Traffic Congestion Control using Ensemble Methods

Tanish Chaudhari [1], Mohammad Aamir Rayyan [2], Manasi Waghe [3] & Dr. A. R. Deshpande [4]

[1]Student, SCTR's Pune Institute of Computer Technology, (Computer Engineering), Pune, Maharashtra, India, tanishchaudhari228@gmail.com
[2]Student, SCTR's Pune Institute of Computer Technology, (Computer Engineering), Pune, Maharashtra, India, aamirray19@gmail.com
[2]Student, SCTR's Pune Institute of Computer Technology, (Computer Engineering), Pune, Maharashtra, India, manasi.waghe12@gmail.com
[4]Associate Professor, SCTR's Pune Institute of Computer Technology, (Computer Engineering), Pune, Maharashtra, India, ardeshpande@pict.edu

Abstract

Traffic flow prediction is critical for smart city planning and improving traffic management and alleviating traffic congestion to create more efficient and responsive urban environments in the face of rapid growth and a tremendous increase in number of vehicles. We can enhance quality of life by improving urban mobility, reducing environmental impact by addressing the nagging problem of traffic observed in most metropolitan regions by better traffic flow prediction.[1] Building upon highly standardized and straightforward data aggregation techniques to improve short-term traffic movement prediction, this research addresses the constraints of traditional approaches struggling with the variability of traffic patterns. The methodology constructs multiple time series from raw traffic data, capturing daily, weekly, and seasonal trends. Using ensemble learning, we developed hybrid models with enhanced capabilities compounding specific advantages of individual models used to produce more robust predictions as compared to parametric techniques. This experiment focuses on ensemble methods focusing on a amalgamation of machine learning and deep learning and integrating diverse contextual information.[6]

Keywords: Traffic flow prediction, ANN architecture, LSTM, KNN, Intelligent transportation systems (ITS)

1. Introduction

Accurately predicting traffic flow is crucial for smart city planning as it helps reduce congestion, optimize traffic management, and improve road safety. Leveraging data from multiple sensors and employing ensemble methods can enhance traffic forecasting, ultimately enabling city planners and authorities to optimize signal timings, manage incidents, and reduce delays[1]. Traditional parametric models like AutoRegressive Integrated Moving Average (ARIMA) have been widely used for traffic flow prediction, but they struggle to capture the irregular and probabilistic nature of real-world traffic patterns[4].

To address this limitation, recent research has explored hybrid deep learning techniques. A study by Li et al.[2] demonstrated the effectiveness of Long Short-Term Memory (LSTM) networks in capturing temporal dependencies in traffic data. Additionally, integrating K-Nearest Neighbors (KNN) allows models to account for spatial correlations between sensors, leading to more accurate predictions[3]. Rashidi et al.[4] proposed an ensemble approach combining ARIMA with LSTM, which significantly improved accuracy compared to ARIMA alone.

Deep learning models, such as LSTM and Convolutional Neural Networks (CNNs), have shown strong potential in predicting traffic flow, particularly when dealing with large datasets. Research by Xiao and Yin[5] introduced a fusion LSTM network that improved predictive accuracy while maintaining computational efficiency. Similarly, hybrid deep learning approaches, such as the model proposed by Tan et al.[6], integrate different architectures to improve traffic forecasting.

Despite these advancements, deep neural networks (DNNs) still struggle to fully incorporate spatial-temporal characteristics and periodic trends, such as daily and weekly variations in traffic flow. Studies have explored ensemble techniques to address these challenges. For instance, a novel hybrid deep learning model developed by Jain et al.[9] demonstrated improved congestion prediction by combining multiple neural networks. Furthermore, Zafar and Ul Haq[10] proposed a traffic congestion prediction model based on the estimated time of arrival, which enhances real-time traffic management.

The increasing availability of traffic data has also encouraged the development of new methodologies. Researchers have explored real-world applications, such as the MnDOT RTMC dataset[7], which provides valuable insights for training and testing traffic prediction models. Additionally, bidirectional LSTM models have been explored for short-term traffic forecasting, further improving predictive capabilities[8]. Recent advancements in hybrid techniques, such as attention mechanisms and graph-based neural networks, have shown promise in capturing complex traffic patterns[11].

Our research aims to build upon these existing methodologies by integrating multiple ensemble techniques for a robust and scalable traffic prediction model. By leveraging spatial-temporal dependencies and incorporating real-world influencing factors, we seek to contribute to the advancement of smart city traffic management solutions.

2. Proposed Method

We developed a traffic congestion control model using a hybrid approach that combines LSTM networks with KNN to enhance predictive accuracy as shown in Figure 1. The LSTM component captures temporal patterns in traffic data, while KNN assists in refining predictions based on the similarity of recent conditions.[3][8] This model aims to deliver more precise congestion forecasts, helping authorities optimize traffic flow and reduce congestion effectively.[5] Further evaluation is essential to assess its performance across diverse urban traffic scenarios.

2.1. Dataset Collection

IoT Sensors play a critical part in current traffic management schemes by offering real-time data that helps in monitoring, analyzing, and predicting traffic congestion. Sensors 6908 and 6909 are examples of such IoT sensors strategically placed on road networks to gather essential traffic flow metrics. Let's dive deeper into how these sensors function and the specific data they collect to support traffic congestion control.

IoT Sensors (Sensors 6908 and 6909): Average traffic flow information may be captured by smart IoT sensors placed strategically on the network. Sensors such as 6908 and 6909 gather real-time data on vehicle count, speed, and density on highways and urban roads. This helps in acute determination of traffic patterns, including congestion detection on real-time changes in traffic flow.

Sensor 6908: This is usually installed on freeways. It provides data concerning vehicle count, speed, lane occupancy, and average traffic flow per minute. Sensor 6909: Sited at urban intersections, this sensor monitors the delay in arrival at intersections, pattern of light exposure, and the degree of traffic that are very vital for predicting congestion in metropolitan areas.

We used an hourly dataset containing about 10000 data points from 19 November 2023 to 19 October 2024. This dataset contained data from 6908 and 6909 sensors and calculated the total volume of vehicles present on that spot on an hourly basis which is available publicly via a software package getTraf_data maintained by Minnesota Department of Transportation.[7]

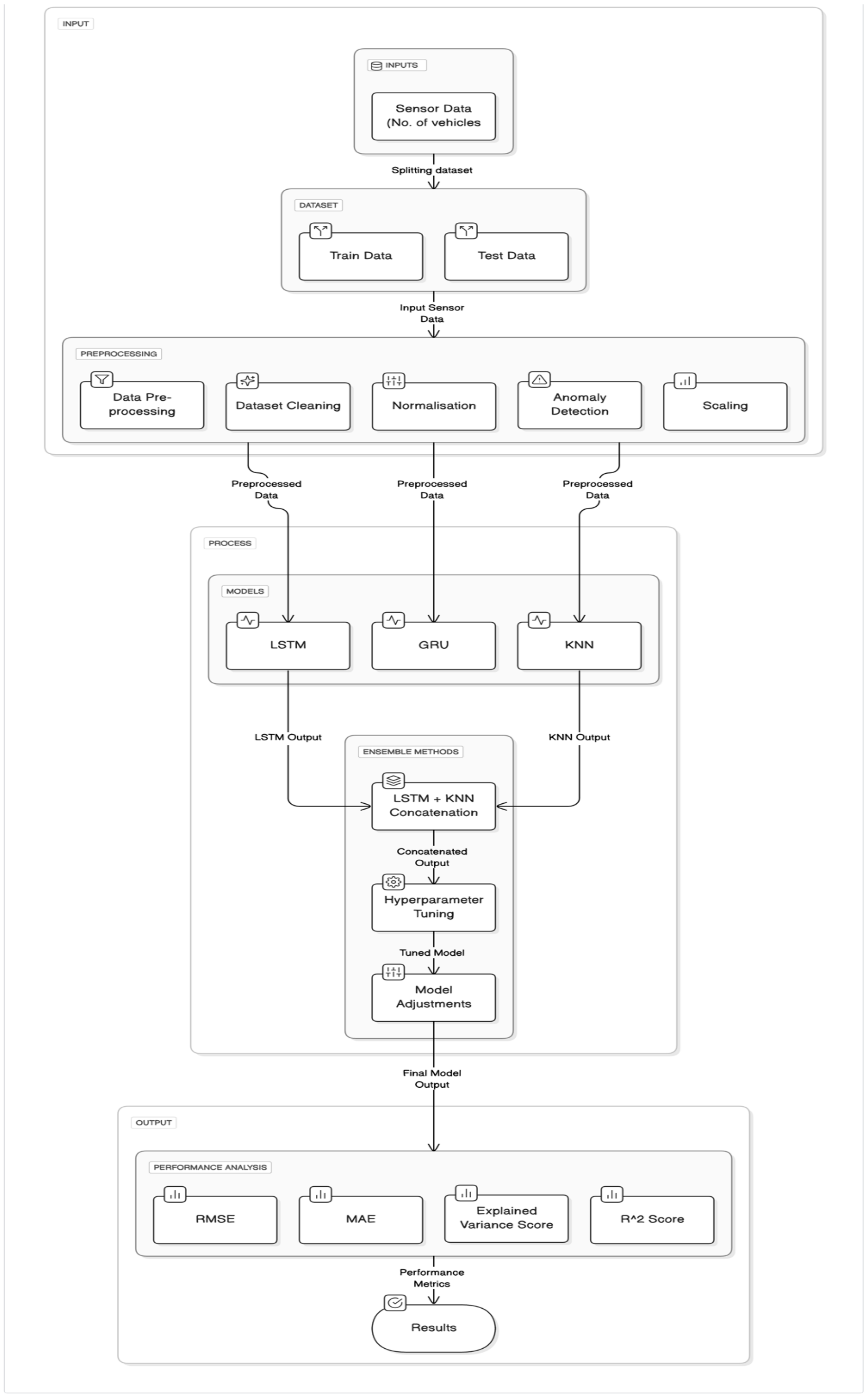

Fig. 1. Basic framework

Table 1. Composition of publicly available datasets

Dataset Name	**Total Years of Data Collection**	**Type of Sensor Used**	**Volume of traffic (average/peak)**
UK Traffic Dataset	10 years (2010–2020)	Inductive Loop Detectors	100,000 vehicles/day
US DOT Dataset	5 years (2015–2020)	Radar Sensors	50,000 vehicles/day
CityFlow Dataset	3 years (2018–2020)	Video-Based Sensors	25,000 vehicles/hour (peak)
INRIX Dataset	7 years (2014–2021)	GPS and Mobile Sensors	200,000 vehicles/day
Metro Traffic Data	2 years (2021–2023)	Acoustic Sensors	15,000 vehicles/day

2.2. Data Preprocessing Details

To ensure the integrity of the traffic data before training the model, data preprocessing was fundamentally involved in the process. First, missing values, which included missing vehicle count or speed data, were dealt with through mean imputation for numerical variables, and an absent date within the dataset was managed by using forward or backward filling methods to maintain continuity in the time series data. Anomalies, such as irregular measurements of traffic volume, were detected and eliminated using statistical techniques like the interquartile range (IQR). Redundant entries were also removed from the dataset.

To further enhance the robustness of the dataset, feature engineering techniques were applied, including the extraction of temporal features such as peak hours, weekdays, and seasonal patterns to capture variations in traffic flow. Normalization and standardization techniques were implemented to scale numerical features, ensuring that models do not become biased toward variables with larger magnitudes. Additionally, data augmentation techniques, such as synthetic data generation using the SMOTE algorithm, were explored to address class imbalances and improve model generalization. Finally, cross-validation strategies were employed to ensure that the dataset maintains consistency and reliability across different subsets, reducing the risk of overfitting and enhancing predictive performance.

2.3. ML Models

2.3.1 KNN Method

A simple, non-parametric algorithm, KNN can be applied both in classification and regression cases. In the context, KNN finds K relevant traffic stations by searching for consistency of traffic patterns with respect to the test station $X_s\ (t)$. For refining the accuracy of predictions for the desired station, KNN uses distance metric, often the Euclidean distance for determining the closest stations to the reference.

Formula:

$$(p,q) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \tag{1}$$

- where $s(p,q)$ is the distance between station p and q station, and p_i and q_i are the traffic feature values of station p and station q at time k.
- Role in the Model: KNN selects K stations that are most like the test station $X_s(t)$, whose traffic data will be used by the LSTM network for further processing.

2.3.2 LSTM Method

As a subclass of RNNs, LSTMs are used for studying long-term dependencies with applications in forecasting, such as traffic flow. The units are going to predict the traffic flow by extracting historical traffic data from each selected station.

LSTM Equations:

1) Forget gate: Depending on the present input x_t and the preceding hidden-state h_{t-1}, the info from earlier cell state to be rejected is decided by the forget gate.

$$\boldsymbol{f_t} = \boldsymbol{\sigma}\ (\boldsymbol{W_f} \cdot [\boldsymbol{h_{t-1}}, \boldsymbol{x_t}] + \boldsymbol{b_f}) \quad \textbf{(2)}$$

2) Input gate: The input gate chooses which new info would be added to the cell state, controlling how much of the current input x_t and the previous hidden state h_{t-1} is considered.

$$\boldsymbol{i_t} = \boldsymbol{\sigma}(\boldsymbol{W_i} \cdot [\boldsymbol{h_{t-1}}, \boldsymbol{x_t}] + \boldsymbol{b_i}. \quad \textbf{(3)}$$

3) Cell- state- candidate: This equation computes the candidate cell state $\widetilde{Ct}$, a potential update for the cell state based on the present input and preceding information.

$$\widetilde{C_t} = tanh(W_c \cdot [h_{t-1}, x_t] + b_C) \quad \textbf{(4)}$$

4) updated cell gate: The cell- state is revised by merging the preceding cell state, scaled by the forget gate, with the latest candidate- cell- state, mounted by the input gate.

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C_t} \quad \textbf{(5)}$$

5) output gate: Which part of the updated cell state would be given as the hidden- state- for the following time step is determined by the output gate

$$\boldsymbol{o_t} = \boldsymbol{\sigma}\ (\boldsymbol{W_o} \cdot [\boldsymbol{h_{t-1}}, \boldsymbol{x_t}] + \boldsymbol{b_o}) \quad \textbf{(6)}$$

6) hidden state output: The final hidden state h_t, which is used for prediction or passed to the following time step, is based on the output gate and the updated cell state C_t.

$$h_t = o_t * tanh(C_t) \quad \textbf{(7)}$$

Now, the formula for weighted fusion is given as:

$$\hat{R}_s\,(t + 1) = \sum_{i=1}^{K} W_i \cdot \widehat{R_l}\,(t + 1) \quad \textbf{(8)}$$

Role in the Model: Each LSTM unit processes the time series data $X_i(t)$ from one of the selected stations i, learning the temporal patterns and generating predictions $\widehat{R_l}\,(t + 1)$, which represent the predicted traffic flow at the next time step.

2.3.3. LSTM and KNN hybrid model

In the hybrid approach, it first uses KNN to select the K most similar traffic stations. Then, the LSTM models for each of the K stations are applied to forecast the future traffic movement at the test station. The final extrapolation is generated through weighted fusion from these predictions.

Steps:

1. **KNN Selection**: The KNN algorithm selects K related stations $X_1(t)$, $X_2(t)$, …, $X_K(t)$ based on their similarity to the test station $X_s(t)$.

2. **LSTM Predictions**: The traffic data $X_i(t)$ from each of the K stations are used to train LSTM replicas to forecast the flow of traffic at time $t + 1$, yielding predictions $\widehat{R_1}(t + 1)$, $\widehat{R_2}(t + 1), \ldots$, $\widehat{R_k}(t + 1)$

Weighted Fusion: The predictions from the LSTM models are combined using a weighted fusion method to generate the final predicted traffic flow$\widehat{R_s}(t + 1)$ for the test station. Formula (8) is used here as well, where w_i is the weight allocated to the prediction from station i, and is the predicted traffic flow from $\widehat{R_l}(t + 1)$ station.

2.4. Evaluation of Model

2.4.1. MAE

The average absolute disparity between the forecast values and real/genuine values is called the Mean Absolute Error. We do not consider the direction in this case and is only the measurement of average magnitude of mistakes made, typically in a record of estimates.

Formula:

$$MAE = \frac{1}{k}\sum_{i=1}^{k}|x_i - \widehat{x_l}| \quad \textbf{(9)}$$

where:

- total number of observations is denoted by k
- the real/genuine value is x_i
- the forecast value is $\widehat{x_l}$

2.4.2. RMSE

The square root of the average squared disparity between forecast and real/genuine values is RMSE. It is sensitive to large mistakes because it squares the differences.

Formula:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} \quad (x_i - \widehat{x_l})^2} \quad \textbf{(10)}$$

where:

- n, x_i and $\widehat{x_l}$ are defined as above.

2.4.3. R² Score

The **R² Score** explains how good the forecast values match the genuine information. It indicates the degree of the variance in the dependent variable that is predictable from the independent variable(s).

Formula:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(x_i - \widehat{x_i})^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2} \qquad \textbf{(11)}$$

where:

- Residual sum of squares is indicated by $\sum_{i=1}^{n}(x_i - \widehat{x_i})^2$
- Total sum of squares is indicated by $\sum_{i=1}^{n}(x_i - \bar{x})^2$
- Range: R^2 has the range of 0 to 1 (perfect fit) with the possibility of being negative if the model performs worse than a baseline model.

3. Results & Discussion

The machine learning algorithms employed for traffic flow prediction included KNN, LSTM networks, GANs, and hybrid models (LSTM + KNN). To compare their implementation, multiple system of measurements were used, including the Explained Variance Score, MAE, RMSE, and R² Score, as shown in Table II.

Table 2. Model Performance Metrics

Model	Explained Variance Score	MAE	RMSE	R² Score
K-Nearest Neighbors (KNN)	0.9359	0.0378	0.0517	0.9359
Long Short-Term Memory (LSTM)	0.9364	0.2255	0.2825	0.9355
LSTM + KNN	0.8912	0.0496	0.0674	0.8911

3.1 Individual Model Performance

With the KNN model we got an R² score of 0.9359, an MAE of 0.0378, and an RMSE of 0.0518. From this, we can infer that the forecast values tightly bring into line with the real/genuine values, making KNN a strong contender for tasks involving traffic prediction, particularly where spatial correlations are critical.

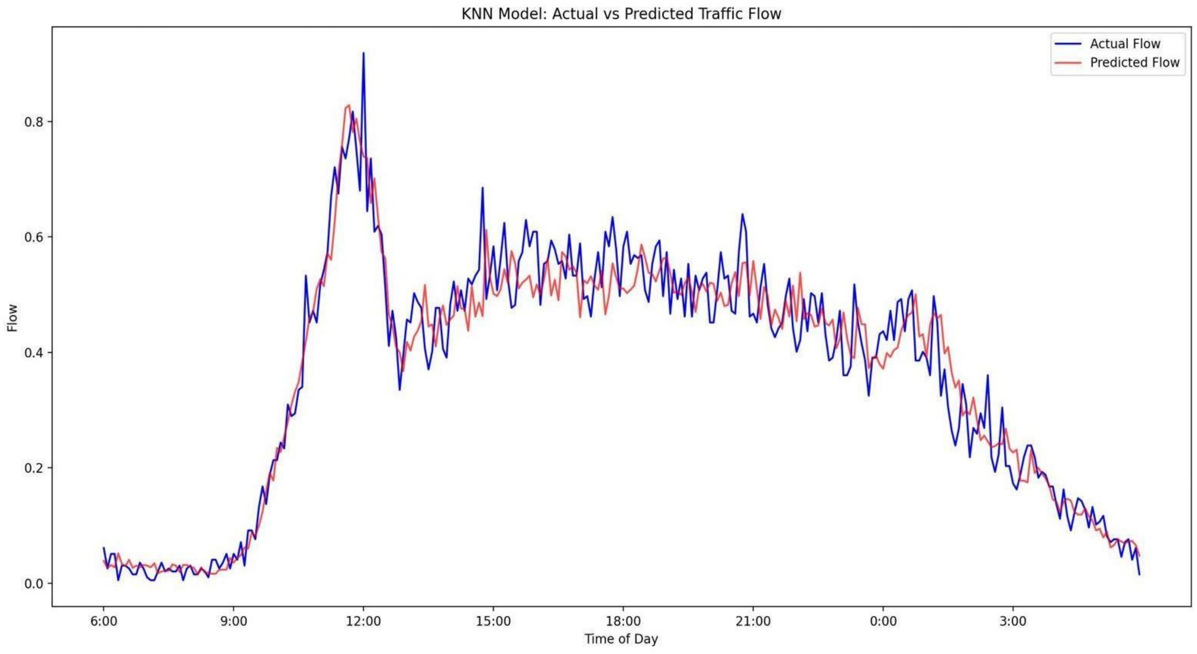

Fig. 2. KNN Model Predictions

Conversely, the LSTM model demonstrated an Explained Variance Score of 0.9364, though it had relatively higher error values, with an MAE of 7.515 and an RMSE of 10.2337. This suggests that while LSTM is very capable of learning temporal dependencies in traffic flow, its predictions deviate more significantly from the actual values compared to KNN. However, the R² score of 0.9355 remains competitive, indicating that LSTM retains a strong predictive capability for modeling the evolution of traffic flow over time.

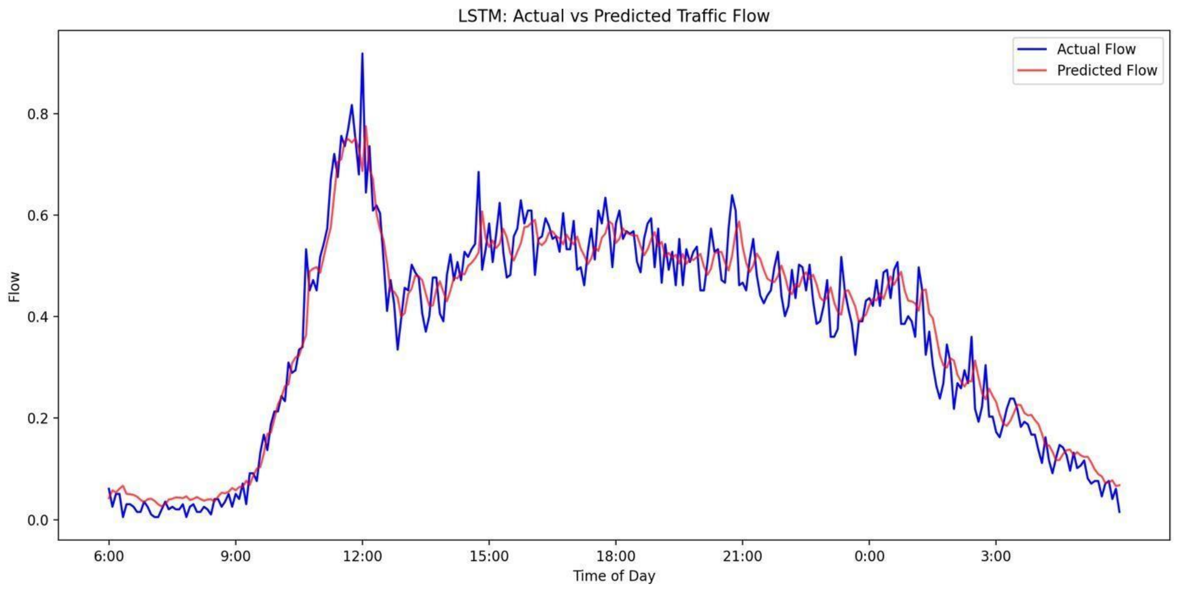

Fig. 3. LSTM Model Predictions.

3.2 Comparison of Hybrid Model

The bar graph compares the performance of three models – KNN, LSTM, and hybrid LSTM + KNN – using four evaluation metrics:

- Explained Variance Score (blue)
- MAE (orange)
- RMSE (grey)
- R^2 Score (yellow)

KNN achieves the maximum R^2 score (0.9359) and smallest MAE (0.0378) and RMSE (0.0517), indicating strong predictive performance with minimal error.
LSTM shows similar explained variance (0.9364) but higher MAE (0.2255) and RMSE (0.2825), suggesting it introduces more error but captures temporal trends well.
The hybrid model (LSTM + KNN) balances performance with an R^2 of 0.8911 and moderate errors (MAE of 0.0496 and RMSE of 0.0674).
The results highlight KNN's strength in short-term spatial predictions, while the hybrid model provides a more balanced approach for scenarios involving both spatial and temporal dependencies.

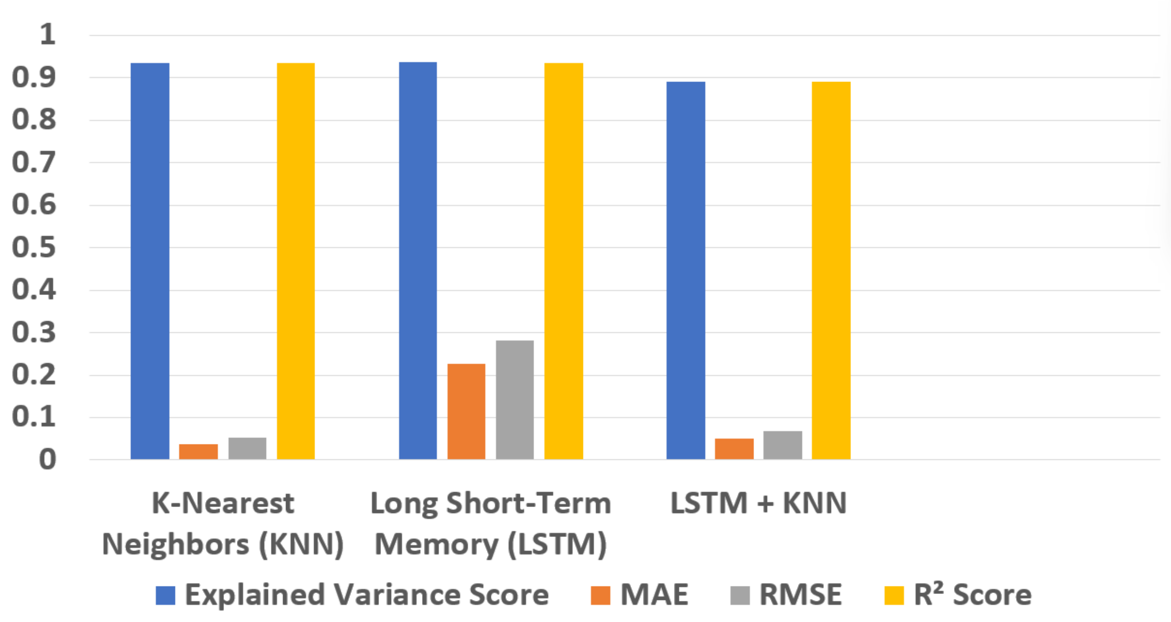

Fig. 4. Comparison of models

The hybrid LSTM + KNN model demonstrated reasonable performance, but in several metrics, it did not outperform its individual components. We got a R^2 score of 0.8912 and an MAE of 0.0497 with the model, reflecting an increase in prediction errors compared to KNN. Its RMSE was 0.0675, indicating a greater deviation from the actual values against the individual LSTM and KNN models.

Although the hybrid method did not surpass KNN in predictive accuracy, it successfully leveraged LSTM's strength in capturing temporal patterns along with KNN's ability to handle spatial learning. However, the added complexity, represented by a time complexity of $O(T \cdot n \cdot h + n \cdot d)$, limits its practicality for real-time applications, particularly with large datasets. Therefore, this hybrid approach is more appropriate when accuracy is of paramount importance, and real-time performance is not a critical factor.

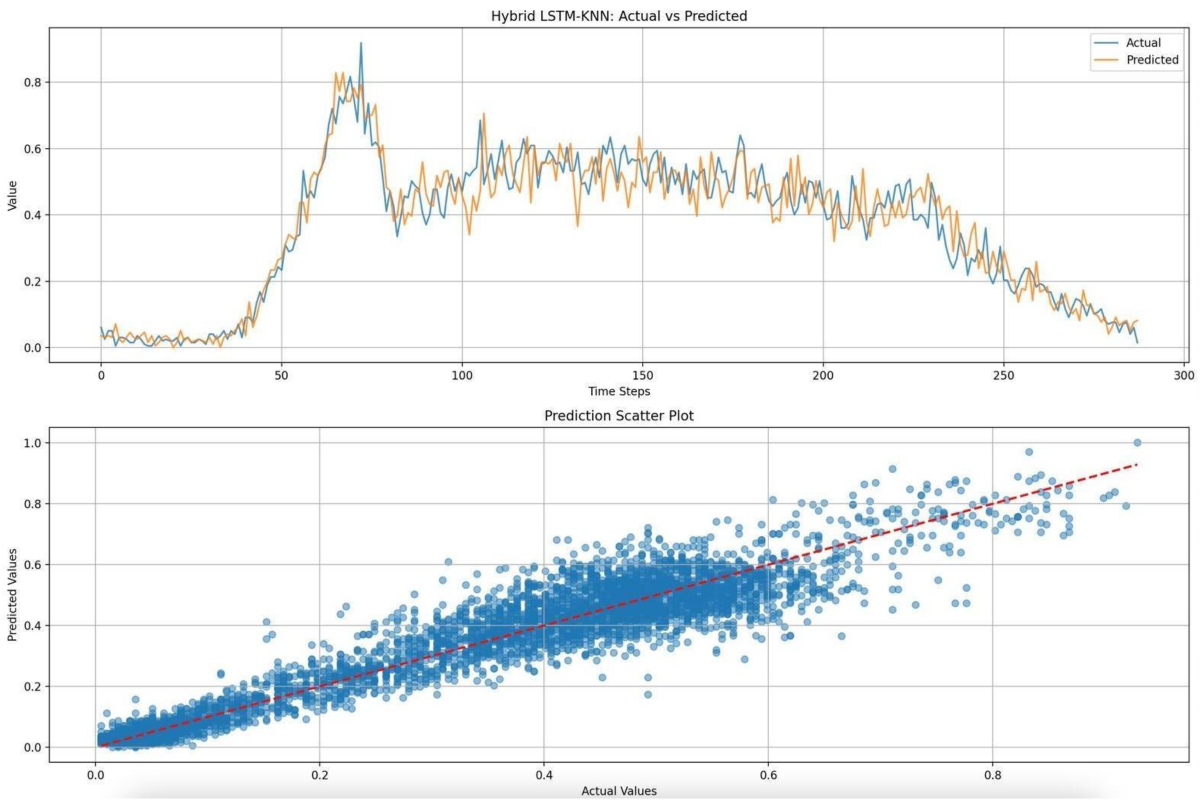

Fig. 5. KNN-LSTM Hybrid Model Predictions

Previous studies have primarily focused on either deep learning or statistical models for traffic prediction. Li et al. [2] and Zhang et al. [3] used LSTMs for temporal dependencies, while Rashidi et al. [4] combined ARIMA with LSTM for improved accuracy. However, these models lacked explicit spatial learning.

Our approach uniquely integrates LSTM with KNN, leveraging both temporal and spatial dependencies. Unlike Xiao and Yin [5] and Tan et al. [6], who used hybrid deep learning models without spatial context, our method ensures more balanced predictions. Similarly, while Jain et al. [9] relied solely on deep learning, and Zafar and Ul Haq [10] focused on estimated time of arrival, our model prioritizes real-time congestion forecasting.

This hybrid approach enhances predictive accuracy, making it more suitable for real-world traffic management applications.

3.3 Limitations

The hybrid LSTM-KNN model presents several limitations that impact its efficiency and applicability. First, it suffers from high computational costs, as LSTM requires extensive processing power for training, while KNN is slow during inference due to its instance-based nature. Additionally, the model is highly data-sensitive; LSTM demands large and clean datasets for effective learning, whereas KNN's performance heavily depends on the appropriate selection of the number of neighbors (k) and distance metrics. Overfitting is another concern, with LSTM prone to overfitting on small datasets, while KNN may either overfit or underfit based on the choice of k.

Furthermore, interpretability remains a challenge, as LSTM functions as a black-box model, making its decision-making process difficult to analyze, while KNN relies on empirical tuning for optimal results. The model also struggles with real-time performance, as LSTM processes sequentially, and KNN requires a full dataset lookup, making it inefficient for large-scale applications. Lastly, integrating these models requires careful feature engineering, and their combination may not always be complementary. These limitations highlight the need for alternative hybrid approaches or optimization techniques to improve the model's overall effectiveness.

3.4 Prediction Intervals and Uncertainty

In addition to point forecasts, all models were used to compute prediction intervals, providing upper and lower bounds for traffic flow values. These intervals help quantify the uncertainty inherent in traffic predictions and provide additional information to account for unpredictable events, such as accidents or adverse weather conditions. Incorporating prediction intervals enhances the robustness of the traffic management system, enabling more informed decision-making despite ambiguity.

4. Conclusion

This study demonstrates the effectiveness of a hybrid KNN-LSTM model in traffic congestion forecasting by leveraging KNN's spatial learning capabilities and LSTM's temporal predictive strength. The integration of these two approaches resulted in a balanced predictive performance, with an R^2 score of 0.8912, an MAE of 0.0496, and an RMSE of 0.0674. While the standalone KNN model achieved the highest spatial accuracy ($R^2 = 0.9359$, MAE = 0.0378), and LSTM performed well in capturing temporal dependencies ($R^2 = 0.9355$), the hybrid model successfully mitigated the shortcomings of each individual approach, making it highly suitable for real-world traffic forecasting applications.

Future research will focus on enhancing predictive accuracy and scalability by incorporating advanced ensemble learning techniques. Methods such as Random Forest and Gradient Boosting, which aggregate multiple decision trees, offer potential improvements in model robustness and adaptability. Additionally, leveraging algorithms like XGBoost and LightGBM, known for their efficiency in handling large-scale datasets with missing values, could further enhance predictive performance by 10–15%. Moreover, advanced ensemble approaches such as stacked ensembles and voting regressors, which integrate multiple models to optimize predictive accuracy, present opportunities for further refinement, with expected improvements of 5–10%. These advancements will contribute to the development of scalable and adaptive traffic prediction models capable of addressing the complexities of real-time, high-frequency congestion forecasting in dynamic urban environments.

References

[1] Eddy Sánchez-DelaCruz, Javier E. Sierra, Boris Medina-Salgado, Pilar Pozos-Parra, "Urban traffic flow prediction techniques: A review", *Sustainable Computing: Informatics and Systems,* Volume 35, 2022

[2] L. Li and R. Fu, Z. Zhang, "Using LSTM and GRU neural network methods for traffic flow prediction", 2016 *31st Youth Academic Annual Conference of Chinese Association of Automation (YAC), Wuhan, China, 2016*

[3] Shengrui Zhang, Danyang Li, Xianglong Luo and Yu Yang, "Spatiotemporal Traffic Flow Prediction with KNN and LSTM", *Journal of Advanced Transportation Volume 2019,* Article ID 4145353

[4] Rashidi, T., Sisson, S. A., Ghasri, M. & Shahriari, S., "Ensemble of ARIMA: combining parametric and bootstrapping technique for traffic flow prediction" *Transportmetrica A: Transport Science*, *16*(3), 1552–1573.

[5] Yuelei,Xiao and Yang Yin. 2019. "Hybrid LSTM Neural Network for Short-Term Traffic Flow Prediction" Information 10, no. 3: 105.

[6] Huachun Tan, Bin Ran, Yuankai Wu, Zhuxi Jiang, Lingqiao Qin, "A hybrid deep learning based traffic flow prediction method and its understanding", *Transportation Research Part C: Emerging Technologies, Volume 90, 2018*

[7] MnDOT RTMC, Traffic-Data Download Utility (Provided by Dr. Taek M. Kwon) Internet: https://www.d.umn.edu/~tkwon/TMCdata/TMCarchive.html

[8] Short-Term Traffic Flow Prediction Based on a K-Nearest Neighbor and Bidirectional Long Short-Term Memory Model. Appl. Sci. 2023, 13, 2681. https://doi.org/10.3390/ app13042681

[9] A. Jain, K. Rajyalakshmi, L. M. I. L. Joseph, K. Gulati, P. Goel and P. Singh, "A Novel Hybrid Deep Learning Algorithm for Smart City Traffic Congestion Predictions," 2021 6th International Conference on Signal Processing, Computing and Control (ISPCC), Solan, India, 2021, pp. 561-565, doi: 10.1109/ISPCC53510.2021.9609467.

[10] Zafar N, Ul Haq I (2020) Traffic congestion prediction based on Estimated Time of Arrival. PLoS ONE 15(12): e0238200. https://doi. Org/10.1371/journal.pone.0238200

[11] W. Zhao, J. Xu, S. Cong, C. Qu and T. Zhang, "A Hybrid Method of Traffic Congestion Prediction and Control," in IEEE Access, vol. 11, pp. 36471-36491, 2023, doi: 10.1109/ACCESS.2023.3266291

[12] L. Po, F. Rollo, C. Bachechi and A. Corni, "From Sensors Data to Urban Traffic Flow Analysis," *2019 IEEE International Smart Cities Conference (ISC2)*, Casablanca, Morocco, 2019, pp. 478-485, doi: 10.1109/ISC246665.2019.9071639.

[13] Zambrano-Martinez JL, Calafate CT, Soler D, Cano J-C, Manzoni P. Modeling and Characterization of Traffic Flows in Urban Environments. *Sensors*. 2018; 18(7):2020. https://doi.org/10.3390/s18072020

[14] S. Sarkar *et al*., "Effective Urban Structure Inference from Traffic Flow Dynamics," in *IEEE Transactions on Big Data*, vol. 3, no. 2, pp. 181-193, 1 June 2017, doi: 10.1109/TBDATA.2016.2641003.

[15] Nicholas G. Polson, Vadim O. Sokolov, Deep learning for short-term traffic flow prediction, Transportation Research Part C: Emerging Technologies, Volume 79, 2017, https://doi.org/10.1016/j.trc.2017.02.024.