IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 28 December 2022, accepted 22 January 2023, date of publication 31 January 2023, date of current version 6 February 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3241013

RESEARCH ARTICLE

Time Series Classification Based on Multi-Dimensional Feature Fusion

SHUO QUAN[1], MENGYU SUN[1], XIANGYU ZENG[2], XULIANG WANG[1], AND ZEYA ZHU[1]
[1]China Telecom Research Institute, Beijing 102209, China
[2]School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China
Corresponding author: Mengyu Sun (sunmy1@chinatelecom.cn)

ABSTRACT Time series classification is a key problem in data mining, most of existing classification methods directly extract one-dimensional data from one-dimensional features, which cannot effectively express the inter-relation between different time points. Besides, some classification methods extract two-dimensional features through encoding raw one-dimensional data into two-dimensional images, and part of information is lost due to the difference of encoding methods. How to make full use of one-dimensional and two-dimensional features to extract valuable information and integrate them in an optimal fashion remains a promising challenge. In this paper, we propose a multi-scale convolutional network to extract one-dimensional features from time series for obtaining more feature information based on multi-scale convolution kernels. Two-dimensional features are constructed in terms of two-dimensional image coding based on *G*ramian angular field, *M*arkov transition field and *R*ecurrence plot (*GMR*) methods. We develop a multi-dimensional feature fusion approach leveraging *S*queeze-and-*E*xcitation (*SE*) and *S*elf-*A*ttention (*SA*) mechanism to effective fusing one-dimensional multi-scale features and two-dimensional image features in terms of weight setting. We conduct experimental verification based on 84 complete data traces from a typical *UCR* dataset in the field. Experimental results show that the accuracy of our proposed approach improves by 3.35% compared with existing benchmark methods. The *Grad*ient-weighted *C*lass *A*ctivation *M*apping (*Grad-CAM*) visualization analysis method is adopted, where our proposed approach extracts more accurate features and effectively distinguishes different time series data categories.

INDEX TERMS Time series classification, multi-scale convolution ResNet, two-dimensional image, feature fusion, accuracy.

I. INTRODUCTION

With the advent of the *I*nternet *o*f *T*hings (*IoT*) technology, billions of sensor devices have been applied to various fields to improve production efficiency, optimize resource utilization, and reduce perceived costs. These sensor devices generate huge amounts of time series data, which can be used to understand the state of the device and provide the basis for decision-making. *T*ime *S*eries *C*lassification (*TSC*), as a most basic data analysis method, plays a significant role in a wide range. For instance, medical institutions use time series generated by medical equipment to assist symptomatic diagnosis [1]. Telecom operators rely on time series to conduct abnormal detection, network telemetry and capacity planning [2]. Therefore, time series classification receives increasing attention.

Extracting one-dimensional features from time series data for classification is the mainstream method of current *TSC*. Chen et al. propose a *D*ynamic *T*ime *W*arping (*DTW*) accelerator [3], which can get better classification results by analyzing time series fluctuation patterns. The *DTW* distance measurement and simple nearest-neighbor method can achieve overwhelming advantages over traditional Euclidean-style distance algorithms. Zhao et al. focus on subsequences of time series and complete classification based on specific subsequences of these categories [4]. The common point of these methods is to design one-dimensional features manually and then build corresponding classifiers

The associate editor coordinating the review of this manuscript and approving it for publication was Yongming Li.

to complete classification tasks. In addition, the end-to-end depth learning method is used to learn one-dimensional features automatically. Convolutional neural networks such as *F*ully *C*onvolutional *N*etwor (*FCN*) and *ResNet* [5], and recurrent neural networks such as *L*ong *S*hort *T*erm *M*emory (*LSTM*) [6], [7] and *G*ate *R*ecurrent *U*nit (*GRU*) [8], as well as some extended versions, are classic deep learning classification models. However, an amount of time series information is hidden in its structure, and single dimension features cannot effectively distinguish linear inseparable and express time correlation [9], [10], so explicit design is required to extract implicit features [11].

Drawing on the excellent performance of *C*onvolutional *N*eural *N*etworks (*CNN*) in the field of computer vision, the method of encoding one-dimensional time series into two-dimensional images to extract two-dimensional features for classification has received extensive attention. In some classical methods, where *G*ramian *A*ngular *F*ield (*GAF*) can maintain the time correlation of time series, *M*arkov *T*ransition *F*ield (*MTF*) can describe the dynamic transformation statistics of time series [12], and *R*ecurrence *P*lot (*RP*) can express rich texture information and long-term correlation [13]. These methods, converting one-dimensional data to two-dimensional images, can effectively reveal numerical information and time information of time series, the prior knowledge of correlation information, similarity and information quantity at different time points is given, providing additional information to improve the accuracy of model classification. However, due to the difference between the ascending and encoding methods, part of detail information in the original one-dimensional data is lost.

This paper argues that the features extracted by converting one-dimensional data into two-dimensional images should complement rather than replace the features extracted directly from one-dimensional data. Therefore, combining the advantages of the two features, this paper proposes a time series classification method based on multi-dimensional feature fusion, through constructing and effectively integrating different dimensional features, to extract valuable information for further improving classification precision. The main contributions are listed as follows:

- An improved *M*ulti-*S*cale *ResNet* (*MSResNet*) is designed to extract one-dimensional original time series features, which solves the problem that traditional convolutional layer cannot represent different time scale features based on a single convolution kernel, while reducing model parameters and operational overhead.
- In order to make full use of the correlation information of time series in two-dimensional space, we use *GAF*, *MTF* and *RP* to encode time series into three-channel images, extract two-dimensional image features through *GMRResNet* (*GMR ResNet*), and analyze the impact of different two-dimensional image feature construction methods on time series classification.
- Two different feature fusion methods including *SE* and *SA* feature fusion architecture are designed, where one-dimensional features and two-dimensional features are obtained by *MSResNet* and *GMRResNet*, respectively.

We adopt 84 complecte *UCR* datasets [14] to evaluate performance. The mechanism proposed in this paper achieves better numerical results than one-dimensional sequence features and two-dimensional image features alone. At the same time, we use *Grad-CAM* approach to visualize the time series classification. By highlighting the sequence regions that contribute greatly to classification, we verify that using two-dimensional feature enhancement can extract more accurate features and effectively distinguish different time series data categories, thus improving the accuracy of the classification.

The structure of this paper is organized as follows. Section II describes the feature construction method in detail. Section III explains the feature fusion method and fusion network architecture. Section IV presents the numerical results and analysis. Section V introduces some relevant research works, and we summarize the whole paper in Section VI.

II. FEATURE CONSTRUCTION METHOD

Generally, most deep learning-based approaches adopt end-to-end deep neural networks to extract features of one-dimensional time series, such as convolution features and distance features, where the precision of *TSC* can be improved to some extent. However, the information contained by single-class features is generally difficult to fully describe time series comprehensively, and thus limiting the performance of neural networks. In this section, we propose two feature construction methods including a multi-scale *ResNet* and a two-dimensional feature construction based on *GAF*, *MTF* and *RP*.

A. MULTI-SCALE ResNet

ResNet is a deep residual convolutional neural network [15] that has a good effect on image classification, where residual learning is applied through directly bypassing the input information to the output for protecting information integrity. This approach solves the problems of gradient disappearance, gradient explosion, and information loss to some extent.

We construct an *MSResNet* based on *ResNet*, where *MSResNet* can obtain a different range of sequence context information, extract sequence features in different time scales, and enrich the feature expression ability, compared with *ResNet*. In this study, we greatly compressed the model complexity and parameter number to reduce storage space and computing overhead. The network framework is shown in FIGURE 1. For a given input $[x_1, x_2, x_3, \ldots, x_T]$ in different sequence lengths, we initially unify the data length to 512 based on the linear interpolation method for implementing the same convolution operation. The input of *MSResNet* is recorded as (*batchsize*, *l*, 512), where *batchsize* represents the amount of data per batch, *l* refers to the number of input channels, and 512 is the length of time series.

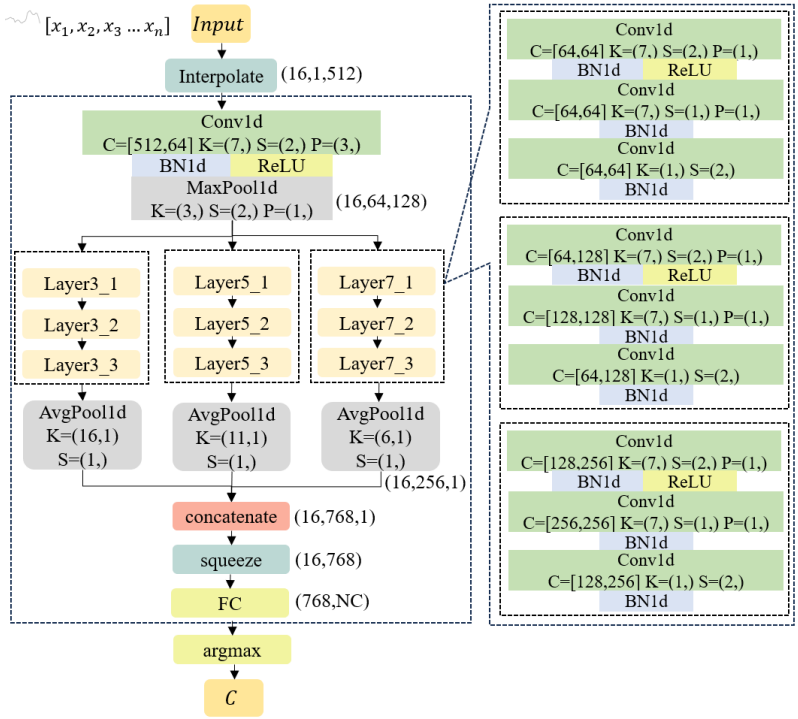


FIGURE 1. The *MSResNet* network framework.

Firstly, the batch normalization of *MSResNet* is executed after one-dimensional convolution, where the parameter adjustment process is simplified to alleviate the gradient disappearance problem and accelerate model learning speed, which has a certain regularization effect. At the same time, *Re*ctified *L*inear *U*nit (*ReLU*) is regarded as activation function of neurons which can effectively alleviate the gradient vanishing problem and fully exploit the discriminant features, compared with traditional *sigmoid* activation function. The maximum pooling layer is adopted to reduce the size of model and improve the computation speed, while improving the robustness of extracted features.

Then, the multi-scale convolution operation is implemented, where *MSResNet* extracted features using three stacked blocks with a convolutional kernel size of 3, 5, 7, each block contains three layers presented by the module on the right side of FIGURE 1. Features are collected based on mean pooling, and the different scale features are stitched by column and sent to the fully connected layer after the squeeze function where the main function of the fully connected layer is to map the feature space to the sample marker space and complete the classification task.

B. TWO-DIMENSIONAL FEATURE CONSTRUCTION METHOD

1) GRAMIAN ANGULAR FIELD

A polar-based matrix is adopted to encode time series into an image for maintaining absolute temporal correlation. A one-dimensional time series in Cartesian coordinate is converted into a polar coordinate, and a trigonometric function is applied to generate a *GAF* matrix. There are three computing processes including: (i) numerical scaling, (ii) polar coordinate conversion, and (iii) trigonometric function transformation.

Time series in Cartesian coordinates are scaled to [0, 1] according to the Equation as follows:

$$\widetilde{x_0^t} = \frac{x(t) - min(x)}{max(x) - minx} \tag{1}$$

where $x(t)$ is the value of time series at time t, $min(x)$ and $max(x)$ refer to the minimum and maximum values of time series respectively.

Polar coordinate conversion based on coordinate transformation formula to convert the Cartesian coordinate series into polar coordinate time services. *GAF* can generate two kinds of images through different functions, where *G*ramian *A*ngular *S*ummation *F*ield (*GASF*) and *G*ramian *A*ngular *D*ifference *F*ield (*GADF*) are formed through leveraging *sin* and *cos* functions, respectively. The computing equations are shown as follows:

$$GASF = \begin{bmatrix} cos(\phi_1 + \phi_1) & \cdots & cos(\phi_1 + \phi_n) \\ cos(\phi_2 + \phi_1) & \cdots & cos(\phi_2 + \phi_n) \\ \vdots & \ddots & \vdots \\ cos(\phi_n + \phi_1) & \cdots & cos(\phi_n + \phi_n) \end{bmatrix} \tag{2}$$

$$GASF = \widetilde{x'} \cdot \widetilde{x} - \sqrt{1 - \widetilde{x'^2}} \cdot \sqrt{1 - \widetilde{x^2}} \tag{3}$$

$$GADF = \begin{bmatrix} sin(\phi_1 + \phi_1) & \cdots & sin(\phi_1 + \phi_n) \\ sin(\phi_2 + \phi_1) & \cdots & sin(\phi_2 + \phi_n) \\ \vdots & \ddots & \vdots \\ sin(\phi_n + \phi_1) & \cdots & sin(\phi_n + \phi_n) \end{bmatrix} \tag{4}$$

$$GADF = \sqrt{1 - \widetilde{x'^2}} \cdot \widetilde{x} - \sqrt{1 - \widetilde{x'^2}} \cdot \widetilde{x'} \tag{5}$$

where each item represents an original time series and is symmetric by main diagonal. Because of this feature, the polar coordinates can be reduced to the original time series according to their transformation principle.

2) MARKOV TRANSITION FIELD

The *MTF* can encode the one-dimensional time series signal into a two-dimensional image according to the Markov process, and the two-dimensional image encoded by this method can well retain the time dependent and frequency structure of the time series signal [16]. Assume that a time series $X = \{x_1, x_2, \ldots, x_N\}$, where $x_i (i \in \{1, 2, \ldots, N\})$ is divided in terms of value range, and each bin maps to a corresponding $q_j (j \in \{1, 2, \ldots, Q\})$. A Markov transition matrix W is constructed by calculating the transition between q_j in a first-order Markov chain, which is shown as follows:

$$W = \begin{bmatrix} w_{11} & \cdots & w_{1Q} \\ \vdots & \ddots & \vdots \\ w_{Q1} & \cdots & w_{QQ} \end{bmatrix} \tag{6}$$

where $w_{ij} (i, j \in \{1, 2, \ldots, Q\})$ is the transition probability between q_i to q_j, and there is $w_{ij} = P(x_t \in q_j | x_{t-1} \in q_i)$.

Markov matrix is extended through arranging each transition probability along with the temporal sequences, and a $N \times N$ *MTF* matrix is generated as follows:

$$\begin{aligned} M &= \begin{bmatrix} M_{11} & \cdots & M_{1N} \\ \vdots & \ddots & \vdots \\ M_{N1} & \cdots & M_{NN} \end{bmatrix} \\ &= \begin{bmatrix} W_{ij|x_1 \in q_i, x_1 \in q_j} & \cdots & W_{ij|x_1 \in q_i, x_N \in q_j} \\ \vdots & \ddots & \vdots \\ W_{ij|x_N \in q_i, x_1 \in q_j} & \cdots & W_{ij|x_N \in q_i, x_N \in q_j} \end{bmatrix} \end{aligned} \tag{7}$$

where $M_{ij} (i, j \in \{1, 2, \ldots, N\})$ refers to the transition probability from bins of x_i to x_j.

3) RECURRENCE PLOT

RP is an important method for analyzing time series periodicity, chaos, as well as non-stationarity, to reveal the internal structure of time series, giving prior knowledge about similarity, informative, and predictive properties. The *RP* maps the series to an m-dimensional phase space, and obtains a *RP* image through calculating the distance matrix in the phase space, it can be expressed as follows:

$$RP_{i,j}(\varepsilon) = \theta(\varepsilon - ||\vec{S_l} - \vec{S_m}), \vec{S(\cdot)} \in \mathcal{R}^n, l, m = 1, 2, \cdots, K \tag{8}$$

where ε is distance threshold, $||\cdot||$ is the norm, $\theta(\ldots)$ is the Heaviside function for the binarization of distance matrix, m is the phase space dimension, K is the state number of $\vec{S}$, and $RP_{i,j}$ is the (i, j) position pixel of the *RP* image.

In order to find distinguishable patterns in recursive images, we use a cycling degree of encoding time series states for effective preprocessing. Assume that T is a time series with N points, and the *RP* of T is denoted as R_T shown as follows:

$$R_{i,j} = f(T_i, T_j) \tag{9}$$

where $i, j \in \{1, 2, \cdots, n\}$, f refers to distance function for encoding time series T_i state, and the implementation of the f function is executed as follows:

$$R_{i,j} = |T_i - T_j| \tag{10}$$

4) MULTIMODING GMR

Based on the difference in coding methods on classification accuracy, it is found that *MTF* focuses on the dynamic change process of the time signal, which loses a large amount of static information. When the input size is too large, the *MTF* image becomes more blurred, so that the fault features disappear. *GAF* retains an amount of static information. Moreover, *RP* has serious problems such as tendency confusion. Three kinds of proposed images including *GAF* image (since the two methods of *GAF* are similar, we only chose the image generated by *GASF* to extract features.), *MTF* image and *RP* image are superimposed as three-channel data inputs into *ResNet*, enabling the network to obtain more complete multi-modal two-dimensional features F_{gmrRe} to enhance one-dimensional features. The *GMRResNet* network framework on a popular dataset *Chinatown* [14] is shown in FIGURE 2.

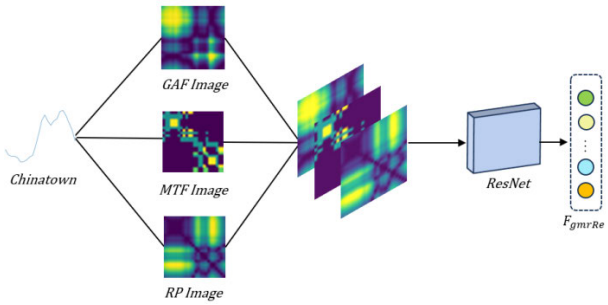


FIGURE 2. The *GMRResNet* network framework.

III. FUSION NETWORK ARCHITECTURE

The network framework of *MS-GMRResNet* is presented in FIGURE 3, where three steps are included: (i): multi-scale one-dimensional time features F_{msRe} are extracted using the *MSResNet* model. (ii): the two-dimensional features F_{gmrRe} are extracted using the *GMRResNet*. and (iii): F_{msRe} and F_{gmrRe} are fused in two different ways, including *SE* and *SA* feature fusion architecture, to finally realize the classification of fusion features.

Specifically, the *SE* feature fusion architecture inspired by the attention mechanism [17], we perform *SE* operations on

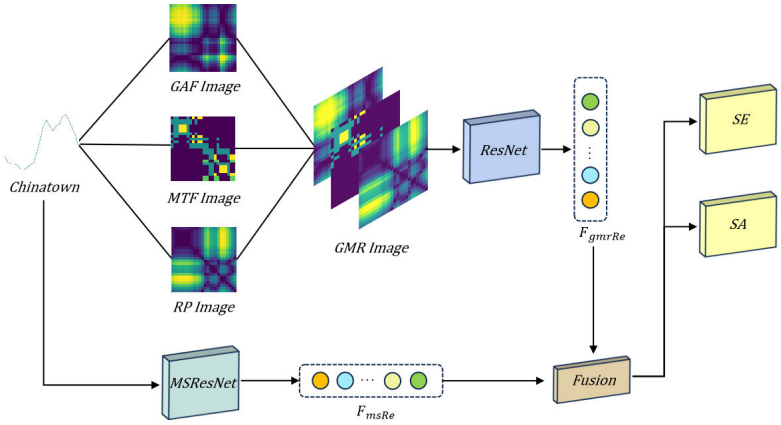


FIGURE 3. The *MS-GMRResNet* network architecture.

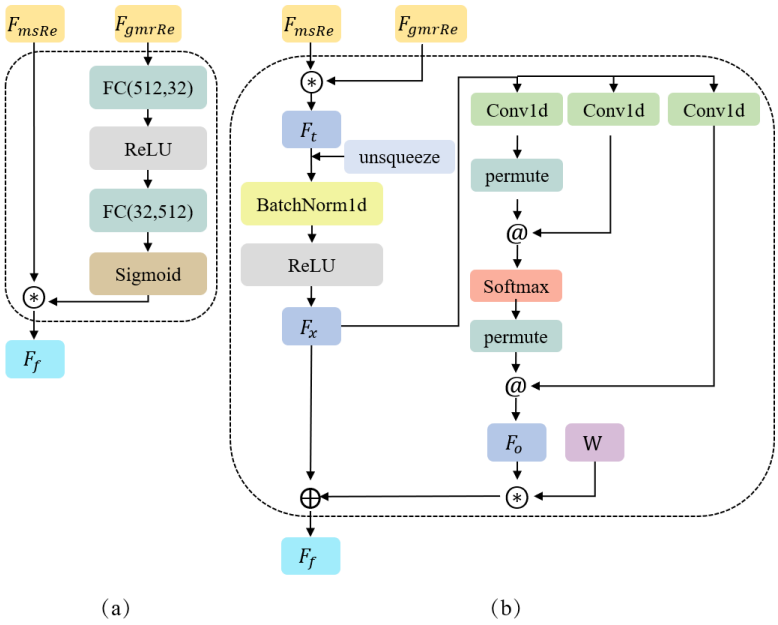


FIGURE 4. (a) Squeeze-and-Excitation Feature Fusion Architecture; (b) Self-Attention Feature Fusion Architecture.

two-dimensional features to get the weights of corresponding one-dimensional features and fuse them through taking the form of multiplying with one-dimensional features. For *SA* feature fusion architecture, both one-dimensional features and two-dimensional features are regarded as feature representations of the original series, where the self-attention mechanism in natural language processing can more effectively fuse these two features [18].

A. SQUEEZE-AND-EXCITATION FEATURE FUSION ARCHITECTURE

The *SE* fusion architecture aims to improve the quality of the fusion features by explicitly modeling the interdependence between the convolutional feature channels. Based on this mechanism, effective features can be selectively retained and suppress less useful features, enabling feature recalibration. The *SE* structure is shown in FIGURE 4 (a), and the calculation process is in Algorithm 1 which includes four following steps: (i) the preprocessed one-dimensional time series data *TS_1D* is input into the *MSResNet* model to obtain the one-dimensional feature F_{msRe}. Homoplastically, the three-channel two-dimensional image *TS_2D* preprocessed by *GMR* is input into the *ResNet* model to gain the two-dimensional feature F_{gmrRe} (lines 3-4). (ii) F_{msRe} is considered as an advanced feature which dominates feature fusion, so we perform *SE* on F_{gmrRe}. Fully connection layer (512, 32) performs squeeze operation on F_{gmrRe} and *ReLu* is used as a nonlinear activation function (line 6). (iii) excitation operation refers to the use of the full connection layer $FC(32, 512)$ to process the results of the previous squeeze operation(line 7). Then, the sigmoid function is used to obtain the weight of two-dimensional features, which is the output

Algorithm 1 Squeeze-and-Excitation Feature Fusion

Input: TS_1D: the preprocessed one-dimensional time series data; TS_2D: the preprocessed three-channel two-dimensional image.

Output: F_f;

1: Use Squeeze-and-Excitation to fuse one-dimensional and two-dimensional features;
2: Obtain the one-dimensional and two-dimensional features;
3: $F_{msRe} = MSResNet(TS\_1D)$
4: $F_{gmrRe} = ResNet34(TS\_2D)$
5: Perform squeeze and excitation operation on F_{gmrRe}, W_1 and W_2 refer to weighting parameters, b_1 and b_2 are bias item;
6: $tempS = reLu(W_1 \times F_{gmrRe} + b_1)$
7: $tempE = W_2 \times tempS + b_2$
8: $weight = sigmoid(tempE)$
9: $F_f = F_{msRe} \times (weight)$
10: **return** F_f

of the *SE* block(line 8). (iv) the obtained weights are directly multiplied by the one-dimensional features to obtain the fusion features F_f (line 9).

B. SELF-ATTENTION FEATURE FUSION ARCHITECTURE

Inspired by [18], we design a self-attention feature fusion architecture as shown by FIGURE 4 (b), and the calculation process is in Algorithm 2 which includes four following steps: (i) the preprocessed one-dimensional time series data TS_1D is input into the *MSResNet* model to obtain the one-dimensional feature F_{msRe}. Homoplastically, the three-channel two-dimensional image TS_2D preprocessed by *GMR* is input into the *ResNet* model to gain the two-dimensional feature F_{gmrRe} (lines 3-4). (ii) multiply one-dimensional feature F_{msRe} with two-dimensional feature F_{gmrRe} to obtain F_t, and rise dimension operation for F_t on $dim = 1$ and normalize and non-linear activation to obtain F_x (lines 5-7). (iii) perform one-dimensional convolution three times to obtain the query Q, key K and value V of self-attention (lines 9-11). (iv) the self-attention mechanism spreads around these three matrices (line 12), where QK^T is the most important interaction step in the self-attention mechanism, Q represents the encoded information, K represents the information of other time periods in the sequence after multiplying the weight value. Step (iv) is to obtain a fusion feature F_f, where W is a trainable weight parameter (line 13).

Algorithm 2 Self-Attention Feature Fusion

Input: TS_1D: the preprocessed one-dimensional time series data; TS_2D: the preprocessed three-channel two-dimensional image.

Output: F_f;

1: Use Self-Attention to fuse one-dimensional and two-dimensional features;
2: Obtain the one-dimensional and two-dimensional features and perform feature processing;
3: $F_{msRe} = MSResNet(TS\_1D)$
4: $F_{gmrRe} = ResNet34(TS\_2D)$
5: $F_t = F_{msRe} \times F_{gmrRe}$
6: $F_t = unsqueeze(F_t)$
7: $F_x = relu(bn(F_t))$
8: Calculate the query Q, Key K and value V to get the output of self-attention;
9: $Q = query\_conv(F_x)$
10: $K = key\_conv(F_x)$
11: $V = value\_conv(F_x)$
12: $F_o = softmax(QK^T)V$
13: $F_f = W \times F_o + F_x$
14: **return** F_f

IV. IMPLEMENTATION AND EVALUATION

A. EXPERIMENTAL SETTING

1) DATA AND PREPROCESSING

We select 84 complete datasets from *UCR* time series classification archive as experimental data [14]. *UCR* time series classification archive is the "ImageNet" of time series field, with 128 kinds of robust data covering many fields such as medical treatment, electricity, geography, etc. It involves many time series tasks such as prediction, regression, clustering, etc. *UCR* datasets are all numerical data and have been divided into training and test sets. Due to the varying time step length of the datasets, we interpolated the time series to a fixed length of 512 during preprocessing to facilitate the subsequent extraction and construction of two-dimensional features. In addition, we normalized the datasets because the data types of each sub-dataset are different, which leads to a tremendous gap in their actual value ranges, such as Sensor, Motion, etc.

2) EVALUATION INDICATOR

In this paper, the error rate is taken as the index to evaluate the model, representing the number of classification errors divided by the total number of samples, which is shown as follows:

$$acc = \frac{TP + TN}{P + N} \tag{11}$$

$$error = 1 - acc \tag{12}$$

where TP represents the number of positive samples predicted as positive samples, FP represents the number of negative samples predicted to be positive samples, FN represents the number of positive samples predicted to be negative samples, $P = TP + FN$ refers to the actual number of positive cases, and conversely, and $N = FP + TN$ denotes the actual number of negative examples.

3) EXPERIMENTAL ENVIRONMENT

In the experiment, the epoch is set to 50, the dropout is set to 0.8, and the batch size is set to 16. Adam optimizer is

selected as the optimizer, and the initial learning rate is set to $1e-5$. We use a desktop with a 2.50GHz CPU, a 64GB RAM, a 12GB video memory, and a NVIDIA RTX A2000 GPU, to conduct experimental evaluation.

B. EXPERIMENTAL RESULTS AND EVALUATION

1) EXPERIMENTAL RESULTS

The experimental results of the *MS-GMRResNet* network model proposed in this paper on 84 *UCR* datasets are shown in TABLE 1, and two competitive approaches are listed as follows:

- *MSResNet*/*GMRResNet*: classify directly without feature fusion, and the features of time series classifier are extracted using model *MSResNet* and model *GMRResNet* respectively.
- *GMR-SE*/*GMR-SA*/*GMR-MULT*: in the converged network architecture *MS-GMRResNet*, *GMR* is used to build two-dimensional features, while *SE* and *SA* are employed as feature fusion methods. At the same time, *MULT*iplication (*MULT*) is used as a feature fusion method for comparison.

We can see that the three-channel feature enhancement method of *GMR* has successfully reduced the average error rate in different fusion methods, of which the *SA* fusion method has the most significant effect, reducing the average error rate by 3.35%, and it has the most significant effect on the *Chinatown* dataset, reducing the error rate by 60.05%.

In the *C*ritical *D*ifference (*CD*) diagram of FIGURE 5, it is clear to see that *GMR-SA* achieve the best performance among several state-of-art approaches. In addition, the experiment also compares *ResNet*, *DTW* and other deep learning methods with traditional machine learning methods. We can see the effectiveness of two-dimensional feature enhancement and feature fusion for time series classification.

2) IMPACT OF TWO-DIMENSIONAL FEATURE CONSTRUCTION

The impact of different feature construction methods that transform one-dimensional to two-dimensional on the classification results are compared. The results are shown in TABLE 2, where column headers (*GMR/GAF/MTF/RP*) represent the different two-dimensional feature construction methods, and the row header has different meanings. Among them, *ResNet* refers to the direct classification of two-dimensional features without fusion, and 0.2305 in the first row and the first column corresponds to the *GMRResNet* proposed above. *SA*, *MULT*, and *SE* represent the different fusion methods of two-dimensional features and one-dimensional features extracted by *MSResNet*, and 0.2201 in the first row and the second column corresponds to the *GMR-SE* mentioned above. *SE-1D* represents the average error rate of a single dimension (*MSResNet*, 0.2316) minus the average error rate of the *SE* fusion method, and *MULT-1D*/*SA-1D* are the same. Ultimately, the cell's specific numerical value represents each model's average error rate on 84 datasets.

TABLE 1. Error rate of MS-GMRResNet network model on 84 datasets.

dataSet Name	MSResNet	GMRResNet	GMR-SE	GMR-SA	GMR-MULT
ECG5000	0.0602	0.0587	0.054	0.0602	0.0742
FaceAll	0.1059	0.2124	0.1888	0.1657	0.1609
MedicalImages	0.2474	0.2316	0.2487	0.2855	0.2697
InsWngSnd	0.3813	0.4	0.4106	0.3702	0.397
SyntheticCon	0.0067	0.3233	0	0	0
CricketX	0.2333	0.2744	0.341	0.3026	0.2667
ElectricDevices	0.2395	0.3054	0.2747	0.2701	0.2682
InsectEPGRegTra	0.0458	0.3735	0.1004	0.0522	0.0522
FreezerRegTra	0.0035	0.0102	0.0049	0.0112	0.0035
NonInvThor1	0.087	0.0784	0.0977	0.0992	0.1206
SmoothSub	0.06	0.1667	0.0133	0.02	0.0267
ItPwDmd	0.1224	0.1293	0.0466	0.1351	0.1147
Chinatown	**0.7259**	**0.1496**	**0.1691**	**0.137**	**0.1254**
Crop	0.2568	0.2845	0.2511	0.276	0.2593
SonyAIBO2	0.2938	0.2455	0.2497	0.1868	0.2267
SonyAIBO1	0.4293	0.4359	0.5707	0.4077	0.5541
DistPhxAgeGp	0.2446	0.2662	0.2518	0.2662	0.259
DistPhxCorr	0.2355	0.25	0.2283	0.2319	0.2101
MidPhxAgeGp	0.3831	0.4481	0.4221	0.4221	0.4351
MidPhxCorr	0.1684	0.1649	0.134	0.2405	0.1684
PhalCorr	0.1876	0.183	0.1667	0.1853	0.2028
ProxPhxAgeGp	0.161	0.1463	0.1415	0.1317	0.122
ProxPhxCorr	0.1306	0.1237	0.1271	0.1753	0.134
TwoLeadECG	0.1027	0.2335	0.0957	0.1449	0.1431
MoteStrain	0.2899	0.2788	0.2284	0.1725	0.135
CBF	0.1122	0.0022	0.0222	0	0.0544
SwedishLeaf	0.0657	0.0608	0.0976	0.0736	0.088
TwoPatterns	0.0023	0.0103	0	0	0
BME	0.2153	0	0.04	0	0.0133
FacesUCR	0.1294	0.1098	0.1498	0.1341	0.1498
ECGFiveDays	0.2983	0.5238	0.1742	0.23	0.1614
Plane	0.0313	0.0476	0.019	0.0095	0.0381
PowerCons	0.0625	0	0.0333	0.0056	0
GunPoint	0.1111	0.02	0.0267	0.0133	0.0333
GunPointAgeSp	0.0263	0.0443	0.0222	0.0443	0.0253
GunPointMVsF	0.0033	0.0063	0.0095	0.0127	0.0032
GunPointOVsY	0.0132	0	0	0	0
UMD	0.1806	0.3194	0.0833	0.0764	0
ChloConc	0.331	0.374	0.412	0.4286	0.4164
Adiac	0.2995	0.3146	0.4476	0.2839	0.3862
Fungi	0.6534	0.3118	0.5161	0.3495	0.371
Wine	0.4167	0.3333	0.3148	0.3704	0.5
Strawberry	0.0462	0.0622	0.0649	0.0757	0.0568
ArrowHead	0.25	0.3029	0.3371	0.2971	0.2629
FiftyWords	0.25	0.233	0.2813	0.2308	0.2571
WordSynonyms	0.3413	0.3354	0.3887	0.3464	0.3793
Trace	0.0417	0	0	0	0
ToeSeg1	0.1563	0.114	0.1096	0.2325	0.307
Coffee	0.25	0	0.0714	0	0
CricketY	0.2474	0.3256	0.2615	0.2513	0.2846
CricketZ	0.2318	0.3077	0.2385	0.2256	0.2282
FreezerSmlTra	0.0643	0.0098	0.073	0.1284	0.0898
UWaveGestLibX	0.1923	0.3155	0.1857	0.1926	0.1681
UWaveGestLibY	0.2968	0.311	0.2901	0.2878	0.2984
UWaveGestLibZ	0.2671	0.3132	0.27	0.2482	0.2736
Lightning7	0.3906	0.3288	0.411	0.411	0.3151
ToeSeg2	0.2188	0.3923	0.0692	0.1154	0.2846
DiatomSizeRed	0.6118	0.0425	0.2778	0.0523	0.098
FaceFour	0.1375	0.4886	0.2273	0.5682	0.4091
Symbols	0.1038	0.392	0.2402	0.2442	0.2965
Yoga	0.1888	0.335	0.1863	0.2073	0.1987
OSULeaf	0.2458	0.2934	0.281	0.1612	0.2769
Ham	0.3021	0.2476	0.3143	0.2762	0.2571
Meat	0.2708	0.2167	0.4167	0.1333	0.15
Fish	0.0875	0.2343	0.1257	0.0743	0.0743
Beef	0.375	0.5	0.5667	0.5667	0.8
ShapeletSim	0.3807	0	0.3111	0	0
BeetleFly	0.5625	0.1	0.5	0.3	0.35
BirdChicken	0.4375	0.2	0.25	0.15	0.3
Earthquakes	0.2578	0.2518	0.259	0.2806	0.3022
Herring	0.5156	0.5313	0.5313	0.4688	0.5156
ShapesAll	0.1767	0.2	0.2283	0.175	0.2933
OliveOil	0.6	0.3333	0.3333	0.3667	0.3333
Car	0.15	0.4333	0.3333	0.1833	0.35
InsectEPGSmlTra	0.3976	0.494	0.6426	0.3534	0.6546
Computers	0.24	0.264	0.284	0.312	0.324
LrgKitApp	0.1467	0.1307	0.16	0.1013	0.1307
RefDev	0.48	0.4933	0.472	0.4373	0.4693
ScreenType	0.5307	0.4693	0.5387	0.5387	0.5733
SmlKitApp	0.248	0.24	0.2293	0.216	0.2373
NonInvThor2	0.0651	0.088	0.0931	0.0702	0.0723
Worms	0.2727	0.4026	0.3766	0.2857	0.3636
WormsTwoClass	0.2338	0.2597	0.2078	0.2727	0.2597
UWaveGestLibAll	0.038	0.2149	0.0248	0.0226	0.0348
Average Value	**0.2316**	**0.2305**	**0.222**	**0.1981**	**0.2179**

The results of ablation experiments show that our proposed *GMR*, a multimodal three-channel coding method, achieves the best performance and has successfully reduced the average error rate in different methods, among which the *SA* model obtained the best result with a reduction of 3.35%. In terms of *GAF*, *MTF* and *RP*, which are three

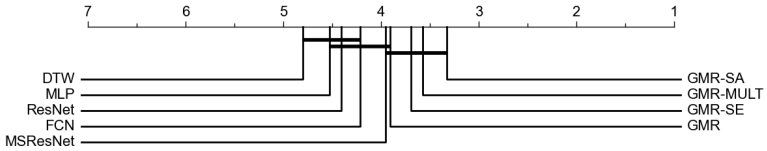


FIGURE 5. The *CD* diagram of the competitive approaches and our proposed models; the thick horizontal lines in the diagram indicate a cluster of classifiers that are not significantly different in terms of classification performance.

TABLE 2. Performance comparison of four encoding methods on *84* datasets.

	ResNet	SE	MULT	SA	SE-1D	MULT-1D	SA-1D
GMR	0.2305	0.2201	0.2179	0.1981	**0.0116**	**0.0138**	**0.0335**
GAF	0.2431	0.2324	0.2189	0.2092	-0.0008	0.0127	0.0224
MTF	0.2863	0.2398	0.2409	0.242	-0.0081	-0.0092	-0.0103
RP	0.2543	0.2349	0.2354	0.2216	-0.0032	-0.0037	0.01

TABLE 3. Performance comparison of three feature fusion methods on 84 datasets.

	SE-1D	MULT-1D	SA-1D	SE+	MULT+	SA+
GMR	0.0116	0.0138	0.0335	52	48	**56**
GAF	-0.0008	0.0127	0.0224	48	47	**57**
MTF	-0.0081	-0.0092	-0.0103	**48**	38	33
RP	-0.0032	-0.0037	0.01	49	46	**56**

single methods to transform one-dimensional time series into two-dimensional images, *GAF* performs relatively best, second only to three-channel *GMR*. The disappointing thing is that the *MTF* method cannot improve the accuracy of classification, and the strategy of employing *RP* to achieve feature enhancement and then integrating features through Self-Attention reduced the error rate by 1%.

3) IMPACT OF THE FUSION MODE

It can be inferred from TABLE 3 that *SA* fusion architecture shows positive effects on different two-dimensional feature construction methods. Correspondingly, no matter which two-dimensional features extraction method is used, the number of datasets that reduce the average error rate by using the fusion model of *SE* and *SA* proposed in this paper is more than the number of datasets that reduce the average error rate by simply multiplying the features directly. When using three-channel *GMR* for feature enhancement, the *SA* feature fusion method proposed in this paper has achieved better results than single dimension on 56 datasets (*SA*+) and reduced the average error rate by 3.35%. In addition, *SE* is comparable to direct multiplication in terms of average error rate, but *SE*+ reduces the error rate for 52 datasets (*SE*+), which is slightly better than simply fusing by direct multiplication.

4) FEATURE VISUALIZATION

Grad-CAM, as a generalized form of *C*lass *A*ctivation *M*apping (*CAM*), provides a natural way to find the contribution region of a specific label in the original data and highlight the discriminant subsequence detected by the convolutional network, which can be applied to the network structure of all *CNN* systems [19].

Suppose that the convolution layer of the last layer in a *CNN* network has n convolution kernels, and the size of the feature map corresponding to each convolution kernel is $u \times v$, so the *Grad-CAM* for a specific category c is recorded as $L^c_{Grad-CAM} \in R^{u \times v}$. Define the activation value of the kth feature map as A^k, and the output score of category c as y^c (before softmax). Then the weight value a^c_k of the kth feature map for category c can be obtained as follows:

$$a^c_k = \frac{1}{u \times v} \sum_{i=1}^{u} \sum_{j=1}^{v} \frac{\partial y^c}{\partial A^k_{ij}} \tag{13}$$

where A^k_{ij} represents the pixel value of the pixel point whose coordinate is (i, j) in the kth feature map. Equation (13) represents that the contribution weight of each feature map to category c is obtained by calculating the average gradient of the output score of category c to each feature map in the convolution layer.

Then, as shown in Equation (14), the weighted sum of n feature maps is first completed, and then through *ReLU* activation function, the class activation heat map is finally obtained as follows:

$$L^c_{Grad-CAM} = ReLU(\sum_{k=1}^{n} a^c_k A^k) \tag{14}$$

In this article, we examined two examples of using *Grade-CAM*. As shown in FIGURE 6, the identification areas of the time series with correct classification are highlighted, and the contribution areas of different categories are different. Label 0 is mainly determined by the region where a sharp drop followed by a rise and a slow fall occurs. For label 1, *MS-GMRResNet* not only focuses on the flat part in the middle of the sequence but also extracts valuable features from the data at the beginning and end, which fall and then rise. The sequence with label 2 has a characteristic pattern of a sharp decline and then a sharp rise followed by a smooth trend. For label 3, the neural network only focuses on the middle gentle part of the time series. Similar results can be obtained on the *Chinatown* dataset.

FIGURE 7 compares the *Grad-CAM* of *MSResNet* with that of *MS-GMRResNet*. The figure shows that *MS-GMRRResNet* has more highlighted areas. In other

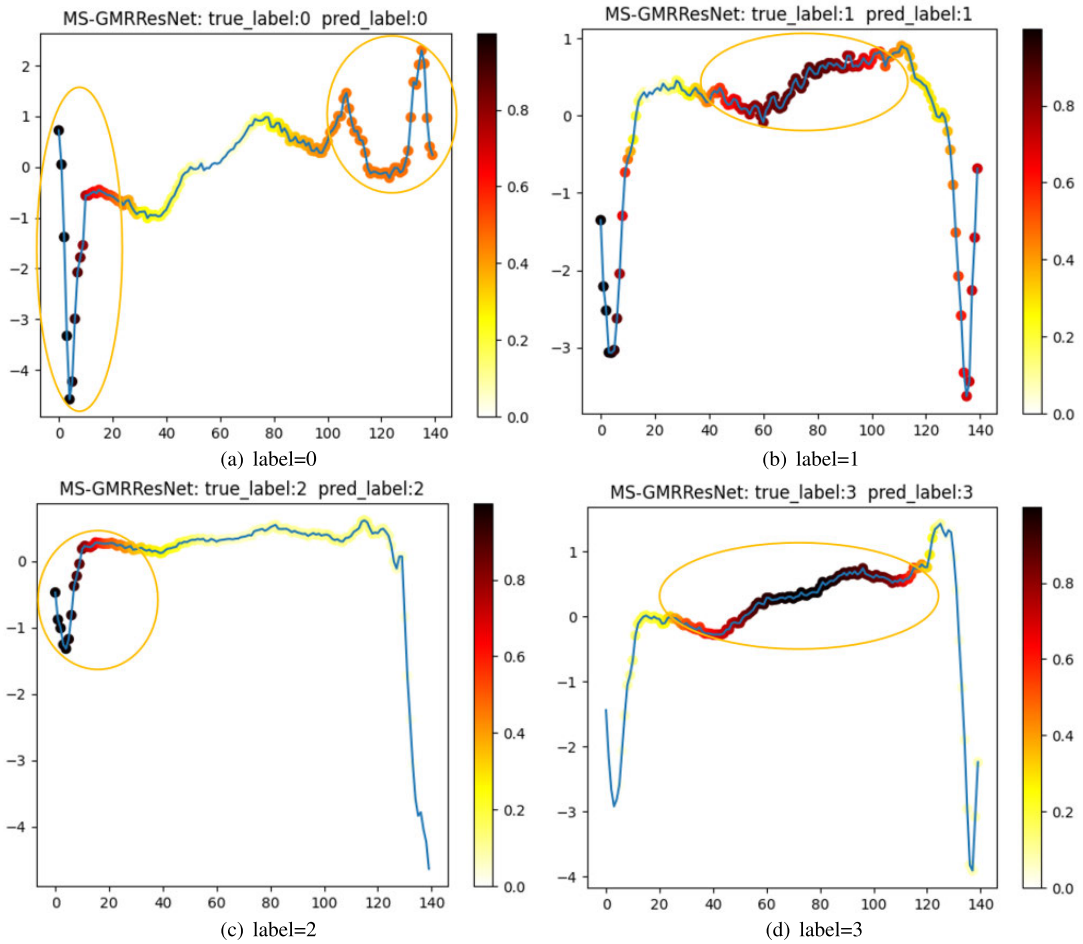


FIGURE 6. *Grad-CAM* diagram of *MS-GMRResNet* on *ECG5000* dataset.

words, the *Grad-CAM* of *MSResNet* contains more white, cyan, and yellow areas, while the *Grad-CAM* of *MS-GMResNet* contains more dark red and black subsequences, which indicates that two-dimensional features can effectively enhance one-dimensional features and may explain why *MS-GMResNet* is more effective than *MSResNet*.

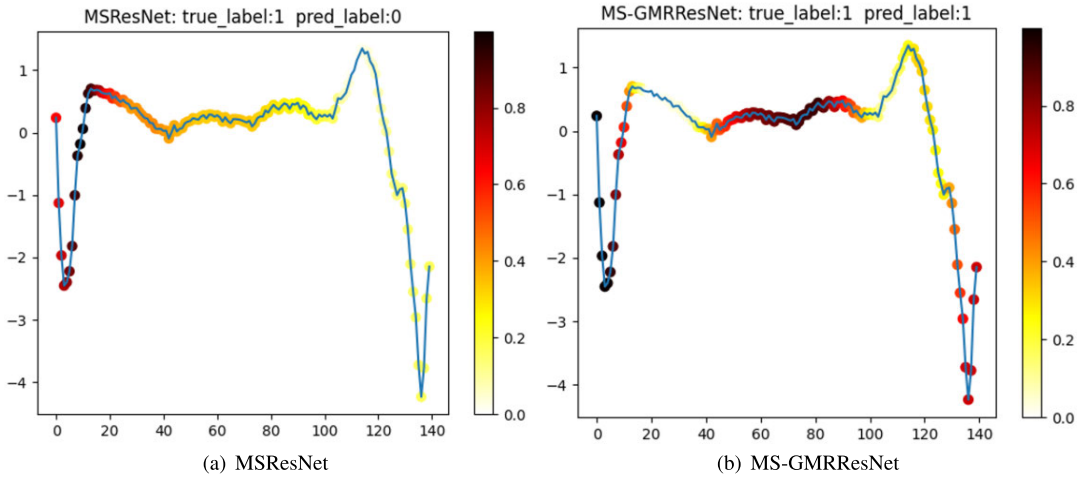
V. RELATED WORKS

In order to classify time series data accurately, researchers have proposed many methods to solve this problem. Currently, from the perspective of features, the classification methods for time series data can be broadly categorized as methods based on one-dimensional features, methods based on two-dimensional features and methods based on feature fusion.

A. METHODS BASED ON ONE-DIMENSIONAL FEATURES

The classification method based on one-dimensional features mainly includes three directions: explicit design of features, end-to-end deep learning, and integrated learning. Distance-based methods reflect the similarity between time series by calculating the distance. The degree of resemblance increases with decreasing distance. On the other hand, the resemblance decreases with increasing distance. Among distance-based methods, the simplest is calculating the *E*uclidean *D*istance (*ED*) between various time series [20]. Although this method is uncomplicated and expeditious, it also has some disadvantages, such as when there are displacements and deformations between time series, the method of Euclidean distance will produce significant errors, which contributes to its infrequent use. *DTW* [21] is a more reasonable method to compare the similarity of time series with the same length. However, the very low efficiency of full search mode makes *DTW* consume a lot of memory in practical application scenarios, which makes it not feasible.

Since time series are one-dimensional signals, they are usually classified by deep learning methods such as one-dimensional *CNN* and *RNN* networks. Among them, *FCN* and *ResNet* show competitive performance in *TSC* tasks under their excellent adaptation to the original time series and adaptive feature extraction. Cui et al. [22] proposed a multi-scale convolutional neural network to classify time series, considering that time series can extract different features on different time scales. Wang et al. [5] proposed *MLP*,



(a) MSResNet (b) MS-GMRResNet

FIGURE 7. ***Grad-CAM*** **comparison of** ***MSResNet*** **and** ***MS-GMRResNet*** **on** ***ECG5000*** **dataset.**

FCN and *ResNet* as the baseline architecture of *TSC*, which is the most traditional form. Then various variants of *CNN* were proposed one after another by directly using time series data as input. Among them, Chen et al. [23] studied a *MACNN* model to solve the *TSC* problem, which has achieved good results. They first captured information at different scales along the time axis by generating feature maps at different scales and then proposed an attention mechanism to enhance useful feature maps by automatically learning the importance of each feature map.

The method based on integrated learning is mainly *COTE* [24], which integrates 35 traditional time series classification models. Lines and others extended the *COTE* structure to *HIVE-COTE* [25] that requires training 37 classifiers and cross-verifying each super parameter of these algorithms by using a new hierarchical structure with probability voting, which makes it impractical to run on real big data mining problems. Recently, some researchers have integrated 60 neural networks for time series classification [26] and Shifaz et al. proposed an extensible precise forest algorithm [27].

Although the above methods improve the accuracy of *TSC* to a certain extent, they have certain defects. For example, the nearest neighbour classification based on distance needs to calculate the distance between time series, which has low performance. Furthermore, considering that the ensemble model needs multiple classifiers, it has high modelling complexity, high computational complexity, and extended training time.

B. METHODS BASED ON TWO-DIMENSIONAL FEATURES

Converting the original one-dimensional signal into two-dimensional image data has aroused extensive research. Thereinto, *GAF*, *MTF* and *RP* are widely used as common coding methods. For example, in order to obtain more complete fault characteristics, Han et al. [28] encoded the original vibration sensor signal into a two-dimensional image through *GAF* and *MTF* and input the *GAF* image and *MTF* image into the capsule network as a dual channel image. They also discussed the impact of two coding methods and different network structures on diagnostic accuracy. Additionally, Wang et al. [16] used *MTF* to convert the original time series into corresponding graphs and used *CNN* to extract the deep feature information in the graph to complete the rolling bearing fault diagnosis.

What's more, Hatami et al. [29] trained *CNN* on *RP* images obtained by encoding time series into images through *RP* for classification, achieving the best performance among the representative sequence-to-image methods. Recently, for the sake of circumventing the defects of *RP*, Zhang et al. [13] proposed the *MSRP* including three components, namely multi-scale *RP*, asymmetric *RP* and symbolic *RP*. Firstly, they used multi-scale *RP* to enrich the image scale and then constructed asymmetric *RP* to represent long sequences. Finally, they multiplied the designed symbol masks to get symbol *RP* images to remove trend confusion. Experimental results on 85 *UCR* datasets demonstrated the superior performance of *MSRP-IFCN*.

The idea of computer vision feature extraction serves as a model for the conversion of time series to images. In fact, this image-based framework has given rise to a new subset of depth learning techniques that considers image transformation to be a form of feature engineering. The convolutional network method reduces the time information loss and the issue that the feature is no longer time invariant faced by the conventional multi-layer perception method by learning the space invariant filter (or feature) from the original input time series.

C. CLASSIFICATION METHOD BASED ON FEATURE FUSION

The method based on feature fusion obtains better accuracy by extracting multiple types of feature information. For instance, the *LSTM-FCN* and *GRU-FCN* fusion networks learn spatial and temporal characteristics simultaneously and

achieve good results [30]. In recent research, Zhao et al. [31] proposed a multimodal network *LSTM-MFCN* consisting of multi-scale *FCN* that can perceive spatial characteristics in different ranges from time series curves because of its multi-scale filter banks and gate-based network *LSTM*, which is naturally suitable for various time dependencies. *LSTM-MFCN*'s comprehensive perception of large-scale multi-scale spatiotemporal features enables it to have a complete and thorough grasp of time series to obtain better accuracy.

It is crucial to combine features from various scales in order to enhance classification performance. Low-level features contain more detailed information, but less convolution means lower semantics and more noise. On the contrary, high-level features have stronger semantic information, but their perception of details is poor. Effectively integrating the two is the key to improving the model's performance.

It is divided into early fusion and late fusion based on the order of fusion and prediction. Early fusion involves first fusing multi-layer features, after which the predictor is trained using the fusing features. Skip connection is another name for this kind of technique. There are two traditional techniques for feature fusion: (1) *Concatenation*: the direct joining of two characteristics in a sequence. (2) *Addition*: a parallel strategy which combines these two eigenvectors into a complex vector.

Feature fusion is a common technique in deep learning. Numerous network models have used *Concatenation* and *Addition* operations particularly. For instance, while *Inception* and *DensNet* [32] use the *Concatenation* operation to combine feature maps, *ResNet* and *FPN* [33] both use the *Addition* operation to fuse feature maps. In most cases, the two fusion methods are effective.

But for two complicated or even semantically distinct features, a simple *Concatenation* and *Addition* operation is not the best option. Dai et al. [34] proposed attention feature fusion, which offers a unified and general method for feature fusion and is applicable to various networks, such as *Inception* and *FPN*, in order to deal with different fusion scenarios and produce better features. The model's performance can be enhanced by attentional feature fusion, according to experimental findings. In order to improve the original features, Pang et al. [35] proposed the Libra *R-CNN*, a fully balanced target detector that uses integrated balanced semantic features. In this way, the information flow is balanced and features are given more differentiation because each resolution in the pyramid receives the same information from other resolutions. In the *MS COCO* dataset, this method has seen a significant improvement.

VI. CONCLUSION

In this paper, a time series classification method based on two-dimensional feature enhancement for one-dimensional time series is proposed. We studied the impact of different feature construction and fusion methods on classification performance, and carried out a series of ablation and comparison experiments. The experimental results on 84 *UCR* datasets show that the proposed *MS-GMResNet* architecture can effectively improve the classification results. The features extracted from two-dimensional coding can be used as a supplement to the original one-dimensional time series data, thus effectively reducing the error rate of time series classification. The selection of a single coding method does not significantly affect the classification results, but the superposition of the three coding methods significantly reduces the average error rate. In addition, we use *Grad-CAM* to visually analyze the classification results of different data set, which further proves the effectiveness of the proposed method.

REFERENCES

[1] L. Feremans, B. Cule, and B. Goethals, "PETSC: Pattern-based embedding for time series classification," *Data Mining Knowl. Discovery*, vol. 36, no. 3, pp. 1015–1061, May 2022.

[2] K. M. Leon-Lopez, F. Mouret, H. Arguello, and J.-Y. Tourneret, "Anomaly detection and classification in multispectral time series based on hidden Markov models," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 5402311.

[3] Z. Chen and J. Gu, "High-throughput dynamic time warping accelerator for time-series classification with pipelined mixed-signal time-domain computing," *IEEE J. Solid-State Circuits*, vol. 56, no. 2, pp. 624–635, Feb. 2021.

[4] H. Zhao, Z. Pan, and W. Tao, "Regularized shapelet learning for scalable time series classification," *Comput. Netw.*, vol. 173, May 2020, Art. no. 107171.

[5] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2017, pp. 1578–1585.

[6] P. Liu, X. Sun, Y. Han, Z. He, W. Zhang, and C. Wu, "Arrhythmia classification of LSTM autoencoder based on time series anomaly detection," *Biomed. Signal Process. Control*, vol. 71, Jan. 2022, Art. no. 103228.

[7] F. Karim, S. Majumdar, H. Darabi, and S. Chen, "LSTM fully convolutional networks for time series classification," *IEEE Access*, vol. 6, pp. 1662–1669, 2017.

[8] S. Xu, J. Li, K. Liu, and L. Wu, "A parallel GRU recurrent network model and its application to multi-channel time-varying signal classification," *IEEE Access*, vol. 7, pp. 118739–118748, 2019.

[9] J.-H. Chen and Y.-C. Tsai, "Encoding candlesticks as images for pattern classification using convolutional neural networks," *Financial Innov.*, vol. 6, no. 1, pp. 1–19, Dec. 2020.

[10] H. Cai, L. Xu, J. Xu, Z. Xiong, and C. Zhu, "Electrocardiogram signal classification based on mix time-series imaging," *Electronics*, vol. 11, no. 13, p. 1991, Jun. 2022.

[11] M. Rußwurm and M. Körner, "Self-attention for raw optical satellite time series classification," *ISPRS J. Photogramm. Remote Sens.*, vol. 169, pp. 421–435, Nov. 2020.

[12] Z. Wang and T. Oates, "Imaging time-series to improve classification and imputation," *CoRR*, vol. abs/1506.00327, pp. 3939–3945, 2015.

[13] Y. Zhang, Y. Hou, K. OuYang, and S. Zhou, "Multi-scale signed recurrence plot based time series classification using inception architectural networks," *Pattern Recognit.*, vol. 123, Mar. 2022, Art. no. 108385.

[14] H. A. Dau, E. Keogh, K. Kamgar, C.-C. M. Yeh, Y. Zhu, S. Gharghabi, C. A. Ratanamahatana, Yanping, B. Hu, N. Begum, A. Bagnall, A. Mueen, and G. Batista. (2018). *The UCR Time Series Classification Archive*. [Online]. Available: https://www.cs.ucr.edu/~eamonn/time_series_data_2018/

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[16] M. Wang, W. Wang, X. Zhang, and H. H.-C. Iu, "A new fault diagnosis of rolling bearing based on Markov transition field and CNN," *Entropy*, vol. 24, no. 6, p. 751, 2022.

[17] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, "Squeeze-and-excitation networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 8, pp. 2011–2023, Aug. 2020.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 30, 2017, pp. 5998–6008.
[19] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," *Int. J. Comput. Vis.*, vol. 128, no. 2, pp. 336–359, Oct. 2019.
[20] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos, "Fast subsequence matching in time-series databases," *ACM SIGMOD Rec.*, vol. 23, no. 2, pp. 419–429, Jun. 1994.
[21] D. F. Silva, R. Giusti, E. Keogh, and G. E. A. P. A. Batista, "Speeding up similarity search under dynamic time warping by pruning unpromising alignments," *Data Mining Knowl. Discovery*, vol. 32, no. 4, pp. 988–1016, Jul. 2018.
[22] Z. Cui, W. Chen, and Y. Chen, "Multi-scale convolutional neural networks for time series classification," *CoRR*, vol. abs/1603.06995, pp. 126–140, 2016.
[23] W. Chen and K. Shi, "Multi-scale attention convolutional neural network for time series classification," *Neural Netw.*, vol. 136, pp. 126–140, Apr. 2021.
[24] A. Bagnall, J. Lines, J. Hills, and A. Bostrom, "Time-series classification with COTE: The collective of transformation-based ensembles," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 9, pp. 2522–2535, Sep. 2015.
[25] J. Lines, S. Taylor, and A. Bagnall, "Time series classification with hive-cote: The hierarchical vote collective of transformation-based ensembles," *ACM Trans. Knowl. Discovery Data*, vol. 12, no. 5, pp. 1–35, 2018.
[26] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Müller, "Deep neural network ensembles for time series classification," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2019, pp. 1–6.
[27] A. Shifaz, C. Pelletier, F. Petitjean, and G. I. Webb, "TS-CHIEF: A scalable and accurate forest algorithm for time series classification," *Data Mining Knowl. Discovery*, vol. 34, no. 3, pp. 742–775, May 2020.
[28] B. Han, H. Zhang, M. Sun, and F. Wu, "A new bearing fault diagnosis method based on capsule network and Markov transition field/Gramian angular field," *Sensors*, vol. 21, no. 22, p. 7762, Nov. 2021.
[29] J. Debayle, N. Hatami, and Y. Gavet, "Classification of time-series images using deep convolutional neural networks," in *Proc. 10th Int. Conf. Mach. Vis. (ICMV)*, Apr. 2018, pp. 242–249.
[30] F. Karim, S. Majumdar, and H. Darabi, "Insights into LSTM fully convolutional networks for time series classification," *IEEE Access*, vol. 7, pp. 67718–67725, 2019.
[31] L. Zhao, C. Mo, J. Ma, Z. Chen, and C. Yao, "LSTM-MFCN: A time series classifier based on multi-scale spatial–temporal features," *Comput. Commun.*, vol. 182, pp. 52–59, Jan. 2022.
[32] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.
[33] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2117–2125.
[34] Y. Dai, F. Gieseke, S. Oehmcke, Y. Wu, and K. Barnard, "Attentional feature fusion," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2021, pp. 3559–3568.
[35] J. Pang, K. Chen, J. Shi, H. Feng, W. Ouyang, and D. Lin, "Libra R-CNN: Towards balanced learning for object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 821–830.

SHUO QUAN received the B.S. and M.S. degrees from the School of Software, Beijing University of Posts and Telecommunications, Beijing, China, in 2013 and 2016, respectively. He is currently an Engineer with the China Telecom Research Institute, Beijing. His current research interests include deep learning, big data, and cloud computing.

MENGYU SUN received the B.S. degree from the Beijing University of Technology, Beijing, China, in 2016, and the Ph.D. degree from the China University of Geosciences, Beijing, in 2022. She is currently an Engineer with the China Telecom Research Institute, Beijing. Her research interests include cloud computing, service computing, and deep learning.

XIANGYU ZENG received the B.E. degree in software engineering from the Chengdu University of Technology, Chengdu, China, in 2022. Currently, she majors in computer science and technology with the University of Electronic Science and Technology of China, Chengdu.

XULIANG WANG received the B.S. degree from Beijing Normal University, Zhuhai, China, in 2010, and the M.S. degree from the Nara Institute of Science and Technology, Japan, in 2016. He is currently a Senior Engineer with the China Telecom Research Institute, Beijing, China. His research interests include cloud computing, cloud native, and edge computing.

ZEYA ZHU received the B.S. degree in telecommunication engineering and management from the Beijing University of Posts and Telecommunications, Beijing, China, in 2018, and the M.S. degree in big data science from the School of Electronic Engineering and Computer Science, Queen Mary University of London, U.K., in 2019. He is currently an Engineer with the China Telecom Research Institute, Beijing. His research interests include cloud computing, cloud native, and big data.