Machine Learning Lab

Project Report

on

TRAFFIC FLOW PREDICTION using GRAPH NEURAL NETWORKS

Submitted by

Anurag Patel, 121CS0201

Palak Deb Patra, 121CS0202

Gavyn Paul, 121CS0203

Department of Computer Science and Engineering

NATIONAL INSTITUTE OF TECHNOLOGY ROURKELA

1. Introduction

1.1 Introduction

Urban transportation systems face challenges such as congestion, environmental impacts, and the need for efficient travel. Traffic flow prediction helps alleviate these issues by providing data-driven insights into expected traffic patterns, allowing city planners to make informed decisions. By accurately predicting traffic flow across a network of sensors, cities can improve road safety, reduce travel times, and mitigate congestion. This project employs Spatio-Temporal Graph Convolutional Networks (STGCN) to predict traffic flow patterns on a city-wide scale, leveraging both spatial and temporal dependencies inherent in traffic data.

1.2 Motivation

With the rising importance of smart cities and efficient urban infrastructure, traffic flow prediction has become crucial. Accurate predictions allow authorities to manage congestion, optimize infrastructure, and improve commuter experience. The STGCN model's unique capacity to model complex spatial-temporal data positions it as an effective solution for traffic flow prediction tasks.

1.3 Problem Statement

The objective of this project is to design and implement a predictive model that accurately forecasts traffic flow across a network of sensors in an urban environment. The model is built to learn both spatial and temporal dependencies, enabling it to forecast future traffic levels and provide insights for effective urban mobility management.

..

2. Literature Review

Traffic flow prediction research has evolved from traditional time-series methods to machine learning approaches that address spatial and temporal complexities. Current methods in spatio-temporal modeling include the following:

- **3D Convolutional Neural Networks (3D CNNs)**:
  These models extend traditional 2D convolutions by adding a temporal dimension, allowing the network to learn spatial and temporal features simultaneously. However, 3D CNNs are computationally intensive and require significant labeled data.

- **Graph Convolutional Networks (GCNs)**:
  GCNs model spatial dependencies by applying convolutions over graph-structured data, where nodes represent locations (such as traffic sensors) and edges represent spatial relationships. GCNs are effective for spatially correlated data but need to be combined with temporal modeling for time-series prediction tasks.

- **Spatio-Temporal Graph Convolutional Networks (STGCNs)**:
  STGCNs combine GCNs with temporal convolution layers, making them ideal for spatio-temporal tasks such as traffic flow prediction. STGCNs capture spatial relationships between sensors while simultaneously learning temporal patterns in traffic data.

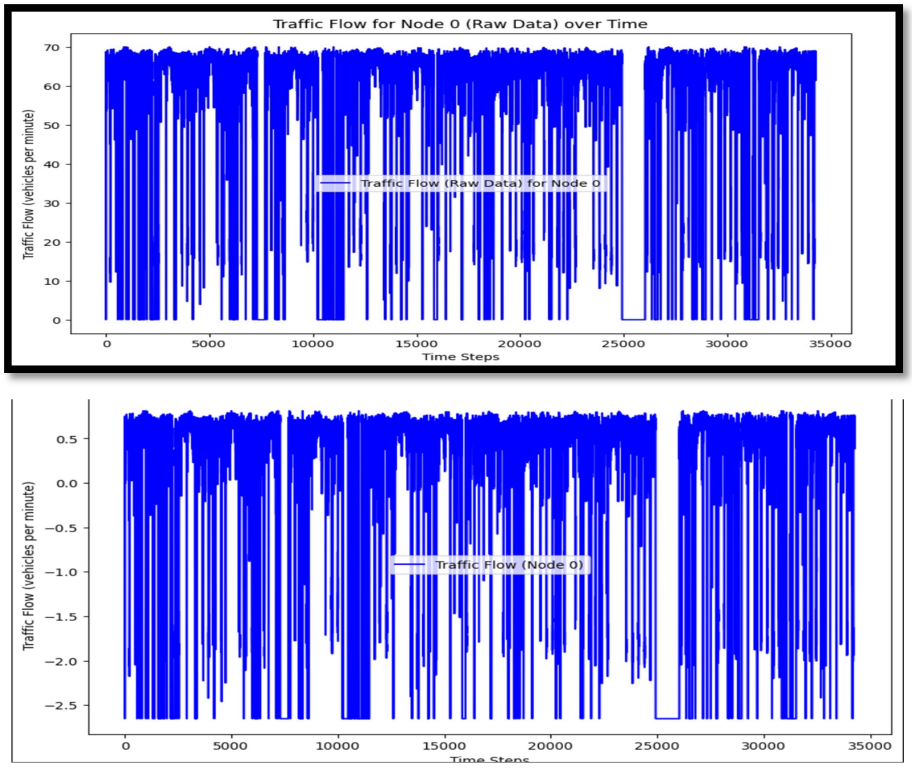
3. Proposed Methodology

This project applies the STGCN model for traffic flow prediction, dividing the methodology into data preprocessing, model architecture, and training and evaluation stages.

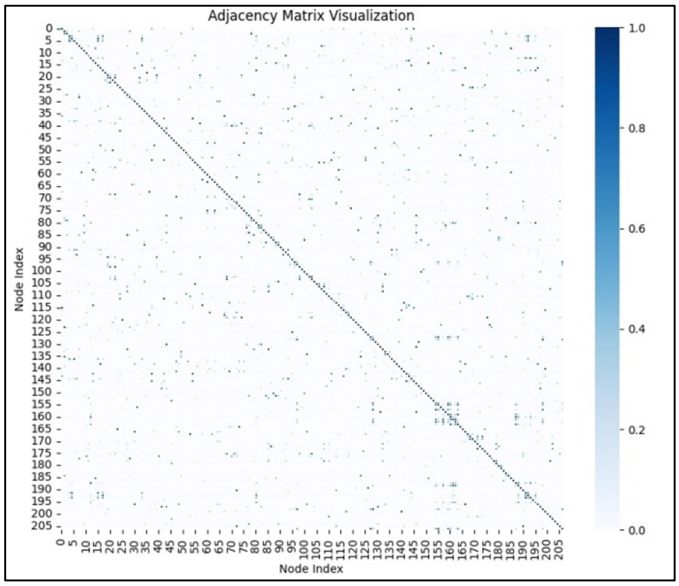
3.1 Data Preprocessing

The METR-LA dataset is used for this project, containing traffic data from 207 sensors distributed across Los Angeles. Each sensor records traffic flow and occupancy over time, capturing both spatial and temporal characteristics of urban traffic.

1. Normalization: The Z-score normalization method is used to scale traffic data, ensuring that input features have zero mean and unit variance, which improves model convergence.



2. Adjacency Matrix Normalization: Degree normalization is applied to the adjacency matrix which captures the spatial relationships between sensors. This normalization helps ensure stable gradient flow during training.



3.2 Model Architecture

Input:

- Graph Structure (Adjacency Matrix): Describes the spatial relationships between sensors (nodes).
- Temporal Features (Sensor Data): Includes data like traffic speed or occupancy for each sensor at different time steps.

Layers:

- Temporal Convolutional Layers (TimeBlock): These layers capture temporal dependencies in the sensor data (traffic flow) over multiple time steps.
- Graph Convolutional Layers (STGCNBlock): These layers apply graph convolutions to the temporal outputs, integrating spatial information between sensors based on the adjacency matrix.
- Batch Normalization: Helps stabilize training by normalizing the output of the convolutional layers.
- Fully Connected Layer: The output is passed through a fully connected layer to map the final features to traffic flow predictions for the required number of timesteps.

Output:

- Predicted Traffic Flow: For each sensor and for each of the predicted time steps, the model outputs traffic flow predictions, which represent the future traffic conditions (e.g., traffic speed or occupancy).

3.3 Training and Evaluation

The model was trained over 100 epochs with a batch size of 8. The Mean Squared Error (MSE) loss function was used to optimize model parameters.

- **Loss Function:** Mean Squared Error (MSE)
- **Optimizer**: Adam Optimizer

4. Results and Discussion

4.1 Experimental Setup

The dataset was divided into training (60%), validation (20%), and testing (20%) sets. A sliding window approach was used to create training and testing samples by shifting a fixed-size window across the time dimension.

4.2 Evaluation Metrics

1. **Training Loss**: The MSE was recorded over epochs to monitor model convergence.
2. **Validation Metrics**: Validation MSE and Mean Absolute Error (MAE) were calculated for each epoch.

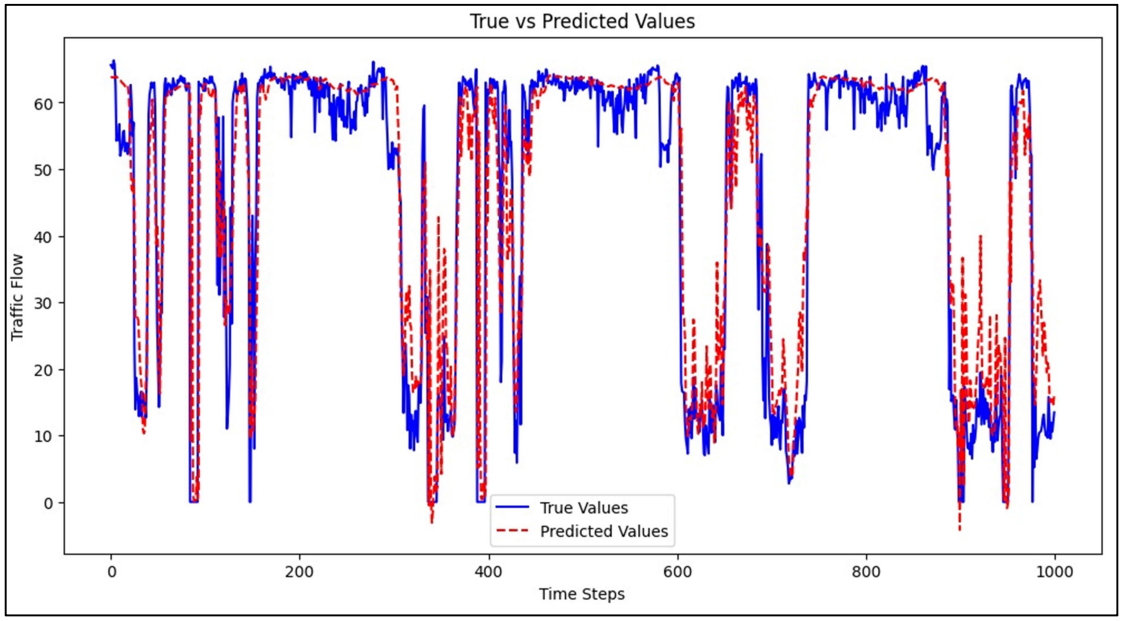
4.3 Model Performance

- **Loss Convergence**: The training and validation losses consistently decreased, indicating that the model was learning effectively.
- **Prediction Accuracy**: Visual comparison between predicted and actual traffic flow values showed a strong match, demonstrating the STGCN model's effectiveness in capturing both spatial and temporal patterns.

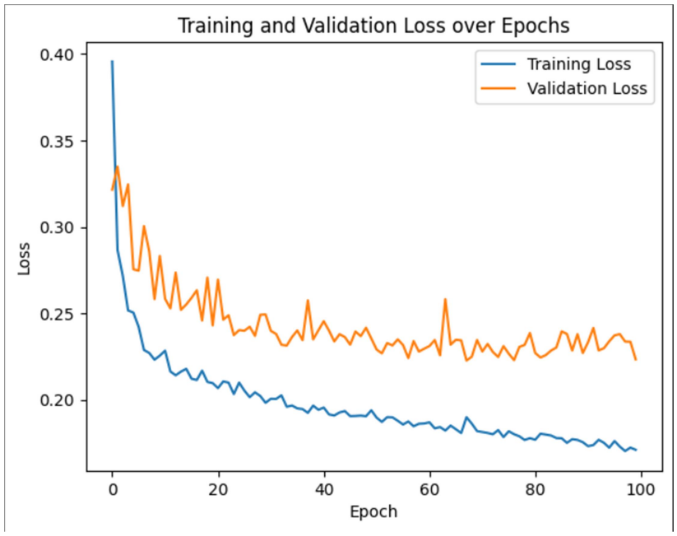
4.4 Visualization

The following plots were generated to evaluate the model:

- **True vs. Predicted Traffic Flow**: Plots of actual and predicted traffic flow values for selected nodes, showing the model's ability to approximate real values.

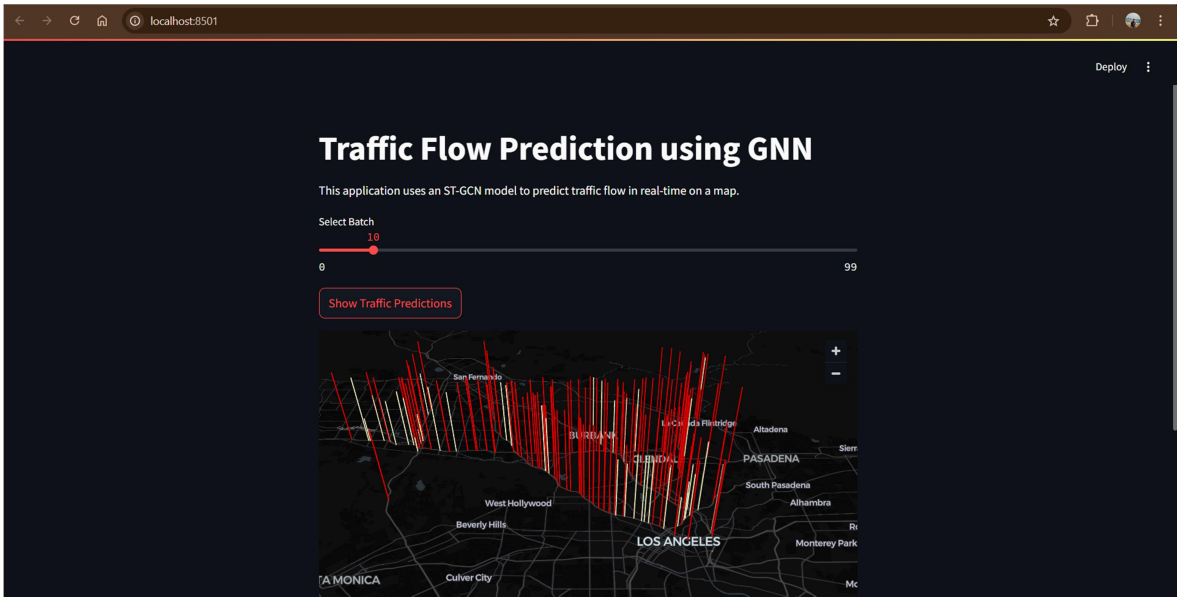


- **Loss Curves**: Training and validation loss plots over epochs, demonstrating loss convergence and model stability.



4.5 Graphical User Interface (GUI)

- **A** Streamlit-based GUI **was implemented to visualize real-time traffic flow predictions on a map. The key features of the GUI include:**
- **Traffic Flow Visualization:**
  The map shows predicted traffic flow at different sensors using color-coded markers. Each sensor's prediction is represented as a tower, with the height corresponding to the traffic flow value and the color reflecting congestion levels (from low to high).
- **Interactive Map:**
  The map is interactive, allowing users to hover over sensor points for detailed information, such as node ID and the predicted traffic flow for each sensor.
- **Prediction Display:**
  Users can load and view traffic flow predictions for different batches of data, with the ability to navigate through different time steps using sliders.
- This GUI provides an intuitive interface to explore the model's predictions and monitor traffic conditions in real-time.



5. Conclusion

1. **Model Effectiveness**:

- The STGCN model demonstrates high accuracy in predicting traffic flow, specifically on the METR-LA dataset.
- By combining spatial and temporal graph convolution layers, the model captures complex patterns in urban traffic data, making it well-suited for analyzing dynamic city environments.

2. **Spatio-Temporal Dependency Capture**:

- STGCN effectively models dependencies not only between different sensor locations (spatial) but also over time (temporal), enabling it to predict how traffic conditions evolve across different areas and time intervals.
- This dual-dependency modeling is essential for understanding and forecasting urban traffic flows, where conditions at one location often influence neighboring areas.

3. **Applicability to Urban Mobility**:

- The model's ability to forecast traffic patterns with accuracy has practical applications for urban mobility management. These predictions can help with traffic congestion mitigation, route optimization, and infrastructure planning in smart cities.

4. **Future Improvements**:

- Incorporating Additional Data: Future work could enhance model performance by integrating other relevant data sources, such as weather conditions, special events, or road incident data, which can significantly impact traffic flow.
- Real-Time Monitoring: Extending the model for real-time applications would enable live traffic monitoring, where predictions could assist in dynamic traffic management systems, such as adjusting traffic signals or rerouting traffic during peak times.
- Scalability and Transferability: Adapting the model for different cities or regions with unique traffic behaviors could broaden its utility, allowing it to support diverse urban areas with distinct transportation patterns.

6. References

- https://www.kaggle.com/code/xiaohualu/mapvisualization-metrla-pydeck?select=graph_sensor_locations.csv
- https://github.com/FelixOpolka/STGCN-PyTorch?tab=readme-ov-file
- https://arxiv.org/abs/2101.11174
- https://github.com/aprbw/traffic_prediction