Explainable Learning Based on Hyperparameters Optimization of Lightweight Spatiotemporal Models for Cellular Traffic Prediction

Wei-Che Chien  
National Dong Hwa University

Hsin-Hung Cho  
National Ilan University: National I-Lan University

Fan-Hsun Tseng  
National Cheng Kung University

Shih-Yeh Chen (✉ sychen@nttu.edu.tw)  
National Taitung University  https://orcid.org/0000-0002-1863-6168

Research Article

Keywords: lightweight neural network, spatiotemporal dependence model, meta-heuristic algorithm, explainable machine learning.

DOI: https://doi.org/10.21203/rs.3.rs-771447/v1

License: © This work is licensed under a Creative Commons Attribution 4.0 International License. Read Full License
Explorable Learning Based on Hyperparameters Optimization of Lightweight Spatiotemporal Models for Cellular Traffic Prediction

Wei-Che Chien ¹, Hsin-Hung Cho ², Fan-Hsun Tseng³, Shih-Yeh Chen⁴*

Abstract

The rapid development of the Internet of Things and multimedia applications has led to an exponential growth in mobile network traffic year by year. In order to meet the demand for large amounts of data transmission and solve the problem of insufficient spectrum resources, millimeter waves are adopted for 5G communication. For B5G/6G, effective use of spectrum resources is one of the key technologies for the development of mobile communications. Therefore, this study uses a lightweight neural network to predict cellular traffic based on regional data, considering the data types of temporal and spatial dependence at the same time. Furthermore, in order to optimize the prediction performance and reduce the number of parameters of the neural network, this study uses a meta-heuristic algorithm to adjust the hyperparameters and combines local and global explanations to interpret the improvement of traffic prediction. The local explanations show the adjustment results of a single hyperparameter, and global explanations show the correlation between different hyperparameters and their influence on the amount and accuracy of model parameters. The simulation results show that compared with adjustment strategies of the manual method and greedy algorithm, the proposed explainable learning method can effectively improve the accuracy of cellular traffic prediction, reduce the number of parameters and provide a reasonable explanation.

Keywords lightweight neural network, spatiotemporal dependence model, meta-heuristic algorithm, explorable machine learning.

1. Introduction

The Fifth Generation Mobile Network (5G) promotes the rapid development of applications such as the Internet of Things (IoT) (Chien et al. 2018), Mobile Augmented Reality and Virtual Reality (Erol-Kantarci et al. 2018), and so on. However, the growth of these applications has also caused an increase in the load on the cellular network. In order to meet the requirements of a large number of users, effective network resource management is one of the challenges for the development of mobile networks in the future. To achieve this goal, cellular traffic prediction has become a key technology. If we can accurately forecast the cellular traffic in the next period, we can quickly make corresponding decisions and effectively manage resources.

In the current research, prediction methods can be divided into two categories: physical methods and statistical methods (Neve et al. 1995). The physical method is to simulate the actual situation and make predictions based on the physical theory. Statistical methods are based on analyzing historical data and finding out the relationship between the data to make predictions. Generally, physical methods have relatively good accuracy in long-term predictions, while statistical methods have good effects in short-term predictions.

5G has various application scenarios, including industrial IoT and smart medical with ultra-reliable and low latency communications (Lai et al., 2019; Magsi et al., 2018); smart city for massive machine-type communications (Rao et al., 2018); extended reality (XR) with enhanced mobile broadband (Patil et al., 2019). In other words, 5G/B5G traffic changes more than 4G. Accurate short-term predictions are required.
to efficiently manage the network resource. Therefore, most studies use statistical methods to predict traffic. Statistical methods can be divided into linear methods and non-linear methods. Since there are many factors in the real environment that will affect the prediction results, non-linear methods can make predictions more effectively. Long-Short Term Memory (LSTM) has been widely used in data prediction with temporal dependence (Gers et al. 2000). However, LSTM is a recursive neural network that is difficult to implement parallel operations. So it takes a lot of time to train the model.

5G will form an Ultra-Dense Networks (UDN) because of the use of millimeter waves for communication (Kamel et al., 2016). Therefore, regional predictions become crucial. The effect of regional prediction based on the Recurrent neural network (RNN) method is very poor because each base station needs to retrain the model based on historical data, resulting in a very large amount of computational resources. In order to solve these problems, Convolution Neural Network (CNN) is adopted to predict regional traffic (Zhang et al. 2019). CNN can perform convolution operations on high-dimensional data and learn the spatiotemporal dependence characteristics of regional base stations. In addition, the convolution operation can use a Graphics Processing Unit (GPU) to perform fast parallel operations.

The architecture of CNN is complex and requires a lot of time to train the model. Therefore, this research uses a lightweight neural network architecture to predict cellular traffic. With the same prediction accuracy, the number of parameters of the model can be reduced and the computing performance can be improved. Since lightweight neural networks also have many hyperparameters like other neural networks, the adjustment of these parameters will greatly affect the accuracy of prediction and quantity of parameters. In order to solve this problem, the contribution of this research is to propose an optimized hyperparameter strategy based on the simulated annealing algorithm for a lightweight neural network, thereby reducing the deployment cost of operators. Furthermore, although most methods can improve the accuracy of network traffic prediction or reduce the number of model parameters, it is difficult to explain the reasons for the improvement. Explainable learning can help researchers to better understand the interaction between features, and design appropriate AI models for different problems. (Xu et al. 2019) Therefore, this study focuses on an explainable learning method and discusses local and global explanations for traffic prediction.

The sections summary of this study is as follows.

In Section 2, the background knowledge of this study and related research papers are briefly described, analyzed and discussed. In Section 3, hyperparameters optimization method will be introduced. The analysis of the experimental results will be presented in Section 4. Finally, the conclusion will be discussed and future developments will be proposed.

2. Related works

There are many studies that propose different models to solve the problem of temporal and spatial dependence. We can divide the methods into single models and hybrid models.

In a single model, Densely Connected Convolutional Network (Densenet) is used to predict cellular traffic (Zhang et al., 2018). Densnet is a tightly connected CNN that can model the traffic of different base stations and learn features of spatiotemporal dependence from different base stations through convolution layers. In this model, the features learned in the previous layer are merged with the features in the next layer. Therefore, important features can be retained. Qiu et al., 2018 use recurrent neural networks and multi-task learning methods to explore the similarities and differences in traffic between base stations. The proposed method does not use traffic information of neighboring base stations but uses base stations with similar service information to learn the spatiotemporal dependence features. Shi et al., 2015 use Convolutional Long-Short Term Memory Network (ConvLSTM) to predict data with spatiotemporal dependence. Although this method can learn spatiotemporal dependence features well and obtain accurate prediction results, the training process requires a lot of computational resources.

In the hybrid model, Lv et. Al, 2015 proposed stacked Autoencoder (AE) to learn the data features of different regions and used regression for data prediction. Wang et al., 2017 proposed a hybrid architecture of AE and LSTM. LSTM is used to extract temporal features and AE is used to extract spatial features between different base stations. AE is divided into Global Stack Autoencoder (GSAE) and Local Stack Autoencoder (LSAE), which extract detailed spatiotemporal features through global search and local search. However, this method will compress data while extracting spatial features, which is easy to destroy spatiotemporal features of the adjacent base stations. Wu et al, 2016 proposed a hybrid model combined with one-dimensional CNN and LSTM. The one-dimensional CNN is used to extract the spatial features and LSTM is used to short-term changes and periodic features of
traffic. Zhang et al., 2019 proposed a Spatial-Temporal Cross-Domain Neural Network (STCNET) architecture to predict traffic. Densenet is used to extract regional traffic features of base stations and ConvLSTM is used to extract cross-regional traffic features. In addition, the author incorporated social activity data into one of the input data to improve the accuracy of the prediction. Due to the diversity and similarity of base stations in different regions, the author proposes a clustering algorithm to divides urban areas into different clusters and a transfer learning strategy to enhance the reuse of features. Zhang et al., 2018 proposes a Double Spatial-Temporal Neural Network (D-STN) prediction model, which combines ConvLSTM and 3D ConvNets for network traffic prediction to improve long-term prediction performance.

3. Hyperparameters Optimization of Lightweight Spatiotemporal Models

3.1 Process architecture

The lightweight prediction architecture diagram of cellular traffic is shown in x, which is divided into five stages: data preprocessing, model building, hyperparameter adjustment, model training, and prediction. First, the input data of regional cellular traffic is converted into a spatiotemporal matrix. The second step is to establish a lightweight prediction model, and then adjust the hyperparameters for the training model. After the model training is completed, the cellular traffic can be predicted.

3.2 Data preprocessing

The cellular traffic data of a region is regarded as input data. We divide the traffic in the area into grids where each grid represents the traffic of a base station. Then, a base station traffic data collected at the time interval $t$ is defined as a spatial sequence $S = \{G_t | t = 1, 2, 3, ..., T\}$. Where $S$ is a tensor $S \in \mathbb{R}^{X \times Y \times N}$ and $v^x_y$ represent the $i^{th}$ cellular traffic in the $(x, y)$ area. $N = x \times y$ is the total number of base stations.

$$
G_t = 
\begin{bmatrix}
    v_1^1, & v_1^2, & \cdots & v_1^N \\
    \vdots & \vdots & \ddots & \vdots \\
    v_i^X, & v_i^X, & \cdots & v_{i,y} \\
\end{bmatrix},
\tag{1}
$$

Considering the temporal dependent characteristics of traffic data, we convert Equation (1) into Equation (2)

$$
G' = 
\begin{bmatrix}
    v_1^1, & v_1^2, & \cdots & v_1^N \\
    v_1^T, & v_1^T, & \cdots & v_1^T \\
    \vdots & \vdots & \ddots & \vdots \\
\end{bmatrix},
\tag{2}
$$

Give $S' = \{G'| t = 1, 2, 3, ..., T| n = 1, 2, 3, ..., N\}$ as a new tensor $S' \in \mathbb{R}^{N \times T}$. Where $v'^n_t$ means the $n^{th}$ cellular traffic at the time interval $t$.

3.2 Lightweight prediction model

In order to accurately predict cellular traffic and significantly reduce computing resources, this research considered the following lightweight mechanism (Chien et al., 2021) and proposed a Lightweight Spatial-Temporal Prediction Network (LSTPN) model based on the MobilenetV3. The characters of spatiotemporal sequence for the traffic data is very important for the accuracy of the prediction. Picture data can remove a lot of useless low-dimensional features to reduce parameters but traffic prediction is not suitable. The predicted data is a spatiotemporal sequence that cannot be greatly compressed. Otherwise,
a large amount of temporal information will be lost and the accuracy rate will be reduced. MobilenetV3 (Howard et al., 2019) has a lot of lightweight mechanisms but not all lightweight mechanisms are suitable for cellular traffic prediction. Some mechanisms will compress a lot of image features, making them unsuitable for traffic prediction.

The LSTPN architecture is shown in Figure 1. The first layer uses a Convolutional Layer (CL) and then connects to a Linear Bottleneck (LB) of five layers. The sixth layer is connected to a convolutional layer and the last layer is connected to a fully connected layer for output. There is a reverse residual structure on the LB. If the training effect of these five layers is not good, it will directly affect the final result. Therefore, we add convolutional layers before and after the LB to ensure the output performance. Although the linear bottleneck structure reduces a lot of parameters than the general convolutional layer, it is still slightly inferior to the general convolutional layer in accuracy. The main purpose of adding the linear bottleneck structure is to increase the deep of the convolutional layer. Since the number of parameters of the BL is much less than that of the general convolutional layer, the effect of reducing the number of parameters and maintaining the original accuracy can be achieved. For predictive models, the deep convolutional layer is very important, because fewer convolutional layers can only learn the characteristics of part of the spatiotemporal sequence. It must use deeper convolutional layers to ensure that all features of the spatiotemporal sequence can be learned. In the LB, Pointwise Convolution (PC), Depthwise Convolution (DC), and PC are included in order. In order to learn more features and reduce the input of parameters in the LB layer, the number of convolution kernels from the first layer of PC to the third layer of PC decreases in order. The purpose of this is to learn more features from the output of the previous layer, output important features during output, and even reduce the amount of parameters. Finally, the convolutional layer is connected and fully connected.

### 3.2 Hyperparameter optimization

This research divides the hyperparameters into two parts, the first part is the external hyperparameters of LSTPN, and the second part is the internal hyperparameters of LSTPN. The external hyperparameters of LSTPN are Batch Size, Epoch and Time Step of base station traffic. The internal hyperparameters of LSTPN are the number of filters, convolution Kernel Size, convolution Stride, the type of activation Function, and the type of linear bottleneck.

This research divides the hyperparameters into two parts for adjustment. The first step will adjust the external hyperparameters of LSTPN; the second step will adjust the internal hyperparameters of LSTPN.

The external hyperparameters of LSTPN will directly affect the prediction accuracy and the training time. We adjust the time step for batch size, Epoch, and base station traffic. First, it is necessary to determine the number of input data for one training, the batch size, and epochs. After confirming that the model has reached a certain accuracy and training speed, modify the time step of base station traffic to improve accuracy.

The adjustment of internal hyperparameters is divided into two steps. The first step tests different types of LB and activation functions. The second step is to use the simulated annealing algorithm to adjust the number of filters, convolution kernel size, convolution stride. The reason why the first step is executed first is to reduce the size of the solution space for hyperparameter optimization and improve the convergence speed. Since the solution space of the LB type and activation function type is smaller than the second step, we can adjust the hyperparameters of the first step to reduce the number of all hyperparameters that need to be adjusted. After the above hyperparameters are adjusted, simulated annealing is adopted to optimize a number of filters, convolution kernel size, and convolution stride. In this way, the parameter quantity and accuracy of LSTPN are optimized at one time. Figure 2 is a flowchart of hyperparameter adjustment strategy.

### 3.4 SA-based hyperparameter optimization

In order to optimize the parameters of LSTPN, this research proposes SA-based algorithm to adjust the convolution kernel size, the number of convolution kernels and the convolution step size.

The hyperparameter adjustment in the neural network will directly affect the parameter quantity and accuracy. There may be four possible results when we adjust hyperparameters. The first type has a large number of parameters and high accuracy; the second type has a large number of parameters and poor accuracy; the third type has a small number of parameters and high accuracy, and the fourth type has a small number of parameters and low accuracy. Therefore, we proposed an SA-based algorithm to optimize hyperparameters so as to achieve a lightweight neural network model with fewer parameters and high accuracy. First, input the base
station data to the LTTPPN model for training and randomly generate hyperparameters for the convolution kernel size, the number of convolution kernels, and the convolution stride. Then, run the main program of the SA-based algorithm. Subsolutions will be generated. We can evaluate the subsolutions according to the fitness function. Please refer to Equation (3).

\[ F = \alpha \times F_{\text{loss}} + \beta \times F_{\text{parameter}}, \]  

(3)

Where \( F \) is fitness function, \( F_{\text{loss}} \) and \( F_{\text{parameter}} \) are the normalized loss value and parameter amount respectively, \( \alpha \) and \( \beta \) are weights. Our goal is to minimize \( F \).

At the beginning of the algorithm, \( k \) sets of descendants (subsolutions) are randomly generated based on the ancestor (original parameters). From these descendants, the best solution is selected to replace the ancestor. If the \( F \) of the selected descendants is worse than that of the ancestor, it will be judged whether to accept the solution according to the annealing temperature. In other words, even if \( F \) is relatively poor, there is still a high probability that a bad solution will be accepted in the initial iteration process so as to avoid quickly falling into the local optimum. The annealing temperature will decrease exponentially with the number of iterations. The SA-based algorithm will continue to search within the temperature range and adjust the convolution kernel size, the number of convolution kernels, and the convolution stride until the \( F \) value no longer changes. That is, the hyperparameters of the LTSTPN have been optimized. Please refer to the algorithm 1.

4. Experimental results and discussion

The environment set up in this experiment is the
Window operating system, using the Python programming language. The TensorFlow version is 1.5 GPU version, and the Keras version is 2.2.5.

4.1 Cellular traffic data

This research uses public data published by the European telecom company Telecom Italia to predict cellular traffic (Barlacchi et al. 2015). The data was collected in Milan from November 10, 2013 to January 1, 2014. The data collection interval is in 10 minutes. There are 10,000 base stations in this data set. Figure 3 shows a visualization of network traffic data.

4.1 Traffic prediction results

We use the traffic data of 900 base stations as training data and calculates the standard deviation of each base station at different times. Two of the base stations that have the largest and the smallest standard deviations respectively are selected to compare the traffic prediction results. Given $G'(n,t)$

We use three commonly used performance indicators to evaluate prediction performance: Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and Mean Absolute Percentage Error (MAPE) for evaluation. Where $v'_n^t$ is the training data, $p_n^t$ is the predicted result, and $M$ is the size of the predicted data.

\[
MAE = \frac{1}{M} \sum_{m=1}^{M} |v'_m - p_m^t|, \quad (4)
\]

\[
RMSE = \sqrt{\frac{1}{M} \sum_{m=1}^{M} (|v'_m - p_m^t|)^2}, \quad (5)
\]

\[
MAPE = \frac{1}{M} \sum_{m=1}^{M} \frac{|v'_m - p_m^t|}{v'_m}, \quad (6)
\]

\[
MA = (1 - MAPE) \times 100\%, \quad (7)
\]

Equation (7) is used to the calculation the Mean Accuracy (MA), which is used for accuracy measurement. Figure 4a shows the base station traffic with the largest standard deviation 0.046; Figure 4b shows the base station traffic with the smallest standard deviation 0.017.
Table 1 The hyperparameters table

| Hyperparameters | value     |
|-----------------|-----------|
| Batch size      | 32        |
| Epoch           | 140       |
| Time step       | 6         |
| Optimizer       | Adam      |

| BL type          | (Kernel size) |
|------------------|---------------|
| The 1\textsuperscript{st} BL : 16 | The 2\textsuperscript{nd} BL : 48 |
| The 3\textsuperscript{rd} BL : 80 | The 4\textsuperscript{th} BL : 48 |
| The 5\textsuperscript{th} BL : 16 | The 6\textsuperscript{th} BL : 80 |
|                  | The 7\textsuperscript{th} BL : 16 |

| Activation function type | (Kernel size) |
|--------------------------|---------------|
| The 1\textsuperscript{st} CL : Relu | The 2\textsuperscript{nd} BL : Relu |
| The 3\textsuperscript{rd} BL : H-swish | The 4\textsuperscript{th} BL : H-swish |
| The 5\textsuperscript{th} BL : H-swish | The 6\textsuperscript{th} BL : H-swish |

| Convolution kernel size | 16 |
|-------------------------|----|
| Number of convolution kernels | 3 |
| Convolution stride      | 6  |

Table 2 Comparison of parameters and accuracy

|            | Parameters | RMSE  | Training Time (s) |
|------------|------------|-------|-------------------|
| Empirical rule       | 564,852    | 296.53|                   |
| Greedy-based algorithm | 510,177    | 287.43| 46,136            |
| SA-based algorithm   | 360,259    | 278.71| 48,216            |

Figure 5a is the traffic prediction result of the first type of base station obtained by using the optimized parameter model; Figure 5b is the traffic prediction result of the second type of base station obtained by using the optimized parameter model. The orange line is the test result, and the red line is the predicted result.

We compared the traffic prediction performance of 10 neural network models, including linear models (ES, ARIMA) and nonlinear models (GRU, LSTM, Selfattention, CNN, CNN+LSTM, ConvLSTM, Densenet, and LSTPN), Figure 6 and Figure 7 show the comparison of traffic predictions of the first type base station and second type base station. Linear models (ES, ARIMA) have the lowest accuracy. The three time-domain models, GRU, LSTM, and Selfattention, only consider temporal dependence. In other words, these three methods only consider the historical traffic data of a single-point base station and do not consider the spatial dependence between neighboring base stations. On the contrary, CNN, CNN + LSTM, ConvLSTM, Densenet, and LSTPN also consider the characteristics of spatiotemporal dependence. Therefore, they have better traffic prediction performance. On the whole, nonlinear models have higher accuracy than linear models because various nonlinear factors must be considered in the real environment. The linear model is difficult to deal with these problems. For nonlinear models, compared with models that consider temporal and spatial dependencies (CNN, CNN+LSTM, ConvLSTM, Densenet, LSTPN) and models that simply consider temporal dependencies (GRU, LSTM, Selfattention), the former has higher accuracy.

The adjustment of hyperparameters will affect the accuracy and parameter quantity of the LSTPN model. We use the adjustment results of the hyperparameters of different LSTPN models to optimize the hyperparameters of the LSTPN model. Table 1 shows the hyperparameters table of the model used in this study.

This study uses the SA algorithm to optimize the hyperparameters of the convolution kernel size, the number of convolution kernels, and convolution stride. In addition, we compare the prediction results based on the SA algorithm and the greedy algorithm.

Figure 8 is the comparison of the value of the fitness function, please refer to Equation 3. The red line is the greedy algorithm, and the blue line is the SA algorithm. Greedy-based algorithms show a relatively stable downward trend, while SA-based algorithms have the opportunity to accept poor solutions at an early stage to avoid quickly falling into the local optimum. Therefore, the magnitude of the fitness function value is relatively large at the beginning. But the cooling started to converge. The result shows that the SA-based algorithm can get better accuracy and fewer parameters. The comparison of RMSE and parameters are shown in Figure 9 and Fig. 10.

Table 2 is the comparison of parameters and accuracy. The first method uses empirical rules to adjust the parameters. The second is to use the greedy-based algorithm; the third is to use the SA-based algorithm. The SA-based algorithm has higher accuracy and fewer parameters than empirical rules and greedy-based algorithm in terms of accuracy and parameter quantity. There are many hyperparameters that need to be adjusted by the deep learning model but the empirical rule is difficult to achieve good results. The algorithm method can search for better hyperparameters, but the search process requires additional training time to search for hyperparameters. In this study, the SA-based algorithm needs to use more running time than the greedy algorithm. Therefore, although better results can be obtained by using this method to adjust the hyperparameters, it takes extra time to train the model for the first time.

5. Conclusion

In order to reduce the computational resources for cellular traffic prediction while maintaining a certain degree of accuracy, this paper proposes a
hyperparameter optimization strategy based on a simulated annealing algorithm for lightweight neural networks. Since the data of the cellular traffic has the characteristics of Spatiotemporal dependence, the regional cellular traffic data is converted into a spatiotemporal matrix in the data pre-processing. Then, we modify the neural network model based on the architecture of MobilenetV3. The mechanism of destroying time series data is discarded, and the total number of model layers is reduced. The hyperparameters in this paper are divided into exte-
hyperparameters and inter-hyperparameters. In order to improve prediction accuracy and reduce parameters, the SA-based algorithm is used to optimize the hyperparameters, including the size of the convolution kernel in the convolution layer, the number of convolution kernels, and convolution stride. The simulation results show that the proposed explainable learning method can effectively improve the prediction accuracy and reduce the number of parameters of the neural network.

In addition, we give local and global explanations to interpret the improvement of traffic prediction. The local explanations show the adjustment results of a single hyperparameter, and global explanations show the correlation between different hyperparameters and their influence on the amount and accuracy of model parameters. The research contributes to the understanding of the reasons behind the hyperparameters adjustment.

**Acknowledgements** Thank Edmund Pok Leng Shiang for providing some useful comments to enhance quality of this paper. This research was partly funded by Ministry of Science and Technology (MOST) in Taiwan under grants 108-2221-E-197-012 -MY3.

**Author Contributions** W.-C. C. and S.-Y. C. contributed to conceptualization and emerging technologies; H.-H. C. and F.-H. T. were involved in methodology of explainable learning; W.-C. C. provided software; H.-H. C. and F.-H. T. contributed to formal analysis; and W.-C. C., S.-Y. C., F.-H. T. and S.-Y. C. contributed to writing—original draft preparation. All authors have read and agreed to the published version of the manuscript.

**Declarations**

**Conflict of interest** the authors declare that they have no conflicts of interest.

**Ethical approval**

This article does not contain any studies with human participants performed by any of the authors.

**Reference**

Barlacchi G, De Nadai M, Larcher R, Casella A, Chitic C, Torrisi G, ... , Lepri B (2015). A multi-source dataset of urban life in the city of Milan and the Province of Trentino. Scientific data, 2(1): 1-15.

Chien WC, Huang YM (2021). A lightweight model with spatial–temporal correlation for cellular traffic prediction in Internet of Things. The Journal of Supercomputing, 1-17.

Chien WC, Lai CF, Hossain MS, Muhammad G (2019) Heterogeneous space and terrestrial integrated networks for IoT: Architecture and challenges. IEEE Network, 33(1):15-21.

Erol-Kantarci M, Sukhmani S (2018). Caching and computing at the edge for mobile augmented reality and virtual reality (AR/VR) in 5G. Ad Hoc Networks, 169-177.

Gers FA, Schmidhuber J, Cummins F (2000). Learning to forget: Continual prediction with LSTM. Neural computation, 12(10):2451-2471.

Howard A, Sandler M, Chu G, Chen LC, Chen B, Tan M, …, Adam H (2019). Searching for mobilenetv3. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 1314-1324).

Kamel M, Hamouda W, Youssef A (2016). Ultra-dense networks: A survey. IEEE Communications Surveys & Tutorials, 18(4):2522-2545.

Lv Y, Duan Y, Kang W, Li Z, Wang F Y (2014). Traffic flow prediction with big data: a deep learning approach. IEEE Transactions on Intelligent Transportation Systems, 16(2):865-873.

Lai CF, Chien WC, Yang LT, Qiang W (2019). LSTM and edge computing for big data feature recognition of industrial electrical equipment. IEEE Transactions on Industrial Informatics, 15(4):2469-2477.

Maghs H, Sodhro AH, Chachar FA, Abro SAK, Sodhro GH, Pirbhulal S (2018). Evolution of 5G in Internet of medical things. In 2018 international conference on computing, mathematics and engineering technologies (iCoMET), p. 1-7.

Neve MJ, Rowe GB (1995). Mobile radio propagation prediction in irregular cellular topographies using ray methods. IEEE Proceedings-Microwaves, Antennas and Propagation, 142(6):447-451.

Patil S, Kumar R (2019). Accelerating Extended Reality Vision With 5G Networks. In 2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA), p. 157-161.

Qiu C, Zhang Y, Feng Z, Zhang P, Cui S (2018). Spatio-temporal wireless traffic prediction with recurrent neural network. IEEE Wireless Communications Letters, 7(4):554-557.

Rao SK, Prasad R (2018). Impact of 5G technologies on smart city implementation. Wireless Personal Communications, 100(1):161-176.

Shi X, Chen Z, Wang H, Yeung DY, Wong WK, Woo WC (2015). Convolutional LSTM network: A machine learning approach for precipitation prediction.
nowcasting. arXiv preprint arXiv:1506.04214.
Wang J, Tang J, Xu Z, Wang Y, Xue G, Zhang X, Yang D (2017). Spatiotemporal modeling and prediction in cellular networks: A big data enabled deep learning approach. In IEEE INFOCOM 2017-IEEE Conference on Computer Communications, p. 1-9.
Wu Y, Tan H (2016). Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework. arXiv preprint arXiv:1612.01022.
Xu, F., Uszkoreit, H., Du, Y., Fan, W., Zhao, D., & Zhu, J. (2019). Explainable AI: A brief survey on history, research areas, approaches and challenges. In CCF international conference on natural language processing and Chinese computing, p. 563-574. Springer, Cham.
Zhang C, Patras P (2018). Long-term mobile traffic forecasting using deep spatio-temporal neural networks. In Proceedings of the Eighteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing, p. 231-240.
Zhang C, Zhang H, Yuan D, Zhang M (2018). Citywide cellular traffic prediction based on densely connected convolutional neural networks. IEEE Communications Letters, 22(8):1656-1659.
Zhang C, Zhang H, Qiao J, Yuan D, Zhang M (2019). Deep transfer learning for intelligent cellular traffic prediction based on cross-domain big data. IEEE Journal on Selected Areas in Communications, 37(6):1389-1401.