2017 International Conference on Computer Science and Application Engineering (CSAE 2017)
ISBN: 978-1-60595-505-6

Internet Traffic Forecasting using Boosting LSTM Method

Guangkuo Bian*, Jun Liu and Wenhui Lin
School of Information and Communication Engineering, Beijing University of Posts & Telecommunications Address: No.10 Xitucheng Rd, Haidian District, 100876 Beijing, China

ABSTRACT

As Internet traffic is a kind of time series, the algorithms which are suitable for time series forecasting are also appropriate for Internet traffic. We study approaches for time series prediction firstly and then apply the algorithms to Internet traffic prediction in this paper. We present an ensemble method based on Long Short-Term Memory (LSTM) Network method for time series prediction with the aim of increasing the prediction accuracy of Internet traffic. We use AdaBoost algorithm to boost LSTM method because AdaBoost algorithm is the most widely used boosting algorithm. However, original AdaBoost algorithm is not suitable for prediction, in view of this fact, we use modified AdaBoost algorithm to be combined with LSTM in this paper. We use two data sets to experiment, and the results show that AdaBoost-LSTM algorithm is more accurate than single LSTM algorithm and even better than ARIMA in some situations. At last, we apply AdaBoost-LSTM algorithm to forecast the Internet traffic.

INTRODUCTION

Time series prediction can solve many problems in different domains ranging from finance, dynamic systems control and marketing. Moreover, the reliable prediction of future values of Internet traffic as a type of time series plays an important role in network security and resource management.

As Internet traffic is a kind of time series, time series prediction methods are always suitable for Internet traffic forecasting. In this way, we do researches on general time series prediction firstly and apply the algorithms to predict Internet traffic next.

In time series prediction domain, there have been many forecasting methods. They can be classified into two kinds: linear and nonlinear prediction algorithms. Linear forecasting algorithms include ARMA/ARIMA model 123 and Holt-Winters methods while nonlinear forecasting algorithms contain neural networks 45. It indicates that nonlinear forecasting algorithms using neural networks perform better than linear prediction algorithms such as ARIMA in 6. All these prediction methods can be applied to Internet traffic flow.

With the development of deep learning, there are more and more researches on how to apply deep learning methods such as RNN or LSTM algorithms to time series prediction. Nowadays, several methods using RNN and LSTM algorithms for predicting traffic flow have been proposed 789. In addition, in time series prediction domain, RNN-based ensembles have been proposed in [10][11] to make RNN algorithm more robust. It has been proved that these RNN-based ensembles are better than single RNN algorithm in [10][11]. Inspired by [10][11], we present a method using the modified AdaBoost algorithm ensemble with LSTM network called "AdaBoost-LSTM" for time series forecasting problems to improve the performance of single LSTM algorithm and apply this algorithm to Internet traffic prediction in this paper. To verify the accuracy of the method, two time series problems are employed in the experiment.

Conventional RNN has the exploding gradient and vanishing problems. LSTM, one special network of RNN, has been proposed to improve the performance of RNN and address the problems of conventional RNN. In some situations, an LSTM can offer a more accurate result in predicting non-linear time series.

There are many boosting algorithms, the most widely used algorithm is AdaBoost algorithm. AdaBoost algorithm trains different weak classifiers based on the same training set and combine these classifiers into a single strong classifier, the ensemble model has a good performance on all samples. However, since original AdaBoost algorithm is only suitable for classification but not regression, we modify original AdaBoost algorithm to make it appropriate for time series prediction.

In this paper, the concept of LSTM networks as well as RNN networks will be described in section 2. In section 3, we will introduce AdaBoost algorithm and in section 4, the modification of AdaBoost algorithm will be proposed. In section 5, AdaBoost-LSTM algorithm will be described. We use AdaBoost-LSTM algorithm to forecast future values on two benchmark data sets and then analyze the results and compare the results with those using ARIMA model in section 6. Then the method we proposed will be applied to Internet traffic time series prediction in section 7. And finally, a conclusion is included at last.

NEURAL NETWORKS

Recurrent Neural Networks

Since Long Short-Term Memory Networks (LSTMs) are a special kind of Recurrent Neural Networks (RNNs), we introduce RNNs firstly. RNNs are called recurrent because the connections between units compose a cycle. The previous computations can also affect the output being. The figure below shows the architecture of a typical RNN:

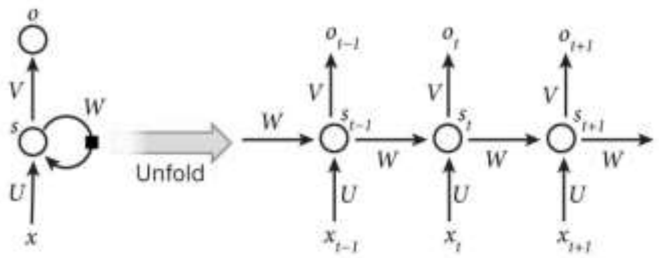

Figure 1. RNN architecture.

The formulas shown in the Figure 1 are as follows: x_t, s_t and o_t respectively represent the input, the hidden state and the output at time step t. s_t is computed as

$s_t = f(Ux_t + Ws_{t-1})$ where the function f is a nonlinear function such as *tanh*. o_t is computed as $o_t = softmax(Vs_t)$.

All recurrent neural networks are composed of a chain of repeating modules of neural network. In order to explain LSTMs intuitively, we provide a Figure 2 which shows the repeating module of standard RNNs below 12. It shows that the repeating module has a very simple structure.

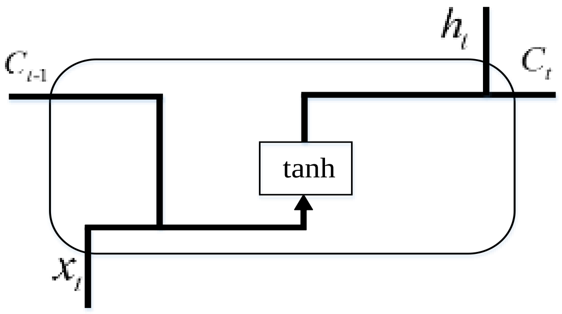

Figure 2. The repeating module of RNNs.

Long Short-Term Memory Networks

Above is a brief introduction of RNNs, the concept of LSTMs is described next. Long Short-Term Memory (LSTM) was proposed by Sepp Hochreiter and Jürgen Schmidhuber [13] and improved by Felix Gers et al in 2000 [14]. It works especially well in a variety of domains such as classification problems and even time series prediction problems.

Standard Recurrent Neural Networks (RNNs) suffer from vanishing and exploding gradient point problems. LSTMs are designed to address these problems. LSTMs deal with these problems by introducing new gates, such as input and forget gates, which allow for a better control over the gradient flow and enable better preservation of long-range dependencies.

Same as RNNs, LSTMs also have a repeating module. But the repeating module is different from RNNs. The architecture of the repeating module is complicated.There are four neural network layers in LSTMs including input layer, gate layer, cell layer and output layer rather than single simple *tanh* layer. The repeating module is shown in Figure 3 as below [12].

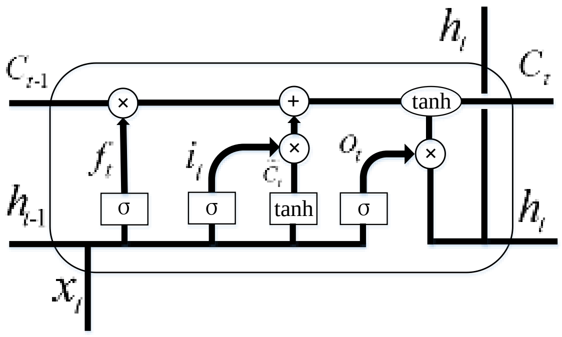

Figure 3. The repeating module of LSTMs.

For a given input sequence $x = (x_1, x_2, \dots, x_T)$, an LSTM network computes an output sequence $h = (h_1, h_2, \dots, h_T)$. When we take the last output h_{t-1}, the last cell state C_{t-1} and the current input x_t as inputs, we can use the following equations to obtain the current output h_t and the current cell state C_t, then h_t and C_t will be used in the next iteration. The output sequence $h = (h_1, h_2, \dots, h_T)$ is calculated iteratively from t = 1 to T with the following equations.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{1}$$

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{2}$$

$$\tilde{C}_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{3}$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{4}$$

$$o_t = \sigma(W_o [h_{t-1}, x_t] + b_o) \tag{5}$$

$$h_t = o_t * tanh(C_t) \tag{6}$$

Here the *W* terms signify weight matrices, for example, W_i denotes the matrix of weights from the input gate to the input, σ is sigmoid function, and the *b* terms represent bias vectors. *i* is the input gate while *f* is the forget gate, *o* is the output gate, *C* represent the cell state and h_t is the output.

ADABOOST

There are many boosting algorithms, the most widely used algorithm is AdaBoost. AdaBoost is a method that combines many simple weak learners into a strong classifier. The final output is calculated as combining the output of weak learners into weight sum. Many other types of learning algorithms can be combined into one model using AdaBoost algorithm to improve their performance. The final model is a strong model when every weak learner satisfies that the performance of it is better than random guessing. The detail of AdaBoost algorithm will be introduced next.

Here we are given a training set, $(x_i, y_i), i = 1, \dots, n$ as inputs where x_i is a sample in domain X, $X = \{x_i, i = 1, \dots, n\}$, and $y_i \in \{-1, +1\}$ is a label associate with x_i. AdaBoost trains for *K* iterations to get *K* weak learners to combine. At each iteration *k*=1, ..., *K*, a set of weights on the training set is assigned by the algorithm. For *k*=1, all the weights are set equally. $D_k(i)$ represents the weight of training sample *i* at iteration *k*. Then the weak learner trains itself with this D_k. At next iteration, the learner will pay more attention to the hard samples in the training set by increasing the weight of incorrectly classified samples and decreasing the weight of correctly classified samples. At the iteration, the weak learner has to compute a weak hypothesis $h_k\text{: X} \rightarrow \{-1, +1\}$ with low weighted error ε_k relative to D_k.The final hypothesis *H* is a linear combination of the weak hypothesis h_k , calculated as a weighted sum of h_k, $\text{H(x)} = \text{sign}(\sum_{k=1}^{K} a_k h_k(x))$, where a_k is the weight of the h_k. The Pseudo code for AdaBoost is shown in Figure 4 [15].

```
Given: $(x_i, y_i) \quad x_i \in X, X = \{x_i, i = 1,...,n\}, y_i \in \{-1,+1\}$
Initialize: $D_1(i) = 1/n$
Iterate: For k=1,...,K:
    1. Train weak learner using distribution $D_k$
    2. Get hypothesis, $h_k$, with the error function $\varepsilon_k$, with respect to $D_k$
            $\varepsilon_k = \Pr_{i \sim D_k}[h_k(x_i) \neq y_i]$
    3. Choose $a_k = \frac{1}{2}\ln(\frac{1-\varepsilon_k}{\varepsilon_k})$
    4. Reweight $D_{k+1}(i)$
            $D_{k+1}(i) = \frac{D_k(i)\exp(-a_k y_i h_k(x_i))}{Z_k}$
        Where $Z_k$ is a normalization factor.
            $Z_k = \sum_{i=1}^{n} D_k(i)\exp(-a_k y_i h_k(x_i))$
Output: The final hypothesis
            $H(x) = sign(\sum_{k=1}^{K}(a_k h_k(x))$
```

Figure 4. The AdaBoost method.

```
Given: $(x_i, y_i) \quad x_i \in X, X = \{x_i, i = 1,...,n\}; y_i \in Y, Y = \{y_i, i = 1,...,n\}$
Initialize: $D_1(i) = 1/n$
Iterate: For k=1,...,K:
    1. Train weak learner using distribution $D_k$
    2. Get hypothesis, $h_k$, with the error function $\varepsilon_k$, in regard to $D_k$.Error
function $\varepsilon_k$ can be the difference between the actual and predicted
values or other error functions which is then weighted by $D_k$
    3. Choose $a_k = \frac{1}{2}\ln(\frac{1-\varepsilon_k}{\varepsilon_k})$
    4. Reweight $D_{k+1}(i)$
            $D_{k+1}(i) = \frac{D_k(i)\exp(a_k \varepsilon_k(i)/\varepsilon_k)}{Z_k}$
        Where $Z_k$ is a normalization factor.
            $Z_k = \sum_{i=1}^{n} D_k(i)\exp(a_k \varepsilon_k(i)/\varepsilon_k)$
Output: The final hypothesis
            $H(x) = \sum_{k=1}^{K}(a_k h_k(x))/\sum_{k=1}^{K} a_k$
```

Figure 5. The modified AdaBoost method for prediction problems.

MODIFIED ADABOOST

Since we are studying on time series prediction problems, here we are interested in how to use AdaBoost algorithm for prediction. It is generally harder to solve prediction problems using boosting methods than solving classification problems. The outputs of classifiers are classes while the outputs of predictors are real value rather than classes. Original AdaBoost algorithm is not suitable for prediction problems. To suit time series forecasting problems, modified AdaBoost algorithm were proposed in 101115.The modified AdaBoost algorithm we use for prediction in this paper is shown in Figure 5.

There are several differences between modified AdaBoost algorithm and original AdaBoost algorithm. Modified AdaBoost algorithm takes a training set, $(x_i, y_i), i = 1, \dots, n$ as inputs where x_i is an instance in domain X, $X = \{x_i, i = 1, \dots, n\}$, and y_i is an instance in domain Y associated with x_i, $Y = \{y_i, i = 1, \dots, n\}$. Different from original AdaBoost algorithm, y_i in modified AdaBoost is real-valued. At iteration *k*, the LSTM network is trained using $D_k(i)$ over the training set. At first, $D_1(i)$ are set equally, $D_1(i) = 1/n$. Then the network computes the weak hypothesis h_k using error function ε_k, in regard to D_k. ε_k is a user-defined unit error function weighted by D_k in modified AdaBoost algorithm, for instance, $\varepsilon_k = (y - y_{pred})^2$, while ε_k is the summation of the probabilities of incorrectly classified samples in original AdaBoost algorithm. D_{k+1} is reweighted by changing the weight of samples in accordance with the error function next. A normalization factor Z_k is used to constrain D_{k+1} as a distribution, Since D_{k+1} is reweighted at each iteration, network is forced to focus on the harder samples. At last, the final output is computed by taking the outputs of trained networks and combining them together according to their parameter a_k which reflects the importance of the network.

ADABOOST-LSTM ALGORITHM

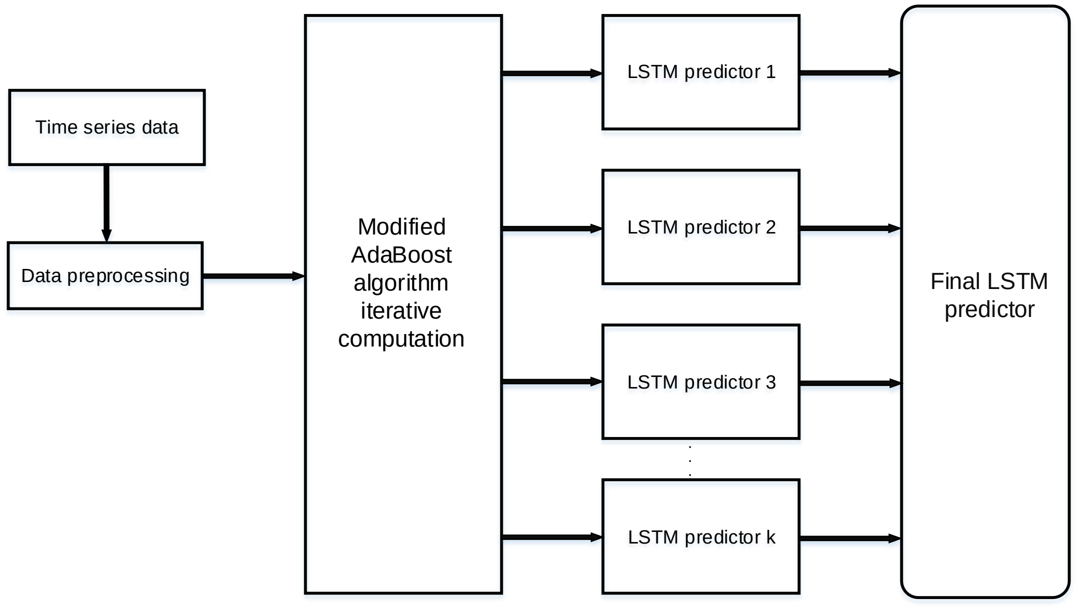

Figure 6. The AdaBoost-LSTM algorithm.

The AdaBoost-LSTM algorithm which we proposed in this paper is shown in Figure 6, we use the modified AdaBoost method to iteratively train LSTM networks, after k iterations, we get k weak LSTM predictors, then we combine these k predictors into one predictor to forecast the Internet traffic data. In section 6, We have done some experiments using AdaBoost-LSTM algorithm.

EXPERIMENTAL RESULTS

In this paper, two data sets are used to verify the effectiveness of AdaBoost-LSTM algorithm. One is a sunspot data set and another is a random data set. All the associated programs are written in Python, the version of Python is 3.7.

We use the root mean squared error (RMSE) to judge the performance of each algorithm and compare with each other. RMSE is computed as follows[16]:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n}\left|y_t - y_{pre}\right|^2} \qquad (7)$$

Here n represents the size of the test set, y_t and y_{pre} respectively represent the actual value and the forecasted value.

For each data set, we firstly display the origin data using diagram and then compare our obtained results with other methods using tabular form. At last we use some graphs to show the test and forecasted values. In every forecast graph, the actual observations are represented by the solid line while the forecasted observations are plotted using the dotted line.

The Wolf's Sunspot Dataset

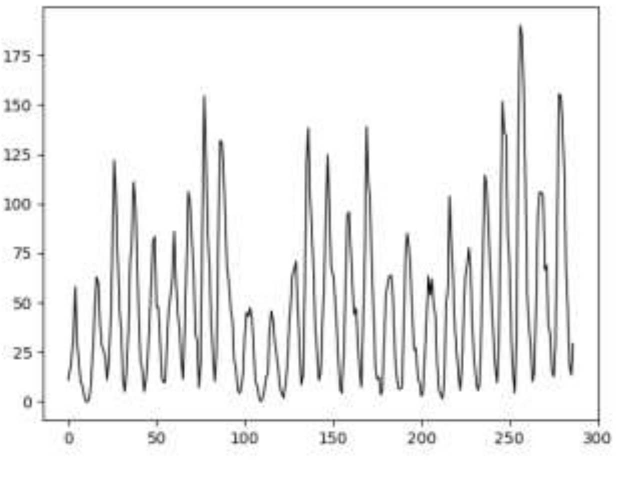

Figure 7. Wolf's sunspot data series (1700-1987).

The Wolf's sunspot series displayed in Figure 7 is the yearly number of sunspots from 1700 to 1987 [17]. The data series is non-linear and has 288 observations in total. In the paper, the first 221 (i.e., 1700 to 1920) observations are taken as training data set and the remaining 67 (i.e., 1921 to 1987) are used for the test data set. In [16], it has been verified that AR (9) is the optimal ARIMA model for this series. We have fitted to the sunspots series using AR(9), LSTM (60 hidden units, 200 epochs), AdaBoost-LSTM (15 boost rounds, 60 hidden units, 200 epochs) and SVM. We measure the performance with RMSE, the results are shown in the TABLE I:

TABLE I. SUNSPOT SERIES PREDICTION RESULTS.

Method	RMSE
AR(9)	**21.99**
LSTM	25.44
AdaBoost-LSTM	22.35
SVM $\begin{pmatrix} C = 43.6432 \\ \sigma = 290.1945 \\ n = 9, N = 212 \end{pmatrix}$	28.16

It is easy to see that the forecasting performance of AR(9) is the best in this test for the sunspot series from the above table. AdaBoost-LSTM is better than LSTM and they are all better than SVM algorithm. Here we just show LSTM and AdaBoost-LSTM forecast diagrams, other models' forecast diagrams can be found in 16. The two prediction diagrams for the sunspot series are shown in the figures below:

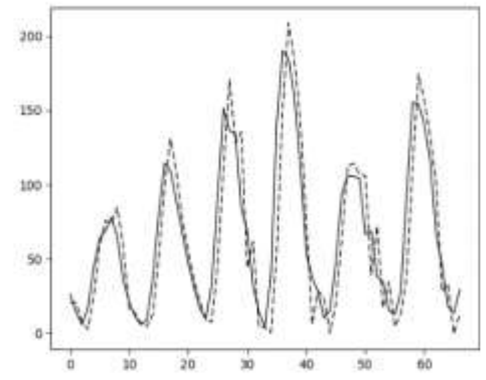

Figure 8. Prediction of Wolf's sunspot using LSTM algorithm.

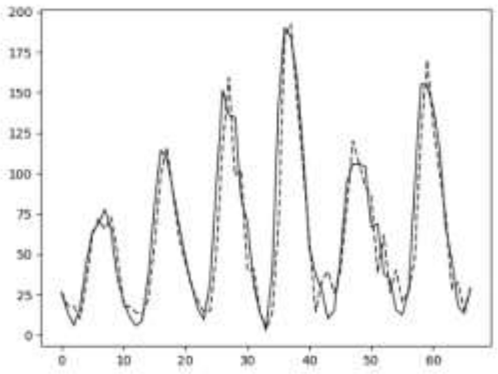

Figure 9. Prediction of Wolf's sunspot using AdaBoost-LSTM algorithm.

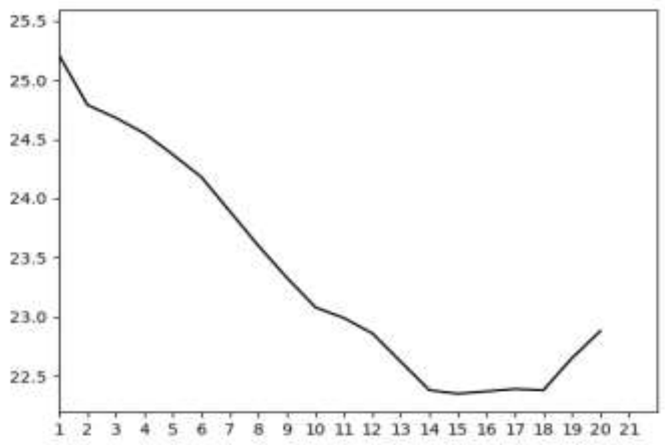

Figure 10. RMSE of AdaBoost-LSTM algorithm with different values of AdaBoost round.

There are 60 hidden units in the LSTM cell, and we specify 200 epochs to train the model. From Figure 8 and Figure 9, we can see that predicting test set with AdaBoost-LSTM algorithm is better than single LSTM algorithm with performance improvement around 12.15%. In Figure 10, AdaBoost-LSTM algorithm is satisfying if the AdaBoost round is small but is upset if the training round is big. Here we can see that the performance becomes worse if AdaBoost round is bigger than 15. The reason of this phenomenon is that the model is overfitting as the AdaBoost round is big.

The Random Dataset

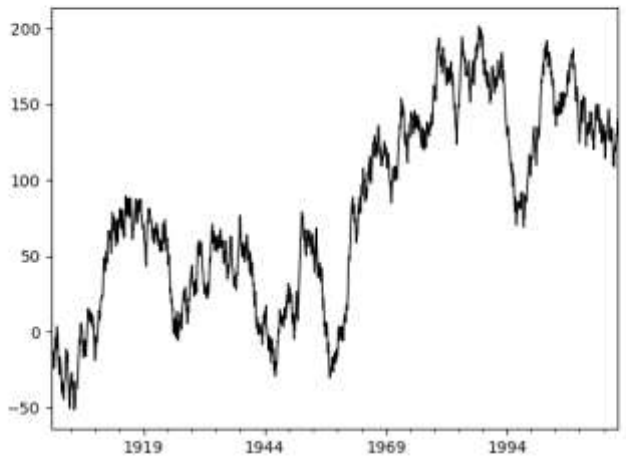

Figure 11. Random data set.

The random series is shown in Figure 11. It is generated randomly. Obviously the data series is non-linear and random, so it’s not suitable to apply ARIMA models to forecast this data series. There are 1404 observations in the data series, we use the first 67% as training data set and the remaining as test data set. In the paper, we compare AdaBoost-LSTM algorithm and Naive model to judge the effectiveness of

AdaBoost-LSTM algorithm. Here, we use 3 rounds to boost and use 200 epochs to train. There are 60 hidden units in the LSTM cell. Naive forecast is a good baseline forecast where the observation from the prior time step t-1 is used to predict the observation at the current time step t. The performance of two algorithms are shown as below TABLE II:

TABLE II. FORECAST RESULTS FOR RANDOM SERIES.

Method	RMSE
Naive	4.85
LSTM	4.75
AdaBoost-LSTM	4.50

From the above table, we can see that AdaBoost-LSTM and LSTM algorithm are both better than Naive model and AdaBoost-LSTM algorithm is greater than LSTM with performance improvement around 7.21%.

APPLY ADABOOST-LSTM ALGORITHM TO INTERNET TRAFFIC PREDICTION

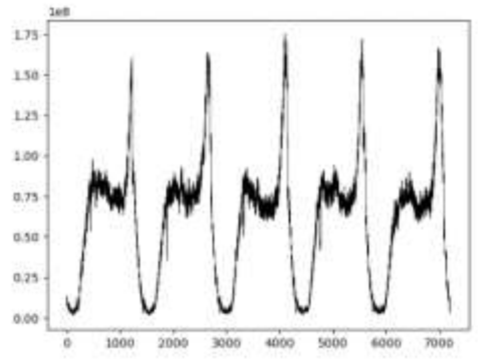
Figure 12. Internet traffic data series.

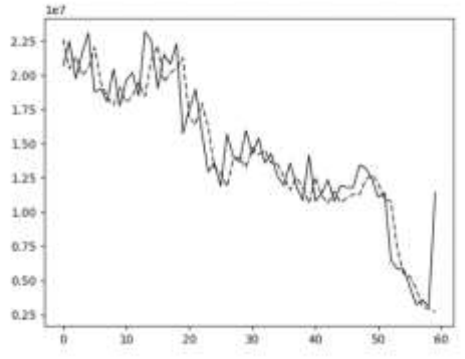
Figure 13. Internet traffic data series prediction.

After above works, we have proved that AdaBoost-LSTM algorithm is better than the standard LSTM algorithm. In purpose of forecasting the Internet traffic time series, we apply AdaBoost-LSTM algorithm to the Internet traffic and forecast its future values. The Internet traffic data series we have is shown in Figure 12.

The data series is periodic and nonlinear, it represents the Internet traffic in our lab from August 1st, 2015 to August 5th, 2015. We use the Internet traffic of the last hour in last day as the test dataset and the remaining as the training dataset. Figure 13 illustrates the prediction result.

CONCLUSIONS

After experimenting and analyzing, there are some conclusions we can obtain as follows:

- AdaBoost-LSTM is proved to be better than the standard LSTM algorithm, marked by the performance increase on all of the testing scenarios.

- AdaBoost-LSTM is easy to configure, there are just few hyperparameters needed to fit, such as AdaBoost round, the number of hidden units in the LSTM cell and training epochs, on the contrast, ARIMA algorithm needs to know which model should be used before training with ACF and PACF analyzing.

ACKNOWLEDGEMENT

This work is supported in part by Fundamental Research Funds for the Central University under Grant No.2016RCGD09 and 111 Project of China (B08004, B17007). This work is conducted on the platform of Center for Data Science of Beijing University of Posts and Telecommunications.

REFERENCES

1. P. Cortez, M. Rio, M. Rocha, and P. Sousa. 2006. “Internet Traffic Forecasting using Neural Networks,” International Joint Conference on Neural Networks, pp. 2635-2642.
2. H. Feng, and Y. Shu. 2005. “Study on Network Traffic Prediction Techniques,” International Conference on Wireless Communications, pp. 1041-1044.
3. J. Dai, and J. Li. 2009. “VBR MPEG Video Traffic Dynamic Prediction Based on the Modeling and Forecast of Time Series”, pp. 1752-1757.
4. V. B. Dharmadhikari, and J. D. Gavade. 2010. “An NN Approach for MPEG Video Traffic Prediction,” 2nd International Conference on Software Technology and Engineering, pp. V1-57V1-61.
5. A. Abdennour. 2006. “Evaluation of neural network architectures for MPEG-4 video traffic prediction,” IEEE Transactions on Broadcasting, pp. 184-192.
6. Barabas, and Melinda, et al. 2011. “Evaluation of network traffic prediction based on neural networks with multi-task learning and multiresolution decomposition,” Intelligent Computer Communication and Processing (ICCP), 2011 IEEE International Conference on, pp. 95-102.
7. P.G. Madhavan. 2002. “Recurrent neural network for time series prediction,” Engineering in Medicine and Biology Society, pp. 250-255.
8. Y.X. Tian, and P. Li. 2016. “Predicting Short-Term Traffic Flow by Long Short-Term Memory Recurrent Neural Network,” Smart City/SocialCom/SustainCom (SmartCity), pp. 153-158.
9. Z. Zhao, W.H. Chen, and X.M. Wu. 2017. “LSTM network: a deep learning approach for short-term traffic forecast,” pp. 68-75.
10. R. Bone, M. Assaad, and H. Cardot. 2008. “A New Boosting Algorithm for Improved time-series forecasting with Recurrent Neural Networks,” Information Fusion, 9, pp. 41-55.
11. R. Soelaiman, A. Martoyo, and Y. Purwananto. 2009. “Implementation of recurrent neural network and boosting method for time-series forecasting,” ICICI-BME, pp. 1-8.
12. Olah, C. 2015. Understanding LSTM Networks. [online] Available at: http://colah.github.io/posts/2015-08-Understanding-LSTMs/ [Accessed 5 July 2017].
13. S. Hochreiter, and J. Schmidhuber. 1997. “Long Short-Term Memory,” Neural Computation, pp. 1735-1780.
14. F.A. Gers, J. Schmidhuber and F. Cummins. 2000. “Learning to Forget: Continual Prediction with LSTM,” Neural Computation, pp. 2451-2471.
15. C.P. Lim and W.Y. Goh. 2006. “The Application of an Ensemble of Boosted Elman Networks to Time Series Prediction: A Benchmark Study,” International Journal of Computational Intelligence, 3(2), pp. 119-126.
16. R. Adhikari, and R.K. Agrawal. “An Introductory Study on Time Series Modeling and Forecasting,” arXiv: 1302. 6613v1.
17. G.P. Zhang. 2003. “Time series forecasting using a hybrid ARIMA and neural network mode”, Neurocomputing, 50, pp. 159-175.