Time Series Prediction Based on Temporal Convolutional Network

Yujie Liu, Hongbin Dong, Xingmei Wang, Shuang Han
College of Computer Science and Technology
Harbin Engineering University
Harbin, China
donghongbin@hrbeu.edu.cn

***Abstract*—With the development of social life, prediction becomes more and more important. As an emerging sequence modeling model, the temporal convolutional network has been proven to outperform on tasks such as audio synthesis and natural language processing. But it is rarely used for time series prediction. In this paper, we apply the temporal convolutional network into the time series prediction problem. Gated linear units allow the gradient to propagate through the linear unit without scaling so we introduce it in temporal convolutional networks. In order to extract more useful features, we propose a multi-channel gated temporal convolution network model. We use the model for stock closing price prediction, Mackey-Glass time series data prediction, PM2.5 prediction, and appliances energy prediction. The experimental results show that compared with the traditional methods, LSTM, and GRU, the temporal convolutional network, gated temporal convolutional network and multi-channel gated temporal convolution network converge faster and have better performance.**

Keywords—temporal convolutional network; time series prediction; gated linear units

I. INTRODUCTION

Thanks to the rapid development of big data technology, there is a large amount of data generated every day. How to use a large amount of data for prediction is an important but hard problem to solve. Prediction needs to discover the internal correlation of the object and predict future trends. How to extract the internal correlation of object is very important for predicting accuracy. The recently proposed temporal convolutional network can also effectively extract temporal information and outperform on tasks such as audio synthesis and natural language processing.

In order to solve the problem of time series prediction, the British statistician Yule proposed the autoregressive model (AR model) in 1927. In 1931, Varg established a Moving average model (MA model) and an autoregressive moving average model (ARMA model). After that, an autoregressive integrated moving average model (ARIMA model) appeared. With the rapid development of machine learning, many methods in machine learning have been applied to prediction. The prediction has gradually become an important part of supervised learning in machine learning. In 1995, Vladimier N.Vapnik proposed the original support vector machine. Support vector machine is a method based on statistical learning theory. It seeks the best compromise between model complexity and learning ability based on limited sample information, in order to obtain the best promotion ability. Autoencoder is a neural network that includes an encoder and a decoder. It contains an implicit layer inside that produces an encoded representation input. After training, the autoencoder can copy the input approximation to the output, enabling it to output the portion which needs to be copied preferentially for learning purposes. The Artificial neural network emerged in the 1980s. It abstracting the network of human brain neurons, establishing a simple model, and forming different networks according to different connections. Support vector machine, autoencoders, artificial neural networks are widely used for prediction. The rapid rise of deep learning has led many people to focus on the convolutional neural network (CNN) and recurrent neural networks (RNNs). The convolutional neural network can extract spatial information and are widely used in image processing. Recurrent neural networks are suitable for sequence modeling and are often used for prediction.

The primary method for dealing with prediction problems in deep learning has long been the Recurrent Neural Networks (RNNs). Jozefowicz et al [1]. have searched thousands of RNN network structures and concluded that if there is a better network structure than long short-term memory network （LSTM）, it is not worth looking for. Klaus et al [2]. also considered no variants can significantly improve the standard LSTM architecture after studying a large number of LSTM structures. This shows the importance of RNNs. However, because of the inherent long-term gradient flow，the RNN suffers from gradient vanishing problem, large numbers of parameters and high computation cost. Due to the gates, LSTM effectively alleviated the problem of gradient vanishing problem.

The temporal convolutional network (TCN) is a novel neural network of fully convolutional structures. It has been proved outperform on tasks like audio synthesis, character-level, and word-level language modeling. The temporal convolutional network can handle serialized data very well and use only the previous data to generate future data. Time series prediction requires that only past data can be used to predict future data, so TCN is well suited for time series prediction. We introduce gated linear units(GLU) into the TCN since it

978-1-7281-0801-8/19/$31.00 ©2019 IEEE
ICIS 2019, June 17-19, 2019, China

helps gradients flow through the layers. In order to extract more information, we propose multi-channel gated temporal convolution network (M-GTCN).

The rest of this paper is organized as follows. In section 2, we introduced the work of using CNN for prediction and the related work of TCN in image processing, natural language processing, sound synthesis. Section 3 overviews the architecture of M-GTCN. Section 4 describes the experiments. We summarized our work in section 5.

II. Related Work

Many scholars have combined convolutional neural networks with recurrent neural networks for prediction. Du et al. [3] combined a convolutional network with a long short-term memory network to propose a hybrid model to predict traffic flow. In this model, convolution and pool operations are used for local trends learning, which can preserve the spatial and temporal locality of time series. CNN mainly considers local trend features and provides more fusion features for the hybrid model. Lv et al. [4] proposed a look-up convolution, which embeds the topology of the road network into the convolution to capture more meaningful spatial characteristics. After that, they input the extracted features into the long short-term memory network to extract long-term dependence from time series data. Then they combined long-term dependence with the periodic information extracted by the fully connected layer and the characteristics of the environmental information to predict the final traffic speed. Wang et al. [5] proposed eRCNN networks, which used matrices to represent spatiotemporal information and introduced separate error feedback neurons to predict the traffic speed. These methods typically use convolutional neural networks to extract features and input features into LSTM for prediction.

Li et al. [6] proposed a hybrid network that uses temporal convolutional network and recurrent neural network for video motion segmentation, which achieved good results. Colin Lea et al. [7] proposed a temporal convolutional neural network and applied it to behavior recognition and detection, achieving better results than RNN. Vijayaditya et al. [8] used a temporal convolutional network for acoustic modeling, which outperforms the state-of-the-art low frame rate (LFR) BLSTM models. Kim et al. [9] used a temporal convolutional network to perform three-dimensional human motion recognition. The final model Res-TCN achieve state-of-the-art results on the largest three-dimensional human motion recognition data set NTU-RGBD. The WaveNet [10] proposed by the Google DeepMind team uses a structure that similar to temporal convolutional network for sound generation.

Bai et al. [11] systematically summarized the previous work and used structures such as causal convolution, dilated convolution, residual connection, and fully connected network to form TCN. The TCN has been proved outperform recurrent network on tasks such as handwritten recognition, audio synthesis, and natural language processing. They propose that two principles must be observed in a temporal convolutional network. Firstly, in the convolution structure, there can be no “information leakage”, that is, when calculating the output at time t, it can only involve input before time t. Secondly, the network structure can take sequences of any length. Since the TCN has no information leakage and can process sequence data well, it is very suitable for time series prediction.

III. Model Structure

The temporal convolutional network is a network composed entirely of convolutional structures. It has achieved good results in sequence modeling tasks without using recurrent structure. However, the temporal convolutional network uses dilated convolution, which causes loss of detail features. In this section, we will introduce the TCN，GTCN and multi-channel gated temporal convolution network.

The temporal convolutional network can be considered as a combination of one-dimensional convolution and causal convolution. Then introduces a dilated convolution to increase the receptive field. The important structures in the temporal convolutional network include causal convolutions [12], dilated convolutions [13], and residual connections [14].

A. Causal Convolution

TCN cannot leak future information to the present when it is predicted. It means it can only use the current time and its previous information when predicting. In order to avoid information leakage causal convolution was introduced. Causal convolution guarantees that only elements at time t and before are involved in convolution at time t. As shown in Fig. 1, for the 5 layer network, only the 5 units X_{t-4}, X_{t-3}, X_{t-2}, X_{t-1}, X_t participating in the convolution, which can effectively avoid information leakage.

B. Dilated Convolution

The dilated convolution was first applied to image semantic segmentation. It solves the problem of downsampling which reduces image resolution and loss of information. The dilated convolution structure, as its name suggests, adds "hole" to the convolution to increase the receptive field. The dilated rate is introduced in the dilated convolution, which refers to the number of intervals between two adjacent points in the convolution. Usually for a one-dimensional sequence input $\mathrm{x} \in \mathbb{R}^n$, with a convolution kernel $f:\{0,\cdots,k-1\} \to \mathbb{R}$, the dilated convolution operation F on the elements S of the sequence is defined as follows.

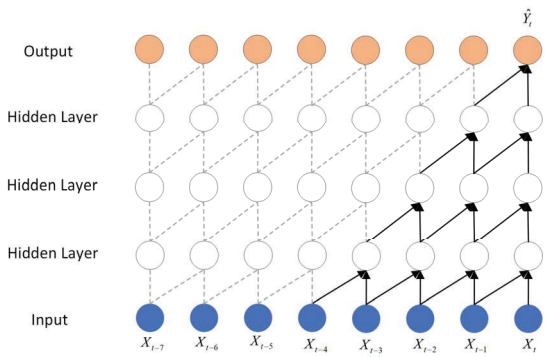

Fig. 1. Causal convolutions.

$$F(s) = (x *_d f)(s) = \sum_{i=0}^{k-1} f(i) \cdot x_{s-d \cdot i} \quad (1)$$

In (1), d is the dilation factor and k is the convolution kernel size, $s - d \cdot i$ indicating the past direction. The dilated convolution increases the receptive field of the convolutional layer exponentially. Increasing the dilated rate enhances the representation of the convolutional network, enabling the final output layer to represent larger inputs. By using dilated convolution, the receptive field of temporal convolutional network increases with fewer parameters and less computational cost.

C. Residual Structure

He et al. proposed the ResNet network structure, which effectively increases the depth of the network by using the residual structure [14]. A residual connection can usually be expressed as follows.

$$output = Activation(input + F(input)) \quad (2)$$

For the temporal convolutional network, the receptive field can be increased by increasing the number of layers, kernel size, and dilated rate. When a prediction task needs a receptive field to be 2^{10}, the temporal convolutional network may require 10 layers. However, as the depth of the network increases, the degradation problem will occur. Introducing the residual structure can effectively solve the problem of deep network degradation.

D. Gated Linear Units

Gated linear units (GLU) have proven to be very effective in natural language processing [16]. It allows the gradient to propagate through the linear unit without scaling while keeping the non-linear capabilities of the layer. GLU can be expressed as $X \otimes \sigma(X)$. The gradient of GLU is calculated as (3).

$$\nabla\left[X \otimes \sigma(X)\right] = \nabla X \otimes \sigma(X) + X \otimes \sigma'(X) \nabla X \quad (3)$$

In (3) the activated gating units $\sigma(X)$ do not downscale since there is a path $\nabla X \otimes \sigma(X)$.

E. Multi-channel Gated Temporal Convolution Network

In order to obtain more complete features, we propose a multi-channel gated temporal convolutional network M-GTCN.

As shown in Fig.2, 1D convolution, Weight Normalization [15], gated linear unit [16], and DropOut [17] are connected to form a convolutional module. Two convolutional modules are stacked together and added to a 1×1 convolution to form a residual structure. The residual structure used can solve the problem of mismatch between input and output channels. We use multiple layers of residual structure and a fully connected layer to form a gated temporal convolutional network (GTCN).

Unlike traditional convolutional neural networks, GTCN does not use two-dimensional convolution and pooling layers. It is also not only used to extract features in time series prediction. It can extract features hidden in time series data through a causal convolution structure and then directly generate prediction results. The traditional convolutional neural network is subject to the fixed receptive field. The GTCN uses the dilated convolution to make the receptive field expand at a small cost. The receptive field can be changed through the dilated convolution. So that the receptive field will become larger as the number of layers increases. Compared with RNN and its variants, GTCN can be parallelized, the processing speed has obvious advantages RNN suffer from exploding and vanishing gradients. GTCN has a back propagation path, using the GLU activation function and the residual structure to avoid exploding and vanishing gradients.

M-GTCN put multiple layers of the residual structure together to form a channel. Each channel randomly forgets the feature when extracting features. Using each channel to extract feature separately， then fuse the features to obtain more information. Then we use a fully connected layer to output the prediction. Fig.3 shows the structure of M-GTCN.

The features extracted for each channel are duplicated. To reduce the effects of repetitive features, we use a fusion matrix. The fusion matrix which trainable uses backpropagation to train. It changes as features change to reduce feature redundancy. The fusion matrix is as follows.

$$Y_{hybrid} = W_{tcn_1} \odot Y_{tcn_1} + \ldots + W_{tcn_i} \odot Y_{tcn_i} \quad (4)$$

We use (4) for fusion in which Y_{hybrid} are features generated by the hybrid network, Y_{tcn_i} are features generated by the i-th channel. Trainable Y_{tcn_i} are the coefficients corresponding to i-th channel. $\odot$ is element-wise multiplication. No intervention

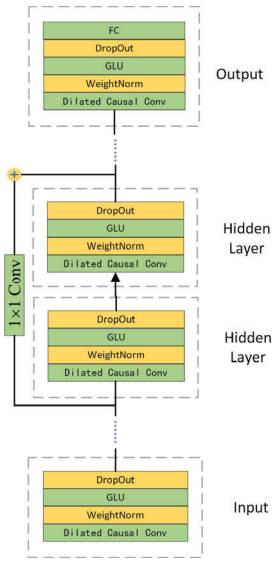

Fig. 2. Structure of GTCN.

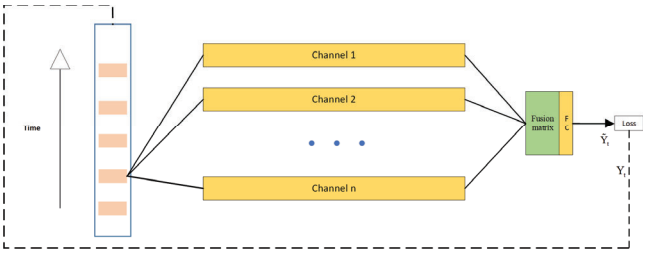

Fig. 3. Structure of M-GTCN.

is required during the fusion process. The end-to-end process improves both fusion efficiency and complexity of manual tuning. Our M-GTCN can be trained via backpropagation to predict Y_t by minimizing the mean squared error between the predicted value and the true value:

$$L(\theta) = \| Y_t - \hat{Y}_t \|_2^2 \tag{5}$$

where θ are all learnable parameters.

IV. Experiments

We evaluate our model on single-factor predictive tasks and multi-factor predictive tasks. On the single variable predictive task, we selected the Microsoft stock daily closing price and Macker-Glass time series data. In the multivariable predictive task, we selected the Beijing PM2.5 data set and appliances energy prediction.

A. Data Preparation

In our experiments, the four datasets are shown as follows.

Microsoft Stock Information. The dataset contains 800 data on Microsoft stock closing prices from September 12, 2007, to November 11, 2010. We select 640 data as the training set and the last 160 as the test set.

Mackey-Glass Time Series Data. Mackey-Glass time series are artificially generated data that is commonly used to test the learning and generalization capabilities of neural networks. It is generated by the following formula.

$$\frac{dx}{dt} = \frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)} - 0.1x(t) \tag{6}$$

When $\tau > 16.8$ the time series generated by (6) can be considered as chaotic, without obvious periodicity. And the Mackey-glass time series data can be considered as a random sequence. We set $x(0) = 1.2$, $\tau = 17$ to generate 1000 data. We select 800 data as the training set and 200 as the test set.

Beijing PM2.5. The Beijing PM2.5 data dataset was got from the UCI Machine Learning Repository [22]. The data set collected PM2.5 concentration, dew point, temperature, air pressure, wind direction, cumulated wind speed, cumulated hours of rain, cumulated hours of snow information hourly from January 1, 2010, to December 31, 2014, in Beijing. We set the PM2.5 concentration as label and using other parameters to predict the PM2.5 concentration. We take three-year data as a training set, one year as a test set, and one year as a validation set from the five-year data.

Appliances energy prediction data. The data set was got from the UCI Machine Learning Repository [23]. We take the appliances energy use as the label. We use light energy consumption, temperature, air pressure, air humidity, wind speed, visibility, and dew point to predict appliances energy consumption. We select 10,000 data as the training set, 4000 data as the validation set, and the rest as the test set.

In data preprocessing, we use the (7) to standardize the data.

$$\begin{cases} X_{std} = \dfrac{X - X_{min}}{X_{max} - X_{min}} \\ X_{scaled} = X_{std} * (max - min) + min \end{cases} \tag{7}$$

In (7), X is the value to be standardized, X_{min} is the minimum value in the feature, X_{max} is the maximum value in the feature, X_{std} is the value which has been standardized, max is the maximum value of feature we want to achieve, min is the minimum value of the feature we want to achieve, X_{scaled} is the scaled value we want. In this experiment, we set max to one and min to zero.

B. Benchmarks

ARIMA: Auto-Regressive Integrated Moving Average (ARIMA) is a classic model for time series predicting.

LSTM: Long-short-term-memory network (LSTM) is a special kind of RNN. It's gated structure avoid exploding and vanishing gradients. It is a famous model for time series predicting.

GRU: Gated-recurrent-unit network, a new kind of RNN, can be used to capture long-term temporal dependencies.

C. Performance Comparisons

We now discuss the results of single-factor predictive tasks. In both Microsoft stock dataset and Mackey-Glass dataset, TCN, GTCN, M-GTCN has a huge advantage compared with traditional method like ARIMA, ANN, ANFIS, Classical RBF, Generating Fuzzy Rules, and SVR's variants SVR-GA, SVR-CGA, SVR-FA, SVR-CFA. They outperform LSTM, GRU. In two dataset M-GTCN outperform GTCN and TCN. In the Microsoft stock experiment the channel of M-GTCN is 2 and in Mackey-Glass experiment is 3. We use the same layers for TCN and GTCN which is 5 for Microsoft stock experiment and 9 for Mackey-Glass experiment. Table I shows the performance of different methods on the task of Microsoft stock prediction. Table II shows the result of the Mackey-Glass time series prediction. Fig.4 shows the comparison of Microsoft stock predicted results.

Then we discuss the results on multi-factor predictive tasks. In Beijing PM2.5 dataset and appliances energy dataset GTCN and M-GTCN outperform LSTM, GRU, and ARIMA. TCN is worse than GRU in both datasets but better than LSTM in Beijing PM2.5 dataset. We use 5 layers in Beijing PM2.5 dataset and 8 layers in appliances energy dataset for both TCN and GTCN. For M-GTCN we use 3 channels for Beijing PM2.5 and 2 channels for appliances energy. Table IV shows the performance of different methods on the task of Beijing PM2.5 prediction. Table III shows the result of appliances energy prediction. Fig.5 shows the comparison of Beijing PM2.5 predicted results.

TABLE I. MSE AND MAPE OF MICROSOFT STOCK INFORMATION ON DIFFERENT METHODS

Method	MSE	MAPE
SVR-GA[18]	0.00192163	0.066795
SVR-CGA[18]	0.00143307	0.061031
SVR-FA[18]	0.00110032	0.052653
SVR-CFA[18]	0.00106339	0.051907
ANN[18]	0.001140	0.054386
ANFIS[18]	0.001212	0.056344
Grey extreme learning machine	0.0005814	0.039151
LSTM	0.00055909	0.03724987
ARIMA	0.03259896	0.35927804
GRU	0.00047792	0.03481491
TCN	0.00037001	0.02961370
GTCN	0.00033233	0.02877396
M-GTCN(2)	**0.00031454**	**0.02728654**

TABLE II. RMSE OF MACKEY-GLASS TIME SERIES DATA ON DIFFERENT METHODS

Method	RMSE
Generating Fuzzy Rules[19]	0.0907
Classical RBF[20]	0.0114
Genetic algorithm and fuzzy system[21]	0.049
Grey extreme learning machine	0.0103
LSTM	0.00437592
GRU	0.00573217
ARIMA	0.21825119
TCN	0.00408165
GTCN	0.00335641
M-GTCN(3)	**0.00266150**

TABLE III. RMSE AND MAE OF APPLIANCES ENERGY PREDICTION DATASET ON DIFFERENT METHODS

Method	RMSE	MAE
LSTM	0.04256562	0.02742011
GRU	0.04218603	0.02680011
ARIMA	0.09261359	0.05281116
TCN	0.04665923	0.02547069
GTCN	0.04134098	0.02501872
M-GTCN(2)	**0.04023805**	**0.02513352**

TABLE IV. RMSE AND MAE OF BEIJING PM2.5 DATESET ON DIFFERENT METHODS

Method	RMSE	MAE
ARIMA	0.09420083	0.06851026
LSTM	0.03253991	0.02306154
GRU	0.02564099	0.01665904
TCN	0.02890710	0.01923636
GTCN	0.02624092	0.01734651
M-GTCN(3)	**0.02294207**	**0.01396601**

D. *Analysis of Results*

TCN is significantly better than LSTM and GRU in single-factor time series prediction, but the advantage is not obvious in multi-factor time series prediction, even worse than GRU. We believe that TCN can extract features well for simple single-factor prediction, but for complex multi-factor prediction, TCN misses some features, making the prediction worse. By using M-GTCN, we have gained more features and improve accuracy. We analyzed the convergence rate of the four sets of experiments and the results are shown in Fig.6. Overall, TCN and M-GTCN converge faster than LSTM and GRU. Although M-GTCN is more complex than the TCN, it does not have a significant increment in time cost because it can be executed in parallel. M-GTCN outperform LSTM, GRU, TCN with little increment in time cost.

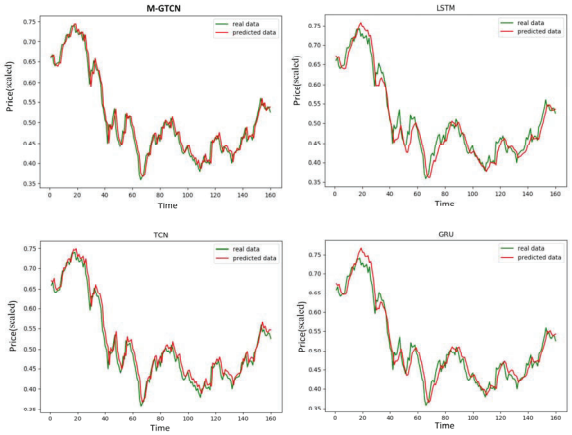

Fig. 4. Comparison of real data and predicted data of different methods in Microsoft stock information dataset.

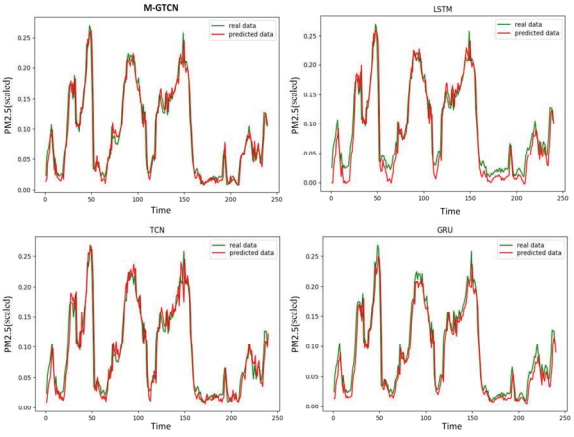

Fig. 5. Comparison of real data and predicted data of different methods in Beijing PM2.5 dataset.

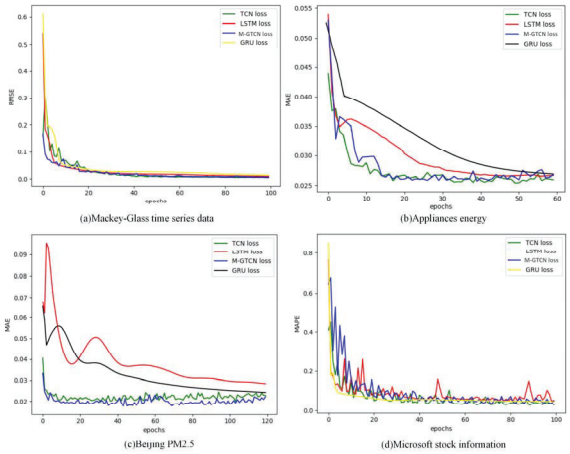

Fig. 6. Convergence results for four dataset.

V. CONCLUSION

In this paper, we apply the temporal convolutional network into the time series prediction problem. Then we introduce gated linear units in TCN to form GTCN. Based on this, we propose M-GTCN which can make end-to-end predictions. Compared with TCN, M-GTCN can extract more features and fuse features with trainable fusion matrix. Thanks to parallelism, our model can improve prediction accuracy without significantly increasing time cost. The experiment results show that the M-GTCN outperform LSTM, GRU, TCN, GTCN in single-factor and multi-factor predictive tasks.

ACKNOWLEDGMENT

We would like to acknowledge the support from the National Science Foundation of China (Nos.61472095).

REFERENCES

[1] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An empirical exploration of recurrent network architectures," in International Conference on Machine Learning, 2015, pp. 2342–2350.

[2] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," IEEE Trans. neural networks Learn. Syst., vol. 28, no. 10, pp. 2222–2232, 2017.

[3] S. Du, T. Li, X. Gong, Y. Yang, and S. J. Horng, "Traffic flow forecasting based on hybrid deep learning framework," in 2017 12th International Conference on Intelligent Systems and Knowledge Engineering (ISKE), 2017, pp. 1–6.

[4] Z. Lv, J. Xu, K. Zheng, H. Yin, P. Zhao, and X. Zhou, "LC-RNN: A Deep Learning Model for Traffic Speed Prediction.," in IJCAI, 2018, pp. 3470–3476.

[5] J. Wang, Q. Gu, J. Wu, G. Liu, and Z. Xiong, "Traffic speed prediction and congestion source exploration: A deep learning method," in 2016 IEEE 16th International Conference on Data Mining (ICDM), 2016, pp. 499–508.

[6] L. Ding and C. Xu, "Tricornet: A hybrid temporal convolutional and recurrent network for video action segmentation," arXiv Prepr. arXiv1705.07818, 2017.

[7] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks for action segmentation and detection," in proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 156–165.

[8] V. Peddinti, Y. Wang, D. Povey, and S. Khudanpur, "Low latency acoustic modeling using temporal convolution and LSTMs," IEEE Signal Process. Lett., vol. 25, no. 3, pp. 373–377, 2018.

[9] T. S. Kim and A. Reiter, "Interpretable 3d human action analysis with temporal convolutional networks," in 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2017, pp. 1623–1631.

[10] A. Van Den Oord et al., "WaveNet: A generative model for raw audio.," SSW, vol. 125, 2016.

[11] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," arXiv Prepr. arXiv1803.01271, 2018.

[12] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang, "Phoneme recognition using time-delay neural networks," Backpropagation Theory, Archit. Appl., pp. 35–61, 1995.

[13] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," arXiv Prepr. arXiv1511.07122, 2015.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[15] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," in Advances in Neural Information Processing Systems, 2016, pp. 901–909.

[16] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," in Proceedings of the 34th International Conference on Machine Learning-Volume 70, 2017, pp. 933–941.

[17] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," arXiv Prepr. arXiv1207.0580, 2012.

[18] A. Kazem, E. Sharifi, F. K. Hussain, M. Saberi, and O. K. Hussain, "Support vector regression with chaos-based firefly algorithm for stock market price forecasting," Appl. Soft Comput., vol. 13, no. 2, pp. 947–958, 2013.

[19] L.-X. Wang and J. M. Mendel, "Generating fuzzy rules by learning from examples," IEEE Trans. Syst. Man. Cybern., vol. 22, no. 6, pp. 1414–1427, 1992.

[20] K. B. Cho and B. H. Wang, "Radial basis function based adaptive fuzzy systems and their applications to system identification and prediction," Fuzzy sets Syst., vol. 83, no. 3, pp. 325–339, 1996.

[21] D. Kim and C. Kim, "Forecasting time series with genetic fuzzy predictor ensemble," IEEE Trans. Fuzzy Syst., vol. 5, no. 4, pp. 523–535, 1997.

[22] X. Liang et al., "Assessing Beijing's PM2. 5 pollution: severity, weather impact, APEC and winter heating," Proc. R. Soc. A Math. Phys. Eng. Sci., vol. 471, no. 2182, p. 20150257, 2015.

[23] L. M. Candanedo, V. Feldheim, and D. Deramaix, "Data driven prediction models of energy use of appliances in a low-energy house," Energy Build., vol. 140, pp. 81–97, 2017.