Comparing LSTM, GRU, and Transformers in Traffic Forecasting Across Varying Context Sizes and Horizons

Carlos E. S. Oliveira[1], **Eduardo J. S. Luz**[1], **Leonardo B. L. Santos**[2], **Didier Augusto Vega Oliveros**[3], **Vander L. S. Freitas**[1]

[1]Postgraduate Program in Computer Science – Federal University of Ouro Preto (UFOP)
Ouro Preto – MG – Brazil

[2]Centro Nacional de Monitoramento e Alertas de Desastres Naturais (CEMADEN)
São José dos Campos – SP – Brazil

[3]Institute of Science and Technology – Federal University of São Paulo (UNIFESP)
São José dos Campos – SP – Brazil

vander.freitas@ufop.edu.br

Abstract. *Forecasting traffic delays is a challenging task due to complex temporal dependencies and the sensitivity of models to input configurations. This paper presents a comparative analysis of the Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and Transformer architectures using time-series data from the PeMS system. We evaluated the sensitivity of the model to different context sizes and forecast horizons. Hyperparameters are optimized using Optuna, and performance is assessed with RMSE and MAE. The results show that GRU achieve the best performance in most scenarios, followed by LSTM and Transformer.*

1. Introduction

Rapid urban growth has intensified road traffic, leading to chronic congestion, pollution, and strain on transport infrastructure [Data 2024, Brookings 2024, of Concerned Scientists 2024, of Public Health 2024], a challenge observed globally, from the US to Brazil and Pakistan [Instituto Brasileiro de Geografia e Estatística 2023, Chen et al. 2022, ul Hissan et al. 2023]. Neural network models have stood out among emerging technologies for traffic congestion forecasting, particularly using datasets like the Performance Measurement System (PeMS) [of Transportation 2017, Sawah et al. 2024, Chen and Chen 2020] – a large-scale system maintained by the California Department of Transportation that collects real-time highway data on traffic flow, speed, and delays, and is among the most utilized datasets in traffic forecasting research.

Several studies have employed LSTM and GRU architectures for traffic forecasting using PeMS data, highlighting their capacity to model temporal dependencies in road networks [Sawah et al. 2024, Hou et al. 2023]; however, most still rely on fixed input windows and forecast horizons [Hou et al. 2023, Chen and Chen 2020], limiting model adaptability to different temporal scenarios, and the impact of these configurations on performance remains underexplored.

This study explores how LSTM, GRU, and Transformer architectures perform under varying input window sizes (context sizes) and forecast horizons for traffic delay

prediction using the PeMS dataset. We hypothesize that model performance depends critically on temporal configurations, with certain architectures exhibiting greater robustness in specific scenarios. To investigate this, we adopted a sliding window approach and systematically varied both window sizes and forecast horizons across a broad range of conditions. The GRU model consistently achieved the best results, particularly with longer input windows, reinforcing its suitability for flexible and robust forecasting systems in intelligent transportation scenarios.

2. Theoretical Background

2.1. Time Series

Time series are sequences of observations recorded at regular time intervals and are often modeled as stochastic processes [Box et al. 2008]. These sequences may contain components such as trend, seasonality, and noise:

$$Y_t = \alpha_t + \beta_t + \varepsilon_t, \tag{1}$$

where Y_t is the observed value at time t, α_t represents the trend, β_t the seasonality, and ε_t the noise [Morettin and Toloi 2018]. For forecasting purposes, we define the model input as a sliding window of past observations. For a lookback window of length T, the input at time t is given by:

$$\mathbf{X}_t = [\mathbf{x}_{t-T+1}, \ldots, \mathbf{x}_t] \in \mathbb{R}^{T \times d}, \tag{2}$$

where d is the number of features per observation. In this work, we consider a univariate time series of traffic delay, thus $d = 1$.

2.2. Evaluation Metrics

To assess prediction performance, we adopt Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE), both widely used in time series forecasting due to their simplicity and interpretability [Hyndman and Athanasopoulos 2018]:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}, \qquad \text{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{y}_i\right|. \tag{3}$$

RMSE penalizes larger errors more heavily, making it sensitive to outliers, while MAE provides a straightforward measure of average absolute deviations. These metrics are especially appropriate for univariate regression tasks where interpretability and comparability between models are desired.

2.3. Statistical Significance Test

The two-step procedure proposed by Demšar [Demšar 2006] begins with the Friedman test, a non-parametric statistical test that does not require the assumption of normally distributed data. In our context, this test evaluates the null hypothesis that the error rates produced by different models are statistically indistinguishable.

If the Friedman test rejects the null hypothesis at a 5% significance level ($\alpha = 0.05$), the procedure proceeds with the Nemenyi post-hoc test. This test identifies statistically significant differences between model pairs by ranking the models and computing

their critical distance (CD). Specifically, if the performance ranks of two models lie within their mutual critical distance, the Nemenyi test fails to reject the null hypothesis, indicating no significant difference in their results.

2.4. Neural Architectures

LSTM (Long Short-Term Memory) networks are recurrent architectures that store information over time using a memory cell and gating mechanisms [Hochreiter and Schmidhuber 1997]. Given an input sequence $\mathbf{X}_t \in \mathbb{R}^{T\times d}$, each observation $\mathbf{x}_t$ is processed sequentially, influencing the state of the internal cell and the hidden state in each step of time. The LSTM cell integrates new inputs with its internal memory via input, forget, and output gates.

The cell state is updated by:

$$c_t = f_t \cdot c_{t-1} + i_t \cdot \tilde{c}_t, \tag{4}$$

where f_t is the forget gate, i_t the input gate, $\tilde{c}_t$ the candidate memory, and c_{t-1} the cell state from the previous time step.

GRU (Gated Recurrent Unit) simplifies LSTM by combining gates, reducing the number of parameters [Cho et al. 2014]. The hidden state is updated as:

$$h_t = (1 - z_t) \cdot h_{t-1} + z_t \cdot \tilde{h}_t, \tag{5}$$

where z_t is the update gate, $\tilde{h}_t$ is the candidate activation, and h_{t-1} is the hidden state from the previous time step.

Transformer models replace recurrence with self-attention mechanisms, improving parallelism and the ability to model long-range dependencies [Vaswani et al. 2017]. In our formulation, each input sequence $\mathbf{X}_t \in \mathbb{R}^{T\times d}$, corresponding to a sliding window of T past observations, is treated as a sequence of T tokens. To encode temporal order, we used the standard sinusoidal positional encoding proposed by Vaswani et al. [Vaswani et al. 2017], which is added to the input embeddings before attention. The attention mechanism is defined as follows:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V, \tag{6}$$

where Q, K, and V represent the query, key, and value matrices, respectively, derived from the input through learned linear projections. The scaled dot-product computes the similarity between tokens, enabling the model to weigh different positions dynamically.

2.5. Related Work

Recent research highlights the predominance of neural architectures for short- and medium-term traffic forecasting tasks, especially LSTM, GRU, and more recently Transformers. These models frequently leverage the PeMS dataset due to its comprehensive traffic flow, speed, and occupancy information. However, a recurring limitation in many of these studies is the adoption of static input window sizes and forecast horizons, which

restricts the models' generalizability across varying temporal scenarios. Table 1 summarizes selected studies that employ LSTM, GRU, or Transformer-based models for traffic forecasting using the PeMS dataset. The table was compiled through exploratory searches in the CAPES Periodicals Portal[1] using combinations of keywords such as "traffic forecasting", "PeMS", "LSTM", "Transformer", and "GRU".

Table 1. Selected studies using LSTM, GRU, or Transformer-based models on the PeMS dataset.

Year	Article	Models	Metrics	Dataset	Study Area	Input Window	Forecast Horizon
2020	[Ranjan et al. 2020]	PredNet (LSTM-based)	RECALL	TOPIS	Seoul	10, 30, 60 min	-
2020	[Ranjan et al. 2020]	LSTM	-	-	-	-	-
2021	[Ramchandra and Raj 2021]	DAN/DBN/RF/LSTM	MSE/RMSE	-	-	-	-
2021	[Joseph et al. 2021]	BLSTME	-	-	-	-	-
2022	[Gao 2022]	Prophet/LSTM	MAE/MSE/RMSE	-	Shanghai	-	-
2022	[Das et al. 2022]	LSTM/BLSTM/SLSTM	-	PeMS	San Francisco	10 min	-
2022	[Feng et al. 2022]	SARIMA/LSTM/Prophet	RMSE/MAE/MAPE	RTA	Jilin	2015-12/2019	-
2022	[Lakhani et al. 2023]	LSTM	RMSE	PeMS	California	-	-
2022	[Huang et al. 2022]	MHODE (GRU component)	MAE/RMSE/MAPE	PeMS	California	60 min	15/30/45/60 min
2022	[Hua and Fan 2022]	GRU/SaEs	MAE/MAPE/MSE	PeMS	California	5 min	5 min
2022	[Fang et al. 2022]	Lastjormer (uses Gated units like GRU)	MAE/RMSE/MAPE	PeMS	California	5 min	60 min
2023	[Pan et al. 2023]	FD-Markov-LSTM	MAE/RMSE/MAPE	PeMS	Los Angeles	-	-
2023	[Zhang et al. 2023]	LSTM	RMSE	-	-	-	-
2023	[Sunny 2023]	LSTM/Adaboost	-	-	-	-	-
2023	[Sharma and Ranjan 2023]	LSTM/RF	RMSE	-	-	-	-
2023	[Tiwari and Prasad 2023]	LSTM/ARIMA/SARIMA	RMSE	-	-	-	-
2023	[Samonte et al. 2022]	LSTM/R-CNN/Fast R-CNN	-	PeMS	California	-	-
2023	[Narmadha and Vijayakumar 2023]	Hybrid CNN-LSTM Model	MAE/MRE	PeMS	California	5 min	-
2023	[Sawah et al. 2024]	LSTM/GRU	MAE/RMSE	PeMS	California	-	-
2023	[Xu et al. 2024]	LSSTF-GWN (attention-based)	MAE/MAPE/RMSE	PeMS	California	60 min	60 min
2023	[Xie and Mallick 2023]	GraphWaveNet/D2STGNN (with GRU)	MAE/RMSE/MAPE	PeMS	California	15 min	-
2024	[Chen et al. 2024]	IDG-PSAtt (attention mechanism)	MAE/RMSE/MAPE	PeMS	California	15, 30, 60 min	15, 30, 60 min
2024	[Yang et al. 2024]	STSSTA (attention-based)	MAE/RMSE	PeMS	California	-	-
2024	[Bhartiya et al. 2024]	CNN/RNN/LSTM/RBM	-	-	-	-	-
2024	[Rui and Dong 2024]	CEEMDAN/CNN/LSTM	MAE	-	-	-	-
2024	[Ma et al. 2024]	H-CNN-LSTM	MAE/RMSE	East Road	Shanghai	-	-
2024	[Jia et al. 2016]	LSTM	RMSE	-	-	-	-
2024	[Xu 2024]	LSTM/GRU/CNN-LSTM/MLR	MSE/MAE	PeMS	California	12-24 hrs	1 hr
2024	[Elmazi et al. 2024]	GRU/SVR	MAE/MSE/RMSE	-	-	-	-
2024	[Ounoughi and Ben Sadok Yahia 2024]	Grizzly (GRU inside)	MAE/RMSE	PeMS	California	5 min	-
2024	[Hou et al. 2024]	FEGC-GRU-ILPSO	MAE/RMSE	PeMS	California	-	-

Several works illustrate common methodological practices in the field. [Huang et al. 2022] introduce MHODE, a hybrid GRU-based architecture that achieves strong performance for 15–60 minute horizons but relies on a fixed 60-minute input window; [Xu et al. 2024] propose LSSTF-GWN, a GRU-based attention model for long-range dependencies under a similarly static configuration; and [Hua and Fan 2022] evaluate a GRU with self-attention using a fixed 5-minute window, without exploring broader temporal variations. Other studies follow similar constraints: [Lakhani et al. 2023], [Zhang et al. 2023], and [Jia et al. 2016] adopt LSTM models with fixed or unspecified window/horizon settings; hybrid models such as CNN-LSTM [Narmadha and Vijayakumar 2023], H-CNN-LSTM [Ma et al. 2024], and CEEMDAN-CNN-LSTM [Rui and Dong 2024] typically evaluate only narrow temporal configurations; and Transformer-inspired approaches like GraphWaveNet [Xie and Mallick 2023] and STSSTA [Yang et al. 2024] introduce architectural novelty without examining temporal parameters explicitly.

Table 1 shows that many papers use the PeMS dataset, while less than 25% report experiments with multiple-window or horizon configurations. Approximately half of the studies omit at least one of these parameters, using fixed values or failing to specify them. This pattern indicates that, although widely used, sliding window strategies are often applied without systematic variation, limiting insights into model sensitivity to temporal configurations. In contrast, our work systematically varies both window sizes and forecast horizons across a dense evaluation grid. By jointly analyzing these dimensions, we uncover interactions often overlooked in the literature and assess model robustness to temporal changes.

[1] https://www.periodicos.capes.gov.br/

3. Experimental Design

This study aims to investigate how variations in input window size and forecast horizon affect the predictive performance of neural models in traffic delay forecasting. We hypothesize that different temporal configurations reveal distinct performance behaviors across model architectures. Source code and data are available at: `https://github.com/kaduoliveira83/eniac_2026`.

3.1. Dataset and Preprocessing

The dataset originates from the Performance Measurement System (PeMS), managed by Caltrans[2] (the California Department of Transportation), which aggregates real-time traffic information from sensors deployed along California highways.

We used traffic data from 2022 and 2023, merging monthly CSV files and retaining three main attributes: timestamp, station ID, and $Delay_60$ (average delay relative to a 60 mph reference speed). A single station (ID 314121, "5NB at Elk Grove HOV") was selected based on data completeness and reliability. Among the candidate stations in the corridor, this station presented the lowest proportion of missing readings and the most consistent sensor operation throughout the study period, which motivated its selection for this study. Records with missing or zero delay values were removed. Zero-valued readings were treated as sensor artifacts rather than valid observations, as constant zero values in the $Delay_60$ variable were found to coincide with known sensor malfunction periods rather than genuine free-flow conditions. All values were normalized using z-score scaling, with mean and standard deviation computed exclusively from the training split to prevent data leakage.

3.2. Sliding Window Strategy and Data Split

The time series was converted into supervised format using a sliding window approach, in which fixed-length historical sequences are associated with forecast intervals to form input-output pairs. Both window and horizon sizes are expressed in minutes, where each unit corresponds to 5 minutes of real-world time. For example, a 12×12 configuration refers to a 60-minute input window predicting 60 minutes ahead. To ensure temporal integrity, the dataset was chronologically split into 70% for training, 15% for validation, and 15% for testing.

3.3. Model Implementation and Training

We implemented LSTM, GRU, and Transformer models in PyTorch[3], using the Adam optimizer with MSE loss. Early stopping based on validation loss was employed to prevent overfitting, with GPU acceleration used when available. Following hyperparameter tuning, each model was retrained on the combined training and validation sets and evaluated on the test set using RMSE and MAE. All models converged smoothly under early stopping, with GRU showing the fastest convergence and Transformer requiring more computational resources.

[2] `https://dot.ca.gov/`
[3] `https://pytorch.org/`

Table 2. Search space and selected hyperparameters for LSTM, GRU, and Transformer models optimized independently using Optuna.

Hyperparameter	Search Space (Optuna)	LSTM / GRU	Transformer
Hidden units / d_model	{16, 32, 64, 128, 256, 512}	256	32
Dropout rate	[0.1, 0.4]	0.3	0.1
Learning rate	[1e-5, 1e-3]	0.001	0.004
Number of layers	{1, 2, 3}	2	2
Number of heads (Transformer only)	{2, 4, 8}	-	8
Feedforward dimension (Transformer only)	{64, 128, 256}	-	256

3.4. Hyperparameter Optimization and Evaluation Strategy

We optimized hyperparameters for each model architecture using **Optuna framework**. Our tuning focused on three representative window-horizon pairs: 12×12 (short-term), 24×24 (medium-term), and 48×48 (long-term), aligning with common operational timeframes in traffic forecasting, such as outlined in works listed on Table 1. The search space included model-specific parameters like hidden units, dropout rates, learning rates, number of layers, and, for the Transformer, attention heads and feedforward dimensions (see Table 2).

The optimal configurations derived from these trials were then generalized to all other window-horizon combinations using a distance-based centroid selection strategy. Specifically, each of the three tuned configurations (12×12, 24×24, and 48×48) defines a centroid in the two-dimensional window–horizon space. For any other configuration (w, h) in the 48×48 grid, the Euclidean distance to each of the three centroids is computed, and the hyperparameter set associated with the nearest centroid is adopted for that configuration. This nearest-centroid assignment kept the tuning process computationally tractable while still providing a configuration-aware approximation across the full evaluation grid. A full 48×48 grid evaluation for each architecture required approximately 48 hours on a workstation equipped with an Intel i9-10900 CPU, 128 GB RAM, and an NVIDIA RTX 3090 GPU. It is also worth noting that the Optuna search converged to a smaller representational capacity for the Transformer ($d_model = 32$) than for the recurrent models (hidden size $= 256$, see Table 2). This asymmetry reflects the independent search space explored for each architecture rather than an intentional design choice, and should be taken into account when interpreting the comparative results, as it may partially disadvantage the Transformer relative to LSTM and GRU.

4. Results and Discussion

This section presents and discusses the results obtained from varying input window sizes and forecast horizons across LSTM, GRU, and Transformer models.

Figure 1 presents the heatmaps of the Root Mean Squared Error (RMSE) and the Mean Absolute Error (MAE) for the LSTM, GRU, and Transformer models. They illustrate how predictive performance varies according to combinations of input window size (horizontal axis) and forecast horizon (vertical axis). The blue regions represent lower error values (better performance), while the red regions correspond to higher errors. It is important to mention that each unit in the window size or horizon corresponds to 5 minutes, thus a horizon of 3 means 15 minutes.

For the LSTM and GRU models, the best performance (dark blue) is consistently observed in the lower-right corner of the heatmaps, corresponding to long input windows

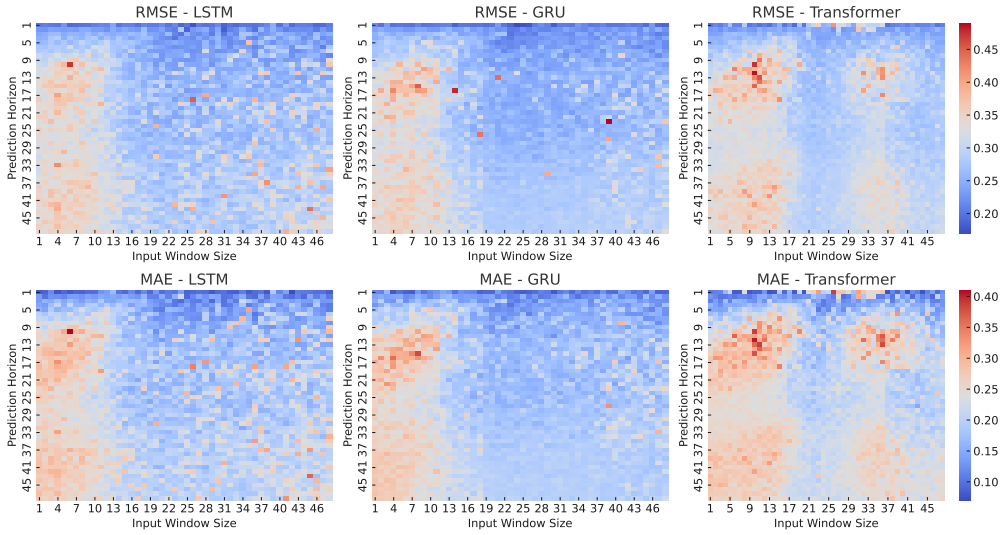


Figure 1. RMSE and MAE heatmaps for LSTM, GRU, and Transformer across input window sizes and prediction horizons (errors in minutes).

($\geq$ 36) combined with short prediction horizons ($\leq$ 6). In particular, GRU displays the most stable low-error zone, with RMSE values below 0.25 across a wide range of configurations in this region. As the prediction horizon increases—especially beyond 30, the error intensifies significantly when smaller windows ($\leq$ 12) are used, as seen in the red zones in the upper-left area. This suggests that both recurrent models rely heavily on sufficient historical context to maintain accuracy for longer-term predictions.

The Transformer model exhibits a more heterogeneous pattern. While it performs reasonably well for some short-horizon forecasts with moderate windows (e.g. 12–24), its heatmaps reveal greater dispersion and sensitivity to temporal changes. Unlike the recurrent models, the Transformer does not present a clearly defined "optimal zone." Its RMSE and MAE values fluctuate more abruptly, particularly as the forecast horizon increases. The upper portion of the Transformer heatmaps contains large orange and red regions, even when paired with longer windows, suggesting diminished capacity for handling long-range forecasts in this configuration.

Overall, the heatmaps reveal that the GRU model is not only the most stable but also the most consistently accurate across varying temporal configurations. LSTM follows closely with similar trends. In contrast, the Transformer's performance landscape is more irregular, indicating a need for further architectural tuning to handle temporal variability effectively.

A consistent pattern emerges across all models: prediction accuracy declines as the forecast horizon increases, especially when using small input windows. This behavior is clearly observed in the upper-left regions of the heatmaps. These results indicate that limited historical context is insufficient for making accurate long-term forecasts in the traffic domain. Conversely, combinations of long input windows and short horizons (visible in the lower-right corners) yield consistently better results, particularly for LSTM and GRU models. This suggests that recurrent architectures benefit from richer temporal context, aligning with their internal memory mechanisms tailored to capture sequential dependencies. These findings are consistent with prior studies, such as [Huang et al. 2022]

and [Xu et al. 2024], which also reported GRU's effectiveness in traffic forecasting with fixed configurations.

The Transformer model, while competitive in certain regions, presents a more heterogeneous error landscape. It demonstrates robustness in a few configurations with shorter windows but tends to degrade more significantly as horizon length increases. This behavior may stem from the lack of an inherent sequential bias in the attention mechanism, requiring additional architectural tuning, such as improved positional encodings or hybrid recurrence-attention structures, to better handle varying temporal dynamics.

Table 3 summarizes the errors averaged across all window-horizon combinations.

Table 3. Summary of RMSE and MAE statistics for all windows and horizons.

Model	RMSE (mean ± std)	MAE (mean ± std)
LSTM	0.290 ± 0.041	0.204 ± 0.046
GRU	0.283 ± 0.040	0.198 ± 0.042
Transformer	0.307 ± 0.035	0.231 ± 0.044

The GRU outperformed the other models with the lowest average RMSE and MAE, followed closely by LSTM. These findings confirm our initial hypothesis: temporal configuration (window size and horizon) plays a crucial role in model performance, and recurrent models handle these variations more effectively than Transformer-based architectures.

Statistical Significance Analysis

We evaluated model performance across three input window size groups: short (1-16), medium (17-33), and long (34-50). Figure 2 presents the Critical Difference plots, where models connected by a line are not statistically different according to the Nemenyi test ($p \geq 0.05$).

The Friedman test indicated statistical significance ($p < 0.05$) in all cases. The Nemenyi test revealed that GRU and LSTM performed comparably, both significantly outperforming Transformer. GRU achieved the best mean rank (1.8489). In these cases, the Nemenyi test demonstrated that GRU, LSTM, and Transformer were all statistically distinct from each other, with GRU consistently achieving the lowest mean ranks (1.4363 for medium, 1.5653 for long), indicating superior performance. These two outcomes correspond to different window groups: for short windows (1–16), GRU and LSTM were statistically comparable and both outperformed the Transformer; for medium (17–33) and long (34–50) windows, all three models were statistically distinct, with GRU consistently achieving the best mean rank.

Practical Implications and Scalability

GRU and LSTM are recommended for real-time traffic systems (e.g., adaptive signal control) due to their superior performance in long-window/short-horizon configurations, while Transformers require enhanced tuning for deployment. This recommendation reflects predictive accuracy alone; a full real-time feasibility assessment would also require measuring inference latency and computational cost, which is beyond this study's

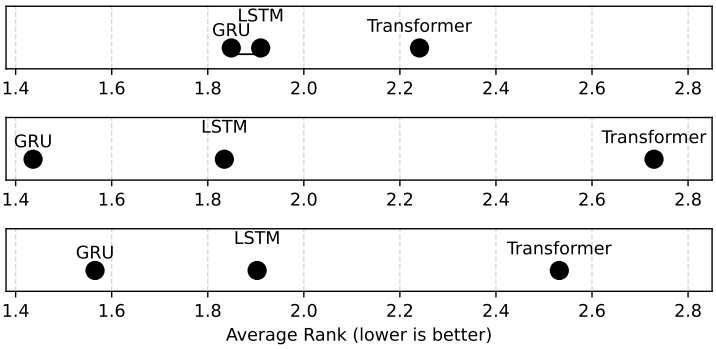


Figure 2. Critical Difference plots for input window groups 1–16, 17–33, and 34–50 (top to bottom). Connected models are not statistically different (Nemenyi test, $p \geq 0.05$).

scope. Our centroid-based hyperparameter tuning strategy enabled computationally efficient evaluation of $2,304$ configurations, offering scalability for large design spaces. Future assessments should incorporate robustness metrics, anomaly detection capabilities, and scenario-specific performance indicators to address operational constraints.

5. Conclusions and Future Work

This study evaluated LSTM, GRU, and Transformer architectures for traffic delay forecasting using univariate PeMS time series. All models were sensitive to temporal configurations (input window size and prediction horizon). GRU achieved the best overall performance, with lower errors and faster convergence. LSTM performed well with larger input windows but was more sensitive to longer horizons. The Transformer exhibited higher error variability and struggled with long-term forecasting, despite robustness to input length. Heatmaps confirmed that GRU and LSTM benefit from extended historical context, while the Transformer maintained stable but less accurate results. All models showed increased error with longer prediction horizons, highlighting the challenge of long-range forecasting in dynamic systems.

Limitations include single-station data, exclusion of multivariate/exogenous factors (speed, flow, weather), and absence of classical time series benchmarks , such as naive persistence or ARIMA/SARIMA baselines, which would help contextualize the practical magnitude of the reported error differences between architectures. Future work should extend the analysis to multiple locations, incorporate multivariate inputs, and explore hybrid/memory-augmented models (e.g., Informers, CNN-attention combinations) to improve long-range forecasting. Uncertainty estimation techniques (Bayesian networks, quantile regression) should also be integrated for reliable real-time applications.

Acknowledgments

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001, and Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq (Grant 446053/2023-6).

References

Bhartiya, P., Bhatele, M., and Waoo, A. A. (2024). A machine learning approach for predictive analysis of traffic flow. *ShodhKosh Journal of Visual and Performing Arts*, 5(5).

Box, G., Jenkins, G., and Reinsel, G. (2008). *Time Series Analysis: Forecasting and Control*. Wiley Series in Probability and Statistics. Wiley.

Brookings (2024). Traffic: Why it's getting worse, what government can do. Accessed on October 7, 2024.

Chen, F. et al. (2022). Quatformer: Quaternion transformer for multi-modal learning. *arXiv preprint arXiv:2205.05982*.

Chen, L., Chen, L., Wang, H., and Zhang, H. (2024). Traffic flow prediction based on interactive dynamic spatio-temporal graph convolution with a probabilistic sparse attention mechanism. *Transportation Research Record*, 2678(9):837–853.

Chen, Y. and Chen, X. (2020). A large scale pems traffic speed dataset.

Cho, K., Van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734.

Das, A., Barisal, S. K., and Dutta, P. (2022). A comparative analysis on traffic flow prediction. In *2022 International Conference on Machine Learning, Computer Systems and Security (MLCSS)*, pages 318–324.

Data, C. (2024). Us number of registered vehicles. Accessed on October 7, 2024.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *JMLR*, 7:1–30.

Elmazi, K., Elmazi, D., Musta, E., Mehmeti, F., and Hidri, F. (2024). An intelligent transportation systems-based machine learning-enhanced traffic prediction model using time series analysis and regression techniques*. In *2024 International Conference on INnovations in Intelligent SysTems and Applications (INISTA)*, pages 1–6.

Fang, Y., Zhao, F., Qin, Y., Luo, H., and Wang, C. (2022). Learning all dynamics: Traffic forecasting via locality-aware spatio-temporal joint transformer. *IEEE Transactions on Intelligent Transportation Systems*, 23(12):23433–23446.

Feng, T., Zheng, Z.-Y., Xu, J., Liu, M., Li, M., Jia, H., and Yu, X. (2022). The comparative analysis of sarima, facebook prophet, and lstm for road traffic injury prediction in northeast china. *Frontiers in Public Health*, 10.

Gao, J. (2022). Time-series prediction research based on combined prophet-lstm models. In *2022 18th International Conference on Computational Intelligence and Security (CIS)*, pages 143–147.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Hou, Y., Zhang, D., Li, D., and Deng, Z. (2024). Regional traffic flow combination forecasting model considering the virtual space of road network. *Physica A: Statistical Mechanics and its Applications*, 637:129598. Available online 10 February 2024.

Hou, Y., Zhang, D., Li, D., and Yang, P. (2023). Forecasting method considering macro and micro characteristics of chaotic traffic flow. *Chinese Physics B*, 32(10):100508.

Hua, C. and Fan, W. D. (2022). Freeway traffic speed prediction under the intelligent driving environment: A deep learning approach. *Journal of Advanced Transportation*, 2022:1–9.

Huang, X., Lan, Y., Ye, Y., Wang, J., and Jiang, Y. (2022). Traffic flow prediction based on multi-mode spatial-temporal convolution of mixed hop diffuse ode. *Electronics*, 11(19).

Hyndman, R. J. and Athanasopoulos, G. (2018). *Forecasting: principles and practice*. OTexts.

Instituto Brasileiro de Geografia e Estatística (2023). Frota nacional de veículos.

Jia, Y., Wu, J., and Du, Y. (2016). Traffic speed prediction using deep learning method. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1217–1222.

Joseph, L. M. I. L., Goel, P., Jain, A., Rajyalakshmi, K., Gulati, K., and Singh, P. (2021). A novel hybrid deep learning algorithm for smart city traffic congestion predictions. In *2021*

6th International Conference on Signal Processing, Computing and Control (ISPCC), pages 561–565.

Lakhani, R., Dalal, D., Dantwala, V., and Bhowmick, K. (2023). Traffic congestion prediction using deep learning models. In Joshi, A., Mahmud, M., and Ragel, R. G., editors, *Information and Communication Technology for Competitive Strategies (ICTCS 2022)*, pages 779–788, Singapore. Springer Nature Singapore.

Ma, L., Wang, Y., Lv, X., He, Y., and Wang, G. (2024). Hybrid deep learning model for traffic flow and speed forecasting at urban junction. *Advances in Information Management and Economic Development Research*, 1:48.

Morettin, P. and Toloi, C. (2018). *Análise de séries temporais: modelos lineares univariados*. BLUCHER.

Narmadha, S. and Vijayakumar, V. (2023). Spatio-temporal prediction of vehicle traffic flow using a multivariate cnn and lstm model. *Materials Today: Proceedings*, 81:826–833.

of Concerned Scientists, U. (2024). Inequitable exposure to air pollution from vehicles in california. Accessed on October 7, 2024.

of Public Health, H. T. C. S. (2024). Decreased vehicle emissions linked to fewer deaths. Accessed on October 7, 2024.

of Transportation, C. D. (2017). *Transit Features in the California Department of Transportation Performance Measurement Systems (PeMS) Transit PeMS User Guide*. California Department of Transportation.

Ounoughi, C. and Ben Sadok Yahia, S. (2024). A gru-based grizzly search optimization approach for traffic flow prediction. *Expert Systems with Applications*, 236:121325.

Pan, Y., Guo, J., Chen, Y., Cheng, Q., Li, W., and Liu, Y. (2023). A fundamental diagram based hybrid framework for traffic flow estimation and prediction by combining a markovian model with deep learning. *Expert Systems With Applications*.

Ramchandra, N. and Raj, R. (2021). Performance evaluation of machine learning algorithms in traffic flow prediction. In -.

Ranjan, N., Bhandari, S., Zhao, H. P., Kim, H., and Khan, P. (2020). City-wide traffic congestion prediction based on cnn, lstm and transpose cnn. *IEEE Access*, 8:81606–81620.

Rui, X. and Dong, S. (2024). Short-term traffic flow prediction based on CEEMDAN-CNN-LSTM. In Mikusova, M., editor, *International Conference on Smart Transportation and City Engineering (STCE 2023)*, volume 13018, page 130182C. International Society for Optics and Photonics, SPIE.

Samonte, M., Balan, G., Gavino, P., Monasterial, J., Reforsado, R., and Samonte, D. (2022). Deep learning in traffic flow control and prediction for traffic management. In *Proceedings of the 12th Annual International Conference on Industrial Engineering and Operations Management*.

Sawah, M., Taie, S., Ibrahim, M., and Hussein, S. (2024). Deep neural network-based approach for accurate vehicle counting. *International Journal of Intelligent Engineering and Systems*, 16.

Sharma, A. and Ranjan, P. (2023). Traffic prediction model using machine learning in intelligent transportation systems. In *2023 International Conference on Circuit Power and Computing Technologies (ICCPCT)*, pages 1165–1173.

Sunny, M. (2023). Vehicular traffic flow prediction model using deep learning. *International Journal For Science Technology And Engineering*, 11(7):275–279.

Tiwari, N. and Prasad, L. (2023). Experimental analysis of time series models for traffic flow prediction. *SAMRIDDHI : A Journal of Physical Sciences, Engineering and Technology*, 15(03):312–320.

ul Hissan, R., Waseem, L. A., Khurshid, M., and Maqbool, M. S. (2023). The impact of urbanization on the flow of traffic in faisalabad city pakistan. *Pakistan Journal of Humanities and Social Sciences*, 11.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Xie, Y. and Mallick, T. (2023). A comparative study of loss functions: Traffic predictions in regular and congestion scenarios. *arXiv.org*, abs/2308.15464.

Xu, W., Liu, J., Yan, J., Yang, J., Liu, H., and Zhou, T. (2024). Dynamic spatiotemporal graph wavelet network for traffic flow prediction. *IEEE Internet of Things Journal*, 11(5):8020–8032.

Xu, Y. (2024). Machine Learning Based Traffic Prediction and Congestion Control Algorithms in Software Defined Networks . In *2024 International Conference on Interactive Intelligent Systems and Techniques (IIST)*, pages 285–289, Los Alamitos, CA, USA. IEEE Computer Society.

Yang, D., Li, H., Wang, P., and Yuan, L. (2024). Multi-step traffic speed prediction: A sequence-to-sequence spatio-temporal attention model. *Physica A: Statistical Mechanics and its Applications*, 638:129636. Available online 27 February 2024.

Zhang, K., Chu, Z., Xing, J., Zhang, H., and Cheng, Q. (2023). Urban traffic flow congestion prediction based on a data-driven model. *Mathematics*.