Machine Learning Algorithms for Transportation Mode Prediction: A Comparative Analysis

Samer Murrar, Fatima Alhaj
Faculty of Information Technology, Applied Science Private University, Amman, Jordan
202215019@students.asu.edu.jo, f_alhaj@asu.edu.jo

Mahmoud H. Qutqut
Faculty of Computer Science, University of New Brunswick, Fredericton, NB, E3B 5A3 Canada
m.qutqut@unb.ca

Keywords: Transportation Mode, GPS Trajectories, Decision Tree Algorithm

Received:

This study investigated the performance of various machine learning algorithms in predicting transportation modes from large datasets. The investigated algorithms include Multilayer Perceptron (MLP), K-Nearest Neighbors (KNN), Decision Tree, Long Short-Term Memory (LSTM), Recurrent Neural Network (RNN), and Logistic Regression. We rigorously evaluated each algorithm's performance using a robust set of metrics such as precision, recall, and F1-score. This study comprehensively explains the algorithm's capabilities, strengths, and potential weaknesses across seven transportation categories: 'walk', 'bike', 'bus', 'car', 'taxi', 'train', and 'subway'. The Decision Tree (DT) model consistently outperformed the others, demonstrating superior accuracy and a better balance of precision and recall across all modes of transportation. Specifically, it achieved precision, recall, and F1 scores of around 83% to 94% across all categories. These findings underline the suitability of the DT model for this classification task and its potential for further applications in transportation mode prediction based on large datasets. However, other algorithms like LSTM and RNN also showed promising results in certain categories, suggesting the value of continued exploration of different models depending on specific use cases.

1 Introduction

The complexities of how people move within a community - their travel behaviors and transportation choices - play a critical role in many aspects of urban planning and development. This intricate mosaic of movement patterns is extremely important, serving as a valuable tool for policymakers, transportation planners, and urban developers. It helps to predict future transportation needs accurately, guides critical decision-making processes, and promotes environmentally friendly practices [1].

Insights gleaned from this data are used by transportation planners and policymakers to accurately forecast future demand for various modes of transportation. It provides recommendations, aiding informed decisions in infrastructure and service investment decisions [2]. For example, if analysis shows that a sizable proportion of the population relies on public transport, there is a clear justification for investments in expanding bus lines or adding tube stations [3].

Furthermore, data on travel behavior is a valuable tool for promoting environmentally sustainable transportation practices. If, for example, a sizable proportion of the population relies solely on private automobiles for commuting, this may indicate a need for more environmentally friendly transportation options. Cycling lanes, carpooling programs, and better public transportation are all potential solutions [4].

Understanding travel behavior can also reveal implications for health and safety. Assume that many people prefer cycling, but that there are a high number of traffic accidents involving cyclists in the area. In that case, this troubling trend may indicate the need for improved bike infrastructure or increased safety education [5].

Moreover, this comprehension can shed light on potential equity issues. When lower-income people rely heavily on public transportation and the service is either inadequate or unaffordable, it is clear that policy changes are needed to ensure equitable transportation access [6].

Understanding travel behavior has a significant impact on economic development. When deciding where to locate, businesses in a variety of industries, including retail, food, and entertainment, frequently consider potential customers' modes of transportation [4].

Another critical application is for communities to understand how their populations travel to effectively prepare for and respond to a disaster. This information can be used to predict which roads may require immediate clearance and which modes of transportation should be restored as soon as possible [7].

This paper is organized as follows. Section 2 explores the background information. Section 3 conducts a thorough review of the existing literature. Section 4 provides a detailed explanation of our proposed approach, which includes data preprocessing, feature construction, and the intricate aspects of the model architecture. Section 5 contains a thorough examination of our experimental findings. Finally, in Section 6, we wrap up our discussion and draw meaningful conclusions.

2 Background

Understanding and predicting travel behavior is a difficult task that necessitates the use of numerous types of data and sophisticated analytical techniques. As location-acquisition technology has advanced, GPS trajectory data has become one of the most important sources of information for researching human mobility patterns. By providing extensive records of individuals' spatial-temporal travels, these data provide significant insights into how, why, and where people travel. However, because of the inherent complexity and variety of human movement, extracting meaningful insights from raw GPS trajectory data is a difficult process [8].

To deal with this difficulty, various computational strategies have been developed over the years. Among these, machine learning algorithms have emerged as particularly promising. They are capable of learning complex patterns from massive amounts of data, making them ideal for jobs such as transportation mode prediction. Decision trees, for example, have been widely used due to their interpretability and versatility [9].

However, the performance of these algorithms is heavily reliant on the quality of the incoming data and how it is handled. As a result, data preprocessing and feature extraction are critical steps in model development. Techniques such as data cleaning, normalization, and encoding are frequently used to convert raw GPS data into a format suitable for machine learning algorithms.

The study intends to use these approaches, specifically the DT algorithm, to forecast transportation modes from GPS trajectory data. This study not only contributes to the larger field of travel behavior analysis, but it also gives legislators, transportation planners, and urban developers practical insights [9].

3 Related Work

Over the years, numerous studies have been conducted to unravel the complexities of travel behavior and transportation mode prediction. These investigations have shed light on various aspects of travel behavior, influencing the evolution of prediction models and methodologies. Despite its simplicity, our approach has the potential to provide significant insights into this multifaceted area. Previous research emphasizes the significance of decoding human mobility patterns - a complex web of numerous factors that influence travel choices. These studies used a variety of methodologies to unravel this complex issue, ranging from traditional statistical methods to advanced machine learning algorithms.

Convolutional Neural Networks (CNNs) have been widely used among these because of their ability to automatically learn and extract features from spatial data. Regardless of their advantages, CNNs require a large amount of train-

Table 1: Comparison of Different ML Models

Model	Accuracy %	F1-Score %
CNN-ensemble [10]	81.92	81.77
LSTM-based DNN [11]	80.99	80.53
GRU-based DNN [12]	81.12	80.83
LRCN [13]	82.32	82.30
STPC-Net [14]	81.22	81.05
CE-RCRF [1]	85.23	85.41

ing data for optimal performance and can be computationally intensive, making them slower to train [10]. Other works were based on deep Neural Networks (DNN) such as [11], [12]. Even though They are effective at learning and remembering long sequences, they are computationally demanding and may be prone to overfitting due to their complexity. On the other hand, long-term Recurrent Convolutional Network (LRCN) combines the strengths of CNN and RNN rather than specifically incorporating LSTMs. LRCN is intended to efficiently process sequential data with spatial features by combining the spatial feature extraction capabilities of CNNs with the temporal modeling capabilities of RNN [13].

The Spatial-Temporal Pattern Chain Network (STPC-Net) is a network that models complex spatial-temporal patterns and is specifically designed for transportation mode identification. Despite its efficacy, the model may be overly complex for tasks where simpler models would suffice [14]. The Contrast-Enhanced Robust Conditional Random Field (CE-RCRF) method combines the advantages of both the Contrast Enhancement (CE) and the Robust Conditional Random Field (RCRF) methods. It is more complex and computationally demanding than other methods for dealing with noise and uncertainties in GPS data [1]. The investigation of these techniques and their effectiveness in predicting travel behavior has yielded useful insights for future research in this field. The results of these techniques, including their accuracy and F1-score, are shown in Table 1.

4 Proposed Methodology

Our paper proposes a comprehensive framework for Travel Mode Identification that includes four steps; namemly data preprocessing, feature construction, predictive models, and evaluation methods. These modules collaborate to form a unified pipeline for identifying travel modes effectively and efficiently.

The first step, data preprocessing, is crucial in preparing raw data for further analysis. To ensure the quality and consistency of the data, this step involves cleaning and standardizing it. We ensure the reliability of the data used for travel mode identification by addressing missing values, outliers, and inconsistencies. This step also included extracting meaningful features. These characteristics are carefully chosen based on their relevance and potential impact on travel mode identification.

Then, we employed an array of predictive models to classify travel modes based on the constructed features. Our goal was to juxtapose the performance of these models to discern the most effective one(s) for our specific task. The models used include Multilayer Perceptron (MLP), Long Short-Term Memory (LSTM), Recurrent Neural Networks (RNN), Decision Trees, Logistic Regression, and K-nearest Neighbors (KNN) algorithms. Each of these models was trained on the same set of training data and evaluated on a common testing set to ensure a fair comparison. As part of the modeling process, we paid attention to appropriate model-specific configurations, such as choosing an optimal number of layers for MLP or tuning the number of neighbors in KNN, to guarantee each model's best possible performance.

In the final step, we evaluate the efficacy of the various predictive models that we have used. We use a set of performance metrics for this purpose, including accuracy, precision, recall, and the F1 score. These metrics enable us to assess each model's ability to correctly identify travel modes. Every individual model is subjected to a thorough evaluation, providing us with detailed insights into the model's strengths, weaknesses, and distinguishing characteristics. We determine the most efficient and effective model for travel mode identification by comparing the performance of all of these models. As part of the evaluation process, we also consider the computational efficiency and ease of tuning the model's hyperparameters. This in-depth examination seeks to provide a clear understanding of the best-performing model on the travel mode identification dataset.

In this study, we use the extensive geographic

data contained in Microsoft's GeoLife GPS Trajectory 1.3 dataset [1], a robust repository that includes a wealth of information about human mobility patterns [15].

The GeoLife GPS Trajectory 1.3 dataset from Microsoft is a rich repository of geographic data that provides a comprehensive view of mobility patterns, making it an invaluable resource for geospatial researchers and developers. This dataset, derived from a variety of location-enabled devices, enables a thorough examination of spatial and temporal behaviors [15].

The GPS Trajectory 1.3 dataset, which was created as part of Microsoft's GeoLife project, contains the mobility data of 182 users from April 2007 to August 2012. This massive dataset includes 17,621 trajectories and over 24.7 million individual location points [15].

Each trajectory in the dataset is a series of time-stamped points that provide not only location information but also a chronological perspective necessary for understanding movement patterns over time. The location points were recorded at five-second intervals, resulting in a high-resolution view of each trajectory [15].

This dataset is geographically diverse, covering a wide range of areas in more than 30 Chinese cities. Because of the broad geographic scope, comparative studies of mobility patterns in various cultural and urban contexts are possible [15].

Furthermore, one distinguishing feature of this dataset is that it includes a wealth of associated information in addition to geographic and temporal data. For some users, a mode of transportation is available, providing insight into the options of walking, cycling, driving, or taking public transportation. This extra layer of data can be especially useful in studies that look at transportation options and travel behavior [15].

4.1 Data Preprocessing

Data preprocessing was the first step in preparing our dataset for further research. This stage essentially ensured that the dataset's format was standardized, that unnecessary attributes were removed, and that all necessary changes were made. The following preprocessing procedures were carried out:

1. Data Integration:
   Data from 18,670 files from 182 people were combined into a single data file, and 69 users' trajectory labels were similarly integrated. After exporting the trajectory data points to a unified dataset, they were linked with their labels, yielding approximately 24,876,978 records.

2. Data Reduction:
   The process of data reduction entailed removing irrelevant attributes from the dataset to streamline it. This includes the removal of the "Param" attribute, which had no informational value because it was consistently 0 across all instances. Furthermore, due to the prevalence of undefined and incongruous values, the "Altitude" attribute was also eliminated. Finally, cases lacking labels and those with a zero value for the time attribute were systematically eliminated from the dataset to ensure data integrity and to assist the requirement of supervised learning.

4.2 Feature Construction

Following data Preprocessing, the next crucial stage is feature construction, which tries to establish meaningful features that will serve as valuable inputs for the modeling process. The following steps were taken to complete this process:

1. Attribute Generation:
   The process of the feature creation procedure began by extracting critical attributes from the existing GPS coordinates and timestamps. To begin, the property denoting the distance to the next location was determined in kilometers using the equation (1). Using the equation (2), the time to the next location was then calculated in hours. Along with this, the velocity attribute was introduced, which was calculated as the distance-to-time ratio and expressed in kilometers per hour using the equation (3). This change improved the dataset's acceptability for further analysis and added an essential predictive feature to the model.

[1] https://www.microsoft.com/en-us/research/publication/geolife-gps-trajectory-dataset-user-guide/

The dataset was supplemented by computing two more nuanced attributes in addition to these fundamental attributes. These included the acceleration, which was stated in kilometers per hour squared and calculated using equation (4), and the angular velocity, which was expressed in radians per hour and calculated using equation (5). The inclusion of these complex features increased the dataset's analytical reach, offering deeper and more detailed insights for the following stages of analysis and modeling.

2. Outlier Removal
To ensure data integrity and improve the robustness of our analysis, we started by removing any instances in the dataset that showed dubious or physically impossible travel situations. Cases with negative Velocity, Time to the next point, or Distance to the next point values, in particular, were immediately eliminated. This critical phase aided in the removal of data irregularities and other errors that may have occurred during the data collection procedure.

3. Data Constraint Application
We established speed limits for each distinct mode of transportation after removing these outliers. This necessitated setting average speed limits for various modes of transportation, including walking and biking, as well as other types of motorized and public transportation. Instances that exceeded the established speed limits were deemed abnormal and were removed from the dataset. By adhering to realistic speed constraints, our dataset remained grounded in reasonable travel conditions. The speed limits for each mode of transportation are depicted in Table 2. The study's reliability and accuracy were significantly improved by this thorough approach to feature development, which included screening out rare situations and adhering to strict travel mode speed thresholds.

4.3 Travel Mode Identifier

The cornerstone of every machine learning endeavor is undeniably the dataset in use. As we transition from the data preprocessing and feature construction phases into model development, it becomes imperative to gain an understanding of the refined dataset. It is the characteristics of this dataset that illuminate the intricacies of the problem at hand and hint at potential challenges that might arise during model construction. Post-processing, our dataset encompasses 5,318,377 instances, each assigned to one of seven distinct classes. A detailed breakdown of these classes can be found in Table 3.

Table 2: Speed Limits for Each Mode of Travel

Travel Mode	**Speed Limit (km/h)**
Walk	12
Bike	50
Car	160
Taxi	140
Bus	120
Subway	150
Train	320

Table 3: Class Distribution After Data Preprocessing and Feature Construction

Transportation Mode	**Counts**	**Percentage (%)**
Walk	1,497,710	28.16
Bus	1,275,389	23.98
Bike	945,077	17.77
Train	560,528	10.54
Car	511,585	9.62
Subway	286,112	5.38
Taxi	241,976	4.55

To compare and assess the performance of various classifiers, we used Multilayer Perceptron (MLP), Logistic Regression, k-nearest Neighbors (KNN), Decision Trees, Long Short-Term Memory (LSTM), and Recurrent Neural Networks (RNN) in our approach. These classifiers were chosen for their demonstrated ability to handle multiclass classification tasks in a variety of contexts. Each classifier has its implementation method, but they all share a common foundation of preprocessing steps and performance evaluation metrics.

We implemented machine learning algorithms in Google Colab notebooks using Python [2]. The initial steps for all algorithms were similar. We imported the necessary libraries, loaded the data

[2]https://colab.research.google.com/

Figure 1: Formula for calculating the distance between two points

$$d = 2 \times R \times \arcsin\left(\sqrt{\sin^2\left(\frac{\text{Lat2} - \text{Lat1}}{2}\right) + \cos(\text{Lat1}) \times \cos(\text{Lat2}) \times \sin^2\left(\frac{\text{Long2} - \text{Long1}}{2}\right)}\right) \quad (1)$$

where:
- d represents the distance to the next point in kilometers (km),
- 2 represents a constant factor used in the calculation,
- R is the radius of the Earth in kilometers, taken to be approximately 6,371 km,
- $Lat1$ and $Lat2$ are the latitude coordinates of the two points,
- $Long1$ and $Long2$ are the longitude coordinates of the two points.

Figure 2: Formula for calculating the time to reach the next point

$$t = (\text{Datetime2} - \text{Datetime1}) \times 24.0 \quad (2)$$

where:
- t represents the time duration between two points in hours (h),
- $Datetime1$ and $Datetime2$ are the timestamps of the two points.

Figure 3: Formula for calculating the velocity

$$v = \frac{d}{t} \quad (3)$$

where:
- v represents the velocity in kilometers per hour (km/h),
- d represents the distance to the next point in kilometers,
- t is the time taken to travel from the first point to the second point in hours.

Figure 4: Formula for calculating the acceleration

$$a = \frac{v_2 - v_1}{t} \quad (4)$$

where:
- a represents the acceleration in kilometers per hour squared (km/h^2),
- v_1 and v_2 are the initial and final velocities respectively,
- t represents the time interval.

Figure 5: Formula for calculating the angular velocity

$$\Delta\text{Term} = \sin^2\left(\frac{\text{Lat2} - \text{Lat1}}{2}\right) + \cos(\text{Lat1}) \times \cos(\text{Lat2}) \times \sin^2\left(\frac{\text{Long2} - \text{Long1}}{2}\right) \quad (5)$$

$$av = \frac{2 \times \text{atan2}\left(\sqrt{\Delta\text{Term}}, \sqrt{1 - \Delta\text{Term}}\right)}{R \times t} \quad (6)$$

where:
- av represents the angular velocity in radians per hour (degrees/h),
- $Lat1$ and $Lat2$ are the initial and final latitudes respectively in radians,
- $Long1$ and $Long2$ are the initial and final longitudes respectively in radians,
- R is the radius of the Earth, taken to be approximately 6,371 km,
- t represents the time interval.

into a pandas DataFrame, and performed preliminary data preprocessing.

The first step was to retrieve the dataset from a CSV file on Google Drive. We then used dictionary mapping to convert categorical variables, such as 'TransportationMode', 'UserCode', and 'TrajectoryCode' into numerical form.

We used a predefined dictionary 'mode_dict' to transform the 'TransportationMode' variable, which served as the target variable. We also converted 'UserCode' and 'TrajectoryCode' into numerical codes to make the dataset suitable for machine learning models. We chose a percentage of the dataset for subsequent analysis to ensure efficient processing.

We separated the dataset into features (X) and labels (y) after the initial preprocessing steps. The 'TransportationMode' column was among the labels, while the other columns were among the features. We used StandardScaler to normalize the features to ensure they were consistent. This step required removing the mean and scaling the features to unit variance, which is a requirement for many machine learning estimators.

To assess the models' performance, we divided the data into training and testing sets using the sklearn train_test_split function. The training set contained 80% of the data, while the test set received the remaining 20%. This division allowed us to evaluate the models' performance on previously unseen data, ensuring a fair evaluation.

In this study, we leveraged multiple machine-learning models to tackle our research objectives. To comprehensively evaluate their performance, we employed a robust set of key metrics, including accuracy, bias, variance, precision, recall, and the F1 score. By considering these metrics, we gained valuable insights into the identifier's efficacy across various modes and ascertained its precision in predicting specific modes.

We used the DecisionTreeClassifier from sklearn to implement the DT model, training it on our data and using it to predict the labels of the test set. Its performance was assessed by comparing predicted and actual labels.

We imported the Logistic Regression classifier from sklearn for the Logistic Regression model, followed the same process as the DT model, and evaluated the model similarly.

The KNN model was built with Sklearn's K-Neighbors Classifier. We used the same evaluation process after fitting the model to the training data to make predictions on the test set.

We defined and built the architecture of the MLP, LSTM, and RNN models using TensorFlow's Keras API. The MLP model had input, hidden, and output layers, with 'relu' as the hidden layer activation function. The LSTM model began with an LSTM layer, while the RNN model began with a SimpleRNN layer. All three models concluded with a Dense layer with 'softmax' as the activation function, which is appropriate for multiclass classification problems.

MLP, LSTM, and RNN models were all built with the 'sparse_categorical_crossentropy' loss function, the 'adam' optimizer, and 'accuracy' as a performance metric. Following training, the models were used to predict test set labels, and their performance was evaluated by comparing these predictions to actual labels.

The performance of the models was assessed using confusion matrices and heatmaps generated with the Seaborn and Matplotlib libraries.

Furthermore, we considered the inherent characteristics and trade-offs of each model when determining their applicability to our specific problem. Decision trees, for example, are interpretable but may struggle with complex patterns, whereas models such as MLP, LSTM, and RNN can capture such patterns but may require more computational resources and time for fine-tuning.

5 Results and Analysis

We used cross-validation to evaluate the effectiveness of the various classifiers used, which included MLP, KNN, Decision Tree, LSTM, RNN, and Logistic Regression. The five-fold cross-validation method was used specifically to provide a reliable estimate of the model's potential performance on unseen data, protecting against overfitting. In addition, we investigated the bias-variance trade-off for each model to better understand its robustness and generalization capabilities.

Six different machine learning algorithms were used in this study to model and predict the multiclass transportation dataset. Several metrics, including accuracy, precision, recall, and the F1-score, were used to evaluate and compare the performance of the models. In analyzing the results,

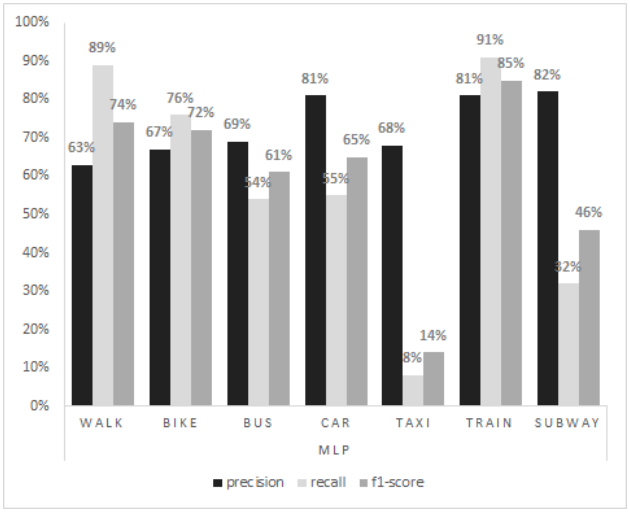

Figure 6: Performance Comparison of Transport Modes with MLP Algorithm

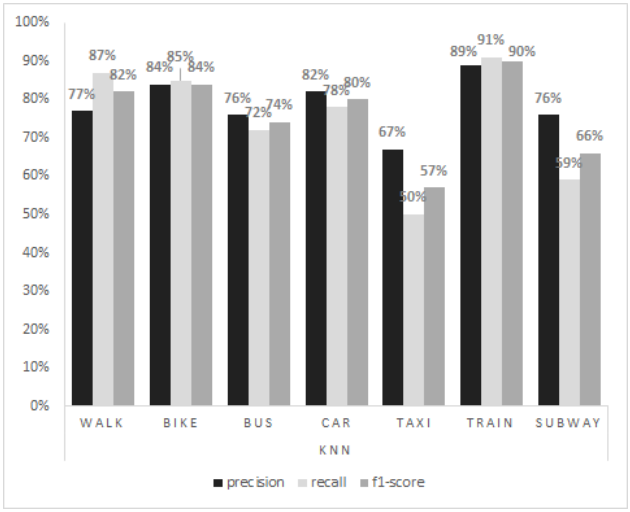

Figure 7: Performance Comparison of Transport Modes with KNN Algorithm

we will look at two perspectives, the first from the Model point of view, and the second from the point of view of the results of the transfer mode.

The MLP model had an overall accuracy of 68.55%. According to the confusion matrix, the model performed best in the 'walk' category, correctly identifying approximately 89% of instances. The 'taxi' and 'subway' categories, on the other hand, had the lowest accuracy, with only about 8% and 32% of instances correctly identified, respectively. This indicates that the model has difficulty distinguishing between these categories. The details of the MLP performance are provided in Table 4 and Figure 6.

The overall accuracy of the KNN model was 79.29%, which ed well in all categories, with the highest accuracy observed in the 'train' category (approximately 91% of instances correctly identified). The model, on the other hand, struggled with the 'taxi' and 'subway' categories, as evidenced by lower recall rates of 50% and 59%, respectively. KNN performance is provided in Table 5 and Figure 7.

The DT model outperformed the previous two algorithms with an overall accuracy of 87.41%. It performed exceptionally well in distinguishing the 'train' category, correctly identifying approximately 96% of instances. Interestingly, this model performed relatively well in the 'taxi' category (approximately 83% correct), a category that presented difficulties for the previous algorithms. The 'bus' category, on the other hand,

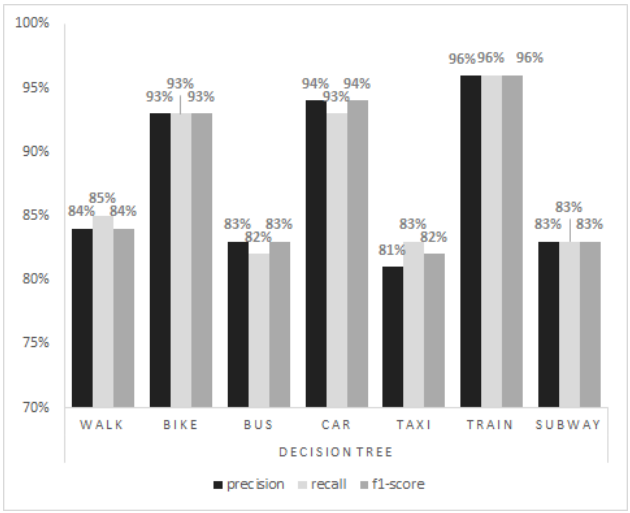

Figure 8: Performance Comparison of Transport Modes with DT Algorithm

had a lower recall rate (82%), indicating some misclassification. The details of the DT model performance, including precision, and recall for each transportation mode, are provided in Table 6 and Figure 8.

The overall accuracy of the LSTM model was 72.46%. The model did well in the 'train' category, with a recall rate of 91%, but struggled in the 'taxi' and 'subway' categories, with recall rates of 24% and 37%, respectively. The details of the LSTM performance, including precision, and recall for each transportation mode, are provided in Table 7 and Figure 9.

The accuracy of the RNN model was 70.86%.

Table 4: Precision, recall, and f1-score for the Multilayer Perceptron (MLP) model

Mode	walk	bike	bus	car	taxi	train	subway	accuracy	precision	recall	support
walk	**268,077**	22,360	8,847	467	1	2	0	89.43%	63%	89%	299,754
bike	30,229	**144,562**	12,078	2,350	34	66	40	76.34%	67%	76%	189,359
bus	78,596	25,694	**136,840**	2,776	603	8,129	1,895	53.76%	69%	54%	254,533
car	21,626	4,906	15,392	**56,029**	879	2,327	978	54.85%	81%	55%	102,137
taxi	10,702	8,664	13,111	2,488	**3,809**	8,683	1,076	7.84%	68%	8%	48,533
train	2,937	667	5,211	1,689	27	**101,754**	81	90.55%	81%	91%	112,366
subway	16,690	7,461	5,498	3,584	253	5,418	**18,090**	31.74%	82%	32%	56,994

Table 5: Precision, recall, and f1-score for the K-Nearest Neighbors (KNN) model

Mode	walk	bike	bus	car	taxi	train	subway	accuracy	precision	recall	support
walk	**259,771**	8,869	23,096	3,076	1,497	385	3,060	86.66%	77%	87%	299,754
bike	14,327	**160,364**	11,116	1,719	787	139	907	84.68%	84%	85%	189,359
bus	40,956	15,132	**183,604**	3,602	3,917	4,798	2,524	72.13%	76%	72%	254,533
car	7,828	2,981	6,503	**79,371**	1,939	746	2,769	77.71%	82%	78%	102,137
taxi	5,431	1,866	8,707	3,392	**24,025**	4,194	918	49.50%	67%	50%	48,533
train	1,514	483	3,817	902	2,403	**102,688**	559	91.38%	89%	91%	112,366
subway	7,754	1,437	5,029	5,291	1,500	2,367	**33,616**	58.98%	76%	59%	56,994

It excelled in the 'train' category, correctly identifying approximately 89% of instances. However, it performed poorly in the 'taxi' and 'subway' categories, with recall rates of 23% and 37%, respectively (8 and Figure 10).

The overall accuracy of the Logistic Regression model was 50.99%. Most categories were difficult for the model to distinguish, particularly 'taxi,' where it failed to correctly identify any instances. Surprisingly, the model performed relatively well in the 'walk' and 'train' categories, correctly identifying approximately 86% and 75% of instances, respectively. The details of the Logistic Regression model performance, including the precision, and recall for each transportation mode, are provided in Table 9 and Figure 11.

According to previous results, the DT model emerges as the optimal choice in a comparative analysis of various machine learning algorithms based on key evaluation metrics, as shown in Table 10, despite a slightly lower accuracy of 85%, as opposed to the highest of 89% manifested by the MLP, LSTM, and RNN algorithms. The DT model is superior because it has the lowest

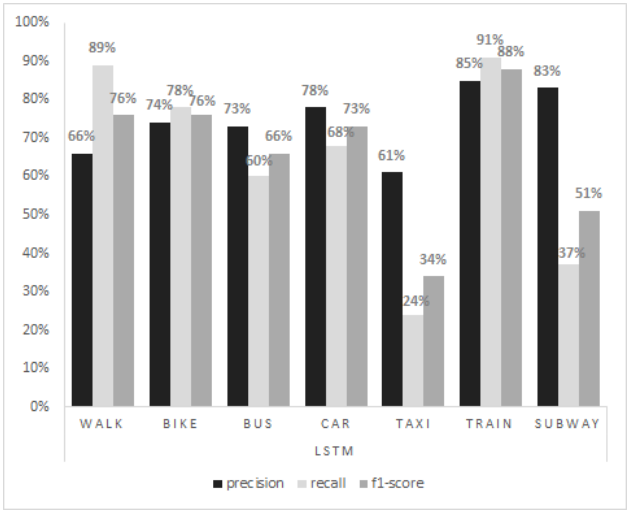

Figure 9: Performance Comparison of Transport Modes with LSTM Algorithm

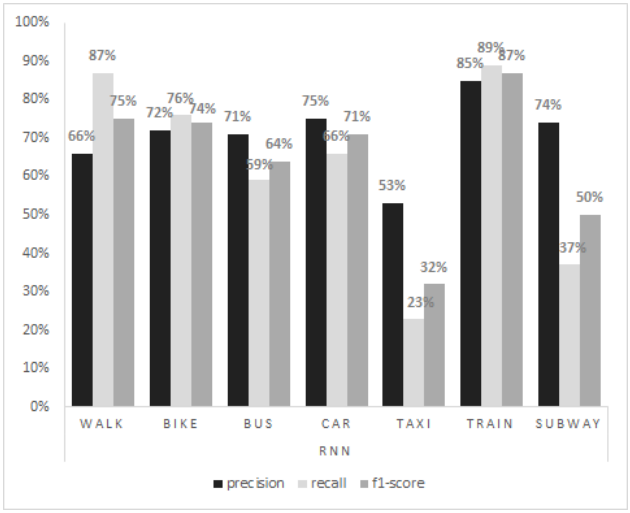

Figure 10: Performance Comparison of Transport Modes with RNN Algorithm

Table 6: Precision, recall, and f1-score for the DT model

Mode	walk	bike	bus	car	taxi	train	subway	accuracy	precision	recall	support
walk	**254,616**	5,355	29,470	2,248	3,936	862	3,267	84.94%	84%	85%	299,754
bike	5,840	**175,546**	6,088	420	327	101	1,037	92.70%	93%	93%	189,359
bus	32,520	6,044	**209,397**	1,167	2,150	1,446	1,809	82.26%	82.26%	83%	254,533
car	2,811	457	1,208	**94,927**	569	279	1,886	92.94%	94%	93%	102,137
taxi	2,910	320	2,090	592	**40,355**	1,362	904	83.15%	81%	83%	48,533
train	785	78	1,515	235	1,407	**107,869**	477	96.00%	96%	96%	112,366
subway	4,987	384	1,926	1,088	1,058	456	**47,095**	82.63%	83%	83%	56,994

Table 7: Precision, recall, and f1-score for the Long Short-Term Memory (LSTM) model

Mode	walk	bike	bus	car	taxi	train	subway	accuracy	precision	recall	support
walk	**266,177**	15,745	13,846	3,686	196	25	79	88.38%	66%	89%	299,754
bike	24,629	**147,263**	14,329	2,804	155	72	107	78.91%	74%	78%	189,359
bus	69,971	17,356	**152,815**	2,622	3,002	6,413	2,354	59.47%	73%	60%	254,533
car	14,940	3,683	9,632	**69,781**	2,060	1,062	979	65.50%	78%	68%	102,137
taxi	7,902	8,769	9,620	3,023	**11,472**	7,033	714	22.60%	61%	24%	48,533
train	2,110	695	4,137	1,560	1,267	**102,408**	189	91.42%	85%	91%	112,366
subway	15,880	5,364	4,447	5,799	717	3,935	**20,852**	36.18%	83%	37%	56,994

recorded bias of 16% and the lowest competitive variance of 15%. These indicators point to improved model robustness in comparison to its counterparts, reducing the risk of overfitting or underfitting.

Importantly, in the context of Table 10, the DT model has 84% precision, indicating a lower probability of false-positive instances. At the same time, it maintains a commendable recall rate of 85%, indicating its effectiveness in identifying true positives. Furthermore, the DT algorithm's F1-score, which represents the harmonic mean of precision and recall, peaks at 84%, indicating a desirable balance between these two critical parameters. As a result of the comprehensive evaluation, the DT model provides the most advantageous trade-off among accuracy, bias-variance equilibrium, precision, recall, and F1-score, as shown in Table 10.

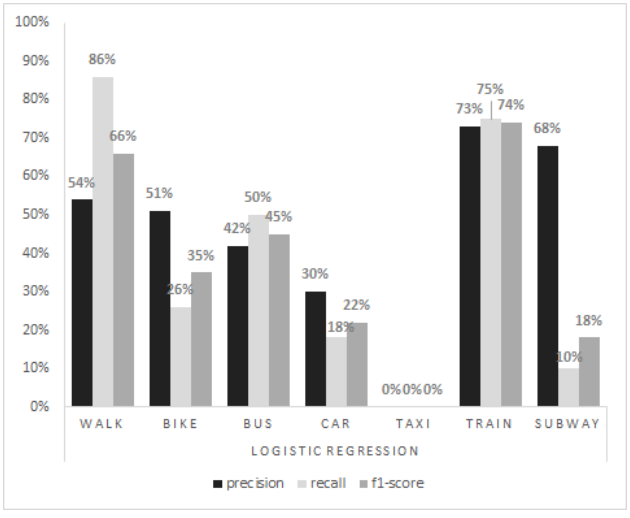

Figure 11: Performance Comparison of Transport Modes with Logistic Regression Algorithm

Bike Travel Mode Given the data in Table 11, which compares various machine learning algorithms for predicting bike travel mode, it is clear that the DT model outperforms the others. It achieves the highest accuracy of 93%, a significant advantage over the second best, the KNN algorithm, which achieves 85%. Furthermore, the DT model is exceptionally stable, with the lowest recorded bias and variance, both of which are 7%. This implies that this model is less prone to overfitting or underfitting, which improves its overall reliability in the context of bike travel mode prediction.

The DT model outperforms all other models in terms of precision, recall, and F1-score, which are critical measures in determining a model's effectiveness at accurately predicting true positives and its balance of false positives and true positives.

Therefore, based on the comprehensive evaluation presented in Table 11, the DT model appears to provide the most beneficial trade-off among accuracy, bias-variance equilibrium, precision, re-

Table 8: Precision, recall, and f1-score for the Recurrent Neural Network (RNN) model

Mode	walk	bike	bus	car	taxi	train	subway	accuracy	precision	recall	support
walk	**260,588**	16,829	17,572	4,395	108	79	183	89.40%	66%	87%	299,754
bike	26,734	**143,799**	14,995	3,059	382	46	344	73.72%	72%	76%	189,359
bus	67,050	20,776	**149,221**	4,603	3,873	5,868	3,142	57.70%	71%	59%	254,533
car	14,682	4,133	10,155	**67,852**	2,806	722	1,787	63.52%	75%	66%	102,137
taxi	8,712	9,012	9,222	2,908	**11,126**	6,565	988	19.56%	53%	23%	48,533
train	2,227	722	4,992	1,886	1,790	**99,845**	904	89.99%	85%	89%	112,366
subway	15,791	5,225	4,380	5,554	752	3,971	**21,321**	35.65%	74%	37%	56,994

Table 9: precision, recall, and f1-score for the Logistic Regression model

Mode	walk	bike	bus	car	taxi	train	subway	accuracy	precision	recall	support
walk	**258,048**	21,357	20,290	54	0	5	0	86.08%	54%	86%	299,754
bike	69,838	**49,492**	66,581	3,425	0	23	0	26.13%	51%	26%	189,359
bus	84,460	20,941	**126,863**	11,361	1	8,738	2,169	49.84%	42%	50%	254,533
car	26,056	1,732	42,307	**17,882**	0	13,679	481	17.50%	30%	18%	102,137
taxi	15,711	1,152	17,547	10,610	0	3,422	91	0%	0%	0%	48,533
train	2,292	256	15,711	9,867	0	**84,233**	7	74.96%	73%	75%	112,366
subway	21,909	1,626	16,115	6,050	0	5,345	**5,949**	10.43%	68%	10%	56,994

Table 10: Performance of Various Machine Learning Algorithms for Walk Mode Prediction

Algorithm	Accuracy	Bias	Variance	Precision	Recall	F1-score
MLP	89%	38%	9%	63%	89%	74%
KNN	87%	23%	13%	77%	87%	82%
DT	85%	16%	15%	84%	85%	84%
LSTM	88%	34%	12%	66%	89%	76%
RNN	89%	35%	11%	66%	87%	75%
Logistic Regression	86%	46%	14%	54%	86%	66%

call, and F1-score in the context of predicting bike travel mode.

Bus Travel Mode As presented in Table 12, the DT model outperforms the other machine learning algorithms considered in the domain of bus travel mode prediction. With an accuracy rate of 82%, it significantly surpasses the second-best performer, KNN, which achieves 72% accuracy. The DT model demonstrates remarkable robustness, evident in its lowest recorded bias of 17% and equally commendable variance rate of 18%. Additionally, the model excels in precision, recall, and F1-score, all measuring at 83%. These results underscore the model's superior ability to predict true positives and effectively balance false positives and true positives.

Therefore, based on the comprehensive evaluation presented in Table 12, the DT model emerges as the optimal choice for bus travel mode prediction, offering the best trade-off between accuracy, bias-variance balance, precision, recall, and F1-score.

Car Travel Mode Based on the data presented in Table 13 for car travel mode prediction, the DT model once again demonstrates exceptional performance compared to the other evaluated machine learning models. With an accuracy rate of 93%, it significantly outperforms the second-best model, KNN, which achieves an accuracy rate of 78%. The robustness of the DT model is further highlighted by its minimal bias of 6% and remarkably low variance of 7%, indicating a reduced likelihood of overfitting or underfitting and enhancing the algorithm's overall reliability.

Furthermore, the DT model excels in precision, recall, and F1-score, achieving a score of 94% in each category. This reflects its superior ability to accurately predict true positives while maintaining a balanced proportion of false positives. In conclusion, the comprehensive evaluation presented in Table 13 solidifies the DT model as the optimal choice for car travel mode prediction, pro-

Table 11: Performance of Various Machine Learning Algorithms for Bike Mode Prediction

Algorithm	Accuracy	Bias	Variance	Precision	Recall	F1-score
MLP	76%	30%	25%	67%	76%	72%
KNN	85%	16%	15%	84%	85%	84%
DT	93%	7%	7%	93%	93%	93%
LSTM	79%	27%	21%	74%	78%	76%
RNN	74%	27%	26%	72%	76%	74%
Logistic Regression	26%	49%	74%	51%	26%	35%

Table 12: Performance of Various Machine Learning Algorithms for Bus Mode Prediction

Algorithm	Accuracy	Bias	Variance	Precision	Recall	F1-score
MLP	54%	33%	45%	69%	54%	61%
KNN	72%	24%	28%	76%	72%	74%
DT	82%	17%	18%	83%	82%	83%
LSTM	59%	26%	41%	73%	60%	66%
RNN	58%	30%	42%	71%	59%	64%
Logistic Regression	50%	58%	50%	42%	50%	45%

viding the most favorable trade-off among accuracy, bias-variance balance, precision, recall, and F1-score.

Subway Travel Mode As shown in Table 14, the DT model outperforms the other machine learning algorithms under consideration for predicting subway travel mode. It has an astounding accuracy rate of 83%, far outperforming the second-best-performing algorithm, KNN, which has an accuracy rate of 59%. The DT algorithm's bias and variance scores of 17% further demonstrate its robustness. These results indicate that the model has an impressive robustness that reduces the likelihood of overfitting or underfitting, thereby increasing its reliability for this prediction task.

Furthermore, the DT model outperforms in terms of precision, recall, and F1-score, scoring 83% across these three critical evaluation metrics. These results demonstrate the model's exceptional ability to identify true positives while maintaining a balanced proportion of false positives. As a result of the comprehensive evaluation in Table 14, the DT model offers the best trade-off among accuracy, bias-variance balance, precision, recall, and F1-score, emerging as the best choice for subway travel mode prediction.

Taxi Travel Mode According to the analysis of the data in Table 15 for taxi travel mode prediction, the DT model outperforms all other evaluated machine learning models significantly. The DT model has an accuracy rate of 83%, which is significantly higher than the next most accurate model, KNN, which has a rate of 50%. The DT model's bias and variance rates, both less than 20%, highlight its exceptional robustness, implying a lower proclivity for overfitting or underfitting and thus contributing to overall model reliability.

Furthermore, the DT model performs admirably in terms of precision, recall, and F1-score, with scores of 81%, 83%, and 82%, respectively. These metrics represent the model's superior ability to correctly identify true positives while maintaining a desirable balance of true and false positives. As a result of the comprehensive evaluation presented in Table 15, the DT model provides the best trade-off between accuracy, bias-variance balance, precision, recall, and F1-score, positioning itself as the best choice for taxi travel mode prediction.

Train Travel Mode According to Table 16, which compares machine learning models for predicting train travel mode, the DT model is superior. The DT model has the highest accuracy of 96%, outperforming the MLP, KNN, LSTM, and RNN models, all of which have accuracies in the lower nineties. The DT model also demonstrates superior robustness, with a recorded bias of 4% and a variance rate of 4%, implying less susceptibility to overfitting or underfitting and thus increasing its reliability for the prediction task.

Table 13: Performance of Various Machine Learning Algorithms for Car Mode Prediction

Algorithm	**Accuracy**	**Bias**	**Variance**	**Precision**	**Recall**	**F1-score**
MLP	55%	17%	55%	81%	55%	65%
KNN	78%	18%	22%	82%	78%	80%
DT	93%	6%	7%	94%	93%	94%
LSTM	66%	21%	34%	78%	68%	73%
RNN	64%	25%	36%	75%	66%	71%
Logistic Regression	18%	70%	82%	30%	18%	22%

Table 14: Performance of Various Machine Learning Algorithms for Subway Mode Prediction.

Algorithm	**Accuracy**	**Bias**	**Variance**	**Precision**	**Recall**	**F1-score**
MLP	32%	18%	69%	82%	32%	46%
KNN	59%	24%	41%	76%	59%	66%
DT	83%	17%	17%	83%	83%	83%
LSTM	36%	18%	64%	83%	37%	51%
RNN	36%	23%	64%	74%	37%	50%
Logistic Regression	10%	32%	90%	68%	10%	18%

Table 15: Performance of Various Machine Learning Algorithms for Taxi Mode Prediction

Algorithm	**Accuracy**	**Bias**	**Variance**	**Precision**	**Recall**	**F1-score**
MLP	8%	37%	90%	68%	8%	14%
KNN	50%	33%	50%	67%	50%	57%
DT	83%	19%	17%	81%	83%	82%
LSTM	23%	39%	77%	61%	24%	34%
RNN	20%	41%	80%	53%	23%	32%
Logistic Regression	0%	100%	100%	0%	0%	0%

The DT model outperforms its competitors in terms of precision, recall, and F1-score, scoring 96% across all three metrics. These scores represent not only the model's ability to identify true positives accurately but also its effectiveness in maintaining a balance between true positives and false positives. As a result of the comprehensive evaluation in Table 16, the DT model can be considered as the best choice for predicting train travel mode.

6 Conclusion

The extensive evaluation of six machine learning algorithms for predicting multi-class transportation modes (MLP, KNN, Decision Tree, LSTM, RNN, and Logistic Regression) revealed distinct results. The Decision Tree model was found to be the most robust model across all modes of transportation, including walking, bike, bus, car, subway, taxi, and train. It had the best overall accuracy and consistently achieved high precision, recall, and F1 scores.

The DT algorithm's superior performance can be attributed to inherent strengths such as simplicity, interpretability, and adaptability, which allow it to effectively handle multi-class prediction problems. These findings highlight the importance of using DT algorithms for transportation mode prediction tasks, and they provide valuable insights for researchers and practitioners in this field.

Despite the encouraging results, it should be noted that the performance of machine learning algorithms is dependent on the specific nature of the dataset, implying that different datasets may yield different results. Future research could look into potential improvements to the other algorithms under consideration, as well as the performance of other deep learning models. Furthermore, testing these models with a broader range of transportation scenarios and datasets, possi-

Table 16: Performance of Various Machine Learning Algorithms for Train Mode Prediction.

Algorithm	Accuracy	Bias	Variance	Precision	Recall	F1-score
MLP	91%	21%	9%	81%	91%	85%
KNN	91%	11%	9%	89%	91%	90%
DT	96%	4%	4%	96%	96%	96%
LSTM	91%	17%	9%	85%	91%	88%
RNN	90%	16%	10%	85%	89%	87%
Logistic Regression	75%	27%	25%	73%	75%	74%

bly incorporating real-world factors such as traffic conditions or weather patterns, would enrich and broaden the study's findings and implications. Furthermore, investigating hybrid models that combine the strengths of multiple algorithms could be a promising direction for future research.

References

[1] J. Zeng, Y. Yu, Y. Chen, D. Yang, L. Zhang, and D. Wang, "Trajectory-as-a-sequence: A novel travel mode identification framework," *Transportation Research Part C: Emerging Technologies*, vol. 146, p. 103957, 2023.

[2] M. Shulajkovska, G. Noveski, M. Smerkol, J. Grabnar, E. Dovgan, and M. Gams, "Eu smart cities: Towards a new framework of urban digital transformation," *Informatica*, vol. 47, no. 2, 2023.

[3] H. Mäenpää, A. Lobov, and J. L. Martinez Lastra, "Travel mode estimation for multi-modal journey planner," *Transportation Research Part C: Emerging Technologies*, vol. 82, pp. 273–289, 2017.

[4] C. L. Hasif, A. Araldo, S. Dumbrava, and D. Watel, "A graph-database approach to assess the impact of demand-responsive services on public transit accessibility," in *Proceedings of the 15th ACM SIGSPATIAL International Workshop on Computational Transportation Science*, pp. 1–4, 2022.

[5] C. Hoffmann, C. Abraham, M. P. White, S. Ball, and S. M. Skippon, "What cognitive mechanisms predict travel mode choice? a systematic review with meta-analysis," *Transport Reviews*, vol. 37, no. 5, pp. 631–652, 2017.

[6] O. Bagdadi and A. Várhelyi, "Development of a method for detecting jerks in safety critical events," *Accident Analysis & Prevention*, vol. 50, pp. 83–91, 2013.

[7] T. Phiophuead and N. Kunsuwan, "Logistic regression analysis of factors affecting travel mode choice for disaster evacuation," *Engineering Journal*, vol. 23, no. 6, pp. 399–417, 2019.

[8] Y. Zheng, Q. Li, Y. Chen, X. Xie, and W.-Y. Ma, "Understanding mobility based on gps data," in *Proceedings of the 10th international conference on Ubiquitous computing*, pp. 312–321, 2008.

[9] S. Ding, Z. Li, K. Zhang, and F. Mao, "A comparative study of frequent pattern mining with trajectory data," *Sensors*, vol. 22, no. 19, p. 7608, 2022.

[10] S. Dabiri and K. Heaslip, "Inferring transportation modes from gps trajectories using a convolutional neural network," *Transportation Research Part C: Emerging Technologies*, vol. 86, pp. 360–371, 2018.

[11] J. J. Q. Yu, "Travel mode identification with gps trajectories using wavelet transform and deep learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 2, pp. 1093–1103, 2021.

[12] Y. Zhu, Y. Liu, J. J. Q. Yu, and X. Yuan, "Semi-supervised federated learning for travel mode identification from gps trajectories," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 3, pp. 2380–2391, 2022.

[13] J. Kim, J. H. Kim, and G. Lee, "Gps data-based mobility mode inference model using

long-term recurrent convolutional networks," *Transportation Research Part C: Emerging Technologies*, vol. 135, p. 103523, 2022.

[14] C. Zheng, C. Wang, X. Fan, J. Qi, and X. Yan, "Stpc-net: Learn massive geo-sensory data as spatio-temporal point clouds," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 11314–11324, 2022.

[15] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, "Mining interesting locations and travel sequences from gps trajectories," in *Proceedings of the 18th international conference on World wide web*, pp. 791–800, 2009.