Deep Learning For Network Traffic Prediction

Literature Review

Justin Myerson
University of Cape Town
Computer Science Honours
myrjus002@myuct.ac.za

Abstract

Network Traffic Prediction is an important tool for managing network congestion and security. It aims to predict future network traffic flows based on previous data, using either time-series or machine learning approaches. Efficient prediction can improve quality of service and lower operating costs for network service providers, and can be used to prevent Distributed Denial of Service attacks too. Multiple studies have been conducted comparing these approaches, with the results suggesting that Deep Learning techniques learn network traffic patterns more efficiently and provide more accurate results. Of these approaches, Long Short Term Memory appears to be the most accurate method. However, there is little mention of the computational resources required to run these algorithms, and whether they are optimised for large networks. Further research to determine whether these approaches are viable for less resourced systems and whether they perform well on less intensive network traffic is required.

CCS Concepts: • **Computing methodologies** → **Unsupervised learning**; **Neural networks**; • **Networks** → **Network performance analysis**.

Keywords: Autoregressive Integrated Moving Average, Gated Recurrent Unit, Long-Short Term Memory, Mean Squared Error, Network Traffic Prediction, Recurrent Neural Network, Time Series

1 Introduction

In today's world, the Internet and its applications have become a vital tool for all types of users. As more individuals are gaining access to the internet for the first time, and others increasing their usage, networks are having to manage their limited bandwidth effectively. Covid-19 especially, has caused a significant surge in data traffic [3]. Part of this traffic was caused by the move to online-learning, and shift from office to work from home. Network Traffic Prediction would help alleviate any congestion issues which may arise, as network operators would be better informed of when increased congestion can be expected and can then make adjustments accordingly. Predicting network traffic in the short term aids in dynamic resource allocation, while longer term prediction provides insight into how a service provider may improve their network capacity and performance [16].

This paper investigates the different approaches that have been used for predicting Network Traffic. There will be mention of preprocessing of network traffic data as a way to improve prediction results by means of wavelet transformation and k-means clustering. We explore the Time Series approach of Autoregressive Integrated Moving Averages (ARIMA), Deep Learning techniques such as Recurrent Neural Networks and their subsets: Long Short Term Memory, Gated Recurrent Units and Stacked Autoencoders. Previous work done in this field suggests these are the most effective methods for network traffic prediction.

Recurrent Neural Networks (RNNs) are particularly powerful models that have demonstrated high accuracy in time series forecasting [21], something ARIMA cannot model [22].

2 Background

2.1 Network Traffic Data

Network traffic refers to the data, made up of packets, moving across devices at any point in time. It can be hard to deduce and label these patterns by just monitoring traffic data. Trends on a particular day do not necessarily correlate to the next day, and seemingly arbitrary spikes of traffic need not be confined to particular time periods. Past studies [25, 27] have used network traffic data from two public datasets: GEANT and Abilene, in order to train and test prediction models.

3 Data Preprocessing

Data preprocessing for network traffic data involves extracting the salient features from the dataset.

3.1 K-Means Clustering

A method for grouping data together into K sets [15], which works by selecting K initial cluster centres. It iteratively refines the clusters by assigning each element to the closest cluster, with each cluster center updated to be the mean of its associated elements.

3.2 Wavelet Transformation

Wavelet Transform is a technique that allows for data of differing frequencies to be processed, while retaining localization information [23]. This is especially useful for network traffic data, since it is measured over multiple time scales

and can be reconstructed into high and low-frequency wavelengths.

3.3 Combining Data Preprocessing

Zang et al. [28] applied a K-means clustering algorithm to group network base stations into groups where base stations in one group are geographically adjacent with correlated traffic flows. Wavelet decomposition was additionally used to preprocess the time series traffic data by decomposing it into low and high-frequency components. The final step of preprocessing was to reconstruct both the low and high-frequency components back into time series by wavelet reconstruction. Their results suggested that running the data through an Elman Neural Network - defined as a simple recurrent neural network - yielded superior results compared to time series methods. However, they did not test their method on any other deep learning approaches which may have yielded more accurate prediction results.
Chen et al. [5] took a different approach, by applying chaotic analysis to network prediction data. They looked at the prediction error in order to feed the analysis generated into a neural network, which eventually was used to determine whether a network was undergoing a Distributed Denial of Service attack. This could potentially be an area to explore if network security is a priority and has implications for network service providers.

4 Time Series Approaches to Prediction

ARIMA is a statistical model that is used for time series forecasting. It is derived from the notion that future values can be predicted by using past values and white noise characteristics [2]. Time series are successful in predicting a few time-steps ahead, but the accuracy depends on data exhibiting seasonality [13]. Network spikes are difficult to anticipate and methods deviate significantly on volatile real statistics. Statistical techniques also fail to learn long-range dependency [13].

Holt-Winters [4] is another time series approach that is well suited to producing short term forecasts by using exponential smoothing. Jirsik et al. [11] compared network traffic prediction using both ARIMA and Holt-Winters, and observed that ARIMA produced more accurate predictions. However, Holt-Winters was far less computationally expensive.

Fractionally integrated moving average model (FARIMA) [6] is a long-range dependent model that is an extension of the ARIMA model, which differs by allowing a non-integer value of the difference parameter. Just like ARIMA, the computational complexity of FARIMA is $O(N^2)$. By comparing FARIMA and ARIMA models to predict network traffic, it was shown by Feng and Shu [6] that FARIMA was more accurate since it is able to capture both short and long-term dependencies in the data.

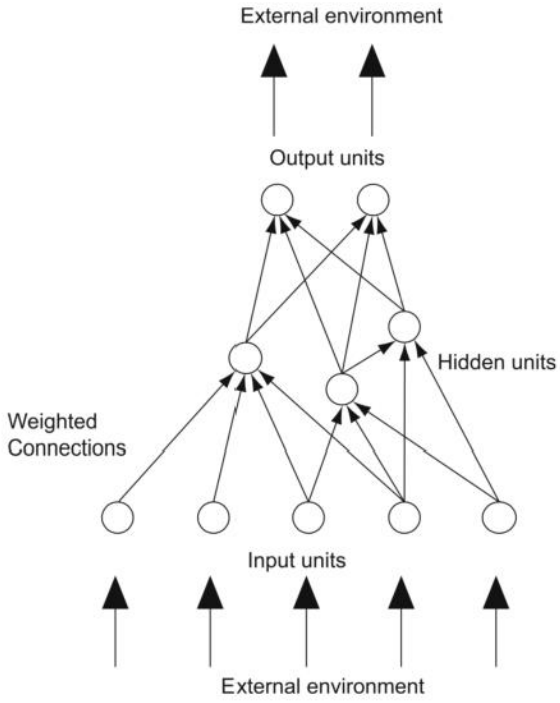

Figure 1. Example of a neural network architecture. It consists of input and output units which are connected to the external environment and a hidden layer comprising of nodes that connect to one another [8]

5 Neural Networks

Neural Networks (NN) are a subset of Machine Learning, inspired by how neurons in the human brain communicate. Their structure is comprised of an input, output and either one or more hidden layers and nodes that connect to one another. These nodes have attributes, a weight and threshold that when a node exceeds, passes along data to the next layer [8]. These networks are trained using data in order to optimize their performance and improve their accuracy, which is a measure of how close the predicted value is to the expected value. In general, machine learning models are evaluated on their accuracy, measured in Mean Squared Error.

$\text{MSE} = \frac{1}{n}\sum_{t=1}^{n} e_t^2$ where **e** is the error, or the observed value subtracted from the actual value.

Feng and Shu [6] used both FARIMA and a NN to predict network traffic, finding that there was very little difference in prediction accuracy, with FARIMA having the slight edge.

5.1 Recurrent Neural Network

A Recurrent Neural Network (RNN) is an extension of the generic Neural Network which is specialised for processing sequential data. The main difference being the ability of outputs from layers being able to cycle back into the network. Here the back-propagation algorithm is employed to factor in dependencies among the data, creating an internal memory that facilitate learning of time series data [8]. An RNN can suffer from the vanishing gradient problem, which occurs when the network is unable to send back useful gradient information from the output layers to those layers that are more shallow. If this occurs, the RNN loses its ability to consider long term dependencies in calculations [1]. It is for this reason that Krishnaswamy et al. [12] propose that using an RNN is unsuitable for network traffic prediction, suggesting rather that a Long Short Term Memory should be used rather to mitigate the vanishing gradient.

Ramakrishnan and Soni [22] broke down the network traffic into smaller time intervals of 5 seconds, which may be more indicative of when large changes in network traffic flows could occur (e.g. Concert tickets being released). In order to train their RNNs they used a sliding window, meaning previous traffic flow values were used as features to predict the next **n** traffic flows. Part of the data that was used was traffic collected from internal requests between two virtual machines, while the other half was from public data sets. The former is potentially useful, since some smaller networks will see far less traffic and therefore may be better suited to different algorithms. Their results suggested that using an RNN approach was twice as accurate as ARIMA on smaller networks. The implication of this finding is that if the RNN is not too computationally expensive, it would be well suited for this situation. The need is therefore to develop and test the RNN on different hardware setups and see what the minimally viable solution could be for less resourced networks.

Vinayakumar et al. [27] ran their RNN experiments on a Graphical Processing Unit (GPU), but did note that shallow networks were run on Scikit-learn. Therefore it may not be necessary for smaller networks to train their model on a GPU, but this needs to be investigated further. However, the data show that RNNs are much quicker to train than Stacked Auto Encoders or Multilayer Perceptrons, also using fewer layers and neurons [20]. One further point they make is that it may be beneficial to train the models on more powerful hardware, such as a High Performance Computer.

Of the approaches discussed, none had taken into account the timestamp and day of the week that the traffic flows were observed. This is an important feature to consider since network traffic flows vary depending on the time of day and week. Typically, peak usage occurs during the early evening and the least usage is observed during the period of midnight until 5am [17]. Tokuyama et al. [24] looked at using one-hot encoding - each input data encoded to a 0-1 vector - to determine whether they could improve on prediction accuracy. Two new input layers were added and merged with the existing RNN they had built. This approach is an important step to adding context to the prediction, as often RNN models misunderstand the relationship between data [24]. Their findings are of particular importance as it illustrates that adding computational complexity is not a silver bullet, as the one-hot encoding did not make a noticeable difference in the observed RMSEs.

5.1.1 Long Short Term Memory.

An LSTM is a type of RNN, which is formed by adding a short and long term memory unit to a RNN [9]. The addition of memory units allows the network to deal with the correlation of time series in the short and long term, and store dependencies that it deems important from earlier epochs of training [29]. Additionally, in order to control the use of historical information, the model uses an in, forgetting and out gate. Once an LSTM is trained, the predicted output relies on the input it receives, and short and long term states that are fed back to it. In order to minimise the Mean Squared Error, the LSTM needs to be trained with optimal parameters.

Jirsik et al. [11] ran an experiment comparing LSTM, ARIMA and Holt-Winters time series approach. The time complexity observed indicated that both ARIMA and LSTM took in order of magnitude ten times longer to compute compared to Holt-Winters for both time periods predicted. ARIMA was faster to compute predictions for 1 hour into the future, while LSTM was faster for 5 minute intervals. They suggested that this time complexity could be reduced by changing the optimizer used in the training and changing the initial weights for the LSTM. This has important implications as it demonstrates that there is a trade off between complexity and accuracy, since Holt-Winters was the least accurate at predicting network traffic. Further findings from Vinayakumar et al. [27] demonstrated that LSTM has more computational complexity than Gated Recurrent Units, but outperformed RNNs by producing more accurate predictions.

Krishnaswamy et al. [12] discussed the differences between using an Simple and Stacked LSTM learning approach. Both have advantages, with Stacked being more accurate by adding more LSTM layers at the cost of higher computational complexity. There is a a trade off here, but it allows network operators to decide what is more practical for their needs, accuracy or computational efficiency. Importantly in their training, they noted that adding extra layers did not cause a noticeable change in accuracy. In fact, the Simple LSTM which requires the least resources had the lowest Mean Squared Error overall, and **tanh** proved to be the best

activation function. There is no mention of whether the training took place on a GPU, so it is important to test these methods with different activation functions to determine if there is a more accurate approach, and only on a CPU since not all network providers will have access to a GPU. One limitation that needs further investigation is whether these LSTM algorithms perform as well for smaller traffic volumes that one may see on a community network, as the results above were for links of capacity of over 100GB/s.

Another approach that was taken and yielded an improvement over LSTM was to create a parallel LSTM architecture. This was developed by [14] as they noted that prediction of network traffic does a poor job of detecting bursts of network traffic. Their proposed architecture comprised of one LSTM which used actual network traffic data, and a second LSTM which used both the network traffic data and a burst pulse series. Results show that the parallel LSTM improved prediction accuracy on average by 37 percent. This could have important implications for emergency networks, when ensuring the network can cope with a burst of traffic is vital in the event of an emergency.

5.1.2 Gated Recurrent Unit.

The Gated Recurrent Unit (GRU) is an improvement based on LSTM [10]. It is able to factor in dependencies based on different time scales, and unlike LSTM, it does not have separate memory units. Any feature that is deemed important by the GRU is maintained, not overwritten [25].

Troia et al. [25] demonstrated a novel approach to the network traffic prediction problem, by combining a GRU and Evaluation Automatic Module (EAM). The GRU was used to train the prediction model, while the EAM evaluated performance based on the prediction error during each iteration. By comparing the error in each iteration, it can choose the lowest error while allowing the model to keep training as well in case a better approximation is found. Their model was used to predict traffic for the next hour, which is a large timescale for network traffic, since fluctuations can happen in much shorter time periods. The results showed that the predicted traffic values were far below the threshold the network operators had anticipated, and shows that this implementation could save resources since there is a large surplus of bandwidth availability.

5.2 Stacked Autoencoder

An autoencoder is an unsupervised deep learning algorithm. It takes in a number of inputs and returns the same number of outputs, aiming to mimic the input by compressing it and applying some activation function [18]. This process helps the network to learn the most salient features of the data. In a stacked autoencoder (SAE), each output layer is connected to the input of the next layer [20]. These models have been used in the past for network security, focusing on network intrusion detection [26].

Oliveira et al. [20] noted that SAEs were the longest model to train since they add additional complexity, when compared to RNNs. Even with this complexity, they did not produce better results than a RNN. In an experiment, Oliveira et al. [19] found that even a Multilayer Perceptron (MLP) which is a far more simple model had superior prediction results over all time frames. An MLP is less computationally expensive than an SAE and it does not use pre-training either. SAE as a method to predict network traffic will be investigated further, even though preliminary findings indicate that they it is not advantageous to use compared to more simple and cheaper approaches. It is possible that applying pre-processing of the network traffic data may improve prediction outcomes.

6 Discussion

Most time-series algorithms are designed to predict one time-step ahead, and predicting multiple steps results in an error build-up that is amplified as prediction goes deeper into future [12]. Discrete wavelet transform (DWT) scopes out both location and frequency information from a signal. Wavelet transform when used in forecasting models can improve the accuracy of the forecast [16].

An attempt to create a network traffic prediction model by creating a forecast vector composed of an ARIMA and RNN component was made by [16]. They used DWT to decompose a network traffic time series into a vector which allowed the data to be processed for ARIMA. Residual Sum of Squares (RSS), or the sum of squared errors of prediction was calculated in order to determine which parameters were to be used in the ARIMA model. Once both the ARIMA model had been optimised and the RNN had been trained, they combined the forecasts to try and predict network traffic. Their data was of differing time periods - per day, hour and five minutes - which ensured that all time periods are evaluated over, and came from a real world Internet Service Provider which means the data is reflective of the real world and what could be expected. Their findings indicated that at any time interval, measured in Normalised Residual Mean Squared Error (NRMSE), combining both an ARIMA and RNN approach yielded superior results as opposed to using an ARIMA approach, but not always better than just using a RNN approach. This is potentially significant since adding time series calculations increases computational cost which may not be suitable for less resourced networks. They add that this approach is potentially easy to deploy at data centers, but fail to take into account smaller network providers.

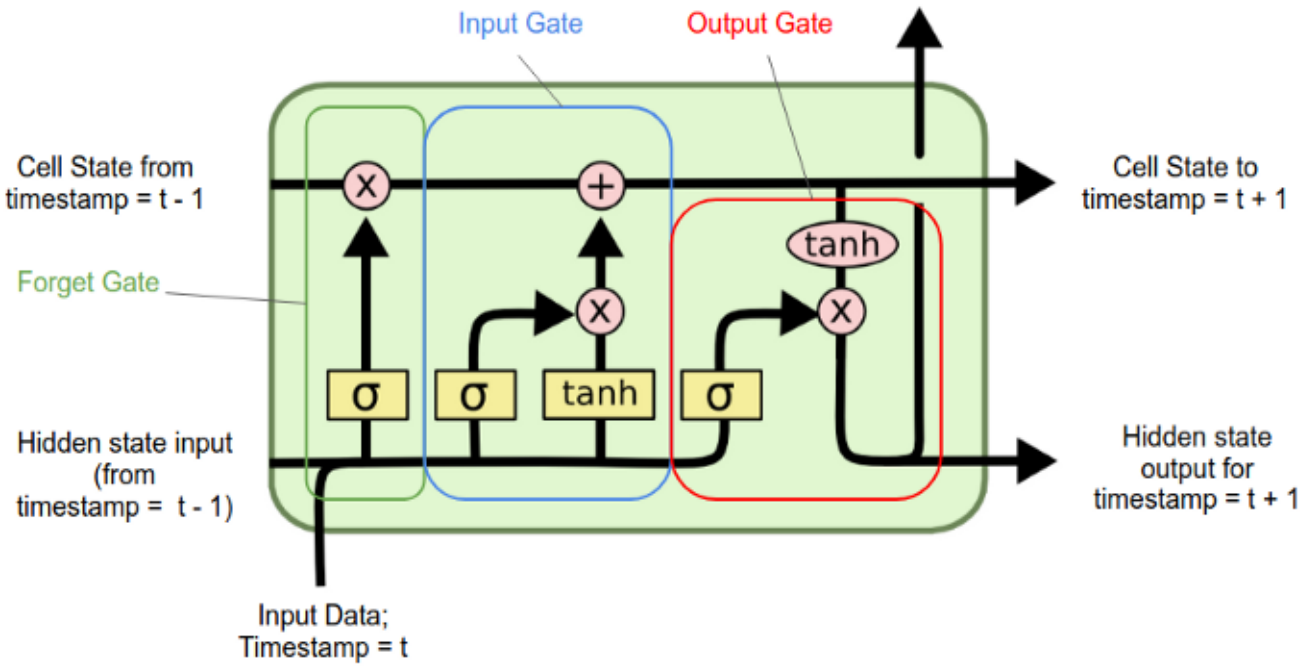

Figure 2. An example of an LSTM cell [7]. The cell takes an input of the current time step, the output from the previous LSTM and the memory from the previous unit. The cell makes a decision by considering these three inputs, generating a new output and altering its memory.

7 Conclusions

Network traffic is increasing in volume as more users are connecting to the internet, putting pressure on networks to adjust. Time Series alone is not a suitable approach to network traffic prediction since it is unable to effectively take into account spikes in network traffic [13].

Data Preprocessing has also yielded accurate predictions [5, 28], not only for network traffic predictions but also for detecting DDOS attacks, however it is unclear whether these approaches will work with other approaches besides RNNs.

Multiple approaches to predicting network traffic were investigated, with the findings suggesting that an increased computational complexity does not translate into more accurate predictions [19, 20]. This is important since some networks have access to less resources to dedicate to traffic prediction.

The findings conclude that Long Short Term Memory produces more accurate network traffic predictions than Recurrent Neural Networks [27], and that Stacked Autoencoders are not as well suited as LSTMs [19]. However, it is unclear whether a Gated Recurrent Unit is more accurate than LSTM, and more work needs to be done here. Combining it with RNN does yield superior results, but this additional computational cost may deem it only feasible for larger and more resourced networks.

References

[1] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[2] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[3] Ibrahim G. Vallisn B. Böttger, T. How the Internet reacted to Covid-19 – A perspective from Facebook's Edge Network. *Proceedings of the ACM Internet Measurement Conference*, October 2020. doi: 10.1145/3419394.3423621.

[4] Chris Chatfield and Mohammad Yar. Holt-winters forecasting: Some practical issues. *Journal of the Royal Statistical Society. Series D (The Statistician)*, 37(2):129–140, 1988. ISSN 00390526, 14679884. URL http://www.jstor.org/stable/2348687.

[5] Yonghong Chen, Xinlei Ma, and Xinya Wu. Ddos detection algorithm based on preprocessing network traffic predicted method and chaos theory. *IEEE Communications Letters*, 17(5):1052–1054, 2013.

[6] Huifang Feng and Yantai Shu. Study on network traffic prediction techniques. In *Proceedings. 2005 International Conference on Wireless Communications, Networking and Mobile Computing, 2005.*, volume 2, pages 1041–1044, 2005. doi: 10.1109/WCNM.2005.1544219.

[7] Jian Fu, Jingchun Chu, Peng Guo, and Zhenyu Chen. Condition monitoring of wind turbine gearbox bearing based on deep learning model. *IEEE Access*, 7:57078–57087, 2019. doi: 10.1109/ACCESS.2019.2912621.

[8] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

[9] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[10] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation.*, 9(8), 1997-11-01. ISSN 0899-7667.

[11] Tomas Jirsik, Štěpán Trčka, and Pavel Celeda. Quality of service forecasting with lstm neural network. In *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, pages 251–260, 2019.

[12] Nandini Krishnaswamy, Mariam Kiran, Kunal Singh, and Bashir Mohammed. Data-driven learning to predict wan network traffic. In *Proceedings of the 3rd International Workshop on Systems and Network Telemetry and Analytics*, pages 11–18, 2020.

[13] Will E Leland, Murad S Taqqu, Walter Willinger, and Daniel V Wilson. On the self-similar nature of ethernet traffic (extended version). *IEEE/ACM Transactions on networking*, 2(1):1–15, 1994.

[14] Huang Lin, Wang Diangang, Liu Xiao, Zhuo Yongning, and Zeng Yong. A predictor based on parallel lstm for burst network traffic flow. In *Proceedings of the 2020 6th International Conference on Computing and Artificial Intelligence*, ICCAI '20, page 476–480, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450377089. doi: 10.1145/3404555.3404632. URL https://doi-org.ezproxy.uct.ac.za/10.1145/3404555.3404632.

[15] J. MacQueen. *Some methods for classification and analysis of multivariate observations*, chapter Proc. the 5th Berkeley symposium on mathematical statistics and probability, pages 281–297. Berkeley, USA, 1st edition, 1967.

[16] Rishabh Madan and Partha Sarathi Mangipudi. Predicting computer network traffic: a time series forecasting approach using dwt, arima and rnn. In *2018 Eleventh International Conference on Contemporary Computing (IC3)*, pages 1–5. IEEE, 2018.

[17] Janine Morley, Kelly Widdicks, and Mike Hazas. Digitalisation, energy and data demand: The impact of internet traffic on overall and peak electricity consumption. *Energy Research & Social Science*, 38:128–137, 2018.

[18] Andrew Ng et al. Sparse autoencoder. *CS294A Lecture notes*, 72(2011): 1–19, 2011.

[19] Tiago Prado Oliveira, Jamil Salem Barbar, and Alexsandro Santos Soares. Multilayer perceptron and stacked autoencoder for internet traffic prediction. In *IFIP International Conference on Network and Parallel Computing*, pages 61–71. Springer, 2014.

[20] Tiago Prado Oliveira, Jamil Salem Barbar, and Alexsandro Santos Soares. Computer network traffic prediction: a comparison between traditional and deep learning neural networks. *International Journal of Big Data Intelligence*, 3(1):28–37, 2016.

[21] Sharat C Prasad and Piyush Prasad. Deep recurrent neural networks for time series prediction. *arXiv preprint arXiv:1407.5949*, 2014.

[22] Nipun Ramakrishnan and Tarun Soni. Network traffic prediction using recurrent neural networks. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 187–193. IEEE, 2018.

[23] Isar A. Moga S. Lenca P. Stolojescu-Crisan, C. Wimax traffic forecasting based on neural networks in wavelet domain. *Proc. the 18th Intl. Conference on Machine Learning*, June 2001.

[24] Yusuke Tokuyama, Ryo Miki, Yukinobu Fukushima, Yuya Tarutani, and Tokumi Yokohira. Performance evaluation of feature encoding methods in network traffic prediction using recurrent neural networks. ICIET 2020, page 279–283. Association for Computing Machinery, 2020. ISBN 9781450377058. doi: 10.1145/3395245.3396441.

[25] Sebastian Troia, Rodolfo Alvizu, Youduo Zhou, Guido Maier, and Achille Pattavina. Deep learning-based traffic prediction for network optimization. In *2018 20th International Conference on Transparent Optical Networks (ICTON)*, pages 1–4. IEEE, 2018.

[26] Ali Moradi Vartouni, Saeed Sedighian Kashi, and Mohammad Teshnehlab. An anomaly detection method to detect web attacks using stacked auto-encoder. In *2018 6th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS)*, pages 131–134. IEEE, 2018.

[27] R Vinayakumar, KP Soman, and Prabaharan Poornachandran. Applying deep learning approaches for network traffic prediction. In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 2353–2358. IEEE, 2017.

[28] Yunjuan Zang, Feixiang Ni, Zhiyong Feng, Shuguang Cui, and Zhi Ding. Wavelet transform processing for cellular traffic prediction in machine learning networks. In *2015 IEEE China Summit and International Conference on Signal and Information Processing (ChinaSIP)*, pages 458–462. IEEE, 2015.

[29] Zheng Zhao, Weihai Chen, Xingming Wu, Peter CY Chen, and Jingmeng Liu. Lstm network: a deep learning approach for short-term traffic forecast. *IET Intelligent Transport Systems*, 11(2):68–75, 2017.