ISSN 2347 - 3983
Volume 13. No.4, April 2025
International Journal of Emerging Trends in Engineering Research
Available Online at http://www.warse.org/IJETER/static/pdf/file/ijeter021342025.pdf
https://doi.org/10.30534/ijeter/2025/021342025

A Comparative Analysis of Machine Learning Models for Cellular Load Prediction - Insights from Real-World Data

Hamidullah Riaz[1], Peri Güneş[2], Harun Benli[3], Fazel Haq Ahmadzai[4]
[1] Kocaeli University, Department of Electronics and Communication Engineering, Kocaeli, Türkiye, hamidullah.riaz@kocaeli.edu.tr
[2] Istanbul Gelisim University, Faculty of Engineering and Architecture, Istanbul, Türkiye, pgunes@gelisim.edu.tr
[3] Infina Software Inc., Istanbul, Türkiye, hbenli@infina.com.tr
[4] Virginia Tech, Faculty of Industrial and Systems Engineering, Virginia, USA, fazel1990@vt.edu

Received Date: February 27, 2025 Accepted Date: March 29, 2025 Published Date : April 07, 2025

ABSTRACT

Accurate network traffic prediction is essential for efficient resource allocation and congestion management in telecommunications. This study evaluates the performance of various Machine Learning (ML) techniques, such as Linear Regression (LR), Random Forest (RF), Support Vector Machine (SVM), Extreme Gradient Boosting (XGBoost), Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU), for predicting total traffic volume in a Long-Term Evolution (LTE) network. A real-world dataset collected from an LTE site of a telecom service provider in Kandahar Province of Afghanistan was used for model training and evaluation. Performance was assessed using RMSE, MAPE, and R^2 metrics. The results demonstrate that Deep Learning (DL) models, particularly LSTM, outperform traditional ML methods by effectively capturing temporal dependencies in traffic patterns. In contrast, traditional models exhibited lower predictive accuracy with high error rates. The findings highlight the potential of Recurrent Neural Networks (RNNs) for time-series forecasting in network traffic prediction, providing valuable insights for selecting suitable predictive models in similar datasets.

Key words : Machine Learning (ML), Mobile Traffic Forecasting, Real-World Dataset, Recurrent Neural Network (RNN).

1. INTRODUCTION

The steady increase in the number of mobile users, mobile applications, interconnected devices, and related services is driving the growth of mobile data traffic [1]. In recent years, both the global mobile user base and data consumption have grown significantly. Estimates suggested that by 2023, the total number of mobile devices worldwide could reach tens of billions. By the end of 2023, mobile data traffic, excluding fixed wireless networks, was expected to reach 130 ExaBytes (EB) per month, with a possible increase to 403 EB per month by 2029. When including fixed wireless networks, total mobile network traffic was projected to reach around 160 EB per month by late 2023, rising further to 563 EB per month by 2029. Additionally, the share of mobile data traffic associated with Fifth Generation (5G) networks was predicted to grow from 15% in late 2022 to 25% by the end of 2023, with forecasts indicating an increase to 76% by 2029 [2].

As mobile data traffic continues to grow rapidly, ensuring efficient management of network resources becomes increasingly challenging. In this context, accurate cellular network traffic prediction is crucial for effective network management. Predicting traffic patterns offers several benefits, such as enabling proactive congestion control, enhancing information security, optimizing network planning, and ensuring efficient bandwidth allocation. Long-term traffic forecasting helps develop detailed models to anticipate future demands, allowing for more precise planning and better decision-making.

Similarly, accurate short-term predictions of mobile traffic load play a vital role in improving network efficiency by enabling proactive adjustments to network resources. By anticipating traffic fluctuations, network operators can dynamically allocate bandwidth and optimize network parameters to maintain stable performance.

Furthermore, short-term mobile traffic load prediction facilitates efficient resource management, congestion mitigation, and load distribution among nearby Base Stations (BSs) [3]. Notably, forecasting mobile traffic load also helps lower overall power consumption in mobile networks by allowing BSs to enter sleep mode during periods of low traffic,

as demonstrated in [4]. This, in turn, enhances network performance, minimizes service disruptions, and improves the overall user experience.

Cellular traffic prediction can be broadly categorized into temporal and spatial-temporal prediction. Temporal prediction focuses on forecasting traffic flow at a single location using only its historical data, considering time-based dependencies. This is typically treated as a univariate time-series problem.

In contrast, spatial-temporal prediction incorporates both time-based and spatial dependencies, aiming to predict traffic patterns across multiple interconnected network elements. For instance, in cellular networks, BSs influence each other due to user mobility and handovers, making their traffic patterns interdependent. Accurately modeling these spatial relationships can improve prediction accuracy in dynamic network environments [5].

Various indicators, such as upload and download traffic volume, the number of connected users, and the number of radio resource control connections, are commonly used to monitor the state of cellular networks. Based on the number of target variables being predicted, cellular traffic prediction can be classified into univariate and multivariate prediction problems [6].

Univariate prediction focuses on forecasting a single variable, such as traffic volume or the number of connected users. In contrast, multivariate prediction involves simultaneously predicting multiple network indicators, which often exhibit interdependencies. For example, forecasting both the number of connected users and traffic volume at a BS is a typical multivariate problem, as an increase in connected users generally leads to higher traffic volume [7].

From the perspective of prediction duration, traffic forecasting can be categorized into short-term and medium-to-long-term predictions. However, there is no universal criterion for this classification, as it depends on factors like time granularity. Generally, for time series data with a 5-minute granularity, short-term prediction typically ranges from 5 to 60 minutes, whereas medium-to-long-term prediction extends beyond 60 minutes [8].

Additionally, based on the number of future time steps being predicted, cellular traffic forecasting can be classified as single-step or multi-step prediction. Single-step prediction focuses on forecasting traffic for the immediate next time step, whereas multi-step prediction extends the forecast over multiple future time intervals [7].

Predicting traffic in cellular networks can be approached using different methods, from traditional time-series forecasting to advanced Machine Learning (ML) techniques. Statistical models, for example, Auto-Regressive Integrated Moving Average (ARIMA), are simple and easy to interpret. However, they struggle to handle the complex and changing nature of mobile network traffic. In contrast, ML models provide better accuracy and scalability, making them more suitable for dynamic network environments [9]. Overall, traffic prediction methods can be grouped into two main categories: (i) statistical-based approaches and (ii) ML and Deep Learning (DL) models. Deep learning, in particular, has shown great potential in dealing with high-dimensional and time-based data.

Although ML techniques improve prediction accuracy, they require large amounts of data and strong computing power. Additionally, they often depend on hand-crafted features, which may not fully capture the complex patterns in cellular traffic. Recently, DL has made significant progress and has been successfully applied in many fields. As a result, researchers have started using DL models for cellular traffic prediction.

Most existing traffic prediction solutions [10], [11], [12] rely on DL models because they can process large datasets and provide accurate long-term forecasts. Many types of DL models have been used in network traffic prediction. For example, Long Short-Term Memory (LSTM) networks are commonly used for predicting cellular network traffic [13], [14], [15]. Gated Recurrent Units (GRU) also has been applied for the same purpose as in [16]. Hybrid models combining LSTM and Convolutional Neural Networks (CNN) have also been explored for better feature extraction [17] with the cost of complexity and processing load. Similarly, a mix of GRU version and CNN has been applied in [18].

In addition to DL models, other ML techniques have been used for traffic forecasting. Tree-based models like Extreme Gradient Boosting (XGBoost) and Random Forest (RF) are effective at handling non-linear data and selecting important features. Support Vector Machines (SVM) work well with high-dimensional data, whereas Multi-Layer Perceptron (MLP) [19] provides a flexible neural network-based approach.

This study presents a comparative performance evaluation of these models, analyzing their predictive accuracy, computational efficiency, and suitability for mobile network traffic forecasting. By systematically assessing their strengths and weaknesses, we aim to identify the most effective model for optimizing network resource management and ensuring reliable traffic predictions based on real-world traffic data.

2. DATASET DESCRIPTION

The dataset used in this study was collected from a telecom service provider and represents traffic data for a single Long-Term Evolution (LTE) site with three cells in Kandahar

Province, Afghanistan. The data spans one month, from November 1 to November 30, 2024, covering 24 hours per day, as shown in **Figure 1**.

The dataset includes the following key attributes: Date and Time (timestamp of each recorded data entry), Cell ID (unique identifier for each of the three cells at the site), Upload Traffic Volume (amount of data uploaded by users within the cell), Download Traffic Volume (amount of data downloaded by users within the cell), and Total Traffic Volume (sum of upload and download traffic, representing the overall data usage in each cell).

The dataset comprises a total of 2,160 samples. The upload traffic volume has a mean of 177.09 Mbps, a standard deviation of 132.81 Mbps, a minimum of 11.39 Mbps, and a maximum of 153.84 Mbps. The download traffic volume has a mean of 1,408.95 Mbps, a standard deviation of 937.84 Mbps, a minimum of 153.84 Mbps, and a maximum of 5,120.32 Mbps. The total traffic volume, on the other hand, has a mean of 1,586.04 Mbps, a standard deviation of 1,037.09 Mbps, a minimum of 173.90 Mbps, and a maximum of 5,920.82 Mbps.

Figure 1: Traffic volume analysis for different time granularities. **(a)** Daily total traffic volume per cell for November 2024. **(b)** Hourly total traffic volume per cell on November 15, 2024

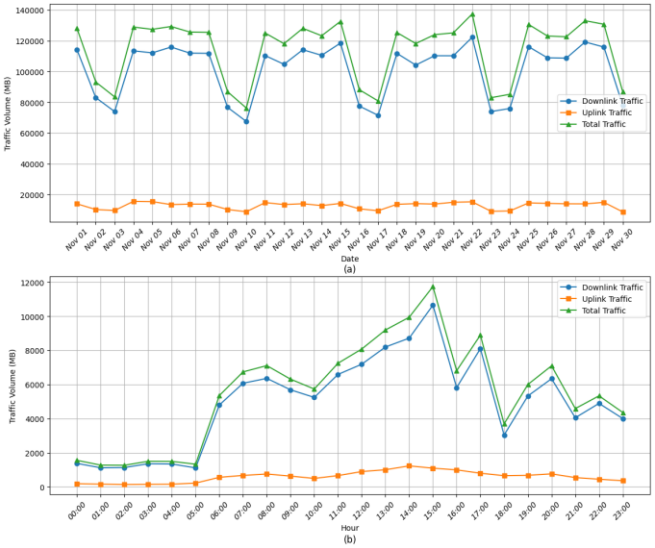

Figure 2: Traffic Volume Analysis for November 2025. **(a)** Monthly traffic trends from November 1 to November 30, showing uplink, downlink, and total traffic volumes. **(b)** Hourly traffic breakdown for November 15, displaying uplink, downlink, and total traffic volumes throughout the day.

In Figure 2, the traffic volume trends for the month of November 2025 are illustrated. Subplot (a) shows the uplink, downlink, and total traffic volumes from November 1 to November 30, while subplot (b) displays the hourly breakdown of uplink, downlink, and total traffic volumes for November 15.

This dataset provides essential insights into network usage patterns, enabling the analysis of traffic trends and serving as a reliable foundation for training machine learning models to forecast future network demand.

To effectively analyze and model the dataset's attributes, it is essential to preprocess the data appropriately. Given that traffic volume metrics can vary significantly in scale, applying Min-Max normalization ensures that all features contribute proportionally to the predictive models. This normalization technique scales each feature to a specified range, typically [0, 1], preserving the relationships among original data values and enhancing model performance. The detailed methodology and rationale for selecting Min-Max normalization are elaborated in the Section x.

3. ML MODELS

3.1. Linear Regression (LR)

LR is a statistical method that estimates how a dependent variable responds to changes in independent variables. The general form of the LR equation can be as (1):

$$P = \beta_0 + \beta_1 F_1 + \beta_2 F_2 + \cdots + \beta_n F_n \quad (1)$$

where the β values represent the estimated coefficients for the independent variables, while P is the dependent variable. Specifically, P represents the predicted house price, and F_i corresponds to the various chosen features. LR is selected because it offers a clear and interpretable method for understanding the relationship between features and the target variable. Additionally, it serves as a valuable point of comparison for more complex models [20].

3.2. Random Forest (RF)

RF is an ensemble learning algorithm that enhances decision trees by introducing randomness for better generalization. It constructs multiple trees using a recursive binary split method. Each tree is trained on a bootstrapped subset of data and selects a random subset of features at each split as illustrated in Figure 3. This approach ensures diversity, as different trees capture varying data patterns, and aggregation, where predictions are combined (e.g., majority voting for classification, averaging for regression). By reducing variance and improving robustness, RF delivers strong predictive performance while mitigating overfitting [21]. Mathematically, for regression, the prediction y for an input x is given in (2) as following:

$$y = \frac{1}{M}\sum_{j=1}^{M} T_j(x) \tag{2}$$

where M is the number of trees in the forest, and $T_j(x)$ represents the prediction from the j[th] tree for input x.

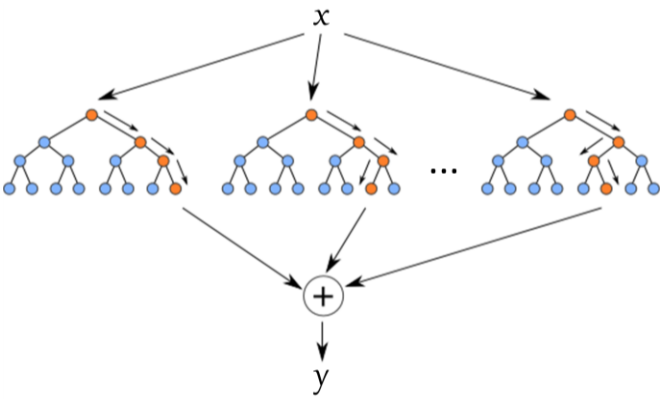

Figure 3: A general illustration of RF regression

3.3. Support Vector Machines (SVM)

SVM is a supervised learning algorithm widely used for predictive modeling in machine learning. While commonly associated with classification tasks, SVM is also effective in making predictions for both classification and regression problems. The core idea behind SVM is to identify a decision boundary that optimally separates data points while minimizing errors. This enables SVM to make accurate predictions based on learned patterns [22].

For prediction, SVM constructs a model based on a set of training data and finds a function $g(X)$ that can generalize well to unseen data. Mathematically, this function can be represented in (3) as below:

$$g(X) = \omega^T X + b \tag{3}$$

where ω is the weight vector, X represents the input features, and b is the bias term.

3.4. Extreme Gradient Boosting (XGBoost)

XGBoost is a powerful ML algorithm based on gradient boosting, excelling in both regression and classification tasks. It builds an ensemble of decision trees sequentially, where each tree corrects the errors (residuals) of the previous one, improving overall accuracy. The model follows the iterative update formula given in (4) as below:

$$f_m(x) = f_{m-1}(x) + \gamma h_m(x) \tag{4}$$

where $f_m(x)$ is the prediction at iteration m, $f_{m-1}(x)$ is the prediction from the previous iteration, γ is the learning rate, and $h_m(x)$ is the prediction of the m^{th} tree.

3.5. Long Short-Term Memory (LSTM)

LSTM is a specialized Recurrent Neural Network (RNN) architecture designed for time-series prediction, effectively capturing long-term dependencies while mitigating the vanishing gradient problem. It consists of four key components: the cell state, input gate, output gate, and forget gate. The cell state serves as a memory chain that carries information across time steps with minimal modifications. The forget gate, defined as (5) determines which information should be discarded.

$$f_t = \sigma\left(\omega_f\left[h_{t-1}, x_t\right] + b_f\right) \tag{5}$$

where σ is the sigmoid activation function, x_t is the new input, h_{t-1} is the previous hidden state, and ω_f and b_f represent the weight and bias parameters, respectively.

Equation (5) produces an output ranging between 0 and 1. An output of 0 indicates that the value should be completely discarded, whereas an output of 1 signifies that the value should be retained. In the following step, the information to be stored in the cell state is determined. The sigmoid layer, also referred to as the input gate layer, selects the values to be preserved, while the tanh layer generates a vector of new candidate values that can be integrated with the current state.

The input gate is defined as (6) while (7) represents the new candidate values:

$$i_t = \sigma(\omega_i[h_{t-1}, x_t] + b_i) \quad (6)$$

$$\tilde{c}_t = tanh(\omega_c[h_{t-1}, x_t] + b_c) \quad (7)$$

where i_t represents the input gate layer, and $\tilde{c}_t$ denotes the vector of new candidate values. The updated cell state can be computed by combining (6) and (7), resulting in the following expression for c_t:

$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t \quad (8)$$

The final step involves computing the output. A sigmoid layer is applied to determine which parts of the cell state contribute to the final output. Subsequently, the updated cell state is passed through a tanh layer to scale the values within the range of [-1, 1]. This result is then multiplied by the output from the sigmoid gate, ensuring that only the selected components are retained in the final output.

$$o_t = \sigma(\omega_o[h_{t-1}, x_t] + b_o) \quad (9)$$

$$h_t = o_t * \tanh(c_t) \quad (10)$$

Here, o_t represents the output of the sigmoid gate, while h_t corresponds to the final selected output. This output is then fed into the next layer as input, allowing the sequence to continue [23]. The general LSTM architecture is illustrated in Figure 4.

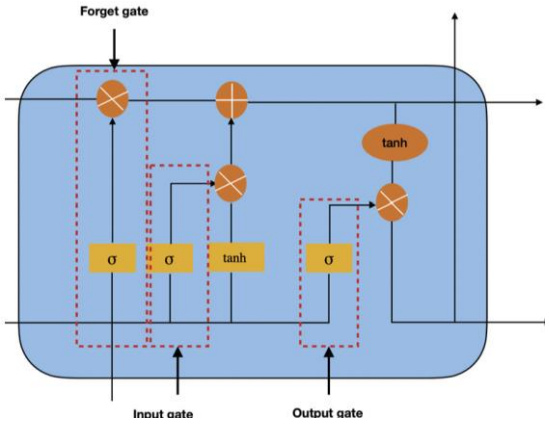

Figure 4: LSTM general architecture [5]

3.6. The Gated Recurrent Unit (GRU)

GRU is a variant of RNNs designed to enhance the computational efficiency of LSTM networks, particularly when handling large-scale datasets. The fundamental objective of GRU is to simplify the internal structure of LSTM blocks, thereby reducing the computational complexity and improving processing speed. Unlike LSTM, which utilizes three gates, the GRU architecture comprises only two: the update gate and the reset gate as illustrated in Figure 5. The update gate controls the flow of information across time steps, while the reset gate determines the extent to which previous information is retained or discarded.

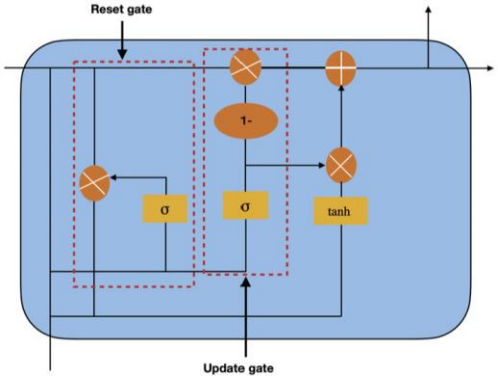

Figure 5: GRU general architecture [5]

Although there is no definitive consensus on whether LSTM or GRU is superior in performance, the choice between them depends on the specific application requirements. GRU is preferred when computational efficiency and speed are of primary importance, whereas LSTM is more suitable for tasks where accuracy takes precedence [24].

4. METHODOLOGY

This study evaluates the performance of different ML techniques in predicting traffic load using a real-world dataset. The dataset consists of upload, download, and total traffic load data from an LTE site with three sectors (cells) as discussed in Section 2. The methodology follows a systematic approach to data preprocessing, model selection, training, evaluation, and comparison.

4.1 Data Preprocessing

Data preprocessing is essential to ensure that the dataset is clean, well-structured, and ready for training ML models. This step involves handling missing values, normalizing data, and preparing the dataset for efficient model training and evaluation.

Since the dataset used in this study did not contain any missing values, a thorough check was performed prior to model training to ensure data completeness. Statistical methods, such as identifying NaN values and checking for outliers, were applied to verify the integrity of the dataset. Given that no missing data was detected, no imputation or substitution was required, ensuring that the dataset remained intact and ready for training and evaluation.

4.1.1 Data Normalization

To ensure that all features are on a comparable scale and to prevent any one feature from disproportionately affecting the model performance, Min-Max normalization was applied to

the traffic data. This technique rescales the data into a predefined range, typically [0, 1], by transforming each feature value according to (11) as following [25]:

$$X_{normalized} = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (11)$$

where X represents the original data point, X_{min} is the minimum value of the feature, and X_{max} is the maximum value of the feature. By normalizing the data in this manner, we ensure that the different traffic load metrics (upload, download, and total) are treated equally, allowing for a fair comparison between them and improving the performance of the ML models.

Additionally, this normalization minimizes bias from variables with varying scales, ensuring a more accurate and fair comparison between models.

4.1.2 Train-Test Split

To evaluate model performance and ensure generalization to unseen data, the dataset was divided into two subsets: a training set (70%) and a testing set (30%) as shown in Table 1. The training set was used to train the models, while the testing set was reserved for evaluating predictive performance. This division simulates real-world scenarios, where the model learns from historical data and makes predictions on new, unseen data. The split was performed randomly to ensure that both subsets accurately represent overall traffic patterns, reducing the risk of overfitting and ensuring that the model does not simply memorize the training data.

Table 1: Train-test split of total traffic data

Feature	Total Samples	Training Set (70%)	Testing Set (30%)
Total Traffic	2160	1512	648

4.2 Selection ML Models

For the evaluation of traffic prediction performance, we consider a diverse set of ML techniques, including LR, RF, SVM, XGBoost, LSTM, and GRU. These models are selected based on their unique strengths in handling different aspects of traffic prediction.

LR serves as a baseline, providing a simple yet interpretable approach to capturing linear relationships. RF and XGBoost, both ensemble-based models, effectively capture complex nonlinear patterns and interactions within traffic data. SVM is included for its ability to handle high-dimensional feature spaces and its robustness to noise.

Given the sequential and time-dependent nature of traffic data, deep learning models such as LSTM and GRU are chosen for their strong temporal learning capabilities, allowing them to capture long-range dependencies and dynamic trends.

By evaluating these models, we aim to compare their predictive capabilities and determine the most effective approach for accurate traffic forecasting.

4.3 Performance Metrics

To assess the prediction performance more effectively, the most commonly used evaluation metrics are considered. Root Mean Squared Error (RMSE) and Mean Absolute Percentage Error (MAPE) are employed to quantify prediction errors, with lower values indicating better model accuracy. Additionally, the coefficient of determination R^2 is used to evaluate how well the predicted values represent the actual data, where higher values signify a better predictive performance.

4.3.1 Root Mean Squared Error (RMSE)

The RMSE is the square root of the Mean Squared Error (MSE) and quantifies the standard deviation of prediction errors. Since it is expressed in the same units as the predicted values, it provides an intuitive measure of the model's accuracy, making it easier to interpret the magnitude of errors.

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{\left(y_i^t - y_i^p\right)^2}{n}} \quad (12)$$

where y^t, y^p, and n are actual target values, predicted values, and sample size (number of observations), respectively.

4.3.2 Mean Absolute Percentage Error (MAPE)

MAPE measures the average percentage deviation between predicted and actual values, providing a relative assessment of prediction accuracy. By expressing errors as a percentage of the actual values, MAPE allows for easier interpretation and comparison across different datasets and scales. This metric is particularly useful in scenarios where understanding the magnitude of errors relative to the actual values is essential, making it a valuable tool for evaluating model performance in various applications. However, MAPE can be sensitive to very small actual values, potentially leading to inflated error percentages. MAPE can be formulated as (13).

$$MAPE = \frac{1}{n}\sum_{i=1}^{n} 100 \times \frac{\left|y_i^t - y_i^p\right|}{\left|y_i^p\right|} \quad (13)$$

4.3.3 Coefficient of Determination (R^2)

R^2 quantifies the proportion of variance in the actual values that can be explained by the predicted values. It ranges from 0 to 1, where a value closer to 1 indicates a stronger correlation and a better model fit. A higher R^2 suggests that the model effectively captures the patterns in the data, while a lower value implies that significant variability remains unexplained. This metric is particularly useful for evaluating the overall goodness of fit, helping to determine how well the model

represents the underlying relationships in the data. However, R^2 alone may not fully reflect model performance, especially in cases where data is nonlinear or contains outliers. Equation (14) represents R^2 as below:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}\left(y_i^t - y_i^p\right)^2}{\sum_{i=1}^{n}(y_i^t - \bar{y}^t)^2} \quad (14)$$

where $\bar{y}^t$ is the mean of the actual target values.

5. RESULTS AND DISCUSSION

This section presents the evaluation of various ML techniques for predicting traffic load using a real-world dataset. As discussed in Section 2, the dataset consists of upload, download, and total traffic load data collected from an LTE site with three sectors. However, only the total traffic load was considered for prediction due to its comprehensive representation of the overall traffic behavior, which includes both upload and download activities. The performance of six ML models, including LR, RF, SVM, XGBoost, LSTM, and GRU, was assessed based on key performance metrics such as RMSE, MAPE, and R^2.

Figure 6 illustrates the true versus predicted traffic values for the LSTM model. The plot demonstrates that the LSTM captures the overall traffic patterns, with its predictions following the actual variations in network load. While the model effectively learns sequential dependencies, some discrepancies appear, particularly during sharp fluctuations.

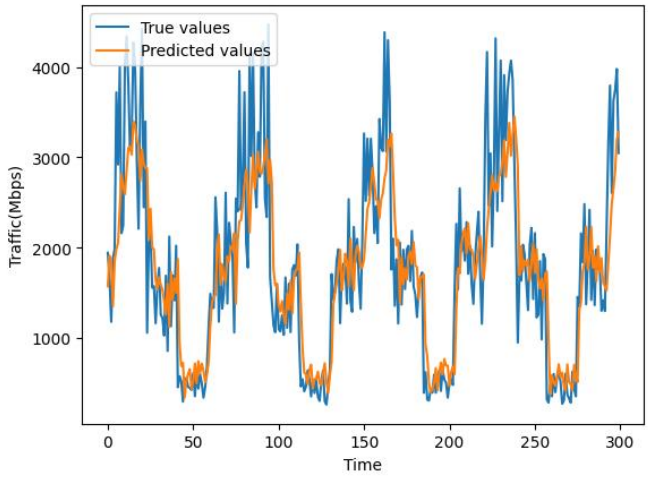

Figure 6: True versus predicted LTE site traffic load using the LSTM model

Similarly, Figure 7 presents the true versus predicted traffic values generated by the GRU model. Like LSTM, GRU captures the general trend of network traffic, but some deviations are noticeable, especially in high-variation regions. The predicted values align well with the actual traffic data but occasionally smooth out certain peaks.

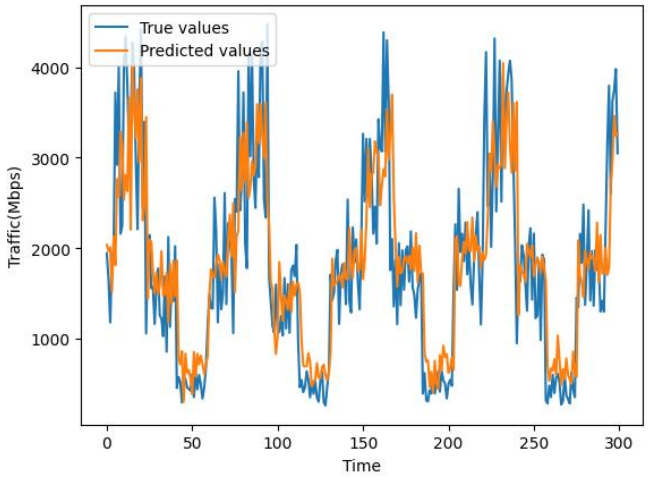

Figure 7: True versus predicted LTE site traffic load using the GRU model

To further assess model performance, Table 2 summarizes key performance metrics for LSTM, GRU, and traditional ML models.

Table 2: Summary of the key performance metrics of different ML techniques in LTE traffic prediction

Model	RMSE	MAPE (%)	R^2 Score
LSTM	688.81	37.94	0.592
GRU	732.47	41.88	0.539
LR	705.26	42.8	0.197
RF	696.21	118.37	0.266
SVM	707.26	111.56	0.22
XGBoost	689.35	118.17	0.296

From these results in Table 2, DL models (LSTM and GRU) significantly outperform traditional ML techniques in terms of lower RMSE, lower MAPE, and higher R^2 scores. Among them, LSTM exhibits the best performance with the lowest RMSE (688.81) and the highest R^2 score (0.592), meaning it explains more variance in the traffic data compared to other models.

On the other hand, traditional ML models struggle to capture complex temporal dependencies. LR has the lowest R^2 score (0.197), indicating that it explains the least variance in traffic patterns. However, it has a relatively lower MAPE (42.80%), meaning it provides more stable predictions in terms of percentage error, though its absolute performance is still weaker than DL models.

RF, SVM, and XGBoost show slightly better R^2 scores (0.266, 0.220, and 0.296, respectively) than LR, but they suffer from extremely high MAPE values (above 100%), suggesting that their predictions have large relative errors compared to the actual traffic values. The high MAPE for these models indicates that while they might capture some traffic trends,

they fail to produce reliable absolute predictions, making them unsuitable for accurate forecasting.

Among all models, LSTM achieves the best balance between RMSE, MAPE, and R^2 score, making it the most effective choice for network traffic prediction.

6. CONCLUSION

This study evaluated various ML techniques for network traffic prediction using real-world LTE data from Kandahar province of Afghanistan. Six models, such as LR, RF, SVM, XGBoost, LSTM, and GRU were compared based on RMSE, MAPE, and R^2 to assess their predictive performance. The results indicate that DL models, such as LSTM and GRU, outperform traditional ML models, effectively capturing temporal dependencies in traffic patterns. LSTM achieved the best performance, making it the most suitable choice for this dataset, while GRU also performed well but slightly below LSTM. In contrast, traditional ML models exhibited lower predictive accuracy with low R^2 scores and high MAPE values, highlighting their limitations in modeling dynamic traffic variations.

Future research could improve prediction accuracy through hyperparameter optimization and hybrid ML-DL approaches. Additionally, incorporating user mobility and network congestion levels could enhance the robustness of traffic forecasting models. This comparative analysis provides a useful reference for selecting appropriate ML techniques in similar datasets, supporting more efficient network management strategies.

REFERENCES

[1] H. Riaz, S. Öztürk, and A. Çalhan, "A Robust Handover Optimization Based on Velocity-Aware Fuzzy Logic in 5G Ultra-Dense Small Cell HetNets," *Electronics*, vol. 13, no. 17, Art. no. 17, Jan. 2024, doi: 10.3390/electronics13173349.

[2] Ericsson, "Mobile data traffic forecast – Mobility Report." Accessed: Feb. 28, 2024. [Online]. Available: https://www.ericsson.com/en/reports-and-papers/mobility-report/dataforecasts/mobile-traffic-forecast

[3] C. Yao, C. Yang, and C.-L. I, "Data-driven resource allocation with traffic load prediction," *J. Commun. Inf. Netw.*, vol. 2, no. 1, pp. 52–65, Mar. 2017, doi: 10.1007/s41650-017-0005-y.

[4] K.-C. Chang, K.-C. Chu, H.-C. Wang, Y.-C. Lin, and J.-S. Pan, "Energy Saving Technology of 5G Base Station Based on Internet of Things Collaborative Control," *IEEE Access*, vol. 8, pp. 32935–32946, 2020, doi: 10.1109/ACCESS.2020.2973648.

[5] O. Aouedi, V. A. Le, K. Piamrat, and Y. Ji, "Deep Learning on Network Traffic Prediction: Recent Advances, Analysis, and Future Directions," *ACM Comput. Surv.*, vol. 57, no. 6, p. 151:1-151:37, Feb. 2025, doi: 10.1145/3703447.

[6] D. Ferreira, A. Braga Reis, C. Senna, and S. Sargento, "A Forecasting Approach to Improve Control and Management for 5G Networks," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1817–1831, Jun. 2021, doi: 10.1109/TNSM.2021.3056222.

[7] X. Wang *et al.*, "A Survey on Deep Learning for Cellular Traffic Prediction," *Intelligent Computing*, vol. 3, p. 0054, Jan. 2024, doi: 10.34133/icomputing.0054.

[8] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, in IJCAI'18. Stockholm, Sweden: AAAI Press, Jul. 2018, pp. 3634–3640.

[9] W. Jiang, "Cellular traffic prediction with machine learning: A survey," *Expert Systems with Applications*, vol. 201, p. 117163, Sep. 2022, doi: 10.1016/j.eswa.2022.117163.

[10] C. Gijón, M. Toril, S. Luna-Ramírez, M. L. Marí-Altozano, and J. M. Ruiz-Avilés, "Long-Term Data Traffic Forecasting for Network Dimensioning in LTE with Short Time Series," *Electronics*, vol. 10, no. 10, Art. no. 10, Jan. 2021, doi: 10.3390/electronics10101151.

[11] C. Zhang, H. Zhang, J. Qiao, D. Yuan, and M. Zhang, "Deep Transfer Learning for Intelligent Cellular Traffic Prediction Based on Cross-Domain Big Data," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1389–1401, Jun. 2019, doi: 10.1109/JSAC.2019.2904363.

[12] X. Cao, Y. Zhong, Y. Zhou, J. Wang, C. Zhu, and W. Zhang, "Interactive Temporal Recurrent Convolution Network for Traffic Prediction in Data Centers," *IEEE Access*, vol. 6, pp. 5276–5289, 2018, doi: 10.1109/ACCESS.2017.2787696.

[13] H. D. Trinh, L. Giupponi, and P. Dini, "Mobile Traffic Prediction from Raw Data Using LSTM Networks," in *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Sep. 2018, pp. 1827–1832. doi: 10.1109/PIMRC.2018.8581000.

[14] S. Jaffry, "Cellular Traffic Prediction with Recurrent Neural Network," Mar. 05, 2020, *arXiv*: arXiv:2003.02807. doi: 10.48550/arXiv.2003.02807.

[15] A. Azzouni and G. Pujolle, "NeuTM: A neural network-based framework for traffic matrix prediction in SDN," in *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*, Apr. 2018, pp. 1–5. doi: 10.1109/NOMS.2018.8406199.

[16] W. Jiang, M. He, and W. Gu, "Internet Traffic Prediction with Distributed Multi-Agent Learning," *Applied System Innovation*, vol. 5, no. 6, Art. no. 6, Dec. 2022, doi: 10.3390/asi5060121.

[17] K. Gao *et al.*, "Predicting Traffic Demand Matrix by Considering Inter-flow Correlations," in *IEEE*

INFOCOM 2020 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Jul. 2020, pp. 165–170. doi: 10.1109/INFOCOMWKSHPS50562.2020.9163001.

[18] D. Aloraifan, I. Ahmad, and E. Alrashed, "Deep learning based network traffic matrix prediction," *International Journal of Intelligent Networks*, vol. 2, pp. 46–56, Jan. 2021, doi: 10.1016/j.ijin.2021.06.002.

[19] H. Riaz, S. Öztürk, S. Aldirmaz-Colak, and A. Çalhan, "A Handover Decision Optimization Method Based on Data-Driven MLP in 5G Ultra-Dense Small Cell HetNets," *J Netw Syst Manage*, vol. 33, no. 2, p. 31, Feb. 2025, doi: 10.1007/s10922-025-09903-6.

[20] D. C. Montgomery, E. A. Peck, and G. G. Vining, *Introduction to Linear Regression Analysis*, Sixth Edition. Wiley, 2021.

[21] P. H. Putra, A. Azanuddin, B. Purba, and Y. A. Dalimunthe, "Random forest and decision tree algorithms for car price prediction," *Jurnal Matematika Dan Ilmu Pengetahuan Alam LLDikti Wilayah 1 (JUMPA)*, vol. 4, no. 1, Art. no. 1, 2024, doi: 10.54076/jumpa.v3i2.305.

[22] C. Comito and C. Pizzuti, "Artificial intelligence for forecasting and diagnosing COVID-19 pandemic: A focused review," *Artificial Intelligence in Medicine*, vol. 128, p. 102286, Jun. 2022, doi: 10.1016/j.artmed.2022.102286.

[23] V. Kurri, V. Raja, and P. Prakasam, "Cellular traffic prediction on blockchain-based mobile networks using LSTM model in 4G LTE network," *Peer-to-Peer Netw. Appl.*, vol. 14, no. 3, pp. 1088–1105, May 2021, doi: 10.1007/s12083-021-01085-7.

[24] A. Oğuz and Ö. F. Ertuğrul, "Chapter 1 - Introduction to deep learning and diagnosis in medicine," in *Diagnostic Biomedical Signal and Image Processing Applications with Deep Learning Methods*, K. Polat and S. Öztürk, Eds., in Intelligent Data-Centric Systems. , Academic Press, 2023, pp. 1–40. doi: 10.1016/B978-0-323-96129-5.00003-2.

[25] H. Riaz, S. Öztürk, S. A. Çolak, and A. Çalhan, "Performance Analysis of Weighting Methods for Handover Decision in HetNets," *Gazi University Journal of Science*, vol. 37, no. 4, pp. 1791–1810, Jan. 2024, doi: 10.35378/gujs.1373452.