Cost-adaptive Neural Networks for Peak Volume Prediction with EMM Filtering

Bin Yu*, Giovanna Graciani†, Anderson Nascimento†, Juhua Hu†
* *Infoblox, Santa Clara, USA*, biny@infoblox.com
† *Center for Data Science, School of Engineering and Technology, University of Washington, Tacoma, USA*
{giograc, andclay, juhuah}@uw.edu

***Abstract*—As the emergence of the Internet of Things (IoT) and the growing number of IoT devices, a stable connection service has become one of the key factors concerning the Quality of Service (QoS) provision. How to anticipate the peak traffic volume is essential. If the resource allocation is under provisioned, the service becomes susceptible to failure or security breach. Unfortunately, peak volumes are not captured in the systematic components of data and as a result conventional trend prediction methods have proven insufficient. We propose a framework that implements neural networks with filtering and a cost-adaptive loss function to improve the ability to predict peak volumes. Implementing this method on a real Domain Name Server (DNS) traffic data, we observe not only the improvement in the prediction performance but also a shorter lag time to predict peak values, which demonstrates our proposed method.**

***Index Terms*—Quality of Service, network traffic, peak volume prediction, cost-adaptive neural networks**

I. Introduction

The Internet of Things (IoT) is grounded on a connection of billions of devices on the Internet. As the growing number of IoT devices, maintaining a stable connection service has become a challenging task to improve the usage experience of IoT applications. To meet the demands, anticipating peak volume of requests is crucial.

For instance, Domain Name Server (DNS) plays an important role in the connection service, which translates domain names (e.g., google.com) entered by devices to IP addresses for connection. If a DNS is under provisioned, it becomes susceptible to failure or security breach. In fact, DNS query flooding or overloading the DNS server, is a common DNS Denial of Service (DDoS) attack strategy as it can render a network services unresponsive and prevent accessing network resources. As shown in Fig. 1, by deploying a large number of Bad Bots, an entity can overload the DNS Servers and render the entire DNS service incapacitated. DNS servers can also become overloaded by a sudden unexpected rise in benign network traffic [1]–[3]. Therefore, it is important to anticipate the peak volume in different aspects (e.g., DNS and DHCP) to provide a stable connection service in terms of QoS for IoT.

Unfortunately, peak values are not captured in the systematic components of data and as a result conventional trend prediction methods have proven insufficient to generate strong predictions as follows. Simple linear models such as Auto Regressive Integrated Moving Average (ARIMA) have been widely used for general time-series forecasting problems [4]–[7]. Besides, exponential smoothing is a technique that can be used for time-series prediction [8]. However, these works do not address the nontrivial task of peak prediction. Linear models such as ARIMA and exponential smoothing have been shown to lack the ability to effectively capture peak values in data. This is because linear models fail to represent the stochastic and nonlinear nature of a traffic flow [9].

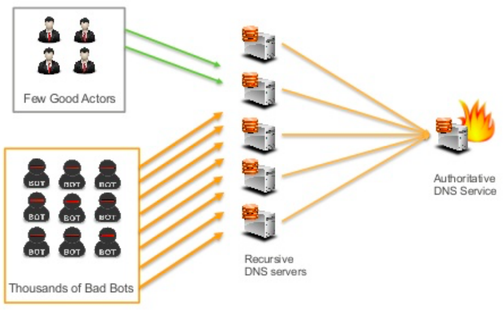

Fig. 1: An Example of DOS attack against a DNS[1].

In contrast, neural networks are well known for capturing the non-linearity of data, and various non-linear models have been applied for network traffic prediction such as Convolutional Neural Networks (CNN), Long Short Term Memory Networks (LSTM), Artificial Neural Networks (ANN), Gated Recurrent Units (GRU) and variants of these models [9]–[11]. However, none of these works address peak prediction. Recently Yu et al. [12], [13] proposed a filter, named EMM, in conjunction with a non-linear model that has shown promising results in peak prediction. EMM filter is also adopted in our work to better prepare the data for peak prediction.

Specifically, we propose a general peak prediction framework. We first prepare the data through the EMM filter [12], [13] to specifically extract peak values within a time window and a local normalization to capture the peak information comparing the most recent volume. More importantly, we propose a cost-adaptive loss function that applies a higher penalty to under-prediction than over-prediction. This is because the effects of under-prediction of traffic volume are more severe than over-prediction, as shortages can result in system failure and security risk. In the application to a real DNS traffic data, we observe an improvement in ability to predict peak volumes, and a shorter lag time for peak predictions. Both of these are essential to maintaining a well provisioned service, which demonstrates our proposed peak prediction framework.

[1]https://www.slideshare.net/AmazonWebServices/dns-ddos-mitigation-using-amazon-route-53-and-aws-shield

978-1-7281-0858-2/19/$31.00 ©2019 IEEE

The rest of the paper is organized as follows. Section II gives an overview of related work. Then, we present our proposed framework in Section III. Section IV reports the results of an empirical study over a real DNS traffic data. Finally, Section V concludes this work and discusses the future work.

II. Related Work

Various time-series forecasting models have been proposed for trend prediction on network traffic, with only a couple focusing on peak value prediction. These models can be classified into two main categories, that is, linear models and non-linear models.

A. Linear Models

There have been many successful applications of ARIMA models to network traffic modeling and predictions [4]–[6]. While these works address time-series forecasting for network traffic data, they lack the capability of peak volume prediction that is a non-trivial task. Peak values are not captured in the systematic components of a dataset and as a result conventional trend prediction methods, such as ARIMA, are insufficient to generate strong predictions on peak values [9]. Some traditional time-series forecasting models such as ARIMA have been applied to peak prediction [14], [15] in some real tasks. However, peak prediction over network traffic data has not been addressed before.

Common filtering approaches such as exponential smoothing [8] suffer from the same caliber of performance as ARIMA models on generating peak prediction due to a similar reliance on the systematic components of data. Recently, Yu et al. [12], [13] proposed an EMM filter which shows promising results when predicting peak values on network data. We adopt this filter to better prepare the data for peak prediction.

B. Non-linear Models

To combat the poor performance of linear models on non-linear data, non-linear models were proposed. Neural networks are well known for capturing the non-linearity of data [16].

Cicek et al. [17] compared the results of an Artificial Neural Network (ANN) and Seasonal ARIMA (SARIMA) models for predicting network traffic and found that their performance are similar. Thereafter, Mozo et al. [10] adopted CNN to forecast short-term changes in the amount of traffic crossing a data center network. They found that CNN model greatly outperforms standard ARIMA models due to the ability of CNNs to exploit the non-linear regularities of network traffic.

Accouni et al. [11] proposed a LSTM model to predict network traffic since LSTMs are well known to capture the temporal dependencies between features more accurately than traditional ANNs. They found that the LSTM model vastly outperforms traditional linear methods and feed forward neural networks, such as CNNs and ANNs. Hua et al. [18] also implemented a LSTM model to predict traffic data. Fu et al. [9] compared the results of LSTM, GRU and ARIMA models to predict short-term traffic flow, and their experiments demonstrate that Recurrent Neural Network (RNN) based deep learning methods such as LSTM and GRU perform better than ARIMA models. Authors of [19] found that various RNN models such as LSTM and GRU achieve similar performance. Unfortunately none of the aforementioned models focus on peak value prediction. As such, the present literature is insufficient to accurately predict peaks in network traffic so as to support the stable connection service for IoT.

III. The Proposed Method

Assume a time-series of the form $\cdots, t_{n-1}, t_n, t_{n+1}, \cdots$ where t_n is the present time in the series, we define a network traffic volume sequence in this time-series as $\cdots, v_{n-1}, v_n, v_{n+1}, \cdots$. Then, our traffic prediction problem can be defined as

$$v_{n+1} = P(v_n, v_{n-1}, ..., v_{n-k+1}) \tag{1}$$

where P is the prediction function to estimate, and k is the step size or the number of history instances used for prediction.

Our peak volume prediction framework consists of two main stages. The first stage is the data preparation with the EMM filter [12], [13] and a local normalization. Then, the second stage is to train a cost-adaptive time-series forecasting model.

A. Data Preparation

To better capture the peak values in history, we apply the EMM filter [12], [13] to the raw traffic data $\cdots, v_{n-1}, v_n, v_{n+1}, \cdots$, which results in the filtered sequence $\cdots, v'_{n-1}, v'_n, v'_{n+1}, \cdots$, where

$$v'_n = \max_{0 \le i \le n} \left\{ \beta^{\frac{i}{\omega}} v_{n-i} \right\} \tag{2}$$

$\beta \in [0, 1]$ is the weight or inheritance coefficient, and ω is the filter kernel size or filter window size. Concretely, the filter is used to focus on the peak values in a given window utilizing a maximum aggregator that applies a magnitude decaying exponentially with time. This effectively keeps all peak values and filters random noise as shown in Fig. 2, where the filter window size is set to 3.

Moreover, traffic volume at different times are often of greatly different scales, and thus normalization is applied. While global normalization is commonly applied for general time-series prediction [5], [11], [18], [20], we apply local normalization to capture peak information comparing the most recent traffic volume. Local normalization results in the sequence $\cdots, w_{n-1}, w_n, w_{n+1}, \cdots$, where

$$w_n = \frac{v'_n - v'_{n-1}}{v'_{n-1} + \epsilon} \tag{3}$$

and ϵ is a small constant.

B. Weighted Signed Error (WSE) Loss Function

Mean Squared Error (MSE) [21] is a commonly used loss function in the last fully connected layer of neural networks. For example, the LSTM model in Fig. 3 has an input layer, one LSTM layer, and an output layer in which MSE is aimed to be minimized.

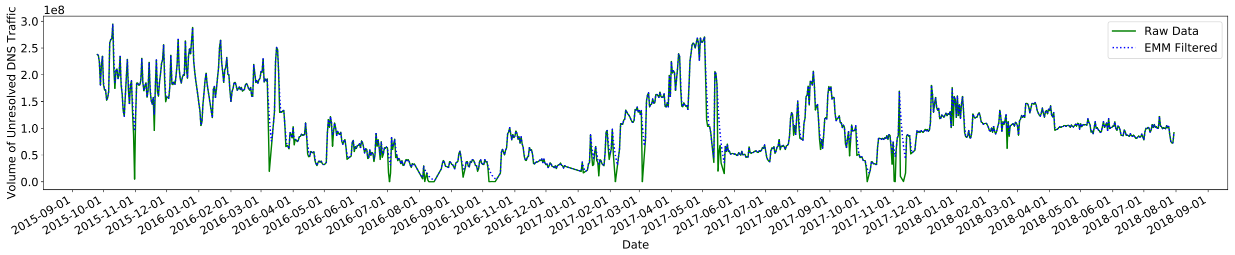

Fig. 2: Raw data vs. EMM filtered data on DNS traffic data. The EMM filter window size is set to 3.

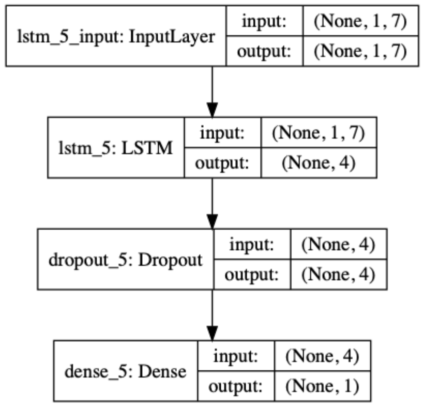

Fig. 3: An example of LSTM model

Under provisioned resources can lead to disastrous consequences for network servers like network failure or a risk of security vulnerability. In another word, under-prediction of traffic volume is more serious than over-prediction. However, MSE treats the empirical errors of under and over prediction equivalently as

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(w_i^p - w_i)^2 \tag{4}$$

where w_i is the observed volume, w_i^p is the predicted volume, and N is the total number of predicted volumes.

To penalize under-prediction more heavily than over-prediction, we propose a weighted loss function that is cost-adaptive. Specifically, we propose a Weighted Signed Error (WSE) as

$$WSE = \frac{1}{N}\sum_{i=1}^{N}\alpha^{\frac{1+sign(w_i^p - w_i)}{2}}(w_i^p - w_i)^2 \tag{5}$$

where $\alpha \in [0, 1]$ is a weight coefficient and $sign(w_i^p - w_i) = 1$ if $w_i^p \geq w_i$, -1 otherwise. Apparently, if α is set to 1, WSE is equivalent to MSE. Moreover, WSE penalizes the under-prediction more heavily than over-prediction. For example, when $\alpha = 0.1$, the weight for under-prediction is 1 while that for over-prediction is 0.1.

IV. Experiments

To demonstrate our proposed framework for peak volume prediction, we apply it to a real DNS traffic data. The data contains a univariate time-series of the daily aggregated volume of unresolved queries on a real DNS over about three years (i.e., from Sep. 24th, 2015 to July 30th, 2018), where missing values were filled with interpolation. Therefore, this DNS traffic data contains 1,040 instances, where each instance is a daily volume of unresolved queries over the real DNS server. The whole time-series is shown as “Raw Data” with green real line in Fig. 2.

The main statistics of the DNS traffic data are summarized in Table I. Specifically, the number of daily queries varies a lot over the time. For example, the minimum volume over the three years is only 46, while the maximum reaches up to around 300 millions. Therefore, allocating a fixed amount of resource all the time for this DNS server is not realistic. For example, if we give the capacity of 300 millions to this server all the time, we waste a lot of resources in many days that can be used by other high demanding servers. Therefore, it is crucial to accurately predict the volumes including the peak values.

TABLE I: A Summary of the DNS Traffic Data

Domain Type	**Mean**	**Min**	**Median**	**Max**
Unresolved	100,498,447	46	94,268,425	294,870,823

A. Evaluation Metric

Metrics of interest are root mean squared error (RMSE) and root weighted signed error (RWSE) defined as follows.

$$RMSE = \sqrt{MSE} \tag{6}$$

$$RWSE = \sqrt{WSE} \tag{7}$$

RMSE is a standard evaluation metric used for time-series forecasting. However, RMSE is not designed for the prediction of peak values, as it assigns equal weight to over and under prediction. Thus, we propose a novel metric RWSE according to our novel loss function WSE to evaluate how the models perform on peak volume predictions.

B. Setup

We embed our cost-adaptive loss function WSE into several effective neural networks for time-series forecasting as follows.

1) ANN [22]: The ANN model constructed consists of an input layer, one hidden layer, and an output layer with a linear activation function.
2) LSTM [23]: The LSTM model is constructed by an input layer, one LSTM layer, and an output layer with a tanh activation function as in Fig. 3.
3) GRU [24]: GRU is a type of recurrent neural network as LSTM. However, GRU models are less complex due to a reduction in inputs [9]. The constructed GRU model has an input layer, one GRU layer, and an output layer with a tanh activation function.
4) MCCNN [25]: MCCNN is multi-channel CNN. We construct a model consisting of one input layer that feeds into three separate channels. Each of these channels varies in their filter size, 3, 5, and 7, respectively. Then, the network is followed by a convolutional layer, a max pooling layer and a flattening layer. The patterns learned by each of the three channels are merged together in a concatenate layer followed by a fully connected layer and an output layer with a linear activation function.

We tune the parameters and evaluate the models using 5-fold cross validation. Specifically, with a set step size k, we can use the time series to produce $N-k$ instances. Then, we split all these instances into 5 almost equal-sized folds, where each fold can be used to test and the rest four are used for training. The average performance of these 5 trials with the standard deviation are reported and compared.

For the EMM filter, we tune the window size ω from $\{3,5,7\}$ to capture the weekly seasonality, where 3 days is half of a week, 5 days is a complete work week, and 7 days is a full week. We find that $\omega = 3$ achieves the best performance. We also tune our inheritance exponent β with values $\{0.1, 0.5, 0.7, 0.9\}$ and find the best performance with $\beta = 0.5$.

For the local normalization, we set the constant $\epsilon = 0$ since the minimum volume is not zero as shown in Table I. Considering that LSTM and GRU models are using the tanh activation function, we re-scale the time-series into the range of $[-1, 1]$ specifically for these two models.

Finally, we tune the step size k from the values $\{3,5,7\}$ to similarly consider the weekly seasonality. We find that $k = 7$ results in the best performance. We also tune the weight coefficient α of our loss function from $\{0.1, 0.2, 0.5\}$ and find the best performance with $\alpha = 0.2$. This means setting the weight of under-prediction 20x higher than over-prediction provides the best performance.

C. Performance Comparison

To demonstrate the helpfulness of each component in our framework, that is, EMM filter, local normalization, and the proposed loss function WSE. We compare the performance of the following methods.

TABLE II: Performance Comparison on DNS Traffic Data (The best performance is in bold where statistical significance test with 95% confidence is used).

Model	RWSE	RMSE
Linear Models		
ARIMA_G	0.275±0.141	0.502±0.338
ARIMA_L	0.170±0.063	0.222±0.080
LR_G	0.236±0.114	0.407±0.273
LR_L	0.169±0.063	0.221±0.081
EMM_LR_G	0.228±0.114	0.422±0.284
EMM_LR_L	0.165±0.060	0.239±0.089
RNN Models		
LSTM_G_WSE	0.622±0.149	1.384±0.329
LSTM_G_MSE	0.382±0.087	0.841±0.187
LSTM_L_WSE	27.024±11.168	60.428±24.971
LSTM_L_MSE	23.141±9.204	51.745±20.581
EMM_LSTM_G_WSE	0.639±0.150	1.427±0.333
EMM_LSTM_G_MSE	0.394±0.088	0.873±0.190
EMM_LSTM_L_WSE	0.171±0.060	0.329±0.114
EMM_LSTM_L_MSE	0.159±0.062	0.240±0.092
GRU_G_WSE	0.618±0.148	1.376±0.326
GRU_G_MSE	0.380±0.085	0.835±0.182
GRU_L_WSE	25.886±11.386	57.883±25.459
GRU_L_MSE	22.723±9.429	50.810±21.083
EMM_GRU_G_WSE	0.636±0.149	1.419±0.330
EMM_GRU_G_MSE	0.391±0.086	0.867±0.186
EMM_GRU_L_WSE	0.172±0.060	0.334±0.115
EMM_GRU_L_MSE	0.160±0.060	0.240±0.090
ANN and CNN Models		
ANN_G_WSE	0.181±0.058	0.333±0.093
ANN_G_MSE	0.173±0.067	0.250±0.114
ANN_L_WSE	0.167±0.062	0.222±0.081
ANN_L_MSE	0.169±0.063	**0.220±0.080**
EMM_ANN_G_WSE	0.180±0.058	0.352±0.102
EMM_ANN_G_MSE	0.166±0.067	0.264±0.125
EMM_ANN_L_WSE	**0.153±0.059**	0.246±0.092
EMM_ANN_L_MSE	0.159±0.061	0.231±0.089
MCCNN_G_WSE	0.202±0.122	0.339±0.298
MCCNN_G_MSE	0.188±0.095	0.303±0.228
MCCNN_L_WSE	0.169±0.064	0.222±0.082
MCCNN_L_MSE	0.170±0.064	0.221±0.081
EMM_MCCNN_G_WSE	0.199±0.127	0.356±0.307
EMM_MCCNN_G_MSE	0.183±0.096	0.317±0.232
EMM_MCCNN_L_WSE	0.157±0.061	0.251±0.095
EMM_MCCNN_L_MSE	0.163±0.063	0.236±0.092

1) ARIMA [26] with two different normalization methods, global (ARIMA_G) and local (ARIMA_L), respectively. The best parameters are used. Specifically, the data is differenced once for ARIMA in order to achieve stationarity.
2) Linear Regression (LR) [27] with or without EMM filter followed by different normalization methods, which includes LR_G, LR_L, EMM_LR_G, and EMM_LR_L.
3) Neural networks with or without EMM filter followed by different normalization methods (i.e., G and L) and different loss functions (i.e., MSE and WSE), which are in the format of <FILTER>_<MODEL>_<NORMALIZATION>_<LOSS>. For example, an ANN model that applies EMM filtering followed by local normalization with our custom loss function WSE is represented as EMM_ANN_L_WSE.

Table II summarizes the empirical results of different methods under the two interested evaluation metrics, i.e., RWSE and RMSE. The best performance for each evaluation metric is in bold (significance test with 95% confidence applied).

From the results, we can observe that RNN models are outliers in terms of performance. Potential factors can be that RNN models cannot capture the DNS traffic pattern well or the data size is not sufficient for RNN models to train strong enough models. Based on the results of linear models and other neural networks, we can have a few major observations.

1) Local normalization improves both RMSE and RWSE: Local normalization improves general prediction (RMSE) and peak prediction (RWSE) for all models. This demonstrates that local normalization is able to capture the DNS traffic pattern much better than global normalization, which can also be observed in Fig. 4. Specifically, to compare the performance between local and global normalization, we fix the model as EMM_ANN_WSE and randomly choose one of the five folds as test to show the difference between the observed volumes (i.e., green dashed line) and the predicted volumes (i.e., blue solid line). It shows that local normalization is a tighter fit to each data point. Global normalization is based on the mean and standard deviation of the whole time-series, and thus it is less accurate for each individual instance in the series since there is such a large range present in the data.

2) EMM filter improves RWSE but worsens RMSE: It can be observed that EMM filter significantly improves RWSE for all models, but worsens RMSE. RWSE is smaller when the model generates over-predictions even if the over-prediction is further from the predicted value, which results in an increase to the RMSE. Therefore, EMM filter is only useful for peak predictions where under-prediction is much more serious than over-prediction.

3) WSE with local normalization improves RWSE: WSE improves peak prediction performance on local normalized data for ANN and MCCNN, regardless if EMM filter is applied. The positive effects of the WSE coupled with local normalization can be observed in the example of Fig. 5. Similarly, we fix the model as EMM_ANN_L and compare the performance of using different loss functions, MSE and WSE, respectively. The areas within the red rectangles show that MSE has the problem of under-prediction due to the delayed the prediction, while WSE helps anticipate a peak value with less delay and has no under-prediction at all. This is significant to our application, where we would like to anticipate peak values with as little delay as possible. This demonstrates that our proposed loss function is better able to anticipate peak values than the standard MSE loss function.

4) ANN with best RMSE and RWSE performance: The ANN model achieves the best performance for both general DNS traffic forecasting and peak volume prediction as shown in Table II. It further demonstrates our proposed framework combined with ANN. However, we have the doubt if this is because of the size limit of the DNS data similar to the doubt on the performance of RNN models, which will be our future work.

V. Conclusion

Aiming to provide a stable connection service for IoT that consists of billions of devices, we propose a new framework for peak volume prediction for network servers. We show that an EMM filter with a cost-adaptive loss function deployed in certain neural networks can help improve the peak volume prediction over a real DNS traffic data.

Compared to DNS data, the Dynamic Host Configuration Protocol (DHCP) or the Wireless Sensor Network (WSN) are also important connection infrastructures for IoT. We will explore the proposed framework on these specific applications in the future.

References

[1] R. K. Chang, "Defending against flooding-based distributed denial-of-service attacks: a tutorial," *IEEE communications magazine*, vol. 40, no. 10, pp. 42–51, 2002.

[2] G. Kambourakis, T. Moschos, D. Geneiatakis, and S. Gritzalis, "Detecting DNS amplification attacks," in *Critical Information Infrastructures Security*, J. Lopez and B. M. Hämmerli, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 185–196.

[3] M. Anagnostopoulos, G. Kambourakis, P. Kopanos, G. Louloudakis, and S. Gritzalis, "DNS amplification attack revisited," *Computers & Security*, vol. 39, pp. 475–485, 2013.

[4] B. Zhou, D. He, Z. Sun, and W. H. Ng, "Network traffic modeling and prediction with ARIMA/GARCH," in *Proc. of HET-NETs Conference*, 2005, pp. 1–10.

[5] H. Z. Moayedi and M. Masnadi-Shirazi, "ARIMA model for network traffic prediction and anomaly detection," in *Proc. of IEEE International Symposium on Information Technology*, vol. 4, 2008, pp. 1–6.

[6] W. H. K. Tsui, H. O. Balli, and H. Gower, "Forecasting airport passenger traffic: the case of hong kong international airport," *Education and Research Proceedings*, vol. 2011, pp. 54–62, 2011.

[7] S. Mehrmolaei and M. R. Keyvanpourr, "A brief survey on event prediction methods in time series," in *Artificial Intelligence Perspectives and Applications*, R. Silhavy, R. Senkerik, Z. K. Oplatkova, Z. Prokopova, and P. Silhavy, Eds. Springer, 2015, pp. 235–246.

[8] G. Mahalakshmi, S. Sridevi, and S. Rajaram, "A survey on forecasting of time series data," in *Proc. of IEEE International Conference on Computing Technologies and Intelligent Data Engineering*, 2016, pp. 1–8.

[9] R. Fu, Z. Zhang, and L. Li, "Using LSTM and GRU neural network methods for traffic flow prediction," in *Proc. of The 31st IEEE Youth Academic Annual Conference of Chinese Association of Automation*, 2016, pp. 324–328.

[10] A. Mozo, B. Ordozgoiti, and S. Gómez-Canaval, "Forecasting short-term data center network traffic load with convolutional neural networks," *PloS One*, vol. 13, no. 2, p. e0191939, 2018.

[11] A. Azzouni and G. Pujolle, "NeuTM: A neural network-based framework for traffic matrix prediction in SDN," in *Proc. of 2018 IEEE/IFIP Network Operations and Management Symposium*, 2018, pp. 1–5.

[12] B. Yu, L. Smith, and M. Threefoot, "Exponential moving maximum filter for predictive analytics in network reporting," in *Proc. of The 5th International Conference on Advances in Information Mining and Management*, 2015, pp. 27–32.

[13] ——, "Exponential moving maximum (EMM) filter for predictive analytics in network reporting," Patent, Jul. 3, 2018, US Patent App. 10/015,059.

[14] C.-L. Hor, S. J. Watson, and S. Majithia, "Daily load forecasting and maximum demand estimation using ARIMA and GARCH," in *Proc. of IEEE International Conference on Probabilistic Methods Applied to Power Systems*, 2006, pp. 1–6.

[15] N. Amjady, "Short-term hourly load forecasting using time-series modeling with peak load estimation capability," *IEEE Transactions on Power Systems*, vol. 16, no. 3, pp. 498–505, 2001.

[16] G. P. Zhang, B. E. Patuwo, and M. Y. Hu, "A simulation study of artificial neural networks for nonlinear time-series forecasting," *Computers & Operations Research*, vol. 28, no. 4, pp. 381–396, 2001.

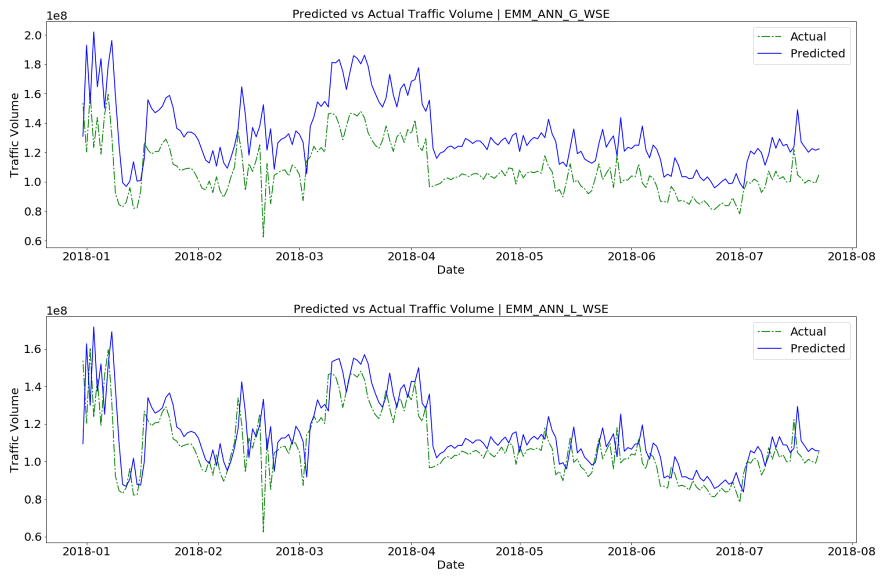

Fig. 4: Global (top) vs. local (bottom) normalization using the model EMM_ANN_WSE.

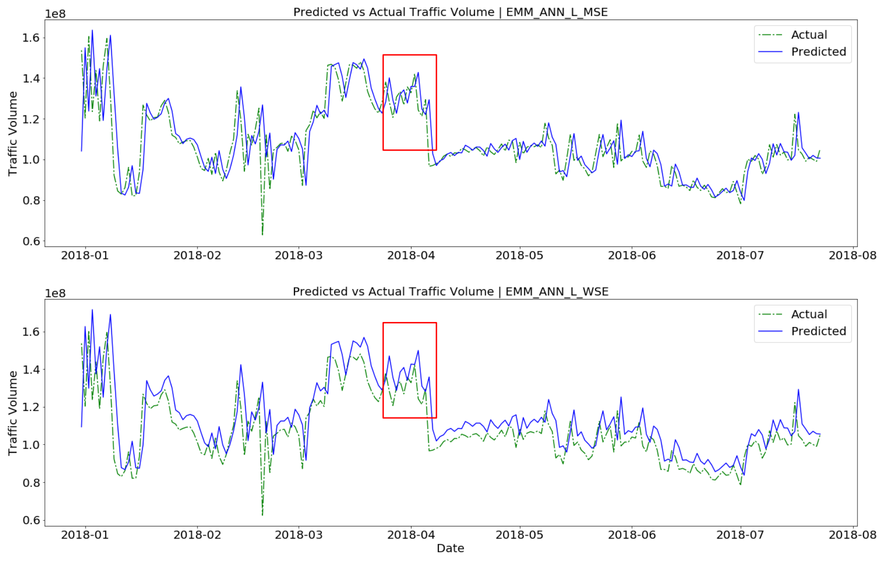

Fig. 5: MSE (top) vs. WSE (bottom) on EMM_ANN_L (red rectangle areas show less delay and no under-prediction for WSE).

[17] Z. I. E. Cicek and Z. K. Ozturk, “Short term traffic flow forecasting using artificial neural networks,” *Sigma*, vol. 9, no. 4, pp. 405–414, 2018.

[18] Y. Hua, Z. Zhao, R. Li, X. Chen, Z. Liu, and H. Zhang, “Traffic prediction based on random connectivity in deep learning with long short-term memory,” in *Proc. of IEEE 88th Vehicular Technology Conference*, 2018, pp. 1–6.

[19] R. Vinayakumar, K. Soman, and P. Poornachandran, “Applying deep learning approaches for network traffic prediction,” in *Proc. of IEEE International Conference on Advances in Computing, Communications and Informatics*, 2017, pp. 2353–2358.

[20] M. J. T. Bantugon and R. J. C. Gallano, “Short- and long-term electricity load forecasting using classical and neural network based approach: A case study for the philippines,” in *Proc. of IEEE Region 10 Conference*, 2016, pp. 3822–3825.

[21] E. Lehmann and G. Casella, *Theory of Point Estimation*. Springer Verlag, 1998.

[22] F. Rosenblatt, “The perceptron: a probabilistic model for information storage and organization in the brain,” *Psychological Review*, vol. 65, no. 6, p. 386, 1958.

[23] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[24] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using RNN encoder-decoder for statistical machine translation,” in *Proc. of the Conference on Empirical Methods in Natural Language Processing*, 2014, pp. 1724–1734.

[25] H. Cecotti and A. Graser, “Convolutional neural networks for P300 detection with application to brain-computer interfaces,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 3, pp. 433–445, 2010.

[26] C. Chatfield and H. Xing, *The Analysis of Time Series: An Introduction with R*, 7th ed. Chapman and Hall/CRC, 2019.

[27] J. Neter, W. Wasserman, and M. H. Kutner, *Applied Linear Regression Models*. Irwin Homewood, IL, 1989.