



Paper Type: Original Article

Traffic Prediction Using Machine Learning

Anchita Padhy¹, Ladly Swain¹, Ayush Das¹, Aditya Mondal^{1*} , Debapriya Jha¹, Ayush Kumar Bhagat¹

¹ Kalinga Institute of Industrial Technology; 21051284@kiit.ac.in; 21052002@kiit.ac.in; 21051212@kiit.ac.in; 21051277@kiit.ac.in; 21051215@kiit.ac.in; 21051213@kiit.ac.in.

Citation:

Received: 08 April 2024

Revised: 01 July 2024

Accepted: 17 August 2024

Padhy, A., Swain, L., Das, A., Mondal, A., Jha, D., & Bhagat, A. K. (2024). Traffic prediction using machine learning. *Soft computing fusion with applications*, 1(3), 169-180.

Abstract

This research investigates the use of sophisticated machine-learning methods to forecast traffic patterns at various urban intersections. By leveraging a detailed dataset collected from strategically situated sensors, this study applies Gated Recurrent Units (GRUs), a type of Recurrent Neural Network (RNN) designed for sequence prediction, to anticipate traffic volumes. The data includes vehicle counts, environmental factors, and time markers from multiple junctions over an extended period. After thorough data preprocessing, feature engineering, and diligent model training, the research demonstrates how GRUs can effectively manage the temporal and sequential relationships present in traffic flow data. The predictive models were assessed using the Root Mean Square Error (RMSE) metric, which showed notable differences in predictive accuracy at different urban junctions. This study adds to the expanding knowledge in traffic management systems and offers a solid framework for real-time traffic forecasting, with the goal of improving urban mobility and alleviating congestion through smart traffic management solutions.


Keywords: Traffic prediction, Gated recurrent units, Machine learning, Urban traffic management, Time series analysis.

1 | Introduction

Urban traffic management faces escalating challenges due to increasing vehicle populations and limited infrastructure expansion capabilities. Efficiently predicting traffic flows is crucial for planning and real-time traffic management, which can significantly mitigate congestion and enhance urban mobility. This research uses Gated Recurrent Units (GRUs) to predict traffic volumes at various urban intersections, leveraging extensive traffic sensor data collected over multiple years.

The intersection of data science and urban planning presents unique opportunities to enhance traffic systems. By applying Machine Learning (ML) models capable of analyzing large datasets, urban planners and traffic management systems can gain unprecedented insights into traffic patterns. These insights are crucial for developing strategies that preempt traffic congestion and optimize traffic flows [1], [2]. The predictive models

 Corresponding Author: 21051277@kiit.ac.in

 <https://doi.org/10.22105/scfa.v1i3.48>



Licensee System Analytics. This article is an open-access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0>).

used in this study are based on GRUs. They are particularly suited for modeling time series data because they maintain information over time and adaptively forget irrelevant data points.

This study evaluates the effectiveness of GRU models in traffic prediction across different urban junctions, which is assessed through Root Mean Square Error (RMSE). The outcomes reveal the potential and limitations of using advanced machine-learning techniques in urban traffic management. By incorporating real-time data and continually adapting to new patterns, the proposed GRU-based models aim to serve as a cornerstone for developing more intelligent and responsive urban traffic systems. The following sections will delve deeper into the feature engineering process, the specifics of the GRU model implementation, and a detailed analysis of the model's performance across various junctions. The insights gained from this research are intended to guide future improvements in traffic prediction models and support the broader goal of enhancing urban traffic systems globally.

1.1 | Problem Statement

Urban traffic management systems struggle to adapt to dynamic traffic conditions, leading to inefficiencies in congestion management and increased travel times. This project addresses these challenges by developing and assessing advanced ML models, specifically GRUs, to enhance the accuracy and responsiveness of traffic predictions at urban intersections.

1.2 | Research Objectives

The primary objectives of this study are:

- I. Develop and evaluate ML models for traffic prediction to improve accuracy and scalability.
- II. To incorporate a comprehensive range of features in the traffic prediction model, including temporal patterns, environmental conditions, and special events, to reflect the complex dynamics that influence traffic flow.
- III. Evaluation of the GRU-based models across different intersections with varying traffic patterns and conditions. This involves assessing the model's accuracy through metrics such as the RMSE and comparing its performance to traditional traffic prediction models.
- IV. To explore the scalability and adaptability of the model in different urban environments. This includes testing the model's ability to generalize across various data sets and traffic conditions to identify model configurations that optimize performance and computational efficiency. Propose a practical framework for implementing ML-based traffic prediction models within urban management systems.

2 | Literature Review

2.1 | Evolution of Traffic Prediction Methods

The field of traffic prediction has evolved over several decades, moving from basic statistical approaches to sophisticated ML and deep learning models:

- I. Pre-2000: Due to limited computational resources and simpler datasets, the focus was on statistical analysis and linear models (e.g., linear regression). While these models are easy to implement, they lack the flexibility to handle complex traffic patterns.
- II. 2000-2010: Time series models like Autoregressive Integrated Moving Average (ARIMA) began to emerge, providing improved temporal modeling capabilities but still struggled with non-linear relationships inherent in traffic data.
- III. 2010-2015: The rise of ML introduced Support Vector Machines (SVMs) and Random Forests (RFs), which offered better pattern recognition and the ability to capture non-linear relationships, albeit with limited temporal understanding.

- IV. 2015-Present: the development of deep learning has transformed traffic prediction. Models like Long Short-Term Memory (LSTM) networks excel at capturing temporal dependencies, making them ideal for time-dependent data like traffic. Convolutional Neural Network (CNN)-LSTM hybrids and attention-based models have since emerged as powerful tools in the field, though they require high computational resources.

Table 1. Evolution of traffic prediction models.

Period	Primary Methods	Key Characteristics	Limitations
Pre-2000	Statistical analysis	Linear modeling, limited adaptability	Poor handling of complex patterns
2000-2010	Time series models	Improved temporal modeling	Weak in non-linear pattern recognition
2010-2015	ML models (SVM, RF)	Better pattern recognition	Limited in capturing temporal patterns
2015-Present	Deep learning (LSTM)	Superior temporal dependencies	High computational requirements

The evolution of ML in the context of traffic prediction demonstrates a significant trajectory from traditional models to sophisticated deep learning techniques, enabling more dynamic and precise forecasting systems. The incorporation of LSTM networks, as discussed by Azzouni and Pujolle [3], highlights the advanced capabilities of deep learning to handle the sequential and temporal complexities inherent in traffic data, offering refined predictions that are crucial for real-time traffic management systems.

Expanding on this, the research by Vlahogianni et al. [4] explores the integration of genetic algorithms with neural networks, optimizing the prediction models to enhance their accuracy and adaptability in complex traffic scenarios. This methodological innovation is pivotal, as it combines traditional ML techniques with optimization strategies to tackle the intricate dynamics of traffic flows.

Further contributions from Jia et al. [5] emphasize the importance of external factors such as weather conditions, integrating these variables into deep learning models to enhance the predictive accuracy under varying environmental influences. This approach not only refines the predictions but also adapts them to real-world variability, highlighting the adaptive nature of modern traffic prediction models.

Loumiotis et al. [6] extends the application of Artificial Neural Networks (ANNs) to traffic prediction, showcasing their versatility in learning from complex datasets and adapting to traffic conditions. This research underscores the robustness of ANNs in traffic management, providing a solid foundation for future innovations.

Integrating GRU networks into traffic prediction represents a significant advancement in the field. The GRU model, known for its efficiency in modeling time series data, offers an alternative to LSTMs with similar capabilities but often with faster training times and fewer parameters to manage. GRUs are particularly useful when computational efficiency and model simplicity are crucial [7].

In recent studies, researchers like Prajam et al. [8] and Loumiotis et al. [6] have explored the application of ML to predict network traffic, focusing on real-time data processing. They emphasize the role of GRU models in achieving high accuracy in traffic forecasts, thereby aiding in efficient traffic management and planning.

Collectively, these studies paint a comprehensive picture of the current landscape of traffic prediction using ML. From using optimized neural networks to adapting models to include external variables, the field is moving towards more integrated and sophisticated systems. Future research is poised to explore hybrid models that combine multiple ML techniques and data sources further to enhance the accuracy and applicability of traffic prediction models [9]–[12].

Lastly, Loumiotis et al. [6] explores the broader application of ANNs for road traffic prediction. The study emphasizes the flexibility of ANNs in adapting to varying traffic conditions and their capacity to learn from complex datasets, showcasing their practical utility in real-world traffic management systems. These studies demonstrate a trajectory towards more sophisticated, accurate, and adaptive traffic prediction models. They

highlight the importance of incorporating both traditional and novel approaches within ML to enhance traffic management systems' predictive capabilities and applicability. The ongoing advancements suggest that future research should explore hybrid models and integrate external data sources to improve traffic prediction accuracy.

3 | Proposed Framework

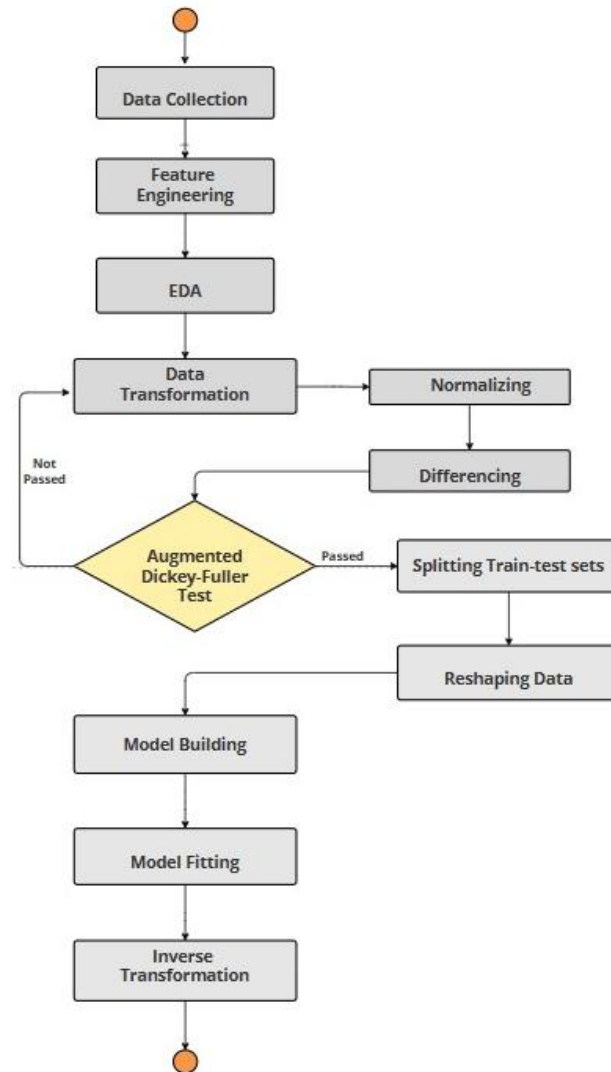


Fig. 1. Framework.

3.1 | Model Development and Deployment

Model development focused on leveraging the preprocessed data to train predictive models that could offer high accuracy and real-time applicability:

- I. Model selection: Based on the insights from the Exploratory Data Analysis (EDA) and the nature of the dataset, GRUs were selected for their effectiveness in handling sequence prediction and efficiency over other complex models like LSTMs.
- II. Model training: GRUs were trained using a robust framework involving multiple hidden layers to adequately capture the deep temporal structures in the data. To prevent overfitting, hyperparameters were optimized through grid search techniques, focusing on learning rates, the number of layers, and dropout rates.

- III. Model evaluation: The trained models were evaluated using RMSE on the test set to assess their predictive accuracy quantitatively. Based on their performance metrics, the models were iteratively refined until an optimal balance of bias and variance was achieved.
- IV. Deployment: The final models were deployed into a production environment where they began making real-time predictions. A monitoring system was also set up to track model performance over time, with mechanisms to retrain the models with new data, ensuring they remained accurate and relevant.

3.2 | Gated Recurrent Unit Neural Network

The GRU is a type of Recurrent Neural Network (RNN) architecture designed to effectively handle sequential data while addressing the limitations of traditional RNNs, such as vanishing and exploding gradients. Proposed by Cho et al. [13], GRUs are a simplified version of LSTM networks with fewer parameters and computational complexity, making them an attractive choice for time-series prediction, natural language processing, and other sequence-based tasks.

Structure of gated recurrent unit

The GRU architecture consists of specialized gating mechanisms that enable it to control the flow of information adaptively. The two primary gates in a GRU are:

- I. Update gate: This gate determines the amount of information from the past that needs to be carried forward to the current time step. It decides how much of the previous memory should be retained or discarded.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z),$$

here, z_t is the update gate vector, W_z represents the weight matrix, x_t is the input at time t , h_{t-1} is the hidden state from the previous time step, b_z is the bias term, and σ is the sigmoid activation function.

- II. Reset gate: This gate decides how much of the previous memory should be ignored in computing the current state. It allows the network to selectively "Forget" past information when it is no longer relevant.

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r),$$

where r_t is the reset gate vector, and the rest of the parameters are analogous to those in the update gate.

Hidden state update

The GRU combines the effects of the update and reset gates to compute the new hidden state h_t . The reset gate first filters the information from the previous hidden state, influencing the computation of a candidate's hidden state:

This update mechanism allows GRUs to capture both short-term and long-term dependencies in the sequence effectively.

$$\tilde{h}_t = \tanh(W_h \cdot [r_t \odot h_{t-1}, x_t] + b_h),$$

here, \tilde{h}_t is the candidate hidden state, and \odot denotes element-wise multiplication. The update gate then blends the candidate hidden state and the previous hidden state to compute the final hidden state:

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t.$$

This update mechanism allows GRUs to effectively capture both short-term and long-term dependencies in the sequence.

Key features of gated recurrent units

- I. Reduced complexity: GRUs have fewer parameters than LSTMs because they lack separate memory cells and use only two gates instead of three. This makes GRUs computationally efficient.
- II. Effective memory management: GRUs balance retaining useful information and forgetting irrelevant details by combining reset and update gates.
- III. Capability to handle vanishing gradients: The gating mechanism mitigates the vanishing gradient problem by allowing gradients to flow unimpeded across time steps.

Applications of gated recurrent units

GRUs are widely used in various domains, including:

- I. Natural language processing: Tasks such as text generation, machine translation, and sentiment analysis leverage GRUs to model sequential word dependencies.
- II. Time-series analysis: Applications like stock price prediction, weather forecasting, and sensor data analysis use GRUs because they can learn patterns over time.
- III. Speech recognition: GRUs process sequential audio data, enabling accurate recognition and synthesis.

Advantages and limitations

Advantages:

- I. Simplified architecture reduces training time and computational cost.
- II. Effective for datasets where long-term dependencies are not as pronounced.
- III. Handles small and moderate-sized datasets efficiently.

Limitations:

- I. It may underperform compared to LSTMs in tasks with complex long-term dependencies.
- II. The reduced complexity may sometimes lead to a loss of representational power.

4 | Experimental Setup

4.1 | Data Collection and Processing

The cornerstone of this study was a comprehensive dataset meticulously collected from an array of sensors strategically located at four major urban intersections. These sensors continuously captured data, providing a rich tapestry of traffic dynamics over multiple years. The primary dataset included several key variables crucial for analyzing traffic flow:

- I. Vehicle counts: Recorded minute-by-minute, these counts are pivotal for assessing traffic density and predicting potential bottlenecks.
- II. Timestamps: Each record is timestamped, which is critical for time-series analysis. This allows the model to detect patterns based on time of day, day of the week, and seasonal variations.
- III. Sensor IDs and locations: Identifying where each data point originated is essential for geospatial analysis and modeling differences in traffic patterns across various parts of the city.
- IV. Environmental conditions: Some sensors also provide weather data, which can significantly influence traffic patterns due to changes in driver behavior under different weather conditions.

This data was initially subjected to rigorous cleaning procedures to rectify any inconsistencies or missing values, ensuring the integrity of the models' inputs. Following cleaning, the data was categorized based on the location to facilitate separate analyses for each junction, recognizing their unique traffic flow characteristics

influenced by local variables such as nearby points of interest, road configurations, and typical congestion levels.

Table 2. Dataset table.

	Date	Time	Junction	Vehicles	Year	Month	Date-no	Hour	Day
0	2015-11-01	00:00:00	1	15	2015	11	1	0	Sunday
1	2015-11-01	01:00:00	1	13	2015	11	1	1	Sunday
2	2015-11-01	02:00:00	1	10	2015	11	1	2	Sunday
3	2015-11-01	03:00:00	1	7	2015	11	1	3	Sunday
4	2015-11-01	04:00:00	1	9	2015	11	1	4	Sunday

4.2 | Feature Engineering

Feature engineering is a critical phase in data analysis, particularly for complex ML tasks like traffic prediction. In this project, feature engineering was conducted meticulously, leveraging insights from the initial EDA to enhance the dataset's predictive power. This section describes the sophisticated approaches used to transform raw traffic data into informative features that could significantly influence the performance of the predictive models.

Temporal features

Understanding and capturing the temporal dynamics were paramount, given the inherent time-related patterns in traffic flow. To this end, several time-based features were extracted from the timestamp data:

- I. Hour of the day: Traffic patterns vary significantly throughout the day. Capturing the hour when the data was recorded helps the model learn hour-specific traffic patterns, such as morning and evening rush hours.
- II. Day of the week: This feature helps to distinguish weekdays from weekends, which typically have different traffic flows. Additionally, it captures the variability in traffic patterns from Monday through Sunday.
- III. Month of the year: Seasonal trends are prominent in traffic data due to varying weather conditions, holidays, and other seasonal activities. Including the month as a feature allows the model to adjust its predictions based on seasonal changes.
- IV. Public holidays: Traffic during public holidays does not follow the typical daily or weekly pattern. A binary indicator for public holidays was created to help the model recognize and adjust for these anomalies.

Lagged features and rolling windows

Lagged features were created to capture the autocorrelation in the traffic data.

4.3 | Exploratory Data Analysis

EDA was integral in understanding the underlying patterns and potential anomalies within the traffic data, setting the stage for effective model design. The EDA focused on visual and statistical analyses to identify trends, seasonality, and irregularities.

The insights gained from the EDA informed the subsequent data preprocessing and feature engineering phases, ensuring that the models developed were data-driven and contextually aware of the traffic dynamics at each junction.

4.4 | Data Preprocessing

Data preprocessing involved transforming raw data into a clean, model-ready format, a critical step to ensure the accuracy of the predictive models:

- I. Cleaning: the dataset was cleansed of erroneous entries, and missing values were imputed based on nearest neighbor techniques, ensuring that the integrity of the time series was maintained.

- II. Feature engineering: New features, such as time lags and rolling averages, were engineered from the existing data, which are significant for capturing temporal dependencies in traffic patterns. Boolean features indicating weekends, holidays, and special event days were added to account for their unique traffic impacts.
- III. Data transformation: Logarithmic transformations were applied to highly skewed data distributions to normalize the influence of outliers and reduce skewness. This step helped stabilize variance, a requirement for many linear-based model assumptions.
- IV. Train-test split: The data was consistently split into training and testing sets in a time series. This approach ensured the models were validated on unseen, future data, mimicking real-world forecasting scenarios.

4.5 | Techstack Used

The project leveraged a carefully selected suite of technologies tailored to efficiently handle the complexities of processing large-scale traffic data, model development, and deployment. Below are the key components of the technology stack used:

- I. Python: chosen as the core programming language due to its extensive ecosystem for data science, including numerous libraries geared explicitly towards data manipulation and ML.
- II. Pandas and NumPy: These libraries provided robust tools for data manipulation and numerical calculations, essential for managing large datasets and performing extensive data preprocessing.
- III. SciPy: Utilized for its advanced scientific computing capabilities, including statistical operations necessary for conducting the Augmented Dickey-Fuller and other data tests.
- IV. Scikit-learn: This ML library facilitated various data preprocessing steps, model evaluations, and the implementation of supplementary ML techniques for comparative studies.

5 | Experimental Results and Discussion

5.1 | System Architecture

This section presents the outcomes of the traffic prediction model applied to four junctions, analyzing the model's accuracy and effectiveness in real-world scenarios. The results are critically assessed using RMSE as the primary statistical metric to quantify prediction errors, which provides a clear measure of model performance.

Results of the model

The predictive model developed using the GRU network was evaluated based on its ability to forecast traffic volumes accurately. The model was trained and tested across four distinct junctions with different traffic patterns and volumes. The RMSE values obtained for each junction are as shown in *Table 3*.

Table 3. Root mean square error values for each junction.

	Junction	RMSE
0	Junction 1	0.246742
1	Junction 2	0.551179
2	Junction 3	0.606312
3	Junction 4	0.997704

These values indicate the model's performance varied significantly across the junctions, with Junction 1 showing the best results, suggesting high accuracy and minimal prediction errors. Conversely, Junction 4 exhibited the highest RMSE, indicating less accuracy, which could be attributed to fewer data points available for training due to its recent operational status.

5.2 | Analysis of Predictive Accuracy

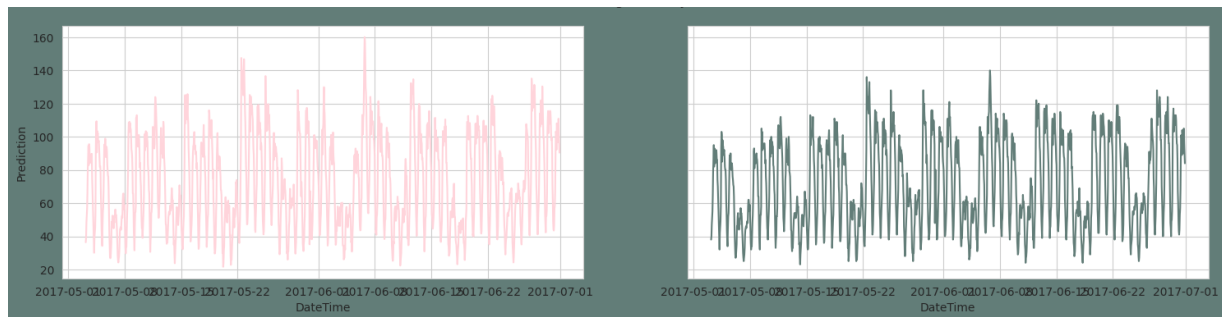
The variance in RMSE across the junctions can be attributed to several factors:

- I. Data volume and quality: Junction 1 had the most comprehensive data set, which likely contributed to the model's high accuracy. In contrast, Junction 4 had limited data, affecting the model's learning capability.
- II. Complexity of traffic patterns: Junctions with more predictable, less volatile traffic patterns (Like Junction 1) tended to have lower RMSE scores. Junctions with more complex patterns influenced by variables not fully captured in the model (Like Junction 4) showed higher errors.

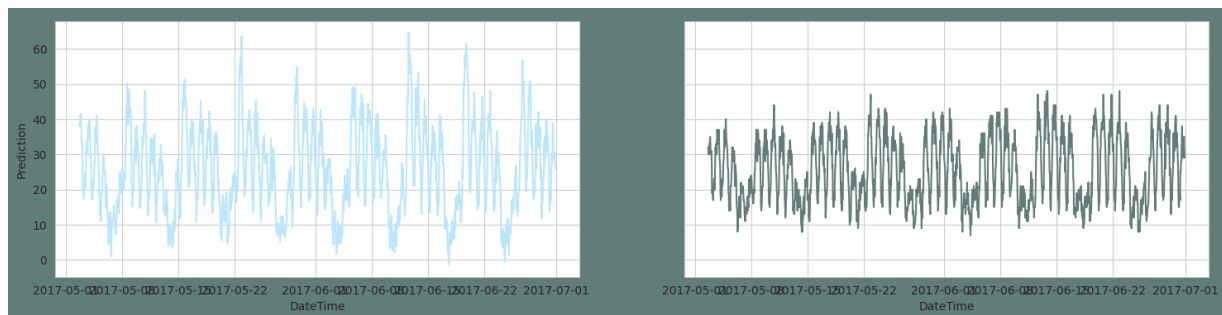
Inverse transformation results

An essential aspect of model validation was applying inverse transformations to the predicted results. This process involved reversing the differencing and normalization steps applied during data preprocessing to return the forecasted values to their original scale and context. This step was crucial for practical applications, allowing traffic management teams to interpret the predictions within the operational traffic volumes and patterns.

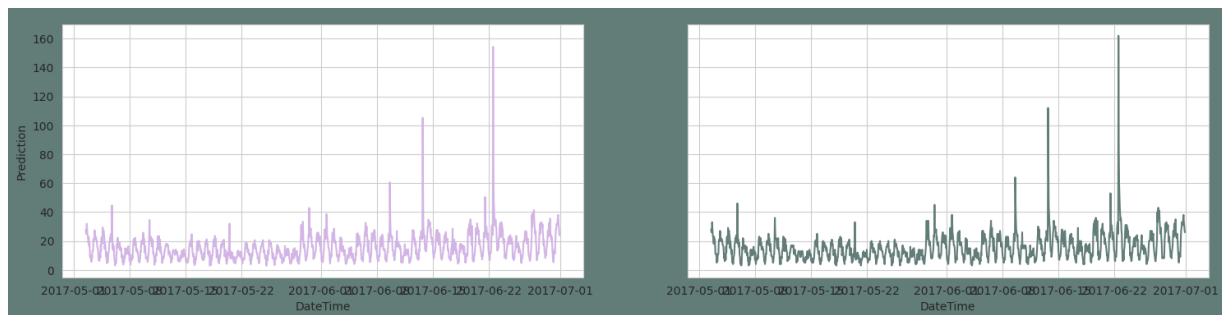
The comparative plots between the original and predicted values post-inverse transformation visually highlighted the model's accuracy, confirming the quantitative assessments made with RMSE. These plots also helped identify specific periods where the model under or over-predicted traffic volumes, providing insights into potential improvements in model training and feature engineering.



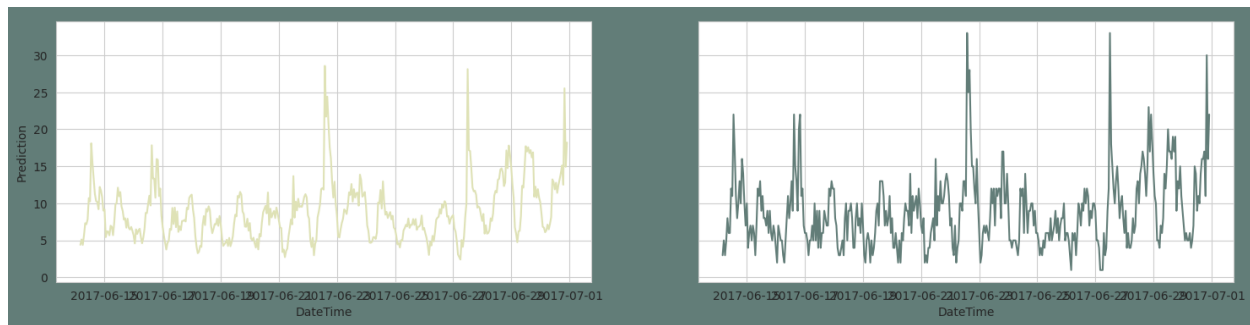
a.



b.



c.



d.

Fig. 2. Inverse transformation results; a. predictions and originals for Junction 1, b. predictions and originals for Junction 2, c. predictions and originals for Junction 3, and d. predictions and originals for Junction 4.

5.3 | Model Performance Comparison

The GRU model was selected for its proficiency in handling sequence prediction tasks. GRUs are RNNs optimized for time series data like traffic flows, where past conditions significantly influence future states. Features that make the GRU advantageous for this application include:

- I. Update and reset gates: These mechanisms help the model decide which information is essential to carry forward and which can be discarded, enhancing the model's ability to manage data over different time scales.
- II. Efficiency over LSTM: GRUs streamline some operations in more complex models like LSTMs by combining the input and forget gates into a single update gate. This reduction in complexity leads to faster training times and lower computational overhead without a substantial sacrifice in performance.
- III. Adaptive memory capabilities: GRUs provide a flexible approach to managing memory, allowing them to dynamically retain or forget information, which is critical for predicting traffic patterns that vary significantly over time.

The GRU was implemented with Backpropagation Through Time (BPTT), a technique well-suited for training models on time-series data. This approach ensures that the network adjusts its weights based on both recent and more extended past data, optimizing its predictions for future traffic flows.

The combination of a sophisticated technology stack and the GRU model provided a robust framework for addressing the challenges of traffic prediction, facilitating improvements in urban traffic management, and contributing to more efficient, safer city environments.

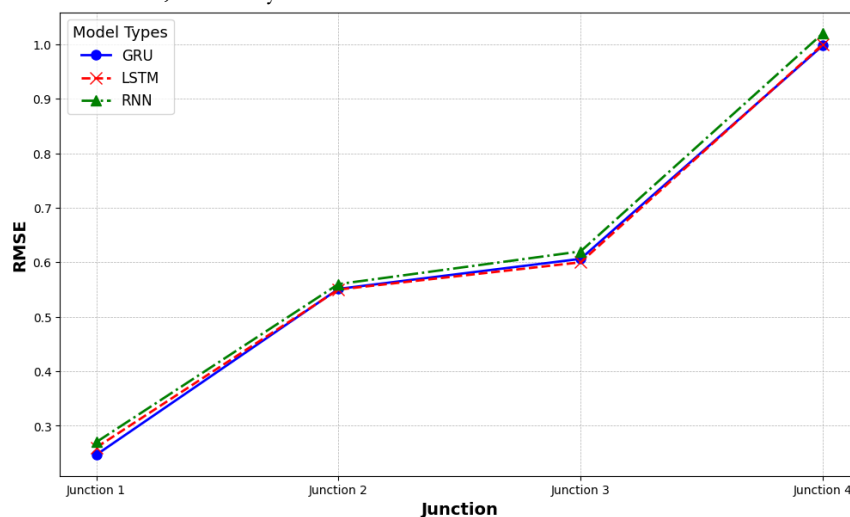


Fig. 3. Comparison of root mean square error value across models.

6 | Conclusion

This study utilized a GRU neural network to predict traffic flow at four junctions, each demonstrating unique data characteristics and traffic behaviors. The project harnessed extensive data preprocessing techniques, including normalization and differencing, to ensure the time series data was stationary, a prerequisite for effective model performance. The GRU model was chosen for its proficiency in handling sequential data and its capacity to manage temporal dependencies intrinsic to traffic flow data. The results, evidenced through the RMSE values across the junctions, affirm the model's capability to predict traffic volumes with varying degrees of accuracy, with Junction 1 showing the best performance and Junction 4 the least.

6.1 | Future Work and Improvements

To further advance the capabilities of traffic prediction models, various enhancements and research directions are being considered:

- I. Incorporating additional data sources: future project iterations should integrate more comprehensive data sets, including real-time traffic data, weather conditions, and event schedules, to enhance the model's predictive accuracy and robustness.
- II. Model enhancement: Exploring more complex models or hybrid approaches that combine several neural network architectures could improve predictive accuracy. Techniques such as feature engineering, including time of day, day of the week, and local events, could also be beneficial.
- III. Real-time data processing: Implementing a system for real-time data processing and prediction could help in dynamic traffic management, allowing city planners and traffic controllers to make more informed decisions promptly.
- IV. Feedback loops: Developing a feedback system in which the model's predictions are continually updated and refined based on new data could help create a more adaptive and resilient traffic prediction system.

6.2 | Challenges and Limitations

- I. Data availability and quality: The limited data for some junctions, especially Junction 4, significantly hampered the model's ability to learn effectively. The absence of long-term historical data for all junctions poses a challenge in training more robust models.
- II. Model sensitivity: The model's performance varied across different junctions, indicating a sensitivity to the underlying traffic patterns and volume discrepancies. This suggests a need for more adaptive, localized modeling approaches.
- III. External factors: The model does not incorporate external factors such as weather conditions, accidents, or roadworks, which can significantly influence traffic patterns and lead to prediction inaccuracies.

References

- [1] Mohapatra, H., & Rath, A. K. (2021). An IoT based efficient multi-objective real-time smart parking system. *International journal of sensor networks*, 37(4), 219–232. <https://doi.org/10.1504/IJSNET.2021.119483>
- [2] Mohapatra, H., Rath, A. K., & Panda, N. (2022). IoT infrastructure for the accident avoidance: An approach of smart transportation. *International journal of information technology*, 14(2), 761–768. <https://doi.org/10.1007/s41870-022-00872-6>
- [3] Azzouni, A., & Pujolle, G. (2017). A long short-term memory recurrent neural network framework for network traffic matrix prediction. <https://doi.org/10.48550/arXiv.1705.05690>
- [4] Vlahogianni, E. I., Karlaftis, M. G., & Golias, J. C. (2005). Optimized and meta-optimized neural networks for short-term traffic flow prediction: A genetic approach. *Transportation research part c: Emerging technologies*, 13(3), 211–234. <https://doi.org/10.1016/j.trc.2005.04.007>
- [5] Jia, Y., Wu, J., & Xu, M. (2017). Traffic flow prediction with rainfall impact using a deep learning method. *Journal of advanced transportation*, 2017(1), 6575947. <https://doi.org/10.1155/2017/6575947>

- [6] Loumiotis, I., Demestichas, K., Adamopoulou, E., Kosmides, P., Asthenopoulos, V., & Sykas, E. (2018). Road traffic prediction using artificial neural networks. *2018 south-eastern european design automation, computer engineering, computer networks and society media conference (SEEDA-CECNSM)* (pp. 1–5). IEEE. <https://doi.org/10.23919/SEEDA-CECNSM.2018.8544943>
- [7] Huang, R., Huang, C., Yubao Liu, Y., Dai, G., & Kong, W. (2020). LSGCN: Long short-term traffic prediction with graph convolutional networks. *Proceedings of the twenty-ninth international joint conference on artificial intelligence* (pp. 2327–2333). IJCAI. <http://dx.doi.org/10.24963/ijcai.2020/322>
- [8] Prajam, S., Wechtaisong, C., & Khan, A. A. (2022). Applying machine learning approaches for network traffic forecasting. *Indian journal of computer science and engineering*, 13(2), 324–335. <https://doi.org/10.21817/indjcse/2022/v13i2/221302188>
- [9] Formosa, N., Quddus, M., Ison, S., Abdel-Aty, M., & Yuan, J. (2020). Predicting real-time traffic conflicts using deep learning. *Accident analysis & prevention*, 136, 105429. <https://doi.org/10.1016/j.aap.2019.105429>
- [10] Neelakandan, S., Berlin, M. A., Tripathi, S., Devi, V. B., Bhardwaj, I., & Arulkumar, N. (2021). IoT-based traffic prediction and traffic signal control system for smart city. *Soft computing*, 25(18), 12241–12248. <https://doi.org/10.1007/s00500-021-05896-x>
- [11] Rzeszótko, J., & Nguyen, S. H. (2012). Machine learning for traffic prediction. *Fundamenta informaticae*, 119(3–4), 407–420. <https://doi.org/10.3233/FI-2012-745>
- [12] Li, C., & Xu, P. (2021). Application on traffic flow prediction of machine learning in intelligent transportation. *Neural computing and applications*, 33(2), 613–624. <https://doi.org/10.1007/s00521-020-05002-6>
- [13] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). *Learning phrase representations using RNN encoder-decoder for statistical machine translation*. <https://doi.org/10.48550/arXiv.1406.1078>