mathematics

MDPI

Article

Dynamic Generation Method of Highway ETC Gantry Topology Based on LightGBM

Fumin Zou [1], Weihai Wang [1], Qiqin Cai [1,2,*], Feng Guo [1] and Rouyue Shi [1]

[1] Fujian Key Laboratory for Automotive Electronics and Electric Drive, Fujian University of Technology, Fuzhou 350118, China; fmzou@fjut.edu.cn (F.Z.); 2211308003@smail.fjut.edu.cn (W.W.); n180310004@fzu.edu.cn (F.G.); 2211301008@smail.fjut.edu.cn (R.S.)

[2] School of Mechanical Engineering and Automation, Huaqiao University, Xiamen 361021, China

* Correspondence: 20011080002@stu.hqu.edu.cn; Tel.: +86-181-4405-6202

Abstract: In Electronic Toll Collection (ETC) systems, accurate gantry topology data are crucial for fair and efficient toll collection. Currently, inaccuracies in the topology data can cause tolls to be based on the shortest route rather than the actual distance travelled, contradicting the ETC system's purpose. To address this, we adopt a novel Gradient Boosting Decision Tree (GBDT) algorithm, Light Gradient Boosting Machine (LightGBM), to dynamically update ETC gantry topology data on highways. We use ETC gantry and toll booth transaction data from a province in southeast China, where ETC usage is high at 72.8%. From this data, we generate a candidate topology set and extract five key characteristics. We then use Amap API and QGIS map analysis to annotate the candidate set, and, finally, apply LightGBM to train on these features, generating the dynamic topology. Our comparison of LightGBM with 14 other machine learning algorithms showed that LightGBM outperformed the others, achieving an impressive accuracy of 97.6%. This methodology can help transportation departments maintain accurate and up-to-date toll systems, reducing errors and improving efficiency.

Keywords: highway; ETC gantry; topology dynamic generation; LightGBM

MSC: 68T09

Citation: Zou, F.; Wang, W.; Cai, Q.; Guo, F.; Shi, R. Dynamic Generation Method of Highway ETC Gantry Topology Based on LightGBM. *Mathematics* **2023**, *11*, 3413. https://doi.org/10.3390/math11153413

Academic Editor: Alicia Cordero

Received: 11 July 2023
Revised: 30 July 2023
Accepted: 3 August 2023
Published: 4 August 2023

1. Introduction

Highways play a pivotal role in the modern transportation system, offering convenient transit services that fuel economic growth across various countries and regions. With societal advancement and technological evolution, toll collection methods on these highways are continually being modernized. Among these is the Electronic Toll Collection (ETC) system, which is renowned for its efficiency and which has been increasingly adopted across nations and regions. However, during practical implementation, the ETC system still faces numerous challenges, such as toll-related issues stemming from inaccurate topological data. This research aims to address this issue, proposing a method for the dynamic updating of the topological data of highway ETC gantries.

Serving as a pivotal piece of infrastructure within the realm of transportation, the highway gantry system gleans data via the On-Board Unit (OBU) and Roadside Unit (RSU) devices. The OBU, an electronic device mounted on vehicles, functions to communicate with the highway toll collection system. In contrast, the RSU, situated at highway toll booths or gantries, interacts with the OBU, facilitating automatic toll collection. The objective behind the installation of gantries is to compute the actual mileage traversed by vehicles. The gantry system operates by scanning the OBU mounted on vehicles, logging the vehicle's travel distance and time. By calculating the length of each gantry topological path that the vehicle traverses on the highway and the per-kilometer travel cost, the system determines the appropriate toll. According to the requirements of the "Implementation Plan for the

Full Promotion of Differentiated Toll Collection on Highways", China is actively promoting the reform of differentiated toll collection on highways to further improve the efficiency and service level of the highway network, reduce the cost of highway travel, and promote cost reduction and efficiency improvement in the logistics industry. To this end, China is implementing a series of strategies, including thoroughly summarizing the experience of differentiated toll collection pilot projects on highways, fully considering factors such as the structure and operating characteristics of local road networks, choosing suitable differentiated toll collection methods, innovating service models, and scientifically and accurately formulating differentiated toll collection schemes [1]. In addition, differentiated toll collection is being implemented by road section, mainly implementing flexible and diverse differentiated tolls on road sections where ordinary national and provincial trunk roads or urban roads are severely congested but parallel highways have small traffic flow, road sections where the traffic volume differs greatly between parallel highways, and road sections where the traffic volume is significantly lower than the design capacity. Although there have been some studies on the toll collection of highways, there are few studies on how to support these differentiated toll collection strategies better through efficient and accurate dynamic updates of topological data. In the domain of ETC gantry topology generation research, Cai and Yi et al. [2] introduced a pioneering approach known as the "Arch-Bridge topology." This method entails the generation of an initial topology candidate set through the analysis of discrete ETC data, followed by the identification and examination of abnormal topology features. Subsequently, the Dijkstra algorithm is applied to optimize the topology, resulting in a comprehensive ETC gantry topology.

A noteworthy limitation of their approach lies in their static nature, as they lack the capability for dynamic updates. Unlike our proposed methodology, which enables the real-time adjustment and refinement of gantry topology based on dynamically changing road network conditions, the Arch-Bridge topology is confined to performing solely static updates without considering real-time traffic variations. However, the topological information of highway ETC gantries is far from static. For instance, the addition or decommissioning of roads may lead to permanent changes in gantry topology, while road maintenance or unexpected events may cause temporary changes. This can result in inaccuracies in the topological data within some vehicle transaction data, thus making it impossible to calculate tolls based on the distance traveled. Instead, tolls are charged based on the shortest distance. This method of toll collection can lead to many problems. Firstly, it could result in financial losses, as drivers might be required to pay for distances that are longer or shorter than the actual distance traveled. Charging more is unfair to commercial vehicles that frequently use highways, and charging less can also lead to financial losses for ETC administrators. This method of toll collection deviates from the original intention of the ETC system, which is to provide drivers with more convenient, faster, and more accurate toll services. Secondly, the research on ETC gantry topology can provide richer application scenarios for intelligent transport systems. For example, based on the topological information of the ETC gantry, more refined applications, such as traffic condition prediction, congestion detection, and route planning, can be developed to provide drivers with more convenient travel services. At the same time, it can also provide optimization solutions for the fields of public transportation and logistics, reduce operational costs, and improve transportation efficiency.

To address the current issues in the dynamic update of highway ETC gantry topology, and to enhance the quality and efficiency of intelligent highway services, this paper proposes a method for dynamically generating highway ETC gantry topology based on LightGBM. Specifically, the method first generates a candidate set of gantry topology using the highway ETC transaction data of a province in China with a high ETC usage rate of 72.8% [3], extracts the candidate topology, and filters out incorrect topology. Then, five feature dimensions are extracted from each topology, including Topology Traffic Volume (*TTV*), Topological Passage Rate (*TPR*), Normalized Start Rate (*NSR*), Normalized End Rate (*NER*), and Topology Distance (*TD*). Next, by combining the Amap API [4] and QGIS [5]

map analysis, the candidate topology set is marked, with real existing topology marked as 1 and non-existing topology marked as 0. In this case, the dynamic generation of gantry topology can be transformed into a typical binary classification problem in machine learning. This study uses supervised learning algorithms (such as SVM, Naive Bayes, Logistic Regression, etc.), tree model algorithms (such as LightGBM, RF, XGBoost, etc.), and ensemble learning algorithms (such as AdaBoost, Bagging, etc.) for the dynamic generation of gantry topology.

Our research makes the following key contributions:

- Introduction of an innovative methodology for dynamic updating of highway gantry topology based on ETC transaction data.
- Rectification of prevalent inaccuracies in topology data within vehicle transaction records, leading to more accurate fee computation for actual traversed distances.
- Utilization of the LightGBM model to facilitate dynamic updating of the gantry topology with an impressive accuracy rate of 97.6%.
- Universally applicable methodology and framework for dynamically updating highway ETC gantry topology, demonstrating extensive applicability and scalability.

The remainder of this paper is structured as follows. An exhaustive literature review graces Section 2. In Section 3, we expound upon the methodologies pertaining to data preprocessing and extraction of candidate topology sets, as well as dynamic generation techniques for gantry topology. Experiments along with a comprehensive analysis of the results form the substance of Section 4. Lastly, a cogent summary encapsulating the novelties and constraints of our research, along with a glance into future possibilities, concludes our discourse.

2. Literature Review

Within our research domain, which involves the dynamic generation of ETC gantry topology, there is currently a lack of directly related work due to the innovative and unique nature of this issue. As a result, we often draw upon methods and theories from other disciplines to inspire new perspectives and identify novel solutions. In this context, we have consulted related literature from fields such as "remote sensing image recognition methods" and "spatio-temporal trajectory data mining." Both of these areas have extensive research experience and significant achievements in dealing with large-scale, complex, and dynamic data. Their methodologies provide us with insightful perspectives. The inspiration drawn from their methods guides our direction and shapes our solutions, thereby enriching the results of our research.

2.1. Remote Sensing Image Recognition Method

Road network generation methods from remote sensing imagery can be broadly categorized as: (1) image-segmentation-based, (2) feature-extraction-based, and (3) methods reliant on machine learning and deep learning approaches.

Image segmentation involves dissecting remote sensing images into various regions, each comprising pixels with similar characteristics, such as color, intensity, or texture. After segmentation, the recognized road regions can be interconnected to formulate the road network. The Snake model, as proposed by Kass et al. [6], signifies a pivotal milestone. In their model, an energy function was defined on the image contour sketched by the user, then adjusted iteratively to converge the image contour at the minimum energy. Despite requiring human intervention, this method effectively addresses noise and large gaps in road images. This approach was further adopted in remote sensing imagery by Péteri and Ranchin [7] and Laptev et al. [8] with their multi-resolution Snake models for road extraction. Subsequent research, like the work of Gruen and Li [9], amalgamated a semi-automatic road extraction strategy utilizing wavelet decomposition for road sharpening alongside a model-driven linear feature extraction method based on dynamic programming. These methods could tackle more intricate road networks. While requiring human intervention, these methods successfully dealt with noise and large gaps in road images.

Regarding feature-extraction-based road recognition, these techniques identify road regions and generate the road network by extracting features like texture, edges, and color information from remote sensing images. For instance, Mokhtarzadeh and Zoej [10] employed artificial neural networks for the detection of roads within high-resolution satellite images, verifying the impact of disparate input parameters on the network's capability, thereby determining the superior network architecture. Moreover, Yager and Sowmya [11] were the trailblazers in utilizing support vector machines(SVM) to extract roads from remote sensing imagery based on road edges. While this technique boasts commendable integrity, its accuracy remains relatively inferior. The methods of feature extraction are incredibly sensitive to image quality and feature selection, with inappropriate features potentially resulting in imprecise identification. Although this methodology can utilize the abundant feature information within remote sensing imagery, it may encounter difficulties with low-resolution or subpar quality images.

In recent years, methods anchored in machine learning and deep learning have garnered attention. These methodologies train models to recognize and extract road regions, creating the road network using machine learning and deep learning algorithms. Deep learning has shown marked effectiveness in handling complex tasks, especially in remote sensing image processing. Mnih and Hinton [12] were the pioneers in applying deep learning techniques to extract road information from high-resolution aerial photographs. He et al. [13] blazed a new trail for training deep neural networks to tackle the vanishing gradient problem, proposing a deep residual learning framework and integrating identity mapping to streamline the training process. In addition, Saito et al. [14] successfully recognized roads and buildings in raw remote sensing imagery in an innovative manner, achieving commendable results on a road dataset from Massachusetts. The fruitful application of Fully Convolutional Networks (FCN) [15] in the semantic segmentation of high-resolution remote sensing imagery [16] is also noteworthy. Deep learning has also played an indispensable role in a myriad of complex remote sensing image tasks, such as automatic object detection [17], semantic labeling of satellite imagery [18], and image classification [19]. U-Net has also shown significant progress in road extraction studies. Inspired by U-Net's successful application in medical image processing, the U-Net model designed by Keramitsoglou et al. [20] has substantially enhanced the handling of complex road networks for road extraction tasks. However, these methodologies still demand a high quality and quantity of training data and computational resources, posing a challenge that has yet to be resolved.

2.2. Spatio-Temporal Trajectory Data Mining

In recent years, with the growing prevalence of vehicle GPS data, numerous researchers have begun to exploit these real-time and precise data to generate maps and road network models. The primary methodologies include: (1) Cluster-Based Approach, (2) Kernel Density Estimation (KDE), and (3) Intersection Linking.

The technique of cluster analysis, which includes clustering algorithms like K-means, is utilized in interpreting GPS sample points, consequently extracting basic points and edges of the road map. These elements are then assembled to create a road network. Initial explorations by Wagstaff and his colleagues [21] demonstrated the potential of this technique for identifying lanes from low-precision GPS data. They later designed a spatial clustering algorithm that independently inferred the map's connectivity structure sans initial inputs, thereby augmenting the process of lane division and merging [22]. In a mining environment devoid of clear road boundaries or lane markings, Worrall S and Nebot E [23] successfully adapted clustering algorithms, illustrating the method's wide-ranging utility and resilience. DBSCAN-centric point clustering techniques have also demonstrated their significance in map matching studies [24]. Edelkamp and Schrodl [25] took the pioneering step of employing K-means clustering on the position of trajectory samples. Each cluster's center was considered a road node, and the subsequent connection of these nodes resulted in a comprehensive road network. Chen C and his team [26] introduced "Traj-Meanshift",

a noise reduction algorithm and a graph-oriented road segment clustering algorithm specifically for de-noising GPS data points to attain precise information processing. They further proposed a new graph-based road segment clustering algorithm that capitalized on prior knowledge of road smoothness to boost the precision and effectiveness of clustering. Huang J and his group [27] employed an ASCDT-based spatial clustering technique and assimilated spatial semantic data to construct roads and their topological relationships. Cluster analysis techniques are adept at handling vast quantities of GPS data without the need for intricate preprocessing. However, the outcome's quality is contingent upon the density and quality of the GPS data. Sparse or sub-standard data might compromise the accuracy of the generated road network.

Kernel Density Estimation (KDE) [28], a principal technique frequently enlisted for spatial data scrutiny and visualization, fundamentally aims at decoding and prognosticating potential event paradigms. In contemporary research, KDE has been synergistically harnessed in the realm of road network generation. Initially, KDE transmutes individual instances or trajectories into a discretized graphic representation, reflecting the density of samples or segments per pixel entity. It implements binary thresholds to actualize road binary illustrations within certain regions, thereby discovering central arterial lines of roads through an array of methodologies such as Voronoi segmentation. Researchers such as Fu Z and his colleagues [29] have successfully constructed an efficacious road network, circumventing the necessity for auxiliary parameter amendments, by employing kernel density analysis, Hidden Markov Models, and map matching techniques. Uduwaragoda E and his team [30] utilized non-parametric Kernel Density Estimation (KDE) to scrutinize the probability density distribution of trajectory nodes, subsequently generating geospatial representations containing lane centerlines. Similarly, Kuntzsch C and others [31] integrated heuristic methodologies with generative modeling, employing KDE to reconstruct an optimized rendition of road maps. Neuhold R and collaborators [32] utilized KDE to process low-precision GPS data, thereby accurately identifying lane centerlines for various road categorizations. KDE, by estimating the location and morphology of roads based on GPS data density, demonstrates an inherent adaptability to a broad spectrum of road types and structures. Furthermore, the KDE methodology is not contingent on a specific clustering algorithm, thus affording substantial flexibility during the processing of GPS data. Nonetheless, the KDE approach necessitates the selection of appropriate kernel functions and bandwidth parameters, which could require profound technical knowledge. Moreover, if the GPS data exhibit a heterogenous distribution, the precision of the KDE output may be compromised.

The Intersection Connection approach begins by identifying vertices at road intersections, subsequently establishing connections between these vertices and edges based on trajectory characteristics to detect road junctions. By interpolating the geometric configurations of trajectories, intersections are interconnected, thus creating and updating road networks. Huang Y and his team [33] deployed a priori knowledge regarding intersection typologies and turn restrictions for the detection of road segments. Deng and colleagues [34] proposed a clustering methodology predicated on hotspot analysis and Delaunay triangulation aimed at spatial coverage detection of road intersections. The accuracy of intersection detection was enhanced by the generation of structural models via K-segmentfit and common subsequence amalgamation. Wu J and others [35] proposed an intersection recognition mechanism founded on an augmented X-means algorithm, successfully implemented for the identification of road network intersections in Shenyang, Liaoning Province, China. Xie X and colleagues [36] introduced a novel intersection definition, delineating intersections as loci connecting three or more different directional road segments, and employed the Longest Common Subsequence (LCSS) for intersection detection under this definition. They managed to effectively identify intersections by discerning common sub-trajectories of multiple GPS trajectories. Fathi and Krumm [37] were among the pioneers who utilized Intersection Connection techniques for cartographic construction, revolutionizing the field with their groundbreaking research and attracting further scholarly

attention and investigation. In subsequent research, they incorporated image processing techniques [35], performing skeletonization on binary trajectory matrices and conducting local "sub-path" detection. Finally, they employed Kernel Density Estimation (KDE) for the identification of road intersections, thereby offering a novel toolkit for intersection recognition and analysis. Karagiorgou and Pfoser [38] devised a heuristic algorithm capable of "bundling" trajectories in the vicinity of intersection nodes, thereby connecting these trajectories. Pu M and others [39] proposed a novel bi-stage road intersection detection framework dubbed RIDF composed of trajectory quality enhancement and intersection extraction. Zhao L and his team [40] pioneered calibration of road intersection impact area topology by introducing a tri-stage calibration framework, denoted as CITT. Qing R and colleagues [41] proposed a GPS trajectory-based road intersection detection methodology that leverages temporal–spatial feature extraction and their interactions to amplify the accuracy of intersection detection. Liu Y and his team [42] utilized the (xDeepFM) model to extract geometric and spatial features from GPS data, and integrated density-based spatial clustering of applications with noise (DBSCAN) and Delaunay triangulation for cluster and intersection radius computations, thereby enhancing the accuracy of road intersection recognition. The intersection connection method is adept at handling extensive GPS data and adapts well to dynamically changing road network environments. However, its reliance on intersection detection could be problematic if the GPS data quality is poor or if the data are sparse. Furthermore, for rural or mountainous roads with no distinct intersections, the intersection connection method might struggle to generate accurate road networks.

In a recent research endeavor, Cai and Yi et al. [2] presented a groundbreaking topology called the Arch-Bridge topology. This innovative construct offers a paradigm shift in the definition of highway network structures. It creates an initial topology candidate set via processing discrete ETC data, meticulously mines and analyzes anomalous topological features, and optimizes them utilizing Dijkstra's algorithm. The experimental results show superior performance in terms of recall, precision, F1 score, and the efficiency of topology generation. This seminal work has profoundly impacted our research, upon which we have further developed. Although this investigation has significantly contributed to topology generation, it exclusively accommodates updates in response to permanent topological changes and does not facilitate dynamic topology generation. Consequently, a comprehensive analysis and synthesis of various road network generation methodologies indicate that extant approaches either overemphasize static road network structure analysis, thus struggling to accurately track dynamic topological changes, or rely heavily on voluminous data and computational resources, limiting their practicality. Therefore, there is an exigent need for an innovative methodology that leverages a stable data source and efficiently captures real-time dynamic changes in the road network topology. To address these limitations, the paper at hand proposes a new method that employs real-time ETC data and the LightGBM algorithm to construct a dynamic generation model for ETC gantry topology, effectively mitigating the main limitations of the existing methodologies.

Our method generates and dynamically updates highway gantry topology information, providing accurate segment information to mitigate the economic losses of highway managers and ETC operators. This approach can offer the intelligent transportation domain a simplistic yet efficacious tool to compensate for the deficiencies of existing methods and enhance traffic management efficiency as well as the precision of segmented toll collection.

3. Methodology

3.1. Data Introduction and Relevant Explanation

The experimental dataset utilized in this study originates from the transaction records of an ETC system on a highway in a certain province of China from 1 to 5 June 2021. This includes transaction data from ETC gantries, entry toll booths, and exit toll booths, all of which are collectively referred to as ETC Transaction Data. Approximately 31 million transaction records were collected, involving around 1.33 million vehicles. These vehicles include four types of passenger vehicles, six types of freight vehicles, and six types of

specialized operation vehicles. The classification of these vehicles is based on their purpose, structure, size, and passenger or freight capacity, following the Chinese transportation industry standard JT/T 489-2019, "Vehicle Classification of Toll for Highway" [43]. The four categories of passenger vehicles are classified according to their approved seating capacity: Category 1 (up to 9 seats), Category 2 (10–19 seats), Category 3 (20–39 seats), and Category 4 (40 seats and above). The six categories of freight vehicles are classified according to the total number of axles, length, and maximum permitted total mass, specifically including vehicles with 2, 3, 4, 5, 6, and more than 6 axles (for oversized transport vehicles). The six categories of specialized operation vehicles are similarly classified according to the total number of axles, length, and maximum permitted total mass. The first five categories follow the same classification method as freight vehicles, while the sixth category includes vehicles with no less than six axles. The ETC gantry transaction data (GData) encompasses anonymized vehicular identifiers, transactional timestamps, error codes related to transactions, and journey identifiers, supplemented by data pertaining to the gantry's identification number, name, type, and geographic coordinates (as shown in Table 1). The ETC entry toll booth data (EnData) includes anonymized vehicle identification, the identification number, names, types, and geographic coordinates of the entry toll booths, as well as the transactional timestamps, journey identifiers, and transaction error codes (as shown in Table 2). Similarly, the ETC exit toll booth data (ExData) includes anonymized vehicle identification, the identification number, names, types, and geographic coordinates of the exit toll booths, as well as the transactional timestamps, journey identifiers, and transaction error codes (as shown in Table 3). Through in-depth analysis and mining of these data, we can better understand the characteristics of the highway gantry topology and provide robust support for dynamically updating the gantry topology.

Table 1. Description of partial fields in ETC gantry transaction data.

Index	Field Name	Field Properties	Example
1	GantryID	gantry id number	340E11
2	GantryName	gantry name	Jinhai to Nanzhou
3	GantryType	gantry type	2
4	GantryCor	gantry geographic coordinates	(119.308, 25.888)
5	OBUid	vehicular identifiers	1452687261
6	PassID	journey identifiers	0142***561
7	TradeTime	transactional timestamps	2021/6/1 12:00:00
8	ErrorCode	transaction error codes	1

This table shows the descriptions of some ETC transaction data fields. PassID serves as a unique identifier used to precisely locate each journey; the ErrorCode field is used to record the status of each transaction, where 0 represents a normal transaction, and 1 signifies an abnormal transaction. In addition, OBUid is used to uniquely identify the OBU device number of a vehicle.

Table 2. Description of partial fields in ETC entrance toll booth data.

Index	Field Name	Field Properties	Example
1	EnBoothID	entrance booth id number	2100
2	EnBoothName	entrance booth name	Jinhai toll booth
3	EnBoothType	entrance booth type	0
4	EnBoothCor	entrance booth geographic coordinates	(118.318, 26.778)
5	OBUid	vehicular identifiers	1452687261
6	PassID	journey identifiers	0142***561
7	TradeTime	transactional timestamps	2021/6/1 12:00:00
8	ErrorCode	transaction error codes	1

Table 3. Description of partial fields in ETC exit toll booth data.

Index	Field Name	Field Properties	Example
1	ExBoothID	exit booth id number	2100
2	ExBoothName	exit booth name	Jinhai toll booth
3	ExBoothType	exit booth type	1
4	ExBoothCor	exit booth geographic coordinates	(118.316, 26.767)
5	OBUid	vehicular identifiers	1452687261
6	PassID	journey identifiers	0142***561
7	TradeTime	transactional timestamps	2021/6/1 12:00:00
8	ErrorCode	transaction error codes	1

In our analysis of ETC transaction data, we identified numerous fields with analogous attributes, such as identifiers, names, and types, across three datasets. Concurrently, these datasets share common fields, including transaction timestamps, geographic coordinates, PassID, and transaction error codes. These similarities aid in our comprehension and exploration of the relationships among these data, thereby facilitating the integration of data from ETC gantries and toll stations.

3.2. Preprocessing of ETC Transaction Data

3.2.1. Data Cleaning and Data Fusion

Data preprocessing is a vital step in data mining and analysis, capable of eliminating invalid data, reducing noise interference, and enhancing the accuracy and efficiency of data analysis. This section introduces how to preprocess ETC transaction data, including the filtering of error codes and transaction IDs, providing critical data support for subsequent updates to gantry topology.

Anomaly Removal: In the realm of ETC transactional data, anomalies may emerge that are attributable to equipment malfunctions or a plethora of diverse factors. Detrimental meteorological phenomena, impairment of hardware integrity, software dysfunctions, electromagnetic disruptions, network inconsistencies, and thermal overloads constitute potential confounding variables that could jeopardize the operational efficacy of RSUs and OBUs. Such circumstances may instigate instances within the ETC transactional dataset that significantly stray from the anticipated or normative patterns. For these divergent instances, the Errorcode field is designated as 1 during the data upload phase. To uphold the precision and authenticity of the ensuing data analysis, data entries where the Errorcode field is denoted as 1 are meticulously and preemptively excised from the dataset.

Related Data Filtering: The highway transaction data includes two types of transaction data, ETC and MTC (Manual Toll Collection). To concentrate on the transaction data of ETC gantries and toll booths, we need to filter out data unrelated to ETC. In this study, data starting with 01 in PassID were retained because a PassID beginning with 01 indicates that the data are ETC transaction data. Such filtering ensures that we focus only on data relevant to the research objectives, improving the specificity of the analysis.

After data cleaning, we used the SQL UNION ALL statement to merge the GData, EnData, and ExData into a single result set containing all data. Next, the merged data were sorted by PassID and TradeTime. Finally, the sorted result set was grouped by PassID. Through this integration, we obtained an ETC transaction data fusion data (EFusionData) containing ETC gantry data, ETC entry toll booth data, and ETC exit toll booth data.

In the ETC fused transaction data (EFusionData), we standardized the names of fields with similar attributes. Herein, we collectively referred to gantries and toll stations as ETC nodes (EtcNode). Based on this concept, the following fields were extracted: node ID, node name, node type, node geographic coordinates, transaction time, vehicle identification, and journey identifier. Importantly, since we removed data with a transaction error code of 1, only data with a transaction error code of 0 remained in the fusion table, so we discarded the transaction error code field in the data fusion table. Additionally, we need to explain

the node type. There are four types of nodes, namely, entry toll booths, exit toll booths, intraprovincial gantries, entry provincial boundary gantries, and exit provincial boundary gantries, represented by numbers 0, 1, 2, 3, and 4, respectively. Table 4 shows the fields of the ETC transaction data fusion table, as well as their descriptions and examples. The detailed process of data cleaning and fusion is depicted in Algorithm 1.

Table 4. Description of fields in ETC transaction fusion data.

Index	Field Name	Field Properties	Example
1	NodeID	node id number	34E102
2	NodeName	node name	Jinhai toll booth
3	NodeType	node type	1
4	NodeCor	node geographic coordinates	(118.456, 26.657)
5	TradeTime	transactional timestamps	2021/6/1 12:00:00
6	OBUid	vehicular identifiers	1452687261
7	PassID	journey identifiers	0142***561

Algorithm 1 ETC Data Cleaning and Fusion

Input: GData, EnData, ExData
Output: EFusionData
1: **GData** = **GData**[**GData**['Errorcode'] != 1] # Anomaly Removal
2: **GData** = **GData**[**GData**['PassID'].startswith('01')] # Related Data Filtering
3: **EFusionData** = **UNION_ALL**(GData, EnData, ExData) # Data Fusion
4: **EFusionData** = **EFusionData**.sort_by('PassID', 'TradeTime').group_by('PassID') # Sorting and Grouping
5: # Standardization of field names
6: **For** each transaction in **EFusionData**:
7: transaction.rename_fields(NodeID, NodeName, NodeType, NodeCor, TradeTime, OBUid, PassID)
8: transaction.encode_node_types(0,1,2,3,4)
9: **End For**

3.2.2. Generation of ETC Vehicle Trajectory Set

Upon obtaining the ETC transaction fusion table, we can construct the driving trajectories of each vehicle on the highway according to the order of vehicle transaction records. This trajectory information includes the journey identifier (PassID), the transaction time at each passed node, the ID of each passed node, the name of each passed node, the type of each passed node, and the topology section passed. In Table 5, we present an example of a vehicle trajectory data table. The detailed procedure for generating the ETC Vehicle Trajectory Set is outlined in Algorithm 2.

Table 5. An example of a ETC Vehicle Trajectory Set.

PassID	Index	Transit Node Transaction Time	Transit Node ID	Transit Node Name	Transit Node Type	Transit Topological Segment
0142***561	1	2021-06-01 08:00:00	2100EN	Zhongnan Jinghai Booth	0	
	2	2021-06-01 09:00:00	350001	Jinghai to Xijin Hub	2	2100EN–350001
	3	2021-06-01 10:00:00	350003	Xijin Hub to Dongcheng	2	350001–350003
	4	2021-06-01 11:00:00	350005	Dongcheng to Xiyu Hub	2	350003–350005
	5	2021-06-01 12:00:00	350007	Xiyu Hub to Nancheng	2	350005–350007
	6	2021-06-01 13:00:00	2200EX	Zhongnan Nancheng Booth	1	350007–2200EX

Algorithm 2 Generation of ETC Vehicle Trajectory Set

Input: EFusionData
Output: ETC Vehicle Trajectory Set (EVTSet)
1: **EVTSet** = Initialize empty list #
2: Unique_PassID_List = ExtractUniquePassID(EFusionData)
3: **For** each PassID in Unique_PassID_List:
4: Transactions = ExtractTransactions(EFusionData, PassID) # Get transactions related to current PassID
5: Sorted_Transactions = SortTransactions(Transactions) # Sort transactions by time
6: Vehicle_Trajectory = GenerateVehicleTrajectory(Sorted_Transactions) # Generate trajectory from transactions
7: **Append** Vehicle_Trajectory to **EVTSet**

3.3. Extraction of ETC Gantry and Toll Booth Node Set

After data preprocessing, we analyzed transaction data from ETC gantries, entry toll booths, and exit toll booths to extract ETC nodes, forming a node set of ETC gantries and toll booths, hereinafter referred to as the ETC node set. We selected gantries and in-province toll booths in normal operation to ensure that the extracted node set represents devices in actual operation, which facilitates further analysis. Through transaction data analysis, we identified and extracted valid gantry and toll booth nodes. We excluded the toll booths with abnormal ID codes, abandoned toll booths, and out-of-province toll booths. During the selection process, we found discrepancies between the entry toll booth set and the exit toll booth set. The booths in the discrepancies only had identifiers in the transaction data but no names, and, in fact, did not exist. This could be due to these toll booths being boundary toll booths, identifier changes, or names not displayed during maintenance. We eliminated these aberrant data. Ultimately, the ETC node set includes 1805 nodes, including 1051 gantry nodes and 754 toll booth nodes (378 entry toll booth nodes and 376 exit toll booth nodes), as shown in Figure 1. This ETC node set is the foundation for subsequent topology candidate set generation (Table 6).

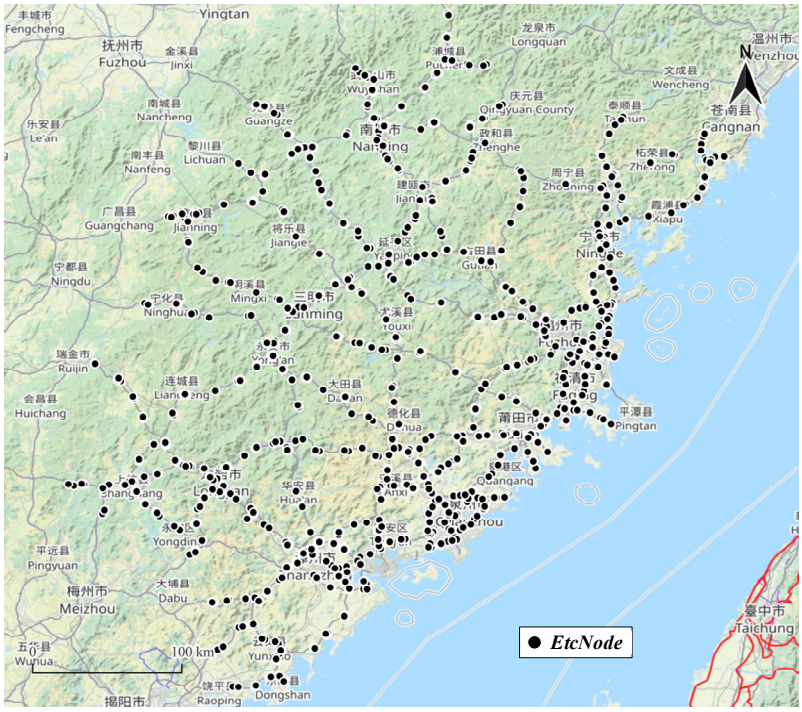


Figure 1. The distribution map of ETC nodes in a certain province.

Table 6. Statistical count of ETC tollgate and toll booth nodes.

Category	Number of Nodes
Gantry Nodes	1051
Entry Toll Booth Nodes	378
Exit Toll Booth Nodes	376
Total	1805

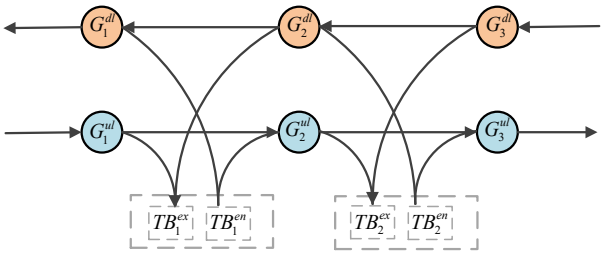
3.4. Extraction and Selection of Candidate Topologies Set

Before proceeding to the extraction of the topological candidate set, it is paramount to familiarize ourselves with the basic topological structure that characterizes highways. This structure is a composite of ETC gantries, toll booths, and the highway lanes themselves. As depicted in Figure 2a, we utilize TB to represent toll booths, with TB^{ex} and TB^{ex} denoting the exit and entrance toll booths, respectively. G signifies ETC gantries, with G^{ul} and G^{dl} representing the gantries on the upward and downward lanes, respectively. A typical ETC topological structure is constituted by these nodes, along with the directional segments linking them. Following the procurement of the ETC vehicle trajectory and node datasets, we were equipped to generate a preliminary set of topological candidates. This assemblage was extracted from the sequential pairing of adjacent nodes within the vehicle trajectory dataset, which resulted in a comprehensive count of 31,379 potential topologies. Subsequently, the candidate set of topologies underwent data preprocessing to screen and eliminate a substantial number of erroneous topologies. Erroneous topologies include the following categories:

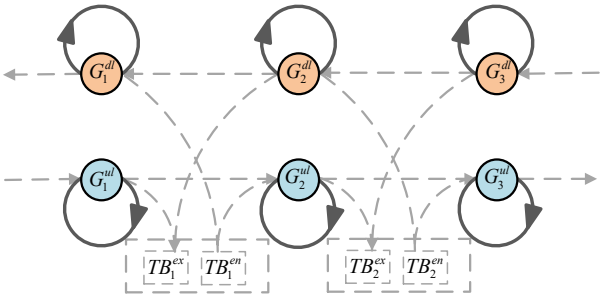
1. Topologies not included in the ETC node set: In these topologies, the starting or ending point, or both, are not part of the node set. Thus, these topologies can be directly eliminated.
2. Circular topologies: As depicted in Figure 2b, in these topologies, the start and end nodes are identical. This type of erroneous topology can be easily removed.
3. Bidirectional topologies (Figure 2c): These candidate topologies feature nodes from the upward (downward) lane that directly reach nodes of the downward (upward) lane.
4. Topologies terminating at toll booth entrances (Figure 2d): Similar to the case of topologies originating from exits, these topologies conclude at a toll booth entrance.
5. Topologies originating from toll booth exits (Figure 2e): These topologies commence from a toll booth exit. However, in actual trajectories, toll booth exits typically appear at the end, hence such topologies do not exist.
6. Topologies from an entrance toll booth to an exit toll booth (Figure 2f): These candidate topologies commence from an entrance toll booth and conclude at an exit toll booth. However, in actual trajectories, several gantries must be passed between the entrance and exit toll booths, rendering such topologies clearly erroneous and subject to direct elimination.
7. Topologies incorporating out-of-province toll booths (Figure 2g): These topologies feature a toll booth node outside the province as the starting point or endpoint. In the figure, the toll station outside the province is OTB. Due to the presence of inter-provincial ETC transaction data in the actual dataset, topologies may contain nodes of out-of-province toll booths. These nodes can be easily identified as their toll booth names differ from those of in-province nodes, despite having identical IDs.

Utilizing the aforementioned rules, we successfully identified and eliminated a substantial number of erroneous topologies, thus ensuring the accuracy of our analytical results. Following preprocessing, we retained a total of 13,598 topologies. Nevertheless, among these, 10,166 topologies had a vehicle traffic volume of less than 5 within a 7-day period and were confirmed as invalid topologies following inspection. Upon elimination of these invalid topologies, we retained 3432 valid topologies. These valid topologies form

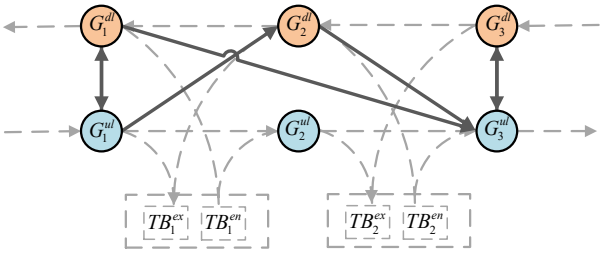
our candidate set of topologies; in Table 7, we present the fields of the candidate set of topologies, as well as their descriptions and examples.



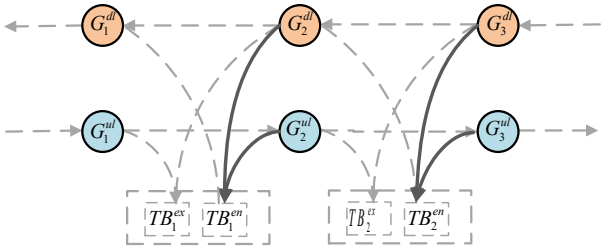
(a) Normal topologies



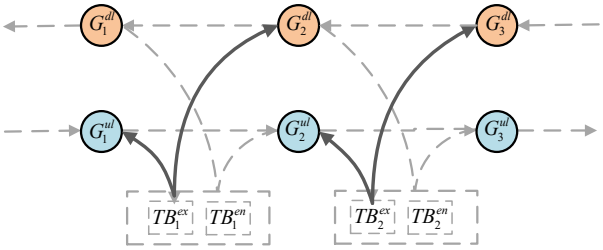
(b) Circular topologies



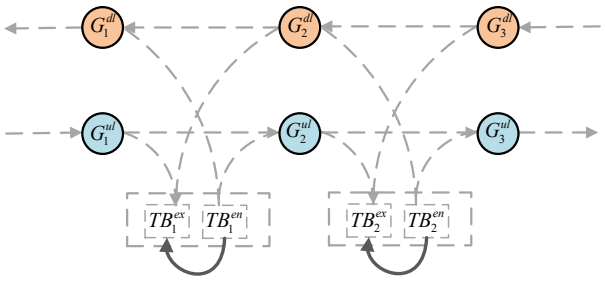
(c) Bidirectional topologies



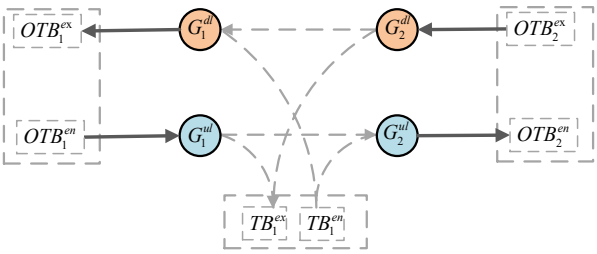
(d) Topologies terminating at toll booth entrances



(e) Topologies with toll booth exits as their origins



(f) Topologies from an entrance toll booth to an exit toll booth



(g) Topologies incorporate out-of-province toll booths

Figure 2. Illustration of normal and erroneous highway topologies.

Table 7. Description of fields in ETC topology data.

Index	Field Name	Field Properties	Example
1	Topology	topology array	['350E01', '350E03']
2	StartID	start node id	350E01
3	EndID	end node id	350E03
4	StartName	start node name	Jinhai to Nanzhou
5	EndName	end node name	Nanzhou to Xicheng
6	StartType	start node type	2
7	EneType	end node type	2
8	StartCor	start node geographic coordinate	(118.2434, 24.6884)
9	EndCor	end node geographic coordinate	(118.4107, 24.7195)
10	TrafficVolume	topology traffic volume	257,396
11	TopologyDistance	topology route distance	18,561 (m)

Herein, 'Traffic Volume' is derived based on the number of vehicles that traversed this topology within a 5-day period according to the vehicle trajectory dataset, and 'Topology Distance' is the planned route distance obtained from the Amap (Amap Map) API, using the geographic coordinates of the starting and ending nodes.

3.5. Feature Vector Modeling

In the analysis of the candidate topology set, we considered five feature dimensions: Topology Traffic Volume (*TTV*), Topological Passage Rate (*TPR*), Normalized Start Rate (*NSR*), Normalized End Rate (*NER*), and Topology Distance (*TD*). Taking the topology (a, b) as an example, we will illustrate the calculation methods for these features.

- Topology Traffic Volume (*TTV*): *TTV*, as a basic indicator of traffic flow, can reflect the importance of the topology in the traffic system. By analyzing this feature, we can understand the traffic differences in different topologies. *TTV* is calculated using Equation (1), where $T_i(a, b)$ represents the number of trajectories of topology (a, b) on the i-th day, and N represents the number of days considered. In this study, $N = 5$.

$$TTV_{(a,\ b)} = \sum_{i=1}^{N} T_i(a,\ b) \tag{1}$$

- Topological Passage Rate (*TPR*): To calculate this feature, first, we calculate the number of trajectories containing both nodes a and b within a 5-day span, regardless of whether these two nodes are directly connected. We refer to this as the Coexisting Nodes Trajectory Volume (*CNTV*). The calculation formula for *CNTV* is shown in Equation (2), where $C_i(a, b)$ represents the number of trajectories containing nodes A and B on the i-th day. The reason for choosing the *TPR* feature is that some topologies may have a large number of erroneous transactions or omissions, resulting in a high value of *CNTV*. By calculating the ratio of *CNTV* to *TTV*, we can more accurately assess the possibility of each topology in actual work. Then, we calculate the Topological Passage Rate, through the formula $TPR = TTV/CNTV$.

$$CNTV_{(a,b)} = \sum_{i=1}^{N} C_i(a, b) \tag{2}$$

$$TPR_{(a,b)} = \frac{TTV_{(a,b)}}{CNTV_{(a,b)}} \tag{3}$$

- Normalized Start Rate (*NSR*): In the actual high-speed ETC gantry topology, each starting node has 1 to 4 endpoints, and the rest of the topologies are likely to be generated by erroneous data. Therefore, by calculating the proportion of the traffic of candidate topology (a, b) in all the topologies starting from a, we can evaluate the

possibility of this topology in actual work. Therefore, *NSR* is one of the crucial features for evaluating the existence probability of a topology. The computation formula for *NSR* is as shown in Equation (4), where *M* denotes the number of end nodes reached by the initial node *a* in the candidate topology.

$$NSR_{(a,b)} = \frac{TTV_{(a,b)}}{\sum_{j=1}^{M} TTV_{(a,j)}} \tag{4}$$

- Normalized End Rate (*NER*): *NER* is the normalized rate of the end node, calculating the ratio of the traffic of topology (a, b) to the total traffic of all topologies ending at b. Since one end node may be connected to multiple start nodes, we need to consider all in-degree topologies of this end node. The calculation of *NER* is similar to that of *NSR*, but it is grouped by end node b. The computation formula for *NSR* is as shown in Equation (5).

$$NER_{(a,b)} = \frac{TTV_{(a,b)}}{\sum_{j=1}^{M} TTV_{(j,b)}} \tag{5}$$

Topology Distance (*TD*): *TD* refers to the path distance between two gantries or toll booth nodes in the ETC highway system, which is used for traffic management and cost calculation. Its calculation is denoted as $TD_{(a,b)}$.

These five features allow us to construct the feature vector of candidate topology, as in Equation (6), and assess their likelihood of actual existence in the traffic system.

$$v(a,b) = \left\{ TTV_{(a,b)}, TPR_{(a,b)}, NSR_{(a,b)}, NER_{(a,b)}, TD_{(a,b)} \right\} \tag{6}$$

3.6. Authenticity Verification and Accuracy Annotation of Candidate Topologies

Prior to deploying the LightGBM model for the dynamic updates of gantry topologies, it is necessary to substantiate the authenticity and annotate the accuracy of each candidate topology, to affirm their legitimate existence and correctness. The benchmarks for existence verification encapsulate: (1) No other gantries or toll booths should be present on the road the trajectory of the topology; (2) No actions indicative of highway exit should be observed within the topology trajectories. In order to attain these benchmarks, we initially utilize the AMap API to gather the trajectory data for each candidate topology. Following this, we project each topology trajectory along with the ETC node set onto the QGIS map, and, through visual inspection, we determine whether any candidate topology trajectories pass through other nodes or exhibit cases of highway exit. If a topology trajectory complies with the aforementioned criteria, it is validated as a legitimate topology. Otherwise, it is designated as a flawed topology.

3.7. Dynamic Generation Method of Highway ETC Gantry Topology Based on LightGBM

LightGBM, an efficient Gradient Boosting Decision Tree (GBDT) algorithm, was proposed by Ke et al., 2017 [44] and has since been extensively utilized in diverse data mining tasks such as classification, regression, and ranking. It exhibits superior performance in both efficiency and effectiveness, and it has a commendable capacity to handle non-linear data relationships. In the present study, we employ the LightGBM model to predict the authenticity of gantry topology, thereby enabling the dynamic update of gantry topology.

The primary steps of LightGBM are as follows:

1. Gradient Boosting: LightGBM is a model based on gradient boosting. Throughout the training process, it repetitively builds decision trees, striving to diminish the discrepancy between the predicted value,$f(x)$, and the true value, y, at every step, thereby continually enhancing the prediction accuracy of the model. Its loss function is defined as $L(y, f(x))$. The iterative model can be expressed as:

$$f_{t+1}(x) = f_t(x) + \nu h(x) \tag{7}$$

where ν represents the learning rate.

2. Decision Tree Construction: Within LightGBM, the decision tree utilizes a depth-first approach for splitting, and it carries out efficient node splits according to the histogram of features. Additionally, LightGBM is capable of handling categorical features and employs a Gradient-based One-Side Sampling (GOSS) method [45] in feature selection, significantly enhancing training efficiency on high-dimensional data. GOSS is a sampling technique that preserves all large gradient samples and randomly selects a fraction of small gradient samples. This practice maintains the data's distribution while reducing computational cost. The update of tree nodes is represented in a form approximated by the least squares method, as shown in Equation (8):

$$c_j = -\frac{\sum_{x \in I_j} g_i}{\sum_{x \in I_j} h_i} \tag{8}$$

where g_i and h_i denote the first and second order gradients, respectively. To maximize the model's performance at each split, LightGBM seeks the optimal split point at every node division. The method of locating the best split point is achieved by maximizing the information gain. The specific calculation of information gain is as follows:

$$Gain = \left(\sum_{x \in I_L} \left[g_i + \nu h_i c^L \right] \right)^2 + \left(\sum_{x \in I_R} \left[g_i + \nu h_i c^R \right] \right)^2 - \left(\sum_{x \in I} [g_i + \nu h_i c] \right)^2 \tag{9}$$

where c^L, c^R, and c are the optimal output values for each leaf node.

3. Ensemble Prediction: After constructing N decision trees, LightGBM aggregates them for prediction. For a new input sample x, it is fed into each decision tree, and the obtained prediction result is the weighted average of all decision tree prediction results:

$$f(x) = \sum_{i=1}^{N} T_i(x) \tag{10}$$

4. Hyperparameter Optimization: A grid search is employed for hyperparameter optimization. The primary hyperparameters encompass the number of decision trees, the maximum depth of each tree, the learning rate, the number of features, etc. A parameter grid is defined, and the optimal hyperparameter combination is identified by iterating over potential parameter combinations.

In this study, we have employed the LightGBM model to predict the veracity of highway ETC gantry topology, thereby enabling the dynamic generation of gantry topology. LightGBM, an efficient Gradient Boosting Decision Tree (GBDT) algorithm, has made a significant contribution to our study.

Firstly, our study involves five feature dimensions, which may have complex interactive relationships. LightGBM is capable of effectively capturing these interactions, thereby enhancing the accuracy of the model's predictions. Within the context of highway ETC gantry topology data, the relationships between features may be complex and non-linear. For instance, the relationship between Topology Distance (*TD*) and Topology Traffic Volume (*TTV*) may not be linear. LightGBM is adept at handling these non-linear relationships, thereby further improving the predictive performance of the model.

Secondly, LightGBM is a robust classification algorithm that can effectively handle binary classification problems. In our study, we labelled the candidate gantry topology as either existing (marked as 1) or non-existing (marked as 0), and allowed LightGBM to train and predict on the candidate gantry topology. This is a typical binary classification problem. Through LightGBM, we are able to effectively solve this problem, thus actualizing the dynamic generation of gantry topology.

Moreover, LightGBM has the advantage of preventing model overfitting. It introduces regularization parameters (such as L1 and L2 regularization) and uses a Gradient-based

One-Side Sampling (GOSS) method, effectively preventing model overfitting and enhancing the model's generalization capability. Simultaneously, LightGBM provides a series of adjustable hyperparameters, such as the number of decision trees, the maximum depth of each tree, the learning rate, and the number of features, etc. In our study, we optimized these hyperparameters through a grid search method and identified the optimal hyperparameter combination, thereby further improving the performance of the model.

In summary, when dealing with large volumes of data and high-dimensional features, LightGBM often exhibits superior efficiency and performance. This is evident in our study, where we were able to utilize the well-trained LightGBM model to automatically generate predictive results for new gantry topology data, thereby actualizing the dynamic generation of gantry topology.

4. Experiment and Results Analysis

The experimental platform utilized an Intel (R) Core (TM) i9-10900K CPU with 10 cores and a base clock of 3.70 GHz, along with 64 GB RAM. The experiments were performed on the CentOS Linux release 7 September 2009 (Core) operating system and utilized Python 3.7.11 as the programming language. The experiment was implemented on Jupyter Notebook—an interactive programming IDE. For comprehensive information regarding Jupyter Notebook, please refer to its official website: https://jupyter.org/ (accessed on 30 July 2023).

4.1. Construction of Feature Vectors

In accordance with our feature vector model, we fabricated a training feature vector set for high-speed gantry topology generation, and several examples are demonstrated in Table 8. Each vector encompasses five-dimensional attributes and their respective sample classification labels. These five attributes include Topology Traffic Volume (*TTV*), Topology Passage Rate (*TPR*), Normalized Start Rate (*NSR*), Normalized End Rate (*NER*), and Topology Distance (*TD*). The sample classification label signifies the existence of the topology: 0 denotes the absence of the topology, whereas 1 indicates the presence of the topology in the actual traffic network.

Table 8. Sample of candidate gantry topological feature vector.

Candidate Topo	TTV	TPR	NSR	NER	TD	Label
['67**EN', '34**19']	6091	0.99918	0.2431	0.1157	1357	1
['67**EN', '35**03']	1212	0.999176	0.212	0.4994	13723	0
['34**07', '34**0B']	1199	0.999167	0.1612	0.3223	1357	1
['35**62', '35**5F']	314	0.996825	0.397	0.139	1802	1
['35**04', '35**11']	933	0.996795	0.1447	0.6839	1346	1
['34**15', '34**19']	2788	0.996782	0.0281	0.1053	1545	1
['79**EN', '35**23']	128	0.711111	0.0242	0.0006	2554	0
['35**13', '35**23']	211	0.710438	0.0179	0.0012	6955	0
['34**07', '35**5F']	571	0.710199	0.1008	0.0049	4523	0
['64**EN', '34**19']	236	0.571429	0.0977	0.278	3254	1
['47**EN', '34**19']	6091	0.99918	0.2431	0.1157	1357	1
['34**EN', '35**03']	1212	0.999176	0.212	0.4994	13723	0

Considering the sensitivity of the data, we have anonymized the toll gate numbers, in which ** represents two characters of the toll gate number, which could be either numbers or letters.

To establish a viable predictive model, an initial correlation analysis was conducted on the five features that define the topology generation problem, the results of which are depicted in Figure 3.

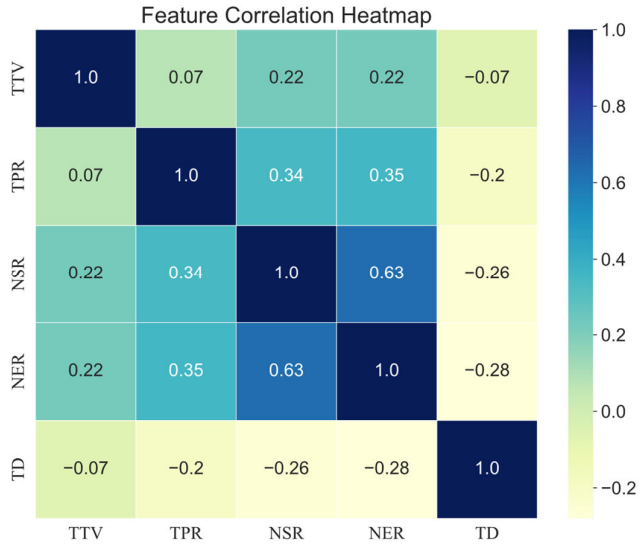


Figure 3. Correlation matrix of feature.

The correlation between *TTV* and *TPR* is 0.07, suggesting that the Topology Traffic Volume and the topology pass rate can independently reflect the characteristics of the topology without being tightly correlated. This finding can assist us in better understanding the uniqueness of each feature in depicting the properties of the topology. The correlation between *NSR* and *NER* is relatively high, reaching 0.63, which may indicate that these two features reflect the same or similar information to a certain extent. Therefore, when interpreting model results or understanding influencing factors, we can consider these two factors jointly. The correlation between feature *TD* and the other features is relatively low, implying that *TD* might contribute additional information necessary for the model. This lends a certain value to the *TD* feature when understanding its role in the model and interpreting prediction results.

4.2. Experimental Setup and Parameter Selection

In our experimental setup, parameter selection and optimization played a pivotal role in the performance outcomes of the LightGBM model. We focused primarily on four crucial categories of parameters: general parameters, core parameters, regularization parameters, and sampling parameters, employing a grid search methodology for fine-tuning. Firstly, general parameters encompass the 'number of estimators (n_estimators)' and 'learning rate'. These two parameters have a direct bearing on the model's learning capability and the pace at which it fits the data. An appropriately set 'number of estimators' ensures that the model possesses ample learning capacity to understand the data, while the 'learning rate' delineates the step size in the model's learning process. Secondly, core parameters dictate the basic structure and complexity of the model, which include 'maximum tree depth (max_depth)', 'number of leaves (num_leaves)', 'minimum child samples (min_child_samples)', 'minimum child weight (min_child_weight)', and 'minimum split gain (min_split_gain)'. In our experiments, the choice of 'maximum tree depth' was particularly salient, as it directly influences the model's complexity and fitting capacity. Further, regularization parameters, composed of 'L1 regularization term (reg_alpha)' and 'L2 regularization term (reg_lambda)', are utilized to inhibit overfitting phenomena in the model. Appropriate regularization helps prevent the model from overfitting the training data, thereby enhancing the model's generalization capacity. Lastly, sampling parameters, which include 'subsample ratio (subsample)', 'column sample by tree (colsample_bytree)', and 'subsample frequency (subsample_freq)', primarily control the sampling of data and features. These parameters aid in mitigating overfitting and augmenting the efficiency of the training process. To get the optimal parameter configuration, we used a grid search for hyperparameter tuning and combined it with 5-Fold Cross-Validation. This approach

effectively prevents overfitting and improves the model's generalization on new data. Specifically, we divided the training set into five subsets, each time using four subsets as training data and the remaining one as validation data. In this way, we could obtain a robust parameter configuration for achieving the best model performance. This parameter optimization strategy has helped us find the best parameter combination in the search space, thereby greatly improving the model's performance. Specific search ranges, step sizes, and optimal values can be referred to in Table 9.

Table 9. Optimal combination of important parameters of LightGBM.

Parameter Categories	Parameter	Search Range	Step Size	Optimal Value
general parameters	n_estimators	[10, 500]	10	80
	learning_rate	[0.1, 0.01, 0.001]	-	0.1
core parameters	max_depth	[3, 10]	1	5
	num_leaves	[2, 50]	1	7
	min_child_samples	[5, 50]	5	5
	min_child_weight	[0.001, 0.01, 0.1]	-	0.001
regularization parameters	min_split_gain	[0, 0.1, 0.5]	-	0
	reg_alpha	[0, 0.1, 0.5]	-	0
	reg_lambda	[0, 0.1, 0.5]	-	0
sampling parameters	subsample	[0.5, 0.9]	0.2	0.5
	colsample_bytree	[0.5, 0.9]	0.2	0.5
	subsample_freq	[1, 5]	2	3

4.3. Empirical Outcomes and Integrated Appraisal

In our dataset, each sample is assumed to be independently and identically distributed (i.i.d), which implies that every sample originates from the same probability distribution and is independent of all other samples. Given this assumption, we chose classifiers that are known for effectively handling i.i.d multivariate feature data. To appraise the performance efficacy of various machine learning paradigms within the scope of the gantry topology generation task, we conducted a series of experiments. Alongside LightGBM, our evaluation paradigm incorporated a range of established machine learning methodologies, such as Logistic Regression (LR), Naive Bayes (NB), Linear Discriminant Analysis (LDA), K-Nearest Neighbors (KNN), Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM), Gradient Boosting (GB), AdaBoost (AB), Extreme Gradient Boosting (XGB), Quadratic Discriminant Analysis (QDA), Gaussian Process Classifier (GPC), Stochastic Gradient Descent (SGD), and Linear Support Vector Machine (Linear SVM). These methodologies were selected due to their demonstrated proficiency in handling datasets with complex, high-dimensional features, which bears a similarity to the nature of our ETC gantry topology data. These were juxtaposed with the LightGBM algorithm to establish comparative performance parameters. For maintaining the replicability of the empirical outcomes, we resorted to the use of default parameters during the algorithmic training phase and designated the random seed as 1. The specifications and configurations of each algorithm have been placed in Appendix A. For LightGBM, we elected the parameter combination that was subject to rigorous optimization to accomplish superior performance efficacy. The algorithmic iterations deployed in our study comprised scikit-learn version 1.0.2, XGBoost version 1.5.1, and LightGBM version 3.3.5.

To evaluate the performance of each model, the evaluation metrics selected in this study include *Accuracy*, *Precision*, *Recall*, and *F*1-Score. These are defined as follows:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{11}$$

$$Precision = \frac{TP}{TP + FP} \tag{12}$$

$$Recall = \frac{TP}{TP + FN} \tag{13}$$

$$F1 - score = \frac{2 \times Recall \times Precision}{Recall + Precision} \tag{14}$$

where *TP* (True Positives) are the count of topologies that our model correctly identifies as existing. *FP* (False Positives) represent the count of topologies incorrectly recognized by our model as existing when in reality, they do not exist. *TN* (True Negatives), on the other hand, refer to the count of topologies that our model correctly identifies as non-existing. Lastly, *FN* (False Negatives) represent the instances where our model incorrectly classifies existing topologies as non-existent.

In the ensuing phase, we executed experiments and evaluations on a homogenous dataset, drawing performance comparisons of each algorithm with respect to evaluation matrices such as accuracy, precision, recall, and *F*1-score, as delineated in Table 10.

Table 10. Evaluation metrics comparison of different algorithms.

Model	*Accuracy*	*Precision*	*Recall*	*F*1-Score
LR	0.8239	0.8918	0.8835	0.8877
NB	0.8705	0.8717	0.9797	0.9225
LDA	0.9403	0.9310	0.9982	0.9634
KNN	0.8574	0.8625	0.9741	0.9149
DT	0.9549	0.9688	0.9741	0.9714
RF	0.9651	0.9624	0.9945	0.9782
SVM	0.8617	0.8517	0.9982	0.9191
GB	0.9636	0.9624	0.9926	0.9773
AB	0.9578	0.9571	0.9908	0.9737
XGB	0.9651	0.9691	0.9871	0.9780
LightGBM	0.9709	0.9668	0.9982	0.9822
QDA	0.9607	0.9606	0.9908	0.9754
GPC	0.4032	0.9456	0.2569	0.4041
SGD	0.7365	0.9186	0.7301	0.8136
Linear SVM	0.7875	0.7875	1.0000	0.8811

In pursuit of offering an enriched visual illustration comparing the performance of LightGBM vis-a-vis other models, we have meticulously crafted a graphical representation (refer to Figure 3). This representation systematically encapsulates the relative efficacies of various algorithms gauged across a range of evaluation metrics. Within the purview of our empirical investigation, LightGBM emerged as a notably superior algorithm for the task of gantry topology generation, outperforming all contenders in terms of accuracy and F1 scores, with the exception of recall, where it trailed marginally behind Linear SVM. An accuracy metric of 0.9709 exemplifies LightGBM's adept capability in forecasting the topology of gantries with a high degree of precision. While LightGBM's recall metric, registered at 0.9982, is slightly outperformed by Linear SVM's optimal recall of 1.0000, it is crucial to note that Linear SVM lags significantly behind in terms of accuracy (0.7875), precision (0.7875), and the F1 score (0.8811). Consequently, in this holistic context, LightGBM's recall metric still demonstrates remarkable excellence. This metric underscores the model's proficiency in identifying true positive gantry topologies, a critical factor within the task of gantry topology generation. Simultaneously, LightGBM's F1 score of 0.9822 epitomizes an exemplary equilibrium between minimizing false positives (indicative of high precision) and maximizing true positives (indicative of high recall). Despite a precision score of 0.9668, which surpasses several models, it only ranks third across the evaluated models, indicating a relative underperformance of LightGBM in identifying negative instances. Nonetheless, it is important to highlight that, despite a marginally lower precision and being second to Linear SVM in recall, these factors do not compromise the overarching

prowess of LightGBM. Particularly in the context of gantry topology generation, recall takes precedence, as the goal is to identify the maximum count of true gantries, even if it involves potential false positives. Therefore, despite certain minor shortcomings, LightGBM's remarkable performance in terms of accuracy, recall, and the F1 score, coupled with the precise requirements of gantry topology generation, unequivocally endorse LightGBM as the recommended algorithm for this task.

In the endeavor to augment the efficacy of our model, we adopted the stratagem of 5-Fold Cross-Validation. This rigorous validation technique not only provided a multifaceted assessment of our model's performance, but also furnished insights into its generalizability after each training iteration. As delineated in Figure 4, we observed a consistent decrement in the log-loss values across all five partitions, underscoring the robust learning capability of our model and the successful mitigation of overfitting. Furthermore, the attainment of lower log-loss values across all folds during the terminal rounds signified the absence of underfitting.

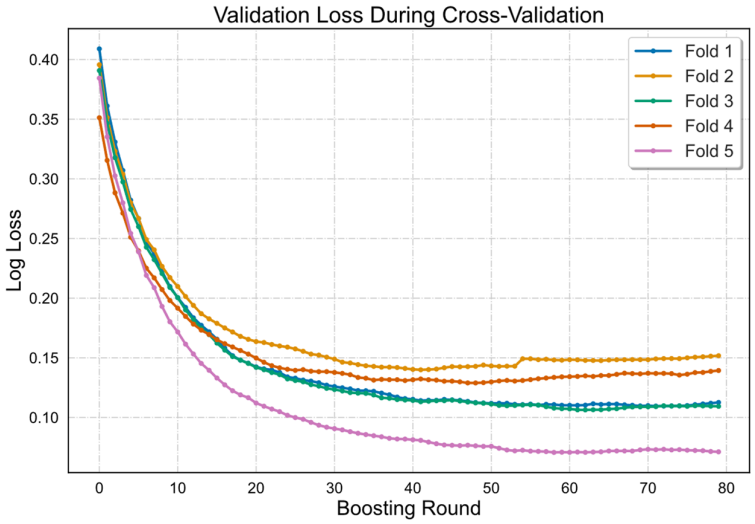


Figure 4. Validation log-loss during training.

Throughout the 5-Fold Cross-Validation process, we meticulously recorded the key performance metrics of accuracy, precision, recall, and F1 score for each fold (refer Table 11). The fifth fold, in particular, demonstrated superior accuracy and precision metrics when juxtaposed with the model devoid of cross-validation. To be precise, the 5-Fold Cross-Validation reported an accuracy of 0.976321 and a precision of 0.980306, both surpassing their counterparts from the non-cross-validated model, which registered an accuracy of 0.9709 and precision of 0.9668. While the recall metric from 5-Fold Cross-Validation (0.991150) marginally trailed behind the non-cross-validated model (0.9982), the F1 score (0.985699) outperformed the non-cross-validated counterpart (0.9822). These outcomes not only demonstrate the quintessential role of 5-Fold Cross-Validation in model appraisal and selection but also testify to its superiority in most scenarios, notwithstanding the volatility it introduces.

Table 11. Evaluation metric performance during 5-Fold Cross-Validation.

Fold	*Accuracy*	*Precision*	*Recall*	*F1*-Score
1	0.9672	0.9631	0.9978	0.9801
2	0.9599	0.9614	0.9911	0.9760
3	0.9599	0.9577	0.9956	0.9763
4	0.9563	0.9667	0.9831	0.9748
5	**0.9763**	**0.9803**	**0.9912**	**0.9857**

The bolded sections represent the best-performing round across all evaluation metrics.

In order to assess the time efficiency of our model when dealing with datasets of varying sizes, we designed a series of experiments to approximate its time complexity. We first established a range of test sets, starting from 10% of the total data volume, incrementing by 1% each time, until the entire dataset was encompassed. Subsequently, we trained and made predictions with the LightGBM model on each test set, precisely recording the execution time of each operation, as shown in Figure 5. As depicted in the graph, the time taken for prediction remains relatively constant as the percentage of data used increases, oscillating slightly around the average time. This observation suggests that the time complexity of our algorithm tends to be a constant, i.e., O(1). This result indicates that our model maintains a high level of execution efficiency even when faced with expanding data scales.

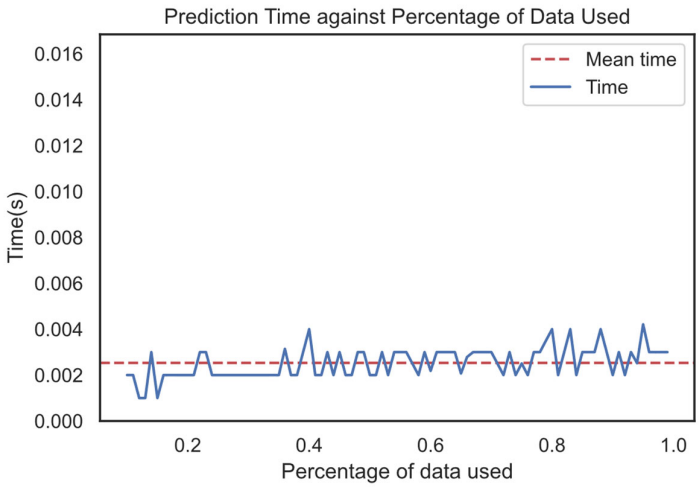


Figure 5. Prediction time against percentage of data used.

In comparison to a closely related study by Cai and Yi et al. [2], our method demonstrates evident superiority in terms of time complexity, evaluation metrics, and operational efficiency, as shown in Table 12.

Table 12. Comparison of our method with Cai and Yi et al. [2].

Metric	Our Method	Cai and Yi et al. [2]
Accuracy	0.9763	-
Precision	0.9803	0.966
Recall	0.9912	0.982
*F*1-Score	0.9857	0.974
Time Complexity	O(1)	O(n)
Average Time to Generate a Topology (ms)	0.00142	2
Total Time to Generate all Topologies (s)	0.004	5.76

Specifically, our method achieved scores of 0.9763, 0.9803, 0.9912, and 0.9857 on accuracy, precision, recall, and F1-score, respectively. Although the approach of Cai and Yi et al. [2] also reported high scores of 0.966, 0.982, and 0.974 for precision, recall, and F1-score, respectively, they did not present an accuracy score. Thus, our method surpasses theirs across all reported metrics.

From the standpoint of time complexity, our method exhibits an almost constant runtime with an increasing amount of data, implying a time complexity nearing O(1). Conversely, the runtime of the method proposed by Cai and Yi et al. [2] escalates linearly with the data volume, indicating a time complexity of O(n). Hence, our method offers superior time efficiency.

Additionally, our method excels in operational efficiency. Specifically, our approach generates 2819 topologies within a mere 0.004 s, averaging less than 0.00142 milliseconds

per generated topology. In contrast, Cai and Yi et al. [2]'s method takes 5.76 s to generate 2950 topologies, averaging less than 2 milliseconds per topology. Therefore, our method significantly outperforms that of Cai and Yi et al. [2] in terms of the efficiency of topology generation.

On this basis, we endeavored to understand the decision-making process of the model and ascertain the most salient features influencing gantry topology generation through a feature importance analysis. This examination encompassed all the features utilized during the training of the LightGBM model, quantifying the contribution of each feature to the predictive performance of the model. The detailed results are displayed in Figure 6. As observed from Figure 6, the feature '*PN*' boasts the highest importance ratio, highlighting its substantial influence on gantry topology generation. The feature '*TD*' trails '*PN*', but still maintains a relatively high importance ratio. On the other hand, features '*PR*', '*NSR*', and '*NER*' have lower importance ratios. These findings suggest that the features '*TTV*' and 'TD' are indispensable to our model's prediction, given their substantial contributions to the predictive capacity for the target variable. Conversely, the importance of the '*TPR*', '*NER*', and '*NSR*' features is less pronounced, indicating a potentially smaller contribution to the model's predictive capabilities.

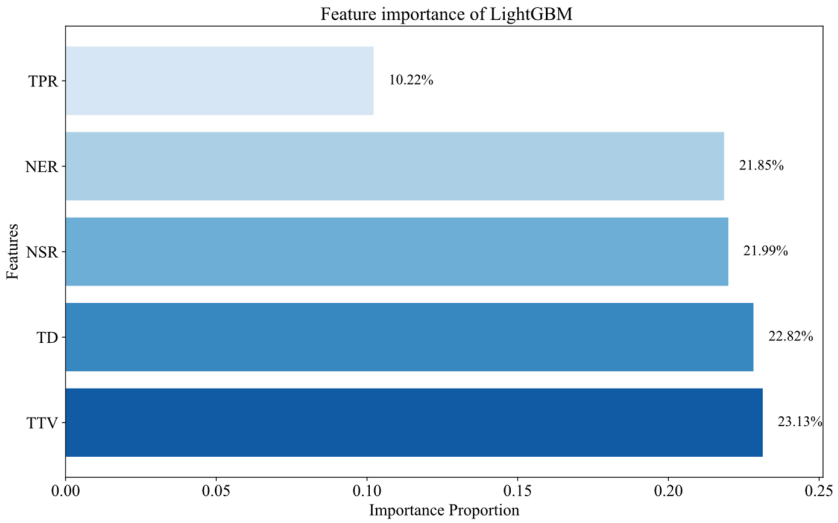
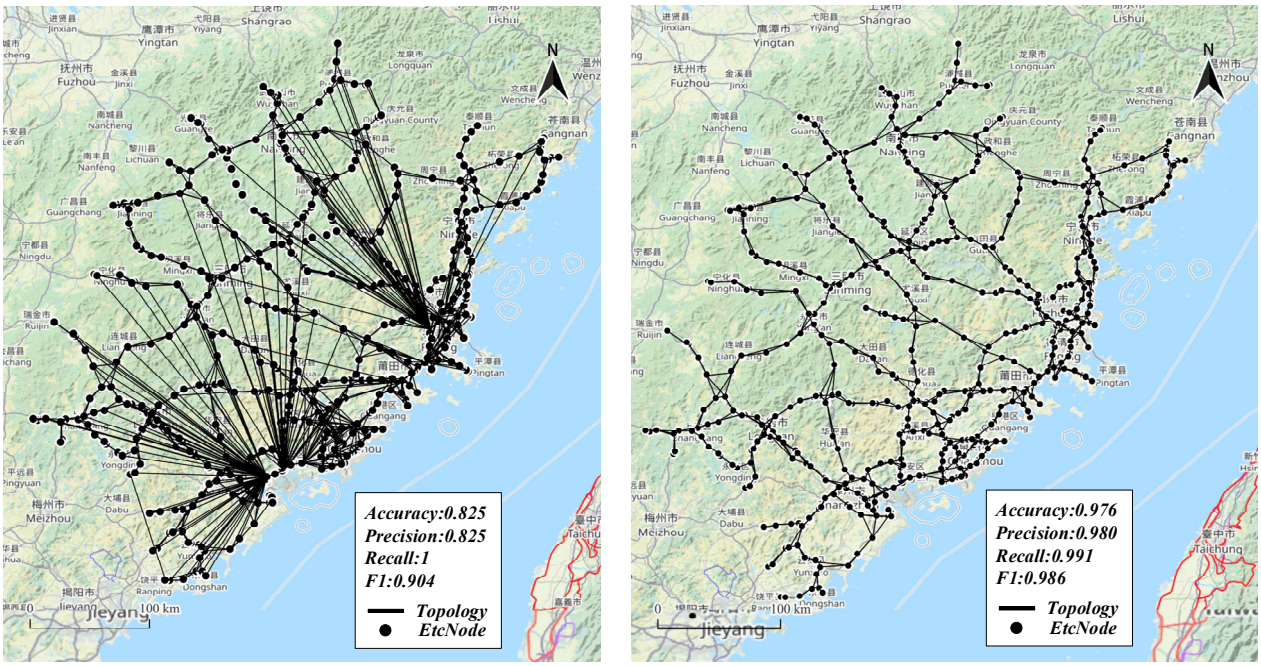


Figure 6. Feature importance of LightGBM.

4.4. Results of Gantry Topology Generation

We apply the trained LightGBM model to classify and predict the candidate topology, and use the predicted topology as the updated gantry topology set.

In this result, we apply the trained LightGBM model to classify and predict the candidate topology, and we use the predicted topology as the updated gantry topology set. As illustrated in Figure 7, we initially generated a preliminary set of candidate topologies and, subsequently, discerned the final topological framework through algorithmic filtration. Among the initially generated topologies in Figure 7a, despite the presence of a large number of erroneous topologies such as long-distance topologies, we still assumed that all 3432 preliminary topologies were correct. This assumption served as our baseline model, which preset all labels as 1. Under these circumstances, the model's accuracy and precision were both 0.825, the recall rate was 1, and the F1 score was 0.904. However, this all-positive-prediction model could not accurately distinguish between the positive and negative categories in the data, thereby exhibiting severe imbalance.



(**a**) Gantry topology prior to optimization (**b**) Gantry topology after optimization.

Figure 7. Comparison diagram of gantry topology pre- and post-LightGBM model generation.

After applying the LightGBM model to generate the gantry topology, we successfully identified 2819 legitimate topologies from these candidates. As shown in Figure 7b, the distribution of gantry topology more accurately reflected reality and was closer to the distribution of highways. In this case, the model's accuracy increased to 0.976, precision increased to 0.980, and the F1 score increased to 0.986. Although the recall rate decreased to 0.991, this actually reflected the model's improved ability to distinguish between positive and negative categories, rather than simply predicting all samples as positive. It also demonstrated that our model could maintain high precision while still achieving a high recall rate. This significant performance improvement fully demonstrated the effectiveness of the proposed method.

5. Conclusions

In the domain of Electronic Toll Collection (ETC) gantry topology dynamic updating on highways, this research has instituted the following pivotal contributions and advancements:

This study introduces an innovative methodology for dynamic updating of highway gantry topology predicated on ETC transaction data. This effectively ameliorates prevalent inaccuracies in topology data within vehicle transaction records, thus rectifying the existing predicament of solely charging based on minimum distance due to the inability to compute fees according to actual traversed distances. By extrapolating potential topologies and systematically eliminating erroneous variants, we have furnished a robust and reliable data source for the genesis and iterative refinement of gantry topology. In tandem with the integration of the Amap API and QGIS mapping analytics, we have substantiated the veracity of our candidate topologies, thereby safeguarding the precision of the resultant gantry topology. We have employed the LightGBM model to facilitate the dynamic updating of the gantry topology. Empirical evidence suggests that this approach yields commendable outcomes, registering an accuracy rate of 97.6%, thereby satisfactorily meeting the requisites for dynamic updating of ETC gantry topology on highways. This research promulgates a universally applicable methodology and framework for dynamically updating highway ETC gantry topology, underscoring its extensive applicability and scalability. In practical implementation, the methodologies propounded by this research can be fine-tuned and

optimized to accommodate actual needs, thereby catering to the varying requirements for updating ETC gantry topology in diverse scenarios.

In summary, this research has heralded groundbreaking methods and techniques for the dynamic generation of highway ETC gantry topology, providing significant reinforcement for highway management and road network analytics. Concurrently, it lays a formidable foundation for the evolution of intelligent transportation systems and enhancement of the overall quality of transportation services.

Author Contributions: Conceptualization, F.Z. and W.W.; methodology, W.W.; software, W.W., Q.C. and R.S.; validation, F.Z., W.W. and Q.C.; formal analysis, W.W. and F.Z.; investigation, W.W.; resources, F.Z.; data curation, W.W. and F.Z.; writing—original draft preparation, W.W. and Q.C.; writing—review and editing, F.Z., W.W. and F.G.; visualization, W.W. and R.S.; supervision, F.Z. and F.G.; project administration, F.Z. and W.W.; funding acquisition, F.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the National Natural Science Foundation of China (Funding number: 41971340), the 2020 Fujian Province "Belt and Road" Technology Innovation Platform (Funding number: 2020D002), the Provincial Candidates for the Hundred, Thousand and Ten Thousand Talent of Fujian (Funding number: GY-Z19113), the Municipal Science and Technology project (Funding number: GY-Z22006, GY-Z220230), the Open Fund project (Funding number: KF-X1902, KF-19-22001), the Patent Grant project (Funding number: GY-Z20074), and the Crosswise project (Funding number: GY-H-21021, GY-H-20077).

Data Availability Statement: The ETC transaction data utilized in this study were obtained from Fujian Expressway Information Technology Co., Ltd. Restrictions apply to the availability of these data, which were used under license for this study and are not publicly available. Data are available from the authors with the permission of Fujian Expressway Information Technology Co., Ltd. All data processing and analyses were conducted in compliance with relevant data protection and privacy laws. No individual or personal data were used in this study.

Conflicts of Interest: The authors declare no conflict of interest.

Glossary

Acronym	Full Form
AB	Adaptive Boosting
API	Application Programming Interface
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DT	Decision Tree
ETC	Electronic Toll Collection
GBDT	Gradient Boosting Decision Tree
GB	Gradient Boosting
GPC	Gaussian Process Classifier
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
KDE	Kernel Density Estimation
KNN	k-Nearest Neighbors
LCSS	Longest Common Subsequence
LDA	Linear Discriminant Analysis
LightGBM	Light Gradient Boosting Machine
LR	Logistic Regression
MTC	Manual Toll Collection
NB	Naive Bayes Classifier
NSR	Normalized Start Rate
OBU	On-Board Unit
QDA	Quadratic Discriminant Analysis
RF	Random Forest
RSU	Road Side Unit
SGD	Stochastic Gradient Descent

SVM	Support Vector Machine
TD	Topology Distance
TTV	Topology Traffic Volume
XGB	XGBoost (Extreme Gradient Boosting)
5	0.9763

Appendix A

In this study, we used the default parameter settings of various models for comparison. The following is a detailed list of the individual model parameters:

Table A1. LR (Logistic Regression) parameter settings.

Parameter	Description	Value
C	Inverse regularization	1
class_weight	Class weights	None
dual	Dual formulation	False
fit_intercept	Add constant to function	True
intercept_scaling	For solver 'liblinear'	1
l1_ratio	Elastic-Net mixing	None
max_iter	Max iterations	100
multi_class	Multiclass option	Auto
n_jobs	CPU cores for parallel	None
penalty	Norm in penalization	L2
random_state	Random number seed	None
solver	Optimization algorithm	Lbfgs
tol	Stopping criteria	0.0001
verbose	Verbose for liblinear	0
warm_start	Reuse previous solution	False

Table A2. NB (Naive Bayes) parameter settings.

Parameter	Description	Value
priors	Prior probabilities	None
var_smoothing	Portion of the largest variance	1×10^{-9}
priors	Prior probabilities	None

Table A3. LDA (Linear Discriminant Analysis) parameter settings.

Parameter	Description	Value
covariance_estimator	Covariance estimator	None
n_components	Number of components	None
priors	Prior probabilities	None
shrinkage	Shrinkage parameter	None
solver	Solver for computation	Svd
store_covariance	If True, compute covariance	False
tol	Tolerance for stopping criteria	0.0001

Table A4. KNN (Decision Tree) parameter settings.

Parameter	Description	Value
algorithm	Algorithm used	Auto
leaf_size	Leaf size	30
metric	Distance metric	Minkowski
metric_params	Metric params	None
n_jobs	Num of jobs	None
n_neighbors	Num of neighbors	5
p	Power parameter	2
weights	Weight function	Uniform

Table A5. DT (Decision Tree) parameter settings.

Parameter	Description	Value
ccp_alpha	Cost complexity pruning	0
class_weight	Class weights	None
criterion	Criterion to split	Gini
max_depth	Max depth of tree	None
max_features	Max features for split	None
max_leaf_nodes	Max leaf nodes	None
min_impurity_decrease	Node impurity decrease	0
min_samples_leaf	Min samples at leaf	1
min_samples_split	Min samples to split	2
min_weight_fraction_leaf	Min weight fraction	0
random_state	Random seed	None
splitter	Split strategy	Best

Table A6. RF (Random Forest) parameter settings.

Parameter	Description	Value
bootstrap	Bootstrap samples	True
ccp_alpha	Cost complexity pruning	0
class_weight	Class weights	None
criterion	Split criterion	None
max_depth	Max tree depth	None
max_features	Max features	None
max_leaf_nodes	Max leaf nodes	None
max_samples	Max samples	None
min_impurity_decrease	Min impurity decrease	0
min_samples_leaf	Min samples at leaf	1
min_samples_split	Min samples to split	2
min_weight_fraction_leaf	Min weight fraction	0
n_estimators	Num of trees	100
n_jobs	Num of jobs	None
oob_score	OOB score	False
random_state	Random seed	None
verbose	Logging level	0
warm_start	Reuse previous solution	False

Table A7. SVM (Support Vector Machines) parameter settings.

Parameter	Description	Value
C	Penalty parameter	1
break_ties	Break ties	False
cache_size	Cache size	200
class_weight	Class weights	None
coef0	Kernel coef	0
decision_function_shape	Decision function	Ovr
degree	Kernel degree	3
gamma	Kernel coef	Scale
kernel	Kernel type	Rbf
max_iter	Max iterations	−1
probability	Estimate prob	True
random_state	Random seed	None
shrinking	Use shrinking	True
tol	Tolerance	0.001
verbose	Verbose	False

Table A8. GB (Gradient Boosting) parameter settings.

Parameter	Description	Value
ccp_alpha	Pruning parameter	0
criterion	Split criterion	Friedman_mse
init	Initial estimator	None
learning_rate	Learning rate	0.1
loss	Loss function	Deviance
max_depth	Max depth	3
max_features	Max features	None
max_leaf_nodes	Max leaf nodes	None
min_impurity_decrease	Min impurity decrease	0
min_samples_leaf	Min samples at leaf	1
min_samples_split	Min samples to split	2
min_weight_fraction_leaf	Min weight fraction	0
n_estimators	Num of estimators	100
n_iter_no_change	Iterations no change	None
random_state	Random seed	None
subsample	Subsample fraction	1
tol	Tolerance	0.0001
validation_fraction	Validation fraction	0.1
verbose	Verbose	0
warm_start	Reuse previous solution	False

Table A9. AB (AdaBoost) parameter settings.

Parameter	Description	Value
algorithm	Algorithm type	Samme.r
base_estimator	Base estimator	None
learning_rate	Learning rate	1
n_estimators	Num of estimators	50
random_state	Random seed	None
verbose	Verbose for liblinear	0
warm_start	Reuse previous solution	False

Table A10. XGB (XGBoost) parameter settings.

Parameter	Description	Value
use_label_encoder	Use label encoder	False
enable_categorical	Categorical data	False
eval_metric	Evaluation metric	Logloss
objective	Objective function	Binary:logistic
n_estimators	Num of estimators	100

In order to not cause redundancy, we did not put the value of empty parameters into the table, The parameters set to null values are: booster, colsample_bylevel, colsample_bynode, colsample_bytree, gamma, gpu_id, importance_type, interaction_constraints, learning_rate, max_delta_step, max_depth, min_child_weight, missing, monotone_constraints, n_jobs, num_parallel_tree, predictor, random_state, reg_alpha, reg_lambda, scale_pos_weight, subsample, tree_method, validate_parameters, verbosity.

Table A11. QDA (Quadratic Discriminant Analysis) parameter settings.

Parameter	Description	Value
priors	Class priors	None
reg_param	Regularization	0
store_covariance	Store covariance	False
tol	Tolerance	0.0001

Table A12. GPC (Gaussian Process Classifier) parameter settings.

Parameter	Description	Value
copy_X_train	Copy training data	True
kernel	Kernel function	None
max_iter_predict	Max iterations	100
multi_class	Multi-class strategy	One_vs_rest
n_jobs	Num of jobs	None
n_restarts_optimizer	Num of restarts	0
optimizer	Optimizer	Fmin_l_bfgs_b
random_state	Random seed	None
warm_start	Reuse previous solution	False

Table A13. SGD (Stochastic Gradient Descent) parameter settings.

Parameter	Description	Value
alpha	Regularization param	0.0001
average	Average coef	False
class_weight	Class weights	
early_stopping	Early stopping	False
epsilon	Epsilon	0.1
eta0	Initial learning rate	0
fit_intercept	Fit intercept	True
l1_ratio	L1 ratio	0.15
learning_rate	Learning rate	Optimal
loss	Loss function	Hinge
max_iter	Max iterations	1000
n_iter_no_change	Iterations no change	5
n_jobs	Num of jobs	None
penalty	Penalty	L2
power_t	Power t	0.5
random_state	Random seed	None
shuffle	Shuffle	True
tol	Tolerance	0.001
validation_fraction	Validation fraction	0.1
verbose	Verbose	0
warm_start	Reuse previous solution	False

Table A14. Linear SVM parameter settings.

Parameter	Description	Value
C	Regularization param	1
class_weight	Class weights	None
dual	Dual formulation	True
fit_intercept	Fit intercept	True
intercept_scaling	Intercept scaling	1
loss	Loss function	Squared_hinge
max_iter	Max iterations	1000
multi_class	Multi-class strategy	None
penalty	Penalty	L2
random_state	Random seed	None
tol	Tolerance	0.0001

References

1. Ministry of Transport, National Development and Reform Commission, Ministry of Finance. Notice on Issuing the Implementation Plan for the Full Promotion of Differentiated Toll Collection on Highways. Jiaogongluhan No. 228. 2021. Available online: https://www.gov.cn/zhengce/zhengceku/2021-06/15/content_5617919.htm (accessed on 30 July 2023).
2. Cai, Q.; Yi, D.; Zou, F.; Wang, W.; Luo, G.; Cai, X. An Arch-Bridge Topology-Based Expressway Network Structure and Automatic Generation. *Appl. Sci.* **2023**, *13*, 5031. [CrossRef]
3. Fujian Provincial State-Owned Assets Supervision and Administration Commission. Fujian: ETC Usage Rate Ranks First in the Country. 12 December 2019. Available online: http://www.sasac.gov.cn/n2588025/n2588129/c13072896/content.html (accessed on 30 July 2023).
4. Amap. Web Service API. Amap API. Available online: https://lbs.amap.com/api/webservice/summary/ (accessed on 27 July 2023).
5. QGIS Development Team. QGIS Desktop User Guide/Manual (QGIS 3.28). Available online: https://docs.qgis.org/3.28/en/docs/user_manual/ (accessed on 27 July 2023).
6. Kass, M.; Witkin, A.; Terzopoulos, D. Snakes: Active contour models. *Int. J. Comput. Vis.* **1988**, *1*, 321–331. [CrossRef]
7. Péteri, R.; Ranchin, T. Extraction of network-like structures using a multiscale representation. *IEEE Geosci. Remote Sens. Lett.* **2005**, *2*, 402–406.
8. Laptev, I.; Caputo, B.; Schuldt, C.; Lindeberg, T. Local velocity-adapted motion events for spatio-temporal recognition. *Comput. Vis. Image Underst.* **2004**, *108*, 207–229. [CrossRef]
9. Gruen, A.; Li, H. Linear feature extraction with dynamic programming and Globally Least Squares. *ISPRS J. Photogramm. Remote Sens.* **1995**, *50*, 23–30.
10. Mokhtarzade, M.; Zoej, M.V. Road detection from high-resolution satellite images using artificial neural networks. *Int. J. Appl. Earth Obs. Geoinf.* **2007**, *9*, 32–40. [CrossRef]
11. Yager, K.; Sowmya, A. Road detection from aerial images using SVMs. *Mach. Vis. Appl.* **2008**, *19*, 261–274.
12. Mnih, V.; Hinton, G.E. *Learning to Detect Roads in High-Resolution Aerial Images*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 210–223.
13. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
14. Saito, S.; Yamashita, T.; Aoki, Y. Multiple object extraction from aerial imagery with convolutional neural networks. *Electron. Imaging* **2016**, *2016*, 1–9. [CrossRef]
15. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3431–3440.
16. Maggiori, E.; Tarabalka, Y.; Charpiat, G.; Alliez, P. Fully convolutional neural networks for remote sensing image classification. In Proceedings of the 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Beijing, China, 10–15 June 2016; pp. 5071–5074.
17. Ševo, I.; Avramović, A. Convolutional Neural Network Based Automatic Object Detection on Aerial Images. *IEEE Geosci. Remote Sens. Lett.* **2016**, *13*, 740–744. [CrossRef]
18. Volpi, M.; Tuia, D. Dense Semantic Labeling of Subdecimeter Resolution Images with Convolutional Neural Networks. *IEEE Trans. Geosci. Remote Sens.* **2017**, *55*, 881–893. [CrossRef]
19. Maggiori, E.; Tarabalka, Y.; Charpiat, G.; Alliez, P. Convolutional neural networks for large-scale remote-sensing image classification. *IEEE Trans. Geosci. Remote Sens.* **2017**, *55*, 645–657. [CrossRef]
20. Keramitsoglou, I.; Kontoes, C.; Sifakis, N.; Konstantinidis, P.; Fitoka, E. Deep learning for operational land cover mapping using Sentinel-2 data. *J. Appl. Remote Sens.* **2020**, *14*, 014503.
21. Wagstaff, K.; Cardie, C.; Rogers, S.; Schroedl, S. Constrained k-means clustering with background knowledge. In Proceedings of the Eighteenth International Conference on Machine Learning (ICML), San Francisco, CA, USA, 28 June–1 July 2001; Volume 1, pp. 577–584.

22. Schroedl, S.; Wagstaff, K.; Rogers, S.; Langley, P.; Wilson, C. Mining GPS traces for map refinement. *Data Min. Knowl. Discov.* **2004**, *9*, 59–87. [CrossRef]
23. Worrall, S.; Nebot, E. Automated process for generating digitised maps through GPS data compression. In Proceedings of the Australasian Conference on Robotics and Automation (ACRA), Brisbane, Australia, 10–12 December 2007; Volume 6.
24. Sasaki, Y.; Yu, J.; Ishikawa, Y. Road segment interpolation for incomplete road data. In Proceedings of the IEEE International Conference on Big Data and Smart Computing, Kyoto, Japan, 27 February–2 March 2019; pp. 1–8.
25. Edelkamp, S.; Schrödl, S. Route planning and map inference with global positioning traces. In *Computer Science in Perspective*; Springer: Berlin/Heidelberg, Germany, 2003; pp. 128–151.
26. Chen, C.; Lu, C.; Huang, Q.; Yang, Q.; Gunopulos, D.; Guibas, L. City-scale map creation and updating using GPS collections. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 1465–1474.
27. Huang, J.; Zhang, Y.; Deng, M.; He, Z. Mining crowdsourced trajectory and geo-tagged data for spatial-semantic road map construction. *Trans. GIS* **2022**, *26*, 735–754. [CrossRef]
28. Silverman, B.W. *Density Estimation for Statistics and Data Analysis*; Routledge: Abingdon, UK, 2018.
29. Fu, Z.; Fan, L.; Sun, Y.; Tian, Z. Density adaptive approach for generating road network from GPS trajectories. *IEEE Access* **2020**, *8*, 51388–51399. [CrossRef]
30. Uduwaragoda, E.; Perera, A.S.; Dias, S.A.D. Generating lane level road data from vehicle trajectories using kernel density estimation. In Proceedings of the 16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013), The Hague, The Netherlands, 6–9 October 2013; pp. 384–391.
31. Kuntzsch, C.; Sester, M.; Brenner, C. Generative models for road network reconstruction. *Int. J. Geogr. Inf. Sci.* **2016**, *30*, 1012–1039. [CrossRef]
32. Neuhold, R.; Haberl, M.; Fellendorf, M.; Pucher, G.; Dolancic, M.; Rudigier, M.; Pfister, J. Generating a lane-specific transportation network based on floating-car data. In *Advances in Human Aspects of Transportation*; Springer: Cham, Switzerland, 2017; pp. 1025–1037.
33. Huang, Y.; Xiao, Z.; Yu, X.; Wang, D. Road network construction with complex intersections based on sparsely sampled private car trajectory data. *ACM Trans. Knowl. Discov. Data (TKDD)* **2019**, *13*, 1–28. [CrossRef]
34. Deng, M.; Huang, J.; Zhang, Y.; Liu, H.; Tang, L.; Tang, J.; Yang, X. Generating urban road intersection models from low-frequency GPS trajectory data. *Int. J. Geogr. Inf. Sci.* **2018**, *32*, 2337–2361. [CrossRef]
35. Wu, J.; Zhu, Y.; Ku, T.; Wang, L. Detecting Road Intersections from Coarse-gained GPS Traces Based on Clustering. *J. Comput.* **2013**, *8*, 2959–2965. [CrossRef]
36. Xie, X.; Philips, W. Road intersection detection through finding common sub-tracks between pairwise GNSS traces. *ISPRS Int. J. Geo-Inf.* **2017**, *6*, 311. [CrossRef]
37. Fathi, A.; Krumm, J. Detecting road intersections from GPS traces. In Proceedings of the International Conference on Geographic Information Science, Zurich, Switzerland, 14–17 September 2010; Springer: Berlin/Heidelberg, Germany, 2010; pp. 56–69.
38. Karagiorgou, S.; Pfoser, D. On vehicle tracking data-based road network generation. In Proceedings of the 20th International Conference on Advances in Geographic Information Systems, Redondo Beach, CA, USA, 6–9 November 2012; pp. 89–98.
39. Pu, M.; Mao, J.; Du, Y.; Shen, Y.; Jin, C. Road intersection detection based on direction ratio statistics analysis. In Proceedings of the 2019 20th IEEE International Conference on Mobile Data Management (MDM), Hong Kong, China, 10–13 June 2019; pp. 288–297.
40. Zhao, L.; Mao, J.; Pu, M.; Liu, G.; Jin, C.; Qian, W.; Zhou, A.; Wen, X.; Hu, R.; Chai, H. Automatic calibration of road intersection topology using trajectories. In Proceedings of the 2020 IEEE 36th International Conference on Data Engineering (ICDE), Kuala Lumpur, Malaysia, 20–24 April 2020; pp. 1633–1644.
41. Qing, R.; Liu, Y.; Zhao, Y.; Liao, Z.; Liu, Y. Using feature interaction among GPS Data for road intersection detection. In Proceedings of the 2nd International Workshop on Human-Centric Multimedia Analysis, Cheng Du, China, 20–24 October 2021; pp. 31–37.
42. Liu, Y.; Qing, R.; Zhao, Y.; Liao, Z. Road Intersection Recognition via Combining Classification Model and Clustering Algorithm Based on GPS Data. *ISPRS Int. J. Geo-Inf.* **2022**, *11*, 487. [CrossRef]
43. Ministry of Transport of the People's Republic of China. Vehicle Classification of the Toll for Highway (JT/T 489-2019). 2019. Available online: https://jtst.mot.gov.cn/kfs/file/read/0de9ee528422ee3a99ff87b1c1295e8e (accessed on 30 July 2023).
44. Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; Liu, T.Y. Lightgbm: A highly efficient gradient boosting decision tree. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 3149–3157. [CrossRef]
45. Meng, Q.; Ke, G.; Wang, T.; Chen, W.; Ye, Q.; Ma, Z.M.; Liu, T.Y. A communication-efficient parallel algorithm for decision tree. *Adv. Neural Inf. Process. Syst.* **2016**, *29*, 1279–1287. [CrossRef]