Neural Processing Letters (2024) 56:173
https://doi.org/10.1007/s11063-024-11449-8

# Time Series Classification Based on Forward Echo State Convolution Network

**Lei Xia[1,5] · Jianfeng Tang[1] · Guangli Li[1] · Jun Fu[1] · Shukai Duan[1,2,3,4] · Lidan Wang[1,2,3,4,5]**

Accepted: 27 November 2023 / Published online: 11 May 2024

## Abstract
The Echo state network (ESN) is an efficient recurrent neural network that has achieved good results in time series prediction tasks. Still, its application in time series classification tasks has yet to develop fully. In this study, we work on the time series classification problem based on echo state networks. We propose a new framework called forward echo state convolutional network (FESCN). It consists of two parts, the encoder and the decoder, where the encoder part is composed of a forward topology echo state network (FT-ESN), and the decoder part mainly consists of a convolutional layer and a max-pooling layer. We apply the proposed network framework to the univariate time series dataset UCR and compare it with six traditional methods and four neural network models. The experimental findings demonstrate that FESCN outperforms other methods in terms of overall classification accuracy. Additionally, we investigated the impact of reservoir size on network performance and observed that the optimal classification results were obtained when the reservoir size was set to 32. Finally, we investigated the performance of the network under noise interference, and the results show that FESCN has a more stable network performance compared to EMN (echo memory network).

**Keywords** Echo state network · Time series classification · Forward topology echo state network · Forward echo state convolution network

✉ Lidan Wang
ldwang@swu.edu.cn

1 College of Artificial Intelligence, Southwest University, Chongqing 400715, China

2 National & Local Joint Engineering Research Center of Intelligent Transmission and Control Technology, Chongqing 400715, China

3 Chongqing Key Laboratory of Brain-Inspired Computing and Intelligent Chips, Chongqing 400715, China

4 Key Laboratory of Luminescence Analysis and Molecular Sensing, Ministry of Education, Southwest University, Chongqing 400715, China

5 State Key Laboratory of Intelligent Vehicle Safety Technology, College of Artificial Intelligence, Southwest University, Chongqing 400715, China

## 1 Introduction

In order to solve the problem of high computational cost and low training efficiency of recurrent neural networks (RNN), several RNN variants have emerged, and ESN is one of them. ESN is a new type of neural network proposed by Jaeger [1] in 2001. It not only overcomes the computational complexity, training inefficiency, and difficulty of the practical application of RNN but also avoids the problem of locally optimal solutions. ESN mimics the structure of recursively connected neuron circuits in the brain and consists of an input layer, an implicit layer (or called reservoir), and an output layer. Among them, the hidden layer is a reservoir composed of large-scale random, sparsely connected neurons, which maps the low-dimensional input signals to the high-dimensional state space and has the ability to memorize and store the dynamic performance of the system through the weights between the neurons in the reservoir. Generating the reservoir and training the ESN are independent processes. As a result, only the weights from the reservoir to the output layer need to be trained using a linear method, which simplifies the training process of the network and avoids the complex training algorithms and the tendency to fall into local minima [2] that are common in traditional neural networks. Traditional ESNs use sparse connections between neurons in the reservoir, which gives them excellent short-term memory [3]. By fully utilizing this short-term memory capability as well as its high-dimensional nonlinear mapping ability, traditional ESNs exhibit very impressive performance in time series prediction tasks.

Due to the remarkable results achieved by ESNs in time series forecasting, researchers began to apply them to time series classification problems. In early research, there existed two main basic approaches to the problem. The first approach was to create a separate model for each category, which was trained and parameterized to enable it to accurately predict the data in the corresponding category. New data is then categorized into the category represented by the best matching model by predicting it and comparing the predictions to the individual models. For example, Skowronski and Harris proposed a predictive ESN classifier [4] for speech classification. The second method classifies the data from each time step independently to obtain a series of predictions. Then, by averaging these predictions, a composite prediction of the entire time series is obtained. For example, in Verstraeten et al. [5], the authors classified 10-digit speech signals by constructing ten one-to-many classifiers. It is easy to realize that both types of methods have some drawbacks. The first class of methods essentially utilizes a predictive model to deal with the problem, and it is unable to map the temporal signals to the class labels directly. The second class of methods does not take into account information about the entire time series and gives more weight to the end series with continued training. Tanisaro and Heidemann [6] proposed a method called Time-Warped Invariant Echo State Network (TWIESN). The method predicts the category of each time series element by training a Ridge classifier [7]. In the testing phase, the trained Ridge classifier outputs probability distributions for all categories in the dataset. The posterior probabilities of each category are then averaged to assign labels with the highest average probability to the input test time series. Until 2019, Ma's team combined traditional ESNs with convolutional neural networks (CNNs) to propose a network framework called EMN [8]. This framework fully utilizes the advantages of ESNs and CNNs and achieves excellent results in time series classification tasks.

In traditional ESNs, the neurons within the reservoir are randomly connected to each other, and the connection weights are also randomly generated. It is this randomness that can have some negative impact on the performance of the network. In order to tackle the issue of network instability arising from randomness and enhance classification performance, we present a novel variant of ESN known as the Forward Topology Echo State Network. This new topology aims to provide improved stability and superior classification capabilities. By combining the FT-ESN with CNN, we have developed a cutting-edge network framework known as the Forward Echo State convolution Network. This innovative architecture leverages the strengths of both models to enhance memory retention and improve overall performance in various tasks. The network framework is divided into two parts: encoding and decoding. In the encoding stage, we use FT-ESN to model the time series and output rich echo states, then collect all the states in a matrix. In the decoding phase, we extract features using convolution and max-pooling operations, which are fully connected and then fed into softmax for classification. Our specific contributions are as follows:

(1) To address the randomness problem of traditional ESNs, we propose a network with a fixed topology, i.e., a forward topology echo state network (FT-ESN).
(2) The readout layer of a traditional ESN is a simple linear readout layer. Instead of a linear readout layer, we utilize a CNN and a maximal pooling layer as the main structure, i.e., we combine the FT-ESN with the CNN and propose a new network framework, i.e., the forward echo state convolutional network (FESCN).
(3) The FESCN model achieves good results in the time series classification task on the UCR dataset and outperforms EMN in noise experiments.

The remaining sections of the paper are structured as follows: Sect. 2 provides a comprehensive introduction to the ESN, explaining its principles and functioning in detail. Section 3 delves into the related work conducted in this field, highlighting the existing research and approaches that have influenced our work. Section 4 outlines our proposed network architecture, known as the FESCN. This section provides an in-depth description of FESCN's design, components, and mechanisms. Moving on to Sect. 4, we present three experiments directly associated with FESCN. Finally, in Sect. 5, we conclude the paper by summarizing the essential findings and contributions discussed throughout the entire study.

## 2 Related Work

### 2.1 Improvement of ESN's Output Layer

ESNs have proven to be effective in various dynamic tasks due to their ability to effectively model temporal data. They offer several advantages, such as high training efficiency and low training cost, making them a preferred choice for many applications. However, the output of traditional ESNs is a simple linear output, and the network decoding ability is weak [9–11] when performing classification tasks, which limits the classification performance of the network. In recent years, improving the output of ESNs has become the focus of research in related directions.

Research over the years has been conducted based on two approaches. One way to improve the decoding capability [11, 12] is by replacing the linear output layer with a multilayer perceptron trained using backpropagation. Another approach is to use a random nonlinear projection from input to output, followed by a single-layer perceptron. This was first proposed for application by Rosenblatt [13] in 1958. This random projection idea [14] is also known as the Extreme Learning Machine (ELMS) [15] later. This nonlinear projection layer is added between the input and reservoir as a way to improve the nonlinear separation capability of the echo state network. The $R^2SP$ model [16] proposed by Butcher et al. in 2010 and the $\varphi - ESN$ [17] proposed by Gallicchio and Micheli in 2011 both exploit this idea of stochastic nonlinear projection. In 2012, Boccato et al. [18] proposed replacing the readout layer with a Volterra filter. In recent years, with the rapid development of deep learning. Several teams have utilized structures such as convolution instead of linear readout layers. In 2019, Ma's team proposed using multi-scale convolution and maxi-pooling to improve the decoding ability of the network. In 2021, Ma's team [19] introduced an attention mechanism on top of their work, adding an attention mechanism between the reservoir and the convolution to strengthen the effect.

### 2.2 Optimization of the Internal Connection Topology of the Reservoir

The randomness of traditional ESNs can negatively affect network performance. Researchers have begun to optimize the connections between neurons and have launched a series of studies on ESN topologies. Fette et al.'s improved network structure [20] has little effect on the performance of traditional ESNs, but its idea of improving the structure is worthy of reference. Rodan et al. [21] proposed a simple ring topology that achieved comparable results to traditional ESNs while reducing the complexity of the network.Some studies also utilize the idea of complex networks to reconstruct the structure of the reservoir. Xue et al. [22] proposed a new structure called a decoupled echo state network (DESN) with better prediction performance and robustness than traditional ESN. Song et al. [23–25] introduced small-world networks and scale-free properties into the reservoir structure of echo state networks to form a new kind of reservoir. Cui et al. [26] came out with three new dynamic reservoir topologies based on complex network theory for echo state networks, specifically better network performance than traditional ESNs. The new topology of ESN proposed by Boccato et al. [27] has a higher information processing capability. A new reservoir structure [28] is created by incorporating algorithms such as K-means into the reservoir of echo state networks. It is empirically verified that this reservoir structure can achieve higher prediction accuracy than the traditional echo state network. Li et al. [29] proposed a new topology, IESN, and achieved good results on the prediction task.

The basic presentation of the above-related work reveals that these works are either improvements to the linear readout layer or modifications to the reservoir structure. And our work improves both parts. Regarding the reservoir structure, we propose FT-ESN. In the linear readout layer, we utilize two-time scale convolutions for max-pooling, respectively, and then classify them through fully connected and softmax layers.

## 3 Methodology

### 3.1 Echo State Networks

A conventional ESN typically comprises three fundamental components: an input layer, a hidden layer (reservoir layer), and an output layer. The hidden layer, which acts as a reservoir, consists of many randomly connected neurons with sparse connections. Its primary function is to map input signals from a lower-dimensional input space to a higher-dimensional state space. Additionally, the reservoir layer possesses memory capabilities, enabling it to store the system's dynamic behavior through weights between neurons. Consequently, only the weights of the output layer require training. This decoupling of the reservoir layer generation and ESN training dramatically simplifies the network training process. Figure 1 illustrates the structure of a traditional ESN.

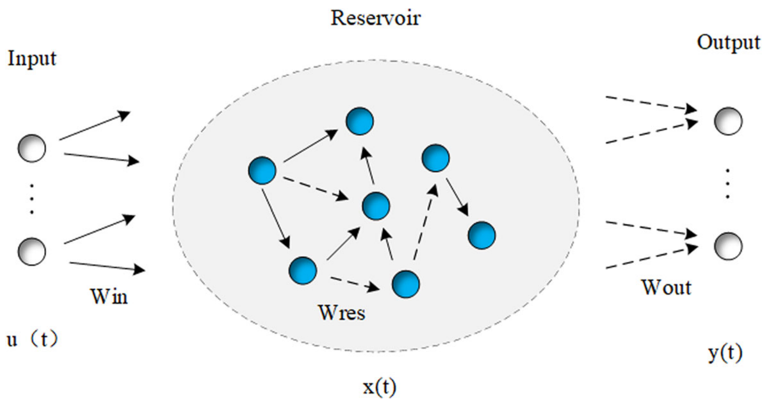


**Fig. 1** The structure of traditional ESN

When the given time step is t, K-input neurons $u(t) = (u_1(t), \ldots, u_K(t))^T$, N-reservoir neurons $x(t) = (x_1(t), \ldots, x_N(t))^T$ and L-output neurons $y(t) = (y_1(t), \ldots, y_L(t))^T$. The input signal first enters the input layer neuron and is transmitted to the hidden layer neuron. The calculation process is generally as follows:

$$x(t+1) = f(W_{res}x(t) + W_{in}u(t+1)) \tag{1}$$

$$y(t+1) = f^{out}(W_{out}x(t+1)) \tag{2}$$

Here, $W_{in}$, $W_{res}$ *and* $W_{out}$ respectively represent the connection weight values input to the reservoir, inside the reservoir, and between the reservoir and the output layer. In the whole network training process, $W_{in}$ *and* $W_{res}$ are predefined and remain unchanged throughout the training process. Only $W_{out}$ needs to be trained. $f$ *and* $f^{out}$ represent the activation functions in the reservoir and output layer, respectively. It is generally tanh().

The description of each weight connection matrix is as follows:

(1) $W_{in}$: In any case, the connection matrix $W_{in}$ of the input layer may be composed of random numbers uniformly distributed from $-a$ to a. The greater the value of a, the greater the nonlinearity of the input data-driven nonlinear unit. That is, the state of neurons in the reservoir is more relevant to the input data. On the contrary, the system is close to zero state. That is, there is no input.

(2) $W_{res}$: In the standard ESN, $W_{res}$ is calculated from a sparse matrix $W_0$. The calculation formula is: $W_{res} = \alpha \cdot \frac{W_0}{|\lambda_{max}|}$,Where, $\lambda_{max}$ is the maximum eigenvalue of $W_0$, and in general, the spectral radius $\alpha < 1$, which can be manually adjusted as required. For ESN, it is essential to determine whether the reservoir has echo state characteristics. According to the research, when the spectral radius $\alpha$ of the internal connection matrix $W_{res}$ of the reservoir is less than 1, it can ensure that the reservoir has the echo state attribute.

(3) $W_{out}$: This weight matrix is the only one that ESN will train. When the matrix is initialized, it can be any value and is usually set to a matrix of all zeros.

## 3.2 The Proposed Model

The whole network structure (named FESCN) is shown in Fig. 2 and is divided into two parts: encoder and decoder. In the encoder, a new topology of ESN is proposed, here called

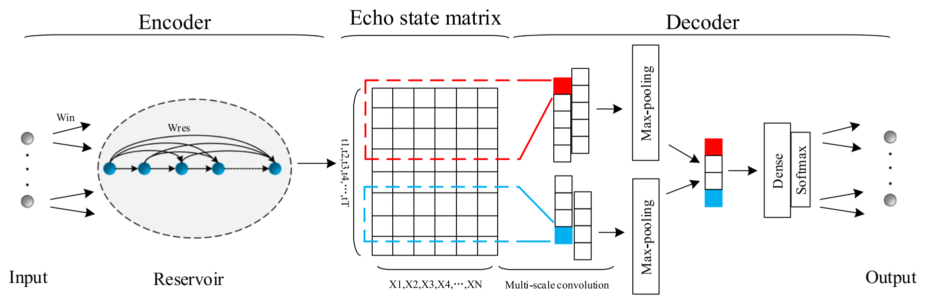


**Fig. 2** Overall network framework diagram

FT-ESN, as shown in Fig. 2. The echo states obtained by FT-ESN for all time steps are collected into a matrix, here called Echo state matrix (ESM). Moving on to the decoder, it involves a series of operations such as multiple time scale convolutions and max-pooling. These operations extract relevant features, which are then fed into a fully connected layer. The final step involves classifying the extracted features using softmax layer computation.

### 3.2.1 Encoder

In the decoder, a new topology of ESN, Forward topology echo state network (FT-ESN), is proposed in this paper, as shown in Fig. 3. It is known from the above introduction about the traditional ESN that the input weight $W_{in}$ and the internal connection weight $W_{res}$ of the storage layer are randomly generated, but both $W_{in}$ and $W_{res}$ are fixed in the FT-ESN. $W_{in}$ and $W_{res}$ are processed using the method proposed by Rodan et al. [21]. All elements of $W_{in}$ and $W_{res}$ are assigned values v and r, respectively, where the input symbols are determined by the irrational decimal expansion $d_1, d_2, d_3, \ldots, d_n$ (here we choose pi). For example, given a threshold of 5 (which is set in this paper), if $0 \leqslant d_n < 5$, then the nth input connection symbol (connecting the input to the nth reservoir unit) will be $-$, and vice versa $+$. The internal connection of the reservoir in Fig. 3 is implemented for a circular nesting operation. Each neuron is to be connected to the following neurons in turn, and the connecting arrows are carried backward.

Here the input is assumed to be a one-dimensional time series $u = (u(0), u(1), \ldots, u(T-1))^T$, each time step is T. The calculation of $x(t)(0 \leqslant t \leqslant T-1)$ is performed by Eq. (1) and the states of all time steps within T are collected into the echo state matrix $X = (x_1, x_2, \ldots, x_N)$, where $x_n = (x_n(0), x_n(1), \ldots, x_n(T-1))^T (1 \leqslant n \leqslant N)$. This part can be understood as mapping the time series to a high-dimensional space, obtaining the enriched states, and finally collecting them in the ESM.

### 3.2.2 Decoder

In the previous coding process, we know that the ESM collects states at time step $0 \sim T-1$. So we can select any number of states under $0 \sim T-1$ time steps to perform the convolution operation, but there are specific parameter settings for each dataset. As shown in Fig. 2, we are performing the convolution at two-time scales. Take the ECG200 dataset as an example, and its sequence length is 96, i.e., T $=$ 96. In the code, we convolve from two-time scales, 57 and 67, so the filter dimensions here are $57 \times 32$ and $67 \times 32$, respectively. Max-pooling is then performed to calculate the maximum value of each feature mapping separately, and then they

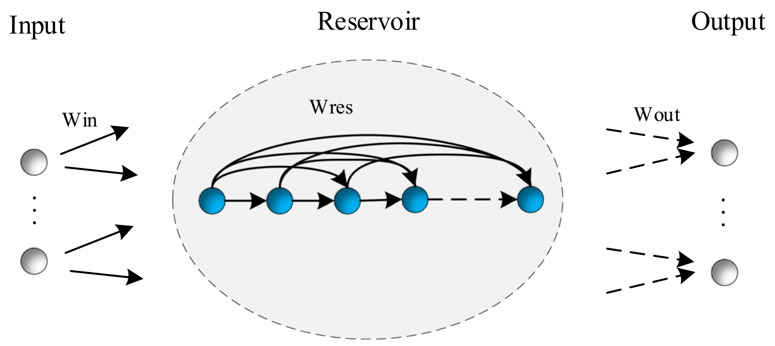


**Fig. 3** The structure of forward topology echo state network

are stitched together. For ECG200, we set the number of filters to 120, so we get 240 outputs after splicing. This gives us feature information for both time scales. The processed data is then sent to the fully connected layer for computation. This ultimately leads to the estimation of conditional probability distributions, which is crucial for accurate categorization. Note that we are selective about the use of the fully connected layer and Dropout. The purpose is to mitigate overfitting and improve network generalization.

## 4 Experiments

We have selected 55 datasets on 85 UCR time series datasets for our experiments. The performance of our proposed network model is evaluated by comparing it with traditional time series classification methods and several mainstream deep learning models, respectively. The experiments are implemented in tensorflow framework, two models are implemented in Python 3.9.12 and tensorflow 2.6.0, and all the experiments are run with CPU AMD Ryzen R7-5800K @ 3.20 GHz, GPU NVIDIA GeForce RTX3050Ti, 16 GB of RAM and equipped with windows 11 operating system were run.

### 4.1 Classification of Univariate Time Datasets

#### 4.1.1 Dataset Introduction

The UCR Time Series Classification Archive [30] contains 85 publicly available time series datasets. These datasets are differentiated according to the number of categories, dataset type, number of samples, and time series length. The datasets are categorized into seven categories, namely, device, ECG, image, motion, sensor, analog, and spectrum. Hence, they prove valuable in evaluating the classifier's overall performance across different scenarios. Table 1 shows the specific parameters of the 55 datasets from which we have selected. Figure 4 visualizes the training sets for the two datasets, Gun-Point and ECGFiveDays. It is clear to see that the curves corresponding to all the time steps under different labels are different, and the job we have to do is to extract the feature information and then distinguish them correctly.

**Table 1** Parameter details for 55 UCR time series classification datasets

| Dataset | Train | Test | Length | Classes | Type |
|---|---|---|---|---|---|
| Adiac | 390 | 391 | 176 | 37 | Image |
| CBF | 30 | 900 | 128 | 3 | Simulated |
| Cricket-X | 390 | 390 | 300 | 12 | Motion |
| Cricket-Y | 390 | 390 | 300 | 12 | Motion |
| Cricket-Z | 390 | 390 | 300 | 12 | Motion |
| Coffee | 28 | 28 | 286 | 2 | Spectro |
| DistPhxAgeGp | 139 | 400 | 80 | 3 | Image |
| DistPhxCorr | 276 | 600 | 80 | 2 | Image |
| DistPhxTW | 139 | 400 | 80 | 6 | Image |
| ECG200 | 100 | 100 | 96 | 2 | ECG |
| ECG5000 | 500 | 4500 | 140 | 5 | ECG |
| ECGFiveDays | 23 | 861 | 136 | 2 | ECG |
| Earthquakes | 139 | 322 | 512 | 2 | Sensor |
| ElectricDevices | 8926 | 7711 | 96 | 7 | Device |
| FacesUCR | 200 | 2050 | 131 | 14 | Image |
| 50words | 450 | 455 | 270 | 50 | Image |
| FISH | 175 | 175 | 463 | 7 | Image |
| FordA | 1320 | 3601 | 500 | 2 | Sensor |
| FordB | 810 | 3636 | 500 | 2 | Sensor |
| Gun-Point | 50 | 150 | 150 | 2 | Motion |
| Ham | 109 | 105 | 431 | 2 | Spectro |
| HandOutlines | 370 | 1000 | 2709 | 2 | Image |
| Haptics | 155 | 308 | 1092 | 5 | Motion |
| InlineSkate | 100 | 550 | 1882 | 7 | Motion |
| InsectWingbeatSound | 220 | 1980 | 256 | 11 | Sensor |
| LargeKitchenAppliances | 375 | 375 | 720 | 3 | Device |
| MedicalImages | 381 | 760 | 99 | 10 | Image |
| MidPhxAgeGp | 154 | 400 | 80 | 3 | Image |
| MidPhxTW | 154 | 399 | 80 | 6 | Image |
| NonInv-Thor1 | 1800 | 1965 | 750 | 42 | ECG |
| NonInv-Thor2 | 1800 | 1965 | 750 | 42 | ECG |
| OSULeaf | 200 | 242 | 427 | 6 | Image |
| PhalCorr | 1800 | 858 | 80 | 2 | Image |
| Phoneme | 214 | 1896 | 1024 | 39 | Sensor |
| Plane | 105 | 105 | 144 | 7 | Sensor |
| ProxPhxAgeGp | 400 | 205 | 80 | 3 | Image |
| ProxPhxCorr | 600 | 291 | 80 | 2 | Image |
| ProxPhxTW | 205 | 400 | 80 | 6 | Image |
| RefrigerationDevices | 375 | 375 | 720 | 3 | Device |
| ScreenType | 375 | 375 | 720 | 3 | Device |
| ShapesAll | 600 | 600 | 512 | 60 | Image |

**Table 1** continued

| Dataset | Train | Test | Length | Classes | Type |
|---|---|---|---|---|---|
| SmallKitchenAppliances | 375 | 375 | 720 | 3 | Device |
| StarLightCurves | 1000 | 8236 | 1024 | 3 | Sensor |
| Strawberry | 370 | 613 | 235 | 2 | Spectro |
| Swedish Leaf | 500 | 625 | 128 | 15 | Image |
| Synthetic Control | 300 | 300 | 60 | 6 | Simulated |
| Trace | 100 | 100 | 275 | 4 | Sensor |
| Two Patterns | 1000 | 4000 | 128 | 4 | Simulated |
| uWaveGest-X | 896 | 3582 | 945 | 8 | Motion |
| uWaveGest-Y | 896 | 3582 | 945 | 8 | Motion |
| uWaveGest-Z | 896 | 3582 | 945 | 8 | Motion |
| UWaveGestAll | 896 | 3582 | 945 | 8 | Motion |
| Wafer | 1000 | 6164 | 152 | 2 | Sensor |
| WordSynonyms | 267 | 638 | 270 | 25 | Image |
| Yoga | 300 | 3000 | 426 | 2 | Image |

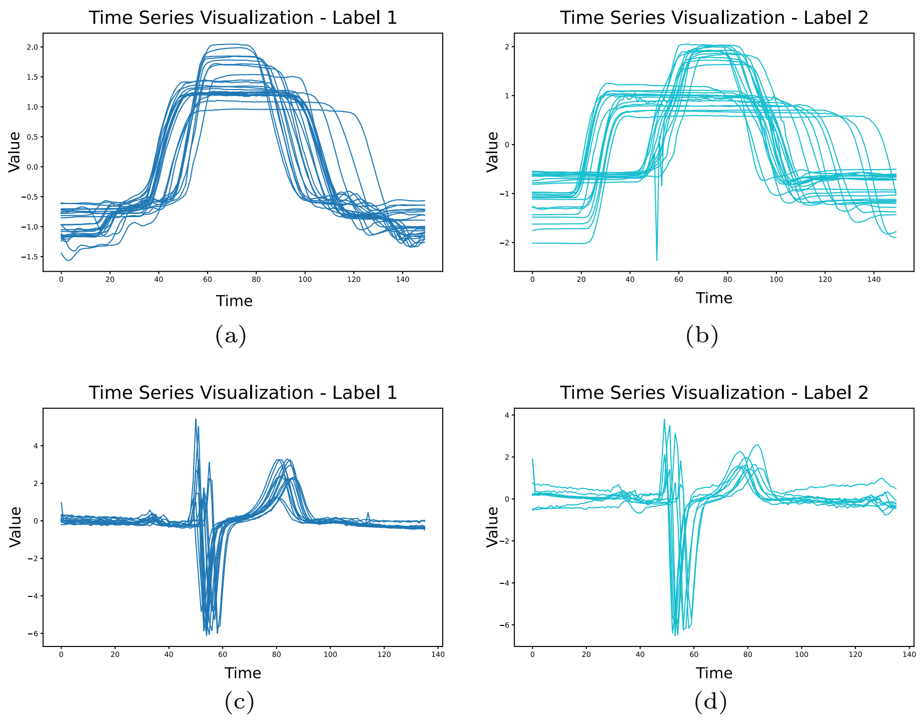


**Fig. 4** Visualization of temporal data in the training set. **a, b** Show the data visualization of the dataset Gunpoint corresponding to different labels, respectively. **c, d** are data visualizations corresponding to different labels for ECGFiveDays, respectively. Here, the horizontal axis is the time step, and the vertical axis is the corresponding value

**Table 2** Critical values for Nemenyi test

| Classifiers (k) | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|
| $q_{0.05}$ | 1.960 | 2.344 | 2.569 | 2.728 | 2.850 | 2.949 | 3.031 | 3.102 | 3.164 |
| $q_{0.1}$ | 1.645 | 2.052 | 2.291 | 2.459 | 2.589 | 2.693 | 2.780 | 2.855 | 2.920 |

### 4.1.2 Evaluation Criteria

There are more datasets in the UCR classification archive, and we cannot achieve the best results on every dataset. So, we use a metric on 55 datasets to evaluate the combined effect of the network. Here we use the mean per class error (MPCE) proposed by Wang et al. [31] to obtain the overall error rate. The following is the specific formula for calculating MPCE:

$$PCE = \frac{1 - Accuracy}{classes} \tag{3}$$

$$MPCE = \frac{1}{k} \sum PCE \tag{4}$$

Here, Eq. (3) corresponds to each dataset; the denominator is the category of the corresponding dataset, and the numerator is the error rate. The k in Eq. (4) is the number of datasets. By normalizing the number of classes on the dataset, an overall error rate per class can be obtained. In addition to MPCE, we also introduce GMR (Geometric Mean Ranking) and AMR (Arithmetic Mean Ranking) to evaluate the performance of network models. By AMR, we then introduce the Nemenyi test [32] to compare the performance of models with each other. Here, we give a parameter called critical difference (CD), which is calculated as follows:

$$CD = q_\alpha \sqrt{\frac{n(n+1)}{6k}} \tag{5}$$

where n is the number of network models participating in the comparison, k is the number of datasets, and the q-value of Eq. (5) is specified in Table 2.

### 4.1.3 Parameter Settings

In the UCR dataset classification experiment, we have several parameters that need to be fixed. The number of reservoir neurons N is set to 32, the leakage rate is 0.3, the input unit scale IS is 0.1, and the assignment elements v and r are 2.1 and 0.1, respectively. We utilize two different time scales for the multi-scale convolution piece. Here, assuming a time series of length T, we can choose a time scale such as $\{mT, nT\}(0 < m, n < 1)$ for the experiment. Here, m and n are generally taken as neighboring values. During the experiment, different datasets take different values for m and n. The same values can achieve good classification for some datasets, but other datasets will show overfitting. Therefore, we need to manually adjust the time scale until each dataset achieves its best results. Each dataset experiment will be adjusted from m = n = 0.1. The number of filters is selected from 30, 60, 90, 120, 150. Here, the fully connected layer size is selected among two values, 64 and 128. For some of the datasets, we use Dropout to improve the generalization ability, and the Dropout rate size is a fixed value of 0.25.

### 4.1.4 Comparison Methods

Both traditional machine learning methods and deep learning models have achieved good results on time series classification. We will compare several traditional machine learning methods with deep learning models.

Traditional machine learning methods can be divided into three primary categories: distance-based methods, feature-based methods, and integration-based methods. Here, we directly used the experimental results collected by Bagnall et al. [33]. We will briefly introduce these methods and select a representative one for comparison. Distance-based methods use various distance metrics to categorize data. Here, we select two classical methods: 1-Nearest Neighbor with Euclidean distance (ED) and 1-Nearest Neighbor with Dynamic Time warp (DTW) [34]. Feature-based methods are methods that extract relevant features through some metric relationship. Here, we select two methods, Learned shapelet (LS) [35] and Time series bag of features (TSF) [36] for comparison. Integration-based methods are combining different classifiers to achieve better classification. Here, we have selected two methods, Elastic ensemble (EE) and collection of transformation ensembles (COTE) [37] for comparison.

For deep learning methods, in 2017, Wang et al. [31] applied three neural network models, namely multilayer perceptron (MLP), residual network (ResNet), and fully convolutional network (FCN), to the UCR dataset and achieved good results. Specific parameter settings are given in the paper. We include their obtained classification results for comparison. In 2019, Ma's team proposed a network framework called EMN [8], which was applied to the UCR dataset and achieved good results. Specific parameter settings are also given in the paper, which we include in our comparison method as well.

### 4.1.5 Results

In the following, we give a comparison of the effectiveness of six traditional machine learning methods and three deep learning models with our proposed network model on the UCR dataset, respectively.

The specific comparison between ED, DTW, LS, TSF, EE, COTE, and FESCN on the UCR dataset is given in Table 3. It can be seen that FESCN has the highest number of best-performing datasets at 26. Here, the COTE method also achieves good results, with the best-performing dataset reaching 24. However, the COTE method utilizes 35 classifiers for weighted voting, and its network size and computation are relatively large. Our proposed FESCN achieves better results on the UCR dataset with a simpler network structure and smaller computation. With the data in the table, we know that FESCN does not perform well on all datasets. So, here we introduce MPCE as an evaluation index to evaluate the comprehensive performance of the network on the dataset more scientifically. As can be seen from the data in Table 2, FESCN has the lowest MPCE value of 0.0343. That is to say, the combined performance of FESCN on 55 datasets is better than the other six traditional methods.

The accuracy of MLP, FCN, ResNet, EMN, and FESCN on the 55 UCR datasets is given in Table 4. We analyze the data in Table 4, summarized in Table 5, using the three evaluation metrics AMR, GMR, and MPCE. Through Table 5, we can see that FESCN achieves the best results with the highest number of datasets of 28. The values of AMR and GMR are also the smallest, with 1.680 and 1.982, respectively. By calculating the values of MPCE for the other four network models, we also find that the MPCE value of FESCN is also the lowest at 0.0343, and only the MPCE value of FESCN at 0.0349 is the closest. Here, we also performed the Nemenyi test on the AMR of the five network models. We choose the value at $\alpha = 0.1$,

**Table 3** Accuracy of FESCN and six traditional machine learning classification methods on 55-time series classification datasets

| Dataset | ED | DTW | LS | TSF | EE | COTE | FESCN |
|---|---|---|---|---|---|---|---|
| Adiac | 0.611 | 0.604 | 0.522 | 0.731 | 0.665 | 0.790 | **0.810** |
| CBF | 0.852 | 0.997 | 0.991 | 0.994 | 0.998 | 0.996 | **1.000** |
| Cricket-X | 0.577 | 0.754 | 0.741 | 0.664 | **0.813** | 0.808 | 0.802 |
| Cricket-Y | 0.567 | 0.744 | 0.718 | 0.672 | 0.805 | **0.825** | 0.792 |
| Cricket-Z | 0.587 | 0.754 | 0.741 | 0.672 | 0.782 | 0.815 | **0.821** |
| Coffee | **1.000** | **1.000** | **1.000** | 0.964 | **1.000** | **1.000** | **1.000** |
| DistPhxAgeGp | 0.626 | 0.770 | 0.719 | 0.748 | 0.691 | 0.748 | **0.863** |
| DistPhxCorr | 0.717 | 0.717 | 0.779 | 0.772 | 0.728 | 0.761 | **0.825** |
| DistPhxTW | 0.633 | 0.590 | 0.626 | 0.669 | 0.647 | 0.698 | **0.805** |
| ECG200 | 0.880 | 0.770 | 0.880 | 0.870 | 0.880 | 0.880 | **0.940** |
| ECG5000 | 0.925 | 0.924 | 0.932 | 0.939 | 0.939 | **0.946** | **0.946** |
| ECGFiveDays | 0.797 | 0.768 | **1.000** | 0.956 | 0.820 | 0.999 | **1.000** |
| Earthquakes | 0.712 | 0.719 | 0.741 | 0.748 | 0.741 | 0.748 | **0.813** |
| ElectricDevices | 0.552 | 0.602 | 0.587 | 0.693 | 0.663 | 0.713 | **0.747** |
| FacesUCR | 0.769 | 0.905 | 0.939 | 0.883 | 0.945 | 0.942 | **0.946** |
| 50words | 0.631 | 0.690 | 0.730 | 0.741 | **0.820** | 0.798 | 0.749 |
| FISH | 0.783 | 0.823 | 0.960 | 0.794 | 0.966 | **0.983** | 0.954 |
| FordA | 0.665 | 0.555 | **0.958** | 0.815 | 0.738 | 0.957 | 0.935 |
| FordB | 0.606 | 0.620 | **0.917** | 0.688 | 0.662 | 0.804 | 0.907 |
| Gun-Point | 0.913 | 0.907 | **1.000** | 0.973 | 0.993 | **1.000** | 0.993 |
| Ham | 0.600 | 0.467 | 0.667 | 0.743 | 0.571 | 0.648 | **0.790** |
| HandOutlines | 0.862 | 0.881 | 0.481 | **0.919** | 0.889 | **0.919** | 0.887 |
| Haptics | 0.370 | 0.377 | 0.468 | 0.445 | 0.393 | **0.523** | 0.510 |
| InlineSkate | 0.342 | 0.384 | 0.438 | 0.376 | 0.460 | **0.495** | 0.465 |
| InsectWingbeatSound | 0.562 | 0.355 | 0.606 | 0.633 | 0.595 | **0.653** | 0.638 |
| LargeKitchenAppliances | 0.493 | 0.795 | 0.701 | 0.571 | 0.811 | 0.845 | **0.904** |
| MedicalImages | 0.684 | 0.737 | 0.664 | 0.755 | 0.742 | 0.758 | **0.792** |
| MidPhxAgeGp | 0.519 | 0.500 | 0.571 | 0.578 | 0.558 | 0.636 | **0.803** |
| MidPhxTW | 0.513 | 0.506 | 0.506 | 0.565 | 0.513 | 0.571 | **0.654** |
| NonInv-Thor1 | 0.829 | 0.790 | 0.259 | 0.876 | 0.846 | 0.931 | **0.936** |
| NonInv-Thor2 | 0.880 | 0.865 | 0.770 | 0.910 | 0.913 | **0.946** | 0.939 |
| OSULeaf | 0.521 | 0.591 | 0.777 | 0.583 | 0.806 | **0.967** | 0.906 |
| PhalCorr | 0.761 | 0.728 | 0.765 | 0.803 | 0.773 | 0.770 | **0.829** |
| Phoneme | 0.109 | 0.228 | 0.218 | 0.212 | 0.305 | **0.349** | 0.248 |
| Plane | 0.962 | **1.000** | **1.000** | **1.000** | **1.000** | **1.000** | **1.000** |
| ProxPhxAgeGp | 0.785 | 0.805 | 0.834 | 0.849 | 0.805 | 0.854 | **0.868** |
| ProxPhxCorr | 0.808 | 0.784 | 0.849 | 0.828 | 0.808 | 0.869 | **0.883** |
| ProxPhxTW | 0.707 | 0.761 | 0.776 | 0.815 | 0.766 | 0.780 | **0.833** |
| RefrigerationDevices | 0.395 | 0.464 | 0.515 | **0.589** | 0.437 | 0.547 | 0.565 |
| ScreenType | 0.360 | 0.397 | 0.429 | 0.456 | 0.445 | **0.547** | 0.533 |
| ShapesAll | 0.752 | 0.768 | 0.768 | 0.792 | 0.867 | **0.892** | 0.870 |

**Table 3** continued

| Dataset | ED | DTW | LS | TSF | EE | COTE | FESCN |
|---|---|---|---|---|---|---|---|
| SmallKitchenAppliances | 0.344 | 0.643 | 0.664 | **0.811** | 0.696 | 0.776 | 0.707 |
| StarLightCurves | 0.849 | 0.907 | 0.947 | 0.969 | 0.926 | **0.980** | 0.970 |
| Strawberry | 0.946 | 0.941 | 0.911 | 0.965 | 0.946 | 0.951 | **0.974** |
| Swedish leaf | 0.789 | 0.792 | 0.907 | 0.914 | 0.915 | **0.955** | 0.942 |
| Synthetic control | 0.880 | 0.993 | 0.997 | 0.987 | 0.990 | **1.000** | 0.997 |
| Trace | 0.760 | **1.000** | **1.000** | 0.990 | 0.990 | **1.000** | **1.000** |
| Two Patterns | 0.907 | **1.000** | 0.993 | 0.991 | **1.000** | **1.000** | 0.999 |
| uWaveGest-X | 0.739 | 0.728 | 0.791 | 0.804 | 0.805 | **0.822** | 0.811 |
| uWaveGest-Y | 0.662 | 0.634 | 0.703 | 0.727 | 0.726 | **0.759** | 0.731 |
| uWaveGest-Z | 0.650 | 0.658 | 0.747 | 0.743 | 0.724 | **0.750** | 0.746 |
| UWaveGestAll | 0.948 | 0.892 | 0.953 | 0.957 | **0.968** | 0.964 | 0.961 |
| Wafer | 0.995 | 0.980 | 0.996 | 0.996 | 0.997 | **1.000** | 0.998 |
| WordSynonyms | 0.618 | 0.649 | 0.607 | 0.647 | **0.779** | 0.757 | 0.674 |
| Yoga | 0.830 | 0.837 | 0.834 | 0.859 | **0.879** | 0.877 | 0.871 |
| Best | 1 | 4 | 7 | 4 | 8 | 24 | 26 |
| MPCE | 0.0677 | 0.0639 | 0.0510 | 0.0489 | 0.0530 | 0.0384 | 0.0343 |

Bold is the maximum value in the corresponding data in each row, which can be used to visualize which method corresponds to the best assessment indicator

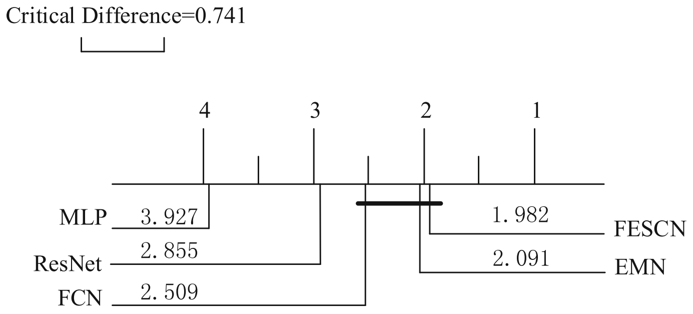


**Fig. 5** Critical difference diagram over the average arithmetic rank of FESCN and five deep learning models

and according to Eq. (5) and the data in Table 2, we can get the CD value of about 0.741. this way we can get a Nemenyi test plot as shown in Fig. 5. Through Fig. 5, we can know that our proposed FESCN is much better than MLP ($3.927 - 1.982 = 1.945 > 0.741$) and ResNet ($2.855 - 1.982 = 0.873 > 0.741$), comparing with FCN ($2.509 - 1.982 = 0.527 < 0.741$) and EMN ($2.091 - 1.982 = 0.109 < 0.741$) achieved better performance.

By comparing the parameter settings of the different methods, we can learn that, except for EMN, the other three network models have relatively complex structures containing multiple convolutional or fully connected layers. In contrast, our proposed FESCN mainly consists of a simple recurrent layer FT-ESN and CNN with higher training efficiency. The fundamental difference compared to EMN is the recurrent layer. By looking at Figs. 1 and 3, we can clearly see the difference in the recurrent layer. We propose the new topology FT-ESN aiming to achieve better classification performance, and the results on 55 UCR datasets

**Table 4** Accuracy of FESCN and four deep learning classification methods

| Dataset | MLP | FCN | ResNet | EMN | FESCN |
|---|---|---|---|---|---|
| Adiac | 0.752 | **0.857** | 0.826 | 0.829 | 0.810 |
| CBF | 0.860 | **1.000** | 0.994 | **1.000** | **1.000** |
| Cricket-X | 0.569 | 0.815 | **0.821** | 0.782 | 0.802 |
| Cricket-Y | 0.595 | 0.792 | **0.805** | 0.787 | 0.792 |
| Cricket-Z | 0.592 | 0.813 | 0.813 | 0.808 | **0.821** |
| Coffee | **1.000** | **1.000** | **1.000** | **1.000** | **1.000** |
| DistPhxAgeGp | 0.827 | 0.835 | 0.798 | 0.843 | **0.863** |
| DistPhxCorr | 0.810 | 0.812 | 0.820 | 0.822 | **0.825** |
| DistPhxTW | 0.747 | 0.790 | 0.740 | 0.795 | **0.805** |
| ECG200 | 0.920 | 0.900 | 0.870 | 0.920 | **0.940** |
| ECG5000 | 0.935 | 0.941 | 0.931 | 0.944 | **0.946** |
| ECGFiveDays | 0.970 | 0.985 | 0.955 | **1.000** | **1.000** |
| Earthquakes | 0.792 | 0.801 | 0.786 | 0.811 | **0.813** |
| ElectricDevices | 0.580 | 0.723 | 0.728 | 0.716 | **0.747** |
| FacesUCR | 0.815 | 0.948 | **0.958** | 0.947 | 0.946 |
| 50words | 0.712 | 0.679 | 0.727 | **0.758** | 0.749 |
| FISH | 0.874 | 0.971 | **0.989** | 0.960 | 0.954 |
| FordA | 0.769 | 0.906 | 0.928 | 0.932 | **0.935** |
| FordB | 0.629 | 0.883 | 0.900 | **0.908** | 0.907 |
| Gun-Point | 0.933 | **1.000** | 0.993 | 0.993 | 0.993 |
| Ham | 0.714 | 0.762 | 0.781 | 0.781 | **0.790** |
| HandOutlines | 0.807 | 0.776 | 0.861 | **0.891** | 0.887 |
| Haptics | 0.461 | **0.551** | 0.506 | 0.519 | 0.510 |
| InlineSkate | 0.351 | 0.411 | 0.365 | 0.460 | **0.465** |
| InsectWingbeatSound | 0.631 | 0.402 | 0.531 | **0.641** | 0.638 |
| LargeKitchenAppliances | 0.480 | 0.896 | 0.893 | 0.901 | **0.904** |
| MedicalImages | 0.729 | **0.792** | 0.772 | 0.775 | **0.792** |
| MidPhxAgeGp | 0.735 | 0.768 | 0.760 | 0.800 | **0.803** |
| MidPhxTW | 0.609 | 0.612 | 0.607 | 0.639 | **0.654** |
| NonInv-Thor1 | 0.942 | **0.961** | 0.948 | 0.933 | 0.936 |
| NonInv-Thor2 | 0.943 | **0.955** | 0.951 | 0.939 | 0.939 |
| OSULeaf | 0.570 | **0.988** | 0.979 | 0.897 | 0.906 |
| PhalCorr | 0.830 | 0.826 | 0.825 | **0.832** | 0.829 |
| Phoneme | 0.098 | **0.345** | 0.324 | 0.239 | 0.248 |
| Plane | 0.981 | **1.000** | **1.000** | **1.000** | **1.000** |
| ProxPhxAgeGp | 0.824 | 0.849 | 0.849 | 0.854 | **0.868** |
| ProxPhxCorr | 0.887 | 0.900 | **0.918** | 0.890 | 0.883 |
| ProxPhxTW | 0.797 | 0.810 | 0.807 | 0.830 | **0.833** |
| RefrigerationDevices | 0.371 | 0.533 | 0.528 | 0.560 | **0.565** |
| ScreenType | 0.408 | 0.667 | **0.707** | 0.555 | 0.533 |
| ShapesAll | 0.775 | 0.898 | **0.912** | 0.873 | 0.870 |

**Table 4** continued

| Dataset | MLP | FCN | ResNet | EMN | FESCN |
|---|---|---|---|---|---|
| SmallKitchenAppliances | 0.389 | **0.803** | 0.797 | 0.699 | 0.707 |
| StarLightCurves | 0.957 | 0.967 | 0.975 | **0.978** | 0.970 |
| Strawberry | 0.967 | 0.969 | 0.958 | 0.971 | **0.974** |
| Swedish Leaf | 0.893 | **0.966** | 0.958 | 0.941 | 0.942 |
| Synthetic Control | 0.950 | 0.990 | **1.000** | 0.997 | 0.997 |
| Trace | 0.820 | **1.000** | **1.000** | **1.000** | **1.000** |
| Two Patterns | 0.853 | **1.000** | **1.000** | 0.999 | 0.999 |
| uWaveGest-X | 0.768 | 0.754 | 0.787 | **0.813** | 0.811 |
| uWaveGest-Y | 0.703 | 0.725 | 0.668 | **0.736** | 0.731 |
| uWaveGest-Z | 0.705 | 0.729 | **0.755** | **0.755** | 0.746 |
| UWaveGestAll | 0.954 | 0.826 | 0.868 | 0.958 | **0.961** |
| Wafer | 0.996 | 0.997 | 0.997 | **0.998** | **0.998** |
| WordSynonyms | 0.594 | 0.580 | 0.632 | 0.663 | **0.674** |
| Yoga | 0.855 | 0.845 | 0.858 | 0.866 | **0.871** |

Bold is the maximum value in the corresponding data in each row, which can be used to visualize which method corresponds to the best assessment indicator

**Table 5** Summary of performance evaluation of five network models

| Dataset | MLP | FCN | ResNet | EMN | FESCN |
|---|---|---|---|---|---|
| BEST | 1 | 15 | 13 | 16 | **28** |
| AMR | 3.927 | 2.509 | 2.855 | 2.091 | **1.982** |
| GMR | 4.236 | 2.345 | 2.449 | 1.985 | **1.680** |
| MPCE | 0.0553 | 0.0365 | 0.0365 | 0.0349 | **0.0343** |

Bold is the maximum value in the corresponding data in each row, which can be used to visualize which method corresponds to the best assessment indicator

have fully verified this. We have plotted six scatter comparison plots, as shown in Fig. 6, in order to better observe directly the performance of FESCN with other methods on 55 UCR datasets. The red points in Fig. 5 correspond to the 55 datasets; the horizontal coordinate is the accuracy of FESCN on the dataset, and the vertical coordinate is the accuracy of the other six models on the dataset. The more points below the diagonal line represent more datasets where FESCN performs well. The farther the red dots are from the diagonal line, the greater the gap between the accuracy of the two networks on that dataset. Through Fig. 6b–f, it is obvious that FESCN has better classification performance compared to the other five models. Looking closely at Fig. 6a, there are a large number of red dots distributed near the diagonal, but it is not difficult to find more red dots distributed below the diagonal; nonetheless, we can intuitively see that FESCN outperforms EMN on the dataset.

We visualize the classification results obtained from part of the dataset through the network with a confusion matrix as in Fig. 7. The labels obtained from the classification are consistent with the true labels, i.e., correctly classified. With the confusion matrix, we can clearly see the specific classification of the corresponding dataset for the full number of samples under the network test. The test accuracy of the corresponding dataset can also be calculated in Fig. 7. Utilizing it can help us to better understand the performance of FESCN.

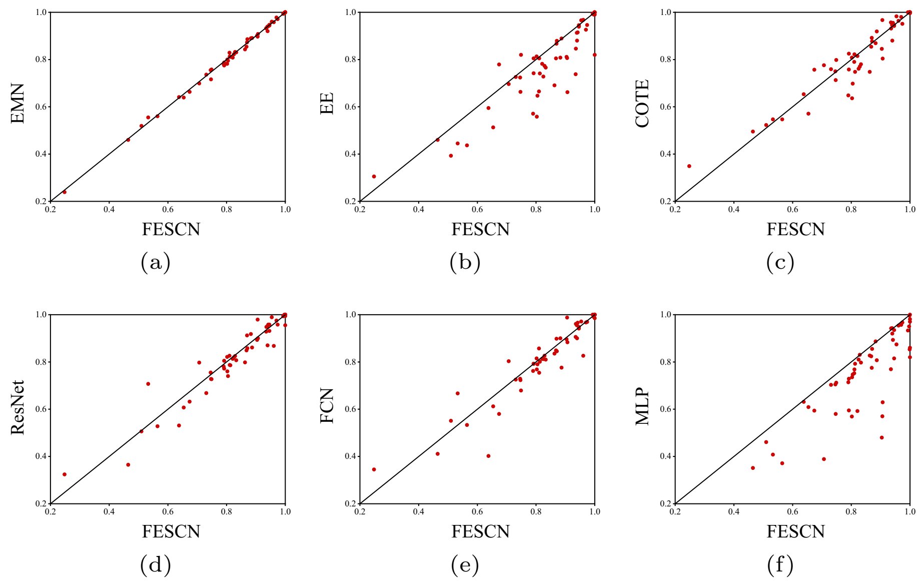


**Fig. 6** Scatter plots of pairwise comparison of six models against FESCN

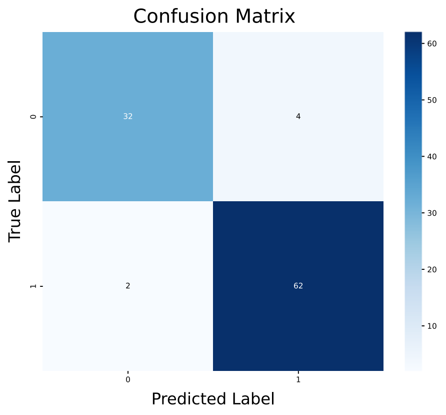
## 4.2 Network Performance Testing with Different Reservoir Sizes

The number of neurons in the reservoir, N, affects the ability of the FT-ESN to process data and, thus, the classification performance of the FESCN. The more the number of neurons, the more complex the dynamic properties that FT-ESN can exhibit, and the size of N directly affects the generalization ability of FT-ESN. To explore the network performance under different reservoir sizes, we performed FT-ESNs on ECG200, DistPhxAgeGp, Synthetic, Earthquakes, LargeKitchenAppliances, Strawberry, Gun-point, ProxPhxAgeGp, MidPhx-AgeGp Experimental tests were conducted on these nine datasets. We set up six reservoir sizes, 8, 16, 32, 64, 128, and 256, conducted experiments on nine datasets, and plotted the obtained data as line graphs, as shown in Fig. 8.

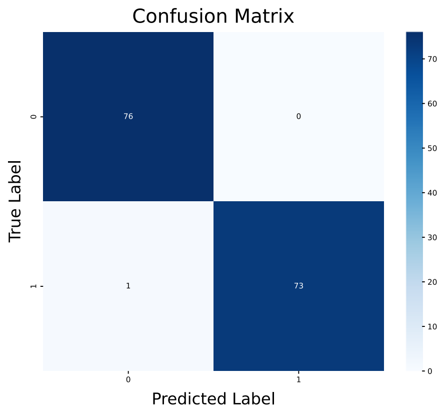
For these nine datasets, we observe that they vary by small and large amounts, but all show an overall trend. As the value of N increases, the accuracy of FESCN improves at first, but when the value of N increases to a certain level, the accuracy decreases. Upon closer inspection, all nine datasets show good results at N = 32. However, the accuracy will gradually decrease if the reservoir size continues to increase. This is because the increase in size leads to a rise in the number of model parameters, which triggers the overfitting phenomenon. Therefore, in the UCR dataset classification experiments, we set the N value to 32.

## 4.3 Network Performance Testing Under Noise Interference

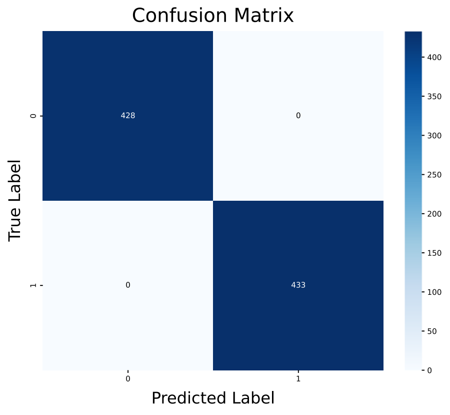
Typical time series data is easily disturbed by environmental background noise, which can lead to some degradation in the algorithm's effectiveness. To test the performance of our proposed FESCN model under noise interference, we add Gaussian white noise to the original



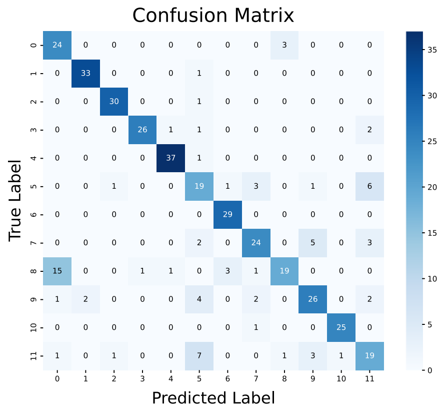
(a) ECG200



(b) Gun-Point



(c) ECGFiveDays



(d) Cricket-X

**Fig. 7** The confusion matrix is used to show the classification effect of the model on the four datasets

dataset for experiments. We obtain the noise signal function using the following equation:

$$P_n = \frac{P_s}{10^{\frac{SNR}{10}}} \tag{6}$$

$$n = \sqrt{P_n} \cdot len(x) \tag{7}$$

where Ps and Pn denote the effective power of the signal and noise, respectively, and SNR represents the signal-to-noise ratio (in dB).

We read the original dataset, add noise to each line of the time series, merge the labeled values and data, and then save to get the new dataset with Gaussian noise. Here, the number of neurons N inside the reservoir remains 32. We set the SNR to 10, 20, 30, 40, and 50, respectively, and utilize the FESCN and EMN models on the nine datasets WordsSynonyms, ShapesAll, Plane, ECG200, Ham, DistPhxAgeGp, CBF, MidPhxAgeGp, and NonInv-Thor1, respectively Experiments were performed. The obtained data were plotted as point-line diagrams, as shown in Fig. 9. We have added a particular scale, 'Inf,' to the horizontal coordinate

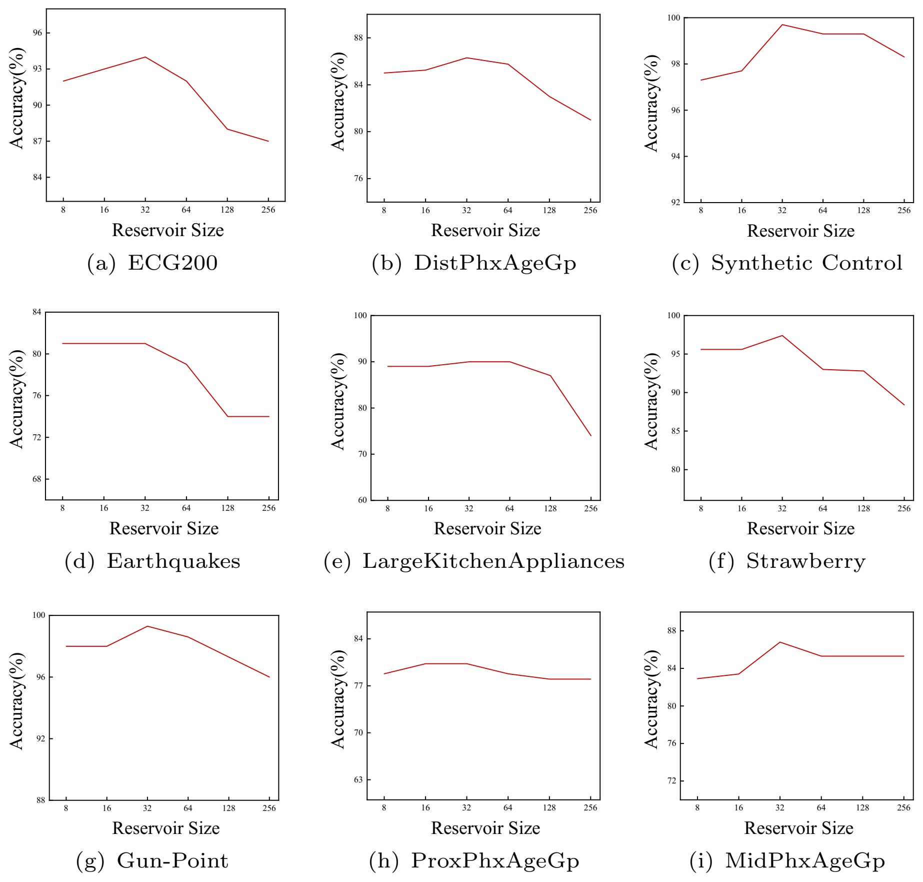


**Fig. 8** Classification accuracy of FESCN on 9 UCR datasets with different number of reservoir units

of the graph, meaning that the SNR is infinite, which is the case without added noise. We said the points without noise to do a better comparison.

With the nine graphs in Fig. 9, it is easy to see that the accuracy of the network decreases gradually after adding noise. When the SNR is large, the network performance decreases very little. But when the SNR is 10 or 20, the accuracy of the network is reduced to some extent. For the dataset with few samples and few categories, the network accuracy decreases by no more than 5% when the SNR is 10. As shown in Fig. 9d, e, g, h, compared with the one without noise processing, the accuracy of the FESCN model decreases by 3, 2.8, 0.3, and 5% in order. It is easy to see that FESCN consistently outperforms EMN, and the decrease in performance is smaller than that of EMN. For the dataset with a large number of categories, as in Fig. 9a–c, f. When the signal-to-noise ratio is 10, the accuracy of the FESCN model decreases by 4.2, 6, 2, and 3.2% in order. It can be seen that the performance degradation of FESCN is smaller than that of EMN. for the dataset with a large number of samples, large number of categories, and considerable sequence length, as in Fig. 9i. It can be intuitively seen that the network accuracy decreases quite a lot for FESCN and EMN when the signal-to-noise ratio is low, and the decrease is more remarkable for EMN.

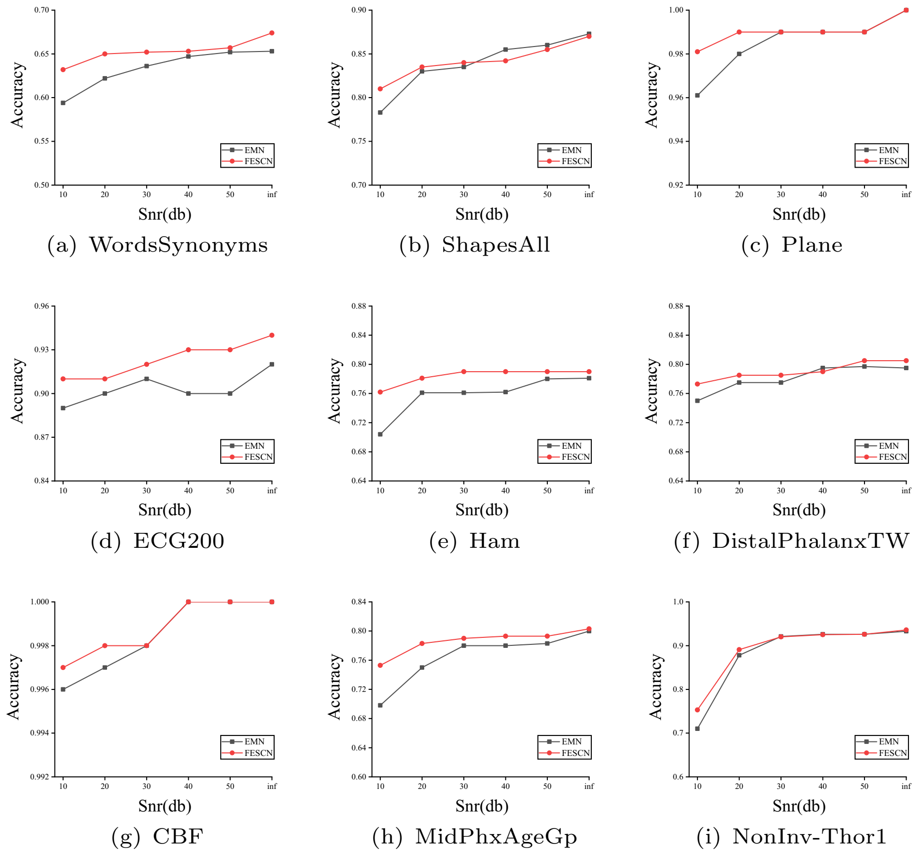


**Fig. 9** Dot-line plots of the performance of FESCN and EMN under different levels of noise interference

## 5 Conclusion

This paper introduces a novel topology called FT-ESN to model time series and obtain rich echo state information. Based on this, we propose a network framework, FESCN, to extract discriminative features by utilizing multi-scale convolution and maximum pooling. After comparing six traditional methods and four neural network models, we find that FESCN performs best on the classification task on 55 UCR datasets, thanks to the temporal modeling capability of its reservoir layers and the feature extraction capability of convolutional neural networks. Subsequently, we investigated the effect of different reservoir sizes on the performance of FESCN and found that good results were achieved on all datasets at N $=$ 32 through data analysis. Finally, we tested the performance of FESCN and EMN when subjected to different levels of noise interference. The experimental results show that FESCN has better noise interference resistance than EMN. However, when dealing with datasets with a more significant number of samples, more categories, and longer sequence lengths, the performance degradation of our network structure is more extensive at low signal-to-noise ratios. Thus, further optimization of the network structure is needed.

In the future, we need to deal with more complex time series data, both univariate and multivariate, often disturbed by noise. Therefore, we need to improve the network structure further to increase its noise immunity and to better cope with the challenges of time series classification tasks.

**Acknowledgements** Project supported by the National Natural Science Foundation of China (Grant Nos. U20A20227, 62076208, 62076207), Chongqing Talent Plan "Contract System" Project (Grant No.CQYC2021 0302257), National Key Laboratory of Smart Vehicle Safety Technology Open Fund Project (Grant No.IVSTSKL-202309) and funded by the Chongqing Technology Innovation and Application Development Special Major Project (Grant No.CSTB2023TIAD-STX0020).

**Author Contributions** X.L. wrote the main manuscript text. All authors reviewed the manuscript.

## Declarations

**Conflict of interest** The authors have no competing interests related to the content of this article.

## References

1. Jaeger H (2001) The "echo state" approach to analysing and training recurrent neural networks-with an erratum note. Bonn, Germany: German National Research Center for Information Technology GMD Technical Report 148(34):13
2. Jaeger H, Maass W, Principe J (2007) Special issue on echo state networks and liquid state machines. 20(3):287–289
3. Jaeger H, Haas H (2004) Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication. Science 304(5667):78–80
4. Skowronski MD, Harris JG (2007) Automatic speech recognition using a predictive echo state network classifier. Neural Netw 20(3):414–423
5. Verstraeten D, Schrauwen B, d'Haene M, Stroobandt D (2007) An experimental unification of reservoir computing methods. Neural Netw 20(3):391–403
6. Tanisaro P, Heidemann G (2016) Time series classification using time warping invariant echo state networks. In: 2016 15th IEEE International conference on machine learning and applications (ICMLA). IEEE, pp 831–836
7. Hoerl AE, Kennard RW (1970) Ridge regression: applications to nonorthogonal problems. Technometrics 12(1):69–82
8. Ma Q, Zhuang W, Shen L, Cottrell GW (2019) Time series classification with echo memory networks. Neural Netw 117:225–239
9. Boccato L, Lopes A, Attux R, Von Zuben FJ (2012) An extended echo state network using Volterra filtering and principal component analysis. Neural Netw 32:292–302
10. Boccato L, Soriano DC, Attux R, Von Zuben FJ (2012) Performance analysis of nonlinear echo state network readouts in signal processing tasks. In: The 2012 international joint conference on neural networks (IJCNN). IEEE, pp 1–8
11. Lukoševicius M (2007) Echo state networks with trained feedbacks. Networks (4)
12. Babinec Š, Pospíchal J (2006) Merging echo state and feedforward neural networks for time series forecasting. In: Artificial neural networks—ICANN 2006: 16th international conference, Athens, Greece, September 10–14, 2006. Proceedings, Part I 16. Springer, Berlin, pp 367–375
13. Rosenblatt F (1958) The perceptron: a probabilistic model for information storage and organization in the brain. Psychol Rev 65(6):386

14. Huang G-B, Zhu Q-Y, Siew C-K (2004) Extreme learning machine: a new learning scheme of feedforward neural networks. In: 2004 IEEE International joint conference on neural networks (IEEE Cat. No. 04CH37541), vol 2. IEEE, pp 985–990
15. Huang G-B, Zhu Q-Y, Siew C-K (2006) Extreme learning machine: theory and applications. Neurocomputing 70(1–3):489–501
16. Butcher J, Verstraeten D, Schrauwen B, Day C, Haycock P (2010) Extending reservoir computing with random static projections: a hybrid between extreme learning and RC. In: 18th European symposium on artificial neural networks (ESANN 2010). D-Side, pp 303–308
17. Gallicchio C, Micheli A (2011) Architectural and Markovian factors of echo state networks. Neural Netw 24(5):440–456
18. Boccato L, Lopes A, Attux R, Von Zuben FJ (2012) An extended echo state network using Volterra filtering and principal component analysis. Neural Netw 32:292–302
19. Ma Q, Zheng Z, Zhuang W, Chen E, Wei J, Wang J (2021) Echo memory-augmented network for time series classification. Neural Netw 133:177–192
20. Fette G, Eggert J (2005) Short term memory and pattern matching with simple echo state networks. In: International conference on artificial neural networks. Springer, Berlin, pp 13–18
21. Rodan A, Tino P (2010) Minimum complexity echo state network. IEEE Trans Neural Netw 22(1):131–144
22. Xue Y, Yang L, Haykin S (2007) Decoupled echo state networks with lateral inhibition. Neural Netw 20(3):365–376
23. Deng Z, Zhang Y (2007) Collective behavior of a small-world recurrent neural system with scale-free distribution. IEEE Trans Neural Netw 18(5):1364–1375
24. Song Q-S, Feng Z-R, Li R-H (2009) Multiple clusters echo state network for chaotic time series prediction. Acta Phys Sinica 58(7):5057–5074
25. Song Q, Feng Z (2010) Effects of connectivity structure of complex echo state network on its prediction performance for nonlinear time series. Neurocomputing 73(10–12):2177–2185
26. Cui H, Liu X, Li L (2012) The architecture of dynamic reservoir in the echo state network. Chaos Interdiscip J Nonlinear Sci 22(3)
27. Boccato L, Attux R, Von Zuben FJ (2014) Self-organization and lateral interaction in echo state network reservoirs. Neurocomputing 138:297–309
28. Najibi E, Rostami H (2015) SCESN, SPESN, SWESN: three recurrent neural echo state networks with clustered reservoirs for prediction of nonlinear and chaotic time series. Appl Intell 43:460–472
29. Li X, Bi F, Yang X, Bi X (2022) An echo state network with improved topology for time series prediction. IEEE Sens J 22(6):5869–5878
30. Yanping C, Keogh E, Hu B, Begum N, Bagnall A, Mueen A, Batista G (2015) The UCR time series classification archive. 2015
31. Wang Z, Yan W, Oates T (2017) Time series classification from scratch with deep neural networks: a strong baseline. In: 2017 International joint conference on neural networks (IJCNN). IEEE, pp 1578–1585
32. Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30
33. Bagnall A, Lines J, Bostrom A, Large J, Keogh E (2017) The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. Data Min Knowl Disc 31:606–660
34. Berndt DJ, Clifford J (1994) Using dynamic time warping to find patterns in time series. In: Proceedings of the 3rd international conference on knowledge discovery and data mining, pp 359–370
35. Yamaguchi A, Nishikawa T (2018) One-class learning time-series shapelets. In: 2018 IEEE International conference on big data (big data). IEEE, pp 2365–2372
36. Deng H, Runger G, Tuv E, Vladimir M (2013) A time series forest for classification and feature extraction. Inf Sci 239:142–153
37. Bagnall A, Lines J, Hills J, Bostrom A (2015) Time-series classification with cote: the collective of transformation-based ensembles. IEEE Trans Knowl Data Eng 27(9):2522–2535