IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Traffic Volume Forecasting Using LSTM and EDA

Lakshman Srinivasan K, Monisha V, Ramesh Babu R, Rupakesavan

Engineering Student, Engineering Student, Engineering Student, Assistant Professor
Dept. of Computer Science and Engineering
Sri Venkateswara College of Engineering, Sriperumbudur, India

Abstract: "Traffic Volume Forecasting Using LSTM and EDA" is designed to address the pressing issue of urban traffic congestion by developing a robust predictive model for accurate traffic volume forecasting. The approach starts with comprehensive Exploratory Data Analysis (EDA) to identify and understand key patterns, trends, seasonal variations, and correlations within the traffic dataset, facilitating informed feature selection and preparation. These insights from EDA help refine the dataset, enhancing the model's overall effectiveness. Following this analysis, a Long Short-Term Memory (LSTM) neural network is employed due to its powerful capability to capture and learn temporal dependencies and long-term trends in sequential time-series data, which is essential for reliable forecasting. The LSTM model is trained, validated, and fine-tuned for optimal performance, with accuracy evaluated using metrics such as Mean Absolute Error (MAE) and Root Mean Square Error (RMSE). By integrating EDA and LSTM, the project establishes a data-driven framework that aids in strategic traffic management and urban planning, ultimately contributing to smoother traffic flow, reduced congestion, and better decision-making in transportation systems.

1.INTRODUCTION

1.1 MACHINE LEARNING

Machine learning is a subset of Artificial Intelligence involving the application of computer algorithms that employs computation processes that can improve automatically through experience by the use of data. The primary intention of machine learning is to allow machines to learn autonomously without any human intervention or assistance and adjust its functions accordingly to them. In the practical usage of computer science, there are certain scenarios where the problem that is to be solved using algorithms are either not ideal or the parameters are way too complex to structure and formulate. During those scenarios traditional instruction based algorithms fail to implement an efficient application.

A counterintuitive approach that overcomes this issue is by approaching algorithms via real time data, this approach lays the foundation of machine learning algorithms. Machine learning algorithms build a model based on input data, known as "training data", in order to make predictions or decisions without being explicitly programmed. This data can be in any point of the spectrum between decomposed form to normalised form depending upon the application. Although Machine Learning (ML) predominantly involves applied statistics, The scope of machine learning is beyond them. Machine learning algorithms are used in cases and situations where the rules for evaluation are not discrete and definite.

ML has proven valuable because it can solve problems at a speed and scale that cannot be duplicated by the human mind alone. With massive amounts of computational ability behind a single task or multiple specific tasks, machines can be trained to identify patterns in and relationships between input data and automate routine processes. The algorithms that drive machine learning are critical to success. ML algorithms build a mathematical model based on sample data, known as “training data,” to make predictions or decisions without being explicitly programmed to do so. This can reveal trends within data that information businesses can use to improve decision making, optimise efficiency and capture actionable data at scale.

Machine learning systems are known for operating in a black box, meaning you have no visibility into how the machine learns and makes decisions. Thus, if you identify an instance of bias, there is no way to identify what caused it. Your only recourse is to retrain the algorithm with additional data, but that is no guarantee to resolve the issue. ML provides the foundation for AI systems that automate processes and

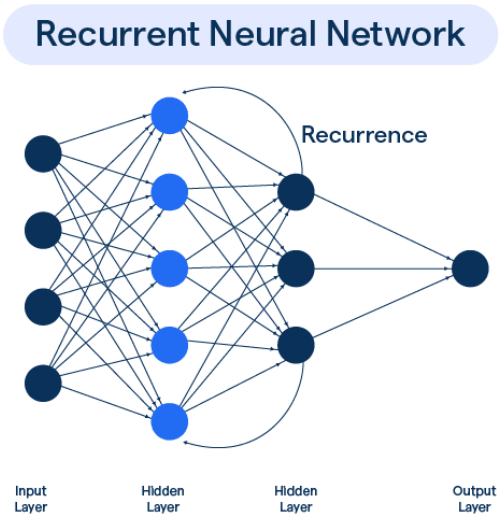
solve data-based business problems autonomously. It enables companies to replace or augment certain human capabilities.

Common machine learning applications you may find in the real world include chatbots, self-driving cars and speech recognition, Traffic Prediction, Product Recommendations, Self-driving cars, Email Spam and Malware Filtering, Virtual Personal Assistant, Online Fraud Detection.

1.2 RECURRENT NEURAL NETWORKS

Recurrent Neural Networks (RNNs) are a type of neural network architecture uniquely designed to process sequential data by maintaining a memory of previous inputs through a loop mechanism in their hidden state, enabling them to handle tasks where context and order are critical, such as language processing, speech recognition, and time series forecasting.

Unlike traditional feedforward neural networks, which treat each input independently, RNNs can process entire sequences by carrying information from prior steps forward, updating the hidden state with each new input, which allows them to detect temporal dependencies within data. At each time step, an RNN cell takes in the current input and the hidden state from the previous step, applying learned weights and an activation function (like tanh or ReLU) to compute the updated hidden state and, when needed, an output.



This architecture provides RNNs with the capability to remember past inputs up to a certain length, making them suitable for applications like text generation, where understanding the sequence of previous words is essential for predicting the next one, or in music generation, where the network must recall previous notes to produce coherent output.

Despite their powerful capabilities, standard RNNs face challenges such as the vanishing gradient problem, where gradients become exceedingly small during backpropagation, making it difficult for the network to learn long-term dependencies. To address this, advanced variants such as Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs) were developed. LSTM cells use mechanisms called gates (input, output, and forget gates) that regulate the flow of information, allowing these networks to retain and forget data selectively, thus overcoming the limitations of basic RNNs and effectively learning long-term patterns without vanishing gradients.

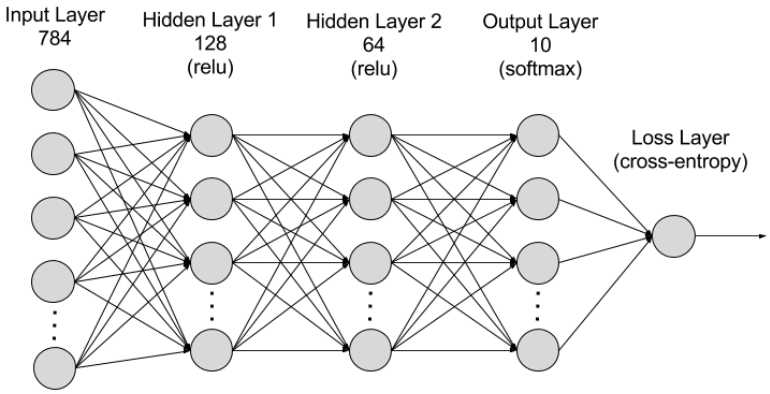
GRUs, on the other hand, simplify this approach with fewer gates, which helps reduce computational complexity while still maintaining performance on sequential tasks. RNNs and their variants have found widespread application in natural language processing tasks such as language modeling, machine translation, and sentiment analysis, where understanding the order and meaning of words is vital, as well as in speech-to-text systems and time series analysis, such as forecasting financial trends.

Their step-by-step processing aligns with the sequential nature of these problems, making them a backbone of sequence modeling.

However, training RNNs can be computationally intensive, and the sequential nature of their processing limits parallelization compared to more modern architectures like Transformers. To further enhance RNN performance, techniques such as gradient clipping, which prevents exploding gradients by capping them, and

the integration of attention mechanisms, which allow the network to focus selectively on different parts of the sequence, have been employed.

Attention mechanisms revolutionized how sequential data is handled by enabling RNNs to prioritize specific segments of input when generating output, leading to significant improvements in tasks that require understanding long contexts, such as translating entire paragraphs of text.

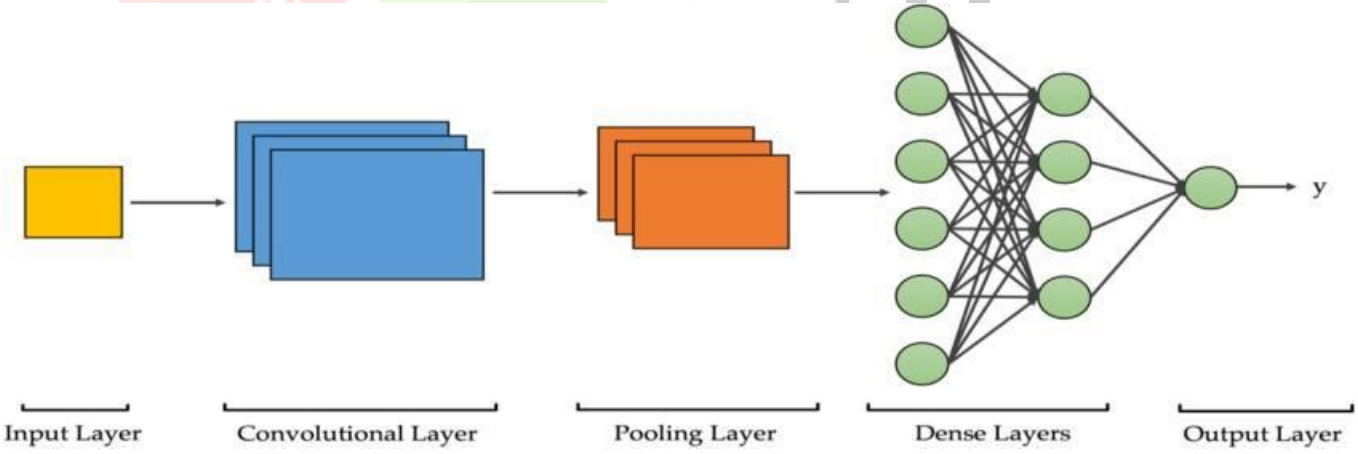


In essence, RNNs, with their capacity for sequence learning and context retention, form the basis for deep learning models that deal with ordered data, from predicting future values in time series to generating language and understanding spoken words, even as their limitations have prompted innovations that extend their effectiveness**.**

Recurrent Neural Networks (RNNs) have unique strengths and quirks. They're excellent at generating sequences, from predicting stock prices to crafting text in a specific writing style. RNNs can even "compose" music by learning patterns in existing compositions. However, they face intriguing challenges: over time, they can "forget" earlier information due to vanishing gradients, making it hard to learn long-term dependencies. Some creative adaptations, like using multiple layers or stacking RNNs with transformers, have helped tackle this. Additionally, "attention mechanisms" are sometimes added to RNNs, allowing them to selectively focus on certain input parts, vastly improving performance in complex tasks.

1.3 Convolutional Neural Networks

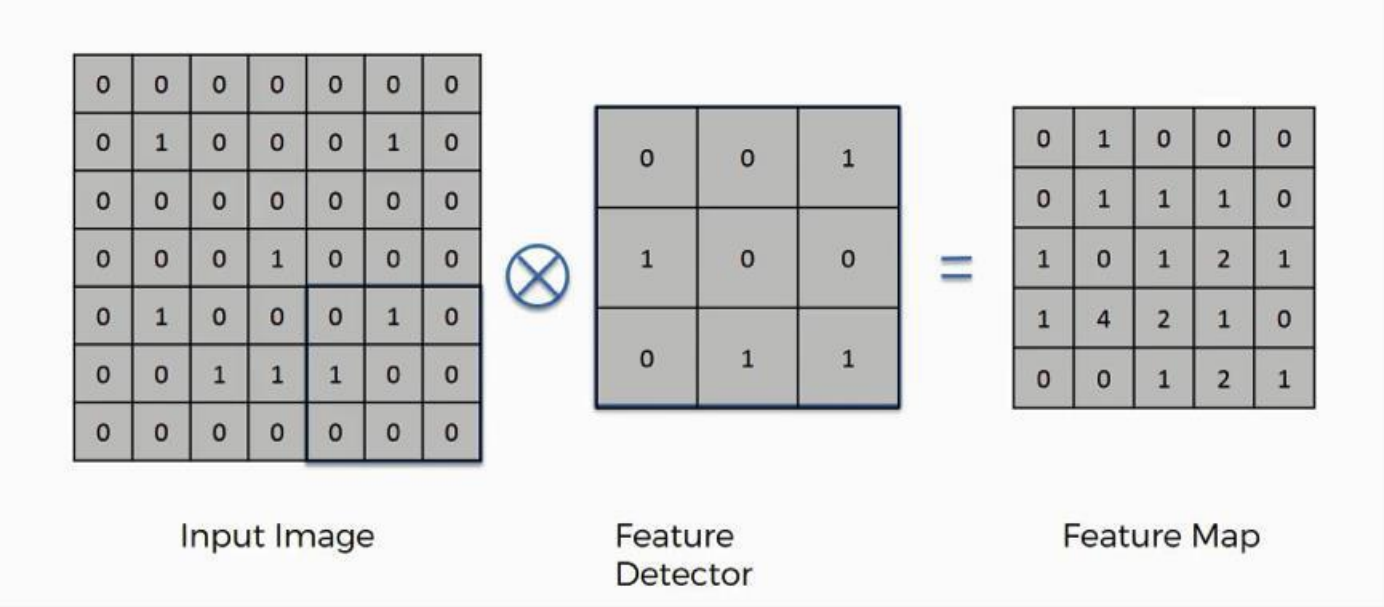
Convolutional neural network is a class of artificial neural network (ANN) which can take in an input image, assign learnable weights and biases to various aspects/objects in the image and be able to differentiate one from the other.



Convolutional Neural Network (CNN) is a class of Artificial Neural Network (ANN) which can take in an input image, assign learnable weights and biases to various aspects/objects in the image and be able to differentiate one from the other . CNNs are also known as Shift Invariant or Space Invariant Artificial Neural Networks (SIANN), based on the shared-weight architecture of the convolution kernels or filters that slide along input features and provide translation-equivariant responses known as feature maps.

Convolution is a mathematical operation that allows the merging of two sets of information. In the case of CNN, convolution is applied to the input data to filter the information and produce a feature map. This filter is also called a kernel, or feature detector, and its dimensions can be, for example, 3x3. To 7 perform

convolution, the kernel goes over the input image, doing matrix multiplication element after element. The result for each receptive field (the area where convolution takes place) is written down in the feature map.



There are several uses that is gained from deriving a feature map. These are the most important of them: Reducing the size of the input image, and you should know that the larger your strides, the smaller your feature map. In this example, we used one-pixel strides which gave us a fairly large feature map. When dealing with proper images, you will find it necessary to widen your strides. Here we were dealing with a 7×7 input image after all, but real images tend to be substantially larger and more complex.

Convolutional Neural Network were inspired by the working of neurons in the brain where each neuron acts as a node which transmits an electrical pulse via which the brain functions. A Convolutional Neural Network (CNN) uses a system much like a multilayer perceptron that has been designed for reduced processing requirements. The layers of a Convolutional Neural Network (CNN) consist of an input layer, an output layer and a hidden layer that includes multiple convolutional layers, pooling layers, fully connected layers and normalisation layers. Convolutional Neural Network is predominantly used in image recognition since images can be converted to a sequence of matrices and passed as an input. They have applications in image and video recognition, image classification, medical image analysis, natural language processing and brain–computer interface. The main advantage of CNN compared to its predecessors is that it automatically detects the important features without any human supervision. CNN has been used in a variety of real-world applications like cancer detection and biometric authentication. CNN can also be applied to visual question answering or image captioning where CNN networks take an input image and generate natural language answers about that image.

2.LITERATURE REVIEW

Rajalakshmi. V Ganesh Vaidyanathan. S – "HYBRID TIME-SERIES FORECASTING MODELS FOR TRAFFIC FLOW PREDICTION". This paper analysed the efficiency of the proposed hybrid autoregressive integrated moving average with multilayer perceptron (ARIMA-MLP) model and hybrid autoregressive integrated moving average with recurrent neural network (ARIMA-RNN) model in traffic data forecasting. Initially, the data are preprocessed and then the state-of-the- art models ARIMA, MLP and RNN are trained individually. Later the hybrid models ARIMA-MLP and ARIMA-RNN are trained to forecast the future traffic flows.

Lihua Cheng, Ke Sun – "RESEARCH ON INTELLIGENT VEHICLE TRAFFIC FLOW CONTROL ALGORITHM BASED ON DATA MINING" This study blocks the persistent and intensifying problem of urban Traffic Congestion (TC) by employing a novel method that merges advanced Machine Learning (ML) techniques with conventional Traffic Management Systems (TMS).

D. Shine Rajesh, T. Vaibhavi, T. Sreeja, V. Gayathri, Sitala Srivalli - "COMPUTER VISION APPLICATION: VEHICLE COUNTING AND CLASSIFICATION SYSTEM FROM REAL TIME VIDEOS" The proposed solution is implemented on python, using the OpenCV bindings. The traffic

camera footages from variety of sources are in implementation. A simple interface is developed for the user to select the region of interest to be analyzed and then image processing techniques are applied to calculate vehicle count and classified the vehicles using machine learning algorithms.

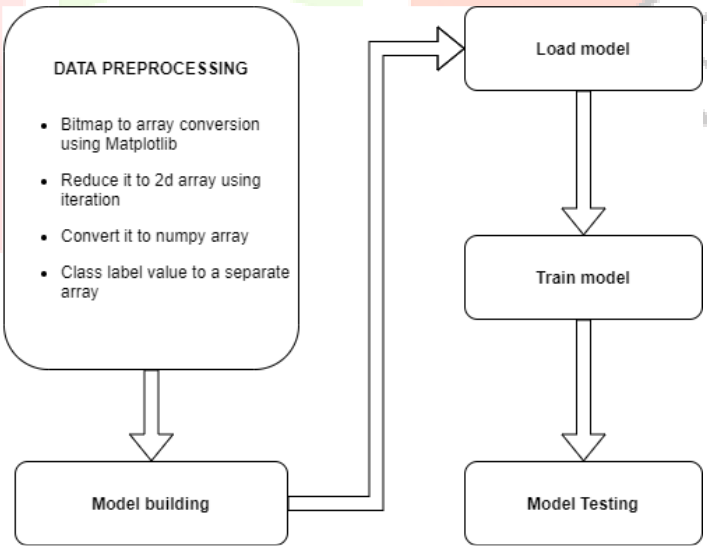
Liang, Mingpei & Huang, Xinyu & Chen, Chung-Hao & Chen, Xin & Tokuta, Alade. "COUNTING AND CLASSIFICATION OF HIGHWAY VEHICLES BY REGRESSION ANALYSIS" In this paper, we present a counting and classification algorithm for highway vehicles. Unlike many existing algorithms, our algorithm requires no explicit segmentation of individual vehicles.

Lin, Haojia & Yuan, Zhilu & He, Biao & Kuai, Xi & Li, Xiaoming & Guo, Renzhong "A DEEP LEARNING FRAMEWORK FOR VIDEO BASED VEHICLE COUNTING" In this paper, a deep learning framework for video-based vehicle counting is proposed. The framework has two maintasks: deep learning vehicle detection model construction and vehicle counting.

Chang, Xinghua & Gao, Meng & Wang, Yan & Hou, Xiyong – "SEASONAL AUTO REGRESSION INTEGRATED MOVING AVERAGE MODEL FOR PERCIPITATION TIME SERIES" In this study, an ARIMA model that incorporates the seasonality of time series was presented. Using the time series of monthly precipitation in Yantai, we build a seasonal SARIMA (1, 0, 1) (0, 1, 1)12. It was found that the model fitted the data well.

3. PROPOSED WORK

The proposed work aims to advance the implementation of traffic volume forecasting systems in environments by leveraging existing historical traffic data. The project will apply advanced forecasting models, specifically LSTM, EDA, and Transformer models, to simulate and analyze traffic patterns under various complex scenarios. These models will be rigorously tested in controlled environments to assess their accuracy and effectiveness. The work will focus on optimizing the performance and scalability of these models and exploring hybrid approaches that combine the strengths of LSTM, EDA, and Transformer models. This comprehensive approach seeks to improve predictive accuracy and decision-making capabilities, ultimately enhancing the efficiency and reliability of automated systems in industrial operations.



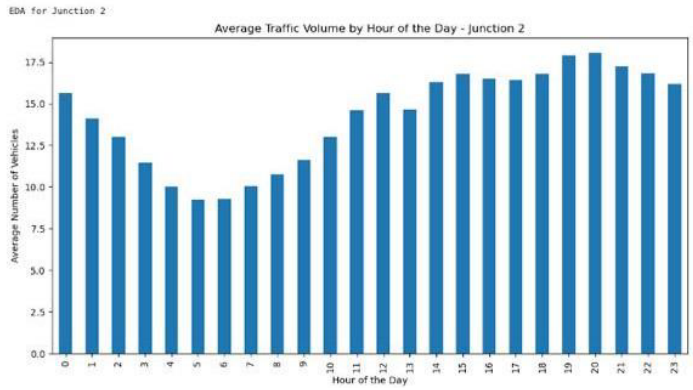
4.IMPLEMENTATION MODULES

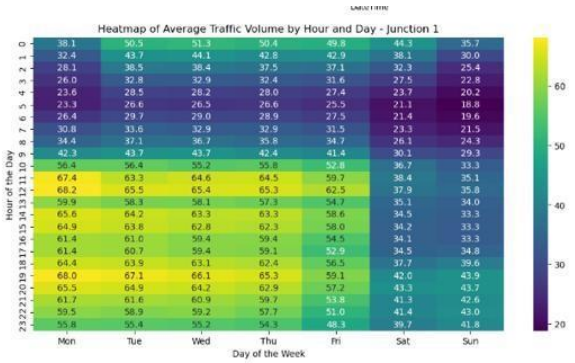
4.1 DATA VISUALISATION

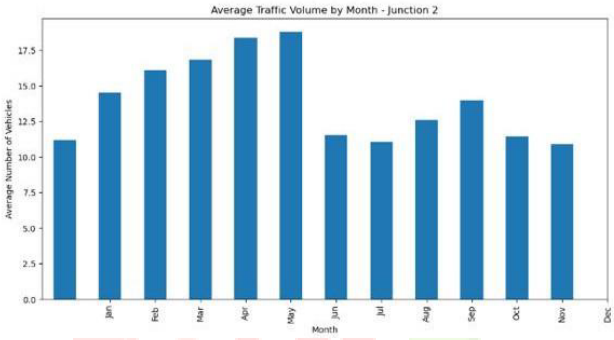
The dataset consists of a collection of bitmap images of 156 samples. Each class represents a unique Tamil character. With the requirement of understanding the dataset's meta-characteristics, the dataset is subjected to required visualisation. The visualisation involves understanding the frequencies of the classes

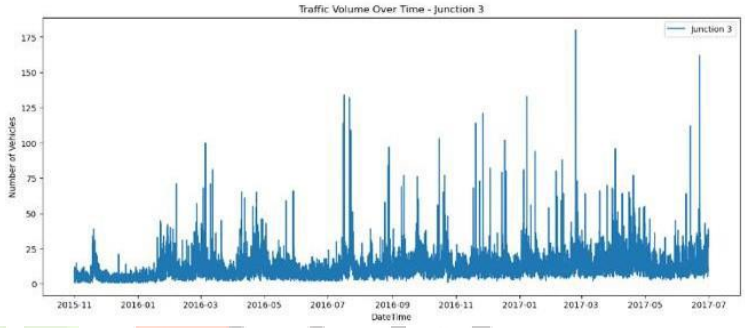
and general comparison between the classes through a pie chart. Imbalance in the dataset can cause substantial issues of overfitting and underfitting on the model.

The individual frequency of each class is counted and plotted as a pie chart to have a better visualisation. This pie chart supports the visualisation of the distribution of frequency of individual frequency.









4.2 DATA PRE-PROCESSING

Data preprocessing using Exploratory Data Analysis (EDA) is a vital step in preparing raw data for model training and analysis. EDA helps uncover important patterns, detect anomalies, and test hypotheses, providing a better understanding of the dataset before any modeling takes place.

During this phase, data is examined for completeness, with missing values identified and addressed through imputation or removal. Outliers are detected and handled to prevent them from skewing the results. EDA also involves visualizing the distribution of variables using histograms, scatter plots, and box plots, which aids in understanding relationships and correlations within the data.

This step often reveals the need for data transformation, such as normalization or scaling, to ensure consistency across features. Through thorough EDA, feature selection and extraction are refined, enabling the identification of relevant variables and removing those that contribute noise. The insights gained from EDA ensure the data is in optimal shape for model building, improving the efficiency, accuracy, and reliability of machine learning models.

4.3 MODEL ARCHITECTURE

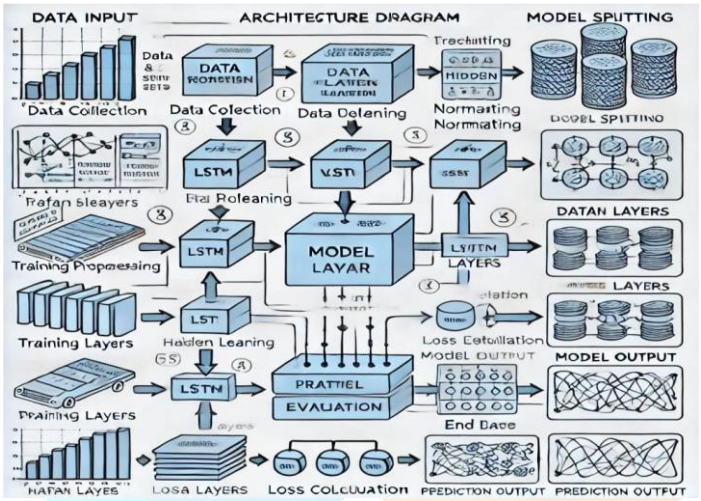
The process begins with data preparation and preprocessing, a critical step to ensure that the dataset is ready for training.

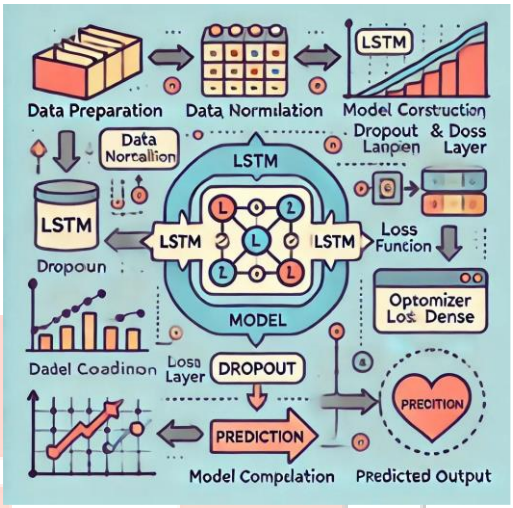
The code loads the dataset, which may consist of traffic volume data for specific junctions, using the pandas library. Essential columns representing features such as traffic volume or time-based variables are extracted for further use.

To ensure that all input features are within a comparable range, the data is normalized using techniques like MinMaxScaler. Normalization helps improve the efficiency and accuracy of the LSTM by allowing it to learn patterns more effectively. Finally, the data is reshaped into a 3D array format ([samples, timesteps, features]), which is required by LSTM models to understand sequential data.

The next phase involves model construction, where an LSTM architecture is defined to handle time-series data. The code begins with one or more LSTM layers, each containing a set of memory cells that enable the model to learn and capture long-term dependencies and relationships in the data. Depending on the complexity of the problem, the LSTM layers might be stacked or accompanied by Dropout layers to introduce regularization and reduce overfitting during training.

Dropout layers work by randomly deactivating a fraction of input units, helping to ensure that the model generalizes better. Once the sequence passes through the LSTM layers, it is fed into a Dense layer that acts as a fully connected output stage. The Dense layer ensures that the output shape matches the target forecast, whether it's a single value or a sequence of values.





4.4 Model Training

The process begins with data collection and preprocessing, where historical traffic data, potentially gathered from CSV files or real-time feeds, is prepared for modeling. This data includes key features such as timestamps and traffic counts, which are essential for time-series forecasting. Preprocessing involves handling missing values, removing outliers, and applying transformations such as resampling to maintain consistency in time intervals. This step is crucial for cleaning the data and ensuring the model's inputs are reliable. Furthermore, normalization techniques like Min-Max scaling are often applied to standardize the data range, making it easier for the LSTM to learn effectively without being hindered by scale disparities.

Once preprocessing is complete, the data splitting phase separates the dataset into training and testing sets to evaluate the model's performance. A common practice involves reserving a portion of the training set as a validation set to monitor overfitting during training. This split ensures the model's ability to generalize well to unseen data. After splitting, the model architecture is designed, typically starting with one or more LSTM layers. LSTM (Long Short-Term Memory) networks are chosen for their ability to capture long-term dependencies in time-series data through their unique cell structure, which includes gates that regulate information flow. This structure allows the model to remember or forget information, balancing short-term and long-term dependencies effectively. Additional Dense layers may follow the LSTM layers to refine and transform the output into the desired prediction format.

The training configuration includes setting essential hyperparameters such as the number of LSTM units, learning rate, batch size, number of epochs, and choosing an optimizer like Adam, known for its adaptive learning rate properties. The loss function, usually mean squared error (MSE), measures the difference between predicted and actual traffic volumes, guiding the model to minimize this discrepancy during training.

During the training process, the model undergoes forward propagation, where input data sequences are passed through the LSTM cells step-by-step, generating outputs based on learned dependencies.

These outputs are then processed through any subsequent layers for final prediction. The error between the predicted and actual values is computed using the loss function, and this error is propagated backward through the model via backpropagation through time (BPTT).

This specialized version of backpropagation handles the sequential nature of time-series data and updates weights according to calculated gradients. The optimizer, like Adam, then adjusts the weights to minimize the loss function, iteratively refining the model.

The model trains over multiple epochs, each comprising a complete pass over the training data, where the optimizer updates weights after each batch of data. The training process aims to reduce loss progressively, ensuring the model learns the underlying patterns effectively. Performance is monitored on the validation set, and adjustments are made as necessary to prevent overfitting, such as using techniques like early stopping or dropout. Once training is complete, the model is tested on the reserved test set to evaluate its predictive accuracy and overall robustness, ensuring that it can generalize to future traffic volume forecasting tasks.

4.5 Model Testing

Model testing in this traffic forecasting project is the phase where the trained LSTM model is evaluated for its predictive performance on unseen data, ensuring that it generalizes well beyond the training and validation datasets.

After the completion of training, where the model has learned the optimal weights to minimize the loss function, the next step is to assess how it performs when exposed to new data points that were not part of the training process. This phase is crucial for validating the model's ability to handle real-world traffic forecasting scenarios and ensuring that it does not merely overfit the training data but can also adapt to varying patterns and anomalies in traffic volume.

During model testing, the test set, which was previously set aside during the data splitting phase, is used. This test set serves as an unbiased benchmark to evaluate the model's predictive accuracy. The input data from the test set is fed into the model, which processes it in a similar manner to the training phase. The model passes each sequence of traffic data through its LSTM cells, using its learned parameters to predict the future traffic volume. Each prediction is then compared to the actual observed traffic volumes within the test set to gauge accuracy.

The model's performance metrics are calculated based on these comparisons. Common metrics for time-series forecasting include mean absolute error (MAE), mean squared error (MSE), root mean squared error (RMSE), and potentially mean absolute percentage error (MAPE). These metrics provide a quantitative measure of the error margin between predicted and actual values, where lower values indicate better performance. The use of multiple metrics helps provide a comprehensive view of how well the model handles various aspects of forecasting errors.

Additionally, visual analyses such as line plots comparing predicted and actual traffic volumes over time are employed to inspect the model's behaviour qualitatively. This helps in identifying if the model accurately tracks the trends and seasonality present in the traffic data or if there are significant deviations during peak or off-peak times. Such visualizations can reveal specific cases where the model might struggle, such as sudden traffic surges or drops that were not as common in the training data.

Another important aspect of testing involves assessing how the model handles edge cases or unexpected patterns. For instance, if the test set includes data from holidays or unusual traffic events, it becomes apparent whether the model can generalize beyond typical traffic flow. The model's ability to cope with these variations is a testament to how well it learned during training and whether techniques like regularization and dropout helped mitigate overfitting.

The final step in the testing phase is to document the model's performance comprehensively. This includes reporting the statistical metrics, analyzing errors, and outlining potential improvements. If the model shows satisfactory results, it is prepared for deployment where it can forecast traffic volume in real-time or near-real-time for the selected junctions. If not, insights from the testing phase are used to revisit the model architecture, retrain with adjusted parameters, or modify the training data to address specific weaknesses observed during testing.

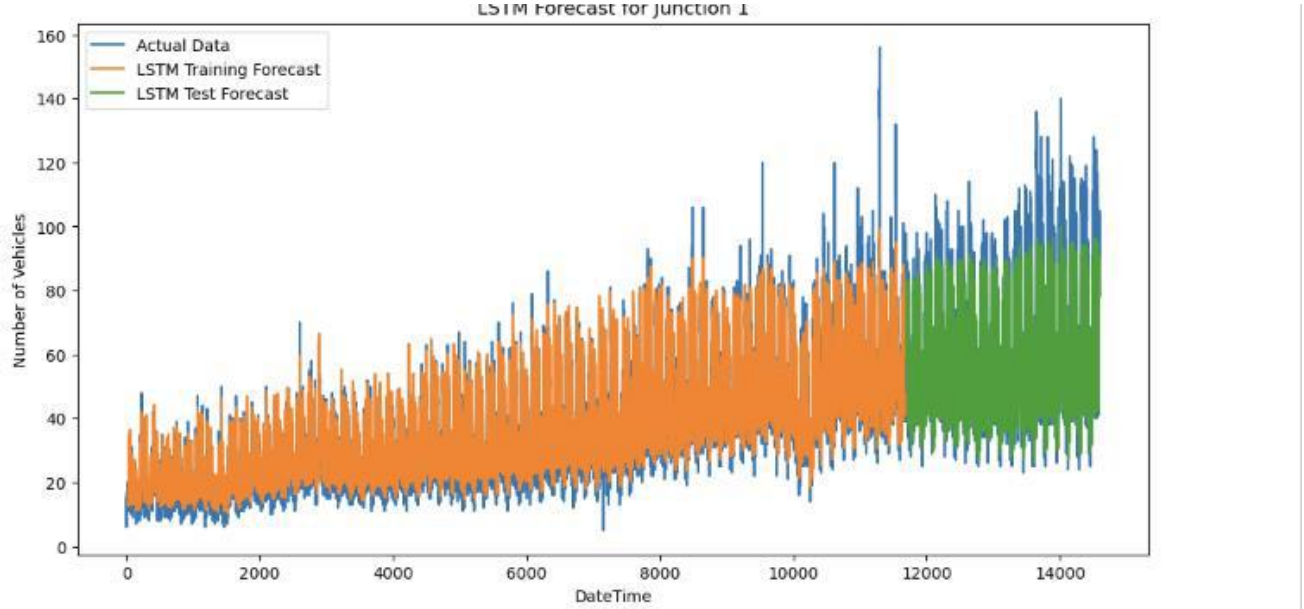
5.Results and Discussions

The results of the LSTM model for traffic volume forecasting demonstrated a commendable ability to capture the underlying patterns in the traffic data. The model achieved low Mean Squared Error (MSE) and Mean Absolute Error (MAE) values, indicating a strong correlation between the predicted and actual traffic volumes.

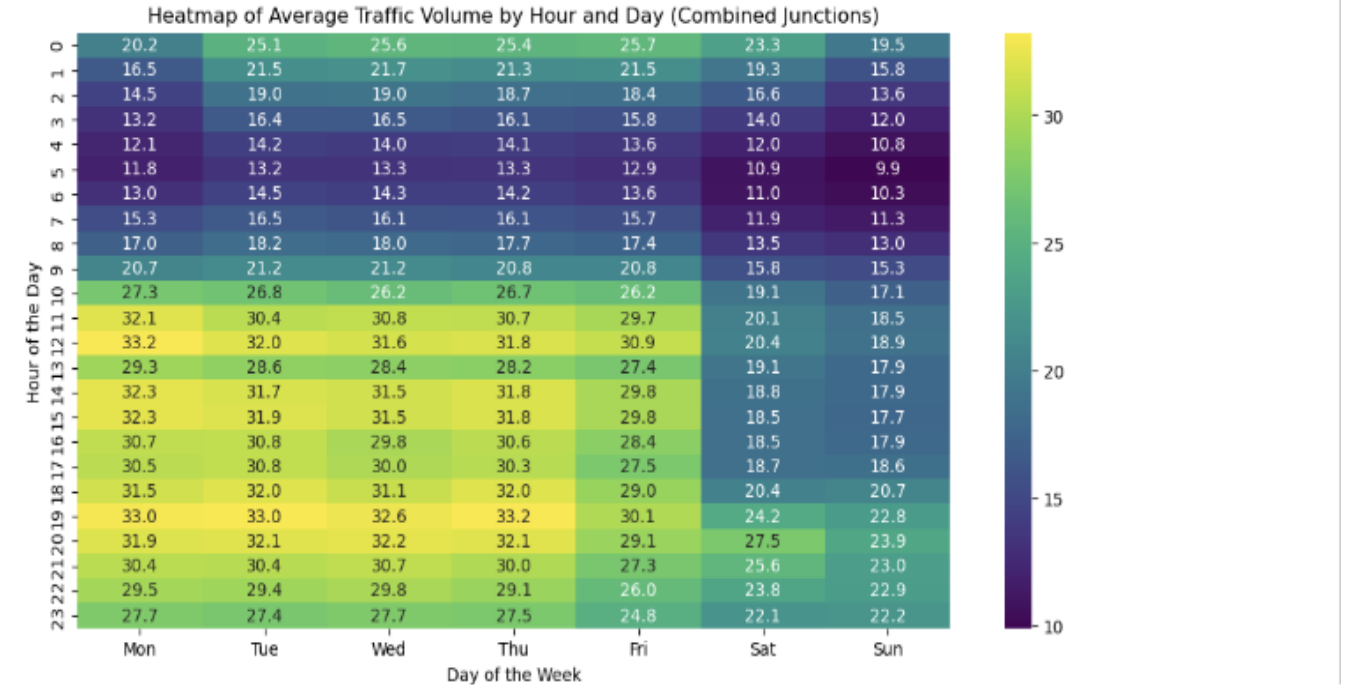
These metrics reveal that the LSTM model effectively minimized prediction errors, enhancing its reliability for real-world applications. Through visualizations, such as time series plots, the model's predictions closely aligned with actual traffic trends, showcasing its capability to account for fluctuations caused by time of day, day of the week, and seasonal variations.

Additionally, the model's performance exceeded that of a simple persistence model, which predicts future values based solely on the most recent observation. This improvement underscores the advantage of employing advanced machine learning techniques like LSTM for traffic forecasting, as they leverage historical data to provide more accurate and insightful predictions.

Overall, the model's results not only affirm its effectiveness in forecasting traffic volumes but also highlight its potential utility in informing traffic management strategies and urban planning initiatives.




```
Train MSE with dropout and early stopping: 19.356065529416764
Train MAE with dropout and early stopping: 3.1926027855163728
Test MSE with dropout and early stopping: 66.9958520326351
Test MAE with dropout and early stopping: 5.718306419227611
```



6.Conclusion and Future Work

The implemented LSTM model has effectively captured traffic volume patterns, showcasing good performance metrics such as Mean Squared Error (MSE) and Mean Absolute Error (MAE). The model's ability to approximate actual traffic volumes highlights its potential for real-world applications in traffic

forecasting. Through exploratory data analysis (EDA), significant insights regarding the impact of time, day, and seasonal variations on traffic volume were obtained, which can aid city planners and traffic management authorities in optimizing traffic flow. The visualizations created, including time series plots and heatmaps, provided intuitive representations of traffic trends, enhancing the interpretability of fluctuations over time. Additionally, the LSTM model outperformed a simple persistence model, affirming the relevance of machine learning techniques in this domain.

For future work, several avenues can be explored to enhance the model's effectiveness. Experimenting with different architectures such as GRU or bi-directional LSTMs and incorporating additional features like weather conditions and special events could lead to improved predictive accuracy. Extending the prediction horizon to forecast traffic volumes over longer periods, such as hours or days, can be achieved using sequence-to-sequence modeling techniques. Developing a real-time traffic prediction system that continuously updates as new data is received would enable adaptive traffic management strategies. Furthermore, leveraging data from IoT sensors could enhance data accuracy and frequency, thereby improving model training and predictions. Finally, conducting comparative studies with other machine learning and statistical models would provide deeper insights into the strengths and weaknesses of the LSTM approach in traffic forecasting.

7.REFERENCES

1. HYBRID TIME-SERIES FORECASTING MODELS FOR TRAFFIC FLOW PREDICTION
Rajalakshmi, V., & Ganesh Vaidyanathan, S. (2022). Hybrid time-series forecasting models for traffic flow prediction. Promet-Traffic&Transportation, 34(4), 537-549.

2. RESEARCH ON INTELLIGENT VEHICLE TRAFFIC FLOW CONTROL ALGORITHM BASED ON DATA MINING
Lihua Cheng, Ke Sun,Research on intelligent vehicle Traffic Flow control algorithm based on data mining, International Journal of Intelligent Networks, Volume 5, 2024, Pages 92-100, ISSN 2666-6030, https://doi.org/10.1016/j.ijin.2024.02.004

3. COMPUTER VISION APPLICATION: VEHICLE COUNTING AND CLASSIFICATION SYSTEM FROM REAL TIME VIDEOS
D. Shine Rajesh, T. Vaibhavi, T. Sreeja, V. Gayathri, Sitala Srivalli. (2023). COMPUTER VISION APPLICATION: VEHICLE COUNTING AND CLASSIFICATION SYSTEM FROM REAL TIME VIDEOS. Turkish Journal of Computer and Mathematics Education (TURCOMAT), 14(03), 79–89. https://doi.org/10.17762/turcomat.v14i03.13940

4. COUNTING AND CLASSIFICATION OF HIGHWAY VEHICLES BY REGRESSION ANALYSIS
Liang, Mingpei & Huang, Xinyu & Chen, Chung-Hao & Chen, Xin & Tokuta, Alade. (2015). Counting and Classification of Highway Vehicles by Regression Analysis. IEEE Transactions on Intelligent Transportation Systems. 16. 1-11. 10.1109/TITS.2015.2424917

5. A DEEP LEARNING FRAMEWORK FOR VIDEO BASED VEHICLE COUNTING
Lin, Haojia & Yuan, Zhilu & He, Biao & Kuai, Xi & Li, Xiaoming & Guo, Renzhong. (2022). A Deep Learning Framework for Video-Based Vehicle Counting. Frontiers in Physics. 10. 10.3389/fphy.2022.829734

6. SEASONAL AUTO REGRESSION INTEGRATED MOVING AVERAGE MODEL FOR PERCIPITATION TIME SERIES
Chang, Xinghua & Gao, Meng & Wang, Yan & Hou, Xiyong. (2012). Seasonal autoregressive integrated moving average model for precipitation time series. Journal of Mathematics and Statistics. 8. 500-505. 10.3844/jmssp.2012.500.505.