagation layers that handle the input and output information through each node. Finally, we obtain the spatial dependency features. Fig. 8 shows the structure of the mix-hop block. The graph convolution layer consists of double mix-hop layers [36], which handle the transpose of the adjacency matrix and adjacency matrix:

$$H_{out} = \sum_{i=0}^{K} H^{(i)} \mathbf{W}^i \tag{13}$$

where H_{out} represents the output information of the graph convolution, H_{out} aggregates the information at propagation depths $H^{(0)}$–$H^{(K)}$, and $\mathbf{W}^i$ represents the parameter that corresponds to propagation depth $H^{(i)}$. $H^{(0)}$–$H^{(K)}$ and H_{out} are tensors with dimensionality $T \times C$.

3.4. Classification layer

Finally, the final features are used to achieve classification by an MLP. In this stage, $\mathbf{X}_5$ is applied to predict the class labels. An MLP is used to convert the final features into class labels. Finally, the predicted class labels are compared with the true labels. The loss function is

$$loss = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{M} y_{i,j} logp(\hat{y}_{i,j})), \tag{14}$$

where N is the number of the training datasets, M is the number of class labels, y denotes the true label, and $\hat{y}$ denotes the model prediction label.

4. Experiments

4.1. Experimental setting

4.1.1. Datasets

We used 14 benchmark datasets from the UEA archive to conduct experiments on MF-Net. The datasets were from various application areas: Motion, EEG/MEG, Human Activity Recognition (HAR), Audio Spectra (AS), ECG, and Other. Some characteristics of each dataset are shown in Table 1.

Table 1
Brief information regarding the 14 benchmark datasets.

Datasets	Type	Train	Test	Dimensions	Lengths	Classes
ArticularyWordRecognition	Motion	275	300	9	144	25
AtrialFibrillation	ECG	15	15	2	640	3
BasicMotions	HAR	40	40	6	100	4
CharacterTrajectories	Motion	1422	1436	3	182	20
FaceDetection	EEG/MEG	5890	3524	144	62	2
HandMovementDirection	EEG/MEG	160	74	10	400	4
Heartbeat	AS	204	205	61	405	2
MotorImagery	EEG/MEG	278	100	64	3000	2
NATOPS	HAR	180	180	24	51	6
PEMS-SF	Other	267	173	963	144	7
PenDigits	Motion	7494	3498	2	8	10
SelfRegulationSCP2	EEG/MEG	200	180	7	1152	2
SpokenArabicDigits	AS	6599	2199	13	93	10
StandWalkJump	ECG	12	15	4	2500	3

4.1.2. *Experimental parameters*

Our experiments were conducted on a CPU running Windows i7 with 16 GB RAM using Python 3.9 and PyTorch 1.10. We performed minimization using a cross-entropy loss function and used Adam as the optimizer. We obtained all the results after approximately 30–50 epochs. Epoch was a hyperparameter that we could adjust manually. After 50 epochs, the loss and accuracy of all the experimental models converged to obtain more stable and convincing results. The details of the loss and accuracy diagrams are in Section 4.6.2.

4.1.3. *Reproducibility*

To enable reproducibility, relevant source code and parameters are released on Github.[2] The results can be reproduced independently.

4.2. *Experimental performance*

We compared MF-Net with ED-1NN [37], DTW-1NN-I [37], variant methods of ED-1NN and DTW-1NN-I [37], MLSTM-FCN [27], WEASEL+MUSE [19], TapNet [26], MR-PETSC [38], SMATE [39], and DA-Net [14]. We used 14 datasets selected from the latest research DA-Net to conduct a fair comparison experiment with the above implementations. Table 2 clearly illustrates the accuracy of MF-Net and the above implementations. In Table 2, “AVG acc” represents the average accuracy of the implementations on the 14 datasets and “Win” represents the number of datasets for which the method achieved the best performance and accuracy data, where the best performances are marked in bold. In the table, the result “N/A” indicates that the corresponding method had no result.

Based on the results provided in Table 2, we clearly observed that MF-Net achieved five wins on 14 datasets, which tied with DA-Net and TapNet, and MF-Net had the highest average accuracy among these methods. Specifically, our method outperformed the state-of-the-art methods by more than 1.7% on the PEMS-SF dataset, more than 5.2% on the AtrialFibrillation dataset, and more than 1.9% on the FaceDetection dataset. The experimental results demonstrated that the performance of MF-Net in MTS classification was superior.

We achieved five wins on 14 datasets, and most of the five datasets had a larger dimensionality than the other datasets, such as PEMS-SF with 963 dimensions, FaceDetection with 144 dimensions, and SpokenArabicDigits with 13 dimensions. MF-Net exploited the spatial dependency features of the datasets easily and improved classification accuracy.

Moreover, for various datasets, the number and length of variables spanned a wide range. For example, the number of MTS variables ranged from 2 to 963, the minimum length of MTS data was 8, and the maximum length reached 3000. With variable dataset parameters, our framework remained quite competitive, thus demonstrating the robustness of our model. Thus, MF-Net obtained good results by adequately modeling local features, global features, and spatial dependency feature correlations.

To evaluate and illustrate the differences between MF-Net and other implementations, we show the critical difference (CD) diagram based on the accuracy in Table 2. The CD diagram ranks the 13 MTS classification classifiers by their rank in ascending order. As shown in Fig. 9, MF-Net had the second-smallest rank, after TapNet. The CD diagram also illustrates the superiority of our approach.

4.3. *Sensitivity analysis*

We removed the global-local and spatial-local block separately to verify the effect of each block on the experimental results. We performed sensitivity experiments on the 14 benchmark datasets in Table 1. Table 3 clearly illustrates the contrast among the global-local black, spatial-local block and MF-Net.

[2] https://github.com/dumingsen/MF-Net.

Table 2
Experimental results from traditional implementations.

Datasets	ED-1NN	DTW-1NN-D	ED-1NN(norm)	DTW-1NN-D(norm)	DTW-1NN-I	DTW-1NN-I(norm)	MLSTM-FCN	WEASEL+MUSE	TapNet	MR-PETSC	SMATE	DA-Net	MF-Net
ArticularyWordRecognition	0.970	0.987	0.970	0.98	0.980	0.980	0.973	0.990	0.987	**0.997**	0.993	0.980	0.983
AtrialFibrillation	0.267	0.200	0.247	0.220	0.267	0.267	0.267	0.333	0.333	0.400	0.133	0.414	**0.466**
BasicMotions	0.675	0.975	0.676	0.975	**1.000**	**1.000**	0.950	**1.000**	**1.000**	**1.000**	**1.000**	0.925	0.950
CharacterTrajectories	0.964	0.990	0.964	0.989	0.969	0.969	0.985	0.990	0.997	0.984	0.987	**0.998**	0.958
FaceDetection	0.519	0.529	0.519	0.529	0.513	0.500	0.545	0.545	0.556	0.574	0.563	0.645	**0.664**
HandMovementDirection	0.279	0.231	0.278	0.231	0.306	0.303	0.365	0.365	0.378	0.365	**0.527**	0.347	0.445
Heartbeat	0.620	0.717	0.619	0.717	0.659	0.658	0.663	0.727	**0.751**	0.702	0.727	0.626	0.692
MotorImagery	0.510	0.500	0.510	0.500	0.390	N/A	0.510	0.500	**0.590**	0.490	**0.590**	0.550	0.540
NATOPS	0.860	0.883	0.850	0.883	0.850	0.850	0.889	0.870	**0.939**	0.917	0.883	0.877	0.927
PEMS-SF	0.705	0.711	0.705	0.711	0.734	0.734	0.699	N/A	0.751	0.861	0.763	0.867	**0.884**
PenDigits	0.973	0.977	0.973	0.977	0.939	0.939	0.978	0.948	0.980	0.905	0.980	**0.989**	0.983
SelfRegulationSCP2	0.483	0.539	0.483	0.539	0.533	0.533	0.472	0.460	0.550	0.533	0.556	**0.561**	0.533
SpokenArabicDigits	0.967	0.963	0.967	0.963	0.960	0.959	**0.990**	0.982	0.983	0.960	0.982	**0.990**	**0.990**
StandWalkJump	0.200	0.200	0.200	0.200	0.333	0.333	0.067	0.333	0.400	0.400	0.200	0.400	0.400
AVG acc	0.642	0.672	0.640	0.673	0.674	0.694	0.668	0.696	0.728	0.721	0.706	0.726	**0.744**
Win	0	0	0	0	1	1	1	1	**5**	2	3	**5**	**5**

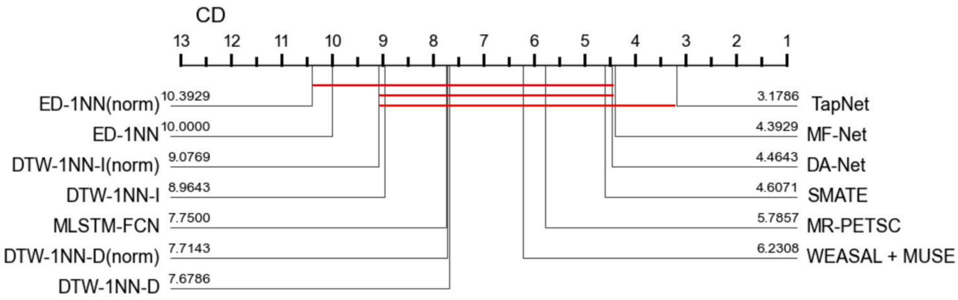
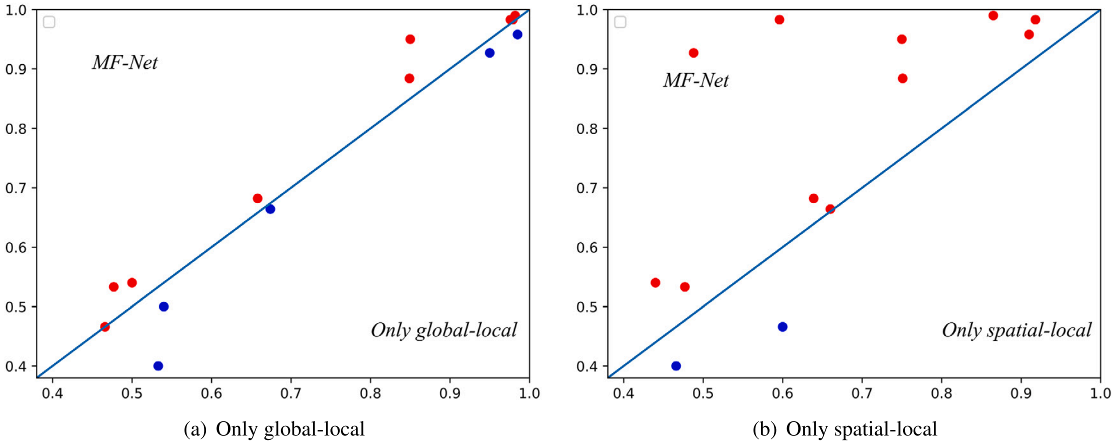


Fig. 9. CD diagram of 13 MTS classification implementations on the 14 UEA datasets. A red horizontal line indicates no significant difference for a set of implementations.

Table 3
Sensitivity study of the global-local and spatial-local block on the 14 UEA datasets.

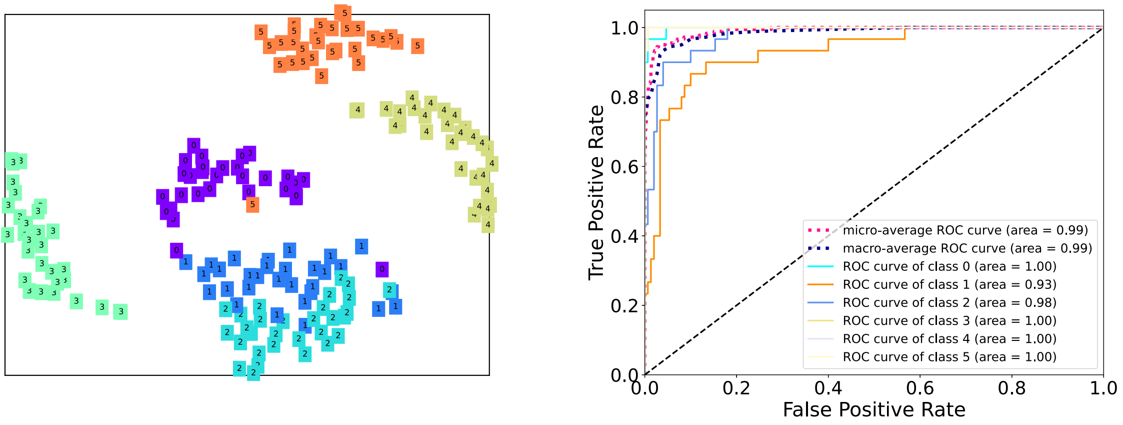
Datasets	Only global-local	Only spatial-local	MF-Net
ArticularyWordRecognition	0.976	0.596	**0.983**
AtrialFibrillation	0.466	**0.600**	0.466
BasicMotions	0.850	0.750	**0.950**
CharacterTrajectories	**0.985**	0.910	0.958
FaceDetection	**0.674**	0.660	0.664
HandMovementDirection	**0.540**	0.337	0.500
Heartbeat	0.658	0.639	**0.682**
MotorImagery	0.500	0.440	**0.540**
NATOPS	**0.950**	0.488	0.927
PEMS-SF	0.849	0.751	**0.884**
PenDigits	0.979	0.918	**0.983**
SelfRegulationSCP2	0.477	0.477	**0.533**
SpokenArabicDigits	0.982	0.865	**0.990**
StandWalkJump	**0.533**	0.466	0.400
AVG acc	0.744	0.635	**0.747**
Win	5	1	**8**



(a) Only global-local (b) Only spatial-local

Fig. 10. Two-by-two comparison of two sensitivity experiments. A point represents a dataset. The closer the point to the upper left corner, the better the dataset performed in MF-Net. The range in the lower quadrant of the $y = x$ line indicates that MF-Net was inferior in the sensitivity experiment.

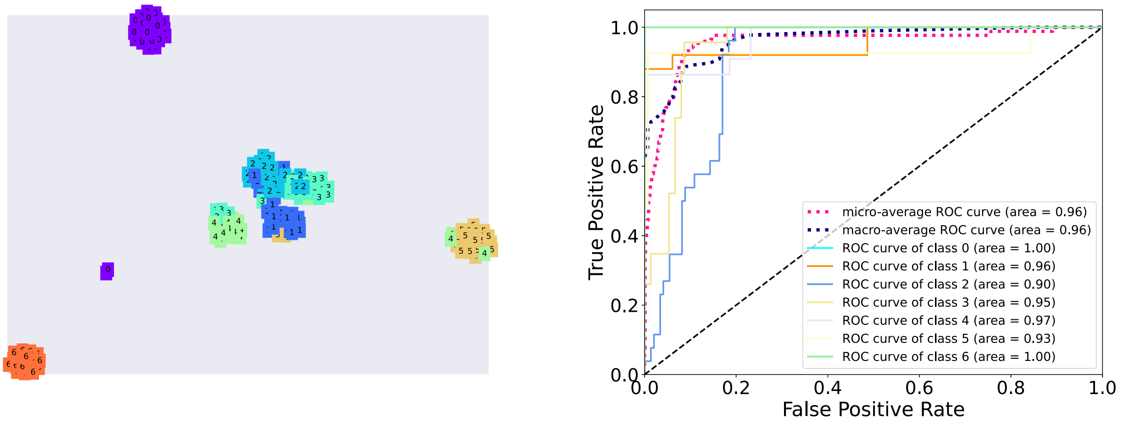
We verified the effect of the spatial-local block on the study. According to the results in Table 3 and Fig. 10(a), the effect of the spatial-local block was obvious. Compared with the network without the spatial-local block, the average accuracy of MF-Net improved, obtaining nine wins on 14 datasets. Additionally, we verified the effect of the global-local block on the study. According to the results in Table 3 and Fig. 10(b), the effect of the global-local block was noticeable. Compared with the network without the global-local block, the average accuracy of MF-Net improved by 1.12%, and 13 wins were obtained on 14 datasets. This result shows that effective local features, efficient global features, and spatial dependency features were essential for MTS classification.



(a) T-SNE visualization.

(b) ROC curve.

Fig. 11. T-SNE analysis and ROC curve of the NATOPS dataset.



(a) T-SNE visualization.

(b) ROC curve.

Fig. 12. T-SNE analysis and ROC curve of the PEMS-SF dataset.

4.4. Reduction of the dimensional feature visualization and ROC curve

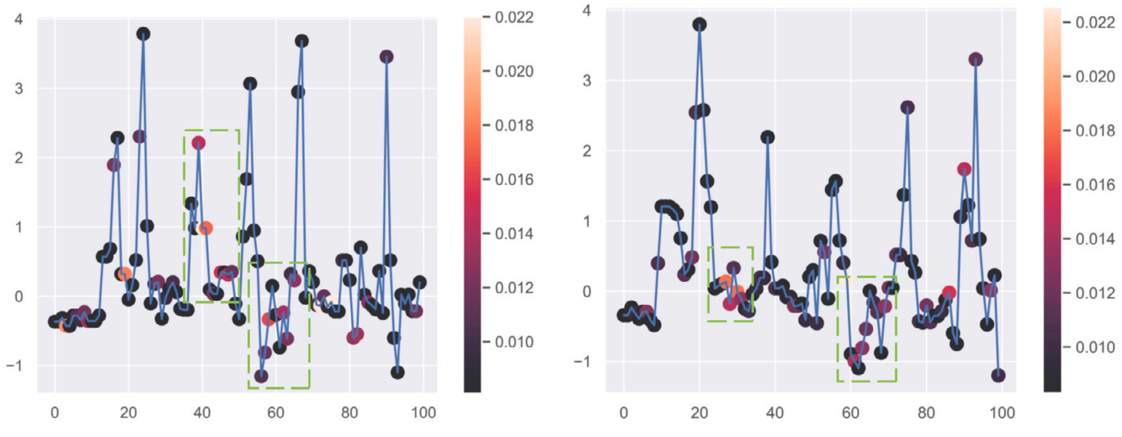
In this section, we represent the time series embeddings in a reduction of dimensionality to prove the effectiveness of our well-trained embeddings. We used the t-SNE algorithm [40] to visualize the NATOPS dataset and BasicMotions dataset embedded as two-dimensional images, with different colors and numbers to distinguish the different categories, as shown in Figs. 11 and 12. Fig. 11(a) shows the embedding of learning results for the NATOPS dataset, and it consists of 180 test samples of six classes. Based on the results obtained, we can draw the following conclusions: (1) The distance among MTS data samples from different classes was larger than the distance among data samples of identical classes. Therefore, we could easily use the learned MTS for embedding in MTS classification. (2) According to this conclusion, class 0 and class 5 were more easily classified by the classifier, whereas the class 1 and class 2 visualization results indicated that we need to adopt alternative measures to improve the classifier.

As shown in Fig. 11(b), we plotted the ROC curves. Classes 0 and 5 reached the top of the curve, whereas classes 1 and 2 were at the bottom of the curve relative to all classes, which is consistent with our analysis of t-SNE: classes 0 and 5 were more easily classified by the classifier.

Fig. 12(a) shows that classes 1, 2, and 3 were more difficult to classify, whereas classes 0, 5, and 6 were relatively easy to classify. Classes 1, 2, and 3 were at the bottom of the curve relative to all classes in Fig. 12(b), which again illustrates the above phenomenon.

4.5. Feature visualization

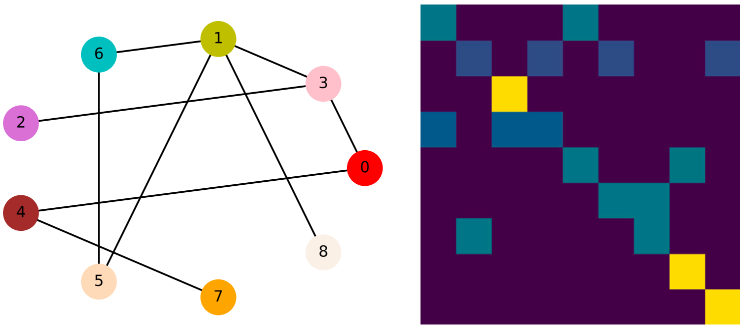
We used Grad-CAM to visualize the local and global feature capture capability of the global-local block. Additionally, we visualized the adjacency matrices formed by the nine dimensions of the ArticularyWordRecognition dataset and formed by the 13 dimensions of the SpokenArabicDigits dataset.



(a) Sample 1.

(b) Sample 2.

Fig. 13. Heat map of the global-local block of the BasicMotions dataset.



(a) Graph structure.

(b) Adjacency matrix.

Fig. 14. Spatial dependency feature visualization of the ArticularyWordRecognition dataset.

4.5.1. *Global-local feature visualization*

We used Grad-CAM to support the interpretation of MF-Net to mine local and global features, thereby providing an interpretation of the contribution of each time series local feature to the classified target time series. We extracted the features of the global-local block of MF-Net as our feature layer. Then we computed the weights of the feature layer by integrating the forward and backward propagation information. Finally, we mapped the weights to the input MTS dimensions to obtain a heat map of those MTS dimensions. Fig. 13 shows that MF-Net captured the distinguishing local features (dashed green boxed area) in the BasicMotions dataset. The visualization well illustrated the capabilities of MF-Net to mine local and global features.

4.5.2. *Spatial dependency feature visualization*

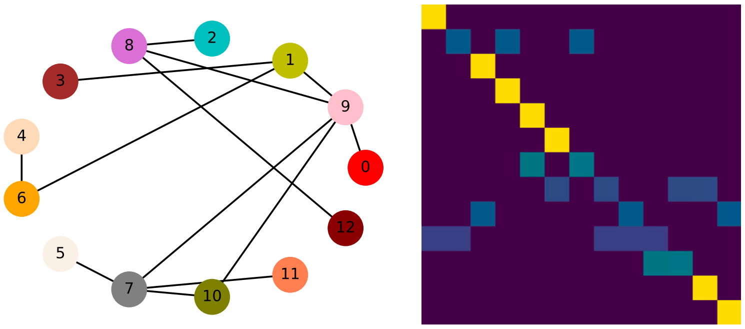
We used the ArticularyWordRecognition dataset and SpokenArabicDigits dataset to visualize the dependency features among various dimensions of the MTS clearly, as shown in Figs. 14 and 15. Figs. 14(a) and 15(a) show an example of the graph structure of the belonging dataset. Each color of dot represents a dimension of the MTS.

MTS do not have access to a fixed natural graph structure like a molecular formula [41]. Many methods require a predefined graph as input, for example, similarity (e.g., distance [5]) and correlation (e.g., Pearson correlation [6]); however, a predefined dependency graph rarely exists in real-life MTS problems.

When the adjacency matrix is not given, the graph structure of MTS needs to be constructed first. Because of the large number of dimensions, the pairwise relationships (e.g., distance) among MTS are computationally very large and the unclear relationships are difficult to construct. Additionally, the graph structure constructed based on a certain metric may also be unreasonable [42], which leads to obtaining indistinguishable graph structure representations.

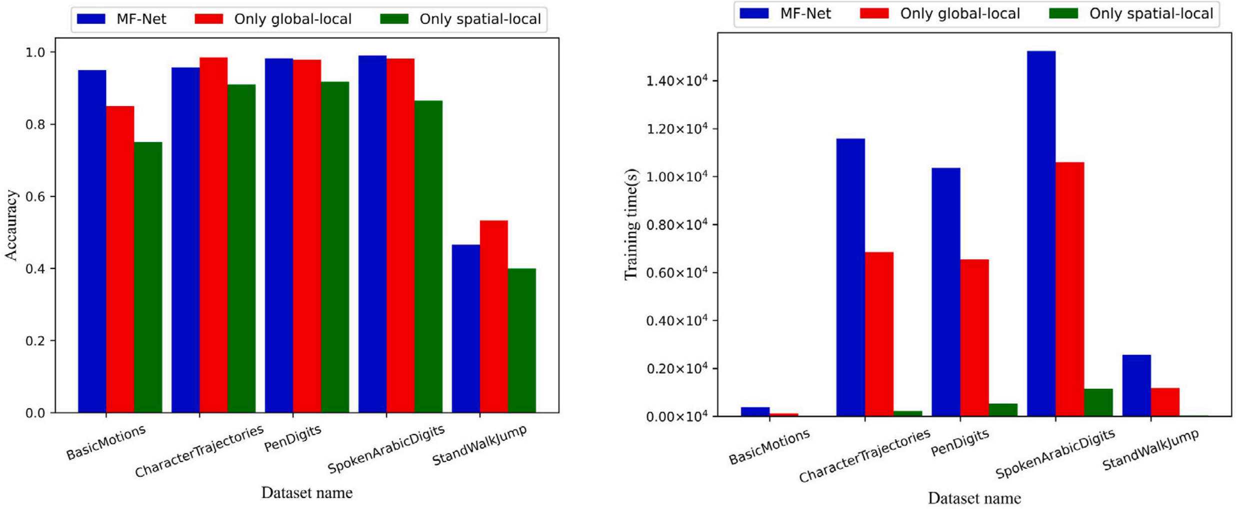
Therefore, we provide an adaptive learning adjacency matrix based on Section 3.3.2. Figs. 14(b) and 15(b) show the learned adjacency matrix of the belonging dataset through the heat map based on various dimensions in the spatial-local block, where the different colors represent the connectivity strengths of the MTS dimensions.

The ArticularyWordRecognition dataset is a nine-dimensional dataset acquired by placing nine electromagnetic articulograph sensors in different organs (e.g., tongue and lips) of the head during speech, which is designed to classify 25 types of words. Combined with the graph structure learned, we can identify which organs are more closely connected to each other during the



(a) Graph structure. (b) Adjacency matrix.

Fig. 15. Spatial dependency feature visualization of the SpokenArabicDigits dataset.



(a) Accaurancy. (b) Training time.

Fig. 16. Training time and accuracy comparison.

vocalization process. Similarly, the graph structure of the SpokenArabicDigits dataset illustrates the connection among 13 frequency cepstral coefficients used to classify 10 Arabic letters.

4.6. Running efficiency and epoch analysis

4.6.1. Running efficiency and time cost comparison

The spatial-local block, as a lightweight block, has fewer parameters and less training time; however, it also results in a decrease in accuracy and therefore cannot be used alone. Therefore, we compared training time and accuracy on five datasets, as shown in Fig. 16.

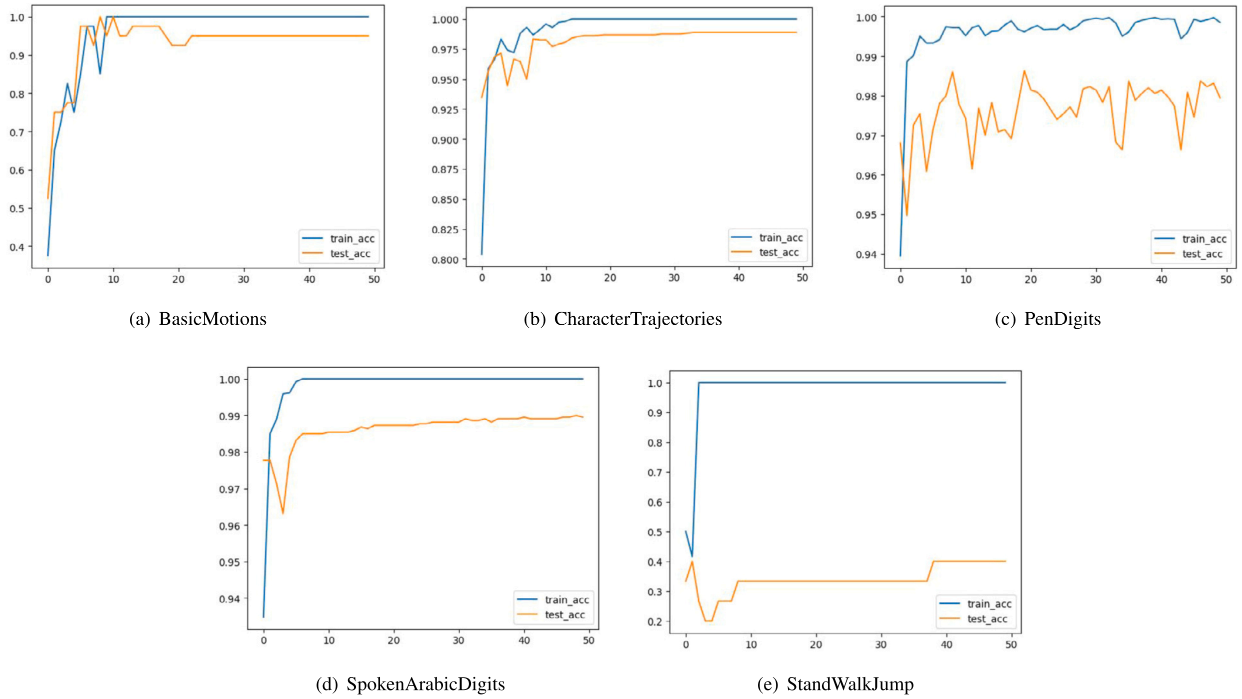
As shown in Figs. 16(a) and 16(b), MF-Net achieved the highest accuracy in general; however, it had the highest training time because the model parameters are the sum of the global-local block and spatial-local block.

4.6.2. Epoch analysis

We obtained loss and accuracy diagrams for various epochs from five datasets in the previous subsection. From Figs. 17 and 18, we conclude the following: (1) In the initial stage, the training and testing losses gradually decreased, despite some fluctuations in the curve, and accuracy increased as the number of epochs increased. (2) When proceeding to certain epochs, the training and testing accuracy curves converged. (3) The loss value fluctuated within a small range, which indicates the better fitting ability of MF-Net.

5. Conclusion

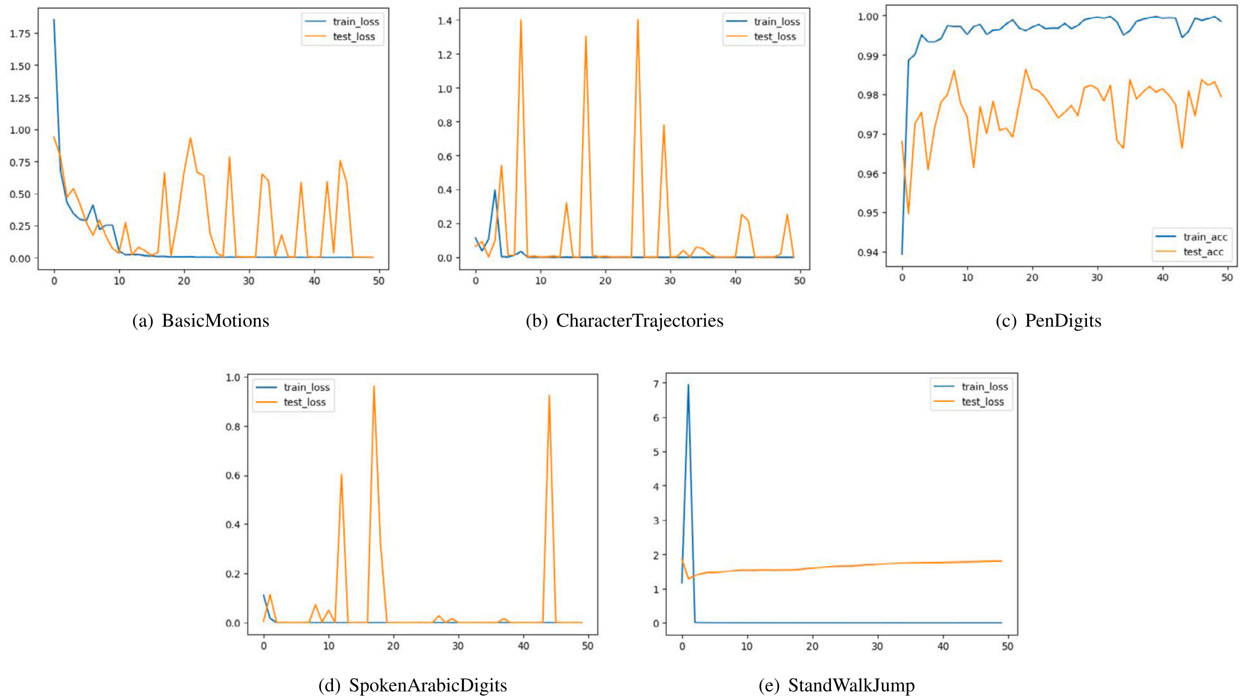
In this paper, we proposed MF-Net, which consists of two parts: global-local and spatial-local blocks. First, the global-local block focuses on more distinguishing local features using the CBAM block attention mechanism. We used the SSA mechanism of the SSA



(a) BasicMotions (b) CharacterTrajectories (c) PenDigits

(d) SpokenArabicDigits (e) StandWalkJump

Fig. 17. Accuracy for various epochs.



(a) BasicMotions (b) CharacterTrajectories (c) PenDigits

(d) SpokenArabicDigits (e) StandWalkJump

Fig. 18. Loss for various epochs.

block to capture global features and computational complexity was reduced. Then, the spatial-local block successfully extracted the spatial dependency features by integrating the global features into a graph using a mix-hop block. Thus, MF-Net successfully modeled the spatial dependency features, and global and local feature correlations for the MTS classification task. Our experiments showed that MF-Net obtained excellent results and outperformed the state-of-the-art methods.

CRediT authorship contribution statement

Mingsen Du: Conceptualization, Methodology, Validation, Writing – original draft, Writing – review & editing. **Yanxuan Wei:** Methodology, Validation, Writing – original draft, Writing – review & editing. **Xiangwei Zheng:** Project administration, Supervision. **Cun Ji:** Methodology, Project administration, Supervision, Validation, Writing – original draft, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgements

This work was supported by the Innovation Methods Work Special Project under Grant 2020IM020100, and the Natural Science Foundation of Shandong Province under Grant ZR2020QF112. We thank Liwen Bianji (Edanz) (www.liwenbianji.cn/) for editing the English text of a draft of this manuscript.

References

[1] C. Ji, M. Du, Y. Hu, S. Liu, L. Pan, X. Zheng, Time series classification based on temporal features, Appl. Soft Comput. 128 (2022) 109494.

[2] S. Qiu, H. Zhao, N. Jiang, Z. Wang, L. Liu, Y. An, H. Zhao, X. Miao, R. Liu, G. Fortino, Multi-sensor information fusion based on machine learning for real applications in human activity recognition: state-of-the-art and research challenges, Inf. Fusion 80 (2022) 241–265.

[3] L. Ye, E. Keogh, Time series shapelets: a new primitive for data mining, in: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2009, pp. 947–956.

[4] Z. Xiao, X. Xu, H. Xing, S. Luo, P. Dai, D. Zhan, Rtfn: a robust temporal feature network for time series classification, Inf. Sci. 571 (2021) 65–86.

[5] Z. Duan, H. Xu, Y. Wang, Y. Huang, A. Ren, Z. Xu, Y. Sun, W. Wang, Multivariate time-series classification with hierarchical variational graph pooling, Neural Netw. 154 (2022) 481–490.

[6] Y. Wang, Z. Duan, Y. Huang, H. Xu, J. Feng, A. Ren, Mthetgnn: a heterogeneous graph embedding framework for multivariate time series forecasting, Pattern Recognit. Lett. 153 (2022) 151–158.

[7] D. Yang, H. Chen, Y. Song, Z. Gong, Granger causality for multivariate time series classification, in: 2017 IEEE International Conference on Big Knowledge (ICBK), IEEE, 2017, pp. 103–110.

[8] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, P.-A. Muller, Deep learning for time series classification: a review, Data Min. Knowl. Discov. 33 (4) (2019) 917–963.

[9] A. Quattoni, S. Wang, L.-P. Morency, M. Collins, T. Darrell, M. Csail, Hidden-state conditional random fields, IEEE Trans. Pattern Anal. Mach. Intell. 29 (10) (2007) 1848–1852.

[10] W. Pei, H. Dibeklioğlu, D.M. Tax, L. van der Maaten, Multivariate time-series classification using the hidden-unit logistic model, IEEE Trans. Neural Netw. Learn. Syst. 29 (4) (2017) 920–931.

[11] Y. LeCun, Y. Bengio, G. Hinton, et al., Deep learning, Nature 521 (7553) (2015) 436–444.

[12] C. Ji, Y. Hu, S. Liu, L. Pan, B. Li, X. Zheng, Fully convolutional networks with shapelet features for time series classification, Inf. Sci. 612 (2022) 835–847.

[13] M. Liu, S. Ren, S. Ma, J. Jiao, Y. Chen, Z. Wang, W. Song, Gated transformer networks for multivariate time series classification, arXiv preprint, arXiv:2103.14438, 2021.

[14] R. Chen, X. Yan, S. Wang, G. Xiao, Da-net: dual-attention network for multivariate time series classification, Inf. Sci. 610 (2022) 472–487.

[15] S. Seto, W. Zhang, Y. Zhou, Multivariate time series classification using dynamic time warping template selection for human activity recognition, in: 2015 IEEE Symposium Series on Computational Intelligence, IEEE, 2015, pp. 1399–1406.

[16] M. Shokoohi-Yekta, B. Hu, H. Jin, J. Wang, E. Keogh, Generalizing dtw to the multi-dimensional case requires an adaptive approach, Data Min. Knowl. Discov. 31 (1) (2017) 1–31.

[17] L. Ye, E. Keogh, Time series shapelets: a novel technique that allows accurate, interpretable and fast classification, Data Min. Knowl. Discov. 22 (1) (2011) 149–182.

[18] I. Karlsson, P. Papapetrou, H. Boström, Generalized random shapelet forests, Data Min. Knowl. Discov. 30 (5) (2016) 1053–1085.

[19] P. Schäfer, U. Leser, Multivariate time series classification with weasel+muse, arXiv preprint, arXiv:1711.11343, 2017.

[20] Z. Li, J. Tang, Semi-supervised local feature selection for data classification, Sci. China Inf. Sci. 64 (9) (2021) 1–12.

[21] J. Yang, M.N. Nguyen, P.P. San, X.L. Li, S. Krishnaswamy, Deep convolutional neural networks on multichannel time series for human activity recognition, in: Twenty-Fourth International Joint Conference on Artificial Intelligence, 2015.

[22] W. Chen, K. Shi, Multi-scale attention convolutional neural network for time series classification, Neural Netw. 136 (2021) 126–140.

[23] H. Ismail Fawaz, B. Lucas, G. Forestier, C. Pelletier, D.F. Schmidt, J. Weber, G.I. Webb, L. Idoumghar, P.-A. Muller, F. Petitjean, Inceptiontime: finding alexnet for time series classification, Data Min. Knowl. Discov. 34 (6) (2020) 1936–1962.

[24] Z. Chen, Y. Liu, J. Zhu, Y. Zhang, R. Jin, X. He, J. Tao, L. Chen, Time-frequency deep metric learning for multivariate time series classification, Neurocomputing 462 (2021) 221–237.

[25] Y. Zheng, Q. Liu, E. Chen, Y. Ge, J.L. Zhao, Time series classification using multi-channels deep convolutional neural networks, in: International Conference on Web-Age Information Management, Springer, 2014, pp. 298–310.

[26] X. Zhang, Y. Gao, J. Lin, C.-T. Lu, Tapnet: multivariate time series classification with attentional prototypical network, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, 2020, pp. 6845–6852.

[27] F. Karim, S. Majumdar, H. Darabi, S. Harford, Multivariate lstm-fcns for time series classification, Neural Netw. 116 (2019) 237–245.

[28] F. Scarselli, M. Gori, A.C. Tsoi, M. Hagenbuchner, G. Monfardini, The graph neural network model, IEEE Trans. Neural Netw. 20 (1) (2008) 61–80.

[29] Z. Duan, H. Xu, Y. Huang, J. Feng, Y. Wang, Multivariate time series forecasting with transfer entropy graph, arXiv preprint, arXiv:2005.01185, 2020.

[30] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, C. Zhang, Connecting the dots: multivariate time series forecasting with graph neural networks, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 753–763.

[31] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, Swin transformer: hierarchical vision transformer using shifted windows, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 10012–10022.

[32] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).

[33] S. Woo, J. Park, J.-Y. Lee, I.S. Kweon, Cbam: convolutional block attention module, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 3–19.

[34] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, W. Zhang, Informer: beyond efficient transformer for long sequence time-series forecasting, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, 2021, pp. 11106–11115.

[35] S. Li, X. Jin, Y. Xuan, X. Zhou, W. Chen, Y.-X. Wang, X. Yan, Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting, Adv. Neural Inf. Process. Syst. 32 (2019).

[36] S. Abu-El-Haija, B. Perozzi, A. Kapoor, N. Alipourfard, K. Lerman, H. Harutyunyan, G. Ver Steeg, A. Galstyan, Mixhop: higher-order graph convolutional architectures via sparsified neighborhood mixing, in: International Conference on Machine Learning, PMLR, 2019, pp. 21–29.

[37] Y. Chen, B. Hu, E. Keogh, G.E. Batista, Dtw-d: time series semi-supervised learning from a single example, in: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2013, pp. 383–391.

[38] L. Feremans, B. Cule, B. Goethals, Petsc: pattern-based embedding for time series classification, Data Min. Knowl. Discov. 36 (3) (2022) 1015–1061.

[39] J. Zuo, K. Zeitouni, Y. Taher, Smate: semi-supervised spatio-temporal representation learning on multivariate time series, in: 2021 IEEE International Conference on Data Mining (ICDM), IEEE, 2021, pp. 1565–1570.

[40] L. Van der Maaten, G. Hinton, Visualizing data using t-sne, J. Mach. Learn. Res. 9 (11) (2008).

[41] D.K. Duvenaud, D. Maclaurin, J. Aguilera-Iparraguirre, R. Gómez-Bombarelli, T.D. Hirzel, A. Aspuru-Guzik, R.P. Adams, Convolutional networks on graphs for learning molecular fingerprints, arXiv:1509.09292, 2015.

[42] Z. Wu, S. Pan, G. Long, J. Jiang, C. Zhang, Graph wavenet for deep spatial-temporal graph modeling, in: International Joint Conference on Artificial Intelligence, 2019.