International Journal of
*Geo-Information*

MDPI

*Article*

# Urban Road Lane Number Mining from Low-Frequency Floating Car Data Based on Deep Learning

**Xiaolong Li [1,2,†], Yun Zhang [3,*,†], Longgang Xiang [4] and Tao Wu [5]**

[1] Key Laboratory of Mine Environmental Monitoring and Improving around Poyang Lake of Ministry of Natural Resources, East China University of Technology, Nanchang 330013, China; lixiaolong@ecut.edu.cn
[2] CNNC Engineering Research Center of 3D Geographic Information, East China University of Technology, Nanchang 330013, China
[3] China Railway Water Resources and Hydropower Planning and Design Group, Nanchang 330001, China
[4] State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan 430079, China; geoxlg@whu.edu.cn
[5] School of Geographical Sciences, Hunan Normal University, Changsha 410012, China; taowu@hunnu.edu.cn
* Correspondence: yrjnszzyy@163.com
† These authors contributed equally to this work.

**Abstract:** Lane-level road information is especially crucial now that high-precision navigation maps are in more demand. Road information may be obtained rapidly and affordably by mining floating vehicle data (FCD). A method is proposed to extract the number of lanes on urban roads by combining deep learning and low-frequency FCD. Initially, the FCD is cleaned using the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) clustering technique. Then, the FCD is split into three categories based on the typical urban road types: one-way one-lane, one-way two-lane, and one-way three-lane, and the deep learning sample data is created using segmentation, rotation, and gridding. Lastly, the number of urban road lanes is obtained by training and predicting the sample data using the LeNet-5 model. The number of urban road lanes was effectively identified from the low-frequency FCD with a detection accuracy of 92.7% through the cleaning and classification of Wuhan FCD. Urban roads can be efficiently covered by the FCD on a regular basis, and lane information can be efficiently collected using deep learning techniques. This method can be used to generate and update lane number information for high-precision navigation maps.

**Keywords:** FCD; number of lanes; DBSCAN clustering algorithm; deep learning; LeNet-5 model

check for updates

**Citation:** Li, X.; Zhang, Y.; Xiang, L.; Wu, T. Urban Road Lane Number Mining from Low-Frequency Floating Car Data Based on Deep Learning. *ISPRS Int. J. Geo-Inf.* **2023**, *12*, 467. https://doi.org/10.3390/ijgi12110467

Academic Editors: Maria Antonia Brovelli and Wolfgang Kainz

Received: 17 July 2023
Revised: 5 November 2023
Accepted: 8 November 2023
Published: 18 November 2023

## 1. Introduction

Lane-level road maps serve as the foundation for automatic driverless systems and intelligent assisted driving systems. They facilitate autonomous vehicle navigation and lane-based traffic analyses. At present, the primary data sources for generating lane-level road maps encompass high-resolution images, on-board/on-board Lidar point clouds, and high-frequency differential GPS trajectories [1]. The acquisition of these data necessitates specialized equipment and procedures, albeit resulting in high-precision lane-level road maps. Challenges such as high costs, protracted collection periods, and an inability to reflect the most recent updates to the road surface remain.

GPS positioning devices are commonly installed in the majority of buses and taxis operating in contemporary urban environments, meticulously recording the time and location of each vehicle's operation. This data, referred to as FCD, is rich in geographical information and has wide-ranging applications in various fields, such as traffic management, cartographic navigation, and urban planning. Harvesting road information from FCD exhibits characteristics of low cost, broad coverage, and real-time functionality [2,3].

Road intersections, centerlines, carriageways, and lanes can be extracted from FCD [4–6]. The determination of the number of lanes has emerged as a current research hotspot.

Uduwaragoda et al. [7] employed kernel density estimation to detect the number of lanes and the position of lane lines from vehicle GPS trajectory data. Chen and Krumm [8] introduced the concept of utilizing Gaussian mixture models to model GPS trajectory distributions across multiple traffic lines and introduced a novel regulator to enhance the lane calculation performance of the new Gaussian mixture model compared to the traditional one. Tang's research group [9,10] analyzed the characteristics of FCD and employed the density clustering method based on Delaunay triangulation to optimize the data. They constructed a Naïve Bayesian classifier by detecting the coverage width of the FCD and its distribution in the road cross-section. This Naïve Bayesian classification method was then utilized to determine the number of lanes in the target road section. Tang's team [11,12] further employed the constrained Gaussian mixture model to simulate the distribution of FCD on the road surface. They compared the advantages and disadvantages of the model under different Gaussian component combinations and selected the number of Gaussian components corresponding to the optimal model for the number of lanes. Yang et al. [13] utilized fuzzy logic to match GPS traces with the lane-level road network. The fuzzy membership degree between GPS data and lane segments was employed to quantify matching, leading to subsequent detection changes. Li et al. [14] employed the gradient lifting decision tree algorithm to identify the expanded lanes at intersections and non-intersections in low-frequency FCD. Arman and Tampere [15] introduced a dissimilarity matrix based on Frechet distance for road center-line construction and employed the Gaussian mixture method for lane estimation, ensuring that the results were unaffected by GPS density distribution on the lane. Shu et al. [16] utilized the least squares estimate to constrain the Gaussian mixture model and developed an efficient and accurate lane-level road information extraction algorithm. Fan et al. [17] established a lane extraction model using the weighted constrained Gaussian mixture model and hidden Markov model to estimate lane parameters, such as lane counts and centerlines, on each road cross section. In summary, the aforementioned literature has reported various methods for extracting the number of lanes from FCD, including kernel density estimation, naive Bayesian, different Gaussian mixture models, and fuzzy logic, with an overall accuracy of less than 90%. The low detection accuracy of the number of lanes affects the effectiveness of practical applications.

In recent years, the rapid advancement of deep learning has led to numerous fruitful results in data mining, natural language processing, computer vision, and other related fields. Scholars have applied deep learning techniques in the field of road information extraction using crowdsourcing data, primarily focusing on video or image analysis [18]. Alternatively, numerous researchers have converted trajectory data into binary images and extracted road centerlines through digital image processing techniques such as dilation and refinement [19,20]. However, no current method exists that combines low-frequency FCD and deep learning techniques to extract the number of lanes.

In this study, a method integrating low-frequency FCD and deep learning techniques is proposed to estimate the number of lanes on urban roads. Firstly, the original FCD data is cleaned to eliminate abnormal points and noise points with significant positional deviations. Subsequently, deep learning samples are generated through segmentation, classification, and gridding operations. Finally, we endeavored to train and test the sample data using the LeNet-5 deep learning model to obtain the number of lanes at the corresponding location. The primary contributions of this research can be summarized as follows:

- After classification and segmentation, we grid and normalize the low-frequency FCD trajectory point data to generate standardized deep learning sample data.
- We employed the LeNet-5 model to train and test the sample data, aiming to enhance the accuracy of lane number detection. This marks the first instance of utilizing a deep learning model to extract lane numbers exclusively from low-frequency FCD trajectory points.

The rest of the paper is organized as follows: The detection process of lane number information on urban roads is introduced in Section 2. In Section 3, we clean the raw FCD and use the DBSCAN clustering algorithm to eliminate drift trajectory points. The method of constructing deep learning samples is introduced in Section 4. Calculation and analysis of lane classification based on deep learning are introduced in Section 5. Finally, we conclude the paper in Section 6.

## 2. Detection Process of Lane Number Information on Urban Roads

The present study utilizes Wuhan taxi trajectory data as its primary dataset to investigate the extraction of urban road lane number information. In order to facilitate the investigation of taxi theft and robbery cases in Wuhan City, it is mandated that each taxi periodically transmits its current longitude and latitude coordinates to the backend server every 40 s. Almost every city in China equips its taxis with such positioning information transmission devices. The frequency of location data automatically sent back by taxis ranges from 15 s to a few minutes, and such data is typically stored in traffic management departments or police departments, constituting a distinct low-frequency FCD. Figure 1 depicts the distribution of sectional low-frequency FCD trajectory points across the road network in Wuhan.

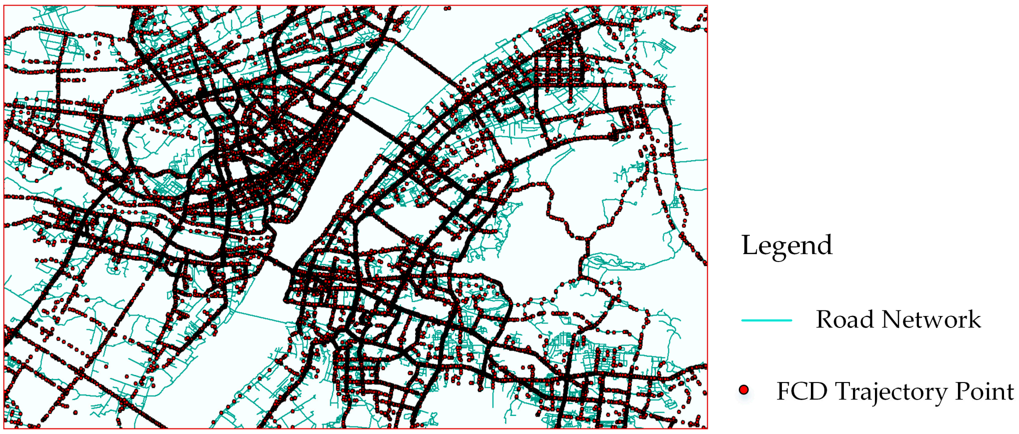


**Figure 1.** The coverage of FCD trajectory points within the road network of Wuhan City.

The detection of the number of lanes on urban roads is primarily divided into seven stages. The initial stage involves matching the FCD with the vector data of the Wuhan road network. The second stage involves preprocessing the FCD to eliminate data that is out of bounds, abnormal, duplicate, or incomplete. The third stage employs the DBSCAN clustering algorithm to remove drift trajectory points and further clean the FCD. The fourth stage classifies the cleaned FCD into three categories based on the types of roads covered: one-way one-lane, one-way two-lane, and one-way three-lane. The fifth stage transforms the categorized FCD into deep learning sample data. The sixth stage utilizes the sample data to train a convolutional neural network model. Upon completion of the training, the converted FCD is classified to obtain information regarding the number of urban road lanes. In this study, the LeNet-5 model is employed to train the FCD, enabling classification and prediction of the FCD to obtain the number of urban road lanes. The specific process is illustrated in Figure 2.

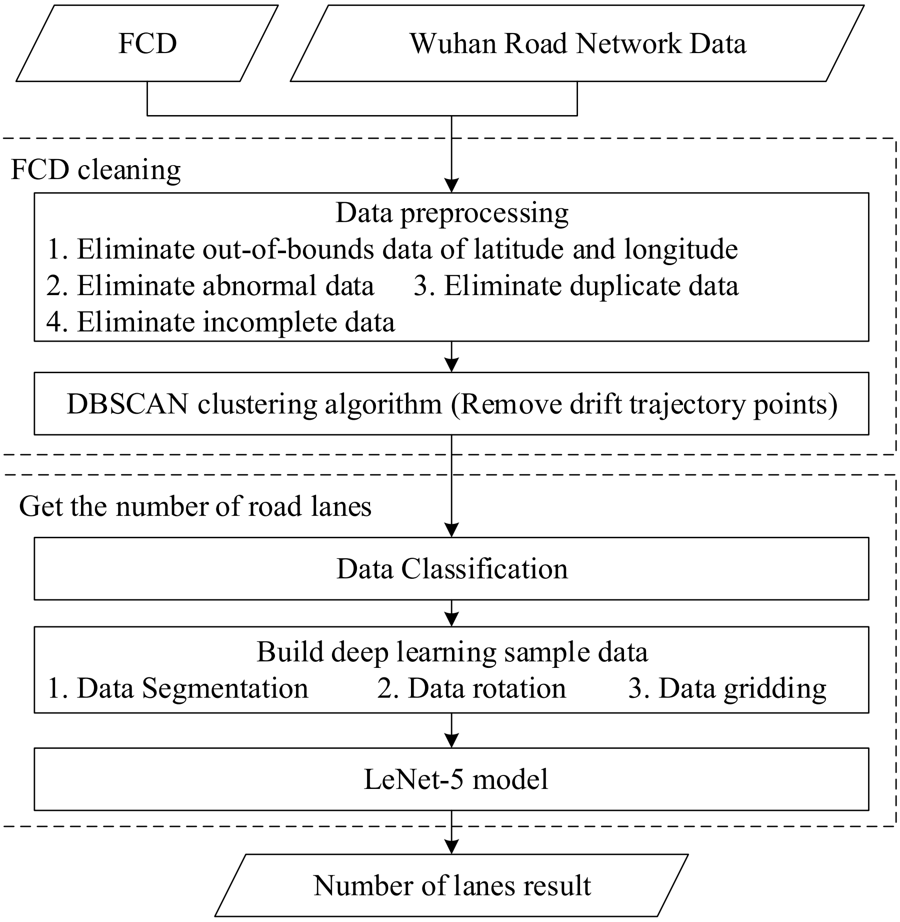

FCD
Wuhan Road Network Data
FCD cleaning
Data preprocessing
1. Eliminate out-of-bounds data of latitude and longitude
2. Eliminate abnormal data
3. Eliminate duplicate data
4. Eliminate incomplete data
DBSCAN clustering algorithm (Remove drift trajectory points)
Get the number of road lanes
Data Classification
Build deep learning sample data
1. Data Segmentation
2. Data rotation
3. Data gridding
LeNet-5 model
Number of lanes result

**Figure 2.** Detection process of urban road lane number.

## 3. FCD Cleaning

The original FCD exhibits defects, such as position drift and missing data, due to the influence of equipment and signals. Consequently, a comprehensive cleaning process is implemented to rectify these issues. This method is essentially divided into two stages: the first stage involves preprocessing the FCD, while the second stage employs the DBSCAN clustering algorithm to eliminate drift trajectory points from the pretreated FCD.

Affected by passenger flow, the data volume of floating cars covered by different roads on the same day varies, as does the data volume of floating cars covered by the same road on working days and rest days. Based on previous experience [14], to ensure that the distribution density and width of FCD on the road can fully express the information of the number of road lanes, a cleaning experiment was conducted on FCD from 55 roads in the urban area of Wuhan. The data collection cycle was 7 days, and the road types of the 55 selected roads included: two-way two-lane 15 roads, two-way four-lane 22 roads, and two-way six-lane 18 roads (most roads in cities belong to these three types). To ensure the representativeness of the data, these 55 selected roads were evenly distributed across the main urban area of Wuhan City, as shown in Figure 3.

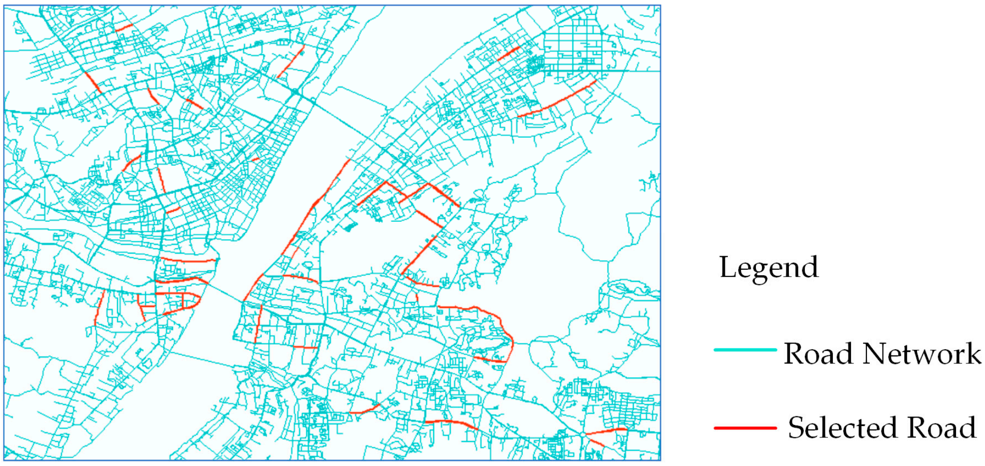


**Figure 3.** Spatial distribution of 55 selected experimental roads.

### 3.1. Data Preprocessing

The primary objective of data preprocessing is to eradicate out-of-bound, outlier, duplicate, and incomplete records from the original FCD. In accordance with the research requirements, four data processing guidelines have been established for FCD [21,22].

(1) Eliminate out-of-bound data. The central urban region of Wuhan is delineated by 113°41′ E–115°05′ E and 29°58′ N–31°22′ N, thus excluding any records falling beyond these coordinates. As illustrated in Table 1, the FCD of car 12,823 surpasses the boundaries of the research zone.

**Table 1.** The original FCD.

| T_TargetID | T_BeijingTime | T_Longitude | T_Latitude | T_Speed | T_Heading | T_Status |
|---|---|---|---|---|---|---|
| 39835 | 2013-08-21-03-03-23 | 114.137515 | 30.561093 | 14.30667 | 79.22 | 0 |
| 12823 | 2013-08-21-03-03-22 | 115.344663 | 32.566431 | 8.525416 | 217.4 | 262144 |
| 15749 | 2013-08-21-03-03-05 | 114.26177 | 30.455383 | 2.106431 | 26.2 | 262145 |
| 15217 | 2013-08-21-03-03-58 | 114.236966 | 30.623555 | 12.85236 | 325.57 | 262144 |
| 15217 | 2013-08-21-03-03-58 | 114.236966 | 30.623555 | 12.85236 | 325.57 | 262144 |
| 15536 | 2013-08-21-03-03-51 | 114.292326 | 30.547933 | null | 278.65 | 262144 |
| … | … | … | … | … | … | … |

The red numbers in Table 1 represent dirty data that has been cleaned.

(2) Eliminate abnormal data. The taxi passenger status is characterized as "0" (empty) and "262144" (occupied), and any values other than these should be removed. As illustrated in Table 1, the abnormal passenger status of the vehicle numbered 15,749 needs to be filtered out.

(3) Eliminate duplicate data. Records containing identical field values lack practical significance and should be removed. As illustrated in Table 1, the vehicle with registration number 15,217 has two identical entries, which represent duplicate data.

(4) Eliminate incomplete data. Missing field values can potentially skew the analysis outcomes. Hence, incomplete data should be removed. As illustrated in Table 1, the speed information for vehicle 15,536 is absent and should be excised.

### 3.2. Using the DBSCAN Clustering Algorithm to Eliminate Drift Trajectory Points

Using the DBSCAN [23] algorithm and employing data density as a similarity index for spatial clustering, it is capable of not only clustering data of arbitrary shape in space but also filtering out noise points [24]. The DBSCAN algorithm primarily requires two parameters: Eps and MinPts, where Eps denotes the scanning radius centered on a point and MinPts represents the minimum number of points contained within the scanning area [25]. The parameter Eps can be calculated using a function, with the MinPts parameter being adjusted accordingly. However, the determination of the MinPts parameter solely relies on personal experience or multiple experiments. According to the experiment in the literature [26], MinPts is set to 1.

The DBSCAN clustering algorithm is implemented in MATLAB to purify the pre-processed FCD. The results are depicted in Figure 4. The blue dot in the figure represents the core point, which necessitates retention, while the red dot signifies the noise point, requiring elimination.

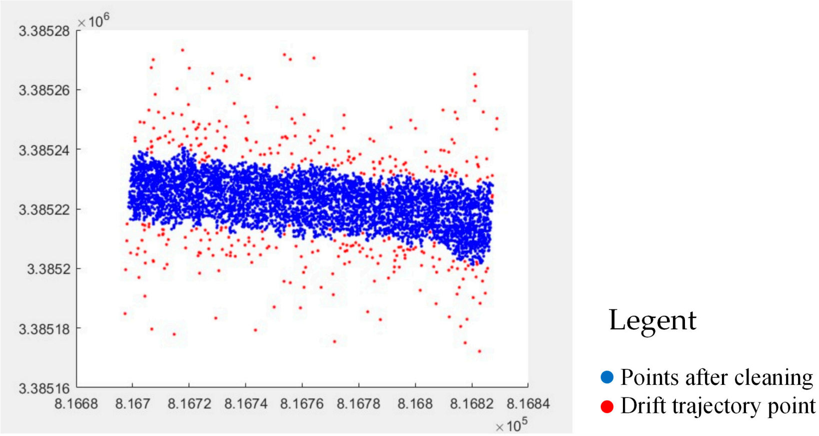


**Figure 4.** Implementing the DBSCAN clustering algorithm to eliminate drift trajectory points.

The application of the DBSCAN method led to the effective cleaning of three distinct types of lanes, as illustrated in Figure 5. The middle black line represents the road's centerline, while the black lines on both sides delineate the road's boundary. This cleaning approach demonstrates promising efficacy and is deemed suitable for a variety of road types.

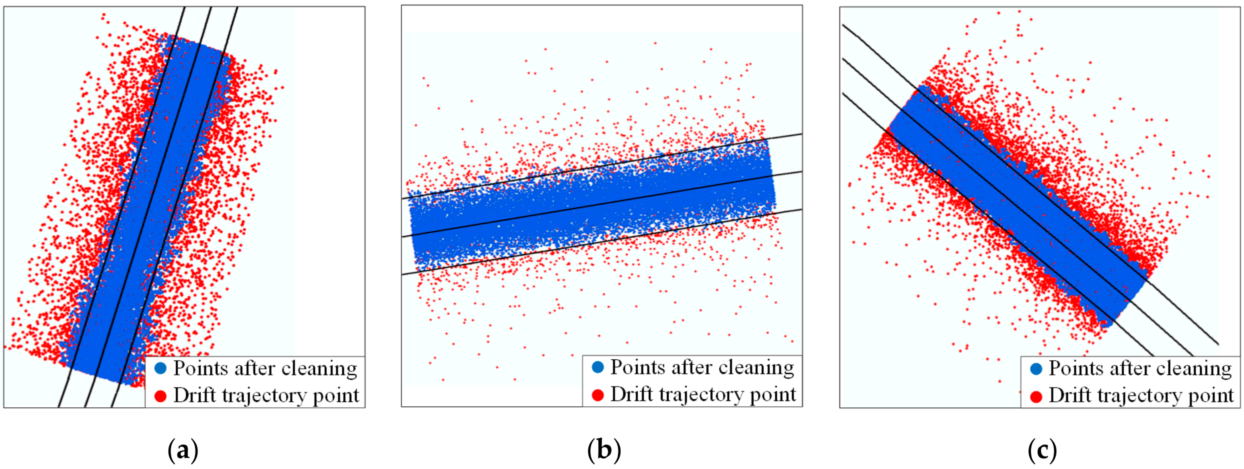


**Figure 5.** Comparison before and after cleaning: (**a**) two-way two-lane; (**b**) two-way four-lane; (**c**) two-way six-lane.

## 4. Construct Depth-Learning Samples

In this study, the LeNet-5 model was employed to classify FCD, yielding more precise information regarding the number of road lanes. Firstly, the cleaned FCD was categorized into three groups: two-way two-lane, two-way four-lane, and two-way six-lane. Subsequently, data conversion was performed, encompassing data segmentation, rotation, and gridding. Consequently, the two-dimensional coordinate point set was transformed into a two-dimensional array, composing a depth learning sample.

### 4.1. Data Segmentation

The cleaned FCD is segmented, and the road is equidistantly divided into several small sections. There are four main steps in data segmentation: first, intercept straight sections and straight sections in curved roads; second, process the FCD in different directions, where the direction angle of the road can be calculated from the Wuhan road network data, and then the FCD on the road can be divided into two types of data with opposite driving directions according to the head direction in the FCD; third, use the least squares method to fit the centerline of urban roads; and fourth, segment the FCD, using the vertical line of the road centerline to segment the FCD equidistantly (the segmentation distance in the experiment is 5 m). As illustrated in Figure 6, the rightmost figure represents the split FCD, with different colors in the figure denoting the FCD in each small section.

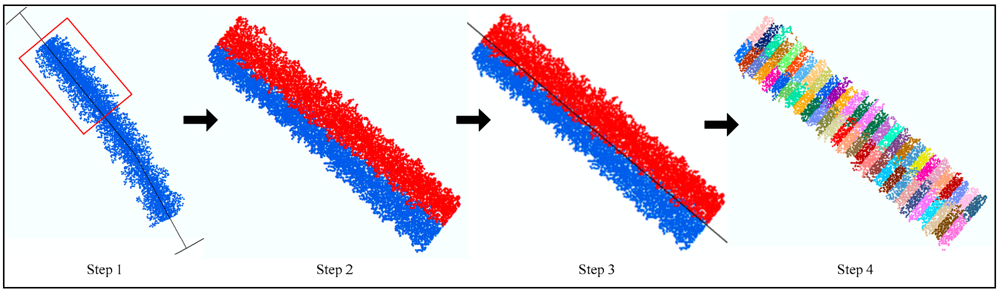


**Figure 6.** FCD segmentation (The dots with different colors in Step 2 and Step 3 represent different driving directions. Different colored dots in Step 4 represent different small sections).

### 4.2. Data Rotation

The intersecting urban road network and the varying driving directions of vehicles on different roads can compromise the classification accuracy of a neural network. To address this issue, the FCD of each road is uniformly rotated towards the due north direction, based on the included angle between the road's fitting centerline and the due north direction. This rotation process consists of two steps: first, the geometric center of the FCD is determined for each small section; second, all data points within the small road section are rotated around this center point, with the rotation angle being the included angle between the road's fitting centerline and the due north direction. As illustrated in Figure 7, the rotation diagrams of some small sections are presented. A total of 1210 small sections are employed in this experiment.

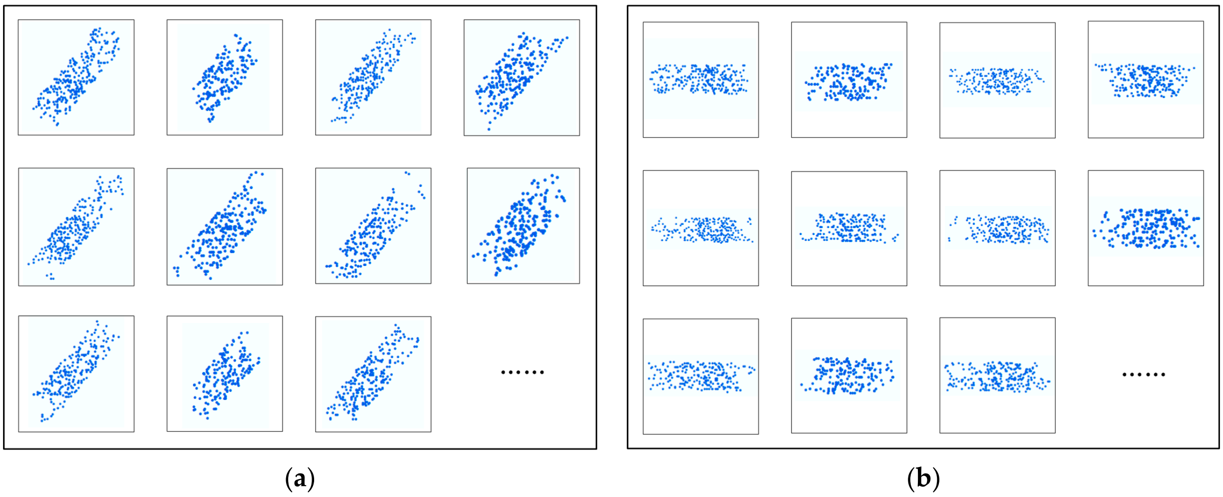
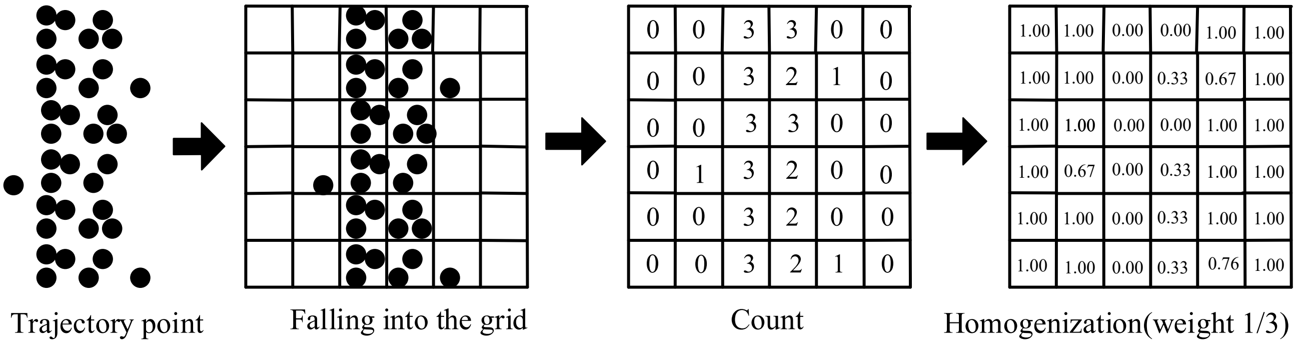


**Figure 7.** FCD rotation: (**a**) before rotation; (**b**) after rotation.

### 4.3. Data Grid

The rotated FCD is transformed into a two-dimensional array through gridding, altering its data format. The gridding process of each small section of the FCD following rotation is fundamentally divided into three stages: initially, a rectangular box of constant size is determined according to the distribution range of the FCD in all small sections, ensuring that all FCD points in each small section are contained within the fixed-size rectangular box. The second stage involves dividing the rectangular box into a 32 × 32 rectangular grid and recording the number of FCD points in each grid. The third stage consists of assigning a weight to achieve homogeneity in the gridded values. This process is elucidated in Figure 8.



| | | | | | |
|---|---|---|---|---|---|
| 0 | 0 | 3 | 3 | 0 | 0 |
| 0 | 0 | 3 | 2 | 1 | 0 |
| 0 | 0 | 3 | 3 | 0 | 0 |
| 0 | 1 | 3 | 2 | 0 | 0 |
| 0 | 0 | 3 | 2 | 0 | 0 |
| 0 | 0 | 3 | 2 | 1 | 0 |

| | | | | | |
|---|---|---|---|---|---|
| 1.00 | 1.00 | 0.00 | 0.00 | 1.00 | 1.00 |
| 1.00 | 1.00 | 0.00 | 0.33 | 0.67 | 1.00 |
| 1.00 | 1.00 | 0.00 | 0.00 | 1.00 | 1.00 |
| 1.00 | 0.67 | 0.00 | 0.33 | 1.00 | 1.00 |
| 1.00 | 1.00 | 0.00 | 0.33 | 1.00 | 1.00 |
| 1.00 | 1.00 | 0.00 | 0.33 | 0.76 | 1.00 |

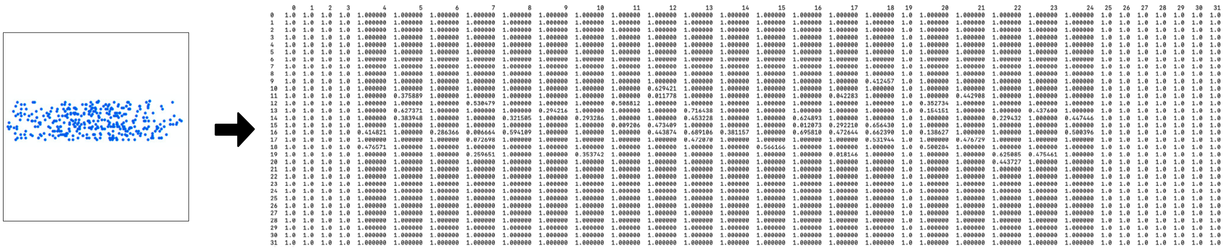
**Figure 8.** Schematic diagram of the FCD grid process.

The objective of homogenization is to proportionally scale the gridded values such that their values are transformed into decimals between 0 and 1. The homogenization formula is outlined as follows:

$$V = 1 - \left(W \times N_{grid\text{-}points}\right) \tag{1}$$

$$W = 1/N_{max\text{-}grid\text{-}points} \tag{2}$$

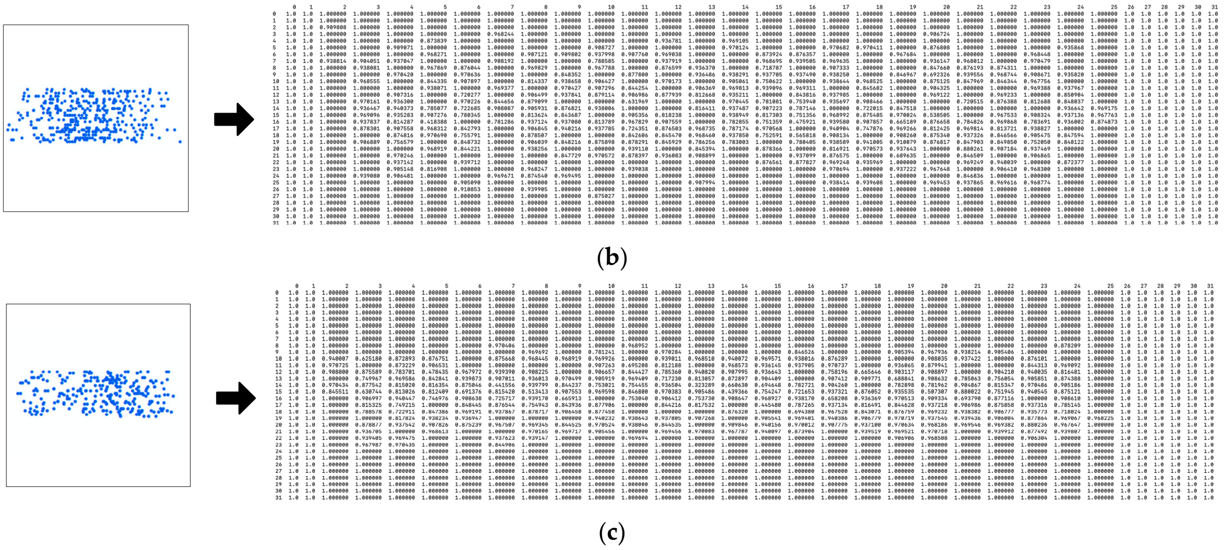
The normalized single grid value is represented by $V$, the weight value by $W$, and the number of points in a single grid by $N_{grid\text{-}points}$. Additionally, $N_{max\text{-}grid\text{-}points}$ denotes the number of points in the grid with the largest density within each small road segment. After data homogenization, a higher number of points in the grid results in a grid value closer to 0, while a lower number of points leads to a value closer to 1. This process enhances the training speed and prediction accuracy of the model, as well as preventing gradient explosion. Figure 9 presents the gridding diagram of FCD for three distinct lane types. In this study, all 1210 small sections were gridded, thus yielding 1210 deep learning samples.



```
      0    1    2    3         4         5         6         7         8         9        10        11        12        13        14        15        16        17        18   19        20        21        22        23        24   25   26   27   28   29   30   31
 0  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
 1  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
 2  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
 3  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
 4  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
 5  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
 6  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
 7  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
 8  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
 9  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  0.412457  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
10  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  0.629421  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
11  1.0  1.0  1.0  1.0  1.000000  0.375889  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  0.011778  1.000000  1.000000  1.000000  1.000000  0.042283  1.000000  1.0  1.000000  0.442908  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
12  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  0.530479  1.000000  1.000000  1.000000  0.508812  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  0.352734  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
13  1.0  1.0  1.0  1.0  1.000000  0.627371  1.000000  1.000000  1.000000  0.294216  1.000000  1.000000  1.000000  0.716438  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  0.154151  1.000000  1.000000  0.437600  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
14  1.0  1.0  1.0  1.0  1.000000  0.383948  1.000000  1.000000  0.321505  1.000000  0.293286  1.000000  1.000000  0.453228  1.000000  1.000000  0.624893  1.000000  1.000000  1.0  1.000000  1.000000  0.229432  1.000000  0.447446  1.0  1.0  1.0  1.0  1.0  1.0  1.0
15  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  0.009206  0.473409  1.000000  1.000000  1.000000  0.012073  0.292210  0.656430  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
16  1.0  1.0  1.0  1.0  0.414821  1.000000  0.286366  0.006664  0.594109  1.000000  1.000000  1.000000  0.443874  0.689106  0.381157  1.000000  0.695810  0.472644  0.662390  1.0  0.138627  1.000000  1.000000  1.000000  0.500396  1.0  1.0  1.0  1.0  1.0  1.0  1.0
17  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  0.072698  1.000000  1.000000  1.000000  1.000000  1.000000  0.472070  1.000000  1.000000  1.000000  1.000000  0.531944  1.0  1.000000  0.476729  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
18  1.0  1.0  1.0  1.0  0.476571  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  0.566166  1.000000  1.000000  1.000000  1.0  0.500284  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
19  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  0.259651  1.000000  1.000000  0.353742  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  0.018146  1.000000  1.0  1.000000  1.000000  0.625085  0.475461  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
20  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.000000  1.000000  0.443727  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
21  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
22  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
23  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
24  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
25  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
26  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
27  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
28  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
29  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
30  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
31  1.0  1.0  1.0  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.000000  1.000000  1.000000  1.000000  1.000000  1.0  1.0  1.0  1.0  1.0  1.0  1.0
```

(**a**)

**Figure 9.** *Cont.*


(**b**)

(**c**)

**Figure 9.** FCD two-dimensional array: (**a**) one-way one-lane; (**b**) one-way two-lane; and (**c**) one-way three-lane.

## 5. Calculation and Analysis of Lane Classification Based on Deep Learning

### *5.1. LeNet-5 Deep Learning Model*

Professor LeCun et al. introduced the LeNet-5 convolutional neural network model in 1998, which was effectively employed in handwritten digit recognition [27,28]. In addition to the input layer, the conventional LeNet-5 architecture consists of seven layers, characterized by the following features: (1) convolution and pooling layers are successively applied; (2) convolution is utilized to extract sample features; (3) the average pooling method is employed as the pooling layer; (4) the sigmoid function serves as the activation function; (5) the sparse connections between layers can diminish computational costs. The LeNet-5 model architecture for lane number classification is shown in Figure 10.

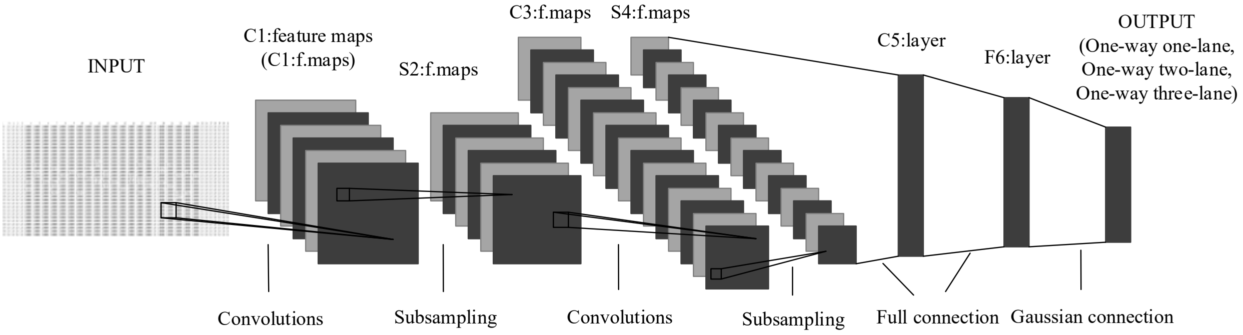


**Figure 10.** LeNet-5 model structure.

According to the distribution characteristics of FCD on the road, the traditional LeNet-5 model has been optimized to enhance the accuracy of lane number information detection. The following aspects have been refined:

(1) Substituting mean pooling with maximum pooling may accentuate the subtle features of the sample and mitigate the computational burden [29].

(2) Substitute the activation function with the ReLU. The ReLU exhibits a faster convergence rate and simpler operation compared to the sigmoid function, effectively mitigating the loss saturation problem induced by excessive training iterations.

(3) Use Dropout [30,31] to suppress overfitting. By discarding neurons from the network according to a predetermined probability, one can effectively curtail overfitting and enhance the robustness of the neural network.

### 5.2. Urban Road Lane Number Detection Experiment

The detection of the number of urban road lanes is essentially a multi-classification task, as it equates to the classification of urban road types. The experimental environment consists of a dual-channel Intel (R) Xeon (TM) X5650, 3 GHz, 40 GRAM, 64-bit Ubuntu 18.04, utilizing a Python programming language and TensorFlow framework. The study employs a previously constructed deep learning dataset, encompassing a total of 1210 samples. Among them, the number of samples in one-way one-lane is 270, the number of samples in one-way two-lane is 456, and the number of samples in one-way three-lane is 484. The samples are randomly divided into a training dataset and a test dataset according to a ratio of 10:1.

In this study, we employed the LeNet-5 model for experimental purposes. The initial learning rate of the model is 0.00125, the number of trainings is 3500, the batch size is 55, and the optimizer uses Adam. The convolutional neural network was trained on the input samples from the training set in batches. The weights and parameters of the convolutional neural network were derived based on the difference between the output results and the actual results. Subsequently, the parameters were updated according to the learning rate and the error back-propagation method to enhance the accuracy, ultimately yielding a stable LeNet-5 model. In order to provide an intuitive and clear visualization of the training process of the LeNet-5 model, we have depicted the corresponding accuracy curve and loss value curve in the Figures 11 and 12.

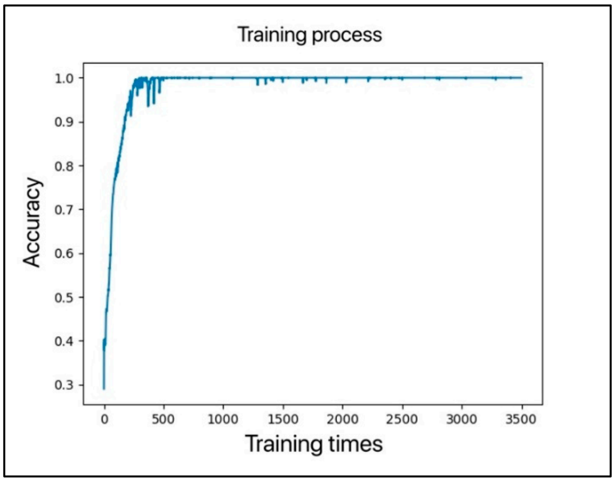


**Figure 11.** Accuracy curve of the LeNet-5 model training process.

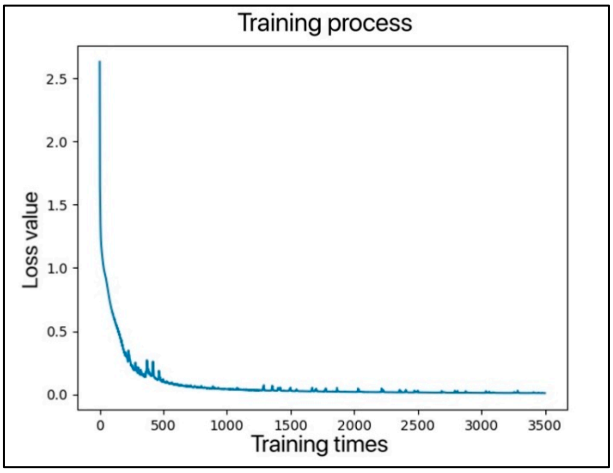


**Figure 12.** Loss value curve of the LeNet-5 model training process.

The loss value curve exhibits a tendency towards stability when the number of training iterations reaches 1500–2000. Upon surpassing 3000 training iterations, the loss value stabilizes at approximately 0.224. Upon completion of the LeNet-5 model training, the recognition accuracy reaches a level of 93.58%.

### *5.3. Comparative Analysis of Experimental Results*

The precision of the LeNet-5 model is subject to several factors, including the activation function, initial learning rate, batch size, and grid size. This study comparatively examines and analyzes the experimental outcomes derived from these aspects during the training dataset process.

(1) Activation function comparison analysis. This study employs sigmoid and ReLU activation functions for a comparative analysis. As illustrated in Table 2, the ReLU activation function demonstrates a more efficient training process, lower loss values, and superior accuracy when compared to the sigmoid activation function.

**Table 2.** Comparison of different activation functions.

| Activation Function | Training Time (s) | Loss Value | Accuracy |
|---|---|---|---|
| ReLU | 915 | 0.224 | 93.58% |
| Sigmoid | 969 | 0.598 | 84.74% |

(2) The choice of the initial learning rate is crucial, as it serves as a means to regulate the learning progression of the model. Learning rate is a hyperparameter in deep learning. As illustrated in Table 3, the training time for the LeNet-5 model increases as the learning rate decreases. Notably, when the learning rate is set at 0.00125, the model demonstrates the highest accuracy on the training dataset and the lowest loss value. Hence, the optimal learning rate for the LeNet-5 model is 0.00125.

**Table 3.** Comparison of different initial learning rates.

| Initial Learning Rate | Training Time (s) | Loss Value | Accuracy |
|---|---|---|---|
| 0.0003625 | 1157 | 0.326 | 90.37% |
| 0.0006250 | 998 | 0.613 | 85.5% |
| 0.0012500 | 915 | 0.224 | 93.58% |
| 0.0025000 | 691 | 0.38 | 89.28% |
| 0.0050000 | 689 | 0.79 | 79.65% |

(3) The selection of the batch size is crucial in the training input neural network, as it denotes the number of samples involved. As illustrated in Table 4, the loss value for the training dataset is minimized and the accuracy is maximized when the batch size is set at 55. Hence, the batch-size value for the LeNet-5 model has been prudently assigned as 55.

**Table 4.** Comparison of different batch size.

| Batch-Size | Training Time (s) | Loss Value | Accuracy |
|---|---|---|---|
| 35 | 1148 | 0.457 | 88.23% |
| 45 | 984 | 0.262 | 88.15% |
| 55 | 915 | 0.224 | 93.58% |
| 65 | 774 | 0.482 | 82.97% |
| 75 | 722 | 0.319 | 88.11% |

(4) The selection of the grid size is crucial, as it can significantly affect the spatial distribution characteristics of the FCD and, in turn, influence the classification accuracy. As illustrated in Table 5, the grid size of 32 × 32 yields the minimal loss value and the highest accuracy in the training dataset. Hence, the grid size is meticulously set to 32 × 32.

**Table 5.** Comparison of different grid sizes.

| Grid Size | Training Time (s) | Loss Value | Accuracy |
|---|---|---|---|
| 24 × 24 | 700 | 0.553 | 83.35% |
| 28 × 28 | 999 | 0.299 | 87.48% |
| 32 × 32 | 915 | 0.224 | 93.58% |
| 36 × 36 | 769 | 0.512 | 86.93% |
| 40 × 40 | 1154 | 0.403 | 86.49% |

Specifically, in the selection experiment of activation functions, initial learning rates, batch sizes, and grid sizes, the optimal candidate factors were initially obtained based on empirical knowledge. Three of these candidate factors were then fixed, with another parameter factor adjusted, and the training time, loss values, and accuracy indicators were recorded. Ultimately, following these experiments, it was verified that the optimal candidate factors indeed constituted the best factors.

Optimal parameter settings for training the LeNet-5 model have been identified. Upon completion of the LeNet-5 model training, a total of 110 test dataset samples were input into the trained network for classification. Of these, 102 samples were accurately classified, achieving a recognition accuracy of 92.7%. The model demonstrates comparable recognition accuracy in both the test and training datasets, indicating its effectiveness.

To assess the performance of the deep learning approach presented in this study, we conducted a quantitative comparison with several classification methods, including Kernel Density Estimation [7], Naïve Bayesian, Constraint Gaussian Mixture Model [11], Fuzzy Logic [13], Gradient Lifting Decision Tree [14], The Least Square Estimate to Constrain Gaussian Mixture Model [16], and the Weighted Constrained Gaussian Mixture Model and Hidden Markov Model [17]. The results of the lane number identification comparisons are presented in Table 6. Based on these findings, our method demonstrates the highest prediction accuracy when compared to other classification techniques.

**Table 6.** Lane number identification comparisons.

| Methods for Lane Number Identification | Accuracy |
|---|---|
| Kernel Density Estimation [7] | 74.2% |
| Naïve Bayesian [9] | 83.7% |
| Constraint Gaussian Mixture Model [11] | 85.2% |
| Fuzzy Logic [13] | 82.9% |
| Gradient Lifting Decision Tree [14] | 83.9% |
| The Least Squares Estimate to Constrain the Gaussian Mixture Model [16] | 83.3% |
| The Weighted Constrained Gaussian Mixture Model and Hidden Markov Model [17] | 78.6% |
| Deep Learning (Our method) | 92.7% |

## 6. Conclusions

In this study, the relevant technical methods for obtaining lane information based on FCD in recent years are analyzed and summarized, and the overall accuracy is not high. A method of deep learning is proposed to classify low-frequency FCD, and the method can detect the number of urban road lanes and improve the detection accuracy. First, the FCD is cleaned and classified. Then, the deep learning sample data is constructed through the processes of segmentation, rotation, and grid. Finally, the LeNet-5 model is used for training and prediction. The results indicate that our approach achieves a prediction accuracy of 92.7% for the number of lanes, significantly outperforming other methods.

The primary contribution of this method is the deep learning model introduced when low-frequency FCD is used to detect the number of urban lanes. With the help of the image sample idea, the point-like FCD is converted into a two-dimensional array of deep learning samples. The utilization of the deep learning model enhances the prediction accuracy for the number of lanes, marking this as a valuable exploration. High-precision map production companies can collaborate with floating car management departments or public security departments to produce or update high-definition maps using FCD.

The findings of this study are subject to two limitations. Firstly, extracting lane number information from low-frequency FCD necessitates a higher density of trajectory points, which in turn requires a longer collection time for FCD. This could potentially affect the timeliness of the results. Secondly, the LeNet-5 employed in this experiment is a conventional deep learning model. Future research could explore the use of more advanced deep learning architectures to achieve higher prediction accuracy.

**Author Contributions:** Conceptualization, Xiaolong Li; methodology, Yun Zhang and Xiaolong Li; formal analysis, Longgang Xiang; investigation, Yun Zhang and Tao Wu; writing—original draft preparation, Yun Zhang; writing—review and editing, Xiaolong Li, Longgang Xiang and Tao Wu; visualization, Yun Zhang. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work is supported by the National Natural Science Foundations of China (Grant No. 42261078), the Jiangxi Provincial Key R&D Program (Grant No. 20223BBE51030), and the Science and Technology Research Project of the Jiangxi Bureau of Geology (Grant No. 2022JXDZKJKY08).

**Data Availability Statement:** Due to privacy and other reasons, experimental data should not be made public.

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

## References

1. Bar Hillel, A.; Lerner, R.; Levi, D.; Raz, G. Recent progress in road and lane detection: A survey. *Mach. Vis. Appl.* **2014**, *25*, 727–745. [CrossRef]
2. Zhao, S.; Zhang, J.; Qu, R. An Improved Map Matching Algorithm for Floating Car. *Bull. Surv. Mapp.* **2018**, *1*, 97–102. [CrossRef]
3. Yang, W.; Ai, T. A Method for Road Network Updating Based on Vehicle Trajectory Big Data. *J. Comput. Res. Dev.* **2016**, *53*, 2681–2693.
4. Yang, X.; Tang, L.; Niu, L.; Zhang, X.; Li, Q. Generating lane-based intersection maps from crowdsourcing big trace data. *Transp. Res. Part C Emerg. Technol.* **2018**, *89*, 168–187. [CrossRef]
5. Zheng, L.; Li, B.; Yang, B.; Song, H.; Lu, Z. Lane-Level Road Network Generation Techniques for Lane-Level Maps of Autonomous Vehicles: A Survey. *Sustainability* **2019**, *11*, 4511. [CrossRef]
6. Li, X.; Zhang, Y. Summary of road information extraction methods. *Bull. Surv. Mapp.* **2020**, *6*, 22–27. [CrossRef]
7. Uduwaragoda, E.R.I.A.C.M.; 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. [CrossRef]
8. Chen, Y.; Krumm, J. Probabilistic Modeling of Traffic Lanes from GPS Traces. In Proceedings of the 18th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (ACM-GIS2010), San Jose, CA, USA, 3–5 November 2010; pp. 81–88. [CrossRef]
9. Tang, L.; Yang, X.; Kan, Z.; Li, Q. Lane-Level Road Information Mining from Vehicle GPS Trajectories Based on Naïve Bayesian Classification. *ISPRS Int. J. Geo-Inf.* **2015**, *4*, 2660–2680. [CrossRef]
10. Tang, L.; Yang, X.; Kan, Z.; Wang, X.; Li, Q.; Shaw, S.L. Traffic Line Numbers Detection Based on the Naïve Bayesian Classification. *China J. Highw. Transp.* **2016**, *29*, 116–123.
11. Tang, L.; Yang, X.; Dong, Z.; Li, Q. CLRIC: Collecting Lane-Based Road Information Via Crowdsourcing. *IEEE Trans. Intell. Transp. Syst.* **2016**, *17*, 2552–2562. [CrossRef]
12. Tang, L.; Yang, X.; Jin, C.; Liu, Z.; Li, Q. Traffic Lane Number Extraction Based on the Constrained Gaussian Mixture Model. *Geomat. Inf. Sci. Wuhan Univ.* **2017**, *42*, 341–347. [CrossRef]
13. Yang, X.; Tang, L.; Stewart, K.; Dong, Z.; Zhang, X.; Li, Q. Automatic change detection in lane-level road networks using GPS trajectories. *Int. J. Geogr. Inf. Sci.* **2018**, *32*, 601–621. [CrossRef]
14. Li, X.; Wu, Y.; Tan, Y.; Cheng, P.; Wu, J.; Wang, Y. Method Based on Floating Car Data and Gradient Boosted Decision Tree Classification for the Detection of Auxiliary Through Lanes at Intersections. *ISPRS Int. J. Geo-Inf.* **2018**, *7*, 317. [CrossRef]
15. Arman, M.A.; Tampere, C.M.J. Lane-level routable digital map reconstruction for motorway networks using low-precision GPS data. *Transp. Res. Part C Emerg. Technol.* **2021**, *129*, 103234. [CrossRef]
16. Shu, J.; Wang, S.; Jia, X.; Zhang, W.; Xie, R.; Huang, H. Efficient Lane-Level Map Building via Vehicle-Based Crowdsourcing. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 4049–4062. [CrossRef]
17. Fan, L.; Zhang, J.; Wan, C.; Fu, Z.; Shao, S. Lane-Level Road Map Construction considering Vehicle Lane-Changing Behavior. *J. Adv. Transp.* **2022**, *33*, 6040122. [CrossRef]

18. Zhou, J.; Guo, Y.; Bian, Y.; Huang, Y.; Li, B. Lane Information Extraction for High Definition Maps Using Crowdsourced Data. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 7780–7790. [CrossRef]
19. Biagioni, J.; Eriksson, J. Map inference in the face of noise and disparity. In Proceedings of the 20th International Conference on Advances in Geographic Information Systems, Redondo Beach, CA, USA, 6–9 November 2012; pp. 79–88. [CrossRef]
20. Kuntzsch, C.; Sester, M.; Brenner, C. Generative models for road network reconstruction. *Int. J. Geogr. Inf. Sci.* **2016**, *30*, 1012–1039. [CrossRef]
21. Chen, L. Research on Information Mining of Taxi GPS Data. Master's Thesis, Beijing Jiaotong University, Beijing, China, 2018.
22. Lv, Z. Research on GPS Data Preprocessing of Floating Car in Urban Traffic Guidance System. Master's Thesis, Lanzhou Jiaotong University, Lanzhou, China, 2016.
23. Ester, M.; Kröger, P.; Sander, J.; Xu, X. A density-based algorithm for discovering clusters in large spatial database. In Proceedings of the 2th International Conference on Knowledge Discovery and Data Mining, Portland, OR, USA, 2–4 August 1996; pp. 226–231.
24. Guo, M. Research about DBSCAN Next Clustering Based on Spark Platform. Master's Thesis, Beijing University of Technology, Beijing, China, 2018.
25. Qu, J.; Wang, Y.; Zhao, Q. Application of DBSCAN Clustering and Improved Bilateral Filtering Algorithm in Point Cloud Denoising. *Bull. Surv. Mapp.* **2019**, *11*, 89–92. [CrossRef]
26. Zhang, Y.; Li, X. Floating Car Data Preprocessing Based on DBSCAN Algorithm. *Jiangxi Sci.* **2020**, *38*, 293–297, 319.
27. Lecun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
28. Qin, Y. Research on Key Technologies of Traffic Sign Detection and Recognition. Master's Thesis, Changchun University of Science and Technology, Changchun, China, 2019.
29. Yu, D.; Wang, H.; Chen, P.; Wei, Z. Mixed Pooling for Convolutional Neural Networks. In Proceedings of the 9th International Conference of Rough Sets and Knowledge Technology, Shanghai, China, 24–26 October 2014. [CrossRef]
30. Hinton, G.E.; Srivastava, N.; Krizhevsky, A.; Sutskever, I. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv Prepr.* **2012**. [CrossRef]
31. Baldi, P.; Sadowski, P. The dropout learning algorithm. *Artif. Intell.* **2014**, *210*, 78–122. [CrossRef] [PubMed]

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.