Neurocomputing 396 (2020) 302–313

Contents lists available at ScienceDirect

Neurocomputing

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

A novel time series forecasting model with deep learning

Zhipeng Shen, Yuanming Zhang*, Jiawei Lu, Jun Xu, Gang Xiao

College of Computer Science & Technology, Zhejiang University of Technology, Hangzhou 310023, China

ARTICLE INFO

Article history:
Received 30 March 2018
Revised 10 November 2018
Accepted 6 December 2018
Available online 24 April 2019

Keywords:
Deep learning
Time series forecasting
Long Short-Term Memory
Dilated causal convolution

ABSTRACT

Time series forecasting is emerging as one of the most important branches of big data analysis. However, traditional time series forecasting models can not effectively extract good enough sequence data features and often result in poor forecasting accuracy. In this paper, a novel time series forecasting model, named SeriesNet, which can fully learn features of time series data in different interval lengths. The SeriesNet consists of two networks. The LSTM network aims to learn holistic features and to reduce dimensionality of multi-conditional data, and the dilated causal convolution network aims to learn different time interval. This model can learn multi-range and multi-level features from time series data, and has higher predictive accuracy compared those models using fixed time intervals. Moreover, this model adopts residual learning and batch normalization to improve generalization. Experimental results show our model has higher forecasting accuracy and has greater stableness on several typical time series datasets.

1. Introduction

The recent advancements in sensor, computing and communication technologies are the leading rich sources in the provision of data for time series. These advancements transform the way real-world complex systems are monitored and controlled [1]. For instance, economists use stock price fluctuation series data to predict economic trends, medical scientists use biological time series data to predict diseases, and environmentalists use atmospheric time series data to predict environmental climate change. In addition, industrial manufacturing also uses time series data to monitor product quality. Particularly, the rich causal and dynamic information contained in time series enables it to be used to predict the evolution of complex biological and engineering system dynamics. Consequently, time series forecasting is emerging as one of the important branches of big data analysis, and its target is to accurately forecast future trends based on past and present time series data.

Time series are generated chronologically, and have characteristics of high dimensionalities and temporal dependencies. The high dimensionalities are that each time point is a dimension, and the temporal dependencies mean that even if two numerically identical points belong to different classes or predict different behaviors. According to the number of sampling variables at time point, time series can be divided into single variable time series and multi-variable time series. These complex characteristics make the accurate time series forecasting very difficult.

In the past, time series forecasting has become an important research area. Several different time series forecasting approaches have been proposed, such as the autoregressive approach [2,3], autoregressive integrated moving average approach [4,5], support vector machine approach [6,7], and neural networks based approaches [8–10]. Based on these basic approaches, several hybrid approaches [11–14] have also been proposed. However, since the time series data, such as financial data, wind data, seismic data, etc., usually are non-linear and non-stationary, and these existing approaches cannot effectively extract enough sequence data features to achieve accurate time series forecasting results.

Recently, deep learning [15,16] is a novel approach that can obtain a multi-levels representation of the original input by combining simple but non-linear modules. Each module of the deep neural network transforms one level of representation into a more abstract level. With these enough transformations, deep learning can learn fully features from input data. The deep learning has been used for Computer Vision [17–21], Speech Recognition [22–24], Natural Language Processing[25,26], etc.

Since traditional methods are usually hard to extract enough time series futures, their forecasting accuracies are limited, especially in non-liner and non-stationary datasets. Therefore, in order to improve the accuracy of time series forecasting. We propose a novel time series forecasting model based on deep learning, named SeriesNet, which can fully learn sequence data features in different interval lengths. Our main contributions include as follows:

* Corresponding author.
E-mail addresses: zym@zjut.edu.cn (Y. Zhang), viivan@zjut.edu.cn (J. Lu), xujun@zjut.edu.cn (J. Xu), xg@zjut.edu.cn (G. Xiao).

https://doi.org/10.1016/j.neucom.2018.12.084
0925-2312/

- We propose a novel time series forecasting model with deep learning and design a new network structure to fully extract series data features.
- We adopt residual learning and batch normalization in the SeriesNet to improve its forecasting accuracy and use a simplified activate gate that makes the SeriesNet can adapt to time series data.
- We evaluate the SeriesNet on several typical time series datasets, and show this model has higher forecasting accuracy and greater stableness than traditional methods.

In Section 2, we analyze the related work, and show the time series data preprocessing in Section 3. Then, we show the structure of SeriesNet model in Section 4, and in Section 5, we evaluate this model with several typical open time series datasets, and conclude this paper in Section 6.

2. Related work

The classical time series forecasting model is autoregressive (AR) proposed by British statistician U. Yule. The AR model is a representation of stochastic process, and its output variables are linearly dependent on their previous values and random conditions. Based on this classical model, other improved AR models, such as the moving average (MA) and the autoregressive moving average (ARMA) are proposed. However, the ARMA model is not suitable for non-stationary time series data. Due to this reason, the Autoregressive Integrated Moving Average model (ARIMA) [27] is proposed later. The one or more times differencing step in the ARIMA makes the time series data stationary. The differencing operation will generally amplifies high-frequency noise in time series data, and then it will affect the forecasting accuracy. Based on ARIMA, Peter and Silvia [28] proposes the Autoregressive Integrated Moving Average model with exogenous inputs model (ARIMAX) to process other time series.

Thissen et al. [29] and Gui et al. [30] show another new forecasting model based on support vector machine (SVM). Time series forecasting belongs to regression problem. The SVM is suitable for handling this problem, which is called support vector regression (SVR). It maps samples from original space to high-dimensional space, and then constructs a linear decision function to achieve linear regression. The SVR can achieve better experimental results than the previous ARIMA model when time series data is very complex. Least squares support vector machines (LS-SVM) changes the constraints of SVM objective function [31]. Wang and Hu [32] compares the SVM and the LS-SVM to solve the regression problem, and finds that the LS-SVM is more suitable for large scale problem, and the computational efficiency is also much higher.

The third forecasting model is based on neural network. This model has great advantages in solving nonlinear problems. According to the universal approximation theorem, the neural network can approximate any Borel measurable function from one finite dimensional space to another finite dimensional space with any precision [33]. Zhang and Berardi [8] shows a neural network ensemble model that combines different neural network structures. It can consistently outperform prediction of single neural network, and also can improve the prediction capability by avoiding overfitting problem. However, the performance of neural network depends on the sample size and noise level for linear problems. Zhang [11] proposes a hybrid model that combines the ARIMA model and neural network. This hybrid model can obtain better forecasting precision on both linear and non-linear data with better performance. In addition, Chow and Leung [34] proposes a nonlinear predictive neural network to predict short-term time series. This network is based on a feedback connection network with nonlinear autoregressive model.

Resent years, deep learning has been proposed for time series forecasting. Connor and Martin [35] gives recurrent neural network (RNN) that can use historical information of time series to predict future results. Later, an improved RNN named Long-Short Term Memory (LSTM) is proposed for time series forecasting [36]. Mittelman [37] proposes the undecimated fully convolutional neural network (UFCNN) to deal with time series problems. The UFCNN will not lead to gradient vanishing or explode, and is easier to be trained. Dasgupta and Osogami [38] proposes the Gaussian Dynamic Boltzmann Machine (Gaussian DyBM) by combining the RNN. The parameter updating method of Gaussian DyBM does not depend on the back propagation algorithm, and therefore the parameter updating is independent of data dimension or maximum delay. These characteristic makes the Gaussian DyBM more robust and more scalable. Moreover, its performance is also superior to that of the popular LSTM model. Naduvil-Vadukootu et al. [39] shows a pipelined model that combines some state-of-the-art time series data processing method with deep neural network (DNN). It can reduce dimensionality of input data. In this way, this model improves the DNNâs computational efficiency.

The other models of time series forecasting include state space neighborhood [40], Gaussian processes(GP) [41], functional decomposition model [42]. These models belong to nonparametric approaches, and provide a complete representation of the dynamics according to the observed data [1].

Compared with above methods, we do not use fixed time intervals to model time series. Instead, we use LSTM and dilated casual convolution to extract features with different time intervals from time series and finally combine these features. This can fully lean features from time series and it can help to improving the forecasting accuracy.

3. Time series data preprocessing

Generally, since time series data are generated by natural system, a variety of unexpected reasons will lead to data redundancy, missing, abnormal data and other issues in the process of data acquisition and transmission. Data preprocessing is necessary to make the series data suitable for SeriesNet. Fig. 1 shows the preprocessing procedure that includes several stages.

The stage of data clean is to deal with the redundancy, NULL and outliers existing in original time series data. Redundant data will make time series not be aligned with the time dimension, and will lead to large errors in the SeriesNet. On the other hand, it is also not suitable to simply discard missing values, since doing this will lost a large amount information and will not get complete times series data.

The stage of normalization is to scale all feature dimensions to the same scale, because the data to be forecasted often have numerical inconsistencies.

The stage of sliding window is to divide dataset according to sliding window length. The sliding window length indicates the length of a single input time step size, which also means that the model will forecast the value of next time based on historical information of window length. For example, there is a time series sequence {1,2,3,4,5,6,7}. When the sliding window size is 3 and sliding step size is 1, the output of time series sequences is {{1,2,3},{2,3,4},{3,4,5},{4,5,6},{5,6,7}}. In this stage, the original series data is constructed as the formal input according to the sliding window size.

The stage of splitting is to split the original data into three parts: the train data that accounts for 60%, the test data that accounts for 20%, and the validate data that accounts for 20% of total dataset.

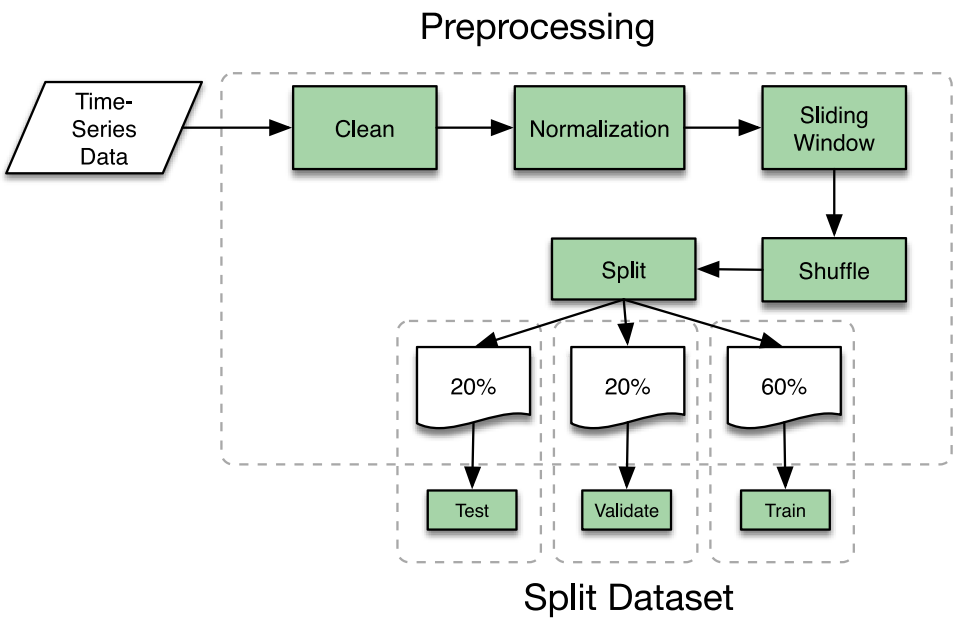

Fig. 1. Stages of a preprocessing.

4. Principal of SeriesNet

4.1. Long Short-Term Memory Network

Compared with traditional artificial neural network, Recurrent Neural Network can use historical information of time series data for special network structure. RNN is able to set output data according to current and past inputs. However, the RNN uses the BPTT algorithm to train the network, when it processes long time intervals in sequence, the gradient will vanish [43]. And the longer the intervals become, the more serious the gradient will become. This makes it hard to effectively train the RNN for long time intervals.

In order to handle this problem, other variants of recurrent neural network are proposed to capture long-term dependencies more easily, such as LSTM [44] and GRU [45] have a special cell structure, GF-GRU [46] use a special connection way between each two layers. The LSTM is a basic variant, it uses three gate structures: forget gate, input gate and output gate, to control the memory history information of the hidden layer. The equations in time t are as follows:

$$i_t = sigmoid(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_t) \quad (1)$$

$$f_t = sigmoid(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_t) \quad (2)$$

$$c_t = f_t c_{t-1} + i_t tanh(W_{xc}x_t + W_{hc}h_{t-1}) \quad (3)$$

$$o_t = sigmoid(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t) \quad (4)$$

$$h_t = o_t tanh(c_t) \quad (5)$$

Where i is the input gate; f is the forget gate; o is the output gate; c is the memory cell; h is the hidden layer output; W is the weight matrix, and the subscript is the weight association item. For example, W_{xi} represents the connection weight from input layer to input gate. The values of i, f, o within the range of [0,1] will control the ratio that the historical data pass through the gate.

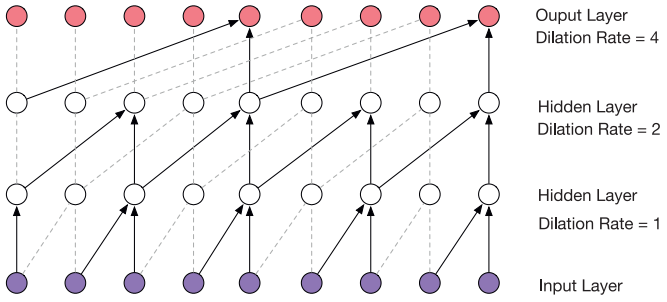

Fig. 2. Visualization of a stack of dilated causal convolution.

4.2. Dilated causal convolution

The dilated convolution is proposed to handle the loss of resolution or coverage due to the down-sampling operation in image semantic segmentation [47]. It uses dilated convolutions to systematically aggregate multi-scale contextual information and improves the accuracy of image recognition. The causal convolution is to ensure that the convolution kernel of convolution neural network can perform convolution operations exactly in time sequence [48], and that the convolution kernel can only read the current information and historical information.

Google DeepMind combines the dilated convolution and causal convolution in the WaveNet model, and proposes a dilated causal convolution to generate a more natural sound than the traditional method [48]. The WaveNet uses dilated causal convolution and skip-connection to utilize long-term information. Fig. 2 shows a stack of dilated causal convolution [48]. The dilation is doubled for every layer, such as $1, 2, 4, \ldots, 2^n$, and has a maximum value. After the maximum value, it begins to repeat. The maximum value in WaveNet is set 2^9. As a result, the receptive field increases with the increasing of the number of layers.

4.3. Structure of SeriesNet

This section describes the structure of SeriesNet. Compare with the traditional LSTM model that cannot extract hierarchical

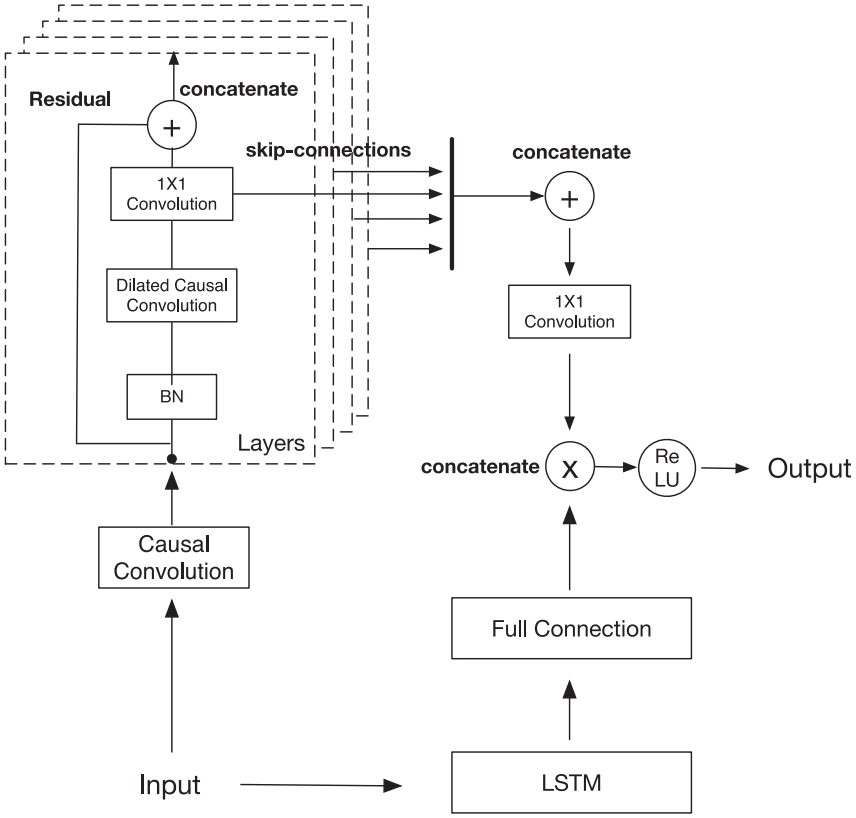

Fig. 3. The structure of the SeriesNet.

features from time series data, the SeriesNet can fully extract features from time data.

Assuming the input sequence is $X = (x^1, x^2, \ldots, x^{t-1}, x^t)$, the maximum likelihood $p(X)$ is as follows:

$$p(X) = \prod_{t=1}^{T} p(x^{t+1}|x^1, x^2, \ldots, x^t) \tag{6}$$

The x_{t+1} will take full advantages of all the time step information of the series.

Fig. 3 gives the structure of SeriesNet. we use the LSTM layer to learn the holistic features of input data, and then uses the full connection layer to set the output dimensionality of LSTM. Another part of the seriesNet, causal convolution is added to ensure time series conform to the causal relationship and then it connects to residual connection block. The residual connection block includes dilated causal convolution operation. Stacking the residual connection block and increasing the dilated rate of dilated causal convolution will make SeriesNet have larger receptive field size. Different dilated rates will get different receptive field sizes. Each residual connection block will extract time series features in different levels.

SeriesNet uses long skip-connection operation to merge the time series features extracted from each residual connection block. This can ensure time series features are not lost in the networks. Furthermore, not only the operation can improve the performance, but also can handle different levels of noise [49].

Finally, we concatenate the two parts of output, the advantages of SeriesNet are that its shallow layers of residual blocks will learn short interval features, its deep layers of residual blocks will learn long interval features and LSTM layer will learn holistic features. It means that SeriesNet can use abundant information of hierarchical features in regression prediction to improve prediction accuracy. In SeriesNet, we use addition operation and multiplication operation to connect different parts of the model. The addition operation is used to connect the output of different layers in the skip-connection. In residual connection block, it is a part of the identity function like in ResNet [50]. The multiplication operation is used to connect two types of network. It can mixture the features, which are extracted from dilated causal convolution network and LSTM network.

At Last, the RMSProp algorithm is used to train the parameters, the pseudo-code of SeriesNet is described in Algorithm 1 . We use the mean square error loss function as objective of SeriesNet:

$$loss_{min} = \frac{1}{T}\sum_{t=1}^{T}(Y^{t+1} - x^{t+1})^2 \tag{7}$$

4.4. Multi-conditional time series

Multi-conditional time series are additional data that may have great impact on target time series to be predicted. The multi-conditional series data can help to improve forecasting accuracy.

For multi-conditional time series, the LSTM and full connection layers can reduce the dimensionality of multi-conditional data. Additional input variables will be introduced. Then, the Eq. (8) will be changed as follows:

$$p(X|y) = \prod_{t=1}^{T} p(x^{t+1}|x^1, x^2, \ldots, x^t, y_1^1, y_1^2, \ldots, y_i^t) \tag{8}$$

where y_i is the input variable of additional condition i. The LSTM network maps the multi-condition time series data to a lower-dimensional time series, which will learn the internal relation among multi-condition time series data.

Moreover, residual learning ensures the accuracy does not decrease at very deep network and uses batch normalization (BN) [51] to avoid gradient vanish.

Algorithm 1: SeriesNet.

Require: Train dataSet $\mathcal{D}_{train} = \{x_i\}_{i=1}^{n}$, test dataset $\mathcal{D}_{test} = \{x_i\}_{i=1}^{k}$
Require: Time step s, model parameter θ, learning rate η, small constant δ, decay rate ρ and batch size m
Require: Forecasting sliding window j
Input: $\mathcal{D}_{in} = \{\{x_i\}_{i=1}^{s}, \{x_i\}_{i=2}^{s+1}, \ldots, \{x_i\}_{i=n-s+1}^{n}\}$
Label: $\mathcal{D}_{label} = \{\{x_i\}_{i=2}^{s}, \{x_i\}_{i=3}^{s+1}, \ldots, \{x_i\}_{i=n-s}^{n-1}\}$
Output: Forecasting results R_{out}

Initialize the SeriesNet parameters θ ;
Initialize gradient accumulation variable $r = 0$;
Sample a mini batch of m examples from the input ;
while *stopping criterion is not satisfied* **do**
  Compute gradient (mini batch): $g \leftarrow \frac{1}{m}\nabla_\theta \sum loss$;
  Accumulate gradient: $r \leftarrow \rho r + (1-\rho) g \odot g$;
  New parameters: $\theta' = \theta - \frac{\eta}{\sqrt{\delta + r}} \odot g$;
  Update parameters: $\theta \leftarrow \theta'$;
end
return θ
Compute forecasting value in $\mathcal{D}_{test} = \{x_1, x_2, \ldots, x_k\}$:
for $i \leftarrow 1$ **to** j **do**
  Compute next step forecasting value: v_i ;
  Append v_i to the end of $\{x_{i+1}, x_{i+2}, \ldots, x_k\}$;
end
Output the forecasting result: $R_{out} = \{v_1, v_2, \ldots, v_j\}$

4.5. Deep residual

Very deep network will lead to gradient vanishing, and then this will decrease the forecasting accuracy. To deal with this key problem, residual learning [52] is proposed.

Suppose the original hidden layer mapping is $\mathcal{H}(x)$, and let each residual block learns the residual mapping between input and output:

$$\mathcal{F}(x) = \mathcal{H}(x) - x$$

That is, the output of the original hidden layer is represented as:

$$\mathcal{H}(x) = \mathcal{F}(x) + x$$

This main modification lets the network only require to learn the residual mapping of each layer instead of learning the mapping of input and output. When the mapping between $\mathcal{H}(x)$ and $\mathcal{F}(x)$ is identity, the network can detect small perturbations. Through stacking the residual blocks, the deep residual network ensures that the forecasting accuracy will not be decreased. When the model learns high-level features, the residual learning makes it possible to train very deep network.

Compared with audio signal data, time series data will be much simpler. The gate activation unit in WaveNet is suitable for complex interactions [53], such as audio signals. It splits feature maps which are output of every causal convolution operation, then use a tanh and a sigmoid activate function to process previous output. Its non-linearity works much better than the rectified linear activation function [48]. However, compared with the audio signal data of sound, time series data is much simpler. That is to say, the rectified linear activation is adequacy for time series data. Then, the activation of dilated causal convolution is changed to:

$$z = ReLU(W_k * x)$$

Where k is the sequence of the residual blocks; W_k is the weight of the convolution filter in the k-th residual block.

4.6. Batch normalization

In order to learn higher level abstract representations of input data, the depth of deep network can be added. However, this will make the distribution of parameters of each layer change when training neural network.

Since the weight distribution of each layer is different, the unified learning rate has different effect on the weight update of each layer. It is difficult to coordinate weight update among different layers. Due to this reason, it is difficult to select a suitable learning rate to train the network. As a result, the network will be gradient vanish, called internal covariate shift [51]. As deepening the network, the negative effects become more serious and lead to ineffective training.

SeriesNet adds the batch normalization (BN) layer after the input of each layer in residual connection block and redistributes the input. It will normalize each scalar feature of input data independently to make the value of mean be 0 and the value of variance be 1. The parameters of the BN are also can be learned during the training, which makes the model easily recover to origin input.

5. Evaluation

We use three typical open time series datasets: the S&P 500 Index, the Shanghai Composite Index and the temperature of Hangzhou to evaluate SeriesNet. The S&P 500 Index and Shanghai Composite Index data are from Yahoo Finance and the weather data is from China Meteorological Administration. Table 1 shows the profiling information of these three datasets. It shows that the time series name, the time range, the train dataset number, validation dataset number and the test dataset number. The economic data is non-stationary and non-linear, the temperature data is relatively stationary.

Table 2 shows the best hyper-parameters of SeriesNet in S&P 500 dataset. In this table, the column of Type means the operation of each layer. We use matrix addition to connect two layers. If the addition is marked with Skip-Connection, it uses skip-connection operation to connect all of the output layers. The column of Units is the number of neurons. When their types are Convolutional, the values of the column are the number of filters. The column of Size is the kernel size of filters. SeriesNet uses different dilation rates in different layers when we stuck the residual connection blocks. In addition, the dilation rate of normal convolutional operation is set to 1. The column of Padding is the way of padding. The column of Output is the output dimensions of each layers. Worthy of note is that the hyper-parameters of the model and the number of residual blocks are slightly adjusted when SeriesNet applies to different datasets.

We compare SeriesNet with the LSTM, the UFCNN, the ANN and the SVM, The LSTM is the most popular deep learning model, and the UFCNN is an advanced convolution neural network model, the

Table 1
Time series dataset.

Time series	Time range	Train data	Validation data	Test data
S&P 500 Index	2001.01–2017.05	2750	878	878
Shanghai Composite Index	2005.01–2017.06	1530	510	510
The temperature of Hangzhou	2011.01–2017.01	1880	470	470

Table 2
Hyper-parameters of SeriesNet.

Type	Units/Filters	Size	Dilation rate	Padding	Output
Convolutional(Input)	8	30	1	causal	90×8
Batch normalization					
Convolutional	8	7	1	causal	90×8
Convolutional	1	1	1	same	90×1
Add					90×8
Batch normalization					
Convolutional	8	7	2	causal	90×8
Convolutional	1	1	1	same	90×1
Add					90×8
Batch normalization					
Convolutional	8	7	4	causal	90×8
Convolutional	1	1	1	same	90×1
Add					90×8
Batch normalization					
Convolutional	8	7	8	causal	90×8
Convolutional	1	1	1	same	90×1
Add					90×8
Batch normalization					
Convolutional	8	7	16	causal	90×8
Convolutional	1	1	1	same	90×1
Add(Skip-connection)					90×1
Convolutional	1	1	1	same	90×1
LSTM(Input)	10				90×10
LSTM	10				90×10
FC	1				90×1
Multiply					90×1

Table 3
The result of the forecasting sliding window is 1.

Time series model	S&P 500 Index			Shanghai Composite Index			The temperature of Hangzhou		
	RMSE	MAE	R^2	RMSE	MAE	R^2	RMSE	MAE	R^2
$SeriesNet_{10}$	**17.32**	**13.15**	**0.982**	63.94	38.37	0.976	2.82	2.06	0.903
$SeriesNet_{50}$	19.97	15.56	0.976	**63.59**	38.41	**0.977**	2.92	2.16	0.896
$SeriesNet_{100}$	33.54	26.72	0.933	64.54	39.37	0.976	2.83	2.09	0.903
$SeriesNet_{NoRes}$	18.55	13.94	0.979	80.60	50.56	0.962	2.91	2.13	0.897
$LSTM^2_{20}$	19.04	14.42	0.978	63.84	**38.05**	0.976	2.86	2.09	0.901
$LSTM^2_{50}$	20.79	15.57	0.974	64.72	40.55	0.975	2.87	2.07	0.900
$LSTM^2_{100}$	18.59	14.02	0.979	64.85	41.21	0.975	2.89	2.09	0.899
$LSTM^4_{100}$	26.31	20.13	0.959	66.18	39.66	0.975	2.87	2.10	0.901
UFCNN	24.36	19.84	0.965	93.06	57.77	0.950	**2.64**	**1.97**	0.907
ANN	24.22	20.21	0.965	66.25	39.35	0.975	2.95	2.14	0.895
SVM	31.55	22.92	0.941	69.02	40.43	0.972	2.74	2.04	**0.909**

Table 4
The result of the forecasting sliding window is 5.

Time Series Model	S&P 500 Index			Shanghai Composite Index			The temperature of Hangzhou		
	RMSE	MAE	R^2	RMSE	MAE	R^2	RMSE	MAE	R^2
$SeriesNet_{10}$	**31.47**	**23.76**	**0.941**	121.53	71.18	0.915	3.95	3.02	0.811
$SeriesNet_{50}$	47.34	35.60	0.866	**116.59**	**67.43**	**0.921**	4.19	3.20	0.788
$SeriesNet_{100}$	51.40	40.72	0.843	121.32	72.24	0.915	4.04	3.14	0.802
$SeriesNet_{NoRes}$	36.94	27.22	0.918	144.55	88.67	0.879	4.10	3.13	0.796
$LSTM^2_{20}$	34.98	25.85	0.927	122.25	69.81	0.914	4.18	3.25	0.788
$LSTM^2_{50}$	40.36	30.11	0.903	121.65	77.13	0.915	4.07	3.16	0.799
$LSTM^2_{100}$	38.55	28.81	0.912	123.64	81.06	0.912	4.11	3.18	0.796
$LSTM^4_{100}$	61.19	43.46	0.778	130.87	78.35	0.901	4.08	3.18	0.799
UFCNN	36.26	29.13	0.922	194.81	126.23	0.781	4.80	3.85	0.692
ANN	67.05	55.68	0.733	129.98	76.43	0.902	4.84	3.69	0.716
SVM	50.34	36.14	0.849	138.64	78.16	0.889	**3.73**	**2.88**	**0.832**

ANN is the most basic neural network model, and the SVM is the most classic time series forecasting model.

This paper uses the root-mean-square error (RMSE), the mean absolute error (MAE) and the coefficient of determination (R^2) as evaluation criteria, the formula of R^2, RMSE and MAE are as follows:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(f_i - y_i)^2}{\sum_{i=1}^{n}(f_i - \bar{y})}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(f_i - y_i)^2}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|f_i - y_i|$$

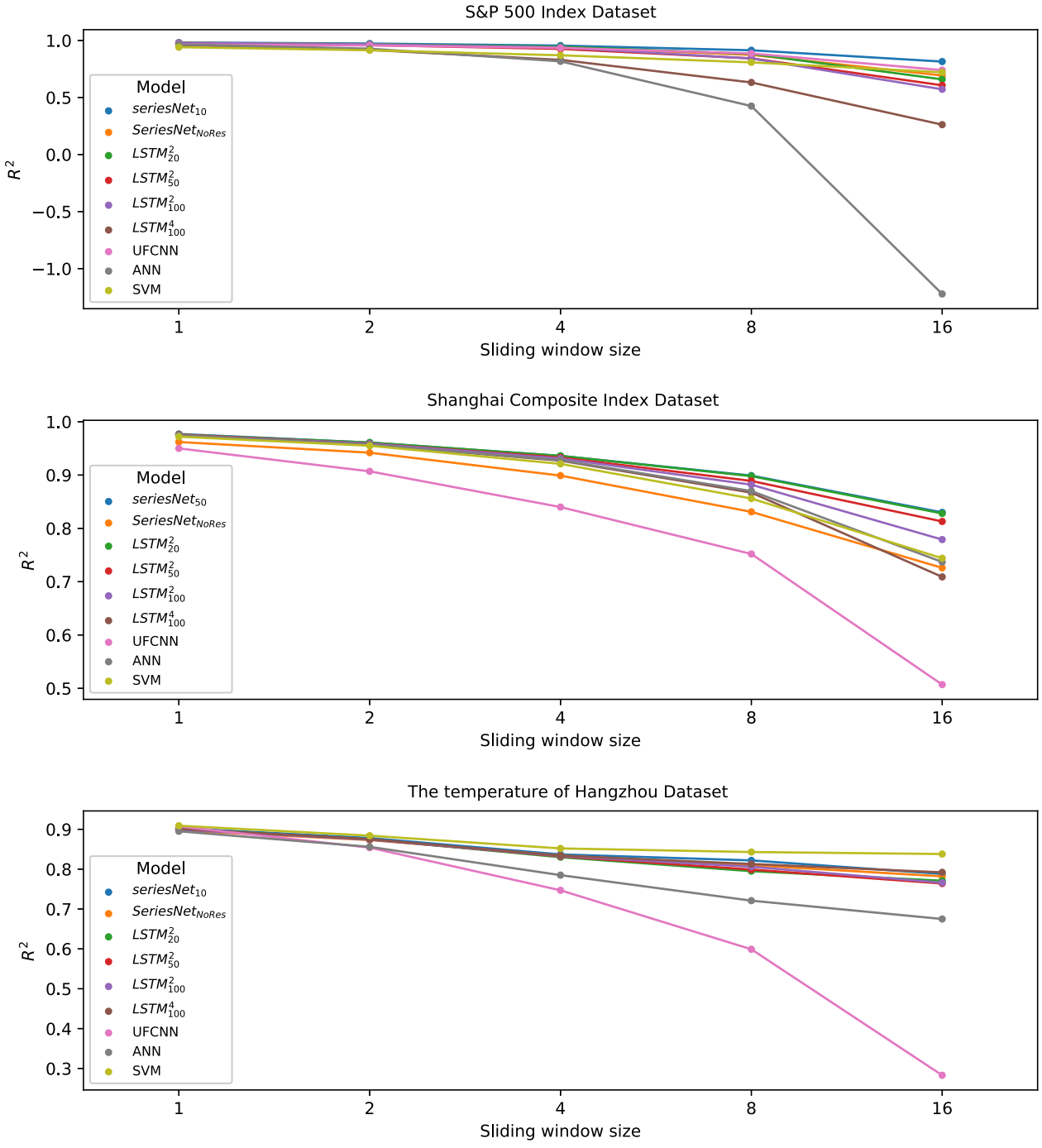

Fig. 4. Comparison of different sliding window size on three datasets.

The closer to 1 the R^2 is, the higher the forecasting accuracy is. And the smaller the RMSE and the MAE are, the smaller the error are.

Table 3 shows the experimental results with three datasets, when forecast sliding window representing future time span is 1.

We use four different super parameters of LSTM as baseline to compare with the SeriesNet, the UFCNN, the ANN and the SVM on three datasets. $LSTM^2_{20}$ has 2 layers of LSTM cell and each layer contains 20 neurons, $LSTM^2_{50}$ has 2 layers of LSTM cell and each layer contains 50 neurons, and so on. In addition, we also use three different SeriesNet models that have different LSTM sizes. Moreover, a SeriesNet without residual model is also used to verify the effectiveness of residual learning.

It can be seen from Table 3 that the SeriesNet models have the best forecasting accuracy in non-linear and non-stationary datasets compared with other models. For the SeriesNet models, different datasets have different accuracies. For instance, SeriesNet_{10} is the best in S&P 500 Index dataset and SeriesNet_{50} is the best in Shanghai Composite Index dataset. The reason of this phenomenon is that the series characteristics of these data sets are quite different. Complex data fluctuations require a larger number of neurons or complex model to fit them. On the contrary, simple data fluctuations adopting complex structure will lead to low generalization ability. That is, there exists overfitting in the model. As a result, hyper-parameters of SeriesNet require to be carefully adjusted. In addition, the result shows the SeriesNet is not the best for

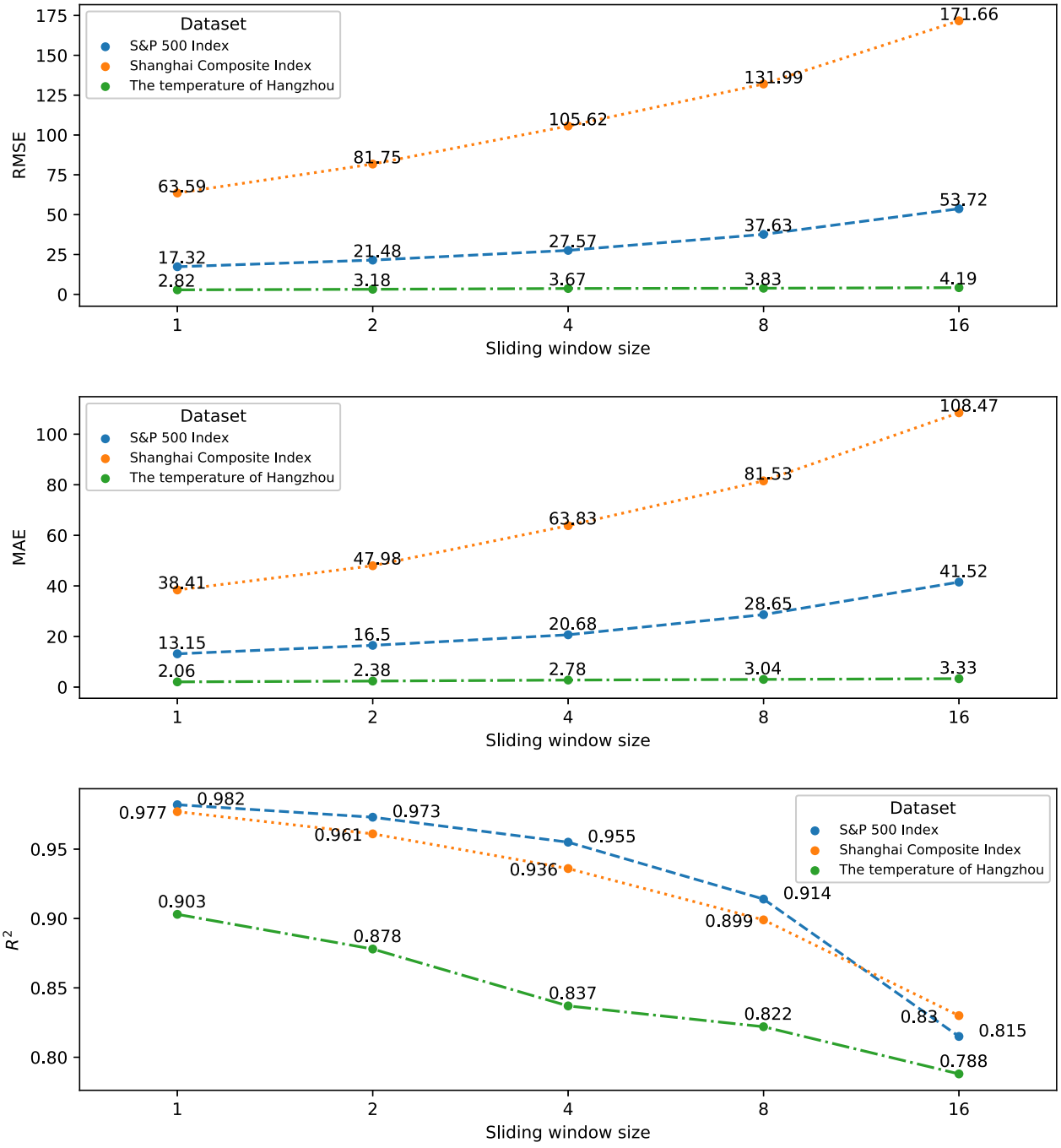

Fig. 5. Comparison of different sliding window size.

stationary time series data such as temperature data, because excessive parameters lead to overfitting easily, while the SVM model adopts the principle of minimizing structural risk to find the optimal solution. It makes the SVM to fit strong regularity time series data well.

Taking the best results of SeriesNet, SeriesNet without residual, LSTM, UFCNN, ANN and SVM in three datasets. The average R^2 value of these six models are 0.954, 0.946, 0.952, 0.941, 0.945 and 0.941 respectively for the three datasets. It can be seen that SeriesNet is generally superior than other models.

Table 4 shows the RMSE, the MAE and R^2 values with the three datasets, when the sliding window size is 5. The average R^2 value of six modes are 0.891, 0.864, 0.880, 0.798, 0.784 and 0.857 respectively for the three datasets.

Generally, large sliding window size will result in low forecasting accuracy. The next forecasting result will be carried out on the basis of previous forecasting result, since increasing the sliding window size will introduce more cumulative error. Fig. 4 shows the decline of R^2 as the sliding window size increases on the three datasets and Fig. 5 shows three evaluation methods in SeriesNet model. These figures show the accuracy rapidly declines along with the increase of sliding window size. Although it is difficult to accurately forecast long time-span time series, it is more valuable to get future trends of time series. The error of SeriesNet slowly declines with the increase of sliding window size compared with other models in the non-stationary datasets.

In order to further demonstrate the experimental results, we use the forecasting and error figures to illustrate. Fig. 6 compares

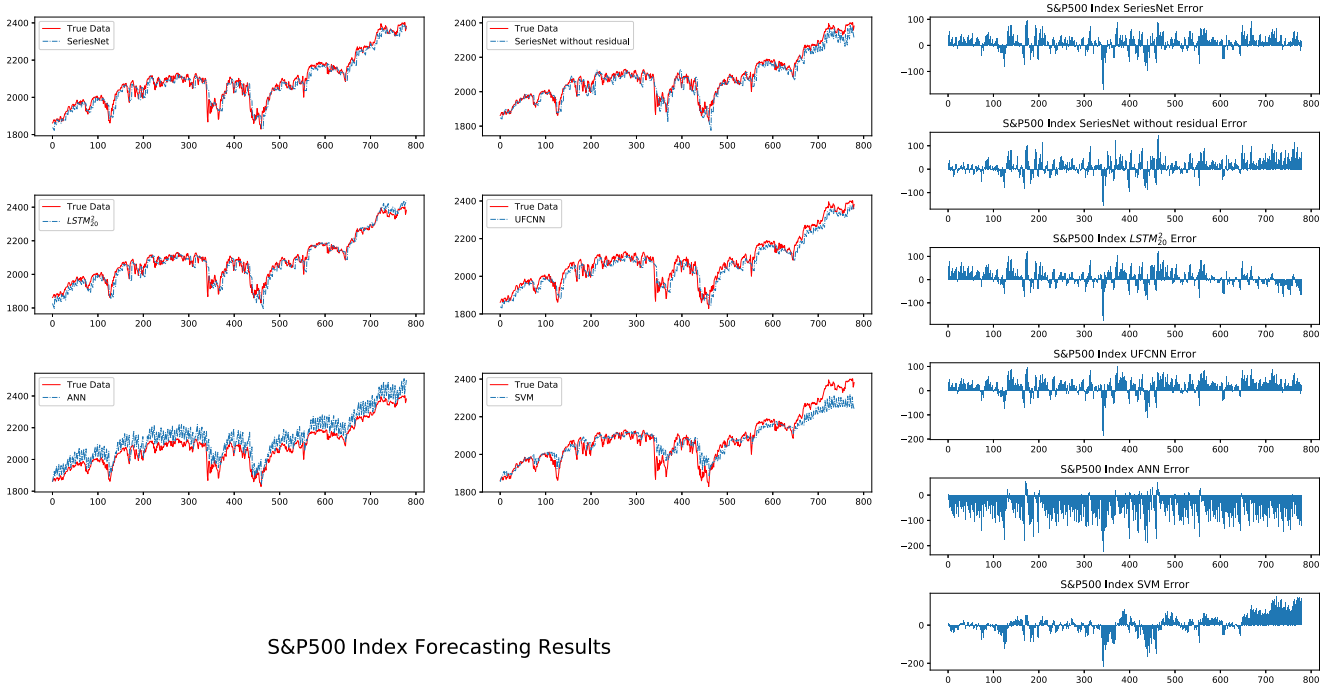

Fig. 6. Comparison of forecasting values and true values for S&P 500 Index.

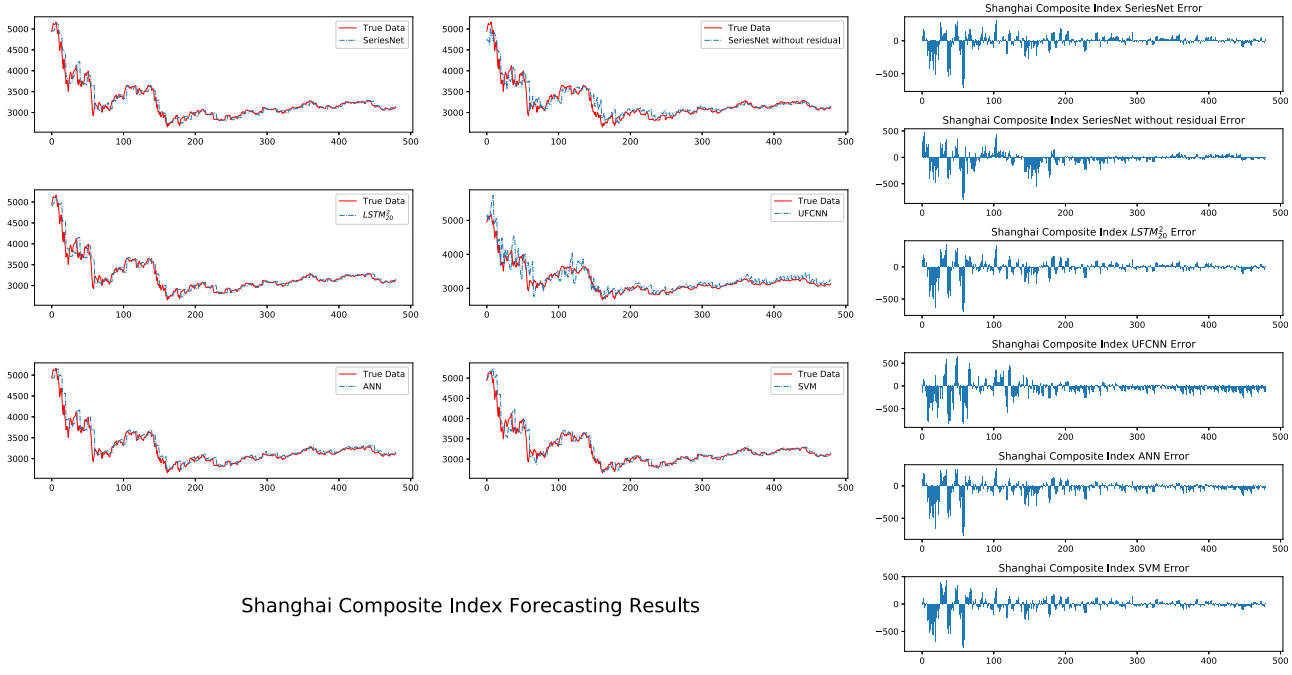

Fig. 7. Comparison of forecasting values and true values for Shanghai Composite Index.

the forecasting values from SeriesNet, the SeriesNet without residual, the LSTM, the UFCNN, the ANN, and the SVM with the true value in dataset of S&P 500 Index. In the left figure, the *x*-axis indicates the time unit and the *y*-axis is the correlation value. This figure has seven cures of forecasting and true values. The error figure is obtained by calculating the true values and the forecasting value in the left figure. And in the error figure, the *x*-axis indicates the time unit, and the *y*-axis indicates the error between forecasting value and true value. According to above forecasting and error figure, it can be seen that the error of SeriesNet model is lower and the error is relatively stable.

In addition, Fig. 7 compares the forecasting values from the SeriesNet, the SeriesNet without residual, the LSTM, the UFCNN, the ANN, and the SVM with the true value in dataset of Shanghai Composite Index. Fig. 8 compares the forecasting values from the SeriesNet, the SeriesNet without residual, the LSTM, the UFCNN, the ANN, and the SVM with the true value in dataset of daily temperature of Hangzhou.

Furthermore, we divide each dataset into 5 parts, and calculate the forecasting error of each part.Tables 5–7 show the forecasting errors of each part when the sliding window size is set to 5. Combined with Figs. 6–8 and Tables 5–7, it can be seen that the

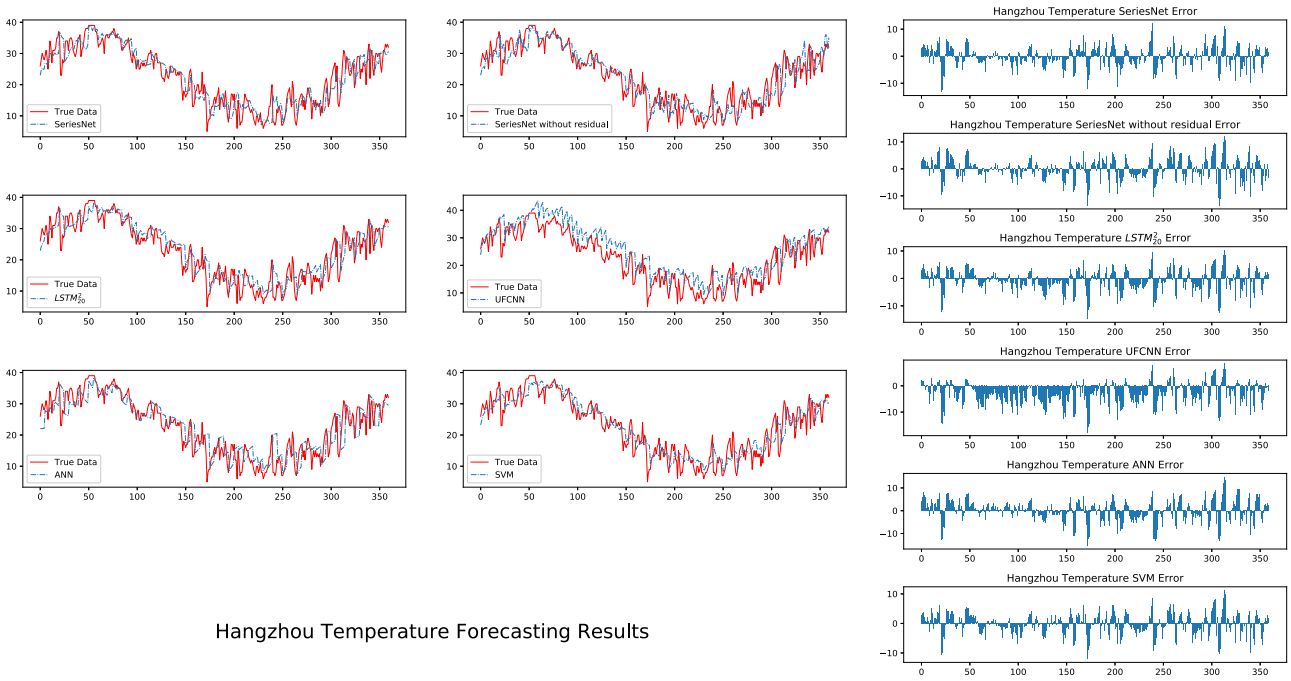

Fig. 8. Comparison of forecasting values and true values for the temperature of Hangzhou.

Table 5
The evaluation result of S&P 500 Index on different start time.

Time series model	1 RMSE MAE R^2	2 RMSE MAE R^2	3 RMSE MAE R^2	4 RMSE MAE R^2	5 RMSE MAE R^2
$SeriesNet_{10}$	27.20 21.27 0.721	30.27 22.69 0.197	**41.27 30.77 0.738**	**27.95** 22.22 **0.793**	28.52 21.88 0.899
$SeriesNet_{50}$	36.74 29.41 0.491	46.34 31.96 −0.882	46.48 35.84 0.668	47.18 37.01 0.411	57.67 43.83 0.586
$SeriesNet_{100}$	54.25 48.12 −0.110	49.62 39.63 −1.157	69.36 51.65 0.260	42.06 35.17 0.532	34.99 29.04 0.848
$SeriesNet_{NoRes}$	**24.00 16.80 0.783**	33.67 24.81 0.007	48.66 37.19 0.636	28.62 **22.00** 0.783	43.89 35.31 0.760
$LSTM^2_{20}$	35.34 28.49 0.529	32.36 23.78 0.082	46.40 35.13 0.669	31.15 22.07 0.743	26.44 19.77 0.913
$LSTM^2_{50}$	41.96 34.49 0.336	37.72 28.32 −0.247	56.12 41.16 0.515	28.32 21.00 0.788	31.75 25.60 0.875
$LSTM^2_{100}$	38.54 30.58 0.440	41.13 32.31 −0.483	51.69 39.22 0.589	30.69 22.98 0.751	**25.32 18.97 0.920**
$LSTM^4_{100}$	42.92 34.56 0.305	28.85 22.40 0.271	47.90 34.90 0.647	36.06 29.57 0.656	111.59 95.87 −0.550
UFCNN	30.73 24.89 0.644	31.94 26.05 0.106	46.57 36.58 0.666	33.99 27.86 0.694	35.87 30.25 0.840
ANN	56.14 47.03 −0.189	70.76 59.83 −3.388	75.70 60.66 0.118	63.68 53.38 −0.073	67.34 57.49 0.435
SVM	28.56 18.96 0.692	**28.74 22.02 0.276**	62.83 45.75 0.393	30.15 25.41 0.759	78.55 68.55 0.232

Table 6
The evaluation result of Shanghai Composite Index on different start time.

Time series model	1 RMSE MAE R^2	2 RMSE MAE R^2	3 RMSE MAE R^2	4 RMSE MAE R^2	5 RMSE MAE R^2
$SeriesNet_{10}$	233.79 169.81 0.852	112.51 84.35 0.897	**60.81** 45.37 **0.367**	37.65 27.99 0.747	37.55 28.37 0.687
$SeriesNet_{50}$	**224.48 162.38 0.864**	104.82 76.56 0.910	63.77 45.81 0.304	38.52 28.85 0.736	32.19 23.58 0.770
$SeriesNet_{100}$	234.98 174.15 0.850	**103.45 74.89 0.913**	61.23 45.80 0.358	50.74 39.11 0.541	36.80 27.25 0.699
$SeriesNet_{NoRes}$	253.00 194.56 0.827	176.25 129.11 0.746	81.90 61.02 −0.148	37.72 28.70 0.747	35.66 29.89 0.718
$LSTM^2_{20}$	237.66 172.44 0.847	110.42 82.36 0.900	61.29 **44.29** 0.357	**35.44 26.68 0.776**	**32.12 23.26 0.771**
$LSTM^2_{50}$	228.75 175.60 0.858	107.71 79.60 0.905	71.91 54.28 0.115	53.62 41.42 0.488	44.92 34.75 0.552
$LSTM^2_{100}$	228.13 177.19 0.859	109.76 80.86 0.902	78.60 60.85 −0.057	60.30 47.06 0.352	50.35 39.35 0.437
$LSTM^4_{100}$	247.87 176.15 0.834	122.23 91.87 0.878	66.32 47.65 0.247	45.99 34.64 0.623	52.36 41.46 0.391
UFCNN	360.59 271.43 0.648	189.92 146.98 0.705	89.63 69.84 −0.375	76.43 59.59 -0.040	98.87 83.29 −1.172
ANN	248.86 176.94 0.832	118.91 88.51 0.884	65.17 46.78 0.273	43.28 32.64 0.666	47.82 37.28 0.492
SVM	276.38 201.61 0.793	114.46 86.73 0.893	62.58 47.98 0.330	38.82 28.62 0.732	34.61 25.87 0.734

Table 7
The evaluation result of the temperature of Hangzhou on different start time.

Time series model	1			2			3			4			5		
	RMSE	MAE	R^2	RMSE	MAE	R^2	RMSE	MAE	R^2	RMSE	MAE	R^2	RMSE	MAE	R^2
$SeriesNet_{10}$	3.95	3.02	0.000	2.39	1.82	0.720	4.50	3.69	−0.048	3.78	**2.76**	0.013	4.70	3.81	0.351
$SeriesNet_{50}$	4.00	3.26	−0.025	**2.08**	**1.59**	**0.788**	4.72	3.59	−0.153	3.98	3.07	−0.095	5.42	4.46	0.137
$SeriesNet_{100}$	4.00	3.21	−0.027	2.51	1.91	0.691	4.40	3.54	−0.004	3.88	3.05	−0.040	4.99	4.01	0.269
$SeriesNet_{NoRes}$	3.73	2.88	0.107	2.18	1.62	0.768	4.40	3.44	−0.004	3.96	3.22	−0.083	5.51	4.49	0.107
$LSTM^2_{20}$	3.67	2.87	0.138	2.83	2.24	0.606	4.87	3.77	−0.228	4.11	3.37	−0.172	5.04	3.98	0.255
$LSTM^2_{50}$	3.65	2.86	0.145	2.67	2.05	0.649	4.74	3.67	−0.165	4.35	3.67	−0.312	4.58	3.56	**0.385**
$LSTM^2_{100}$	3.51	**2.67**	0.210	2.54	1.95	0.684	4.60	3.57	−0.098	4.34	3.66	−0.307	5.05	4.04	0.252
$LSTM^4_{100}$	3.60	2.85	0.172	2.32	1.82	0.735	4.64	3.57	−0.113	4.34	3.67	−0.303	4.94	4.00	0.284
UFCNN	3.51	2.88	−0.708	4.71	4.16	−1.611	6.79	5.32	−1.087	4.50	3.76	−0.931	**3.83**	**3.13**	0.170
ANN	4.49	3.50	−0.289	2.33	1.84	0.734	5.38	4.21	−0.501	4.82	3.77	−0.605	6.28	5.11	−0.160
SVM	**3.41**	2.69	**0.256**	2.45	1.87	0.707	**4.03**	**3.21**	**0.158**	**3.69**	2.97	**0.057**	4.68	3.68	0.356

SeriesNet is more adaptable than other models and can achieve better forecasting accuracies as the dataset fluctuations are dramatic. As the data changes are not obvious, other models also have chances to achieve better forecasting accuracies than our SeriesNet.

6. Conclusion

In order to improve the forecasting accuracy of time series, this paper designs a new time series forecasting model named SeriesNet. This model can fully learn features of time series data in different interval lengths. It uses the LSTM to learn holistic features and to reduce dimensionality of multi-conditional data. In addition, it uses the dilated causal convolutions to learn different time interval features. Furthermore, it adopts the residual learning and the batch normalization to improve generalization. This model can learn multi-range and multi-level features of time series data. Experimental results show this model has much higher forecasting accuracy and much greater stableness.

Conflict of interest

None.

Acknowledgments

The authors gratefully acknowledge the support by the Zhejiang special science and technology special project (No. 2018C01064); Public Projects of Zhejiang Province (No. 2017C31014).

References

[1] C. Cheng, A. Sa-Ngasoongsong, O. Beyca, T. Le, H. Yang, Z. Kong, S.T. Bukkapatnam, Time series forecasting for nonlinear and non-stationary processes: a review and comparative study, IIE Trans. 47 (10) (2015) 1053–1071.
[2] H. Akaike, Fitting autoregressive models for prediction, Ann. Inst. Stat. Math. 21 (1) (1969) 243–247.
[3] C.M. Hurvich, C.-L. Tsai, Regression and time series model selection in small samples, Biometrika 76 (2) (1989) 297–307.
[4] G.E. Box, D.A. Pierce, Distribution of residual autocorrelations in autoregressive-integrated moving average time series models, J. Am. Stat. Assoc. 65 (332) (1970) 1509–1526.
[5] B. Williams, P. Durvasula, D. Brown, Urban freeway traffic flow prediction: application of seasonal autoregressive integrated moving average and exponential smoothing models, Transp. Res. Rec. J. Transp. Res. Board 1644 (1998) 132–141.
[6] L.-J. Cao, F.E.H. Tay, Support vector machine with adaptive parameters in financial time series forecasting, IEEE Trans. Neural Netw. 14 (6) (2003) 1506–1518.
[7] K.-R. Müller, A.J. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, V. Vapnik, Predicting time series with support vector machines, in: Proceedings of the International Conference on Artificial Neural Networks, Springer, 1997, pp. 999–1004.
[8] G.P. Zhang, V. Berardi, Time series forecasting with neural network ensembles: an application for exchange rate prediction, J. Oper. Res. Soc. 52 (6) (2001) 652–664.
[9] M.M. Noel, B.J. Pandian, Control of a nonlinear liquid level system using a new artificial neural network based reinforcement learning approach, Appl. Soft Comput. 23 (2014) 444–451.
[10] Y. Chen, B. Yang, J. Dong, Time-series prediction using a local linear wavelet neural network, Neurocomputing 69 (4-6) (2006) 449–465.
[11] G.P. Zhang, Time series forecasting using a hybrid arima and neural network model, Neurocomputing 50 (2003) 159–175.
[12] A. Jain, A.M. Kumar, Hybrid neural network models for hydrologic time series forecasting, Appl. Soft Comput. 7 (2) (2007) 585–592.
[13] C.H. Aladag, E. Egrioglu, C. Kadilar, Forecasting nonlinear time series with a hybrid methodology, Appl. Math. Lett. 22 (9) (2009) 1467–1470.
[14] L.P. Maguire, B. Roche, T.M. McGinnity, L. McDaid, Predicting a chaotic time series using a fuzzy neural network, Inf. Sci. 112 (1-4) (1998) 125–136.
[15] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436.
[16] J. Schmidhuber, Deep learning in neural networks: an overview, Neural Netw. 61 (2015) 85–117.
[17] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Proceedings of the Advances in Neural Information Procezssing Systems, 2012, pp. 1097–1105.
[18] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1–9.
[19] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 580–587.
[20] X. Lu, B. Wang, X. Zheng, X. Li, Exploring models and data for remote sensing image caption generation, IEEE Trans. Geosci. Remote Sens. 56 (4) (2018) 2183–2195.
[21] X. Lu, X. Zheng, Y. Yuan, Remote sensing scene classification by unsupervised representation learning, IEEE Trans. Geosci. Remote Sens. 55 (9) (2017) 5148–5157.
[22] A.-r. Mohamed, G.E. Dahl, G. Hinton, Acoustic modeling using deep belief networks, IEEE Trans. Audio, Speech, Lang. Process. 20 (1) (2012) 14–22.
[23] G.E. Dahl, D. Yu, L. Deng, A. Acero, Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition, IEEE Trans. Audio, Speech, Lang. Process. 20 (1) (2012) 30–42.
[24] I. Sutskever, O. Vinyals, Q.V. Le, Sequence to sequence learning with neural networks, in: Proceedings of the Advances in Neural Information Processing Systems, 2014, pp. 3104–3112.
[25] Y. Bengio, R. Ducharme, P. Vincent, C. Jauvin, A neural probabilistic language model, J. Mach. Learn. Res. 3 (2003) 1137–1155. Feb
[26] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in: Proceedings of the Advances in Neural Information Processing Systems, 2013, pp. 3111–3119.
[27] G.E.P. Box, G.M. Jenkins, Time series analysis: forecasting and control, J. Am. Stat. Assoc. 68 (342) (1970) 199–201.
[28] Ď. Peter, P. Silvia, Arima vs. Arimax–which approach is better to analyze and forecast macroeconomic time series, in: Proceedings of the 30th International Conference Mathematical Methods in Economics, 2012, pp. 136–140. Karviná, Czech Republic
[29] U. Thissen, R. Van Brakel, A. De Weijer, W. Melssen, L. Buydens, Using support vector machines for time series prediction, Chemom. Intell. Lab. Syst. 69 (1) (2003) 35–49.
[30] B. Gui, X. Wei, Q. Shen, J. Qi, L. Guo, Financial time series forecasting using support vector machine, in: Proceedings of the Tenth International Conference on Computational Intelligence and Security (CIS), IEEE, 2014, pp. 39–43.
[31] J.A. Suykens, T. Van Gestel, J. De Brabanter, Least Squares Support Vector Machines, World Scientific, 2002.
[32] H. Wang, D. Hu, Comparison of svm and ls-svm for regression, in: Proceedings of the International Conference on Neural Networks and Brain, ICNN&B'05., 1, IEEE, 2005, pp. 279–283.
[33] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural Netw. 2 (5) (1989) 359–366.
[34] T. Chow, C.-T. Leung, Nonlinear autoregressive integrated neural network model for short-term load forecasting, IEE Proc. Gener. Transm. Distrib. 143 (5) (1996) 500–506.
[35] J.T. Connor, R.D. Martin, L.E. Atlas, Recurrent neural networks and robust time series prediction, IEEE Trans. Neural Netw. 5 (2) (1994) 240–254.

[36] F.A. Gers, D. Eck, J. Schmidhuber, Applying LSTM to time series predictable through time-window approaches, in: Proceedings of the Neural Nets WIRN Vietri-01, Springer, 2002, pp. 193–200.
[37] R. Mittelman, Time-series modeling with undecimated fully convolutional neural networks, arXiv:1508.00317 (2015) 1–9.
[38] S. Dasgupta, T. Osogami, Nonlinear dynamic Boltzmann machines for time-series prediction, in: Proceedings of the AAAI, 2017, pp. 1833–1839.
[39] S. Naduvil-Vadukootu, R.A. Angryk, P. Riley, Evaluating preprocessing strategies for time series prediction using deep learning architectures., in: Proceedings of the FLAIRS Conference, 2017, pp. 520–525.
[40] N. Hamid, M. Noorani, Modeling of prediction system: an application of nearest neighbor approach to chaotic data, Appl. Math. Comput. Intell. 2 (1) (2013) 137–148.
[41] C.E. Rasmussen, Gaussian processes in machine learning, in: Advanced lectures on machine learning, Springer, 2004, pp. 63–71.
[42] M.G. Frei, I. Osorio, Intrinsic time-scale decomposition: time–frequency–energy analysis and real-time filtering of non-stationary signals, Proc. R. Soc. Lond. A: Math. Phys. Eng. Sci. 463 (2007) 321–342. The Royal Society
[43] Y. Bengio, P. Simard, P. Frasconi, Learning long-term dependencies with gradient descent is difficult, IEEE Trans. Neural Netw. 5 (2) (1994) 157–166.
[44] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[45] K. Cho, B. van Merrienboer, Ç. Gülçehre, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using RNN encoder-decoder for statistical machine translation, arXiv:1406.1078 (2014) 1–15.
[46] J. Chung, C. Gulcehre, K. Cho, Y. Bengio, Gated feedback recurrent neural networks, in: Proceedings of the International Conference on Machine Learning, 2015, pp. 2067–2075.
[47] F. Yu, V. Koltun, Multi-scale context aggregation by dilated convolutions, arXiv:1511.07122 (2015) 1–13.
[48] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A.W. Senior, K. Kavukcuoglu, Wavenet: a generative model for raw audio, arXiv:1609.03499 (2016) 1–15.
[49] X. Mao, C. Shen, Y.-B. Yang, Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections, in: Proceedings of the Advances in Neural Information Processing Systems, 2016, pp. 2802–2810.
[50] K. He, X. Zhang, S. Ren, J. Sun, Identity mappings in deep residual networks, in: Proceedings of the European Conference on Computer Vision, Springer, 2016, pp. 630–645.
[51] C. Laurent, G. Pereyra, P. Brakel, Y. Zhang, Y. Bengio, Batch normalized recurrent neural networks, in: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2016, pp. 2657–2661.
[52] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[53] A. van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, et al., Conditional image generation with pixelCNN decoders, in: Proceedings of the Advances in Neural Information Processing Systems, 2016, pp. 4790–4798.

Zhipeng Shen is currently pursuing the Master degree from College of Computer Science and Technology at Zhejiang University of Technology, China. He received B.E. degree from Fujian Normal University, China. His research interests include deep learning and machine learning.

Yuanming Zhang received the Ph.D. degree from Department of Information Science at Utsunomiya University, Japan, in 2010. He currently is an associate Professor in the College of Computer Science and Technology at Zhejiang University of Technology China. His research interests include big data analysis, parallel computing. He is a member of CCF.

Jiawei Lu is a lecturer in the college of Computer Science and Technology at Zhejiang University of Technology China, His current research interests include big data and software architecture. He is a member of CCF.

Xu Jun is a Senior Experimentalist at Zhejiang University of Technology China. His research interests include software service, interaction technology. He is a member of CCF.

Gang Xiao is a Professor in the College of Computer Science and Technology at Zhejiang University of Technology China. His research interests include software component, software product family and intelligent information system. He is a member of CCF.