Analyzing NPMRDS Using Advanced Machine Learning to Enhance Road Safety, Public Health and System Performance

Prepared by:
Miad Faezipour, Ph.D.[1]
Sangho Park, Ph.D.[2]
Shuju Wu, Ph.D.[2]

Report Number:
CT-2337-F-25-4
Final Report

9/12/2025

Research Project:
SPR-2337

Name of the performing organization
Electrical and Computer Engineering Technology[1]
Purdue University

Department of Electrical & Computer Engineering Technology[2]
Central Connecticut State University

Submitted to:
Connecticut Department of Transportation
Bureau of Policy and Planning
Research Section

TECHNICAL REPORT DOCUMENTATION PAGE

1. Report No. CT-2337-F-25-4	2. Government Accession No.	3. Recipients Catalog No.
4. Title and Subtitle Analyzing NPMRDS Using Advanced Machine Learning to Enhance Road Safety, Public Health and System Performance		5. Report Date 9/12/2025 6. Performing Organization Code
7. Author(s) Miad Faezipour, Ph.D. Sangho Park, Ph.D. Shuju Wu, Ph.D.		8. Performing Organization Report No.
9. Performing Organization Name and Address Central Connecticut State University 1615 Stanley Street New Britain, CT 06050		10. Work Unit No. (TRIS) 11. Contract or Grant No. SPR-2337 13. Type of Report and Period Covered Final Report, 2004-2025
12. Sponsoring Agency Name and Address Connecticut Department of Transportation 2800 Berlin Turnpike Newington, CT 06131-7546		14. Sponsoring Agency Code
15. Supplementary Notes A study conducted in cooperation with the U.S. Department of Transportation and Federal Highway Administration		

16. Abstract

This project aims to leverage NPMRDS data and machine learning techniques to analyze the relationships between speed variations, crash occurrences, weather conditions, and road geometry. The goal is to identify patterns that can measure the performance of road segments and inform targeted safety interventions and infrastructure improvements. The project progressed in an organized sequence We began with systematic data collection and the selection of study segments, followed by rigorous data processing and feature engineering to support both speed prediction and roadway reliability assessment. Multiple modeling approaches are developed, validated, and refined to ensure robustness, while exploratory analyses are conducted to assess crash-related and health-related impacts, extending the scope of insights beyond prediction alone. The findings are carefully examined, leading to actionable recommendations for practice and policy. Implementation activities further demonstrate the practical applicability of the developed methods.

17. Key Words Machine Learning, Traffic Speed, NPMRDS Dataset	18. Distribution Statement No restrictions. This document is available to the public through the National Technical Information Service, Springfield, VA. 22161		
19. Security Classif. (Of this report) Unclassified	20. Security Classif.(Of this page) Unclassified	21. No. of Pages 28	22. Price N/A

DISCLAIMER

The contents of this report reflect the views of the author(s), who is/are responsible for the facts and accuracy of the data presented herein. The contents do not reflect the official views or policies of the State or the Federal Highway Administration. This report does not constitute a standard, specification or regulation. This material is based upon work supported by the Federal Highway Administration under 23 U.S.C. 505(B).

ACKNOWLEDGEMENTS

The team at Central Connecticut State University and Purdue University are very grateful for the funded support from the Connecticut Department of Transportation to work together and bring this project to completion. This research was funded by the Federal Highway Administration under CT-DOT Grant No. SPR-2337.

METRIC CONVERSION FACTORS

APPROXIMATE CONVERSIONS TO SI UNITS				
SYMBOL	**WHEN YOU KNOW**	**MULTIPLY BY**	**TO FIND**	**SYMBOL**
LENGTH				
in	inches	25.4	millimeters	mm
ft	feet	0.305	meters	m
yd	yards	0.914	meters	m
mi	miles	1.61	kilometers	km
AREA				
in^2	square inches	645.2	square millimeters	mm^2
ft^2	square feet	0.093	square meters	m^2
yd^2	square yard	0.836	square meters	m^2
ac	acres	0.405	hectares	ha
mi^2	square miles	2.59	square kilometers	km^2
VOLUME				
fl oz	fluid ounces	29.57	milliliters	mL
gal	gallons	3.785	liters	L
ft^3	cubic feet	0.028	cubic meters	m^3
yd^3	cubic yards	0.765	cubic meters	m^3
NOTE: volumes greater than 1000 L shall be shown in m^3				
MASS				
oz	ounces	28.35	grams	g
lb	pounds	0.454	kilograms	kg
T	short tons (2000 lb)	0.907	megagrams (or "metric ton")	Mg (or "t")
TEMPERATURE (exact degrees)				
^oF	Fahrenheit	5 (F-32)/9 or (F-32)/1.8	Celsius	^oC
ILLUMINATION				
fc	foot-candles	10.76	lux	lx
fl	foot-Lamberts	3.426	candela/m^2	cd/m^2
FORCE and PRESSURE or STRESS				
lbf	poundforce	4.45	newtons	N
lbf/in^2	poundforce per square inch	6.89	kilopascals	kPa

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1 Introduction

1.1 Background and Motivation

Traffic congestion imposes severe economic, environmental, and public health burdens. It increases crash risks while exposing communities to elevated air pollution levels (Zhang et al., 2013). In the U.S. alone, congestion caused an estimated $81 billion in economic losses in 2022 (INRIX). On average, a commuter loses 54 hours and 166 gallons of fuel annually due to congestion, at an approximate personal cost of $1,080 (Ounoughi et al., 2024). As congestion intensifies, average traffic speed typically declines, making speed a leading indicator of delays and bottlenecks within transportation networks (Pandove et al., 2024).

Accurate traffic speed prediction is therefore critical for anticipating and managing congestion. Forecasting future speeds enables transportation agencies to proactively identify when and where congestion is likely to occur. These insights support real-time route optimization, adaptive traffic signal control, and rapid incident response (Liu et al., 2021). Within the broader framework of intelligent transportation systems (ITS), speed prediction helps assess congestion levels, optimize resource deployment, and improve road safety and sustainability (Ounoughi et al., 2024). Predictive capabilities also facilitate strategies such as alternative routing (Zhang et al., 2020), dynamic demand-based pricing (Qian et al., 2015), and automated infrastructure management through lane control and signal timing (De et al., 2011; Zhang et al., 2020).

Researchers have developed a range of approaches for traffic speed forecasting. Traditional statistical models are effective for short-term prediction but often fail to capture the nonlinear and spatio-temporal complexities of real-world traffic systems (Le et al., 2016; Lobo et al., 2016). Machine learning (ML) methods, such as Support Vector Machines (SVM) and Random Forests (RF), handle nonlinearity better but generally lack the ability to capture sequential dependencies in time-series data (Kamarianakis et al., 2012; Bratsas et al., 2019). In contrast, deep learning (DL) architectures—including Long Short-Term Memory (LSTM), Convolutional Neural Networks (CNN), Gated Recurrent Units (GRU), Transformers, and hybrid CNN-LSTM models—have shown superior performance by automatically learning temporal patterns and complex interactions in traffic data (Ma et al., 2015; Tran et al., 2024; Zhang et al., 2020).

1.2 Synthesis of Research Studies on Speed Forecasting

This section synthesizes key research studies that highlight various statistical, ML, and DL-based methods used in speed forecasting for road segments and outlines their quantitative performance metrics.

1.2.1 Statistical Approaches

Wibisana et al. [17] utilized the Greenshield model to predict speeds on arterial roads with a mean absolute error (MAE) of approximately 4.1 miles per hour (mph) under various traffic densities. An advancement in speed forecasting has emerged through the use of Expectation Maximization (EM) combined with Cumulative Sum (CUSUM) algorithms. Cetin and Comert [18] revealed that their model improved average speed prediction to a 90% confidence interval, yielding a Root Mean Squared Error (RMSE) of 1.2 mph.

Beyond singular models, Le et al. [10] utilized Bayesian methodologies to predict traffic speeds. This approach utilized historical and real-time data, achieving an R^2 metric of up to 0.85 for speed predictions. The stochastic frontier approach, as explored by Lobo et al. [11], [19], provides insights into the variability of speeds across different two-lane highway segments. They reported the accuracy of predicted speeds, with R^2 values reaching 0.83 under typical conditions and 0.76 during peak hours [11], [19].

Incorporating real-world incident data further enhances traditional predictive methods. Pan et al. [20] showed that predictions could improve by as much as 91% in accuracy when integrating historical incident patterns with standard traffic variables. Their work highlighted an RMSE reduction from 2.3 to 0.9 mph when contextual factors such as accidents were incorporated into traditional models.

However, traditional statistical approaches operate primarily under the assumption of linear or quasi-linear relationships among traffic parameters. These models simplify complex traffic dynamics, which can sometimes lead to suboptimal outcomes during incidents, adverse weather conditions, or unexpected surges in demand. Gasser & Werner [21] emphasize that traditional methodologies may overlook the inherently non-linear characteristics of traffic systems, failing to adequately account for the rich dynamics present even in simple models. Furthermore, Kamarianakis et al. [12] argue that real-world traffic data exhibits non-stationarity and non-linearity due to rapid oscillations and extreme fluctuations. Dealing with the dynamic nature of traffic systems requires adopting non-linear modeling techniques to improve traffic forecasting performance.

1.2.2 Machine Learning Approaches

The study by Xing et al. [22] proposed a Quantum-behaved Particle Swarm Optimization–Multikernel Extreme Learning Machine (QPSO-MKELM) model for probabilistic traffic flow forecasting. At a 90% confidence level, the model achieved a Prediction Interval Coverage Probability (PICP) of 90.49% and a Prediction Interval Normalized Average Width (PINAW) of 21.45%. Bratsas et al. [13] compared multiple machine learning techniques for predicting traffic speed, including RF, Support Vector Regression (SVR), multilayer perceptron, and multiple linear regression. The SVR achieved the best performance with an MAE of 6.24 mph in urban traffic conditions. Vanajakshi & Rilett [23] evaluated the application of SVR for short-term travel time prediction using real-world data. This

model was tested and compared against the performance of Artificial Neural Networks (ANN) for prediction intervals ranging from 2 minutes to 1 hour. Results showed that SVR consistently outperformed other methods in conditions where training data was limited or exhibited high variability, achieving the lowest mean absolute percentage error (MAPE) of 7.38% for 2-minute-ahead predictions better than ANN (8.64%). However, the ML models used in these works are not inherently designed to capture sequential or temporal dependencies. Traffic speed data are time-series in nature, and the absence of time-aware mechanisms such as memory or recurrence can limit the performance of these models' ability to forecast dynamic, evolving traffic patterns over time [24], [25].

1.2.3 Deep Learning Approaches

Ma et al. [14] employed LSTM networks using remote microwave sensor data, achieving an RMSE of 2.8 mph and outperforming the traditional Autoregressive Integrated Moving Average (ARIMA) method. To further enhance LSTM performance, Tran et al. [15] proposed an ensemble-based LSTM model, which combined multiple LSTM learners for improved generalization. This ensemble approach reduced RMSE and MAE by over 10% compared to single-LSTM configurations.

Zhang et al. [16] developed a 3D-CNN model for network-wide traffic speed forecasting, achieving an R^2 value of 0.88. Yu et al. [26] introduced a graph convolutional network (GCN) model for traffic speed prediction, improving prediction accuracy by 15% compared to traditional models. Li et al. [27] proposed a Diffusion Convolutional Recurrent Neural Network (DCRNN) that combines GCNs with recurrent neural networks, achieving an RMSE of 2.5 mph.

Rajalakshmi & Vaidyanathan [28] presented a CNN-LSTM model that achieved an RMSE of 1.5 mph, demonstrating strong performance on dynamic traffic patterns. Singh et al. [29] extended this idea by combining CNN, GRU, and LSTM networks into a unified architecture, achieving an R^2 exceeding 0.90 and demonstrating superior generalization across multiple traffic scenarios. In another hybrid study, Bi et al. [30] proposed a model that combines a TCN with an LSTM. The model leverages TCNs for capturing long-range temporal features and LSTM for sequence learning, achieving an RMSE of 3.29 mph and an MAE of 2.55 mph, outperforming traditional CNN and LSTM models. Likewise, Mead [31] proposed a hybrid CNN-LSTM Model (HCLM), which outperformed standalone CNN, LSTM, and ARIMA models. For a 75-minute forecast horizon, HCLM achieved an MAE of 6.9 mph and an RMSE of 9.4 mph.

Zhao et al. [32] implemented an attention-based deep learning framework, achieving an RMSE of 0.8 mph and improving accuracy during high-variance periods such as peak traffic. Similarly, Tian et al. [33] developed a dual-GRU model that integrates neighborhood aggregation and attention mechanisms. This model achieved a MAE of 0.6 mph and effectively captured spatial and temporal dependencies in traffic data.

1.3 Objectives of the Study

The main contributions of our research are as follows:

- We present a cleaned, seven-year (2017–2024) time-series dataset from the National Performance Management Research Data Set (NPMRDS) for a specific Traffic Message Channel (TMC) segment '120+05618', aggregated at one-hour intervals. Pre-processing ensures temporal alignment for sequence modeling.
- We conduct a comprehensive evaluation of deep learning models, including RNN, LSTM, GRU, Encoder-Decoder LSTM, Attention-Based Sequence-to-Sequence, Transformer, CNN, hybrid CNN-LSTM, and Temporal Convolutional Network (TCN), under univariate, bivariate, and multivariate configurations.
- We implement a traditional Seasonal AutoRegressive Integrated Moving Average with eXogenous factors (SARIMAX) model as a baseline and show that all deep learning models outperform it.
- We analyze the effect of incorporating temporal features (hour and day) on prediction accuracy. Results indicate that CNN-LSTM benefits from additional contextual features, while GRU and Transformer models perform best with minimal inputs, highlighting their varying sensitivity to contextual enrichment.

CHAPTER 2 Data Collection and Integration (Tasks 1-3)

2.1 Data Collection

We collaborated with CT DOT on various data collection by leveraging NPMRDS dataset in the state of Connecticut. Data extraction and preprocessing was performed on the dataset. Other data sources such as crashes and road geometry are combined with travel time reliability index focusing on interstate highway segments of Connecticut state.

Various factors that can affect traffic speed and travel time were explored by collecting heterogeneous data and mapping them to the common road geometry map. Algorithms and codes were developed to integrate the heterogeneous data sets into the road geometry.

The explored data sets include traffic speed, crash information, precipitation, and number of wet days. Work zone data sets do not exist and are not available for the project.

2.2 Select Study Segments

- A list of candidate road segments have been studied and evaluated for suitability of the work. We have discussed the potential segments with the CTDOT to ensure alignment with broader transportation objectives and regulations.
- We communicated with CCSU and CTDOT on the selection of study segments task.
- Segment '120+05618' has been selected and its data from 2017 to 2024 is analyzed in this project.

2.3 Data Processing and Feature Engineering for Speed Prediction

The NPMRDS dataset includes traffic-related features such as speed, historical average, and volume, used to forecast speed using the data of the segment '120+05618' from 2017 to 2024. The analysis focuses on evaluating various forecasting models and the impact of different feature combinations (univariate, bivariate, and multivariate).

Features

- Speed
- Historical speed
- Volume

Target Variable

The target variable represents future traffic performance (forecasting), evaluated through Mean Squared Error (MSE). A lower MSE indicates better predictive accuracy.

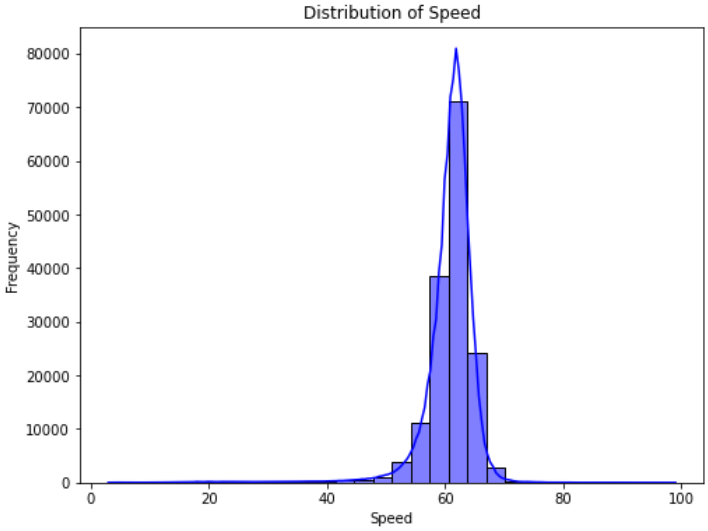

Fig. 2. 1 Histogram Plot of Speed Values

Fig. 2.1 presents histogram of speed values showing the frequency distribution with a superimposed density plot. This histogram reveals the overall distribution of speed values, which appears to be unimodal and slightly skewed. Most speeds cluster around 60 mph, with fewer occurrences of extreme high or low speeds.

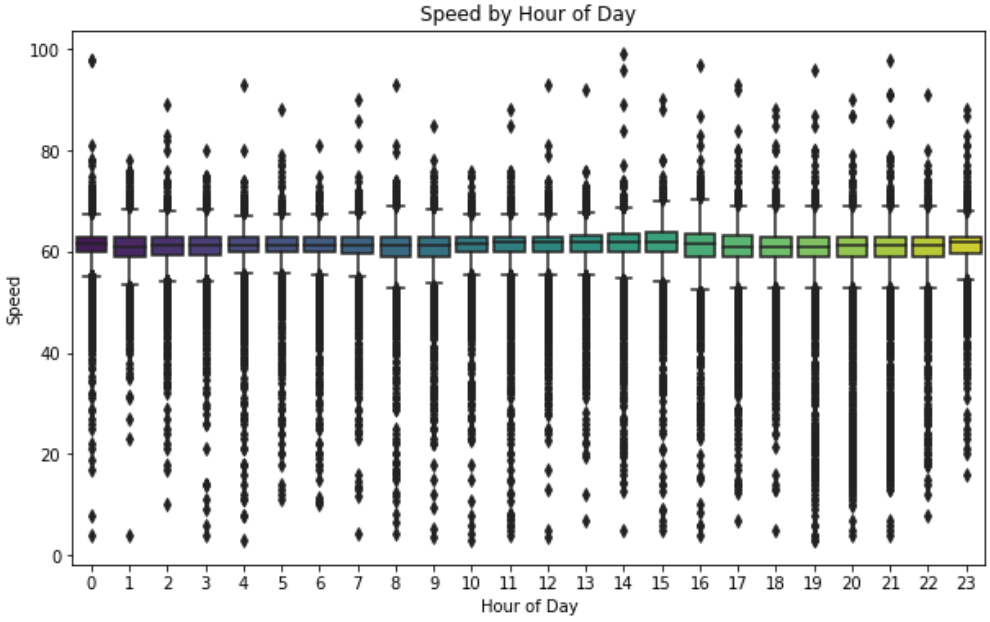

Fig. 2. 2 Boxplot of Speed Value Distribution by Hour of the

Fig. 2.2 presents boxplot illustrating the distribution of speed by hour of the day. This plot reveals temporal variations in speed, indicating whether certain times of the day experience higher or lower speeds. The outliers in the plot represent unusual speed values, which might be due to specific events or conditions.

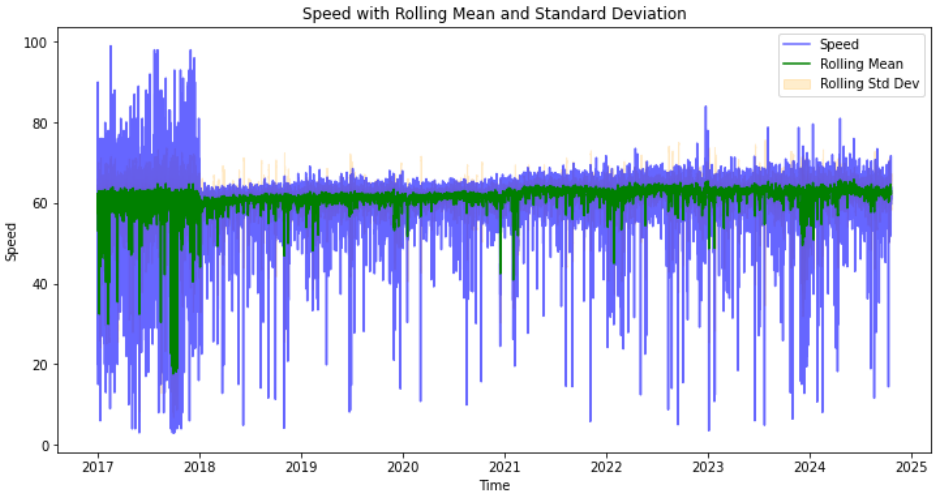

Fig. 2. 3 Rolling Mean and Standard Deviation of Speed Over Time

Fig. 2.3 presents rolling mean and standard deviation of speed over time to highlight trends and variability. The green line represents the rolling mean (average speed over a moving window), while the shaded area reflects rolling variability. The rolling mean suggests a consistent trend in speed after 2018. The reduction in the rolling standard deviation after 2018 indicates a more predictable and stable pattern.

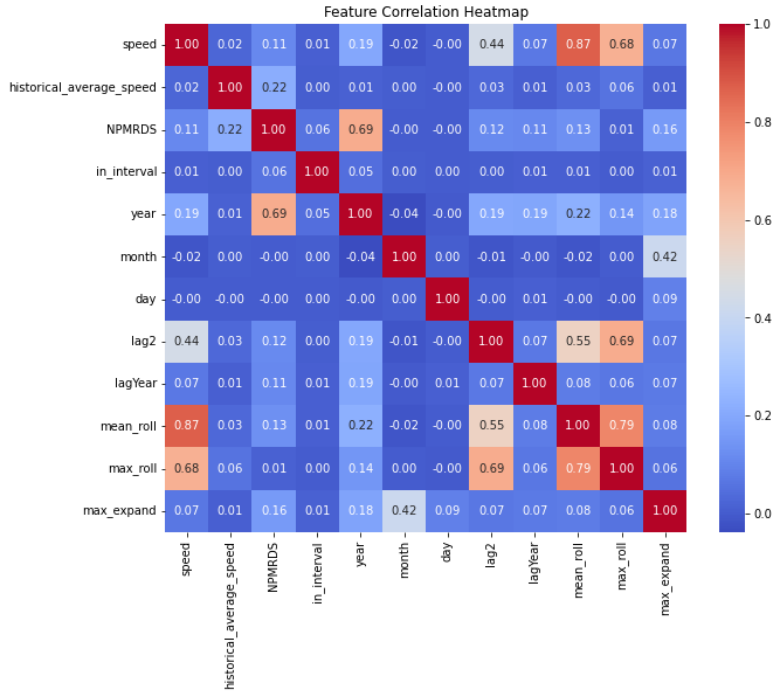

Fig. 2. 4 Correlation Heatmap of Dataset Features

Fig. 2.4 presents correlation heatmap of dataset features, showing the relationships between speed, historical averages, and other variables. This heatmap highlights the linear relationships between features. Strong correlations, such as between Speed and mean_roll, suggest potential

predictors for speed forecasting. Conversely, weak or near-zero correlations indicate minimal direct relationships.

CHAPTER 3 Speed Prediction Modeling (Task 4)

3.1 Methods

This section outlines the detailed methodology of the study, encompassing data pre-processing, model architecture design, training procedures, and evaluation. The overall pipeline, illustrating each stage of the model development process, is depicted in Fig. 3.1.

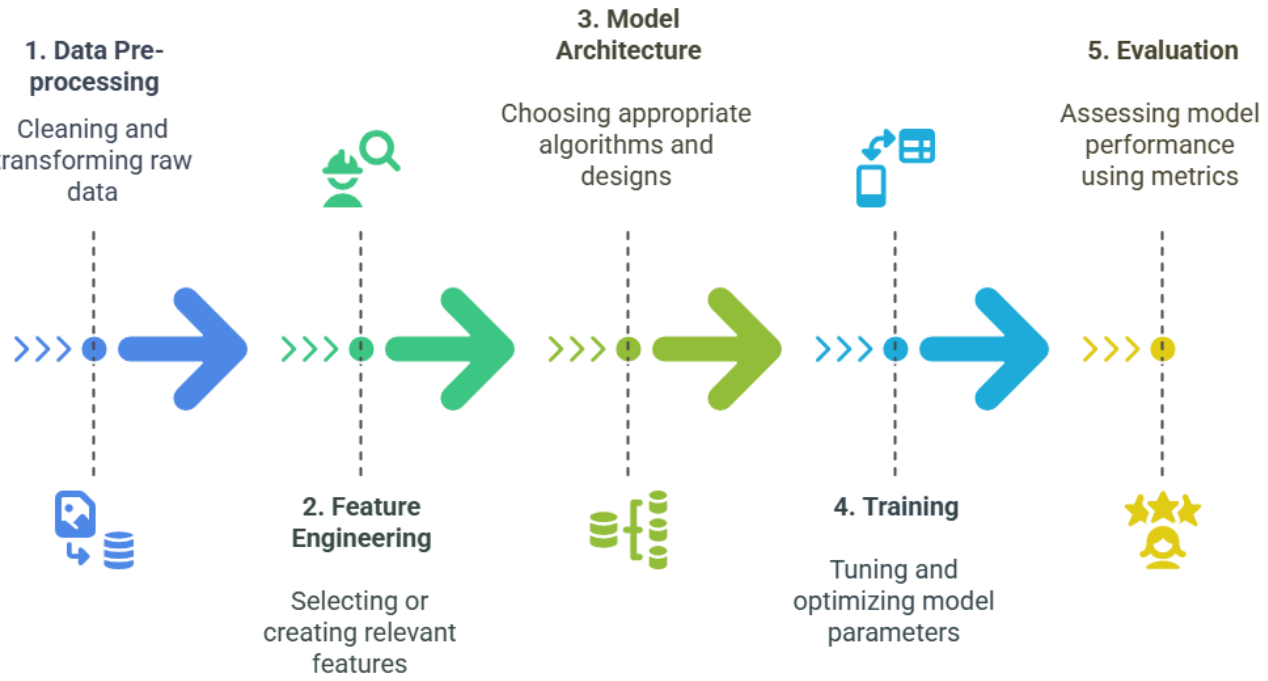

Fig. 3. 1 The Sequential Stages of Machine Learning Model

The pipeline comprises five major components: (1) Data pre-processing, which includes data cleaning and transformation to prepare raw data for modeling; (2) Feature engineering, where relevant features are selected or created to enhance model input; (3) Model architecture, involving the selection of appropriate algorithms and network designs based on the task; (4) Training, where model parameters are tuned and optimized; and (5) Evaluation, where the model's performance is assessed using validation techniques and performance metrics.

3.2 Data Pre-processing

The data preprocessing pipeline for all models begins with loading the raw dataset, which contains sequential traffic-related features such as speed (univariate) and other metrics such as AADT and historical average speed. The dataset is cleaned by removing rows with missing or invalid values. The data set is resampled to hourly frequency, ensuring that each hour has a record, and any missing hourly data points are removed. This is crucial for time-series data to maintain consistent intervals between observations, which is particularly important for temporal models. If the data set includes an in_interval column (indicating crash hours or other event markers), the preprocessing

adjusts the values of this column. Specifically, the in_interval values are updated to account for one hour before and after any identified crash hours, ensuring that the model considers surrounding context for each event. A specific set of features is selected for model training, such as traffic speed, traffic volume (labelled as NPMRDS), and others, based on the task at hand. These features are then normalized using Min-Max scaling to ensure they are in the range [0, 1]. This is important because most machine learning models perform better when input features are on a similar scale, avoiding issues with some features dominating the model's learning process. A sliding window approach is employed to create sequences of length 24 (sequence_length), where each sequence is treated as the model input, and the subsequent timestep serves as the target for prediction. The data is then split into training, validation, and test sets, maintaining consistency in the test set across all experiments. For efficient processing during training, the data is converted into PyTorch tensors and loaded into DataLoader objects for batching and shuffling. This standardized preprocessing ensures consistency and compatibility across all models, enabling a fair comparison of their performances.

In cases where the in_interval column is present, additional preprocessing is applied to enhance contextual awareness. Specifically, for every time point marked as an in_interval (e.g., crash hour), the surrounding one hour before and after is also marked as part of the interval. This adjustment ensures that the models are exposed not only to the event itself but also to its immediate temporal context.

3.3 Traffic Data Features

We downloaded raw time-stamped traffic data for the time period 2017 to 2024 from the NPMRDS, specifically for TMC '120 + 05618' in the state of Connecticut, USA. This TMC is on the National Highway System (NHS) and is about 0.893895 miles in urban areas. We merge the yearly files into a single dataset to form a continuous time series. The dataset includes sequential traffic data such as the target variable speed and contextual and operational features like Annual Average Daily Traffic (AADT), traffic volume, and historical average speed. A set of engineered features is extracted from the merged data to aid in forecasting. These include month, day, hour, day of the week (weekday), and binary flags for weekdays and weekends. A custom categorical feature called period is generated to classify each observation into meaningful time-of-day segments such as morning peak, midday, evening peak, weekend hours. We use a Random Forest Regressor to estimate the relative importance of input features for traffic speed prediction. The model assigns importance scores based on the reduction in mean squared error (MSE) attributed to each feature during the construction of decision trees. Fig. 3.2 shows feature importance scores for traffic speed prediction, aggregated over 24 hours. The figure shows that the historical speed feature exhibits the highest importance by a significant margin, indicating its dominant role in predictive performance. Temporal features such as hour, day, and weekday also contribute to a lesser extent, suggesting that time-based patterns offer supplemental predictive value.

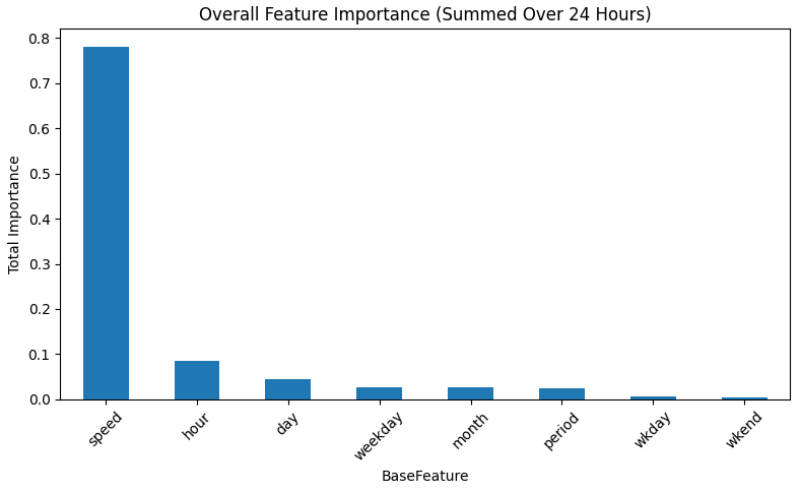

Fig. 3. 2 Feature Importance Scores for Traffic Speed Prediction

All selected features are then normalized using min-max scaling to ensure they fall within the range [0, 1]. Normalization is crucial because it prevents input features with larger ranges from disproportionately influencing the model's learning. Following normalization, the data is structured for supervised learning using a sliding window approach. Here, for each training instance, a sequence of 24 consecutive hourly observations is used as input to predict the traffic speed of the next hour. This design captures short-term temporal patterns relevant to accurate speed forecasting.

Finally, the sequences are split into training, validation, and test sets, with the test set fixed across experiments to ensure consistency in evaluation. The data is converted into PyTorch tensors and wrapped in DataLoader objects to facilitate efficient batch processing and shuffling during training. This standardized and reproducible pre-processing pipeline enables a fair comparison of deep learning-based forecasting models.

3.4 Model Architectures

We design and evaluate nine DL models to predict traffic speed, varying in architecture and input configuration.

3.4.1 Recurrent Neural Network (RNN)

We develop the RNN model with two stacked RNN layers, each with 64 hidden units. The final hidden state is passed through a dense layer for prediction. While effective for short-term dependencies, RNNs often struggle with long-range sequences due to vanishing gradients [wongburi2023prediction, katari2022crop].

3.4.1 Long Short-Term Memory (LSTM)

The LSTM model is designed to handle long-term dependencies using its gating mechanism. This model is particularly effective for time series tasks requiring memory of long-term dependencies [pylov2023algebraic, azzouni2018neutm]. We have developed the LSTM model comprising two

stacked LSTM layers, each with 64 hidden units. The LSTM layers process sequences considering the shape (batch size, sequence length, input size), where the input size corresponds to the number of input features. The final hidden state of the last time step is mapped through a dense layer to generate the output.

3.4.1 Gated Recurrent Unit (GRU)

The GRU model is a simplified and computationally efficient alternative to LSTMs, utilizing fewer gates while retaining the ability to capture temporal dependencies [pandey2022framework, abumohsen2023electrical]. Like the LSTM model, we developed the GRU comprising two stacked layers with 64 hidden units. It processes input sequences of the same format and outputs the final prediction using a dense layer connected to the last hidden state of the sequence. The GRU's efficiency makes it suitable for scenarios with computational constraints while maintaining performance in sequential tasks.

3.4.1 Encoder-Decoder LSTM Model

This model adopts a sequence-to-sequence architecture using separate LSTM-based encoder and decoder layers. The encoder processes the input sequence and outputs the final hidden and cell states passed to the decoder. The decoder generates predictions timestep-by-timestep, with the final output being mapped through a dense layer. This architecture is particularly suited for tasks requiring outputs of different lengths from the input sequence [liu2019improving, abumohsen2023electrical]. The encoder processes the input sequence and updates its hidden state. The final hidden state of the encoder is used as the context vector. The decoder generates the output sequence based on the previous output, hidden state, and the context vector.

3.4.1 Attention-Based Sequence-to-Sequence Model

The attention-based sequence-to-sequence model enhances the traditional encoder-decoder architecture by incorporating an attention mechanism. The attention mechanism allows the model to handle sequences with variable importance across time steps effectively [liang2018automated, khandelwal2025enhancing]. The LSTM-based encoder processes the input sequence and generates hidden states in this architecture. An attention layer computes weights for each encoder output, focusing on relevant parts of the sequence. The context vector formed by weighted summation is passed to the LSTM-based decoder, which generates predictions.

3.4.1 Transformer Model

The transformer model uses self-attention mechanisms to model long-range dependencies in sequences. The model's ability to handle long-range dependencies makes it a robust choice for time-series data with complex relationships across time steps [hua2019deep]. This architecture includes a linear embedding layer to project the input to a higher-dimensional space and multiple transformer encoder layers. A dense layer maps the final representation from the transformer to the output. We

have used a linear embedding layer (d_model=64) and multiple encoder layers with four heads (nhead=4).

3.4.1 Convolutional Neural Network (CNN)

The CNNs effectively extract spatial features from data [wiratmo2020indonesian, mulyani2022comparison]. We utilize a CNN model employing two convolutional layers, each having 64 filters and a kernel size of 3, to extract local spatial features from the input. Rectified Linear Unit (ReLU) activations are applied, followed by global average pooling and a dense layer for final prediction.

3.4.1 CNN-LSTM

This hybrid model architecture integrates the advantages of CNNs (spatial modeling) and LSTMs (temporal modeling) to address the complex spatio-temporal nature of data. This architecture models sequences with prominent spatial-temporal relationships [muhammed2023deep, ghorbani2020convlstmconv]. We have applied a convolutional layer with 32 filters and a kernel size of 3, processing sequences with the input reshaped to batch size, channels, and sequence length. The output of the CNN is passed to the LSTM layers, configured with 64 hidden units and two layers. Finally, the output from the LSTM is mapped to the final output through a dense layer.

3.4.1 Temporal Convolutional Network (TCN)

The TCN is another hybrid model that uses dilated convolutions to capture long-range dependencies in sequential data. Dilation allows the receptive field to grow exponentially with depth. In our model, three convolutional layers with dilation factors d = 1, 2, 4 are used to efficiently expand the temporal receptive field. Each layer also incorporates residual connections to improve gradient flow and stability. We have used the ReLU activation function and dropout for regularization. The output is passed through a dense layer.

3.5 Training

All models are trained using the Adam optimizer with a learning rate 0.001 and mean squared error (MSE) loss. Training is performed on batches of size 32, with early stopping employed to retain the best-performing models based on validation loss.

3.6 Evaluation

The performance of each model is evaluated using mean absolute error (MAE) on the test set, with results averaged across multiple runs to ensure robustness. The MAE is a widely used regression metric measuring the average magnitude of errors between predicted and actual values. It shows how wrong the predictions are, on average, in the same unit as the target variable.

3.7 Implementation

The implementation of all deep learning models is carried out using Python as the primary programming language. We have utilized PyTorch as the deep learning framework. The data manipulation and pre-processing were performed using Pandas and NumPy libraries. Scikit-learn was used for feature scaling and label encoding. Matplotlib was employed to generate plots comparing actual versus predicted traffic speeds for model evaluation and visualization.

The experiments are conducted on CPU and GPU environments to assess computational efficiency. GPU-based training is performed on NVIDIA A100 GPUs with 40 GB memory configurations. On average, training a single model for 50 epochs took approximately 10 minutes on the GPU compared to around 50 minutes on the CPU, showcasing the performance gains from hardware acceleration. The experiments were reproducible across different runs using fixed random seeds.

3.8 Results

Table 3.1 summarizes the performance of various deep learning models in predicting speed using different combinations of the input features. The bold values represent the lowest MAE for each model across different feature combinations. To provide a traditional baseline for comparison, we also implemented a SARIMAX (Seasonal AutoRegressive Integrated Moving Average with eXogenous variables) model using univariate input (speed only). The SARIMAX model achieved an MAE of 2.30 mph, worse than all deep learning models, even those using only the speed feature. The best performing deep learning model in the univariate setting was GRU (MAE = 1.74 mph), representing a 24% improvement over SARIMAX. This comparison highlights the advantage of neural network architectures in capturing non-linear temporal patterns that SARIMAX struggles to model. The addition of temporal features such as hour and day had varying impacts across the models. In general, models like CNN-LSTM, CNN, and Encoder-Decoder LSTM showed improved or stable performance when more features were included, indicating their ability to extract and utilize temporal context. For example, CNN-LSTM achieved its best MAE (1.75 mph) when both speed and hour were used, outperforming its univariate version. On the other hand, models like GRU, LSTM, and transformer performed best with minimal input, suggesting potential overfitting or lack of benefit from additional contextual features in those cases.

Table 3. 1 Model performance comparison (MAE) for speed prediction

Models	Speed (univariate)	Speed, Hour (bivariate)	Speed, Day (bivariate)	Speed, Hour, Day (multivariate)	Mean ± sd
Attention_Seq2Seq	**1.85**	1.87	1.80	1.97	1.872 ± 0.062
CNN_LSTM	1.78	**1.75**	1.83	1.89	1.812 ± 0.053
CNN	1.79	**1.77**	1.84	1.79	**1.798 ± 0.026**
Enc_De_LSTM	1.79	**1.78**	1.80	1.85	1.805 ± 0.027
GRU	**1.74**	1.87	1.80	1.85	1.815 ± 0.05
LSTM	1.78	1.87	**1.77**	1.84	1.815 ± 0.042
RNN	**1.83**	1.91	1.86	1.89	1.872 ± 0.03
Transformer	**1.85**	1.98	2.18	2.12	2.032 ± 0.128
TCN	**2.17**	2.27	2.20	2.24	2.22 ± 0.038

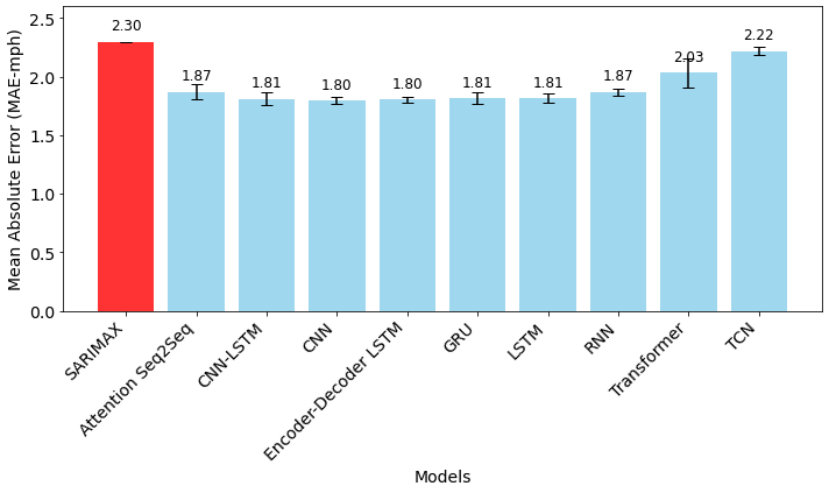

Fig. 3. 3 Model performance using MAE with Standard Deviation (SD) Error Bars

In Fig. 3.3 the baseline SARIMAX model is shown in red, while all other models are deep learning-based. SARIMAX, which uses only univariate input (speed), performs the worst (MAE = 2.30 mph), highlighting its limitations in capturing non-linear and temporal dependencies. In contrast, models such as GRU, CNN, and CNN-LSTM achieve significantly lower MAEs, demonstrating the advantage of data-driven architectures in learning complex spatio-temporal patterns. Error bars indicate variability in performance across different input configurations (univariate, bivariate, and multivariate), with models like Transformer and TCN exhibiting higher variance. suggests that the model generalizes well for normal traffic conditions. There is no significant phase shift between actual and predicted values, meaning the model correctly predicts speed changes in the expected timeframe but struggles with some extreme fluctuations.

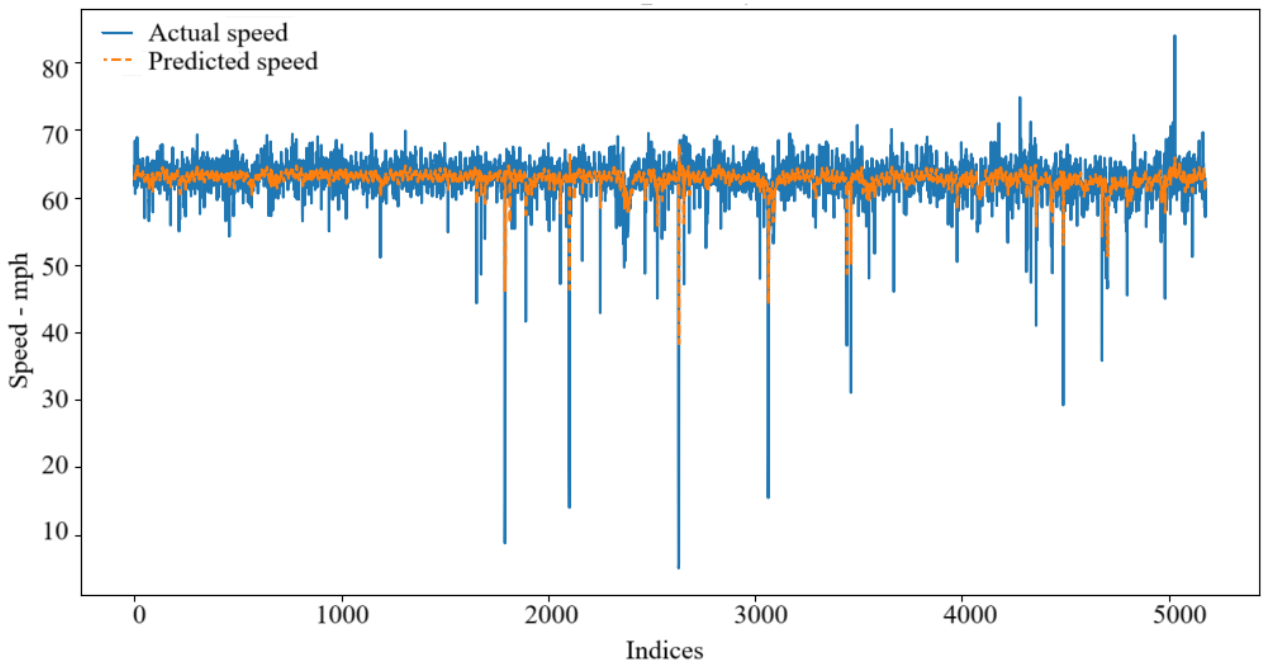

Fig. 3. 4 Actual vs. predicted outcomes using the best performing CNN model

Fig. 3.4 shows the actual vs. predicted outcomes achieved from CNN model using speed and hour. The predicted speed (orange dashed line) closely follows the overall pattern of the actual speed (blue line), indicating that the model effectively learns long-term dependencies in the data. The predicted values remain stable and well-aligned with actual values in regions where speed variations are moderate. This suggests the model generalizes well for normal traffic conditions. There is no significant phase shift between actual and predicted values, meaning the model correctly predicts speed changes in the expected time frame but struggles with some extreme fluctuations.

CHAPTER 4 Other Exploratory Analysis (Tasks 5-7)

4.1 Feature Engineering for Crash Prediction

The Crash Data Repository (CTCDT) dataset was utilized for crash-related feature engineering tasks. The dataset went through several processing steps, including merging multiple files. The missing values were checked and handled, where categorical columns were imputed with their mode while numeric columns were filled using the median. Categorical features were converted to numerical values to be compatible with machine learning models. Numerical values were also normalized to prevent any single feature from having too much impact on the model. After cleaning and processing, the dataset ended up with 207 features.

For feature selection and engineering, the study utilized a Random Forest (RF) model, an ensemble learning method that builds multiple decision trees trained on random subsets of the dataset. The importance of each feature was assessed based on its contribution to impurity reduction, which was measured using Gini impurity at each split within the decision trees. The overall feature importance scores were obtained by averaging impurity reduction values across all trees in the forest.

A study was conducted using a subset of the most significant features identified through RF to predict crash severity. The dataset contains a column with three levels of crash severity. These three are vehicle only (O), mild injury (A), and killed (K). Initially, the RF model ranked the top 20 features based on importance to predict the severity of the crash. From these 20 features, a refined selection of six key features was used: Number of Motor Vehicles, Manner of Crash/Collision Impact, Weather Condition, Time of Crash, Month, and Longitude.

We have split the dataset into training and testing sets to train the model. 80% of the data (278,488 samples) was used for training, and 20% (69,622 samples) for testing. The model's performance was evaluated on the test data, yielding an accuracy of 99.93%. The classification report displayed near-perfect precision, recall, and F1 scores across all crash severity categories. The confusion matrix shows that the model is reliable, with few misclassifications. Table 4.1 provides a detailed classification report. Fig. 4.1 presents the confusion matrix, highlighting the model's effectiveness in predicting crash severity based on the chosen features.

The precision, recall, and F1-scores for all three severity classes: A (injury), K (killed), and O (vehicle only) are very close to 1.00, indicating minimal misclassifications. From the confusion matrix, most instances were correctly classified, with only a few misclassified cases. Specifically, Class A had 31,470 correct predictions, Class K had all 6 cases correctly classified, and Class O had 38,146 correct predictions, with minor misclassifications. These results confirm that the selected features effectively distinguish between crash severity levels, making the model highly reliable for real-world crash severity prediction.

Table 4. 1 Classification report of crash severity prediction

	Precision	Recall	F1-score
A	0.9996	0.9990	0.9993
K	1.0000	1.0000	1.0000
O	0.9992	0.9996	0.9994
Accuracy	0.9993		

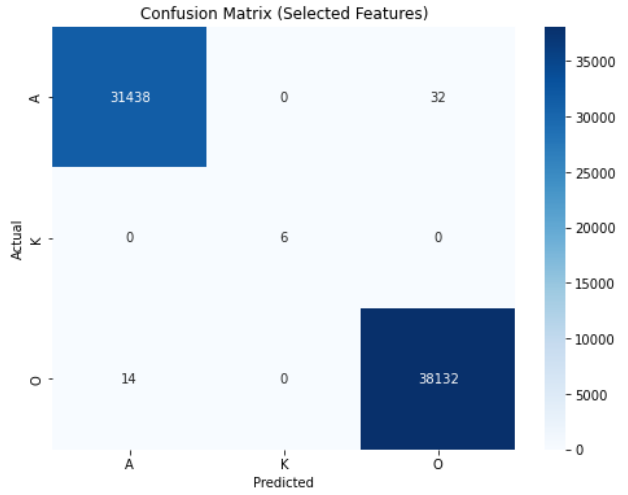

Fig. 4. 1 Confusion matrix of the crash severity features

4.2 Crash Related Impact – Exploratory task

Connecticut Crash Data Repository (CTCDT) dataset is utilized for crash related impact exploration tasks. After merging, cleaning, and processing the original dataset contains 207 features. The data processing steps include handling missing values, categorical encoding, and feature scaling. Categorical columns with missing values are filled with their mode and numeric columns with median. Categorical variables are encoded into numeric format to make them suitable for machine learning models. Numerical values are normalized by ensuring that all features contribute equally. The findings are presented in the third quarterly report.

4.3 Study Health Related Impact – Exploratory task

We have explored health attributes (asthma, coronary heart disease, and diabetes) and air pollution in relation to the proximity of transportation infrastructure (e.g., highway, public transit) using census tract data. The data preprocessing phase involves extracting infrastructure locations, computing census tract centroids, and determining the shortest distance from each centroid to the nearest transportation infrastructure.

Shapefiles, a geospatial vector data format used in Geographic Information Systems (GIS), were utilized to extract the locations of census tract centroids and transportation infrastructure. Spatial analysis techniques, including centroid computation and proximity analysis, are used to compute the shortest distance from each centroid to the nearest transportation infrastructure, such as highways and public transit stations. 3770 such centroids were extracted for this task. These computed distances were then integrated with health attribute data corresponding to each census tract.

Then the final analysis is performed including visualizing health attributes against these distances to identify potential trends and correlations, interpreting the relationship between proximity to transportation infrastructure and health outcomes. The findings are presented in the third quarterly report.

CHAPTER 5 Recommendations and Implementation (Tasks 8-9)

The speed prediction model results confirm that deep learning architectures such as Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), Convolutional Neural Network-LSTM (CNN-LSTM), CNN, Attention-based Seq2Seq, and Transformer effectively capture temporal dependencies and forecast traffic speed using historical speed measurements and time-based features. Several machine learning models including Logistic Regression, Random Forest, Support Vector Machine (linear and RBF kernels), Gradient Boosting, XGBoost, and Multilayer Perceptron are developed using crash data, roadway attributes, and Annual Average Daily Traffic (AADT) to predict roadway reliability. These models provide valuable insights into the factors influencing reliable and unreliable road segments, supporting data-driven decisions for traffic management and safety planning.

5.1 Recommendations

Looking ahead, several avenues are recommended to further enhance predictive performance and strengthen the model's value for real-world traffic management.

5.1.1 Integration of Additional Exogenous Variables

Traffic patterns are influenced by a wide array of external factors beyond historical speed and timestamp features. Future work should incorporate exogenous variables that directly or indirectly affect roadway conditions. Potential variables include:

Weather attributes: Hourly or sub-hourly measurements of temperature, rainfall, snowfall, fog, wind speed, and visibility. These factors have well-documented effects on congestion and travel speeds. Weather Application Programming Interface (APIs) such as National Oceanic and Atmospheric Administration (NOAA), OpenWeatherMap, or local Department of Transportation (DOT) data feeds can be aligned with traffic timestamps.

Incident and event data: Road closures, accidents, construction activities, and major events (sporting events, concerts) can create localized disruptions. Integrating incident reports and event schedules enables the model to account for sudden anomalies.

Calendar effects: Public holidays, school schedules, and seasonal travel patterns influence traffic load. Binary holiday flags or school term indicators can be engineered to capture such dynamics. Incorporating these features enriches the context available to the model and allows it to better generalize to atypical conditions.

5.1.2 Enhanced Feature Engineering and Temporal Resolution

The current lag and rolling features provide a foundation, but more nuanced representations can improve accuracy:

Congestion indices and regime classification: Features capturing the ratio of observed speed to free-flow or reference speed help quantify congestion severity. Models can also jointly classify congestion regimes (free-flow, moderate, heavy) alongside regression, improving stability.

Seasonality signals: Incorporate Fourier or wavelet-based seasonal terms to capture weekly, monthly, and yearly periodicities that recur in traffic data.

Lagged exogenous features: Introduce lagged versions of weather or event indicators to reflect delayed effects (e.g., snow accumulation impacting traffic over several hours).

Higher temporal granularity: If data availability permits, using 5- or 15-minute intervals rather than hourly can capture sharper dynamics such as sudden slowdowns or peak-hour onset. Downstream models may then forecast both short-term and longer horizons.

5.1.3 Advanced Modeling Approaches

Our current implementation leverages a diverse set of sequence models, including LSTM, GRU, Simple RNN, CNN, CNN-LSTM hybrids, Transformers, and Attention-based Seq2Seq architectures. This variety already captures both short-term patterns and long-range dependencies in traffic speed data. Based on these foundations, we recommend the following directions to enhance accuracy, robustness, and interpretability:

Enhance Attention and Hybrid Architectures: We have already integrated hybrid designs such as CNN-LSTM (convolutions for local temporal patterns, LSTMs for longer trends) and attention-based models (Transformers and Attention Seq2Seq). Future improvements can include richer attention mechanisms such as multi-head attention or Temporal Fusion Transformers (TFT), which provide multi-horizon forecasts and interpretable feature importances. These upgrades would make our current attention models more powerful for variable-length and multi-step forecasting.

Residual Hybrid Models (Statistical-Neural): To capture both linear and nonlinear components effectively, we can combine a statistical time-series model (e.g., Seasonal Autoregressive Integrated Moving Average (SARIMA) or exponential smoothing) with our existing deep models. The statistical layer would model recurring seasonal patterns, while the neural network predicts nonlinear residuals. This gray-box approach improves stability and generalization across regimes.

Uncertainty-Aware and Probabilistic Forecasting: Instead of producing only point estimates, our models can be extended to output prediction intervals or quantiles using techniques like quantile regression heads or Bayesian neural nets. This will quantify uncertainty, which is crucial for risk-aware traffic management and planning under uncertain conditions (e.g., weather disruptions or incidents).

Spatio-Temporal Extensions: When data from multiple connected road segments are available, incorporating spatial correlations can significantly improve forecasts. Adding graph-based layers (e.g., Graph Convolution or Diffusion Convolution) on top of our attention or CNN-LSTM

architectures would enable the model to exploit upstream/downstream traffic dependencies while continuing to learn temporal dynamics effectively.

Physics-Informed AI Models: Integrating domain knowledge from traffic flow theory can make predictions more robust and interpretable. Physics-guided approaches involve adding soft constraints based on the fundamental diagram (flow-density-speed relationships), conservation laws, or Cell Transmission Models (CTM). These can be incorporated as regularization terms in the loss function or as differentiable physics layers. This helps the model respect physical limits (e.g., non-negative speeds, capacity bounds) and improves extrapolation during rare events.

By strengthening our existing hybrid and attention models, adding uncertainty estimation, and exploring spatio-temporal and physics-driven methods, we can build upon the strong foundation already implemented in our code. These enhancements will lead to models that are not only more accurate, but also more interpretable, generalizable, and suitable for deployment in real-world intelligent transportation systems.

5.1.4 Model Explainability and Robustness Testing

For stakeholders to trust and adopt forecasts, interpretability is critical:

Explainable Artificial Intelligence tools: Appling SHapley Additive exPlanations (SHAP), Local Interpretable Model-agnostic Explanations (LIME), or integrated gradients to quantify the contribution of each feature (e.g., weather, time, lagged speed) to a given prediction. For attention models, visualize attention weights across time and features.

Stress testing under rare scenarios: Evaluating model robustness during unusual but important conditions such as snowstorms, hurricanes, or holiday traffic surges. Simulating or collecting data on these edge cases ensures resilience in deployment.

Fairness and bias audits: Assessing whether errors are disproportionately higher for certain regions, times, or conditions; if so, apply bias mitigation strategies.

5.1.5 Practical Deployment Considerations

Transitioning from a research prototype to an operational tool requires engineering and cost considerations:

Real-time data pipelines: Design streaming architectures to ingest live sensor and weather data, process features on the fly, and feed predictions to dashboards or control systems.

Cost-efficient infrastructure: Utilize preemptible/spot Graphics Processing Unit (GPUs) or autoscaling clusters to manage training and inference cost. Implement checkpointing to handle interruptions gracefully.

Model monitoring and retraining: Deploy monitoring for data drift and model performance; establish retraining pipelines at regular intervals or when performance degrades.

5.1.6 Comprehensive and Granular Evaluation Metrics

While Mean Absolute Error (MAE) is a standard metric, relying solely on it can mask important nuances:

Multiple error metrics: Reporting Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), and median absolute error to capture different aspects of model performance.

Segmented evaluation: Analyzing errors across different hours of day, weekdays vs. weekends, weather conditions, and congestion levels to identify systematic weaknesses.

Out-of-sample validation: Testing models on future time horizons and entirely different road segments to ensure generalization beyond the training context.

Benchmarking: Comparing against naïve baselines (last observed value, seasonal average) and simple statistical models to contextualize improvements.

By systematically enriching the dataset with contextual variables, deepening the feature set, exploring hybrid and probabilistic models, and embedding explainability and robust evaluation, this project can evolve into a truly production-ready traffic forecasting system. Such a system would not only deliver accurate speed predictions but also provide interpretable insights, resilience under diverse scenarios, and scalable deployment for intelligent transportation applications.

5.2 Implementation

The Implementation task of this project comprised the working codes of the various parts/models designed for this project. We also fine-tuned the code to make it more robust and consistent across models. Specifically, we improved the data preprocessing pipeline by adding lag features, rolling statistics, and period-based categorical encoding. We ensured categorical and Boolean variables were properly encoded, scaled all features with MinMaxScaler, and aligned predictions with inverse transformations to recover the true scale of speed. In the training loop, we implemented validation-based model selection by saving the best-performing state during training. Finally, we added dynamic plotting and saving prediction results with filenames that include both the model's name and its corresponding Mean Absolute Error (MAE), making it easier to compare models. All codes have been refined and have been provided to Connecticut DOT.

5.3 Publication/Presentation

A conference paper publication has been resulted from this work (accepted, presented, currently in press):

S. Rahman, M. M. Rahman, A. Rahman, S. Park, S. Wu, and M. Faezipour, “Deep Learning-based Time-Series Prediction of Traffic Speed Using NPMRDS Dataset,” (accepted to appear), *International Conference on Artificial Intelligence* (ICAI-CSCE’25), 2025.

References

1. Pandey, S., Biswas, T., & Kumar, S. (2022). Framework for stock market prediction using deep learning technique.
2. Abumohsen, M., Owda, A. Y., & Owda, M. (2023). Electrical load forecasting using LSTM, GRU, and RNN algorithms. Energies, 16(5), 2283.
3. Muhammed, A. O., Isbeih, Y. J., El Moursi, M. S., & Al Hosani, K. H. (2023). Deep learning-based models for predicting poorly damped low-frequency modes of oscillations. IEEE Transactions on Power Systems, 39(2), 3257–3270.
4. Ghorbani, M., Bahaghighat, M., Xin, Q., & Özen, F. (2020). ConvLSTMConv network: a deep learning approach for sentiment analysis in cloud computing. Journal of Cloud Computing, 9(1), 16.
5. Hua, Y., Zhao, Z., Li, R., Chen, X., Liu, Z., & Zhang, H. (2019). Deep learning with long short-term memory for time series prediction. IEEE Communications Magazine, 57(6), 114–119.
6. Wongburi, P., & Park, J. K. (2023). Prediction of wastewater treatment plant effluent water quality using recurrent neural network (RNN) models. Water, 15(19), 3325.
7. Katari, S., Bhowmik, T. K., Nair, S. S., Nayak, A. R., & Pankajakshan, P. (2022). Crop phenology stage forecasting and detection using NDVI time-series and LSTM. IGARSS 2022 IEEE International Geoscience and Remote Sensing Symposium, 6264–6267.
8. Haider, S. A., Naqvi, S. R., Akram, T., Umar, G. A., Shahzad, A., Sial, M. R., Khaliq, S., & Kamran, M. (2019). LSTM neural network based forecasting model for wheat production in Pakistan. Agronomy, 9(2), 72.
9. Liu, X., Liu, Y., Zhang, M., Chen, X., & Li, J. (2019). Improving stockline detection of radar sensor array systems in blast furnaces using a novel encoder–decoder architecture. Sensors, 19(16), 3470.
10. Wiratmo, A., & Fatichah, C. (2020). Indonesian Short Essay Scoring Using Transfer Learning Dependency Tree LSTM. International Journal of Intelligent Engineering & Systems, 13(2).
11. Mulyani, Y., Septiangraini, D., Muhammad, M. A., & Nama, G. F. (2022). Comparison Study of Convolutional Neural Network Architecture in Aglaonema Classification. International Journal of Electronics and Communications Systems, 2(2), 75–83.
12. Liang, G., On, B.-W., Jeong, D., Kim, H.-C., & Choi, G. S. (2018). Automated essay scoring: A siamese bidirectional LSTM neural network architecture. Symmetry, 10(12), 682.
13. Khandelwal, R., & Goyal, H. (2025). Enhancing Drought Forecasting Using GNN-LSTM with Attention Mechanism: A study of Jaisalmer district, Rajasthan.
14. Pylov, P., Maitak, R., & Protodyakonov, A. (2023). Algebraic reconfiguration of LSTM network for automated video data stream analytics using applied machine learning. E3S Web of Conferences, 458, 09023.
15. Azzouni, A., & Pujolle, G. (2018). NeuTM: A neural network-based framework for traffic matrix prediction in SDN. NOMS 2018 IEEE/IFIP Network Operations and Management Symposium, 1–5.

16. Wibisana, H., Estikhamah, F., & Rodhi, N. N. (2025). Comparative analysis of vehicle speed on arterial roads with the greenshield model approach. IOP Conference Series: Earth and Environmental Science, 1454(1), 012053.
17. Zhu, D., Shen, G., Liu, D., Chen, J., & Zhang, Y. (2019). FCG-ASpredictor: an approach for the prediction of average speed of road segments with floating car GPS data. Sensors, 19(22), 4967.
18. Le, T. V., Oentaryo, R., Liu, S., & Lau, H. C. (2016). Local Gaussian processes for efficient fine-grained traffic speed prediction. IEEE Transactions on Big Data, 3(2), 194–207.
19. Lobo, A., Couto, A., & Rodrigues, C. (2016). Flexible stochastic frontier approach to predict spot speed in two-lane highways. Journal of Transportation Engineering, 142(8), 04016032.
20. Pan, B., Demiryurek, U., & Shahabi, C. (2012). Utilizing real-world transportation data for accurate traffic prediction. 2012 IEEE International Conference on Data Mining, 595–604.
21. Lobo, A., Rodrigues, C., & Couto, A. (2014). Estimating percentile speeds from maximum operating speed frontier. Transportation Research Record, 2404(1), 1–8.
22. Cetin, M., & Comert, G. (2006). Short-term traffic flow prediction with regime switching models. Transportation Research Record, 1965(1), 23–31.
23. Gasser, I., & Werner, B. (2010). Dynamical phenomena induced by bottleneck. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 368(1928), 4543–4562.
24. Kamarianakis, Y., Shen, W., & Wynter, L. (2012). Real-time road traffic forecasting using regime-switching space-time models and adaptive LASSO. Applied Stochastic Models in Business and Industry, 28(4), 297–315.
25. Zhao, Z., Chen, W., Wu, X., Chen, P. C. Y., & Liu, J. (2017). LSTM network: a deep learning approach for short-term traffic forecast. IET Intelligent Transport Systems, 11(2), 68–75.
26. Ma, X., Dai, Z., He, Z., Ma, J., Wang, Y., & Wang, Y. (2017). Learning traffic as images: A deep convolutional neural network for large-scale transportation network speed prediction. Sensors, 17(4), 818.
27. Jiao, J., & Wang, H. (2023). Forecasting traffic speed during daytime from Google Street View images using deep learning. Transportation Research Record, 2677(12), 743–753.
28. Yu, B., Yin, H., & Zhu, Z. (2017). Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. arXiv preprint arXiv:1709.04875.
29. Lemieux, J., & Ma, Y. (2015). Vehicle speed prediction using deep learning. 2015 IEEE Vehicle Power and Propulsion Conference (VPPC), 1–5.
30. Goudarzi, S., Kama, M. N., Anisi, M. H., Soleymani, S. A., & Doctor, F. (2018). Self-organizing traffic flow prediction with an optimized deep belief network for internet of vehicles. Sensors, 18(10), 3459.
31. Bratsas, C., Koupidis, K., Salanova, J.-M., Giannakopoulos, K., Kaloudis, A., & Aifadopoulou, G. (2019). A comparison of machine learning methods for the prediction of traffic speed in urban places. Sustainability, 12(1), 142.

32. Xing, Y., Ban, X., & Guo, C. (2017). Probabilistic forecasting of traffic flow using multikernel based extreme learning machine. Scientific Programming, 2017(1), 2073680.
33. Vanajakshi, L., & Rilett, L. R. (2007). Support vector machine technique for the short term prediction of travel time. 2007 IEEE Intelligent Vehicles Symposium, 600–605.
34. Li, Y., Yu, R., Shahabi, C., & Liu, Y. (2017). Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. arXiv preprint arXiv:1707.01926.
35. Ma, X., Tao, Z., Wang, Y., Yu, H., & Wang, Y. (2015). Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. Transportation Research Part C: Emerging Technologies, 54, 187–197.
36. Singh, V., Sahana, S. K., & Bhattacharjee, V. (2025). A novel CNN-GRU-LSTM based deep learning model for accurate traffic prediction. Discover Computing, 28(1), 38.
37. Rajalakshmi, V., & Vaidyanathan, S. G. (2022). Hybrid CNN-LSTM for traffic flow forecasting. Proceedings of 2nd International Conference on Artificial Intelligence: Advances and Applications: ICAIAA 2021, 407–414.
38. Tian, X., Du, L., Zhang, X., & Wu, S. (2023). Mat-wgcn: Traffic speed prediction using multi-head attention mechanism and weighted adjacency matrix. Sustainability, 15(17), 13080.
39. Mead, M. A. (2022). Hybrid CNN and LSTM Model (HCLM) for Short-Term Traffic Volume Prediction. International Journal of Intelligent Computing and Information Sciences, 22(4), 51–61.
40. Bi, J., Zhang, X., Yuan, H., Zhang, J., & Zhou, M. (2021). A hybrid prediction method for realistic network traffic with temporal convolutional network and LSTM. IEEE Transactions on Automation Science and Engineering, 19(3), 1869–1879.
41. Tran, D. Q., Tran, H. Q., & Van Nguyen, M. (2024). An Enhanced Ensemble-Based Long Short-TermMemory Approach for Traffic Volume Prediction. Computers, Materials & Continua, 78(3).
42. Zhang, S., Zhou, L., Chen, X., Zhang, L., & Li, M. (2020). Network-wide traffic speed forecasting: 3D convolutional neural network with ensemble empirical mode decomposition. Computer-Aided Civil and Infrastructure Engineering, 35(10), 1132–1147.
43. Liu, D., Tang, L., Shen, G., & Han, X. (2019). Traffic speed prediction: An attention-based method. Sensors, 19(18), 3836.
44. Saha, S., Haque, A., & Sidebottom, G. (2024). Multi-Step Internet Traffic Forecasting Models with Variable Forecast Horizons for Proactive Network Management. Sensors, 24(6), 1871.
45. Fu, X., Wu, M., Ponnarasu, S., & Zhang, L. (2023). A hybrid deep learning approach for real-time estimation of passenger traffic flow in urban railway systems. Buildings, 13(6), 1514.
46. INRIX. (2024). INRIX 2024 Global Traffic Scorecard. Retrieved from https://inrix.com/scorecard/
47. Schrank, D., Albert, L., Eisele, B., & Lomax, T. (2021). Urban Mobility Report--Appendix A: Methodology. Texas A&M Transportation Institute.
48. Ounoughi, C., & Yahia, S. B. (2024). Sequence to sequence hybrid Bi-LSTM model for traffic speed prediction. Expert Systems with Applications, 236, 121325.
49. De Gier, J., Garoni, T. M., & Rojas, O. (2011). Traffic flow on realistic road networks with adaptive traffic lights. Journal of Statistical Mechanics: Theory and Experiment, 2011(04), P04008.

50. Qian, Z., & Rajagopal, R. (2015). Optimal dynamic pricing for morning commute parking. Transportmetrica A: Transport Science, 11(4), 291–316.
51. Zhang, Y., Li, Y., Wang, R., Hossain, M. S., & Lu, H. (2020). Multi-aspect aware session-based recommendation for intelligent transportation services. IEEE Transactions on Intelligent Transportation Systems, 22(7), 4696–4705.
52. Zhang, R., Ishikawa, A., Wang, W., Striner, B., & Tonguz, O. K. (2020). Using reinforcement learning with partial vehicle detection for intelligent traffic signal control. IEEE Transactions on Intelligent Transportation Systems, 22(1), 404–415.
53. Zhang, K., & Batterman, S. (2013). Air pollution and health risks due to vehicle traffic. Science of the Total Environment, 450, 307–316.
54. Pandove, G. (2024). Enhancing urban mobility: predicting traffic congestion with optimized ML model. Engineering Research Express, 6(4), 045242.
55. Liu, D., Xu, X., Xu, W., & Zhu, B. (2021). Graph convolutional network: Traffic speed prediction fused with traffic flow data. Sensors, 21(19), 6402.