

High-performance traffic volume prediction: An evaluation of RNN, GRU, and CNN for accuracy and computational trade-offs

Andri Pranolo^{1*}, Shoffan Saifullah^{2,3}, Agung Bella Utama⁴, Aji Prasetya Wibawa⁴, Muhammad Bastian⁵, Cicin Hardiyanti P⁶

¹ Department of Informatics, Faculty of Industrial Technology, Universitas Ahmad Dahlan, Yogyakarta 55166 Indonesia

² Department of Informatics, Universitas Pembangunan Nasional Veteran Yogyakarta, Yogyakarta 55281, Indonesia

³ Faculty of Computer Science, AGH University of Krakow, Krakow 30-059, Poland

⁴ Department of Electrical Engineering and Informatics, Faculty of Engineering, Universitas Negeri Malang, Jl. Semarang no. 5, Malang 65145, Indonesia

⁵ Scientific Publication, Universitas Ahmad Dahlan, Yogyakarta 55166 Indonesia

⁶ Association for Scientific Computing Electrical and Engineering, Jl. Raya Janti No.130B, Karang Janbe, Karangjambe, Kec. Banguntapan, Kabupaten Bantul, Daerah Istimewa Yogyakarta 55198, Indonesia

Abstract. Predicting urban traffic volume presents significant challenges due to complex temporal dependencies and fluctuations driven by environmental and situational factors. This study addresses these challenges by evaluating the effectiveness of three deep learning architectures—Recurrent Neural Network (RNN), Gated Recurrent Unit (GRU), and Convolutional Neural Network (CNN)—in forecasting hourly traffic volume on Interstate 94. Using a standardized dataset, each model was assessed on predictive accuracy, computational efficiency, and suitability for real-time applications, with Mean Absolute Percentage Error (MAPE), Root Mean Square Error (RMSE), R^2 coefficient, and computation time as performance metrics. The GRU model demonstrated the highest accuracy, achieving a MAPE of 2.105%, RMSE of 0.198, and R^2 of 0.469, but incurred the longest computation time of 7917 seconds. Conversely, CNN achieved the fastest computation time at 853 seconds, with moderate accuracy (MAPE of 2.492%, RMSE of 0.214, R^2 of 0.384), indicating its suitability for real-time deployment. The RNN model exhibited intermediate performance, with a MAPE of 2.654% and RMSE of 0.215, reflecting its limitations in capturing long-term dependencies. These findings highlight crucial trade-offs between accuracy and efficiency, underscoring the need for model selection aligned with specific application requirements. Future work will explore hybrid architectures and optimization strategies to enhance further predictive accuracy and computational feasibility for urban traffic management.

* Corresponding author: andri.pranolo@tif.uad.ac.id

1 Introduction

As urban areas expand, effective traffic forecasting has become essential for managing transportation networks in real-time, improving traffic flow, reducing congestion, and enhancing urban safety [1,2]. Predicting traffic volumes is inherently challenging due to temporal dependencies, non-linearity, and data complexity. Traditional statistical models, while helpful, often fail to capture these complexities, especially in high-dimensional, time-dependent datasets such as traffic volume records [3].

A primary challenge in traffic prediction is managing long-term dependencies in sequential data [4]. Although Recurrent Neural Networks (RNNs) are well-suited for handling sequences, they struggle with retaining long-term information due to the vanishing gradient problem. Gated architectures like Long Short-Term Memory (LSTM) [5] and Gated Recurrent Units (GRU) address this with mechanisms to preserve essential information over extended sequences [6]. However, they come with high computational costs, limiting their practicality for real-time applications [7].

Another significant issue is balancing prediction accuracy with computational efficiency, which is crucial for real-time forecasting [8]. Convolutional Neural Networks (CNNs) have shown promise in time series tasks due to their capacity for local pattern extraction [9]. However, CNNs are limited in capturing long-term dependencies compared to gated RNN models. While previous research has explored CNN, RNN, and GRU applications for time series tasks, comprehensive evaluations specifically for traffic volume prediction remain limited [10,11].

Recent advancements in traffic forecasting through deep learning have led to notable improvements. [12] applied RNNs to short-term traffic speed prediction, achieving moderate success in capturing immediate patterns but showing limited long-term dependency handling. [13] improved this by employing LSTMs, achieving enhanced accuracy in capturing extended dependencies, albeit with high computational demands. CNNs have also been applied to improve local feature extraction efficiency. [14] demonstrated CNN's ability to capture spatially structured patterns in traffic data, though limitations in memory retention reduce their suitability for sequential applications. [15] compared CNNs and LSTMs, finding that LSTMs capture long-term dependencies more effectively. CNNs offer computational efficiency, advantageous for real-time applications with minor accuracy trade-offs.

These advancements reflect two directions: gated architectures like LSTM and GRU for enhanced memory retention and CNNs for efficient pattern extraction. However, current research lacks a comparative analysis of these models regarding predictive accuracy and computation time in metro traffic volume prediction. This study addresses this gap by benchmarking RNN, GRU, and CNN models to explore trade-offs between accuracy and computational efficiency across these architectures.

This article presents a comparative analysis of RNN, GRU, and CNN architectures for metro traffic volume prediction, providing a side-by-side evaluation to determine each model's suitability for real-time forecasting. The primary contributions are as follows:

- An in-depth comparison of RNN, GRU, and CNN models, highlighting performance in Mean Absolute Percentage Error (MAPE), Root Mean Square Error (RMSE), and computation time.
- This study addresses the often-overlooked aspect of processing efficiency crucial for real-time applications by analyzing computational time.
- The findings offer valuable guidance for selecting a model based on trade-offs between accuracy and computational speed. These findings are valuable for urban planners, smart city developers, and traffic management systems.

The article is organized as follows: Section 2 reviews related works and the evolution of deep learning in traffic forecasting. Section 3 outlines the methodology, including dataset

specifications and model configurations. Section 4 presents and discusses the experimental results, comparing model accuracy and computation time performance. Finally, Section 5 concludes the study, summarizing findings and suggesting directions for future research to further enhance traffic prediction through hybrid or optimized deep learning models.

2 Related Works

Traffic volume prediction is crucial for urban planning and real-time traffic management [2]. It has evolved from traditional statistical methods like ARIMA and SVM to advanced deep learning models capable of handling complex temporal dependencies and high-dimensional data [16,17]. Despite their robustness with linear and stationary datasets [18], traditional approaches often struggle with the non-linear and dynamic nature of traffic data [19], prompting a shift towards more flexible deep learning techniques [20].

Recurrent Neural Networks (RNNs) have become foundational for sequential data processing [21], excelling in short-term dependency tasks [22]. However, RNNs suffer from the vanishing gradient problem, limiting their ability to capture long-term dependencies [23]. Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU) were introduced to address this, incorporating gating mechanisms to retain relevant information over extended sequences [24]. While LSTMs are effective for long-term memory retention, they are computationally intensive [25]. GRUs offer a more efficient alternative, achieving similar accuracy with reduced training times, but remain more resource-demanding than traditional RNNs [26].

Convolutional Neural Networks (CNNs), known for efficiently extracting local features, have also been applied to time series forecasting [27–30]. CNNs excel in real-time scenarios due to their computational speed but cannot retain long-term dependencies as effectively as RNN-based models [10]. Comparative studies, such as [31], have highlighted that while LSTMs and GRUs are superior for long-term dependency tasks. CNNs offer faster performance, making them suitable for applications where speed is prioritized over perfect accuracy [32].

Despite these advancements, comprehensive benchmarking of RNN, GRU, and CNN models for traffic volume prediction remains limited [33]. This study fills this gap by systematically evaluating these models, providing insights into the trade-offs between accuracy and computational efficiency in urban traffic forecasting.

3 Method

This section describes the methodology used to evaluate the effectiveness of Recurrent Neural Network (RNN), Gated Recurrent Unit (GRU), and Convolutional Neural Network (CNN) models for metro traffic volume prediction. We outline the dataset, data preprocessing steps, model architectures, and evaluation metrics. Each model was trained and tested on the same dataset, allowing for a fair comparison of predictive accuracy and computational efficiency.

3.1 Dataset

The Metro Interstate Traffic Volume dataset [34], sourced from the UCI Machine Learning Repository, contains 48,204 hourly traffic records from 2012 to 2018, collected on westbound I-94 in Minneapolis-St. Paul, MN. This multivariate, sequential dataset includes eight features capturing temporal, weather, and holiday data, making it ideal for regression-based traffic forecasting tasks. Key features include temperature, rain, snow, cloud cover,

holiday indicators, categorical weather descriptors, and timestamp data. Traffic volume serves as the target variable, while contextual features such as holiday (U.S. and regional events), temp (average temperature in Kelvin), rain_1h and snow_1h (precipitation in mm), clouds_all (cloud cover percentage), and weather_main/ weather_description (weather descriptors) provide essential environmental factors. With no missing values, this dataset is well-suited for modeling the impact of external conditions on traffic flow, providing a comprehensive basis for time-series analysis.

3.2 Data Preprocessing

Data preprocessing was done to optimize the dataset for time-series modeling and enhance model performance. The original dataset contains nine attributes: holiday, temp, rain_1h, snow_1h, clouds_all, weather_main, weather_description, date_time, and traffic_volume. To streamline the data and focus on relevant features, non-numeric and redundant columns (holiday, weather_main, weather_description, and date_time) were removed, resulting in a dataset with five key attributes: temp, rain_1h, snow_1h, clouds_all, and traffic_volume. This selection focuses on continuous features that provide meaningful quantitative data for the model, eliminating categorical and descriptive fields that may introduce unnecessary complexity.

Following feature selection, Min-Max normalization was applied to scale each continuous feature to a range between 0 and 1 [35], which helps stabilize gradient descent and improve convergence during model training. For a given feature x , Min-Max normalization is defined as (1).

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (1)$$

This normalization strategy allows comparability across features with varying scales, a crucial factor in deep learning models sensitive to input magnitudes [36]. Categorical variables, specifically holiday and weather descriptors, were converted using one-hot encoding. This transformation enables the model to interpret categorical information effectively without imposing an ordinal relationship on these features, which could otherwise introduce bias.

The dataset was split into training, validation, and test sets in an 80-10-10 ratio to maintain temporal integrity in the sequential data. This division allows for robust model evaluation, where the validation set aids in hyperparameter tuning, and the test set provides an unbiased assessment of model generalization. By preserving the chronological order of the data, the preprocessing ensures that time-based dependencies, essential for accurate traffic volume prediction, are fully retained across the training and evaluation phases. This preprocessing pipeline effectively standardizes the dataset for modeling while enhancing interpretability and performance in deep learning architectures.

3.3 Model Architectures

This study employs three deep learning architectures—Recurrent Neural Networks (RNN), Gated Recurrent Units (GRU), and Convolutional Neural Networks (CNN)—to predict traffic volume on Interstate 94. Each architecture is selected for its unique strengths in time-series analysis, enabling a comprehensive evaluation of their effectiveness in capturing temporal dependencies in traffic data. Figures illustrating the architectures of the RNN, GRU, and CNN models are provided for reference.

3.3.1 Recurrent Neural Network (RNN)

The Recurrent Neural Network (RNN) model, shown in Figure 1, is structured to capture sequential dependencies by maintaining a hidden state across timesteps [37]. Each RNN layer processes an input sequence step-by-step, where each hidden state h_t at timestep t is updated as a function of both the current input x_t and the previous hidden state h_{t-1} , as defined (2).

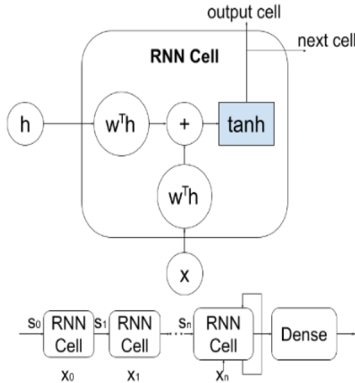


Fig. 1. RNN Architecture – This figure illustrates the standard RNN structure, where each recurrent layer processes sequential data to update hidden states, followed by dense layers for final prediction..

$$h_t = \tan h (W_h h_{t-1} + W_x x_t + b) \tag{2}$$

where W_h and W_x are the weight matrices, and b is the bias term. The $\tan h$ activation function is employed to regulate the values of h_t within a manageable range, mitigating gradient issues. This architecture comprises three recurrent layers with 50 units each, followed by two dense layers for final prediction. The primary limitation of RNNs is their struggle with retaining long-term dependencies due to the vanishing gradient problem, which is addressed in more advanced models like GRU and LSTM.

3.3.2 Gated Recurrent Unit (GRU)

The GRU model (Figure 2) enhances the basic RNN by introducing gating mechanisms—specifically, the update and reset gates—that allow it to selectively update or reset parts of the hidden state, making it more effective at preserving long-term dependencies [6]. The GRU cell at each timestep t is defined by (3)-(5).

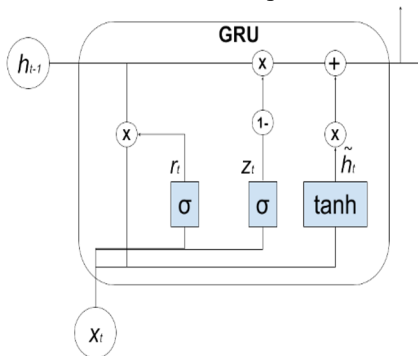


Fig. 2. GRU Architecture – The GRU model incorporates update and reset gates in each layer, enabling efficient handling of long-term dependencies while maintaining training efficiency.

$$z_t = \sigma(W_z x_t + U_z h_{t-1}) \tag{3}$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1}) \tag{4}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tan h(W_h x_t + r_t \odot (U_h h_{t-1})) \tag{5}$$

where z_t and r_t represent the update and reset gates, respectively, σ is the sigmoid function, and \odot denotes element-wise multiplication. By leveraging these gates, GRU effectively mitigates the vanishing gradient issue and enhances the model's capability to capture both short-term and long-term dependencies in traffic data. The GRU architecture consists of three layers with 50 units each, followed by two dense layers. This configuration enables the model to capture intricate temporal patterns in traffic volumes while improving training efficiency compared to LSTMs.

3.3.3 Convolutional Neural Network (CNN)

In contrast to RNN and GRU, which are sequential models, the Convolutional Neural Network (CNN) (Figure 3) uses convolutional layers to extract localized patterns in time-series data [38,39]. The CNN model begins with a one-dimensional convolutional layer with 64 filters and a kernel size of 3, applying the Rectified Linear Unit (ReLU) activation function, which outputs zero for negative values to introduce non-linearity while maintaining computational efficiency. Three GRU layers follow the convolutional layer, enabling the model to benefit from local pattern extraction and sequential processing. A final set of two dense layers generates the predicted traffic volume.

The CNN architecture's reliance on convolutional filters enables it to process patterns within a short temporal window efficiently, making it computationally efficient for tasks requiring rapid predictions. However, CNNs cannot generally retain information over extended sequences, which may limit their performance in applications requiring long-term temporal understanding. In this architecture, combining CNN and GRU layers aims to balance the CNN's local pattern extraction strengths with the GRU's sequential processing capability.

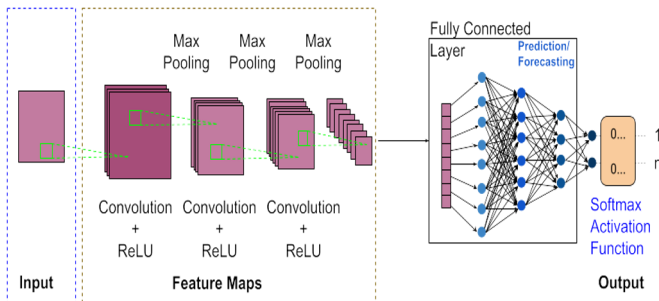


Fig. 3. CNN Architecture – The CNN model initiates with a convolutional layer for local pattern extraction, followed by GRU layers to capture temporal dependencies, culminating in dense layers for final predictions.

Each model is trained with identical hyperparameters (e.g., 100 epochs, learning rate adjustments via the Adam optimizer) and evaluated using standardized metrics, ensuring a consistent basis for performance comparison. This multi-architecture approach allows us to systematically assess each model's trade-offs in accuracy and computational efficiency when applied to metro traffic volume prediction.

3.4 Training Process and Parameter Settings

A standardized training process and parameter settings were applied to ensure a fair and consistent evaluation across the RNN, GRU, and CNN models. Each model was trained on the same dataset with identical hyperparameters to assess its relative effectiveness and computational efficiency objectively. The Adam optimizer was used for all models, chosen for its adaptive learning rate capabilities, which improve convergence speed and stability in complex neural networks.

All models were trained for 100 epochs, allowing sufficient time for convergence while maintaining computational feasibility. The batch size was 64, balancing memory efficiency with gradient stability. The training and validation data were used to monitor model performance, with validation accuracy and loss recorded after each epoch to prevent overfitting and to adjust training strategies if necessary. Early stopping was considered but not applied to allow each model to reach full convergence within the designated epoch range.

The Mean Squared Error (MSE) was selected as the loss function due to its suitability for regression tasks [40], penalizing more significant errors more heavily and enhancing prediction accuracy. MSE is defined as (6).

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (6)$$

where y_i is the actual traffic volume, \hat{y}_i is the predicted traffic volume, and n represents the number of samples. MSE directly measures the average squared difference between predictions and actual values, making it ideal for continuous output predictions such as traffic volume.

The parameter settings for each model are listed in Table 1. While each model architecture differs in terms of layers and structure, core hyperparameters like the optimizer, batch size, and epoch count were kept consistent to compare performance and computational efficiency directly.

In the RNN and GRU models, each recurrent layer consists of 50 units, with a Tanh activation function to control the range of values and mitigate gradient issues. The CNN model begins with a one-dimensional convolutional layer with a kernel size of 3 and 64 filters, using the Rectified Linear Unit (ReLU) activation to enhance non-linearity without significant computational costs. The subsequent GRU layers in the CNN model contribute sequential processing capabilities, aiming to balance feature extraction with temporal dependence handling.

Table 1. Parameter Settings for RNN, GRU, and CNN Models in Traffic Volume Prediction.

Model	Layer Type	Units /Filters	Activation Function	Loss Function	Optimizer	Epochs	Additional Layers
RNN	Simple RNN	50	Tanh	MSE	Adam	100	3 Simple RNN layers, 2 Dense
GRU	GRU	50	Tanh	MSE	Adam	100	3 GRU layers, 2 Dense
CNN	Conv1D (kernel size=3)	64	ReLU	MSE	Adam	100	1 Conv1D, 3 GRU layers, 2 Dense

This uniform approach to parameter settings across all models supports a rigorous evaluation of their strengths in terms of both accuracy and efficiency, allowing for a comprehensive comparison of model suitability in real-time traffic prediction tasks.

3.5 Training Process and Parameter Settings

To evaluate the performance of the RNN, GRU, and CNN models in predicting traffic volume, we used three core metrics: Mean Absolute Percentage Error (MAPE), Root Mean Square Error (RMSE), and R^2 (R-squared) [40]. Additionally, computation time was recorded to assess each model's real-time applicability.

3.5.1 Mean Absolute Percentage Error (MAPE)

MAPE measures the model's prediction accuracy by calculating the average absolute error as a percentage of actual values, providing a relative measure of error. MAPE is defined as (7).

$$MAPE = \frac{1}{n} \sum_{i=1}^n \left| \frac{y_i - \hat{y}_i}{y_i} \right| \times 100 \quad (7)$$

where y_i represents the actual traffic volume, \hat{y}_i is the predicted traffic volume, and n is the total number of observations. By expressing error as a percentage, MAPE provides an intuitive understanding of the model's accuracy relative to the scale of the traffic data, making it particularly useful when comparing predictions across varying magnitudes of traffic volume. Lower MAPE values indicate higher predictive accuracy, with values under 10% generally considered highly accurate in forecasting contexts.

3.5.2 Root Mean Square Error (RMSE)

RMSE measures the average magnitude of error by calculating the square root of the average squared differences between predicted and actual values. RMSE is defined as (8).

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \quad (8)$$

RMSE heavily penalizes more significant errors, making it sensitive to outliers and helpful in understanding the spread of prediction errors. A lower RMSE value indicates that the model's predictions are closer to the actual values, making RMSE a valuable metric for assessing model accuracy on an absolute scale. RMSE's emphasis on larger errors is significant in traffic forecasting, as significant deviations in predicted traffic volumes could lead to suboptimal traffic management decisions in real-world applications.

3.5.3 R-squared (R^2)

The R^2 metric, also known as the coefficient of determination, indicates how well the model's predictions approximate the actual values by explaining the proportion of variance in the target variable that the model captures. The R^2 value is computed as (9).

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2} \quad (9)$$

where \bar{y} is the mean of the actual traffic volume values, and an R^2 value closer to 1 indicates that the model has effectively captured the variance in the data. In contrast, values

closer to 0 indicate a poor fit. For traffic volume prediction, a high R^2 value suggests that the model can accurately account for fluctuations in traffic, reflecting its potential utility in real-time forecasting tasks where capturing data variability is critical.

To evaluate the real-time applicability of each model, the computation time (in seconds) required for training and inference was recorded. Computation time is particularly relevant in traffic forecasting scenarios where timely predictions are essential [3]. Lower computation times imply that a model is more efficient and suitable for applications where rapid forecasting is critical, such as live traffic management systems. This metric complements the accuracy-focused metrics (MAPE, RMSE, and R^2) by providing insight into the operational feasibility of each model, especially for large-scale, high-frequency traffic data.

4 Results and Discussion

This section provides an in-depth analysis of the performance of the RNN, GRU, and CNN models in forecasting traffic volume, focusing on their accuracy, generalization ability, and computational efficiency. We present quantitative results through Mean Absolute Percentage Error (MAPE), Root Mean Square Error (RMSE), R^2 , and computation time, as shown in Table 4. Additionally, we analyze each model’s learning process through training and validation loss curves (Figures 4) and evaluate the accuracy of their predictions in terms of actual versus predicted traffic volume (Figures 5).

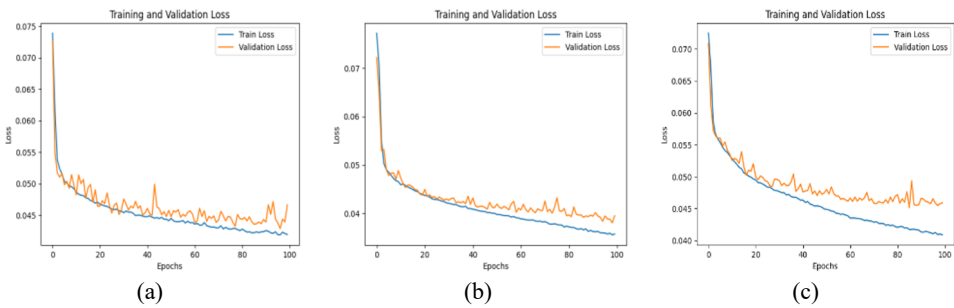


Fig. 4. Training and Validation Loss of (a) RNN, (b) GRU, and (c) CNN Models.

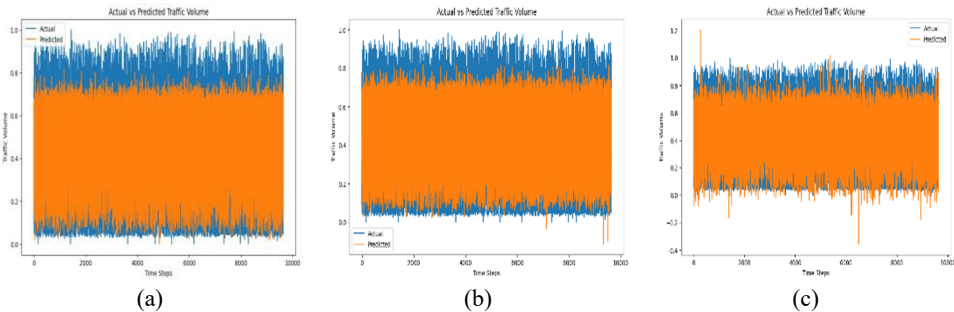


Fig. 5. Actual vs. Predicted Traffic Volume of (a) RNN, (b) GRU, and (c) CNN Models.

4.1 Performance Comparison

Table 2 presents the primary evaluation metrics for each model, highlighting distinct trade-offs between accuracy and computational demands. The GRU model achieved the best overall accuracy, with a MAPE of 2.105%, RMSE of 0.198, and the highest R^2 of 0.469,

indicating its strong capacity to capture variations in traffic volume. However, GRU also required the longest computation time at 7917 seconds, underscoring its computational intensity. In contrast, the CNN model, which completed predictions in just 853 seconds, offered moderate accuracy (MAPE of 2.492%, RMSE of 0.214, R^2 of 0.384). It is well-suited for real-time forecasting applications where speed is critical. The RNN model, with a MAPE of 2.654% and RMSE of 0.215, displayed intermediate accuracy but struggled with handling long-term dependencies, likely due to the absence of gating mechanisms in GRU.

Table 2. Performance Metrics and Computation Time for RNN, GRU, and CNN Models.

Model	MAPE (%)	RMSE	R^2	Computation Time (s)
RNN	2.654	0.215	0.374	3226
GRU	2.105	0.198	0.469	7917
CNN	2.492	0.214	0.384	853

The performance differences between models highlight the specific strengths of each architecture. GRU's gating mechanism allows it to retain information over long sequences, improving accuracy in complex time-series tasks like traffic forecasting. While lacking memory capabilities, CNN leverages convolutional filters to effectively capture local patterns, resulting in fast computation times with moderate accuracy. The RNN, as a simpler recurrent model, cannot effectively manage long-term dependencies, impacting its performance relative to GRU.

4.2 Performance Comparison

The training and validation loss curves in Figure 4 provide insights into each model's convergence behavior over 100 epochs. The GRU model (Figure 4. (b)) exhibits a steady reduction in training and validation loss, with a stable trend after approximately 40 epochs, indicating effective learning and generalization. The fluctuations in validation loss are minor, reflecting the model's ability to avoid overfitting while capturing complex dependencies. This stable convergence aligns with GRU's high accuracy metrics, suggesting that it effectively balances learning capacity with generalization.

The CNN model (Figure 4. (c)) displays a sharp initial drop in training and validation loss, followed by gradual convergence. By around 30 epochs, both losses reach a stable plateau, with the validation loss remaining close to the training loss. This behavior reflects CNN's capacity to capture short-term dependencies in the data efficiently, achieving moderate accuracy without significant overfitting. The CNN's rapid convergence supports its role as a computationally efficient model suitable for real-time applications.

In contrast, the RNN model (Figure 4. (a)) exhibits more variability in the validation loss and a relatively slower convergence rate. The gap between training and validation loss remains wider than in the GRU and CNN models, indicating that RNNs struggled with generalization. These patterns suggest that RNN faced challenges in retaining information over extended sequences, reducing its effectiveness in accurately predicting traffic volume.

4.3 Forecasting Accuracy Analysis

Figure 5 presents actual versus predicted traffic volumes for each model over time steps, visually assessing each model's forecasting accuracy. The GRU model (Figure 5. (b)) closely follows the traffic volume trends, reflecting its strong capability to capture temporal dependencies and produce accurate forecasts. Minor deviations exist, but overall, the GRU

model provides a reliable approximation of the actual traffic patterns, consistent with its low MAPE and high R^2 scores.

The CNN model (Figure 5. (c)) also reasonably captures the general trend of actual traffic volume, though it occasionally exhibits deviations, especially during abrupt changes in traffic patterns. These deviations are expected, given CNN's limitation in handling long-term dependencies. Despite this, the CNN model demonstrates adequate predictive accuracy for real-time applications where computational speed is prioritized.

The RNN model (Figure 5. (a)) exhibits more pronounced deviations from the actual traffic values, particularly during rapid fluctuations. This inconsistency reflects the RNN model's limitations in managing extended dependencies and complex patterns, resulting in less accurate predictions. As in GRU, the lack of gating mechanisms impairs RNN's ability to retain information over long sequences, further justifying its relatively higher error metrics.

4.4 Forecasting Accuracy Analysis

The results highlight essential trade-offs between model accuracy, efficiency, and complexity:

- **GRU:** The GRU model, with the highest accuracy, is best suited for applications demanding precise forecasting, such as historical traffic analysis and planning. However, the high computation time may limit its practicality for real-time traffic systems where predictions are needed within seconds. GRU's strengths lie in its ability to capture long-term dependencies, making it a powerful tool for complex forecasting tasks where quality takes precedence over speed.
- **CNN:** The CNN model achieves a balance between speed and accuracy, making it well-suited for real-time traffic management systems where latency is a critical concern. Although CNN does not capture long-term dependencies as effectively as GRU, its ability to extract local patterns quickly enables it to provide reliable predictions within short timeframes. This characteristic makes CNN advantageous in scenarios where real-time data is needed for immediate decision-making, such as dynamic traffic control.
- **RNN:** The RNN model, while exhibiting moderate computational demands, struggles with accuracy due to its inability to manage long-term dependencies effectively. It may be useful in simpler forecasting tasks where intermediate accuracy is acceptable, but its limitations suggest it may not be optimal for high-stakes forecasting applications.

These trade-offs illustrate that model selection should align with the specific requirements of the application. GRU's superior accuracy benefits strategic analysis tasks, while CNN's speed and efficiency make it ideal for real-time applications. RNN, though less accurate, may serve as a baseline model in contexts with constrained computational resources.

4.5 Trade-offs and Practical Implications

The results highlight essential trade-offs between model accuracy, efficiency, and complexity:

- **GRU:** The GRU model, with the highest accuracy, is best suited for applications demanding precise forecasting, such as historical traffic analysis and planning. However, the high computation time may limit its practicality for real-time traffic systems where predictions are needed within seconds. GRU's strengths lie in its ability to capture long-term dependencies, making it a powerful tool for complex forecasting tasks where quality takes precedence over speed.
- **CNN:** The CNN model balances speed and accuracy, making it well-suited for real-time traffic management systems where latency is critical. Although CNN does not capture long-term dependencies as effectively as GRU, its ability to extract local patterns quickly

enables it to provide reliable predictions within short timeframes. This characteristic makes CNN advantageous in scenarios where real-time data is needed for immediate decision-making, such as dynamic traffic control.

- **RNN:** The RNN model exhibits moderate computational demands but struggles with accuracy due to its inability to manage long-term dependencies effectively. It may be useful in simpler forecasting tasks where intermediate accuracy is acceptable, but its limitations suggest it may not be optimal for high-stakes forecasting applications.

These trade-offs illustrate that model selection should align with the application's specific requirements. GRU's superior accuracy benefits strategic analysis tasks, while CNN's speed and efficiency make it ideal for real-time applications. RNN, though less accurate, may serve as a baseline model in contexts with constrained computational resources.

4.6 Limitation

Several limitations should be considered when interpreting these results. The dataset is specific to westbound traffic on Interstate 94, which may limit the generalizability of the findings to other regions or types of traffic data. Additionally, while the models were trained using a fixed set of hyperparameters, further tuning could improve accuracy and efficiency, particularly for CNN and RNN. The study also focuses on hourly traffic volume predictions; applying these models to finer or coarser temporal resolutions might yield different results. Furthermore, while computation time was measured, other resource metrics like memory usage and energy consumption were not evaluated, which could influence model feasibility in specific deployment environments.

5 Conclusion

This study compared RNN, GRU, and CNN models for traffic volume prediction on Interstate 94, assessing their accuracy, ability to capture temporal dependencies, and computational efficiency. GRU achieved the highest accuracy with effective handling of long-term dependencies but required significant computation time, making it less ideal for real-time applications. In contrast, CNN demonstrated the fastest computation time, balancing moderate accuracy with high efficiency, suitable for real-time traffic monitoring. RNN, with its intermediate accuracy and computation time, struggled with complex temporal dependencies, suggesting limited applicability for high-stakes forecasting tasks.

These findings offer valuable insights into the trade-offs between accuracy and efficiency, providing a practical basis for selecting predictive models based on specific application needs in traffic management. Future research can build on this foundation by exploring hybrid architectures, such as combining CNN and GRU layers, to balance accuracy with speed effectively. Additionally, hyperparameter tuning and testing on diverse datasets could further validate these models' applicability across various traffic scenarios, enhancing their potential for real-time urban forecasting solutions.

References

1. B. Medina-Salgado, E. Sánchez-DelaCruz, P. Pozos-Parra, and J. E. Sierra, Urban traffic flow prediction techniques: A review, *Sustain. Comput. Informatics Syst.* **35**, 100739 (2022). <https://doi.org/10.1016/j.suscom.2022.100739>
2. K. Nellore and G. Hancke, A Survey on Urban Traffic Management System Using Wireless Sensor Networks, *Sensors* **16**, 157 (2016). <https://doi.org/10.3390/s16020157>

3. A. Boukerche and J. Wang, Machine Learning-based traffic prediction models for Intelligent Transportation Systems, *Comput. Networks* **181**, 107530 (2020). <https://doi.org/10.1016/j.comnet.2020.107530>
4. A. Khan, M. M. Fouda, D.-T. Do, A. Almaleh, and A. U. Rahman, Short-Term Traffic Prediction Using Deep Learning Long Short-Term Memory: Taxonomy, Applications, Challenges, and Future Trends, *IEEE Access* **11**, 94371 (2023). <https://doi.org/10.1109/ACCESS.2023.3309601>
5. Y. Mao, A. Pranolo, A. P. Wibawa, A. B. Putra Utama, F. A. Dwiyanto, and S. Saifullah, Selection of Precise Long Short Term Memory (LSTM) Hyperparameters based on Particle Swarm Optimization, in *2022 Int. Conf. Appl. Artif. Intell. Comput.* (IEEE, 2022), pp. 1114–1121. <https://doi.org/10.1109/ICAIC53929.2022.9792708>
6. I. D. Mienye, T. G. Swart, and G. Obaido, Recurrent Neural Networks: A Comprehensive Review of Architectures, Variants, and Applications, *Information* **15**, 517 (2024). <https://doi.org/10.3390/info15090517>
7. E. Elgohary, M. Galal, M. Aref, and M. Gharib, Long Short-Term Memory and Gated Recurrent Unit for Automated Deep Learning Prediction, *IJCI. Int. J. Comput. Inf.* **0** (2024). <https://doi.org/10.21608/ijci.2024.235027.1119>
8. K. Cao, T. Zhang, and J. Huang, Advanced hybrid LSTM-transformer architecture for real-time multi-task prediction in engineering systems, *Sci. Rep.* **14**, 4890 (2024). <https://doi.org/10.1038/s41598-024-55483-x>
9. D. Salman, C. Direkoglu, M. Kusaf, and M. Fahrioglu, Hybrid deep learning models for time series forecasting of solar power, *Neural Comput. Appl.* **36**, 9095 (2024). <https://doi.org/10.1007/s00521-024-09558-5>
10. Y. He, P. Huang, W. Hong, Q. Luo, L. Li, and K.-L. Tsui, In-Depth Insights into the Application of Recurrent Neural Networks (RNNs) in Traffic Prediction: A Comprehensive Review, *Algorithms* **17**, 398 (2024). <https://doi.org/10.3390/a17090398>
11. K. Ibrahim Mohammad Ata, M. Khair Hassan, A. Ghany Ismaeel, S. Abdul Rahman Al-Haddad, T. Alquthami, and S. Alani, A multi-Layer CNN-GRUSKIP model based on transformer for spatial –TEMPORAL traffic flow prediction, *Ain Shams Eng. J.* **103045** (2024). <https://doi.org/10.1016/j.asej.2024.103045>
12. H. Hewamalage, C. Bergmeir, and K. Bandara, Recurrent Neural Networks for Time Series Forecasting: Current status and future directions, *Int. J. Forecast.* **37**, 388 (2021). <https://doi.org/10.1016/j.ijforecast.2020.06.008>
13. H. Abbasimehr and R. Paki, Improving time series forecasting using LSTM and attention models, *J. Ambient Intell. Humaniz. Comput.* **13**, 673 (2022). <https://doi.org/10.1007/s12652-020-02761-x>
14. N. Ranjan, S. Bhandari, H. P. Zhao, H. Kim, and P. Khan, City-Wide Traffic Congestion Prediction Based on CNN, LSTM and Transpose CNN, *IEEE Access* **8**, 81606 (2020). <https://doi.org/10.1109/ACCESS.2020.2991462>
15. M. Dwarampudi and N. V. S. Reddy, Effects of padding on LSTMs and CNNs, (2019)
16. D. Munandar, B. N. Ruchjana, A. S. Abdullah, and H. F. Pardede, Literature Review on Integrating Generalized Space-Time Autoregressive Integrated Moving Average (GSTARIMA) and Deep Neural Networks in Machine Learning for Climate Forecasting, *Mathematics* **11**, 2975 (2023). <https://doi.org/10.3390/math11132975>
17. F. Kaytez, A hybrid approach based on autoregressive integrated moving average and least-square support vector machine for long-term forecasting of net electricity

- consumption, *Energy* **197**, 117200 (2020).
<https://doi.org/10.1016/j.energy.2020.117200>
18. J. Wang, Vehicular Traffic Flow Prediction Model Using Machine Learning-Based Model, These (2021). <https://doi.org/10.20381/ruor-26510>
 19. A. Boukerche, Y. Tao, and P. Sun, Artificial intelligence-based vehicular traffic flow prediction methods for supporting intelligent transportation systems, *Comput. Networks* **182**, 107484 (2020). <https://doi.org/10.1016/j.comnet.2020.107484>
 20. B. Lim and S. Zohren, Time-series forecasting with deep learning: a survey, *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.* **379**, 20200209 (2021). <https://doi.org/10.1098/rsta.2020.0209>
 21. A. Sherstinsky, Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network, *Phys. D Nonlinear Phenom.* **404**, 132306 (2020). <https://doi.org/10.1016/j.physd.2019.132306>
 22. C. M. Nawej, P. A. Owolawi, and T. Walingo, Toward a Realistic Comparative Analysis of Recurrent Neural Network's Methods via Long-Term Memory Approaches, Yang, X.S., Sherratt, R.S., Dey, N., Joshi, A. *Proc. Ninth Int. Congr. Inf. Commun. Technol. ICICT 2024* 2024. *Lect. Notes Networks Syst.* **1054**, 245 (2025). https://doi.org/10.1007/978-981-97-5035-1_19
 23. A. H. Ribeiro, K. Tiels, L. A. Aguirre, and T. Schön, Beyond exploding and vanishing gradients: analysing RNN training using attractors and smoothness, in *Proc. Twenty Third Int. Conf. Artif. Intell. Stat.*, edited by S. Chiappa and R. Calandra (PMLR, 2020), pp. 2370–2380
 24. P. B. Weerakody, K. W. Wong, G. Wang, and W. Ela, A review of irregular time series data handling with gated recurrent neural networks, *Neurocomputing* **441**, 161 (2021). <https://doi.org/https://doi.org/10.1016/j.neucom.2021.02.046>
 25. G. Van Houdt, C. Mosquera, and G. Nápoles, A review on the long short-term memory model, *Artif. Intell. Rev.* **53**, 5929 (2020). <https://doi.org/10.1007/s10462-020-09838-1>
 26. S. Zhang, L. Yao, A. Sun, and Y. Tay, Deep Learning Based Recommender System, *ACM Comput. Surv.* **52**, 1 (2020). <https://doi.org/10.1145/3285029>
 27. H. H. Goh, B. He, H. Liu, D. Zhang, W. Dai, T. A. Kurniawan, and K. C. Goh, Multi-Convolution Feature Extraction and Recurrent Neural Network Dependent Model for Short-Term Load Forecasting, *IEEE Access* **9**, 118528 (2021). <https://doi.org/10.1109/ACCESS.2021.3107954>
 28. L. Chen, H. Yan, J. Yan, J. Wang, T. Tao, K. Xin, S. Li, Z. Pu, and J. Qiu, Short-term water demand forecast based on automatic feature extraction by one-dimensional convolution, *J. Hydrol.* **606**, 127440 (2022). <https://doi.org/10.1016/j.jhydrol.2022.127440>
 29. H. Jayadianti, B. A. Arianti, N. H. Cahyana, S. Saifullah, and R. Drezewski, Improving sentiment analysis on PeduliLindungi comments: a comparative study with CNN-Word2Vec and integrated negation handling, *Sci. Inf. Technol. Lett.* **4**, 75 (2023). <https://doi.org/10.31763/sitech.v4i2.1184>
 30. R. Munif and A. Prahara, Betta fish classification using transfer learning and fine-tuning of CNN models, *Sci. Inf. Technol. Lett.* **5**, 16 (2024). <https://doi.org/10.31763/sitech.v5i1.1378>
 31. A. Shewalkar, D. Nyavanandi, and S. A. Ludwig, Performance Evaluation of Deep Neural Networks Applied to Speech Recognition: RNN, LSTM and GRU, *J. Artif. Intell. Soft Comput. Res.* **9**, 235 (2019). <https://doi.org/10.2478/jaiscr-2019-0006>

32. I. Ebtelahaj and H. Bonakdari, CNN vs. LSTM: A Comparative Study of Hourly Precipitation Intensity Prediction as a Key Factor in Flood Forecasting Frameworks, *Atmosphere (Basel)*. **15**, 1082 (2024). <https://doi.org/10.3390/atmos15091082>
33. F. M. Shiri, T. Perumal, N. Mustapha, and R. Mohamed, A Comprehensive Overview and Comparative Analysis on Deep Learning Models: CNN, RNN, LSTM, GRU, (2023)
34. J. Hogue, *Metro Interstate Traffic Volume*, (2019)
35. M. Shantal, Z. Othman, and A. A. Bakar, A Novel Approach for Data Feature Weighting Using Correlation Coefficients and Min–Max Normalization, *Symmetry (Basel)*. **15**, 2185 (2023). <https://doi.org/10.3390/sym15122185>
36. L. B. V. de Amorim, G. D. C. Cavalcanti, and R. M. O. Cruz, The choice of scaling technique matters for classification performance, *Appl. Soft Comput.* **133**, 109924 (2023). <https://doi.org/10.1016/j.asoc.2022.109924>
37. C. Xu, P. Zhao, Y. Liu, J. Xu, V. S. S. S.Sheng, Z. Cui, X. Zhou, and H. Xiong, Recurrent Convolutional Neural Network for Sequential Recommendation, in *World Wide Web Conf.* (ACM, New York, NY, USA, 2019), pp. 3398–3404. <https://doi.org/10.1145/3308558.3313408>
38. C.-L. Liu, W.-H. Hsaio, and Y.-C. Tu, Time Series Classification With Multivariate Convolutional Neural Network, *IEEE Trans. Ind. Electron.* **66**, 4788 (2019). <https://doi.org/10.1109/TIE.2018.2864702>
39. K. Fauvel, T. Lin, V. Masson, É. Fromont, and A. Termier, XCM: An Explainable Convolutional Neural Network for Multivariate Time Series Classification, *Mathematics* **9**, 3137 (2021). <https://doi.org/10.3390/math9233137>
40. D. Chicco, M. J. Warrens, and G. Jurman, The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation, *PeerJ Comput. Sci.* **7**, e623 (2021). <https://doi.org/10.7717/peerj-cs.623>