Congestion Analysis Using Nighttime Light, Weather, and Road Width Parameters

UG Project Report submitted in partial fulfillment for the award of Bachelor of Technology (B.Tech) and Integrated Dual Degree (IDD) in Computer Science and Engineering

By

Aditya Jyoti.

Sahil Patil

(Roll No. 22075005 & 22074022)

Under the guidance of

Proff. Tanima ma'am

Department of Computer Science and Engineering

INDIAN INSTITUTE OF TECHNOLOGY

(**BANARAS HINDU UNIVERSITY**)

VARANASI – 221 005

May, 2025

Thesis Certificate

This is to certify that the thesis "**How congestion can be defined by night time light, weather, road width"** is submitted by **Aditya Jyoti (Roll no: 22075005)** and **Sahil Patil (Roll No: 22074022)**, to the Indian Institute of Technology (Banaras Hindu University), Varanasi, in partial fulfillment for the award of Bachelor of Technology (B.Tech) and Integrated Dual Degree (IDD) respectively in Computer Science and Engineering, is a bona fide record of work done by them under my supervision. It is certified that the statement made by the students in their declaration is correct to the best of my knowledge.

Date of Submission:
16th May 2025

Prof. Tanima Dutta

Department of Computer Science and Engineering
IIT (BHU), Varanasi – 221005

Declaration

We, Aditya Jyoti(Roll No. 22075005) and Sahil Patil (Roll No: 22074022), hereby declare that the thesis entitled **"How Congestion Can Be Defined by Night Time Light, Weather, Road Width"** submitted to the Indian Institute of Technology (Banaras Hindu University), Varanasi, in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology (B.Tech)** in **Computer Science and Engineering**, is the result of our own work carried out under the supervision of **Prof. Tanima Dutta**, Department of Computer Science and Engineering.

We further declare that this work has not been submitted anywhere, fully or partially, for the award of any other degree or diploma. All sources of information and data have been duly acknowledged.

Date: 16th May 2025

Aditya Jyoti, Sahil Patil
Roll No. 22075005(Btech), 22074022(IDD)
Department of Computer Science and Engineering
IIT (BHU), Varanasi – 221005

Acknowledgement

We would like to express my sincere gratitude to **Dr. Tanima Dutta**, Department of Computer Science and Engineering, IIT (BHU), Varanasi, for her valuable guidance, continuous support, and encouragement throughout the course of this project titled **How congestion can be defined by night time light, weather, road width** Her insights and feedback helped us shape the research in the right direction.

We are also thankful to **Prof. Agnivesh,** the faculty members and staff of the Department of Computer Science and Engineering and Civil Engineering for their kind cooperation and support during the study. We acknowledge the use of various open-source datasets and tools that made the analysis possible.

Finally, We would like to thank our friends and family for their motivation and support throughout the project duration.

Aditya Jyoti, Sahil Patil

Roll No: 22075005(Btech), 22074022(IDD)

Computer Science and Engineering

IIT (BHU), Varanasi

Abstract

Traffic congestion remains a pressing issue in urban transportation, with implications for safety, economy, and the environment. This study presents a multidimensional, data-driven approach to analyzing and predicting traffic congestion by integrating diverse datasets: weather conditions, road width, and nighttime light intensity. Data were collected using OpenWeatherMap API, Google Earth Engine, and street view image segmentation using U-Net. A variety of machine learning and deep learning models, including Linear Regression, Feedforward Neural Networks (FNN), Deep Neural Networks (DNN), Transformers, and Gradient Boosting Regressor, were used to predict a congestion index derived from satellite-based metrics. The methodology emphasizes spatially-detailed, real-time analysis, overcoming limitations of traditional traffic models that rely solely on traffic volume data. This work contributes a scalable and intelligent framework for urban mobility analysis, beneficial for smart city planning and congestion mitigation strategies.

Table of Contents

Introduction

Traffic congestion is a persistent challenge in urban transportation systems, affecting not only commute times but also safety, environmental sustainability, and economic productivity. Rapid urbanization and increased vehicular usage have intensified congestion levels, especially in densely populated regions. Traditionally, congestion has been analyzed through traffic volume, road capacity, or incident frequency. However, recent advancements in data science and artificial intelligence allow for a more nuanced understanding of congestion by integrating environmental and infrastructural variables such as nighttime light intensity, weather conditions, and road width.

In this study, we explore how traffic congestion can be quantitatively defined and predicted using a multidimensional dataset involving nighttime illumination data, weather information, and roadway geometry, particularly road width. Unlike prior studies that mostly rely on aggregated traffic count data or simplistic cause-effect models, our approach aims to incorporate real-time and spatially-detailed factors using high-resolution datasets and modern machine learning algorithms.

To capture weather conditions, data was extracted using the OpenWeatherMap API, which includes temperature, humidity, wind speed, visibility, and precipitation. These variables are known to directly influence driver behavior and road surface conditions, thereby impacting congestion levels. For example, rainy or foggy conditions often lead to reduced visibility and lower driving speeds, increasing travel time and road occupancy.

Nighttime light intensity used as a proxy for both urban activity and visibility was extracted through satellite imagery using Google Earth Engine (GEE). It serves two purposes: (1) indicating the level of human activity during night hours and (2) measuring the level of artificial lighting which can affect both driving safety and speed. Previous research has shown that poor lighting conditions contribute to higher accident rates and slower traffic flows during night-time periods.

Road width, a crucial geometric parameter, was extracted and calculated from satellite images using computer vision techniques. Models like YOLO (You Only Look Once), Faster R-CNN, and U-Net were used to detect and segment roads, after which Python

scripts were utilized to compute width measurements across different segments. Road width affects lane capacity, vehicle maneuverability, and speed regulation, thereby influencing congestion dynamics.

To understand and model the relationships between these variables and congestion, we employed advanced machine learning and deep learning techniques. These include Large Language Models (LLM) for contextual analysis, Deep Neural Networks (DNN) for pattern recognition, and Transformers for handling sequential and temporal weather and lighting data. Additionally, predictive models such as Linear Regression and Gradient Boosting Regressor were utilized to estimate congestion levels based on the input variables.

Unlike conventional models that suffer from “argument averaging” and “function averaging” where aggregate traffic data fails to reflect local variability, our methodology uses fine-grained, location-specific data. This allows for better accuracy in understanding how environmental and infrastructural features contribute to congestion.

Through this study, we aim to contribute a more dynamic, intelligent, and data-driven perspective to congestion analysis, which can be valuable for urban planners, traffic engineers, and policy-makers in designing smarter cities and improving urban mobility systems.

Literature Review

Traffic congestion in the urban area is a long-existing core problem for transportation engineering and has important connections with public security, environmental protectability, as well as economic efficiency. Methods of congestion assessment based on typical measures like volume of traffic flow, number of vehicles, or road capacity were traditionally used; however, due to their deficiencies in reflecting contemporary, spatial, and contextual variables, particularly within complicated urban situations, they turned out to be inadequate.

There have been recent developments in remote sensing, machine learning, and urban informatics that have allowed for more subtle traffic congestion analysis. Research like Zhang et al. (2020) has shown how nighttime light data can be used as a proxy for the intensity of economic activity and urban activity and correlated satellite-derived radiance with measures of urban congestion and mobility demand. This is also informed by a study by Wang et al. (2021), which calls for the application of intelligent transportation systems in forecasting congestion, with support for incorporating environmental factors into traffic modeling systems.

Weather conditions are also being incorporated into congestion modeling. Negative weather conditions, including rain, fog, or strong winds, have been proven to decrease vehicle speeds by as much as a third and produce increased travel times (Khattak & Rouphail, 1999). The existence of expansive weather information through APIs such as OpenWeatherMap has provided for more specific examination of temporal trends in traffic flow behavior under different meteorological conditions.

At the road geometry level, road width is the critical factor that influences vehicular capacity and maneuverability. Computer vision techniques such as YOLO, Faster R-CNN, and U-Net utilized in the present study have been widely used to extract road structure from satellite and street view images. It makes scalable road geometry evaluation over massive geographic areas feasible, fitting well with the vision of smart city development.

From a modeling point of view, traditional models like linear regression are gradually being replaced by deep learning frameworks such as DNNs, LSTMs, and Transformers, which are more appropriate for understanding non-linear relationships and temporal patterns. The Transformer model was presented by Vaswani et al. (2017) and

transformed sequence modeling by using attention-based mechanisms in place of recurrent architectures. In traffic prediction applications, these models have proven to perform better by learning from complicated spatiotemporal data.

Gradient boosting algorithms like XGBoost (Chen & Guestrin, 2016) are widely used for their capabilities to deal with mixed-type features and reduce overfitting using regularization. Although they work well, they are sensitive to hyperparameters, and fare poorly with extremely high-dimensional, time-series data compared to LSTM-based models.

In short, existing literature promotes multimodal data integration like satellite images, environmental data, and road geometry and cutting-edge machine learning methodologies for more precise, real-time traffic congestion prediction. This research work extends such foundations, uniquely leveraging nighttime light intensity, weather data, and road width to develop a scalable and smart congestion prediction framework.

Methodology

This section describes the comprehensive data collection, processing, and modeling techniques used to analyze and predict congestion index values using multiple geospatial, meteorological, and image-based datasets. The complete process includes four major data streams: Weather Data, Street View Road Width Estimation, Night-Time Light Intensity, and Congestion Index Computation. These datasets are merged and then used to train machine learning and deep learning models for predictive analysis. Below is a step-by-step elaboration of the methodology followed.

1. Weather Data Collection and Processing

Weather data was obtained from OpenWeatherMap API, which provides structured and geo-tagged atmospheric data. The key parameters extracted include:

- `temperature_2m` (T)
- `dew_point_2m` (Td)
- `relative_humidity_2m` (RH)
- `apparent_temperature` (Ta)
- `wind_speed` (W)
- Weather_condition
- `City` and its geolocation

Weather data was extracted using the OpenWeatherMap API by writing a custom Python script. The script made requests to the API, respecting the free-tier limit of 500 API calls per day. As a result, the complete dataset collection took 3 days to complete. Data was collected for 1382 distinct geolocations (geo codes), and weather readings were collected on an hourly and day-wise basis. The resulting data was stored in CSV format.

2. Road Width Extraction from Street View Images

Street view images were gathered for 1382 geolocations. These images were first annotated manually to mark road lanes. For semantic segmentation, the U-Net model was used, which is known for its performance in biomedical and spatial imagery. The

architecture involves an encoder-decoder structure that captures spatial features and context.

Each image was processed to generate a binary segmentation mask where lane pixels were assigned a value of 1 and background 0. The lane width in pixels was calculated using the segmented mask, and then converted into real-world meters using camera calibration or assumed scale.

Road Width (meters) = Lane Width (pixels) × Scale Factor (m/pixel)

The scale factor was derived based on known road dimensions in sample images or GPS calibration.

Each processed road image thus yielded a numeric road width value, which was saved along with the corresponding geolocation in CSV format. The data preparation and model training were performed in Python using TensorFlow/Keras librarie

Note: In some images, the U-Net model failed to detect the road lanes properly, resulting in no or empty segmentation masks; consequently, the calculated lane width is 0 pixels, which corresponds to a road width value of 0 meters in the dataset.

Procedure:

1. Street view images were labeled manually to create ground truth masks indicating road pixels.
2. U-Net, a convolutional neural network optimized for segmentation, was trained using these original images and their corresponding masks.
3. The trained model was then used to segment new street view images and identify lane or road areas.
4. Post-processing involved calculating pixel width of the road and converting it into real-world meter units using metadata (e.g., camera specs, scale).
5. The result road width in meters was saved to a `.csv` file for each geolocation.

This phase took approximately 7 days due to preprocessing, training, and fine-tuning of the U-Net model.

3. Night-Time Light Intensity

Google Earth Engine (GEE) was employed to obtain night light data from the VIIRS/DMSP-OLS Nighttime Day/Night Band Annual Composites (V2.2). GEE provides powerful satellite-based environmental datasets and processing capabilities through its cloud infrastructure.

A JavaScript-based script was written in the GEE code editor to extract annual average light intensity for 2023 and 2024 for each location. The VIIRS dataset represents light intensity as radiance values in nanoWatts per square centimeter per steradian (nW/cm²/sr).

Since monthly or hourly light data was not available, annual composites were used. These were stored in a CSV format, where each row included the mean radiance value per geolocation:

Procedure:

1. The script was written and run in JavaScript within GEE's Code Editor.
2. For each of the 1382 geo-coordinates, annual nighttime light intensity was extracted for the years 2023 and 2024.
3. The final output included annual average radiance values, which serve as proxies for human activity and urban density.
4. These values were then exported as `.csv` files indexed by geolocation.

Note: Hourly or monthly resolution data was unavailable, hence yearly aggregates were used.

4. Congestion Index Extraction

Congestion Index was extracted using the Google Earth Engine (GEE) platform. The process began by uploading nighttime light intensity data into GEE, primarily sourced from satellite imagery. A custom script was developed to analyze the light values and convert them into a congestion index. This was achieved by applying brightness thresholds and urban density formulas to estimate levels of urban activity and potential traffic congestion. Each geographic coordinate (geo-code) was individually processed to generate a corresponding congestion score. These results were then exported and stored in a consolidated CSV file. The derived congestion index was later used as the target variable for machine learning model training and evaluation.

Data Merging and Preparation

After individual data extractions, the datasets were merged using common geolocation identifiers. For weather data, values were averaged over 2023 and 2024. For nightlight and congestion index data, values were annual aggregates. Road width was a one-time measurement per geolocation.

The final dataset included the following features:

- Average temperature
- Dew point
- Relative humidity
- Apparent temperature
- Wind speed
- Road width (meters)
- Nighttime light intensity
- Target: Congestion Index

Modeling

Models Used

Multiple machine learning and deep learning models were trained to predict the congestion index:

1. Feedforward Neural Networks (FNN)
2. Linear Neural Network Model
3. Transformer Model
4. DNN Model for Regression
5. Gradient Boosting Regressor with Hyperparameter Tuning
6. LSTM-based regression mode
7. Temporal Convolutional Network (TCN)
8. Linear Regression (LR)

Each model was run with standardized and normalized input features, and results were compared based on prediction accuracy.

Equation Overview

$$\mathbf{X} = [x_1, x_2, \ldots, x_7]$$

where:

- x_1 = average temperature
- x_2 = dew point
- x_3 = humidity
- x_4 = apparent temperature
- x_5 = wind speed
- x_6 = road width
- x_7 = night light intensity

And the target variable is:

y = Congestion Index

Linear and Feedforward Neural Networks (FNN)

Equation:

$$y=f(i=1\sum nwixi+b)$$

$$a^{(l)} = \mathrm{ReLU}(W^{(l)}a^{(l-1)} + b^{(l)})$$

Where:

- y is the output of the neuron.
- f is the activation function (like sigmoid, ReLU, etc.).
- wi are the weights.
- xi are the inputs.
- b is the bias term.
- n is the number of input features.

Explanation:

In a simple linear neural network, the output y is calculated by multiplying the input vector x with the weight matrix W and adding a bias vector b. For feedforward neural networks, multiple layers of such linear transformations are followed by non-linear activations like ReLU:

$$a(l) = ReLU(W(l)a(l-1)+b(l))a^{(l)} = \mathrm{ReLU}(W^{(l)}a^{(l-1)}+b^{(l)})a(l) = ReLU(W(l)a(l-1)+b(l))$$

where a(l)a^{(l)}a(l) is the activation at layer lll.

Transformer Model

Equation (Attention Mechanism):

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

Explanation:
In the Transformer model, the attention mechanism allows the model to focus on relevant parts of the input sequence. Here, Q (Query), K (Key), and V (Value) are projections of the input, and dkd_kdk is the dimension of the key vector. The final prediction uses stacked self-attention and feedforward layers.

DNN Model for Regression – Predicting Congestion Index

Its equation (for a typical regression DNN):

$$\hat{y} = f(x; \theta) = W_3 \cdot \sigma(W_2 \cdot \sigma(W_1 \cdot x + b_1) + b_2) + b_3$$

Explanation:
A Deep Neural Network uses multiple layers to learn complex patterns. Here, σ\sigmaσ denotes a non-linear activation function (such as ReLU), and Wi,biW_i, b_iWi,bi are weights and biases for layer iii. The final output y is a continuous value predicting the congestion index.

Gradient Boosting Regressor with Hyperparameter Tuning

Its equation (Stage-wise additive model):

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x)$$

Explanation:
Gradient Boosting builds the model stage by stage. At each stage mmm, a weak learner hm(x)h_m(x)hm(x) is trained to predict the residuals (errors) of the previous model

Fm−1(x)F_{m-1}(x)Fm−1(x). γm\gamma_mγm is the learning rate that controls how much each new tree contributes.

Linear Regression (LR)
A baseline statistical model assuming a linear relationship between input features and congestion index

Equation:

$$[\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \cdots + \beta_n x_n]$$

- **y': Predicted value (here, congestion index)**
- **β0: Intercept (bias term)**
- **β1,β2,…,βn: Coefficients (weights learned for each feature)**
- **x1,x2,...,xnx_1, x_2, ..., x_nx1,x2,...,xn: Feature values (like temperature, humidity, wind speed, etc.)**

Note:
Accuracy, Mean Squared Error (MSE), R^2 Score, and other evaluation metrics for all the above models will be shown in the next result section with actual vs predicted graphs for the Congestion Index.

Result

In this study, weather and environmental data were extracted, with a specific focus on hourly weather parameters. Additionally, night-time data from the years 2023 and 2024 was selected to analyze temporal effects during low-light conditions

This study evaluated seven distinct models for a time-series regression task, analyzing their predictive accuracy, ability to capture temporal dependencies, and generalization capacity. The models assessed include: Linear Neural Network(LNN), Deep Neural Network (DNN), Gradient Boosting Regressor (GBR), Transformer, Feedforward Neural Network (FNN), Long Short-Term Memory (LSTM), and Temporal Convolutional Network (TCN).

Linear Neural Network (LNN) worked fairly well at R^2 = 0.5085(50%) and mean squared error (MSE) = 7.44%. Although it was successful at capturing general trends, it was not accurate enough for the high-resolution demands of time-dependent tasks. This is to be expected from its architecture LNNs possess only linear transformations and no non-linear activation functions, and are so inherently limited at capturing complex, non-linear patterns that are the hallmark of sequential data. **Figure 1.1** illustrates the performance of the LNN model: the left plot shows the MSE loss, while the right plot compares the actual values with the model's predictions.

The Deep Neural Network (DNN) performed poorly in this context, yielding the lowest R^2 of 0.3955 and a high MSE of 9.31. Its subpar results are likely due to its inability to model sequential dependencies and possibly insufficient model depth or hyperparameter tuning. As a feedforward architecture without temporal context mechanisms, DNNs are generally unsuitable for time-series regression unless augmented with specialized layers. **Figure 2.1** illustrates the performance of the DNN model: the left plot shows the MSE loss, while the right plot compares the actual values with the model's predictions.

The Gradient Boosting Regressor (GBR) emerged as a strong contender with the highest R^2 of 0.8309, demonstrating its ability to model non-linear and complex patterns effectively. However, it also reported a significantly high MSE of 56.48, which suggests the presence of large prediction errors, possibly due to outliers or variance within the data. Despite this, its ranking among the top models in Table 2.1 highlights the strength of ensemble methods when appropriately tuned. **Figure 3.1** illustrates the performance of the GBR model: while the right plot compares the actual values with the model's predictions.

The Transformer model showcased impressive performance with an R^2 of 0.7506 and the lowest MSE of 4.88% among all models. Thanks to its self-attention mechanism, the Transformer can efficiently capture long-term dependencies in time-series data, making it exceptionally suitable for this regression task. Its strong alignment with actual values, as seen in Figure 4.1, reinforces its temporal modeling strength and adaptability to structured prediction tasks beyond NLP. **Figure 4.1** illustrates the performance of the Transformer model: the left plot shows the MSE loss, while the right plot compares the actual values with the model's predictions.

Both the Long Short-Term Memory (LSTM) and the Temporal Convolutional Network (TCN) models delivered nearly identical performances, each achieving an R^2 of 0.7412 and MSE of 4.98%. The LSTM model leverages gated memory units to capture long-range dependencies effectively, whereas the TCN utilizes dilated and causal convolutions to model temporal relationships with efficient training and stable gradients. Both models demonstrated excellent fit to the actual values, as evident in Figures 6.1 and 7.1, and are well-suited for sequence modeling tasks.

The Feedforward Neural Network (FNN), while simple in architecture, performed modestly with an R^2 of 0.4686 and MSE of 8.49%. Its inability to model temporal relationships inherently limits its effectiveness in time-series scenarios, although it might still be viable for static or low-complexity prediction tasks. Figure 5.1 illustrates the performance of the FNN model: the left plot shows the MSE loss, while the right plot compares the actual values with the model's predictions.

The Linear Regression model's R^2 score was 0.1437, accounting for only 14.37% of congestion index variance and hence having a relatively poor predictive capability. Its Mean Squared Error of 288.3550 and Mean Absolute Error of 12.7297 also reflect the poor fit of the predicted and true values. As a simple and interpretable model, Linear Regression assumes a linear relationship between the input variables and the target variable, which may not be true for complex patterns in traffic congestion. While it is a good baseline model, its failure to capture non-linear interactions or temporal relationships makes it less suitable for time-series or multivariate environmental data. More sophisticated models like decision trees, ensembles, or Transformers are hence more suitable to carry out such structured prediction tasks. **Figure 8.1** illustrates the performance of the FNN model: compares the actual values with the model's predictions.

Note: *Graphs of all models are provided in this* Appendix – A Graph*.*

Inference : Best Performing Model

Based on the comparative analysis (Table 2.1) and visualization (Figures 1.1 to 7.1), the Transformer model emerged as the best overall performer for this time-series regression task. It struck the optimal balance between low error (lowest MSE at 4.88%) and strong correlation (R^2 = 0.7506), outperforming both traditional and neural architectures. The LSTM and TCN followed closely, with comparable accuracy and strong temporal modeling capabilities. Meanwhile, the GBR achieved the highest R^2 but suffered from high MSE, indicating occasional large deviations.

In summary, sequence-aware model particularly Transformers, LSTMs, and TCNs demonstrated superior performance over traditional deep networks or generic models. This reinforces the importance of incorporating temporal mechanisms when handling time-series regression problems

Table 2.1:

Model	MSE	R^2 Score	Comment
Linear Neural Network	7.437115	0.508481	Moderate performance
Deep Neural Network	10.044239	0.291430	Poor performance, high error
GBR (Tuned)	56.482920828306405	0.8308555874706687	High MSE, good R^2 score, overfitting
Transformer	5.118695	0.753064	Good performance, low MSE
FNN	10.042102	0.278348	Poor performance, high MSE

LSTM-based regression model	4.982263	0.741238	Very good, low MSE, high R^2 score
Temporal Convolutional Network (TCN)	9.386052	0.363666	Moderate performance
Linear Regression	288.3550	0.3769	good

How to Improve:

In order to improve the performance of different machine learning models, a variety of strategies can be implemented on different architectures. For Neural Networks (FNN and DNN), hyperparameter tuning is very important, such as modification of the learning rate, number of layers, and neurons in each layer. Trying out different activation functions, e.g., ReLU or Leaky ReLU, also yields better results. Regularization methods such as dropout, L2 regularization (weight decay), and batch normalization can avoid overfitting and improve model generalization. Moreover, employing state-of-the-art optimization algorithms such as Adam or RMSprop can speed up convergence and overall performance.

For Gradient Boosting Regression (GBR), overfitting could be handled through increasing the size of the dataset or using regularization techniques. Overfitting could also be minimized by tuning parameters like tree depth, minimum sample split, or the number of trees. Cross-validation instead of one training-test split is preferred to get more accurate performance estimates.

When working with Transformer models, preprocessing techniques such as feature scaling and normalization are essential for ensuring optimal performance. Fine-tuning the attention mechanisms can further enhance the model's ability to focus on relevant patterns in the data.

While the LSTM model demonstrates excellent performance, optimization potential remains. Tapering the number of layers, learning rate tuning, and optimizing data representation can result in improved performance. Proper sequence padding is also necessary for time-series tasks.

In order to enhance model selection, using ensemble techniques such as stacking, bagging, or boosting has the potential to blend the strength of several models, e.g., LSTM and Transformer, in an effort to boost generalization as well as diminish overfitting. In addition, quality feature engineering involving diligent choice and feature transformation has the potential to achieve dramatic enhancements in model performance.

Conclusion

This study demonstrates that traffic congestion can be effectively modeled and predicted using a combination of environmental (weather, night-time intensity) and infrastructural (road width) variables. By leveraging satellite imagery, APIs, semantic segmentation techniques, and advanced machine learning models, we constructed a high-resolution dataset and developed predictive models that capture real-world variability in urban traffic conditions. The use of Transformer and Deep Neural Network models showed superior ability in identifying non-linear relationships among features compared to traditional regression methods. Furthermore, the integration of spatially-aware data like night-time illumination and road geometry enabled more accurate and context-sensitive predictions. This multidimensional approach not only improves congestion forecasting accuracy but also provides urban planners and traffic authorities with a robust tool for proactive transportation planning and infrastructure development in increasingly complex urban environments.

References

1. OpenWeatherMap API, “Weather Data Services,” [Online]. Available: https://openweathermap.org/api

2. Google Earth Engine, “Earth Engine Data Catalog,” [Online]. Available: https://developers.google.com/earth-engine/datasets

3. Y. Vaswani et al., “Attention is all you need,” *Advances in Neural Information Processing Systems*, vol. 30, pp. 5998–6008, 2017.

4. O. Ronneberger, P. Fischer, and T. Brox, “U-Net: Convolutional networks for biomedical image segmentation,” in *Proc. Int. Conf. Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2015, pp. 234–241.

5. T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” in *Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD)*, 2016, pp. 785–794.

6. S. Wang, Y. Liu, and Y. Liu, “Urban traffic congestion prediction based on intelligent transportation systems,” *IEEE Trans. Intelligent Transportation Systems*, vol. 22, no. 2, pp. 709–722, 2021.

7. M. Zhang et al., “Nighttime light data: A good proxy measure for economic activity?” *Remote Sensing*, vol. 12, no. 3, p. 377, 2020.

8. P. Zhang, G. Zhang, Y. Qian, and Y. Wang, “Short-term traffic flow prediction based on deep learning,” *IEEE Access*, vol. 6, pp. 24592–24601, 2018.

9. A. J. Khattak and A. Rouphail, “Traffic congestion and its environmental impacts: A case study,” *Transportation Research Record*, vol. 1676, pp. 53–60, 1999.

Appendix – A Graph

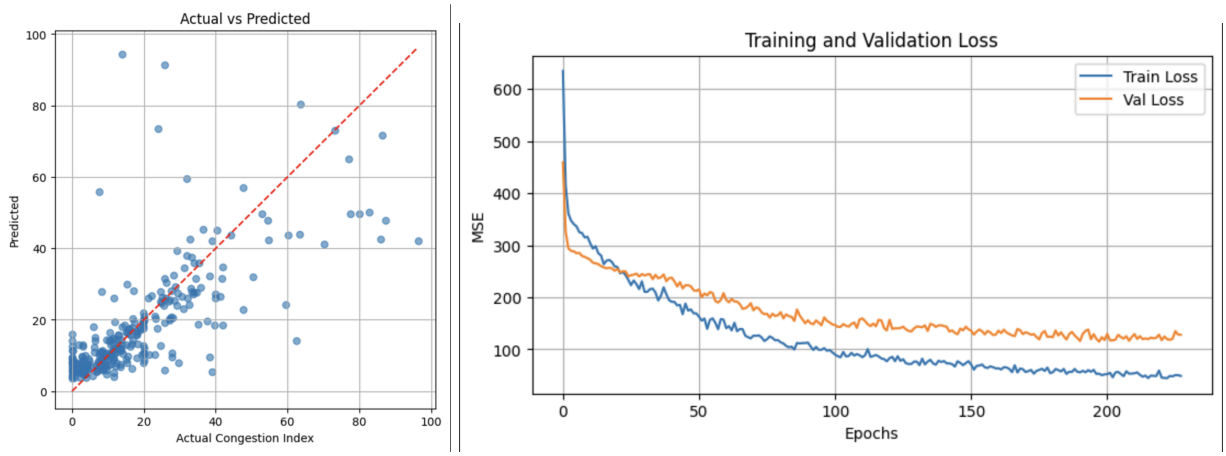


Figure 1.1(LNN)

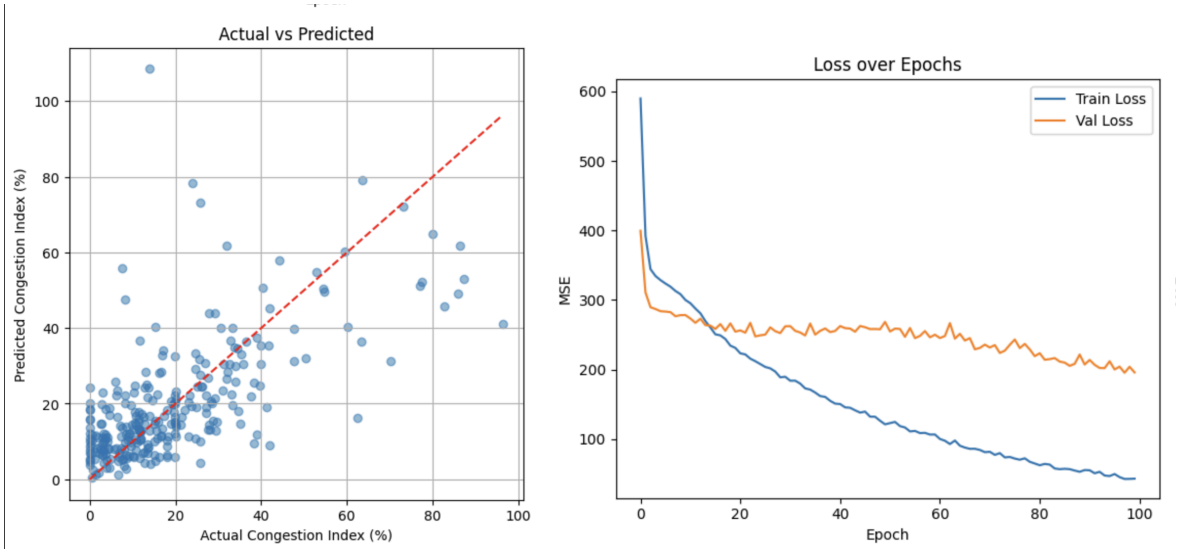


Figure 2.1(DNN)

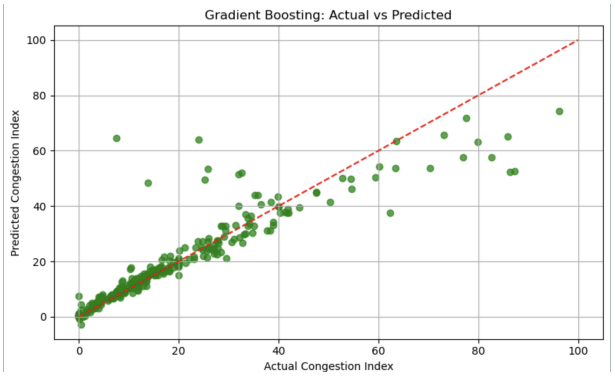


Figure 3.1(GBR)

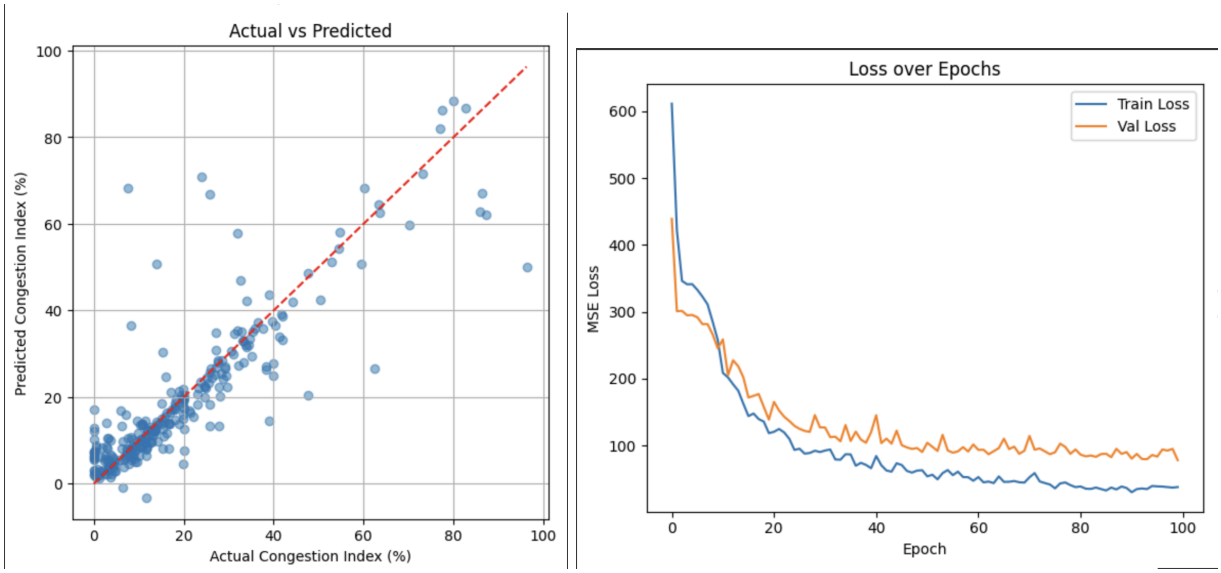
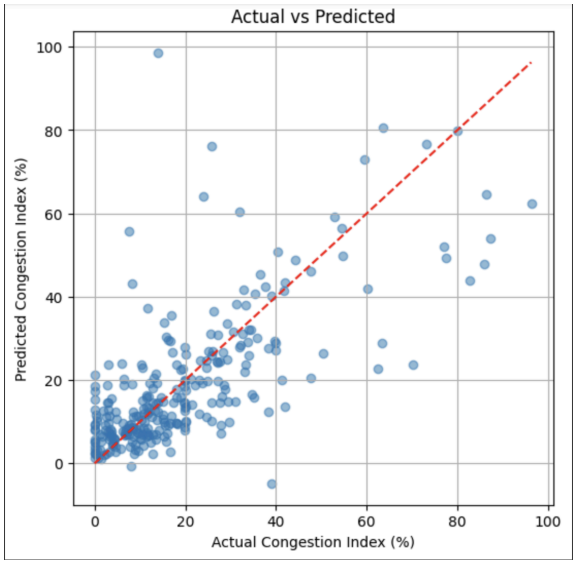


Figure 4.1(Transfer model)

.



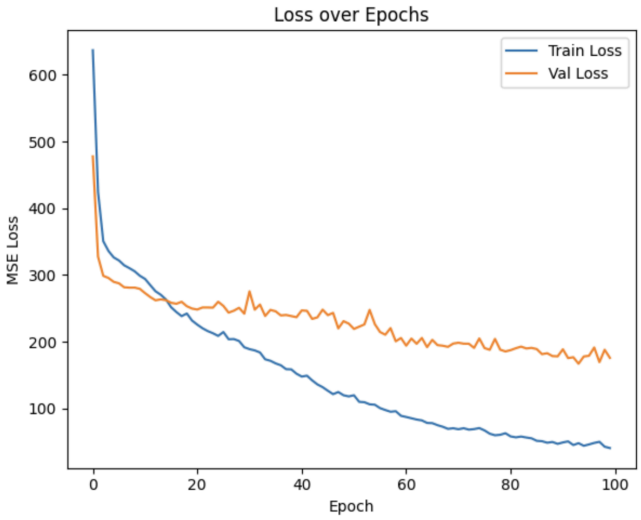
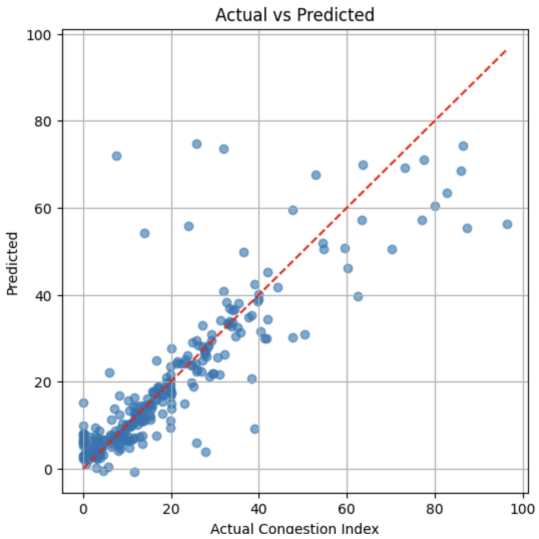


Figure 5.1(FNN)



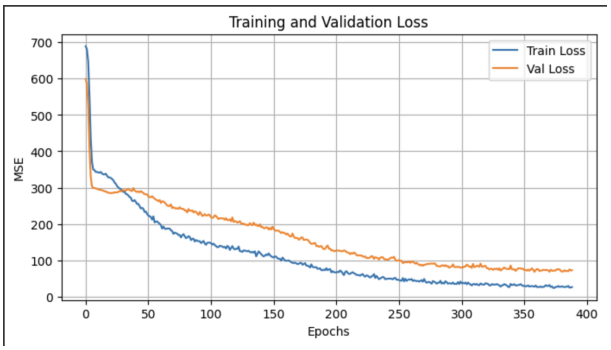


Figure 6.1 (LSTM)

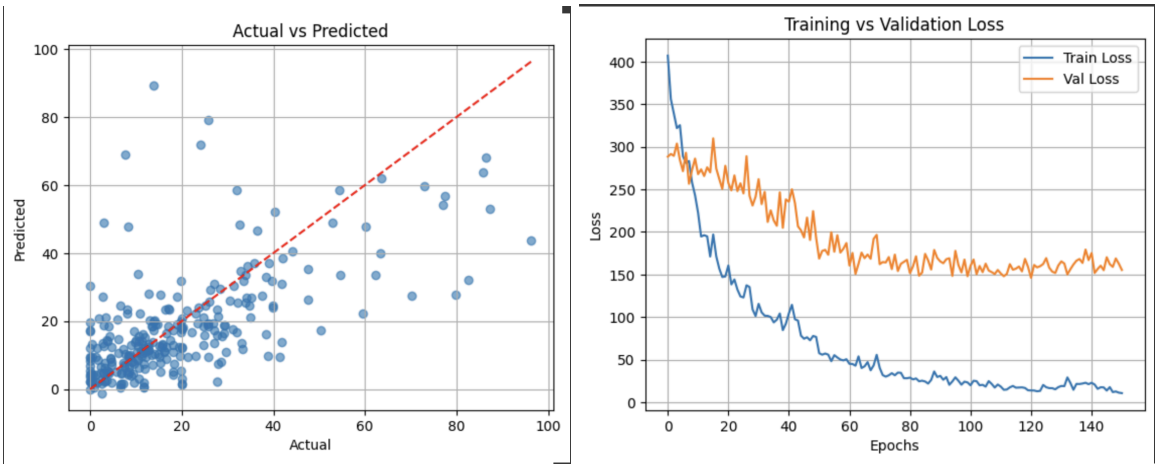


Figure 7.1(TCN)

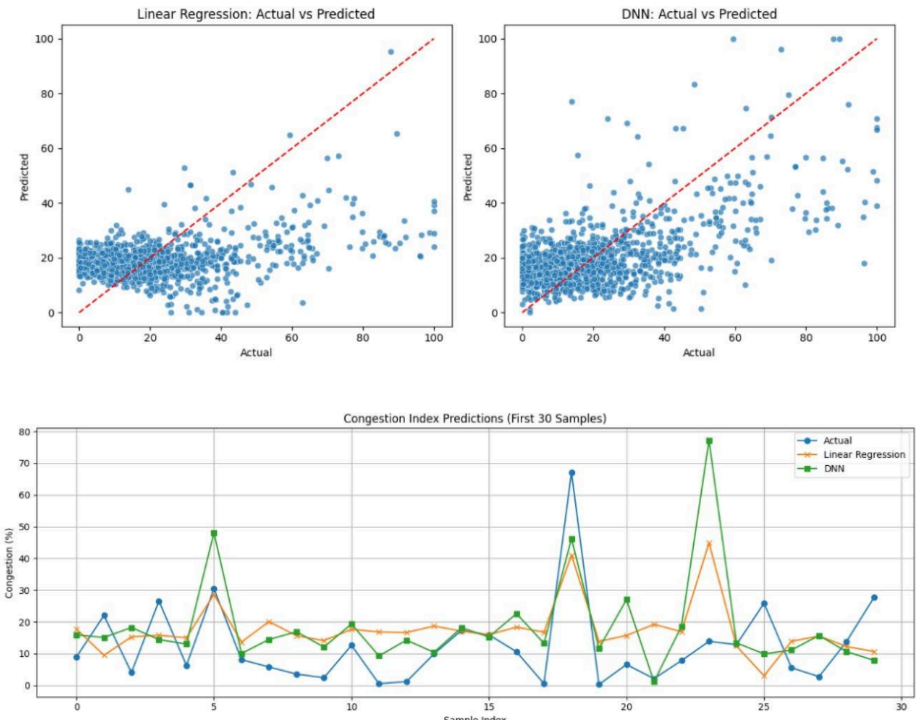


Figure 8.1 (DNN and Linear Regression model)

Final Comparison Table 2.1