Neurocomputing 520 (2023) 60–72

Contents lists available at ScienceDirect

Neurocomputing

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

Multiscale echo self-attention memory network for multivariate time series classification

Huizi Lyu [a], Desen Huang [b], Sen Li [b], Wing W.Y. Ng [b,c,**], Qianli Ma [b,*]

[a] Aircraft Maintenance Engineering College, Guangzhou Civil Aviation College, Guangdong, China
[b] School of Computer Science and Engineering, South China University of Technology, Guangzhou, China
[c] Guangdong Provincial Key Laboratory of Computational Intelligence and Cyberspace Information, Guangzhou, China

ARTICLE INFO

Article history:
Received 15 August 2021
Revised 19 November 2022
Accepted 21 November 2022
Available online 24 November 2022

Keywords:
Multi-head self-attention
Echo state network
Long-term dependencies
Multivariate time series classification

ABSTRACT

Recently, ESN has been applied to time series classification own to its high-dimensional random projection ability and training efficiency characteristic. The major drawback of applying ESN to time series classification is that ESN cannot capture long-term dependency information well. Therefore, the Multiscale Echo Self-Attention Memory Network (MESAMN) is proposed to address this issue. Specifically, the MESAMN consists of a memory encoder and a memory learner. In the memory encoder, multiple differently initialized ESNs are utilized for high-dimensional projection which is then followed by a self-attention mechanism to capture the long-term dependent features. A multiscale convolutional neural network is developed as the memory learner to learn local features using features extracted by the memory encoder. Experimental results show that the proposed MESAMN yields better performance on 18 multivariate time series classification tasks as well as three 3D skeleton-based action recognition tasks compared to existing models. Furthermore, the capacity for capturing long-term dependencies of the MESAMN is verified empirically.

1. Introduction

Neural networks are playing an increasingly important role in our real life [1,2]. The Echo State Network (ESN) [3] is a type of recurrent neural networks and achieves good performance in many areas, including time series forecasting [1,4,5], system control [6–8], nonlinear system model identification [9]. A typical ESN mainly consists of three layers: the input layer, the hidden layer (also known as the reservoir), and the output layer. The key advantage of ESN is its training efficiency because only weights of reservoir-to-output (a.k.a. readout weights) need to be learned while both weights of input-to-reservoir and reservoir-to-reservoir are randomly set and fixed during the training stage. Due to the large sparsely connected structure of the reservoir, the ESN generates abundant echo states for the input signal which enables the ESN to capture the historical information with short-term memory. However, the long-term memory is essentially important in sequence modeling [10] and time series modeling [11]. How to capture long-term dependencies information in ESN remains a challenging problem in complex time-series systems.

Along with the significant improvements of ESN in time series prediction, researchers also apply the ESN to time series classification tasks [1,12,13]. They can be roughly categorized into three types. The first type constructs several ESN classifiers with the indicator of accumulating mean square error (MSE). M. D. Skowronski et al. [9] propose a competitive network of readout filters that dramatically exploit the MSE over observations as the indicator by using the Viterbi algorithm. The second type models the echo states of ESN for time series classification. L. Wang et al. [14] and Q. Ma et al. [15] propose the C_{ADE} and the ConvESN for time series classification, respectively, which use echo states as the discriminative features and achieve superior performance. The last type utilizes the readout weight of ESN for time series classification. H. Chen et al. [16] propose the model-metric co-learning method, which uses the readout weights in ESN as model representations. Z. Gong et al. [12] propose the MOMM algorithm, which combines the MSE of the prediction in ESN and the separation capacity of readout weights for multiobjective time series classification. Although these ESN-based classifiers have achieved certain improvements, they are not competent in complex time series modeling tasks because the traditional ESN is difficult to capture

* Corresponding author.
** Co-Corresponding author.
E-mail addresses: wingng@ieee.org (W.W.Y. Ng), qianlima@scut.edu.cn (Q. Ma).

https://doi.org/10.1016/j.neucom.2022.11.066
0925-2312/

long-term dependencies information [15]. Without capturing the long-term dependencies existing in the time series, applying the ESN to time series classification tasks still has drawbacks.

In recent years, many methods have been proposed and made significant progress in the field of deep learning. Among these methods, self-attention has become a popular choice for modeling sequential data due to the excellent capacity of capturing the long-term dependencies [17]. Recently, A. Vaswani et al. [17] propose a multi-head attention mechanism called Transformer to capture the global dependence of features. It achieves state-of-the-art results on two translation tasks without using the convolutional neural networks and recurrent neural networks.

Inspired by the self-attention mechanism, we propose a Multiscale Echo Self-Attention Memory Network (MESAMN), which combines the high training efficiency of ESN and the powerful multi-head self-attention mechanism for modeling long-term dependencies in the time series. Specifically, the MESAMN consists of two parts. The first part consists of multiple ESNs to project the time series onto multiple high-dimensional reservoir spaces and generate corresponding Echo State Representations (ESRs). The ESRs of each ESN contains the whole historical echo states in the ESN (i.e., the original representation matrix in Fig. 8). A multi-head self-attention mechanism is then applied to these ESRs. Echo states are enhanced according to weights of importance which is computed from the original ESRs on time scales. The Enhanced ESRs (E^2SRs, i.e., self-attention feature maps in Fig. 8) enable the MESAMN to focus on the long-term dependencies effectively. The second part of the architecture is a shallow convolutional neural network with one convolutional layer. By performing simple feature extraction on multiple high-dimensional E^2SRs, features from multiple reservoirs are effectively integrated and trained for classification.

In short, MESAMN is a novel effective classification framework, which takes the complementary benefits of ESNs, multi-head self-attention mechanism, and multiscale convolutional layer. Compared with the existing methods, the proposed MESAMN can be more excellent in capturing time series features, especially for time series with long-term dependencies. Extensive experiments on 18 multivariate time series (MTS) benchmark datasets and three skeleton-based action recognition datasets demonstrate that the proposed method captures long-term dependencies in the time series and outperforms existing methods.

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

1) We propose a novel neural network architecture named Multiscale Echo Self-Attention Memory Network (MESAMN), which employs the multi-head self-attention mechanism to enable ESN to capture the long-term dependencies of temporal data. Owning to the merit of the training-free property in ESN, the proposed MESAMN is a training-efficient neural network.
2) Based on the memory-Enhanced Echo State Representations (E^2SRs), we leverage a multiscale convolutional layer to learn the multiscale temporal dependencies. The MESAMN not only models long-term dependencies but also extracts multiscale features of the time series.
3) Experiments conducted on 21 datasets show that the MESAMN achieves superior results ranging from MTS benchmark datasets to 3D skeleton-based action recognition datasets. Furthermore, we empirically verify the ability of the proposed MESAMN in modeling long-term dependencies on the Copying task and the Pixel-by-Pixel MNIST task.

The rest of this paper is organized as follows. In Section II, we describe the related work of ESNs on enhancing the long-term modeling capabilities and for classification. In Section III, we introduce the ESN and the multi-head self-attention mechanism. Then the proposed method is described in Section IV. In Section V, we evaluate the performance of the proposed model through experiments. Finally, we make conclusions in Section VI.

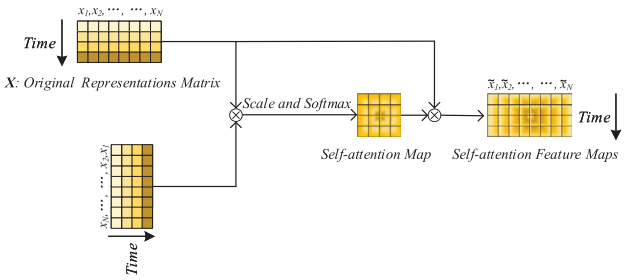

Fig. 8. Schematic diagram of the self-attention mechanism in our proposed model. Note that in the case of modeling ESN, all of the historical echo states are collected together in the Original Representation Matrix.

2. Related work

In this section, related work about the improvement of ESN for modeling time series and some ESN-based time series classification methods are briefly introduced.

2.1. ESNs for modeling time series

The first representative work is the Leaky-ESN [1] proposed by H. Jaeger et al. in 2007. In this model, leaky integration units are introduced in the evolution of the reservoir, which enables the improved ESN to model time series systems flexibly with different time scales. Based on this work, a variable-memory-length ESN (VML-ESN) [18] is proposed to adjust the state update mode according to the autocorrelation property of the input signal adaptively. In [19], arbitrary infinite impulse response (IIR) neurons are introduced to replace traditional one in ESN, which enables the improved ESN to learn more abundant signals in different timescales, thus improving the memory capacity of ESN. Moreover, H. Cui et al. [20] proposed the hybrid circle reservoir (HCR) ESN architecture which injects wavelet-neurons in the reservoir and realizes to increase the variety of the neurons in the reservoir without extra topology complexity. Some work [21,22] employ different neural plasticity learning rules to the weights updated in the reservoir of ESN and help model time series. Besides, Wen et al. propose a memristor-based echo state network (MESN) with an online least mean square (LMS) algorithm [2]. Chouikhi et al. [23] combine ESN and autoencoders to improve the model's ability to learn the data representations.

Although the methods mentioned above have made improvements for modeling time series, it is still challenging for ESN to model the long-term dependencies in complex time series systems explicitly.

2.2. ESN-based classification methods

In the past decade years, many researchers [9,16,14,21,24] exploited ESN for time series classification. In [9], M. D. Skowronski et al. combined a state machine framework with the traditional ESN (which is called predictive ESN) for speech recognition. A model-metric co-learning (MMCL) method [16] is proposed for time series classification by using a cycle reservoir in ESN. It encourages samples in the same classes to construct similar representations, while samples in different classes are enforced to construct different representations separated from each other. In [14],

the C_{ADE} is proposed for MTS classification, which combines an adaptive differential evolution (ADE) algorithm [25] with the conceptors [26]. Q. Ma et al. [24] proposed a functional ESN (FESN) for time series classification tasks. Specifically, the FESN introduced a temporal aggregation operator, which collects the time-varying output weights in ESN and projects the time series into discrete labels. Moreover, they proposed the ConvMESN [13], which adds skip recurrent connections in ESN to capture multitimescale structures in time series and has a promising result on MTS classification. Besides, Z. K. Malik et al. [27] proposed a multi-layered echo state machine (ML-ESM) for time series forecasting and time series classification. However, the ML-ESM still treated the time series classification tasks as predictive tasks, which might cause the loss of global information for modeling long time series. Furthermore, D. Yang et al. propose a novel method for approximating the sequential dynamics and learning the relationship among multiple variables explicitly in a unified framework [28]. Specifically, each MTS is modeled by one model, and the distance of original MTS can be evaluated by the distance of the relevant model. Most recently, Z. Gong et al. propose the MOMM algorithm for multiobjective learning in the model space of ESN and have promising time series classification results [12]. Among the models mentioned above, none of them could explicitly exploit the long-term dependencies in the time series. However, long-term dependencies information is critically important for improving the performance of sequence learning [10] and time series modeling [11]. To this end, we proposed the Multiscale Echo Self-Attention Memory Network (MESAMN) to address this problem.

3. Multiscale echo self-attention memory network

In this section, we formulate the two main components of the proposed Multiscale Echo Self-Attention Memory Network (MESAMN): the Multi-head Self-attention Memory Encoder (MSME) and the Convolutional Memory Learner (CML). The schematic diagram of the MESAMN is shown in Fig. 1.

The MSME consists of multiple ESNs initialized independently to capture temporal dynamics of the input time series. After that, a self-attention mechanism is applied to the Echo State Representations (ESRs) to obtain the Enhanced Echo State Representations (E^2SRs), which contains the long-term dependencies information in the time series. The E^2SRs are then fed to the CML to extract the multiscale discriminative features for classification.

3.1. Multi-head self-attention memory encoder (MSME)

First, we use multiple independently initialized ESNs to generate diversified ESRs. Then the self-attention mechanism is used to construct the E^2SRs which is helpful to model the long-term dependencies of the input time series. Note that each independently initialized ESN is regarded as one head in the multi-head mechanism. We give the formal descriptions as follows.

1) Multi-ESN for Multiple High-Dimension Projection: Given a D-dimension and T-length input time series

$$\mathbf{U} = (\mathbf{u}(1), \dots \mathbf{u}(t), \dots, \mathbf{u}(T))^{\mathrm{T}}, \tag{1}$$

where $\mathbf{u}(t) \in \mathbb{R}^{D\times 1}$ and T denotes the length of the input time series. The multiple high-dimensional projection is formulated as

$$\mathbf{x}_i(t) = f\left(\mathbf{W}_i^{in}\mathbf{u}(t) + \mathbf{W}_i^{res}\mathbf{x}_i(t-1)\right)), \tag{2}$$

where $\mathbf{x}_i \in \mathbb{R}^{N\times 1}, \mathbf{W}_i^{in} \in \mathbb{R}^{N\times D}, \mathbf{W}_i^{res} \in \mathbb{R}^{N\times N}$, and i indicates the i^{th} ESN respectively. Such that, $\mathbf{x}_i(t)$ represents ESRs generated from the i^{th} reservoir at timestamp t.

The ESRs of the input time series $\mathbf{U}$ is constructed as a matrix by collecting the ESRs from the i^{th} reservoir at all timestamps sequentially. Formally, the ESRs can be formulated as follows:

$$\begin{aligned}\mathbf{X}_i &= \Psi_i(\mathbf{U}) = \Psi_i\left((\mathbf{u}(1), \dots, \mathbf{u}(t), \dots, \mathbf{u}(T))^{\mathrm{T}}\right) \\ &= (\mathbf{x}_i(1), \dots, \mathbf{x}_i(t), \dots, \mathbf{x}_i(T))^{\mathrm{T}} \\ &= \begin{pmatrix} x_i^1(1) & \cdots & x_i^n(1) & \cdots & x_i^N(1) \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_i^1(t) & \cdots & x_i^n(t) & \cdots & x_i^N(t) \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_i^1(T) & \cdots & x_i^n(T) & \cdots & x_i^N(T) \end{pmatrix},\end{aligned} \tag{3}$$

where $\mathbf{X}_i \in \mathbb{R}^{T\times N}$ and Ψ_i denote the i^{th} ESN reservoir operators (i.e., Eq. (15)) at all timestamps. To construct multiple ESNs with capability of capturing multiscale features, sparsity of multiple ESNs is uniform sampling between [0.1,0.9] according to the number of ESNs.

2) Self-Attention: After ESRs being generated by multiple reservoirs, the self-attention mechanism is applied to model the long-term dependencies existing in the multiple ESRs to yield the E^2SRs. The self-attention mechanism applied to each ESN is formulated as follows:

$$\mathbf{W}_i = \mathrm{Softmax}\left(\frac{\mathbf{X}_i\mathbf{X}_i^{\mathrm{T}}}{\gamma}\right), \tag{4}$$

$$\widetilde{\mathbf{X}}_i = \mathbf{W}_i\mathbf{X}_i, \tag{5}$$

where $\mathbf{X}_i$ represents the ESRs in the i^{th} ESN and γ is chosen to be the relevant L_2 norm of the ESRs at different timestamp (i.e., Eq. (19) and Eq. (20)). $\widetilde{\mathbf{X}}_i$ corresponds to the E^2SRs, which is capable of representing long-term dependencies existing in the time series.

3.2. Convolutional memory learner (CML)

The E^2SRs has high dimensionality which is difficult to apply for downstream tasks directly. Here, we propose a Convolutional Memory Learner (CML) to apply the E^2SRs for classification tasks. As shown in Fig. 2, the CML consists of four components: a multiscale convolution layer, a max-over-time pooling layer, a fully connected layer, and a softmax layer. Let $\widetilde{\mathbf{X}} = \left\{\widetilde{\mathbf{X}}_i, i = 1, 2, \cdots, Q\right\}$ denotes the E^2SRs collected from Q ESNs in the MESAMN. For the i^{th} ESN, a l-length E^2SRs from timestamp t to $t+l-1$ (i.e., the red box surrounding E^2SRs in Fig. 2) is formulated as follows:

$$\mathbf{z}_i^{t:t+l-1} = \tilde{\mathbf{x}}_i(t) \oplus \tilde{\mathbf{x}}_i(t+1) \oplus \cdots \oplus \tilde{\mathbf{x}}_i(t+l-1), \tag{6}$$

where i indicates the i^{th} ESN, $\oplus$ denotes the concatenation operator, and $\mathbf{z}_i^{t:t+l-1} \in \mathbb{R}^{l\times N}$. Therefore, given an E^2SRs $\widetilde{\mathbf{X}}_i \in \mathbb{R}^{T\times N}$ with the convolution stride set to be 1, the candidate E^2SRs is formulated as follows:

$$\mathbf{Z}_i = \left\{\mathbf{z}_i^{t:t+l-1}\right\}_{t=1}^{T-l+1}. \tag{7}$$

Here, we define the convolutional filters of the i^{th} E^2SRs as $\mathbf{w}_i^{l,k} \in \mathbb{R}^{l\times N}$ where $l \in \{l_j\}_{j=1}^{J}$ and $k \in \{1, 2, \cdots, K\}$ denote the height and the index of the convolutional filter, respectively. Then the convolutional result of $\mathbf{Z}_i$is computed as follows:

$$c_i^{l,k}(t) = f\left(\mathbf{w}_i^{l,k} * \mathbf{z}_i^{t:t+l-1} + b_i^{l,k}\right), \tag{8}$$

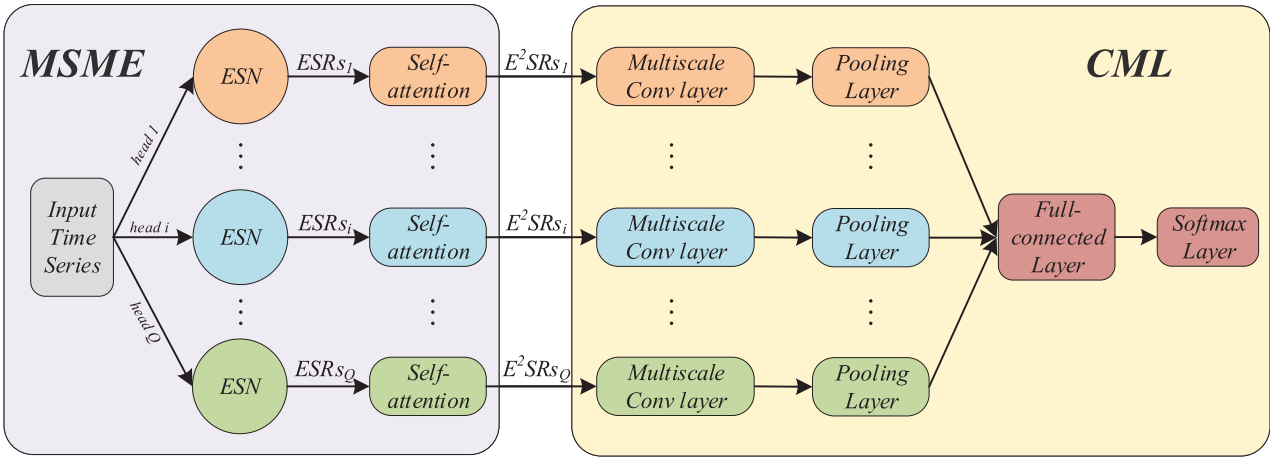

Fig. 1. The schematic architecture of the proposed MESAMN. The MESAMN mainly contains two parts: the Multi-head Self-attention Memory Encoder (MSME) and the Convolutional Memory Learner (CML). Specifically, the MSME consists of high-dimension projection from multiple traditional ESNs followed by a self-attention mechanism, while the CML utilizes a simple convolutional neural network for feature learning and classification.

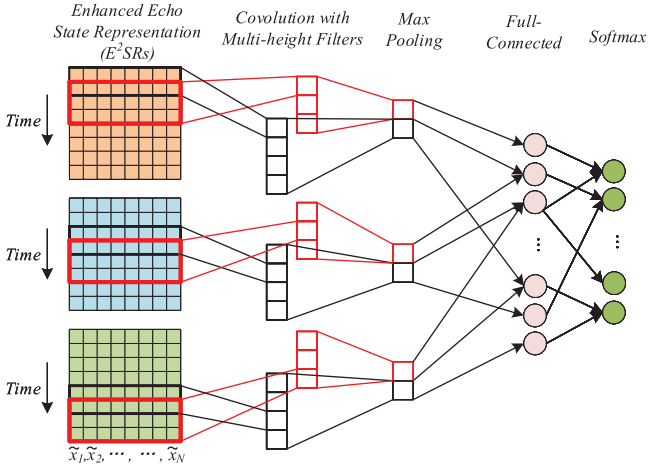

Fig. 2. The architecture of the convolutional neural network used in the CML. In this architecture, a simple convolutional neural network with multiscale filters (time windows in red and black, which can be set differently among different E^2SRs) is used for feature learning and classification.

where $t = 1, 2, \cdots, T - l + 1$, and $*, f$, and $b_i^{l,k}$ denote the convolution operator, the ReLU activation function, and the corresponding bias, respectively. Note that convolutional filters with different heights can capture multiscale features along time. The max-over-time pooling operations are then applied in the max pooling layer after the multiscale convolutional layer, which can be computed as follow:

$$r_i^{l,k} = max_{t\in[1,T-l+1]} c_i^{l,k}(t), \tag{9}$$

where $r_i^{l,k}$ denotes the max value of the activations of the i^{th} E^2SRs with the k^{th} l-height convolutional filters. Therefore, the pooling result after multiscale convolutional layer is as follows:

$$\mathbf{r}_i = \left\{ r_i^{l_1,1}, \cdots, r_i^{l_j,1}, \cdots, r_i^{l_j,k}, \cdots, r_i^{l_j,K}, \cdots, r_i^{l_J,K} \right\}^{\mathrm{T}}, \tag{10}$$

where $\mathbf{r}_i \in \mathbb{R}^{JK}$ denotes the learned features of i^{th} E^2SRs after the convolutional layer and pooling layer. Then, the learned feature of the E^2SRs is as follows:

$$\mathbf{r} = \mathbf{r}_1 \oplus \cdots \oplus \mathbf{r}_i \oplus \cdots \oplus \mathbf{r}_Q. \tag{11}$$

Algorithm 1: The Proposed MESAMN.

Input: input time series $\mathbf{U}$

(1) Multi-head Self-attention Memory Encoder
  for $i \in [1, 2, \ldots, Q]$ **do**
    $\mathbf{X}_i = \Psi_i(\mathbf{U}) = \Psi_i\left((\mathbf{u}(1), \ldots, \mathbf{u}(t), \ldots, \mathbf{u}(T))^{\mathrm{T}}\right)$
    $\mathbf{W}_i = Softmax\left(\frac{\mathbf{X}_i\mathbf{X}_i^{\mathrm{T}}}{\gamma}\right)$
    $\widetilde{\mathbf{X}}_i = \mathbf{W}_i\mathbf{X}_i$
  end for
(2) Convolutional Memory Learner
  repeat
    for $i \in [1, 2, \ldots, Q]$ **do**
      for l in $\{l_j\}_{j=1}^{J}$ **do**
        for k in $[1, 2, \ldots, K]$ **do**
          $\mathbf{z}_i^{t:t+l-1} = \tilde{\mathbf{x}}_i(t) \oplus \tilde{\mathbf{x}}_i(t+1) \oplus \cdots \oplus \tilde{\mathbf{x}}_i(t+l-1)$
          $c_i^{l,k}(t) = f\left(\mathbf{w}_i^{l,k} * \mathbf{z}_i^{t:t+l-1} + b_i^{l,k}\right)$
          $r_i^{l,k} = max_{t\in[1,T-l+1]} c_i^{l,k}(t)$
        end for
      end for
      $\mathbf{r}_i = \left\{ r_i^{l_1,1}, \cdots, r_i^{l_j,1}, \cdots, r_i^{l_j,k}, \cdots, r_i^{l_j,K}, \cdots, r_i^{l_J,K} \right\}^{\mathrm{T}}$
    end for
    Set $\mathbf{r} = \mathbf{r}_1 \oplus \cdots \oplus \mathbf{r}_i \oplus \cdots \oplus \mathbf{r}_Q$.
    $\mathbf{F} = f\left(\mathbf{W}^{fus}\mathbf{r} + \mathbf{b}^{fus}\right)$
    $p(C_s|\mathbf{U}) = p(C_s|\mathbf{F})$
    Minimize the objective in Eq. (14) by the Adam optimizer;
  until convergence

In the fully connected layer, all learned features from different ESNs are fused together as follows:

$$\mathbf{F} = f\left(\mathbf{W}^{fus}\mathbf{r} + \mathbf{b}^{fus}\right), \tag{12}$$

where $\mathbf{W}^{fus}$ and $\mathbf{b}^{fus}$ denote the connection weights and the corresponding bias, respectively. Additionally, the softmax layer used for classification is expressed as follows:

$$p(C_s|\mathbf{U}) = p(C_s|\mathbf{F}) = \frac{exp(\mathbf{W}_s^{cls}\mathbf{F})}{\sum_{m=1}^{N^c} exp(\mathbf{W}_m^{cls}\mathbf{F})}, \quad (13)$$

where C_s, N^c and $\left\{\mathbf{W}_m^{cls}\right\}_{m=1}^{N^c}$ denote the s^{th} class of the input time series, the number of the classes and the corresponding connection weights of the softmax layer, respectively.

3.3. Training of MESAMN

The training procedure of the MESAMN is very efficient because both weights in reservoirs and weights between the input layer and reservoirs are randomly initialized and fixed during the training stage. Only the CML needs to be trained which is a shallow convolutional neural network. The loss function of the MESAMN is the negative logarithm likelihood functions as follows:

$$L(\mathbf{U}) = -\sum_{n=1}^{N^s}\sum_{s=1}^{N^c}\delta(s-v)\ln p(C_s|\mathbf{U}_n), \quad (14)$$

where $N^s, N^c, \delta(\cdot), v, \mathbf{U}_n$, and C_s denote the number of the training samples, the number of the classes, the Kronecker delta function, the ground truth label of the sample, the n^{th} training example, and the s^{th} class, respectively. The procedure of the proposed MESAMN is shown in Algorithm 1.

4. Experimental studies

In this section, a wide range of experiments are conducted to evaluate the proposed model ranging from MTS classification tasks to 3D skeleton-based action recognition tasks. We first briefly describe datasets being used and then introduce data processing procedures and hyper-parameters setting, respectively. Finally, the MESAMN is compared with existing state-of-the-art methods on these challenging datasets. All experiments are conducted on a Linux server with an Intel(R) Core(TM) i7-6850 K CPU @ 3.60 GHz CPU and a GeForce GTX 1080-Ti 11G graphics card.

4.1. Datasets

In this study, 18 MTS benchmark datasets and three 3D skeleton-based action recognition datasets are used to evaluate the performance of the MESAMN. The detail of these datasets is shown in Table 1. **Multivariate Time Series (MTS) Classification Tasks:** 18 benchmark datasets come from various fields, including the UCI repository [29], UCR repository [30], CMU [31], Olszewski [32], Blankertz [33], and Leeb [34] are selected. These datasets have diversified numbers of classes, variables, and instances. In order to fairly compare with well-known methods in [13,14,35–39], fivefold cross-validation on the "Robot Failure" dataset and ten-fold cross-validation on other datasets are performed in experiments. **3D Skeleton-Based Action Recognition Tasks:** The "UTD-MHAD" [40], the "HDM05" [41], and the "Florence3D" [42]. 3D skeleton-based action recognition tasks are very challenging because semantically motions are likely to be numerically dissimilar and different action classes might have similar samples, which is quite confusing. Here, we briefly introduce the three chosen 3D skeleton-based action recognition datasets as follows:

1) The UTD-MHAD dataset is a multi-modal dataset consisting of 864 sequences of three-dimensional coordinates from twenty joints for 27 classes of actions performed by four females and four males. These actions range from hand gestures (such as "draw circle"), sports actions (such as "bowling"), training exercises (such as "arm curl") to daily activities (such as "stand to sit"). Each action is performed four times by each person. These actions are captured by a wearable inertial sensor and a Microsoft Kinect camera while some samples might be damaged. The dataset consists of 861 instances after three corrupted instances are removed.
2) The HDM05 dataset consists of 2339 sequences of three-dimensional coordinates from 31 joints which come from actions performed by five amateurs. There are 130 actions in the dataset which are captured by an optical marker. In our experiments, samples of similar actions are grouped into the same classes (such as jogging starting from the air and the floor) following [43]. After this preprocessing, the dataset has 65 action classes.
3) The Florence3D dataset consists of 215 sequences of three-dimensional coordinates from 15 joints for actions performed by ten amateurs. There are 9 actions performed two or three times by a person. This dataset is quite challenging for high intra-class variation and similar samples in different classes (such as "drink from a bottle" and "answer a phone").

In order to make a fair comparison with other methods, we follow the evaluation protocols as below. For the "UTD-MHAD" dataset and "Florence 3D" dataset, we follow [40,13] use half of the subject as training set while half of the subjects as the testing set. For HDM05, we use 90% - 10% splits for training and testing.

4.2. Data processing

The following preprocessing is performed to make the MESAMN suitable for experiments of all the datasets. For samples with different sequence lengths in the dataset, zeros are padded to make all sequences having the same length as the longest one in the dataset. After this prepocessing, we can generate the E^2SRs with the same dimensionality for each time series, thus can be fed into the CML structure. It should be noted that this kind of zero-padding operation does not bring redundant information because the max-over-time pooling operation is applied to the features along the time scale. In the MSME, we only use the original real information in the multi-head self-attention mechanism. This means that the ESRs are updated for all timestamps while the self-attention mechanism is used only for the existing timestamps except for zero-padding ones. For 3D skeleton-based action recognition datasets, data preprocessing becomes more challenging because different skeleton joints may use different coordinate systems and may have different degrees of smoothness. Therefore, we normalize these coordinate systems using the average value of the hip center, left and right joints as the origin of the coordinate systems. Additionally, the Savitzky-Golay smoothing filter is applied to address the aforementioned smooth problem. These preprocessing greatly denoise the raw data and smooth the joint trajectories. Specifically, for the "HDM05" dataset, these operations are performed for every four flames as in [15].

4.3. Hyper-parameters setting

In the experiments, the parameters of the proposed MESAMN are jointly trained by the back-propagation [44] algorithm with the gradient descent method Adam [45]. In this section, the detailed information of the hyper-parameters in MESAMN are introduced.

1) Multi-head Self-attention Memory Encoder (MSME): The MSME has several hyper-parameters, including the number of reservoirs (Q), the number of reservoir units (N_{res}), the input scal-

Table 1
Attributes of all datasets.

Dataset	#Classes	#Variables	#Instances	Length	Task	Source
Arabic Digits	10	13	8800	[4, 93]	Speech Recognition	UCI
Australian Language (AL)	95	22	2565	[45, 136]	Sign Language Recognition	UCI
BCI	2	28	416	[500, 500]	EEG Classification	Blankertz
Character Trajectories (CT)	20	3	2858	[109, 205]	Handwriting Classification	UCI
CMU Subject 16 (CMU16)	3	62	58	[127, 580]	Action Recognition	CMU
ECG	2	2	200	[39, 152]	ECG Classification	Olszewski
Graz	2	3	280	[1152, 1152]	EEG Classification	Leeb
Japanese Vowels (JV)	9	12	640	[7, 29]	Speaker Recognition	UCI
Libras	15	2	360	[45, 45]	Sign Language Recognition	UCI
Non-invasive Fetal ECG (NIF-ECG)	42	2	3765	[750, 750]	ECG Classification	UCR
Pen Digits	10	2	10992	[8, 8]	Handwriting Recognition	UCI
Robot Failure LP1	4	6	88	[15, 15]	Robot Failure Recognition	UCI
Robot Failure LP2	5	6	47	[15, 15]	Robot Failure Recognition	UCI
Robot Failure LP3	4	6	47	[15, 15]	Robot Failure Recognition	UCI
Robot Failure LP4	3	6	117	[15, 15]	Robot Failure Recognition	UCI
Robot Failure LP5	5	6	164	[15, 15]	Robot Failure Recognition	UCI
Uwave Gesture Library (UWGL)	8	3	4478	[315, 315]	Gesture Recognition	UCR
Wafer	2	6	1194	[104, 198]	Manufacturing Classification	Olszewski
UTD-MHAD	27	60	861	[37, 121]	Action Recognition	Chen
HDM05	65	78	2339	[3, 221]	Action Recognition	Müller
Florence3D	9	45	215	[8, 35]	Action Recognition	Seidenari

ing factors (*IS*), the spectral radius (*SR*), and the sparsity factor (*SP*). Specifically, the number of reservoirs in the MSME (Q) is set in the range of [2,5]. The number of input unit N_{in} is set to be the number of variables for each dataset while N_{res} is set to be either 3 or 30 times of N_{in} for constructing a high-dimensional projection. *IS* is set between [1e-0, 1e-1. 1e-2, 1e-3] according to the range of values for each dataset, which helps to initialize the weight matrix $\mathbf{W}^{in}$ between the input layer and the reservoir to ensure the transformed values to fall into the unsaturated zone of the activation function. The *SR* is set to be one of [0.1,0.5,0.9] and *SP* is selected from the range of [0.1,0.9] according to the number of reservoirs[1], which are used for the initialization of the weight matrix W^{res}.

2) Convolutional Memory Learner (CML): The CML module has several hyper-parameters to be set, including the height of filters, the width of filters, and the number of filters. In our experiments, the height of filters of each ESN is set as a multiple (from 1 to $nb_{multiscale}$) of $2^{(n-1)}$, where the multiscale number ($nb_{multiscale}$) is set in [2, 3, 4, 5][2]. The width of filters is set to be equal to the number of reservoir units (N_{res}). Additionally, for each dataset, the number of filters (nb_{filter}) is set to be either 2 or 20 times of the number of classes. The dropout rate for the fully connected layer is set between [0, 0.5], which can prevent the model suffering from overfitting.

A grid-search and cross-validation are applied to select hyper-parameters using the aforementioned constraints. The average results on five independent experiments are reported for each dataset. Detail information of the final experimental hyper-parameters for each dataset are shown in Table 2.

4.4. Comparison with the existing methods

Multivariate Time Series (MTS) Classification Task: In this section, we conduct experiments on 18 MTS bench-mark datasets and compare the MESAMN with existing well-known methods: including distance-based methods (TDVM [35], DD_{DTW} [36], and MD_{DTW} [37]), feature-based methods (such as SMTS [38] and TSML [39]), ESN-based methods (C_{ADE} [14] and ConvMESN [13]), and a LSTM-based deep learning method.

1) TDVM: This is a classification method based on a temporal extension of discrete support vector machines and cardinality warping distances, which greatly benefits from the notions of warping distance and softened variable margin.

2) DD_{DTW}: This approach combines the DTW distance between MTSs with the DTW distance between derivatives of MTSs. Then, the nearest neighbor rule is used for classification.

3) MD_{DTW}: This method proposes a Mahalanobis distance-based dynamic time warping (MD_{DTW}) measure. Specifically, the Mahalanobis distance is aiming at calculating the local distance in MTSs and finding the relationships between variables to facilitate MTS classification tasks.

4) SMTS: This method proposes a new symbolic representation of the MTS (SMTS), which transforms an MTS into a feature matrix. Specifically, rows of the feature matrix represent a time index, the original values, and the gradient of time series at this timestamp.

5) TSML: This is a time series manifold learning (TSML) method utilizing the kernel principal component analysis (KPCA) to transform an MTS into a lower-dimensional feature space for classification.

6) C_{ADE}: This is a new MTS classification method which combines the conceptor model of ESN [26] and the adaptive differential evolution (ADE) algorithm. Firstly, the C_{ADE} projects the input time series into different state clouds by a conceptor. Then it constructs classifiers from these state clouds for different classes while the ADE algorithm is used for parameter optimization.

7) ConvMESN: This state-of-art multivariate time series classification method combines several multi-timescale reservoirs (each in different time spans) with a convolutional layer to capture the multitimescale structures in temporal data.8) LSTM: A deep learning baseline is constructed for comparision. Specifically, the average-pooling of hidden states (32 hidden units in LSTM hidden layer) is concated along time and then fed into a fully connected layer and softmax layer for classification. In addition, we replace the ESN of the proposed MESAMN with LSTM as a baseline named ConvLSTM. The hidden unit of LSTM is set to be one of {16, 32, 64, 128}. The other parameter settings are the same as those of MESAMN, and the best parameters are determined by cross-validation. Besides, we construct the "MESAMN w/o SA" as a

[1] To enable the echo states generated by multiple reservoirs more complementary with each other, the sparsity is uniform sampling between [0.1,0.9]. In fact, we have tried several configurations and find that this design is helpful to improve the performance of the MESAMN.

[2] n represents the n^{th} ESN in the MESAMN. The intuition of this design is that a larger convolutional filter is needed for better feature learning because the larger sparsity of the ESN has, the more difficult to capture dynamic echo states.

Table 2
Hyper-paremeters of MESAMN for each dataset.

Data set	Q	N_{in}	N_{res}	IS	SR	SP	$nb_{multiscale}$	nb_{filter}	dropout
Arabic Digits	4	13	3×13	1e-1	0.1	[0.1, 0.37, 0.63, 0.9]	3	2×10	0
Australian Language	4	22	3×22	1e-1	0.1	[0.1, 0.37, 0.63, 0.9]	4	2×95	0
BCI	2	28	3×28	1e-3	0.5	[0.1, 0.9]	2	20×2	0
Character Trajectories	5	3	30×3	1e-1	0.5	[0.1, 0.3, 0.5, 0.7, 0.9]	5	2×20	0.2
CMU Subject 16	3	62	3×62	1e-2	0.1	[0.1, 0.5, 0.9]	3	2×3	0
ECG	4	2	30×2	1e-2	0.1	[0.1, 0.37, 0.63, 0.9]	3	2×2	0
Graz	5	3	30×3	1e-1	0.1	[0.1, 0.3, 0.5, 0.7, 0.9]	5	2×2	0
Japanese Vowels	4	12	3×12	1e-0	0.1	[0.1, 0.37, 0.63, 0.9]	3	2×9	0
Libras	3	2	30×2	1e-0	0.5	[0.1, 0.5, 0.9]	3	2×15	0
Non-invasive Fetal ECG	5	2	30×2	1e-1	0.9	[0.1, 0.3, 0.5, 0.7, 0.9]	4	2×40	0
Pen Digits	3	2	3×2	1e-2	0.5	[0.1, 0.5, 0.9]	2	2×10	0
Robot Failure LP1	2	6	30×6	1e-3	0.5	[0.1, 0.9]	5	2×4	0.4
Robot Failure LP2	3	6	30×6	1e-2	0.9	[0.1, 0.5, 0.9]	2	2×5	0.2
Robot Failure LP3	2	6	30×6	1e-2	0.5	[0.1, 0.9]	4	2×4	0
Robot Failure LP4	2	6	30×6	1e-3	0.9	[0.1, 0.9]	3	2×3	0
Robot Failure LP5	2	6	30×6	1e-2	0.9	[0.1, 0.9]	3	2×5	0
Uwave Gesture Library	5	3	30×3	1e-1	0.1	[0.1, 0.3, 0.5, 0.7, 0.9]	5	2×8	0
Wafer	5	6	30×6	1e-2	0.5	[0.1, 0.3, 0.5, 0.7, 0.9]	4	20×2	0
UTD-MHAD	4	60	3×60	1e-1	0.1	[0.1, 0.37, 0.63, 0.9]	3	2×27	0
HDM05	4	78	3×78	1e-3	0.1	[0.1, 0.37, 0.63, 0.9]	3	2×65	0.5
Florence3D	4	45	3×45	1e-1	0.1	[0.1, 0.37, 0.63, 0.9]	2	2×9	0.4

baseline model to verify the effectiveness of the self-attention operations in the MESAMN. Note that hyper-parameters of the "MESAMN w/o SA" are set to be the same as the MESAMN's for a fair comparison.

Table 3 shows experimental results of methods which are evaluated by the cross-validation error rate. Results of these methods are collected from their original works [13,14,35–39]. In order to make a fair comparison with the compared methods, we have removed results produced by cross-validation which are different from ours. For instance, the results of the DD_{DTW} and the C_{ADE} on "Robot Failure" are not used because both of them apply 10-fold cross-validation on this dataset while other methods apply 5-fold cross-validation. To present a performance comparison, we calculated the rate of datasets that each method achieves the best results among all datasets. As shown in Table 3, we can see that:

a) The proposed MESAMN yields good performance on a wide range of downstream tasks. It outperforms other strong baselines by 11 of 18 tasks, which demonstrates its effectiveness for multivariate time series classification tasks.
b) ConvLSTM is not as good as MESAMN on multivariate time series classification tasks. In the experiments, we find that the cost time of the training procedure for ConvLSTM is more than ten times that of MESAMN, and the inference time is almost the same when the total parameters of two networks are comparable. These also demonstrate that the proposed MESAMN is a training-efficient network and has a good generalization on multivariate time series classification tasks.
c) The proposed MESAMN outperforms the baseline model "MESAMN w/o SA" on 16 out of 18 datasets in experiments. This demonstrates that the self-attention on multiple ESRs are important to improve the model's capacity for modeling complex time series. Combining Tables 1 and 3, we can find that MESAMN yields better performance on the time-series datasets with a maximum length larger than 50. For example, in the case of the 10 time-series classification tasks (the maximum length of samples is larger than 50), MESAMN shows the best performance on 8 out of 10 datasets, accounting for 80% best rate. On the other hand, MESAMN shows better performance than the compared methods on only 3 out of 8 short time series tasks (the maximum length of samples is shorter than 50), accounting for 37.5% best rate. These show that the MESAMN is more suitable for modeling long time series while it is natural to assume that long-term dependencies are more likely to exist in long time series.

3D Skeleton-Based Action Recognition Task: In our experiments, the MESAMN is compared with several existing action recognition methods [13,40,42,43,51,15,46–48,52–56,49,57,58,50] in recent years and the baseline model "MESAMN w/o SA" on three 3D skeleton-based action recognition datasets. Specifically, we take the cross-subject test accuracy as the evaluation criteria for the "UTD-MHAD" dataset and choose the cross-validation recognition accuracy as evaluation criteria for the "HDM05" and "Florence3D" datasets. Experimental results are shown in Table 4–6.

According to Table 4–6, the MESAMN outperforms the baseline "MESAMN w/o SA" in all of the three 3D skeleton-based action recognition datasets. These results demonstrate that the self-attention mechanism is useful for modeling complicated time series when combined with the traditional ESN. Furthermore, the MESAMN outperforms all methods under comparison on all of these 3D skeleton-based action recognition datasets in experiments. Note that most of the action sequence recognition methods (i.e., [42,43,51,15,46–48,52,53,13]) learn both spatial dependencies between different skeleton joints and temporal dependencies between different timestamps. Instead, the MESAMN only learns temporal dependencies through the multi-head self-attention mechanism without learning spatial dependencies explicitly. This also demonstrates that capturing the long-term dependencies of action sequences is essentially necessary and helpful to make accurate classifications.

4.5. Analysis of long-term dependencies

In this part, we verify that the self-attention mechanism combined with ESN can indeed help to capture long-term dependence. The specific two tasks, Copying and Pixel-by-Pixel MNIST, are typical tasks for verifying the model's ability to capture long-term dependence [59,10]. In these tasks, critical features for the objective task are far away from each other. Therefore, how to capture such long-term dependent feature is very important for downstream tasks. As shown in Fig. 3, the goal of the Copying task is to predict historical information a long time ago.

Table 3
Experimental results comparison between our model and other methods.

Dataset	TDVM [35]	DD_{DTW} [36]	MD_{DTW} [37]	SMTS [38]	TSML [39]	C_{ADE} [14]	ConvMESN [13]	LSTM	ConvLSTM	**w/o SA**	**MESAMN**
Arabic Digits	–	**0.002**	0.004	–	–	0.008	**0.002**	0.009	0.003	0.003	**0.002**
Australian Language	–	0.185	0.041	0.027	0.024	0.247	**0.018**	0.121	0.041	0.020	0.019
BCI	–	0.402	–	–	–	0.326	**0.310**	0.392	0.387	0.328	**0.310**
Character Trajectories	–	0.009	0.010	–	–	0.020	**0.004**	0.015	0.020	0.005	**0.004**
CMU Subject 16	–	0.037	–	0.007	**0.000**	**0.000**	**0.000**	**0.000**	0.026	**0.000**	**0.000**
ECG	–	0.145	–	0.134	**0.090**	0.100	**0.090**	0.111	0.261	0.115	**0.090**
Graz	–	0.307	–	–	–	0.304	0.247	0.382	0.351	0.246	**0.231**
Japanese Vowels	0.034	0.020	0.009	0.035	**0.005**	0.006	0.020	0.038	0.019	0.022	0.018
Libras	–	0.050	0.092	–	–	**0.042**	0.050	0.170	0.078	0.053	0.050
Non-invasive Fetal ECG	–	0.095	–	–	–	0.148	**0.068**	0.069	0.158	0.082	0.069
Pen Digits	0.037	**0.005**	0.006	0.010	0.057	0.016	0.006	0.014	0.010	0.006	**0.005**
LP1	0.148	–	0.079	0.062	–	–	0.078	0.126	0.100	0.068	**0.058**
LP2	0.362	–	0.277	0.384	**0.128**	–	0.269	0.289	0.280	0.248	0.244
LP3	0.319	–	0.255	0.189	**–**	–	**0.161**	0.229	0.189	0.167	0.164
LP4	0.145	–	**0.034**	0.063	0.060	–	0.072	0.102	0.100	0.063	0.063
LP5	0.329	–	0.275	0.261	0.250	–	0.236	0.287	0.238	0.232	**0.225**
Uwave Gesture Library	–	0.015	–	–	–	0.019	**0.012**	0.049	0.019	0.015	**0.012**
Wafer	–	0.019	0.010	0.010	–	0.017	0.023	0.037	0.030	0.018	**0.009**
No. datasets	7	13	12	11	8	13	**18**	**18**	**18**	**18**	**18**
No. best	0/7	2/13	1/12	0/11	4/8	2/13	9/18	1/18	0/18	1/18	**11**/18
Best rate	0.000	0.154	0.083	0.000	0.500	0.154	0.500	0.056	0.000	0.056	**0.611**

Table 4
Recognition accuracy (%) on "UTD-MHAD" data set (cross-subject test).

Methods	Accuracy(%)
Kinect & Inertial [40]	79.10
Con3DJ [46]	85.58
JTM [47]	85.81
SOS [48]	86.97
STDL-Net [49]	93.95
Bag-of-Poses [50]	95.30
ConvMESN [13]	96.74
MESAMN w/o SA	95.81
MESAMN	**96.79**

Table 5
Recognition accuracy (%) on "HDM05" data set (90% training – 10% testing splits).

Methods	Accuracy (%)
DNN [43]	95.59
Hierarchical LSTM [51]	96.92
Deep LSTM [52]	97.25
ConvESN-MSMC [15]	97.25
EMN-ESIF [54]	97.25
S3DGRF [55]	97.30
ConvMESN [13]	96.52
MESAMN w/o SA	96.67
MESAMN	**97.58**

Table 6
Recognition accuracy (%) on "FLORENCE3D" data set (cross-subject test).

Methods	Accuracy (%)
Skeletons Lie Group [53]	90.88
FTP representation [56]	91.40
ConvESN-MSMC [15]	91.72
ConvMESN [13]	93.30
MESAMN w/o SA	93.05
MESAMN	**93.38**

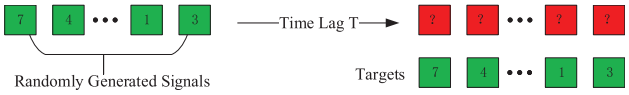

Fig. 3. Illustration of the Copying task. In this task, the objective of the model is to predict the original signals with time lag T, which causes a challenge for the model to capture the critical information a long time ago.

As a special RNN, ESN is easy to lose history information with the growth of time in the reservoir layer because the current hidden state is only influenced by the last one. In this paper, we employ the self-attention mechanism to model the long-term dependence of echo state representations globally.

Specifically, the convolutional memory learner is replaced by a softmax layer and only one ESN in the encoding structure is used to clearly demonstrate how the self-attention operator is of great benefit to the traditional ESN. We regard these two tasks as an eight-class classification task and a ten-class classification task, respectively, and choose the ESRs at the last timestamps for classification. Experimental results on these two benchmarks show that ESRs followed by the self-attention mechanism capture long-term dependencies well in the existing time series systems while the traditional ESRs can hardly capture long-term dependencies.

1) Copying: The Copying task is introduced in [59]. Supposed we have a $(T+20)-length$ array, the first ten numbers are randomly chosen from integers between 0 and 7, followed by $(T-1)$ numbers 8, and then followed by number 9. Specifically, the number 9 is a signal to represents the beginning of the prediction, and the task is to generate the last ten numbers as having chosen at the beginning. In this task, the number of neurons in the reservoir is set to be 30 while the input scaling factor and the spectral radius of the reservoir are both set to be 0.1. We repeat the experiment five times per group and Table 7 reports both the average accuracy and the last ten number cross-entropy, with their standard deviations. As shown in Table 7, results of the traditional ESRs used for classification are nearly equivalent to random guesses (the random guess accuracy on the Copying task is 0.1250). This means that the echo state features generated by the traditional ESN have little effect on classification. In contrast, the

Table 7
Accuracy and cross-entropy on the Copying task with different time lags.

Timelag	ESRs		Self-attenion ESRs	
	Accuracy	CE_{10}	Accuracy	CE_{10}
T = 10	0.1266 ± 0.0035	2.0825 ± 0.0018	**0.2238 ± 0.0104**	**1.8927 ± 0.0507**
T = 50	0.1229 ± 0.0044	2.0834 ± 0.0032	**0.1931 ± 0.0048**	**1.9796 ± 0.0219**
T = 100	0.1226 ± 0.0041	2.0817 ± 0.0021	**0.1735 ± 0.0103**	**2.0721 ± 0.0050**
T = 200	0.1212 ± 0.0066	2.0834 ± 0.0022	**0.1564 ± 0.0124**	**2.0770 ± 0.0040**
T = 500	0.1235 ± 0.0040	2.0824 ± 0.0018	**0.1395 ± 0.0068**	**2.0797 ± 0.0036**

echo state features after the self-attention mechanism demonstrate a significant discriminant performance and the accuracy increases by nearly 10% when the time lag is 10. As the forgetting time extends, the Copying task becomes more challenging and the performance becomes worse. Nevertheless, when the time lag increases to 200, the accuracy of the Copying task based on self-attention ESRs is higher than the traditional ESRs by 3%. Even when the time lag increases to 500, the performance of the Copying task based on self-attention ESRs still shows a little advantage.

2) MNIST: The MNIST classification task consists of 70 K images with 28 × 28 size, 60 K for training and 10 K for testing. In this study, we flattened 28 × 28 pixels to a 784-length sequence and treated it as a sequence classification task to verify our proposed method. In this task, the number of neurons in the reservoir is set to be 30 while the input scaling factor and the spectral radius of the reservoir are both set to be 0.1. We repeat the experiment five times per group and report the average accuracy and the standard deviation in Table 8. In the Sequential MNIST classification task, the 784-length sequences are used for classification by flattening 28 × 28 pixels of MNIST images, as shown in Fig. 4. According to Table 8, the result of the traditional ESRs used for classification is a bit better than random guess (the random guess accuracy on the sequential MNIST classification task is 0.1000 while the traditional ESRs based classification accuracy is 0.1115). This also means that the echo state features generated by the traditional ESN show little effect on classification. The echo state features after the self-attention mechanism yield better performance compared with the traditional ESRs and improve the accuracy by about 8% (from 0.1115 to 0.1932).

In the Permuted MNIST classification task, the order of the 784-length sequence is randomly permuted to make the classification task more challenging than the sequential MNIST task. Surprisingly, the performance of the traditional ESRs-based method is much better than it's performance on the sequential MNIST classification task. This further demonstrates that the traditional ESN is great in modeling chaotic time series but lacks the capacity for capturing long-range dependency (compared with the sequential MNIST). Nevertheless, the accuracy of the Permuted MNIST classification task based on self-attention ESRs is higher than the traditional ESRs by 0.71%.

Table 8
Accuracy on the MNIST classification task.

Method	Sequential MNIST	Permuted MNIST
ESRs	0.1115 ± 0.0028	0.1265 ± 0.0013
Self-attenion ESRs	**0.1932 ± 0.0031**	**0.1336 ± 0.0061**

Fig. 4. Illustration of the Sequential MNIST classification task. In this task, the objective of the model is to predict the label of the sequential MNIST (right, 1 × 784), which is reshaped from the original image (28 × 28). This requires the proposed method to have the ability to capture long-term dependencies existing in the sequence.

It is worth noting that in this subsection, our purpose is to verify the ability of ESN for capturing long-term dependencies when it is combined with the self-attention mechanism. In order to eliminate other interference, we only use the echo state features of the last timestamp in ESNs for feature learning, and the memory learner is simplified as a single softmax layer. These result in poor accuracies for both the "Copying" and the "MNIST" experimental tasks. Nevertheless, we can conclude that the self-attention mechanism helps to capture long-term dependencies in the traditional echo states.

4.6. Model ablation study

1) The impact of multi-head and self-attention: we systematically research whether the multi-head mechanism and the self-attention mechanism yield great importance to the MESAMN. Note that we only explore the MSME structure in this study for improving the memory capacity of the ESN in this part and the CML structure is kept invariant. Therefore, several baselines are constructed to compare with the MESAMN:

a) MESAMN w/o multi-head: a single ESN followed by the self-attention mechanism is used for classification.
b) MESAMN w/o self-attention: multiple ESNs without any self-attention mechanism are used for classification.
c) MESAMN w/o multi-head and self-attention: a single ESN without self-attention mechanism is used for classification.

For the multi-head mechanism, we control the single-head and multi-head mapping variables to evaluate its impact on the model performance while maintaining the consistency of other parameter variables. The single-head model has the same parameter selection rules as the multi-head one. The corresponding difference is that Q is set to be 1 in the single-head model. This corresponding network structure does not contain multiple ESN features. So, the extracted features are not as abundant as those extracted from the multi-head one and results in a poor performance in the classification task. For the self-attention mechanism module, we keep the search range of all the parameters of the model consistent between MESAMN and MESAMN w/o self-attention (also between "MESAMN w/o multi-head" and "MESAMN w/o multi-head and self-attention"). The self-attention mechanism module does not introduce additional parameters, so the comparative experiment verifies the importance of the self-attention mechanism. We choose the 3D skeleton-based action recognition datasets as the experimental datasets to make a comparison between the full model and baseline models.

Results of the ablation study are shown in Fig. 5. They show that models without the multi-head mechanism yield higher variances (shown in "the red bar" and "the green bar") in comparison with ones with the multi-head mechanism (shown in "the purple bar"

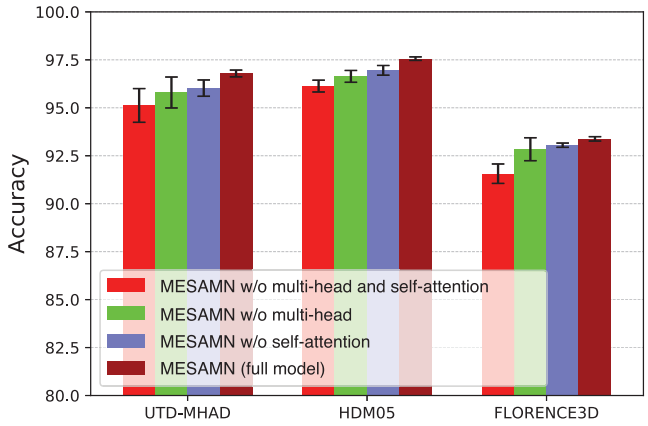

Fig. 5. The ablation model accuracy and the corresponding standard deviation towards three 3D skeleton-based action recognition tasks.

and "the brown bar"). Moreover, the multi-head mechanism helps to boost the performance (e.g., from "the red bar" to "the purple bar"). Furthermore, the self-attention mechanism also improves the performance of models in comparison (e.g., from "the red bar" to "the green bar"). We can conclude that the multi-head mechanism is helpful to improve the robustness of models (yielding a smaller standard deviation). Additionally, both the self-attention mechanism and the multi-head mechanism are helpful to enhance the modeling ability of the model which improves the performance of the models. Our full model, MESAMN, is the most robust one and yields the best performance in all of these datasets compared with the baseline methods, which further demonstrates its effectiveness.

2) Why multiscale conv layer: we conduct ablation experiments on the multiscale convolutional layer. Firstly, we remove all convolution operations and flatten the Enhanced Echo State Representations (E^2SRs). Then the E^2SRs are followed by a fully connected layer and a softmax layer is performed for classification. Experimental results are shown in Table 9. The MESAMN without convolutional layer (MESAMN w/o conv layer) yields far inferior classification performance in comparison to MESAMN. In our experiments, we find that only using only one multiscale convolution layer in the MESAMN is enough for developing an effective classification framework.

In order to verify the necessity of multiscale convolution, we also make a comparison baseline with single-scale convolution in this ablation study. We find that the multiscale convolution layer is more suitable for modeling the multiscale echo state representations compared with the single-scale convolution one. Specifically, we keep the structure of single-scale convolution consistent with the proposed MESAMN and control the number of parameters on the convolution layer so that the amount of convolution parameters of single-scale convolution is comparable to that of multiscale convolution. The results are shown in Table 10. It can be seen that multiscale convolution is more conducive to capturing the features formed by ESN. This shows that the multiscale convolution layer is indeed a necessary choice for capturing multiscale features in ESN when applying to downstream tasks.

Table 9
Accuracies (%) of the MESAMN w/o conv layer and the proposed MESAMN.

Methods	UTD-MHAD	HDM05	FLORENCE3D
MESAMN w/o conv layer	70.14	81.03	76.07
MESAMN	**96.79**	**97.58**	**93.38**

Table 10
Accuracies (%) of the MESAMN with a single-scale conv layer and the proposed MESAMN.

Methods	UTD-MHAD	HDM05	FLORENCE3D
single-scale conv layer	95.44	96.64	93.06
MESAMN	**96.79**	**97.58**	**93.38**

4.7. Visualization of features

In this part, we visualize the characteristics of the echo states in different ESNs, which shows great complementary features. Here we show a visualization of the ESN features in MESAMN on a "jog" sample in the UTD-MHAD dataset. As shown in Fig. 6, the signal input at the same time will be sent to three ESNs with different internal structure connection to generate different Echo State Representations (ESRs). For instance, we highlight the ESRs at 20^{th} timestamp which shows great difference among ESRs between different reservoirs. The ablation experiments on the multi-head mechanism also show that multiple ESNs have a good impact on the proposed method, which further demonstrates that ESRs from different reservoirs are complementary to each other and helps the model to generalized on downstream tasks. Actually, ESRs in different reservoirs may create redundancy, but also yields superior performance compared to a single one, which has been demonstrated in the ablation study. In the proposed MESAMN, we use a multiscale convolutional layer to capture the abundant features generated by the ESN, and we use the max-pooling-over-time layer to avoid the redundancy of the features.

4.8. Computational efficiency analysis

The proposed MESAMN has high computational efficiency because the MSME structure is training-free and the CML structure is just a shallow convolutional neural network. Specifically, the MSME structure consists of several reservoirs in which connection weights are randomly initialized and fixed during the training stage. Moreover, the CML structure consists of only one convolutional layer and one fully connected layer, which is quite efficient to train. Quantitatively, we make a comparison with the baseline on the computational time. For the training stage, the proposed

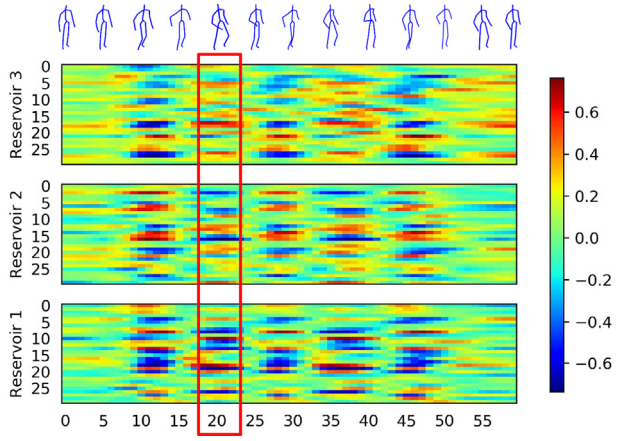

Fig. 6. Heatmap of original echo state representations of a sequence (60 frames) of the "jog" action in "UTD-MHAD" dataset yielded by three reservoirs (each has 30 neurons). Specifically, we show the skeleton-based postures of the action at every five moments. Echo State Representations (ESRs) from different ESNs corresponding to the twentieth timestamp are highlighted in the red boxes, which shows complementary features.

MESAMN is almost efficient as the "MESAMN w/o SA" while the ConvLSTM costs nearly 10 × time. The ConvLSTM has high computational costs since the network parameters in LSTM need to be trained. For the testing stage, the proposed MESAMN is a little slower (less than 2 × time) than the other two baselines since the self-attention mechanism needs extra calculation. Nevertheless, our proposed MESAMN yields superior performance on various downstream tasks.s.

Although the proposed MESAMN exhibits superior performance on downstream tasks, our model still has following the limitation. Since we need to collect the Echo State Representations (ESRs) along time and employ the multi-head self-attention mechanism over all the ESRs to obtain E^2SRs, MESAMN will consume relatively large memory space.

5. Conclusion

In this paper, we find that the short-term memory ability of ESN is not friendly enough for downstream classification tasks, while the self-attention mechanism helps to enhance the global features of echo state representations. Besides, the multiscale convolutional layer is a good choice for capturing the multiscale features generated by the echo state networks. In addition, the multi-head mechanism helps the model to generate abundant features, which helps to enhance the performance and stability of the proposed model. To the best of our knowledge, the proposed MESAMN is the first to combine the advantage of ESN and the multi-head self-attention mechanism, which is efficient for modeling complex time series data. Experiments on a wide range of datasets from MTS benchmark datasets to 3D skeleton-based action recognition datasets demonstrate the effectiveness of the MESAMN. In the future, we will make further studies on the CML structure and design powerful architectures. Moreover, due to the roughness of self-attention for modeling long-term dependencies, it is quite interesting to further improve the performance of MESAMN by using the recent techniques [60], which models the localness of self-attention networks.

CRediT authorship contribution statement

Huizi Lyu: Writing - review & editing. **Desen Huang:** Software, Writing - original draft. **Sen Li:** Investigation. **Wing W.Y. Ng:** Supervision. **Qianli Ma:** Conceptualization, Methodology, Supervision.

Data availability

Data will be made available on request.

Declaration of Competing Interest

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

Acknowledgement

The authors would like to thank Peitian Ma in computer science from the South China University of Technology, Guangzhou, China, for supporting partial experiments. The work described in this paper was partially funded by the National Natural Science Foundation of China (Grant Nos. 62272173, 61872148), the Natural Science Foundation of Guangdong Province (Grant Nos. 2022A1515010179, 2019A1515010768).

Appendix A. Appendix

Here, we present the preliminaries of "Echo State Network" and "Self-Attention Mechanism" for readers.

A.1. Echo state network

In this section, we will briefly introduce the general architecture and the characteristics of the Echo State Network (ESN). As shown in Fig. 7, the general architecture of an ESN mainly consists of three

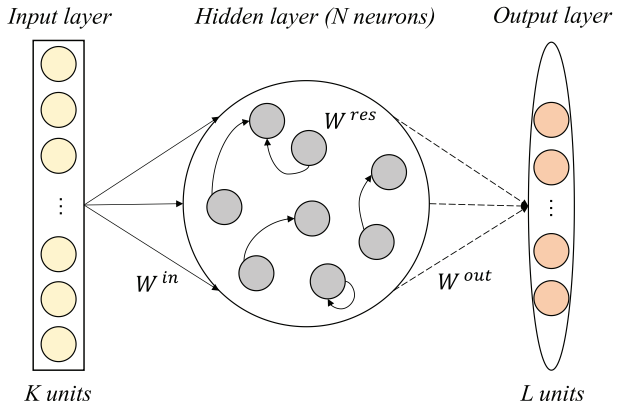

Fig. 7. The general architecture of ESN.

essential components: an input layer, a recurrent hidden layer (also called a reservoir) and an output layer. The weights of the input layer to the hidden layer and the ones in the hidden layer are randomly initialized and fixed during training. The only adaptable parameters are the weights of the hidden layer to the output layer and can be efficiently learned by linear regression [3]. Let $\mathbf{u}(t), \mathbf{x}(t)$ and $\mathbf{y}(t)$ indicate the original signals, the echo states, and the predict outputs at timestamp t, respectively, the mathematical formulations for the simple ESN are given as follows:

$$\mathbf{x}(t) = f\left(\mathbf{W}^{res}\mathbf{x}(t-1) + \mathbf{W}^{in}\mathbf{u}(t)\right), \tag{15}$$

$$\mathbf{y}(t) = f^{out}\left(\mathbf{W}^{out}\mathbf{x}(t)\right), \tag{16}$$

where $\mathbf{u}(t) \in \mathbb{R}^{K\times 1}, \mathbf{x}(t) \in \mathbb{R}^{N\times 1}, \mathbf{W}^{in} \in \mathbb{R}^{N\times K}, \mathbf{W}^{res} \in \mathbb{R}^{N\times N}$ and $\mathbf{W}^{out} \in \mathbb{R}^{L\times N}$. K, N, and L denote the numbers of neurons in the input layer, the hidden layer, and the output layer, respectively. An ESN has three major characteristics:

1) Temporal Kernel: The reservoir is the core of the whole system which consists of a large number (typically 100 to 1000 times of the dimensions of inputs) of neurons and acts as a "temporal kernel". The input time series is projected onto a high dimensional space by the reservoir.

2) Echo State Property (ESP): The ESP guarantees the stability of the reservoir in ESN by generating similar echo states when the input time series have similar short-term histories [3,61].

3) High Training Efficiency: Weights of the linear readout layer can be computed by a simple linear regression algorithm because weights of the input layer and the reservoir are randomly initialized and fixed during the training stage.

Overall, the ESN has a very efficient training procedure and is very useful for modeling dynamical systems owing to the high-dimensional projection and highly sparse connectivity of neurons in the reservoir. However, most ESN-based classifiers lack the ability to capture long-term dependencies, which is important for modeling complex time series systems. In this paper, we exploit

the "Temporal Kernel" and the "Echo State Property" to construct Echo State Representations (ESRs) for time series. Moreover, we model the long-term dependencies existing in the time series by combining the ESRs and the self-attention mechanism.

A.2. Self-attention mechanism

The self-attention mechanism (also called intra-attention) is a way of capturing dependencies within representations, which has made great progress in natural language processing [17]. Fig. 8 presents an intuitive graphical explanation of the self-attention mechanism in our proposed model. In the case of echo state representation learning, the original representation matrix collects all echo states of the reservoir along time. Then, the self-attention map is computed according to the similarities of echo states at different timestamps. After that, the self-attention map was multiplied with the original representations matrix, resulting in the self-attention feature maps. In this way, the self-attention feature at each timestamp contains the context information, which is essential for modeling the existing long-term dependencies in time series.

The mathematical formulation of the self-attention in our work can be summarized as follows:

$$\mathbf{X} = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1N} \\ x_{21} & x_{22} & \dots & x_{2N} \\ \dots & \dots & \dots & \dots \\ x_{T1} & x_{T2} & \dots & x_{TN} \end{pmatrix}, \tag{17}$$

$$\mathbf{W} = \mathrm{Softmax}\left(\frac{\mathbf{X}\mathbf{X}^{\mathrm{T}}}{\gamma}\right), \tag{18}$$

$$\mathbf{X}_{scale} = (\|\mathbf{x}_1\|_2, \|\mathbf{x}_2\|_2, \cdots, \|\mathbf{x}_T\|_2)^{\mathrm{T}}, \tag{19}$$

$$\gamma = \mathbf{X}_{scale}\mathbf{X}_{scale}^{\mathrm{T}}, \tag{20}$$

$$\widetilde{\mathbf{X}} = \mathbf{W}\mathbf{X}, \tag{21}$$

where $\mathbf{X} \in \mathbb{R}^{T\times N}$, $\mathbf{W} \in \mathbb{R}^{T\times T}$ and $\widetilde{\mathbf{X}} \in \mathbb{R}^{T\times N}$. Specifically, $\mathbf{X}, \mathbf{W}, \widetilde{\mathbf{X}}, \mathbf{x}_i$ and γ denote the original representations matrix, the self-attention map, the self-attention feature maps, the echo state representations at timestamp i, a scaling factor being set according to tasks, respectively. Note that the Softmax function are applied for each row of the input and we use a elementy-wise division in Eq. (18). In this study, we used the L_2 norm of $\mathbf{X}$ along time to compute γ by Eq. (19) and Eq. (20). Elements in γ represent different constraints for scaling echo states at different timestamps, which enable the proposed method to model the dependencies flexibly.

References

[1] Herbert Jaeger, Mantas Lukoševičius, Dan Popovici, Udo Siewert, Optimization and applications of echo state networks with leaky-integrator neurons, Neural Networks 20 (3) (2007) 335–352.

[2] Shiping Wen, Rui Hu, Yin Yang, Tingwen Huang, Zhigang Zeng, and Yong-Duan Song. Memristor-based echo state network with online least mean square. IEEE Trans. Syst., Man, Cybern.: Syst., 2018.

[3] Herbert Jaeger, 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) (2001) 13.

[4] Ning Li, Jianyong Tuo, Youqing Wang, Menghui Wang, Prediction of blood glucose concentration for type 1 diabetes based on echo state networks embedded with incremental learning, Neurocomputing 378 (2020) 248–259.

[5] Cuili Yang, Junfei Qiao, Honggui Han, Lei Wang, Design of polynomial echo state networks for time series prediction, Neurocomputing 290 (2018) 148–160.

[6] Seong Ik Han, Jang Myung Lee, Fuzzy echo state neural networks and funnel dynamic surface control for prescribed performance of a nonlinear dynamic system, IEEE Trans. Ind. Electron. 61 (2) (2014) 1099–1112.

[7] Chong Liu, Huaguang Zhang, Shaoxin Sun, He Ren, Online h∞ control for continuous-time nonlinear large-scale systems via single echo state network, Neurocomputing 448 (2021) 353–363.

[8] Qiang Chen, Linlin Shi, Jing Na, Xuemei Ren, Yurong Nan, Adaptive echo state network control for a class of pure-feedback systems with input and output constraints, Neurocomputing 275 (2018) 1370–1382.

[9] Mark D Skowronski, John G Harris, Automatic speech recognition using a predictive echo state network classifier, Neural Networks 20 (3) (2007) 414–423.

[10] Nan Rosemary Ke, Anirudh Goyal ALIAS PARTH GOYAL, Olexa Bilaniuk, Jonathan Binas, Michael C Mozer, Chris Pal, and Yoshua Bengio. Sparse attentive backtracking: Temporal credit assignment through reminding. In Advances in Neural Information Processing Systems, pages 7651–7662, 2018.

[11] Nikolaos Passalis, Anastasios Tefas, Juho Kanniainen, Moncef Gabbouj, Alexandros Iosifidis, Temporal bag-of-features learning for predicting mid price movements using high frequency limit order book data, IEEE Trans. Emerg. Top. Comput. Intell. 4 (6) (2018) 774–785.

[12] Zhichen Gong, Huanhuan Chen, Bo Yuan, Xin Yao, Multiobjective learning in the model space for time series classification, IEEE Trans. Cybern. 49 (3) (2018) 918–932.

[13] Q. Ma, E. Chen, Z. Lin, J. Yan, Z. Yu, W.W.Y. Ng, Convolutional multitimescale echo state network, IEEE Trans. Cybern. (2019) 1–13.

[14] Lin Wang, Zhigang Wang, Shan Liu, An effective multivariate time series classification approach using echo state network and adaptive differential evolution algorithm, Expert Syst. Appl. 43 (2016) 237–249.

[15] Qianli Ma, Lifeng Shen, Enhuan Chen, Shuai Tian, Jiabing Wang, Garrison W Cottrell, Walking walking walking: Action recognition from action echoes, in: Proceedings of the 26th International Joint Conference on Artificial Intelligence, AAAI Press, 2017, pp. 2457–2463.

[16] Huanhuan Chen, Fengzhen Tang, Peter Tino, Anthony G Cohn, Xin Yao, Model metric co-learning for time series classification, in: Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, AAAI Press, 2015, pp. 3387–3394.

[17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Advances in Neural Information Processing Systems, pages 5998–6008, 2017.

[18] Shu-xian Lun, Xian-shuang Yao, Hu. Hai-feng, A new echo state network with variable memory length, Inf. Sci. 370 (2016) 103–119.

[19] Georg Holzmann, Helmut Hauser, Echo state networks with filter neurons and a delay&sum readout, Neural Networks 23 (2) (2010) 244–256.

[20] Hongyan Cui, Chen Feng, Yuan Chai, Ren Ping Liu, and Yunjie Liu. Effect of hybrid circle reservoir injected with wavelet-neurons on performance of echo state network. Neural Networks, 57:141–151, 2014.

[21] Mohd-Hanif Yusoff, Joseph Chrol-Cannon, Yaochu Jin, Modeling neural plasticity in echo state networks for classification and regression, Inf. Sci. 364 (2016) 184–196.

[22] Rahma Fourati, Boudour Ammar, Javier Sanchez-Medina, Adel M Alimi, Unsupervised learning in reservoir computing for eeg-based emotion recognition, IEEE Trans. Affect. Comput. (2020).

[23] Naima Chouikhi, Boudour Ammar, Amir Hussain, Adel M Alimi, Novel single and multi-layer echo-state recurrent autoencoders for representation learning, Eng. Appl. Artif. Intell. 114 (2022).

[24] Qianli Ma, Lifeng Shen, Weibiao Chen, Jiabin Wang, Jia Wei, Yu. Zhiwen, Functional echo state network for time series classification, Inf. Sci. 373 (2016) 1–20.

[25] Lin Wang, Yi Zeng, Tao Chen, Back propagation neural network with adaptive differential evolution algorithm for time series forecasting, Expert Syst. Appl. 42 (2) (2015) 855–863.

[26] Herbert Jaeger. Controlling recurrent neural networks by conceptors. arXiv preprint arXiv:1403.3369, 2014.

[27] Zeeshan Khawar Malik, Amir Hussain, Wu. Qingming Jonathan, Multilayered echo state machine: a novel architecture and algorithm, IEEE Trans. Cybern. 47 (4) (2017) 946–959.

[28] Dandan Yang, Huanhuan Chen, Yinlong Song, Zhichen Gong, Granger causality for multivariate time series classification, in: Big Knowledge (ICBK), 2017 IEEE International Conference on IEEE, 2017, pp. 103–110.

[29] Dua Dheeru, Efi Karra Taniskidou, Uci machine learning repository, University of California, School of Information and Computer Sciences, Irvine, 2017.

[30] Eamonn Keogh. The ucr time series classification/clustering home-page. http://www. cs. ucr. edu/ eamonn/time_series_data/, 2006.

[31] M Shell. Carnegie mellon university motion capture database, 2012.

[32] Robert T Olszewski. Generalized feature extraction for structural pattern recognition in time-series data. Technical report, Carnegie-mellon Univ Pittsburgh PA School of Computer Science, 2001.

[33] Benjamin Blankertz, Gabriel Curio, and Klaus-Robert Müller. Classifying single trial eeg: Towards brain computer interfacing. In Advances in neural information processing systems, pages 157–164, 2002.

[34] Robert Leeb, Felix Lee, Claudia Keinrath, Reinhold Scherer, Horst Bischof, Gert Pfurtscheller, Brain–computer communication: motivation, aim, and impact of exploring a virtual apartment, IEEE Trans. Neural Syst. Rehabil. Eng. 15 (4) (2007) 473–482.

[35] Carlotta Orsenigo, Carlo Vercellis, Combining discrete svm and fixed cardinality warping distances for multivariate time series classification, Pattern Recogn. 43 (11) (2010) 3787–3794.

[36] Tomasz Górecki, Maciej Łuczak, Multivariate time series classification with parametric derivative dynamic time warping, Expert Syst. Appl. 42 (5) (2015) 2305–2312.

[37] Jiangyuan Mei, Meizhu Liu, Yuan-Fang Wang, Huijun Gao, Learning a mahalanobis distance-based dynamic time warping measure for multivariate time series classification, IEEE Trans. Cybern. 46 (6) (2016) 1363–1374.
[38] Mustafa Gokce Baydogan, George Runger, Learning a symbolic representation for multivariate time series classification, Data Min. Knowl. Disc. 29 (2) (2015) 400–422.
[39] Colin O'Reilly, Klaus Moessner, Michele Nati, Univariate and multivariate time series manifold learning, Knowl.-Based Syst. 133 (2017) 1–16.
[40] Chen Chen, Roozbeh Jafari, Nasser Kehtarnavaz, Utd-mhad: A multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor, in: Image Processing (ICIP), 2015 IEEE International Conference on IEEE, 2015, pp. 168–172.
[41] Meinard Müller, Tido Röder, Michael Clausen, Bernhard Eberhardt, Björn Krüger, and Andreas Weber. Documentation mocap database hdm05. Computer Graphics Technical Reports, 2007.
[42] Lorenzo Seidenari, Vincenzo Varano, Stefano Berretti, Alberto Bimbo, Pietro Pala, Recognizing actions from depth cameras as weakly aligned multi-part bag-of-poses, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2013, pp. 479–485.
[43] Kyunghyun Cho and Xi Chen. Classifying and visualizing motion capture sequences using deep neural networks. In Computer Vision Theory and Applications (VISAPP), 2014 International Conference on, volume 2, pages 122–130. IEEE, 2014.
[44] David E Rumelhart, Geoffrey E Hinton, Ronald J Williams, Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for, Cognit. Sci. (1985).
[45] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.
[46] Mohamed E Hussein, Marwan Torki, Mohammad Abdelaziz Gowayyed, Motaz El-Saban, Human action recognition using a temporal hierarchy of covariance descriptors on 3d joint locations, IJCAI 13 (2013) 2466–2472.
[47] Pichao Wang, Zhaoyang Li, Yonghong Hou, Wanqing Li, Action recognition based on joint trajectory maps using convolutional neural networks, in: Proceedings of the 2016 ACM on Multimedia Conference ACM, 2016, pp. 102–106.
[48] Yonghong Hou, Zhaoyang Li, Pichao Wang, Wanqing Li, Skeleton optical spectra based action recognition using convolutional neural networks, IEEE Trans. Circuits Syst. Video Technol. (2016).
[49] Qianli Ma, Shuai Tian, Jia Wei, Jiabing Wang, and Wing WY Ng. Attention-based spatio-temporal dependence learning network. Inform. Sci., 503:92–108, 2019.
[50] Saeid Agahian, Farhood Negin, Cemal Köse, Improving bag-of-poses with semi-temporal pose descriptors for skeleton-based action recognition, Visual Comput. 35 (4) (2019) 591–607.
[51] Du. Yong, Wei Wang, Liang Wang, Hierarchical recurrent neural network for skeleton based action recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1110–1118.
[52] Wentao Zhu, Cuiling Lan, Junliang Xing, Wenjun Zeng, Yanghao Li, Li Shen, and Xiaohui Xie. Co-occurrence feature learning for skeleton based action recognition using regularized deep lstm networks. In Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, vol. 30. AAAI Press, 2016.
[53] Raviteja Vemulapalli, Felipe Arrate, Rama Chellappa, Human action recognition by representing 3d skeletons as points in a lie group, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 588–595.
[54] Qianli Ma, Wanqing Zhuang, Lifeng Shen, Garrison W Cottrell, Time series classification with echo memory networks, Neural Networks 117 (2019) 225–239.
[55] D. Anil Kumar, A.S.C.S. Sastry, P.V.V. Kishore, E. Kiran Kumar, M. Teja Kiran, Kumar: S3drgf: Spatial 3-d relational geometric features for 3-d sign language representation and recognition, IEEE Signal Process. Lett. 26 (1) (2019) 169–173.
[56] Raviteja Vemulapalli and Rama Chellapa. Rolling rotations for recognizing human actions from 3d skeletal data. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016.
[57] Wu. Yirui, Lianglei Wei, Yucong Duan, Deep spatiotemporal LSTM network with temporal pattern feature for 3d human action recognition, Comput. Intell. 35 (3) (2019) 535–554.
[58] Bin Sun, Dehui Kong, Shaofan Wang, Lichun Wang, Yuping Wang, Baocai Yin, Effective human action recognition using global and local offsets of skeleton joints, Multim. Tools Appl. 78 (5) (2019) 6329–6353.
[59] Sepp Hochreiter, Jürgen Schmidhuber, Long short-term memory, Neural Computation 9 (8) (1997) 1735–1780.
[60] Baosong Yang, Tu. Zhaopeng, Derek F Wong, Fandong Meng, Lidia S Chao, Tong Zhang, Modeling localness for self-attention networks, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, 2018, pp. 4449–4458.
[61] Izzet B Yildiz, Herbert Jaeger, Stefan J Kiebel, Re-visiting the echo state property, Neural Networks 35 (2012) 1–9.

Huizi Lyu received the master's degree in Control Theory and Control Engineering from Guangdong University of Technology, Guangdong, China, in 2009. She is currently a Lecturer with the Aircraft Maintenance Engineering College, Guangzhou Civil Aviation College, Guangdong, China. Her research interests include machine learning and deep learning.

Desen Huang Received the master's degree in School of Computer Science & Engineering, the South China University of Technology in 2021. He is currently working as a machine learning and data mining engineer in Baidu. His current research interests include machine learning, deep learning, and time series modeling.

Sen Li received the master's degree in computer science from the South China University of Technology, Guangzhou, China, in 2020. His research interests include machine learning and deep learning.

Wing W. Y. Ng (S'02-M'05-SM'15) received the B.Sc. and Ph.D. degrees in computer science from Hong Kong Polytechnic University, Hong Kong, in 2001 and 2006, respectively. He is a Professor with the School of Computer Science and Engineering, South China University of Technology, Guangzhou, China, where he is currently the Deputy Director of the Guangdong Provincial Key Laboratory of Computational Intelligence and Cyberspace Information. His current research interests include neural networks, deep learning, smart grid, smart health care, smart manufacturing, and nonstationary information retrieval. Dr. Ng is currently an Associate Editor of the International Journal of Machine Learning and Cybernetics. He is a Principle Investigator of four China National Nature Science Foundation projects and a Program for New Century Excellent Talents in University from China Ministry of Education. He served as the Board of Governor of IEEE Systems, Man and Cybernetics Society from 2011 to 2013.

Qianli Ma (M'17) received the Ph.D. degree in computer science from the South China University of Technology, Guangzhou, China, in 2008. He is a Professor with the School of Computer Science and Engineering, South China University of Technology. From 2016 to 2017, he was a Visiting Scholar with the University of California at San Diego, La Jolla, CA, USA. His current research interests include machine learning algorithms, data-mining methodologies, and time-series modeling and their applications.