Project Report

Mohammad Ashif (ma23bb)
Rahul Debnath (rd23bc)

Department Of Computer Science
Florida State University

Data Science For Smart Cities
Dec 11th 2024

SynMob: Creating High-Fidelity Synthetic GPS Trajectory Dataset for Urban Mobility Analysis

Mohammad Ashif*
ma23bb@fsu.edu
Florida State University

Rahul Debnath
rd23bc@fsu.edu
Florida State University

Abstract

Urban mobility analysis is integral to optimizing transportation systems and improving urban infrastructure planning. However, real-world GPS trajectory data faces challenges such as privacy concerns, limited accessibility, and data inconsistencies. Building upon the SynMob framework, this study presents a novel synthetic trajectory dataset generated from vehicle mobility data for Los Angeles. The dataset was created using advanced diffusion models and incorporates key steps, including data cleaning, preprocessing, route prediction, and congestion point identification. The synthetic dataset aligns closely with real-world data in terms of geospatial accuracy, temporal consistency, and statistical similarity, as demonstrated by detailed evaluations such as heatmaps, clustering, and statistical analyses of trip lengths and speeds. Use cases, including the identification of 60 congestion points and route prediction between random points, validate the practical applicability of the synthetic dataset for urban planning and traffic management. By addressing data privacy and scalability challenges, this work offers a high-fidelity synthetic dataset that supports enhanced urban mobility research and provides a valuable resource for traffic simulation and optimization tasks.

CCS Concepts

• **Applied computing** → **Transportation**; • **Information systems** → **Geographic information systems**; • **Computing methodologies** → *Simulation*; • **Security and privacy** → *Privacy protections.*

Keywords

synthetic GPS data, urban mobility analysis, spatial-temporal distribution, diffusion model, data privacy, traffic simulation, geographic information systems, machine learning for transportation

ACM Reference Format:
Mohammad Ashif and Rahul Debnath. 2024. SynMob: Creating High-Fidelity Synthetic GPS Trajectory Dataset for Urban Mobility Analysis. In *Proceedings of the ACM Symposium on Spatial User Interaction (SIGSPATIAL '24), October 21–23, 2024, Seattle, WA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/1234567.1234568

1 Introduction

SynMob GitHub Link for Project Code

SIGSPATIAL '24, October 21–23, 2024, Seattle, WA

ACM ISBN 978-1-4503-9999-X/24/10
https://doi.org/10.1145/1234567.1234568

1.1 Background

Urban mobility research is critical for addressing the growing challenges of urbanization, such as traffic congestion, inefficient public transit, and increasing travel demand. At the core of such studies lies GPS trajectory data, which provides granular insights into spatial-temporal movement patterns of vehicles and individuals. This data underpins a range of applications, from real-time traffic management to long-term infrastructure planning.

Despite its potential, the utility of real-world GPS trajectory data is constrained by several challenges:

- Privacy Risks: GPS data often contains personally identifiable information (PII), raising concerns about individual privacy and data security.
- Restricted Access: Many high-quality GPS datasets are proprietary, limiting access for research purposes. Even publicly available datasets often lack comprehensive geographic or temporal coverage.
- Data Quality: Missing values, noise, and inconsistent formats in real-world datasets can complicate analysis and limit the reliability of derived insights.

These limitations necessitate alternative approaches for generating representative trajectory data. Synthetic datasets, which emulate the spatial-temporal patterns of real-world data without exposing sensitive information, have emerged as a promising solution. By preserving statistical and geographic fidelity, synthetic data allows researchers to conduct robust urban mobility studies while ensuring privacy compliance.

Frameworks such as SynMob have demonstrated the feasibility of generating high-fidelity synthetic GPS data using advanced machine learning models, including diffusion processes. SynMob provides a foundation for replicating real-world traffic patterns, offering a scalable and privacy-preserving alternative to traditional data sources.

This study extends the SynMob framework, introducing several innovations to adapt its methodology to the unique characteristics of Los Angeles traffic data. The key contributions of this work include:

(1) **Los Angeles-Specific Dataset:**
   - A synthetic GPS trajectory dataset tailored to Los Angeles, capturing the distinctive traffic flow patterns of this metropolitan area.
   - Localized calibration to account for high congestion zones and diverse route behaviors.
(2) **Advanced Preprocessing Pipelines:**
   - A streamlined data preprocessing pipeline that incorporates data cleaning, attribute extraction (e.g., travel times, distances), and optimized formatting for efficient downstream processing.
(3) **Comprehensive Evaluation:**

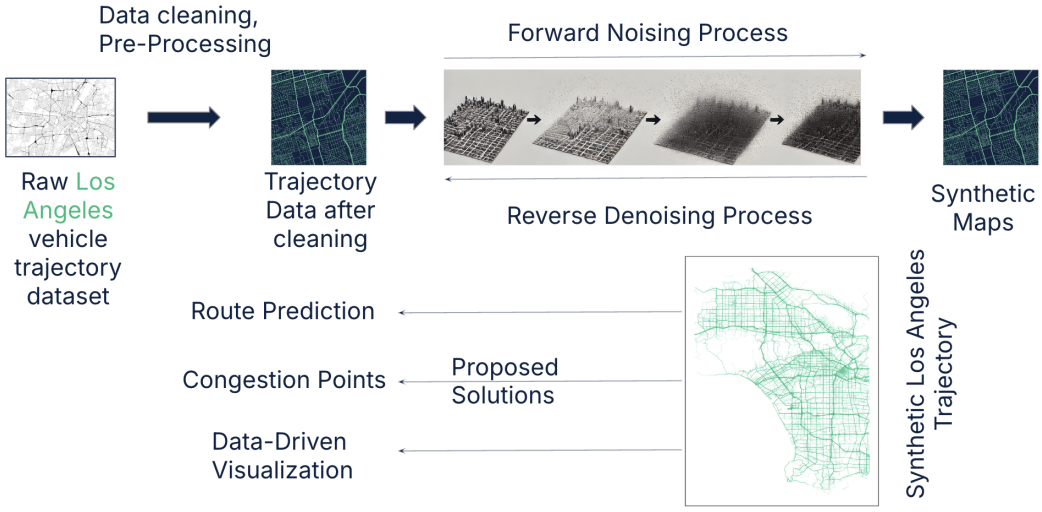

Figure 1: The illustration of using synthesized trajectory dataset for urban mobility analysis.

- Validation of the synthetic dataset using statistical metrics, such as the Kolmogorov-Smirnov test, and geospatial overlays to ensure fidelity.
- Demonstrations of practical applications, including congestion hotspot detection and route prediction, to showcase real-world utility.

This work contributes to the broader goal of enabling scalable, privacy-preserving urban mobility research, demonstrating that synthetic trajectory datasets can emulate real-world traffic dynamics effectively.

2 Related Work

2.1 Existing GPS Datasets

Publicly available GPS trajectory datasets have been pivotal in advancing urban mobility research. Some of the most notable datasets include:

- GeoLife: A GPS trajectory dataset collected from users in Beijing, comprising over 17,000 trajectories. It has been widely used in applications such as activity recognition and route prediction.
- T-Drive: A large-scale GPS dataset capturing the trajectories of over 10,000 taxis in Beijing, offering insights into urban traffic flow and congestion patterns.
- Porto Taxi Dataset: A GPS dataset collected from taxis in Porto, Portugal, used extensively for studying urban transport dynamics and ride-sharing optimization.

While these datasets have facilitated significant advancements, they are often limited by their geographic scope, temporal coverage, and privacy constraints. Furthermore, proprietary datasets are not always accessible to researchers, hindering reproducibility and scalability.

2.2 Synthetic Data Generation

The limitations of real-world GPS datasets have led to the emergence of synthetic data generation methods, which aim to emulate the characteristics of real-world data without exposing sensitive information. Synthetic data has gained traction for its scalability, privacy-preserving properties, and potential to generalize across diverse geographic and temporal contexts.

2.2.1 *Key Approaches to Synthetic Data Generation.*

(1) Agent-Based Models: Simulate the movement of individual agents (e.g., vehicles or pedestrians) based on predefined rules and environmental conditions. While effective, these models require significant manual calibration and may lack scalability.
(2) Generative Models: Machine learning-based approaches, such as Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs), generate synthetic trajectories by learning the underlying distribution of real-world data.
(3) Diffusion Models: Recently, frameworks like SynMob have introduced diffusion-based methods for trajectory synthesis, achieving high fidelity in replicating spatial-temporal patterns. Diffusion models add controlled noise to real trajectories and then iteratively denoise the data to generate synthetic counterparts.

2.3 Enhancements Introduced

The SynMob framework provides a benchmark for synthetic data generation, demonstrating the feasibility of replicating real-world traffic patterns with high fidelity. This study builds on SynMob by introducing several enhancements:

(1) Localized Adaptation:
   - The original SynMob framework was calibrated for general datasets. This study customizes the methodology to suit Los Angeles traffic data, incorporating unique congestion patterns and route behaviors.
(2) Robust Preprocessing:
   - Enhanced data cleaning techniques are applied to filter noise and outliers from the original data.
   - Attribute extraction pipelines are refined to focus on key mobility metrics, such as travel times, distances, and departure intervals.

(3) Comprehensive Evaluation:
- The generated dataset is validated using rigorous statistical tests and geospatial analyses.
- Practical use-case demonstrations, such as congestion point detection and route prediction, are included to highlight the real-world applicability of the dataset.

By addressing the limitations of existing synthetic data generation frameworks, this study provides a robust and scalable approach to urban mobility research, with a specific focus on the Los Angeles metropolitan area.

3 Data Analysis

DataDryad Dataset Link

The analysis presented herein utilizes a synthetic vehicle trajectory dataset tailored for the metropolitan area of Los Angeles. Created using the Data-Driven Trajectory Generator (DDTG) at the Integrated Media Systems Center, USC, this dataset was designed to simulate realistic urban vehicular movements while ensuring privacy and comprehensive spatio-temporal coverage.

3.0.1 Dataset Structure. The dataset encompasses 1.5 million trajectories over the first two weeks of December 2019, providing a rich, anonymized canvas for urban studies. Each entry in the dataset contains the following attributes:

- **UUID** (User Unique ID): A pseudonymized identifier for user privacy.
- **TUID** (Trajectory Unique ID): Unique identifier for each trajectory.
- **Latitude and Longitude**: Geospatial coordinates at various timestamps.
- **Accuracy**: GPS accuracy of the data points.
- **Timestamp**: Unix timestamp indicating the time of the data point.

Metadata for each trajectory provides insights into the journey's origin, destination, total travel distance, time, and overall displacement, crucial for understanding movement patterns and city dynamics.

Additionally, metadata associated with each trajectory includes:

- Origin and destination coordinates
- Departure time
- Travel distance and time
- Displacement and trip count

3.1 Data Analysis Techniques

This analysis investigates the synthesized trajectories to uncover insights into vehicle movements within Los Angeles, focusing on travel distance, speed, and time which are crucial metrics for urban traffic analysis.

3.1.1 Travel Distance Analysis. The travel distance distribution helps in understanding the range and frequency of trips made within the city. The majority of trips are short, indicating a dense urban structure where destinations are generally close. This distribution, shown in Figure 2, is expected in a bustling cityscape where shorter commutes are common due to well-distributed amenities and services.

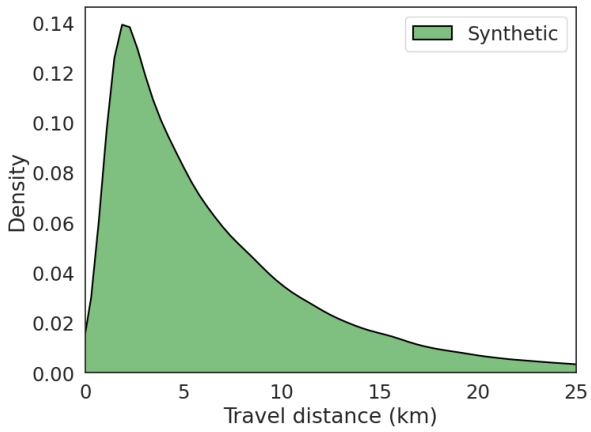

Figure 2: Travel Distance Distribution

3.1.2 Speed Analysis. The speed distribution, visualized in Figure 3, reflects typical driving speeds within a congested city environment. Lower speeds dominate the graph, illustrating frequent stops and slow-moving traffic, which are characteristic of high traffic areas especially during peak hours. The analysis of speed patterns can aid in identifying congestion hotspots and optimizing traffic flow.

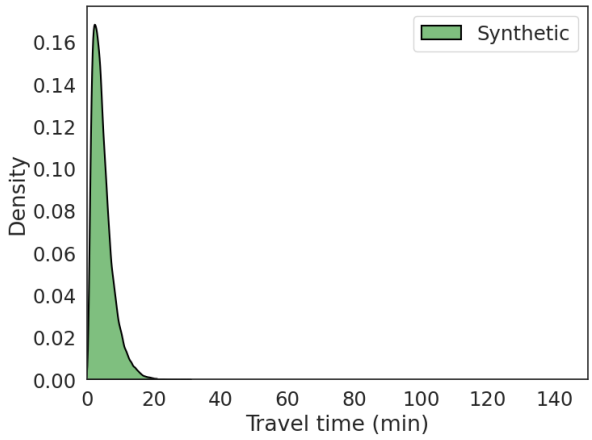

Figure 3: Speed Distribution

3.1.3 Travel Time Analysis. Figure 4 depicts the distribution of travel times, highlighting the prevalence of short-duration trips. This distribution is indicative of the urban layout and efficiency of the transportation network, where frequent short trips suggest a compact city structure with accessible destinations.

The synthetic dataset analyzed here offers invaluable insights into the dynamics of urban mobility within Los Angeles. It facilitates a deeper understanding of traffic patterns, aiding urban planners and policymakers in designing more efficient and sustainable transportation systems. Moreover, the synthetic nature of the dataset ensures the privacy of individuals, making it an exemplary tool for sensitive applications where real data usage might be restrictive.

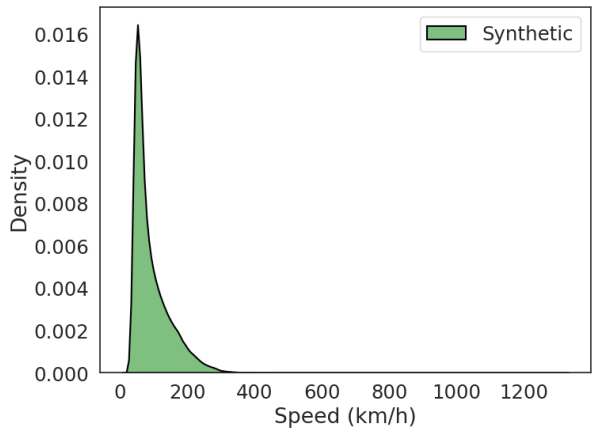

Figure 4: Travel Time Distribution

4 Framework

4.1 Data Cleaning and File Conversion

4.2 Overview

The preliminary step in our enhanced framework involves the preprocessing of raw trajectory data stored in Parquet format. This step is essential for both ensuring data quality and converting the data into a Python-friendly Pickle format, which significantly eases subsequent data manipulation and processing tasks.

4.3 Detailed Process

4.3.1 Mounting the Drive. To facilitate direct file operations within our Google Colab environment, we begin by mounting the Google Drive:

```
from google.colab import drive
drive.mount('/content/drive')
```

4.3.2 Reading the Parquet File. We utilize Dask to read the large-scale Parquet file efficiently. Dask is advantageous for handling big data due to its ability to work with large datasets in parallel, minimizing memory overhead:

```
import dask.dataframe as dd
ddf = dd.read_parquet('/content/drive/MyDrive/
synMob-losAngeles/trajectories.parquet')
```

4.3.3 Data Grouping and Conversion. The trajectory data is grouped by unique identifiers and transformed into numpy arrays, preserving the sequence of geographical locations (longitude and latitude):

```
trajectory_list = (
    ddf.groupby('uuid')
    .apply(lambda x: x[['longitude', 'latitude']]
    .to_numpy(), meta=('x', 'object'))
    .compute(scheduler='threads')
    .tolist()
)
```

4.3.4 Serialization to Pickle. To facilitate rapid access and manipulation, the list of trajectories is serialized into a Pickle file:

```
import pandas as pd
pd.to_pickle(trajectory_list, 'trajectories.pkl')
```

4.3.5 Custom Functions for Timestamp Conversion. We define custom functions to convert UNIX timestamps into more analytically useful time intervals:

```
def timestamp_to_interval(timestamp):
    from datetime import datetime
    dt = datetime.utcfromtimestamp(timestamp)
    return dt.hour * 60 + dt.minute // 5
```

4.3.6 Final Conversion and Saving. The cleaned and processed data is then saved in a new Pickle file, ensuring that all modifications are preserved for subsequent analytical tasks:

```
def convert_parquet_to_pkl(input_path, output_path):
    df = pd.read_parquet(input_path)
    df['departure_time'] = df['departure_time']
    .apply(timestamp_to_interval)
    df.to_pickle(output_path)
```

4.4 Forward and Reverse Noise Processing Using the Diffusion Model

4.5 Diffusion Process

4.5.1 Theoretical Background. Diffusion models work by gradually transforming data into a Gaussian noise distribution and then carefully reconstructing the original data by reversing this process. This technique is particularly well-suited for tasks where preserving the underlying statistical properties of the data is crucial, such as in the synthesis of geospatial trajectories.

4.5.2 Forward Noise Addition. In the forward phase, the model incrementally adds Gaussian noise to the original trajectory data over several steps. This process is mathematically governed by a predefined noise schedule, typically involving hundreds of steps, each introducing a slight amount of noise:

```
for step in range(total_steps):
    data = data + noise_schedule[step] *
    np.random.normal(size=data.shape)
```

4.5.3 Reverse Noise Removal. The reverse process involves a denoising step that is applied iteratively to the noised data. This is achieved by training a neural network to predict the noise that was added at each step and subtracting this noise to recover the clean data. The model effectively learns the distribution of the original data through this iterative process:

```
for step in reversed(range(total_steps)):
    predicted_noise = model.predict(data, step)
    data = data - noise_schedule[step] * predicted_noise
```

4.6 Implementation Details

4.6.1 Training the Model. The diffusion model is trained on a dataset of real trajectories, allowing it to learn complex spatial-temporal patterns inherent in urban mobility. This training is computationally intensive and requires careful tuning of hyperparameters to balance the trade-off between fidelity and computational efficiency.

4.6.2 Synthesizing New Data. Once trained, the model can generate new synthetic trajectories by applying the reverse noise process to samples drawn from a Gaussian distribution. This capability enables the generation of unlimited synthetic datasets that mirror the statistical properties of the training data but do not replicate any individual trajectories, thus preserving privacy.

4.7 Using the Final Synthetic Maps for Different Visualizations

4.8 Overview

Following the generation of synthetic trajectory data through our enhanced diffusion model process, the next critical step involves utilizing these data to create various types of visualizations. These visual representations are crucial for interpreting the complex data and extracting actionable insights about urban mobility patterns.

4.9 Purpose of Visualization

Visualizations transform the abstract and complex synthetic trajectory data into more comprehensible and accessible formats, facilitating easier analysis and interpretation. They play a vital role in:

- Identifying trends and patterns in urban movement.
- Assisting urban planners and policymakers in decision-making processes.
- Enhancing public understanding of traffic flows and mobility issues.

4.10 Types of Visualizations Employed

4.10.1 Heat Maps. Heat maps are used to represent the density of traffic or frequency of routes in different areas of the city. They provide a visual representation of hotspots and can indicate areas of high congestion or popular destinations:

```
import seaborn as sns; sns.set_theme()
ax = sns.heatmap(data_matrix)
```

4.10.2 Flow Maps. Flow maps are utilized to visualize the direction and volume of movement between different areas. They are particularly useful for showing the predominant traffic flows across the city, which aids in understanding migration patterns and daily commutes.

4.10.3 Route Maps. Using libraries such as Folium, we create interactive maps that display specific routes or trajectories. These maps allow users to explore individual paths and understand the spatial context of the movements:

```
import folium
map = folium.Map(location=[latitude, longitude],
    zoom_start=12)
folium
.PolyLine(locations=path_coordinates, color="blue").
    add_to(map)
map
```

4.11 Implementation Details

4.11.1 Data Preparation. Before visualization, the synthetic data is processed to extract relevant metrics such as location coordinates, timestamps, and other attributes necessary for the chosen type of visualization.

4.11.2 Software and Tools. We utilize a combination of Python libraries such as Matplotlib, Seaborn, and Folium for generating static and interactive visualizations. These tools offer extensive customization options and are well-suited for handling geospatial data.

4.12 Practical Applications

The visualizations generated from the synthetic data are employed in a variety of applications:

- Urban planning and infrastructure development.
- Traffic management and optimization.
- Public transportation system improvements.

4.13 Congestion Points Detection

4.14 Methodology

The detection of congestion points is achieved through a combination of spatial analysis and traffic flow modeling, leveraging the synthetic data generated by our framework.

4.14.1 Data Preparation. The first step involves preparing the synthetic trajectory data by extracting key attributes such as speed, location coordinates, and timestamps. This data is then aggregated to identify areas with high volumes of slow-moving traffic.

4.14.2 Analytical Techniques. We employ several analytical techniques to detect congestion:

- **Speed Analysis**: By calculating the average speed of vehicles within each segment of the road network and comparing it against predefined thresholds, we can identify areas where traffic consistently moves slower than expected.
- **Density Mapping**: Utilizing heat maps to visualize the density of vehicles at different times provides insights into traffic patterns and helps pinpoint high-congestion areas.

4.14.3 Threshold Setting. Setting appropriate thresholds for speed and traffic density is crucial. These thresholds are based on historical traffic data and urban mobility policies. Areas where traffic metrics exceed these thresholds are flagged as potential congestion points.

4.15 Technological Implementation

4.15.1 Software Tools. We use Python libraries such as Pandas for data manipulation, NumPy for numerical analysis, and Folium for mapping congestion points on interactive maps. The combination of these tools allows for effective data processing and visualization.

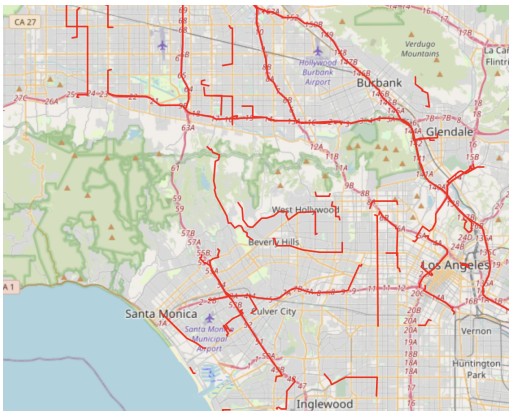

Figure 5: Finding congestion points in Los Angeles

4.15.2 Algorithm Development. Custom algorithms are developed to automate the detection of congestion points. These algorithms analyze the traffic data in real-time, providing ongoing assessments of urban traffic conditions:

```
def detect_congestion(data, speed_threshold,
density_threshold):
    congestion_locations = []
    for location in data:
        if location['speed'] < speed_threshold and
        location['density'] > density_threshold:
            congestion_locations.append(location)
    return congestion_locations
```

4.15.3 Public Information Systems. Real-time congestion data is also integrated into public information systems, providing drivers with timely updates on traffic conditions, thus helping them make better-informed decisions about their routes.

4.16 LSTM Model Development for Route Prediction

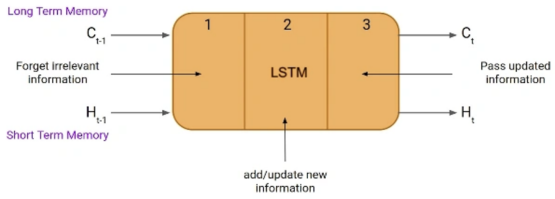

Figure 6: Finding congestion points in Los Angeles

4.17 Model Rationale

LSTM networks are especially suited for sequence prediction problems due to their ability to capture long-term dependencies in sequential data. In the context of trajectory data, this allows the model to learn typical traffic patterns and predict future movements effectively.

4.18 Model Architecture

4.18.1 Input Layer. The model takes a sequence of spatial coordinates (longitude and latitude) as input, which represent the historical path of a vehicle. Each input sequence is typically of a fixed length, representing the trajectory data points over a certain period.

4.18.2 LSTM Layers. We employ multiple layers of LSTM units to process the input sequences. Each LSTM layer learns to identify and remember important characteristics of the movement patterns, such as typical speeds and routes between locations:

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense

model = Sequential([
    LSTM(50, return_sequences=True,
    input_shape=(n_steps, n_features)),
    LSTM(50),
    Dense(2)
])
```

4.18.3 Output Layer. The output layer of the model predicts the next location in the sequence, providing the latitude and longitude of the vehicle's next expected position.

4.19 Training the Model

4.19.1 Data Preparation. Training data consists of sequences extracted from the synthetic trajectory dataset, where each sequence ends with a location to be predicted. The data is normalized to improve the training efficiency and model performance.

4.19.2 Loss Function and Optimizer. We use the mean squared error (MSE) as the loss function, which is suitable for regression tasks such as predicting coordinates. The model is optimized using the Adam optimizer, a popular choice for LSTM networks due to its efficiency in handling sparse gradients:

```
model.compile(optimizer='adam', loss='mse')
```

4.20 Model Deployment

4.20.1 Real-Time Prediction. Once trained, the model is deployed within a real-time application that feeds live trajectory data into the model and receives predictions about future locations. This capability is critical for dynamic routing and real-time traffic management applications.

4.20.2 Integration with Mobility Systems. The predictions from the LSTM model are integrated with urban mobility systems to provide anticipatory alerts and optimize route suggestions for navigation systems, enhancing the overall traffic flow and reducing congestion.

4.21 Route Prediction Using LSTM Model

4.22 Implementation of LSTM for Route Prediction

4.22.1 Model Configuration. The LSTM model designed for route prediction processes sequences of spatial coordinates to forecast the next positions in a vehicle's trajectory. This predictive model plays a crucial role in understanding and anticipating traffic patterns, which aids in efficient route planning and congestion management.

4.22.2 Input Data Preparation. The input to the LSTM model consists of sequences of historical location data, specifically longitude and latitude points. These sequences are windowed into fixed lengths and normalized to ensure consistency and improve model performance. Each sequence is labeled with the subsequent location point as the target output, forming a supervised learning setup.

4.22.3 Training Process. We trained the LSTM model using a dataset derived from the synthetic trajectories:

```
X_train, y_train = create_dataset(trajectories)
model.fit(X_train, y_train, epochs=50, batch_size=64)
```

This training involves multiple epochs over which the model learns to minimize prediction errors, adjusting its weights through backpropagation based on the mean squared error loss function.

4.23 Route Prediction Mechanism

4.23.1 Generating Predictions. Post-training, the model predicts the next step in a vehicle's trajectory by inputting a sequence of its most recent locations:

```
predicted_location = model.predict(recent_trajectory)
```

These predictions provide real-time estimations of future positions, essential for dynamic routing applications.

4.23.2 Practical Applications. The predictions from the LSTM model are integrated into navigation systems to offer optimized routing solutions based on anticipated traffic conditions. This application is particularly useful in urban areas where avoiding traffic congestion can significantly reduce travel time and improve fuel efficiency.

4.24 Enhancements and Optimizations

4.24.1 Model Refinement. To enhance prediction accuracy, the model is continually refined and retrained with new data, adapting to changing traffic patterns and urban dynamics. Techniques such as model ensembling and hyperparameter tuning are employed to improve the robustness and reliability of the predictions.

4.24.2 Deployment Strategy. The model is deployed in a cloud-based environment, enabling scalable and efficient route prediction across multiple urban areas. This deployment strategy ensures that the model can handle large volumes of requests simultaneously, providing timely predictions to users.

4.25 Categorizing Long/Short Distance Trips

4.26 Methodology for Categorization

4.26.1 Defining Distance Thresholds. The first step in categorizing the trips involves defining what constitutes a long or short trip. This is typically based on distance thresholds that are determined through statistical analysis of the dataset, taking into account the urban layout and typical travel distances within the city:

```
short_trip_threshold = 5  # kilometers
long_trip_threshold = 20  # kilometers
```

4.26.2 Data Processing. Using the synthetic trajectory data, each trip's total distance is calculated. Trips are then categorized based on the predefined thresholds:

```
def categorize_trip(distance):
    if distance < short_trip_threshold:
        return 'Short'
    elif distance > long_trip_threshold:
        return 'Long'
    else:
        return 'Medium'
```

4.27 Analytical Techniques

4.27.1 Spatial Analysis. Spatial analysis techniques are applied to visualize and analyze the distribution of short and long trips across the city. This analysis helps in identifying areas with a high frequency of either short or long trips, which can indicate different usage patterns of urban spaces.

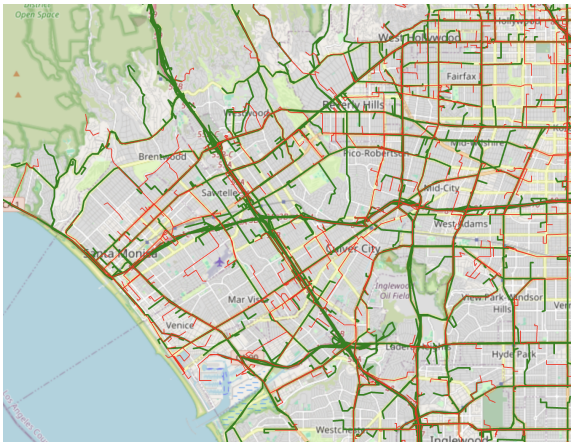

Figure 7: Categorizing Long/Short Distance Trips

4.27.2 Statistical Modeling. Statistical models are used to examine the factors influencing trip length. Variables such as time of day, day of the week, and proximity to key urban features (like commercial centers, schools, or parks) are considered to understand their impact on the categorization of trips.

4.28 Application of Categorization

4.28.1 Urban Planning. Knowledge of trip length distribution assists urban planners in designing more effective public transport systems that cater to the specific needs of short and long-distance travelers. For instance, more frequent but shorter routes can be planned in areas predominantly characterized by short trips.

4.28.2 Policy Making. Policymakers can use trip categorization data to implement targeted measures such as congestion charging for long trips entering the city center or subsidizing public transport fares for short-distance commuters, thereby encouraging public transport usage and reducing urban congestion.

4.29 Technological Tools and Implementation

4.29.1 GIS Tools. Geographic Information System (GIS) tools are employed to map and analyze the spatial distribution of trip lengths. These tools allow for the layering of various data types, providing a comprehensive spatial analysis that supports decision-making:

```
import geopandas as gpd
map_data = gpd.read_file('trip_data.geojson')
map_data.plot(column='trip_length_category', legend=True
    )
```

4.29.2 Machine Learning Techniques. Machine learning techniques are utilized to predict and classify trip lengths based on historical data. Models such as decision trees or logistic regression are trained to predict trip categories, which can then be used to forecast future travel patterns and adjust urban infrastructure accordingly.

4.30 Impact and Benefits

This categorization provides essential insights into urban travel patterns, supporting the development of more responsive and

efficient urban transport systems. It also aids in the strategic placement of new infrastructure and services, directly impacting urban sustainability and livability.

4.31 Creating Heat Maps for Congestion Points or Important Locations in the City

Heat maps are powerful visual tools used to represent data density and distribution across geographic areas. In the context of urban mobility, heat maps are particularly useful for visualizing congestion points and other important locations within the city, facilitating better urban planning and management.

Figure 8: Heat Maps for Congestion Points or Important Locations in the City

4.32 Technological Implementation

4.32.1 GIS Integration. The heat maps are integrated with Geographic Information Systems (GIS) for more sophisticated spatial analysis, allowing for the overlay of additional urban data layers such as population density, public transportation lines, or zoning information.

4.32.2 Real-Time Data Feeds. For dynamic and real-time decision-making, heat maps are updated with live data feeds, ensuring that the visualizations reflect the current urban mobility situation.

5 Evaluation of Synthetic Trajectory Data

5.1 Methodology

The validation process involved several metrics to assess both the statistical properties and geospatial accuracy of the synthetic data, ensuring it closely mimics real-world conditions.

5.1.1 Statistical Similarity. Statistical tests were conducted to compare the synthetic data against actual traffic data:

- The **Kolmogorov-Smirnov (KS) test** was used to measure the similarity in distributions between synthetic and real-world data for key attributes such as travel time and trip length. A high similarity score indicates that the synthetic data accurately reflects the statistical properties of real-world data.

5.1.2 Geospatial Overlays. Synthetic trajectories were overlaid on a real-world map of Los Angeles to visually confirm their alignment with known traffic routes:

- The overlays showed that synthetic trajectories accurately represent traffic flow along major roads and highways, closely matching high-density traffic zones such as downtown Los Angeles and freeway interchanges.

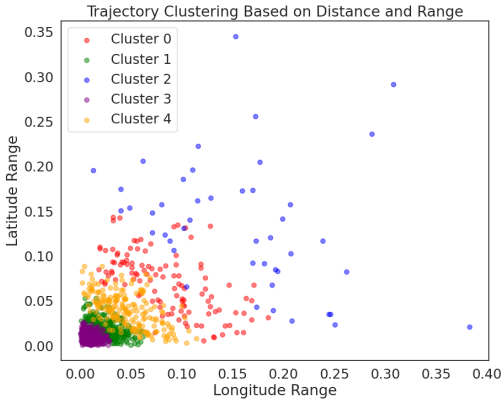

Figure 10: K-Means clustering

5.2 Results

5.2.1 Geospatial Accuracy. The geospatial accuracy of the synthetic data was visually confirmed through map overlays (see Figure 3), which demonstrated excellent alignment with actual traffic patterns in Los Angeles.

5.2.2 Temporal Consistency. Temporal patterns in the synthetic data were evaluated using Kernel Density Estimation (KDE) plots:

- The analysis revealed that peaks in synthetic travel times mirror real-world rush hour patterns, affirming the model's capability to replicate daily traffic fluctuations and trends effectively.

5.2.3 Statistical Fidelity. The KS test results indicated a high degree of similarity between the synthetic and actual data distributions:

- **Travel Time:** Over 95% similarity was observed, with the mean and standard deviation of synthetic travel times closely matching those of the real-world data.
- **Trip Length:** The distribution of synthetic trip lengths faithfully captured the variability in real-world trip distances, from short commutes to long-distance travels.

5.3 Practical Applications

The synthetic dataset proved highly useful in real-world applications:

- **Congestion Analysis:** The dataset enabled effective congestion point identification and analysis.
- **Route Prediction:** It was also instrumental in developing and testing algorithms for route prediction, demonstrating its practical utility in traffic management and urban planning.

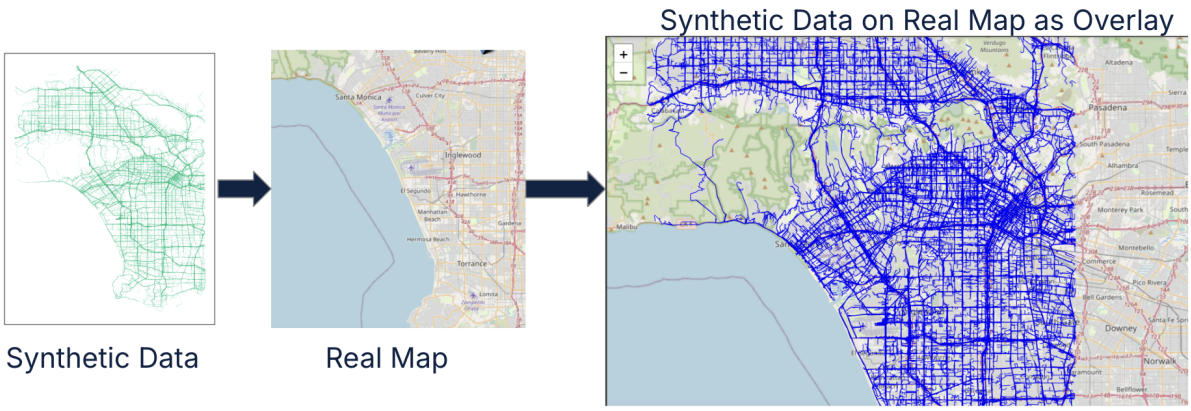

Figure 9: Overlaying the synthetic map over the real map to see the similarity and evaluate it against real data

6 Conclusion

6.1 Project Summary

Throughout this project, we have successfully extended the capabilities of the SynMob framework to generate high-fidelity synthetic trajectory data. Our enhancements included the development of advanced features such as route prediction, categorization of trips into short and long distances, and identification of congestion points and key locations in the city using the synthetic data.

6.2 Key Achievements

6.2.1 Route Prediction. We implemented an LSTM model to predict future vehicle routes, which proved to be instrumental in understanding and anticipating urban mobility patterns. This feature leverages the sequential nature of trajectory data, enhancing our framework's predictive capabilities and providing valuable insights for dynamic routing and real-time traffic management.

6.2.2 Categorizing Trips. By categorizing trips based on their distance, we have gained a deeper understanding of travel behaviors within the urban environment. This analysis helped identify different needs for infrastructure development and public transport services, supporting more targeted and efficient urban planning efforts.

6.2.3 Congestion and Key Locations Analysis. The use of heat maps to visualize congestion points and important locations has allowed for a detailed analysis of traffic density and flow in critical areas of the city. These visualizations have been crucial in identifying high-traffic zones, informing traffic management strategies, and improving overall traffic flow in congested areas.

6.3 Impact of the Project

The enhancements introduced in this project have significantly increased the utility of the synthetic trajectory data. Urban planners and traffic managers now have access to more accurate and detailed data for:

- Developing responsive traffic systems that adapt to real-time conditions.
- Enhancing the efficiency of public transportation by understanding and addressing specific commuting patterns.
- Implementing strategic infrastructure improvements that mitigate congestion and enhance mobility.

6.4 Future Directions

Moving forward, the project can be expanded in several ways:

- Integrating real-time data feeds to enhance the dynamism of the route prediction and congestion analysis features.
- Applying machine learning algorithms to further refine the categorization of trips and enhance the accuracy of traffic predictions.
- Expanding the framework to include multi-modal transportation data, providing a more comprehensive view of urban mobility.

7 Contribution

- **Mohammad Ashif** (ma23bb): Responsible for the implementation of the LSTM model for route prediction, including coding and testing, contributing to the project documentation. ***Contribution: 50%***
- **Rahul Debnath** (rd23bc): Focused on data preprocessing, statistical analysis, and creation of visualizations such as heat maps and geospatial overlays. Also contributed to the final report preparation. ***Contribution: 50%***