Hindawi
Mobile Information Systems
Volume 2021, Article ID 5579451, 16 pages
https://doi.org/10.1155/2021/5579451

Research Article

The Traffic Flow Prediction Method Using the Incremental Learning-Based CNN-LTSM Model: The Solution of Mobile Application

Yanli Shao,[1] **Yiming Zhao**,[1] **Feng Yu**,[1] **Huawei Zhu**,[2] **and Jinglong Fang**[1]

[1]*Key Laboratory of Complex Systems Modeling and Simulation, School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018, China*
[2]*School of Information and Electrical Engineering, Zhejiang University City College, Hangzhou 310015, China*

Correspondence should be addressed to Jinglong Fang; fjl@hdu.edu.cn

Received 16 January 2021; Revised 28 March 2021; Accepted 24 May 2021; Published 1 June 2021

Academic Editor: Xiaoxian Yang

With the acceleration of urbanization and the increase in the number of motor vehicles, more and more social problems such as traffic congestion have emerged. Accordingly, efficient and accurate traffic flow prediction has become a research hot spot in the field of intelligent transportation. However, traditional machine learning algorithms cannot further optimize the model with the increase of the data scale, and the deep learning algorithms perform poorly in mobile application or real-time application; how to train and update deep learning models efficiently and accurately is still an urgent problem since they require huge computation resources and time costs. Therefore, an incremental learning-based CNN-LTSM model, IL-TFNet, is proposed for traffic flow prediction in this study. The lightweight convolution neural network-based model architecture is designed to process spatio-temporal and external environment features simultaneously to improve the prediction performance and prediction efficiency of the model. Especially, the K-means clustering algorithm is applied as an uncertainty feature to extract unknown traffic accident information. During the model training, instead of the traditional batch learning algorithm, the incremental learning algorithm is applied to reduce the cost of updating the model and satisfy the requirements of high real-time performance and low computational overhead in short-term traffic prediction. Furthermore, the idea of combining incremental learning with active learning is proposed to fine-tune the prediction model to improve prediction accuracy in special situations. Experiments have proved that compared with other traffic flow prediction models, the IL-TFNet model performs well in short-term traffic flow prediction.

1. Introduction

With the acceleration of urbanization, the number of motor vehicles has increased rapidly, which will cause many serious social problems, such as traffic congestion and environmental pollution, resulting in time-consuming travel, property losses, and even safety threats caused by traffic accidents [1]. Recently, the intelligent transportation system (ITS) based on cyber-physical systems (CPSs), as a key way to solve urban traffic problems, has received extensive attention and rapid development [2]. It is hoped that engineering systems and computing devices can reasonably and efficiently meet the requirements of system resource allocation and performance efficiency optimization. CPS is an intelligent technology that integrates computing, communication, and control (3C) technology with some excellent features, such as, real-time, security, reliability, and high performance. The fusion calculation and the prediction of traffic flow and vehicle speed received by the current road sensor can improve the urban traffic conditions to avoid traffic congestion and accidents as much as possible [3]. Traffic flow prediction is to predict the changes in traffic flow of a certain road in the next few minutes or hours. During the traffic peak period, if the traffic flow in a certain area can be accurately predicted in the future, it will help passengers adjust their travel routes to avoid traffic congestion.

Meanwhile, the transportation departments can also develop countermeasures in advance to divert traffic to ensure smooth traffic flow. Thus, in ITS, the use of existing traffic data to predict future short-term traffic flow accurately and in real-time is very important for urban traffic planning, management, and control. Short-term traffic prediction is of great significance to the construction of ITS and has become a current research hot spot.

Nowadays, there have been many studies on short-term traffic flow prediction. The transportation system is a complex nonlinear system with randomness and uncertainty that changes over time and space. With the explosive growth of traffic flow data, traditional statistical and machine learning algorithms remain insufficient to extract complex and deep features. Therefore, it is difficult to use large-scale data sets effectively to build accurate prediction models for scientific traffic guidance. Recently, deep learning technology has developed rapidly. Due to its powerful deep feature extraction and complex problem modeling capabilities, it has become more popular among researchers and has been successfully applied in image processing, natural language processing, computer vision, and so on. The application of traffic flow prediction is also increasingly widespread.

Although the performance of traffic flow prediction based on deep learning models is generally better, the training and updating of deep learning models require a lot of time costs and computational overhead, and it is often difficult to meet the real-time demand for short-term traffic flow prediction. Short-term traffic flow prediction is more practical and important to alleviate traffic congestion and realize intelligent traffic control and management. Thus, while ensuring high accuracy, the model update speed should be fast enough to meet the high real-time performance requirement of short-term traffic flow prediction. Meanwhile, the traffic flow prediction model may be deployed in any scenario, that is, in mobile applications, the computing power of mobile computing devices may not meet the computing resource requirements of deep learning models. Designing a lightweight deep learning model with low computational cost and high real-time performance is very important. Furthermore, other than temporal and spatial characteristics of traffic flow data, the uncertainty features caused by external environment have a greater impact on short-term traffic flow prediction. Therefore, how to introduce external environmental features to represent uncertainty to improve the prediction accuracy further without affecting the prediction efficiency remains a problem that needs to be studied urgently.

Based on the above analysis, an incremental learning-based CNN-LTSM model, IL-TFNet, is proposed for short-term traffic flow prediction in mobile applications. The contributions are summarized as follows: a lightweight model architecture is designed to process spatiotemporal and external environment features simultaneously to improve the model prediction performance wherein the traffic accident information that is difficult to obtain are derived from traffic flow data through K-means clustering algorithm. In particular, instead of the traditional batch learning algorithm, the incremental learning algorithm is applied for model training and update to reduce the cost of updating the model and improve prediction efficiency. This not only meets the real-time requirements but also enables the model suitable for mobile computing devices with limited computing resources. Moreover, the idea of combining incremental learning with active learning is proposed; some typical samples are used to fine-tune the model to ensure real-time performance and prediction accuracy in special scenarios.

The structure of this study is as follows. Section 2 describes the related works. Section 3 gives the motivation. Section 4 introduces the core method and its implementation details. Section 5 illustrates the related experiments and discussion. The last part gives the conclusions and future work.

2. Related Works

Short-term traffic flow prediction is of great significance for scientific traffic guidance and the construction of ITS. Currently, there are many related researches on traffic flow prediction, and various prediction models were also emerged, which were mainly divided into three categories: statistical, machine learning, and deep learning models.

The first is the statistical model, which is based on probability theory and mathematical statistics methods. Abdi et al. proposed an autoregressive integrated moving average (ARIMA) model in 1977 to predict short-term highway traffic conditions and achieved better prediction accuracy [4]. Williams and Hoel proposed a seasonal ARIMA (SARIMA) model, which designed univariate traffic data flow as a seasonal autoregressive integrated moving average process, which further improved the performance of ARIMA [5]. Jin et al. developed a prediction model combining Gaussian mixture model and Kalman filter to predict the highway traffic flow values [6]. Kong et al. used the vehicle speed parameters extracted from the GPS collection information to predict traffic conditions effectively [7].

As the amount of data increased significantly, statistical models showed problems such as poor fitting ability and low prediction accuracy. To improve the prediction performance further, the data-driven machine learning methods have gradually replaced the statistical models. Lv et al. introduced a nonparametric regression-based k-nearest neighbor (kNN) dynamic multi-interval traffic flow prediction model to capture the directionality of future states and minimize prediction errors [8]. With the widespread use of support vector machine (SVM), Gao et al. used kNN in combination with SVM for time-series problems [9]. Castro-Neto et al. proposed a supervised statistical learning technique based on online SVM for short-term highway traffic flow prediction [10]. Jiang and Adeli proposed a wavelet neural network model for simultaneous short-term and long-term traffic flow prediction [11]. Jeong et al. proposed a weighted SVM model for online learning for short-term traffic flow prediction with the consideration of the relative importance of time differences between traffic flow data [12]. However, only the temporal features are considered in the above studies, which reduced the prediction accuracy to a certain

extent. More studies show that traffic flow data in adjacent areas will also affect future traffic flow in the current area. Traffic flow prediction is not a purely time-series prediction problem, and the spatial feature of traffic flow data also affect the prediction performance, which needs to be considered to improve the prediction accuracy further. Vlahogianni et al. used neural network to predict short-term traffic flow and used genetic algorithm to optimize model structure [13]. Sun et al. used traffic flow data containing adjacent road information as training data and constructed a Bayesian network-based traffic flow prediction model to analyze current road trends [14]. Similarly, Lv et al. used the traffic flow data for a period as the stacked autoencoder model input to predict the traffic flow at the next moment [8]. Li et al. designed an imputation model to solve the problem of feature disappearance caused by original data corruption to deal with spatiotemporal features better [15]. Although most of the above methods improve the accuracy of traffic flow prediction, they are not well adapted to the large-scale data sample space due to the ability of complex feature extraction and model expression.

Recently, deep learning models are increasingly favored by the researchers owing to their powerful feature extraction capabilities and complex problem modeling capabilities in large-scale data sets. They have been successfully applied in various fields including computer vision, natural language processing, image processing, and real-time e-commerce [16], as well as the transportation field. Dunne and Ghosh proposed a neural network-based multivariate traffic condition prediction algorithm with adaptive learning strategy [17]. Zhang et al. designed an end-to-end residual convolutional neural network, ST-ResNet, based on the spatiotemporal features of traffic flow data to predict two-dimensional traffic flow data in each region of the city at a certain time [18]. Xue and Xue combined K-means clustering algorithm with the long short-term memory network (LSTM) model to predict the traffic flow [19]. Tian et al. completed the missing data of the traffic flow data set and then used LSTM model for training, which improved the prediction accuracy [20]. When considering only the spatiotemporal characteristics of traffic flow data cannot further improve the prediction performance, researchers began to consider introducing the uncertainty features for fusion prediction.

The uncertainty features caused by the external environment of traffic flow data affect the accuracy of traffic flow prediction, especially for short-term traffic flow prediction. In order to simultaneously consider multi-seasonality, nonstationarity, temporal and spatial correlations, and dynamics between traffic variables to improve prediction accuracy, Ma et al. proposed a time-space threshold vector error correction (TS-TVEC) model for short-term traffic flow prediction [21]. Polson and Sokolov illustrated the impact of important event in urban traffic by analyzing the change of traffic flow during the Chicago Bears game and extreme snowstorms. The experimental results show that the deep learning model achieves good prediction performance when the above events occur [22]. Based on historical traffic flow data, Zhang et al. used the weather conditions and holidays as the input of external environment feature and combined the two to make prediction to reduce the impact of uncertainty [18]. Chen et al. constructed a fuzzy deep-learning convolutional neural network (FDCN) model to alleviate the accuracy impact caused by the uncertainty while also increasing the complexity and time cost of the model. An et al. used a fuzzy approach to obtain traffic accident information and combined with CNN to achieve the accurate prediction of traffic flow [23]. However, the expert knowledge base for traffic accident information may not be completely accurate. How to extract the traffic accident feature accurately from traffic flow data remains to be studied. Yu et al. proposed an incremental learning-based CNN model, which introduced partial uncertainty features into the model to improve the prediction accuracy to some extent [24].

The above methods consider the impact of uncertainty by adding model branches or uncertainty processing modules, but the consideration of temporal and uncertainty features remain not comprehensive, and the prediction accuracy still needs to be improved. Moreover, although the prediction accuracy is improved to a certain extent, it also increases the model complexity and training time and cost and affects the prediction efficiency. In particular, a deep learning model based on large-scale training data set itself requires a lot of computing resources and training costs, and the processing of uncertainty further magnifies this defect. Therefore, without increasing training costs, how to introduce uncertainty features to improve prediction accuracy remains a difficult problem, especially for large-scale traffic flow data; how to reduce the cost of model training and update while improving the prediction accuracy and efficiency remains worthy of in-depth study.

3. Motivation

Traffic flow prediction is a key element in the construction of urban ITS. It is not an easy task to predict traffic flow data accurately and quickly. As described above, the scale of traffic flow is large, and the characteristics are complex, and it is greatly affected by temporal and spatial characteristics and the external environment. A large number of studies have used the spatiotemporal features for traffic flow prediction. However, if only temporal dependence and spatial dependence are considered, the prediction accuracy may be low, and the deviation may be large in some special circumstances since the accurate prediction of traffic flow is affected by many factors, including internal and external factors. The internal factors refer to the characteristics of the road, the operation of the vehicle, or the like, which are difficult to predict accurately. The external factors refer to changes in weather, major gatherings, holidays, traffic accidents, or the like, which have a certain pattern and can be predicted to a certain extent. The uncertainty caused by various external environmental factors has a greater impact on the traffic flow prediction, and the irregular fluctuations of traffic flow will affect the accuracy of the prediction. It is necessary to introduce the external environmental factors as uncertainty features during model training to improve prediction accuracy further.

The traffic flow changes in real time, especially on busy roads in big cities. Only when the traffic flow prediction result is accurate and real-time, can the intelligent dispatching and grooming of urban traffic be effectively carried out. Traditional model training uses batch learning method, that is, when a new data set is generated, it is added to the original data set to form a new training data set to train the model from scratch. This leads to massive amount of data for model training every time, and the model training is expensive. When the data set reaches a certain size, the model update speed may not keep up with the real-time data generation speed. Currently, the historical traffic flow database is very large, and its scale has increased sharply; the deep learning model with good prediction performance has high requirements on hardware facilities, and the model training and update time is relatively long. Especially in some special cases, that is, mobile application or real-time application, the model update time is too long, and the prediction accuracy is low; it is difficult to achieve the efficient and accurate prediction of short-term traffic flow. Especially, the deep learning models need to perform higher prediction efficiency and accuracy under limited computing resources in some mobile applications. Hence, it is necessary to improve the online training and update and prediction speed of the model to ensure its prediction stability and real-time performance while enabling the model to be effectively deployed on mobile computing devices.

Therefore, how to reduce the training and update cost of deep learning models to meet real-time prediction and limited computing resource requirements and use uncertainty features effectively to improve prediction accuracy further without increasing the complexity of the model remain a challenge.

4. IL-TFNet Model

4.1. Problem Formulation. Traffic flow prediction is based on the historical traffic flow data of a certain area and other related factors to predict the future traffic conditions of the area quickly and accurately. The road traffic system is a complex nonlinear system with human participation. The traffic flow data changes with time and space and has strong randomness and high uncertainty, which greatly affects the prediction accuracy of traffic flow. As mentioned earlier, time-series traffic flow prediction models, even deep learning models such as LTSM, cannot effectively process the spatial characteristics of traffic flow data, which reduces the prediction accuracy to some extent. The spatial information should be retained to fit the spatial dependence better in traffic flow data. Therefore, the incremental learning-based CNN-LTSM model, IL-TFNet, is proposed to capture the temporal and spatial characteristics of traffic flow data better without affecting the prediction efficiency. Meanwhile, the external environment factors, that is, weather, holidays, and traffic accident information, are also considered as uncertainty features to improve the prediction accuracy further. Here, the uncertainty features and 2D traffic flow data are stacked into three-dimensional (3D) model input to fit both the spatiotemporal and uncertainty of traffic flow data simultaneously.

Generally, the city is very large with many streets, and the computational cost and time required to predict the traffic flow data of each street are very large. In addition, the deployment of road-monitoring equipment on the road is often unbalanced. There may be multiple types of monitoring on the main road, but there may not be little or no monitoring on the side road. However, from a regional perspective, the number of monitoring equipment in each region is relatively balanced, so it is reasonable to divide the city into multiple regions to predict traffic flow data. Currently, most of the widely used data sets divide the traffic flow (32×32) data sets into regional data. The data set used in this study also provides regional traffic flow information with a size of 32×32. Therefore, the urban area is divided into $W \times H$ (32×32) two-dimensional (2D) grids here. The whole traffic flow data is defined as a tensor form $X \in R^{L \times C \times W \times H}$, where L is time; C is the flow type (input flow or output flow); and W and H are the height and width. $X_T \in R^{C \times W \times H}$ refers to the traffic flow data at time T, and $Y_T \in R^{W \times H}$, $Z_T \in R^{W \times H}$, and $A_T \in R^{W \times H}$ represents the weather, holidays, and traffic accident information, respectively.

Based on the above definition, the traffic flow prediction problem is formalized as: traffic flow prediction is to predict the traffic flow data X_n based on the historical traffic flow data X_t and uncertain external environmental features: $\{X_t, Y_n, Z_n, A_n | t = 1, 2, 3, \ldots, n-1\}$. Note that the predicted traffic flow value at time n is based on historical traffic flow data at time step $1 \sim n-1$ and the uncertainty features including weather conditions, holidays, and traffic accident information at time n. The reason for only considering the uncertainty of time step n but not multiple time steps is that unlike historical traffic flow information, only the uncertainty information in a short time close to time point n will obviously affect the traffic flow.

4.2. Method Overview. To address above issues, an incremental learning-based CNN-LTSM model, IL-TFNet, is proposed for short-term traffic flow prediction in this study. The overview of the method is shown in Figure 1, which is primarily divided into three steps.

Step 1. Data preparation: This step is primarily to prepare for the model input, especially to extract uncertainty features, such as weather, holidays, and traffic accident information. The 2D traffic flow data and the uncertainty features are stacked into 3D data as the model input to achieve more accurate prediction. Here, K-means clustering algorithm is used to derive the implicit traffic accident information since it is difficult to obtain directly from historical traffic flow data.

Step 2. IL-TFNet model training: Here, the incremental learning rather than batch learning is used for IL-TFNet model training to improve training efficiency [25]. The historical training data set is divided into multiple subtraining data sets and one testing data set.

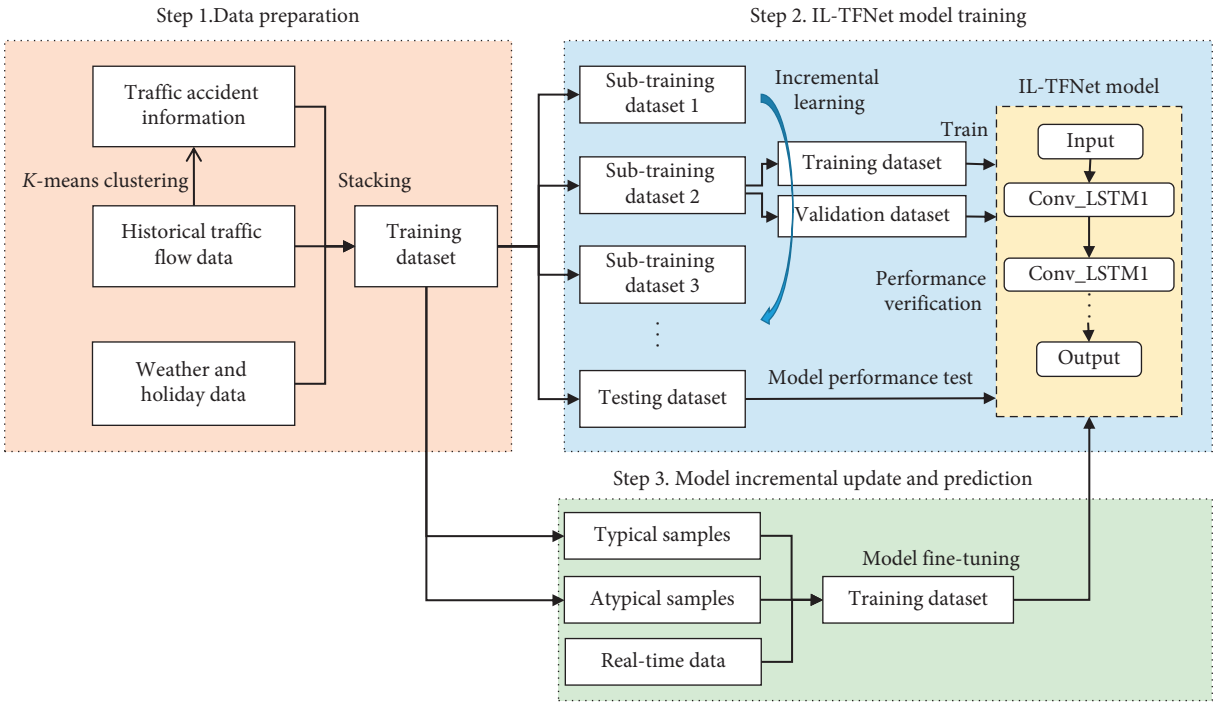

Figure 1: Method overview.

The model is iteratively trained through multiple subtraining data sets. The validation data set extracted from subtraining data set is used to measure model performance when a certain number of training iterations reaches. In this way, the original prediction model can be iteratively optimized based on the newly added real-time data set, thereby ensuring the real-time and accuracy of the prediction model.

Step 3. Model incremental update and prediction: When new data arrives, incremental learning method are still used for model updates to improve update efficiency and prediction accuracy. In special traffic situations, such as foggy days or unexpected traffic accidents, the prediction accuracy may be significantly reduced due to the differences in different sample spaces. Therefore, based on incremental learning, active learning is introduced to obtain typical samples in special situations, so that a small amount of typical data sets can be used to fine-tune the prediction model, thereby improving its prediction accuracy while ensuring real-time performance.

4.3. Data Preparation

4.3.1. Model Input Preparation. As mentioned before, the traffic flow data set used in this study divides the urban area into 32×32. The input of the model consists of historical traffic flow data set of the latest n consecutive time periods and external environmental characteristics, such as the weather conditions, holidays, and traffic accident information. These factors are deliberately introduced as the uncertainty features to improve the prediction accuracy in some special cases. Here, the weather, holidays, and traffic accident information are also organized into the size of 32×32 (2D matrix) to ensure data consistency and availability. Each value in the weather matrix refers to the severity level of the impact of weather conditions on traffic. The level value ranges from 0 to 16, wherein 0 represents sunny and the 4 largest values (13~16) represent heavy snow, heavy fog, sandstorms, and amnesty. Similarly, if the current prediction time belong to the holiday, the holiday matrix value is all 1s; otherwise all 0s. Weather and holiday information is easy to get, but traffic accident information is difficult to obtain and often incomplete. Thus, traffic accident information extraction is detailed in the next section.

Different from using additional model branches to process external environment features [18], these features and traffic flow data are stacked together to form the model input $(32 \times 32 \times (2n + 3))$. Especially, this scheme can deeply integrate the traffic flow information with the external environment information, rather than the simple linear addition between branches, which can better deal with the impact of uncertainty and further improve the prediction accuracy without increasing the model complexity and training cost.

4.3.2. K-Means Clustering-Based Traffic Accident Feature Extraction. Generally, the external environment features can be divided into local and global influence features according to their influence range. Here, the global influence feature refers to weather and holiday information, which will affect the traffic flow of the entire city or most of the urban area. The local influence feature mainly refers to traffic accident information, which is difficult to obtain directly. In

the work of An et al. [23], fuzzy inference is used to estimate traffic accident information from historical data. However, the accuracy of traffic accident prediction largely depends on the fuzzy inference rule base, lacking sufficient theoretical basis. Therefore, the K-means clustering algorithm is proposed to extract features related to traffic accident from traffic flow data and then discover the intrinsic relationship between the samples and different clusters to obtain the traffic accident feature.

Traffic flow data can be roughly divided into accident-free and accident data sets, and the characteristics of the two are obviously different. The traffic flow of the former changes smoothly, while the latter may change suddenly. According to the severity of traffic accidents, the occurrence of traffic accidents will have different impacts on the current or neighboring areas. In fact, according to traffic flow fluctuation caused by traffic accidents, there are obvious differences in the characteristics of each cluster, and the clustering situation is basically clear and limited. Therefore, the number of clusters can be set in advance, and the K-means algorithm is a good choice. As the K-means algorithm is an unsupervised clustering algorithm commonly used in the industry, it is relatively simple and efficient to implement, and the clustering effect is usually good. Meanwhile, experiments show that this method performs well in current scenario, and the clustering time is relatively less in line with the real-time requirements of traffic flow prediction task. After clustering, one cluster represents that traffic flow tends to be normal when there is no traffic accident, and the other cluster represents the traffic flow trend under different levels of traffic accidents, which can be further subdivided according to the sample characteristics. The sample category information is introduced into the model as traffic accident feature. Note that the features obtained are not only traffic accidents but also general concept, including a series of events that may change the traffic flow, such as parades, concerts, and so on, which are uniformly referred to as traffic accident features.

Since traffic accidents are characterized by abnormal fluctuations in the traffic flow data, instead of the original data, the relative flow information is used as the feature for clustering to obtain traffic accident information indirectly. Here, traffic accident information is obtained by extracting two features including relative traffic flow fluctuation (RTFF) and relative flow change rate (RFCR).

RTFF: Each traffic flow data consists of inflow and outflow data. The relative flow F_t is defined as the difference between inflow and outflow over a certain period of time in a region. Accordingly, RTFF, represented by W_t, is defined as the difference between the relative flow rate $F_t^{(i,j)}$ at a specific time point in a specific area and the corresponding average value $\widetilde{F}_{\tilde{t}}^{(i,j)}$.

$$F_t^{(i,j)} = x_{t-in}^{(i,j)} - x_{t-out}^{(i,j)}, \tag{1}$$

$$W_t^{(i,j)} = F_t^{(i,j)} - \widetilde{F}_{\tilde{t}}^{(i,j)}, \tag{2}$$

where x_t is the 3D traffic flow data $(32 \times 32 \times 2)$ at each moment t.

RFCR: The rate of change in traffic flow is related to the previous moment. RFCR, expressed in R_t, is the ratio of previous relative traffic flow data to current relative traffic flow data.

$$R_t^{(i,j)} = \frac{F_{t-1}^{(i,j)}}{F_t^{(i,j)}}. \tag{3}$$

In the clustering task, in addition to selecting an appropriate clustering algorithm, the appropriate distance measurement method is also important. The commonly used distance measurement methods include Euclidean distance, Manhattan distance, and Cosine distance. Since the distribution of samples with a higher probability of traffic accidents is obviously different from that of ordinary samples, the Euclidean distance is more sensitive to this, so the Euclidean distance is used to measure the differences between traffic flow samples.

The specific process of K-means clustering-based traffic accident feature extraction is as follows: different from the original data samples used in traffic flow prediction, the cluster samples are represented by two features: RTFF and RFCR. The traffic flow data $(32 \times 32 \times 2)$ is divided into 1,024 subsamples firstly. Next, based on formula [2,3], $W_t^{(i,j)}$ and $R_t^{(i,j)}$ of each original data sample in the region (i and j) at the moment t are calculated. Then, K-means clustering algorithm is applied to the subsamples to divide them into k clusters based on the similarity between samples. The similarity measurement used here is Euclidean distance; the formula is as follows:

$$\operatorname{dist}(X, Y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}, \tag{4}$$

where X and Y are two different samples, and x_i and y_i are the values of the i-th feature of samples X and Y, respectively.

After initializing the centroids of each cluster, iterative optimization is performed until the convergence condition is reached. Both the k value setting and the initial cluster center positions will greatly affect the clustering results, so multiple attempts are required to obtain appropriate parameter values. After the clustering is completed, the cluster category label is added as the traffic accident feature to the input of the traffic flow prediction model. The data format remains $32 \times 32 \times 2$, but the third dimension is changed from inflow and outflow to W (RTFF) and R (RFCR). In this way, the uncertainty contained in the traffic accident feature is modeled to improve the prediction accuracy further.

4.4. IL-TFNet Model Training

4.4.1. IL-TFNet Model Architecture. Generally, the models based on traditional statistics or machine learning are difficult to learn the temporal and spatial characteristics of traffic flow data effectively. This study proposes an incremental learning-based CNN-LTSM model, IL-TFNet, to achieve accurate short-term traffic flow prediction. The structure of the model is shown in Figure 2.

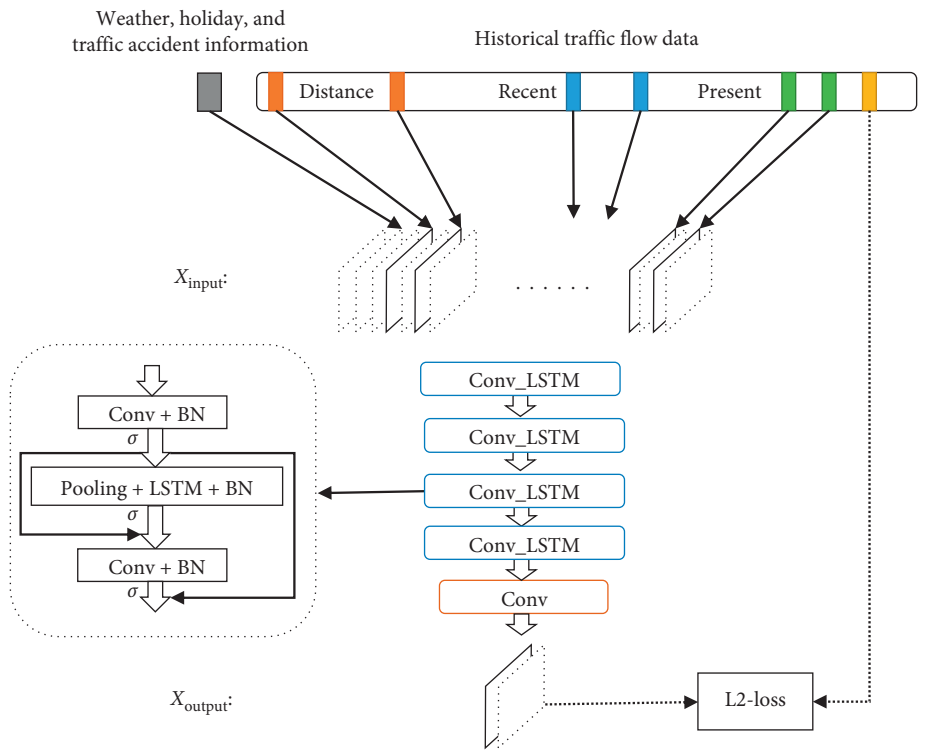

Figure 2: IL-TFNet model architecture.

The model architecture design motivation is as follows: on the one hand, CNN has been proven to have good spatial feature recognition capabilities [26–28]. In addition, the traffic flow data is very similar to the visual image data. Take single time step traffic flow data as an example; its data structure is a two-dimensional matrix, and each element represents the traffic flow data in the corresponding area. The entire two-dimensional matrix contains a large amount of spatial feature information, and CNN is very good at processing data with such structure. Furthermore, the traffic-monitoring system is capturing a large amount of traffic flow data at all times, and the scale of data is increasing rapidly. When faced with a large amount of data, machine learning method exposes the problem that the model cannot be further optimized as the data size increases. CNN has a deeper network structure and performs well on large-scale data, thereby reducing model calculation and optimization costs. On the other hand, in addition to spatial features, traffic flow data also contains temporal features. Although CNN has excellent ability to process spatial information, its ability to extract temporal information is poor, while LSTM has been proven to have relatively good temporal feature extraction capabilities in other fields. In particular, the time characteristics of traffic flow data include long-term dependence and short-term dependence. RNN cannot handle long-term dependence characteristics, so LSTM is more suitable for traffic flow prediction.

Historical traffic flow data and external environment features are jointly organized as 3D inputs and processed using the same Conv_LTSM unit instead of being processed in separate model branches in most other research works. In this way, by stacking historical traffic flow data and external environment features at different time steps (forming 3D data from 2D spatial data), each Conv_LTSM unit can effectively extract the temporal and spatial features of historical traffic flow data and the external environment features simultaneously. This makes up for the defects of previous work [18], which considers only the particularity between different features while ignoring the correlation between features. Especially, simple linear addition cannot make the model to consider fully the effects of different external environmental features on traffic flow. The introduction of uncertainty can further improve the model generalization ability, thereby improving the prediction performance. Meanwhile, the increase in training cost and model complexity can be neglected, since the convolutional layer parameter scale of CNN does not increase with increasing input size due to the parameter sharing strategy. Therefore, the proposed IL-TFNet model can effectively fit the time dependence, spatial dependence, and uncertainty of the traffic flow data to get better prediction performance without increasing the model complexity and prediction cost and is suitable for lightweight devices.

The depth of the deep learning model network structure determines the performance of the model [29], but a deeper neural network will make the model prone to problems such gradient disappearance during the training process. Thus, the proposed model uses the deep residual network structure to combine CNN and LTSM [30]. Moreover, the Conv_LTSM unit formula is as follows:

$$
\begin{aligned}
X_s^{(k+1)} &= f\left(\mathrm{Conv}\left(X^{(k)}\right)\right),\\
X_L^{(k+1)} &= f\left(\mathrm{Pool}\left(X_s^{(k+1)}\right)\right),\\
X^{(k+1)} &= f\left(\mathrm{Conv}\left(X_L^{(k+1)} \cdot X_s^{(k+1)}\right)\right) + X_s^{(k+1)},
\end{aligned} \tag{5}
$$

where $X^{(k)}$ and $X^{(k+1)}$, respectively, represent the input and output data of the k-th layer Conv_LTSM unit. $X_s^{(k+1)}$ represents the spatiotemporal features obtained through the convolutional layer, and $X_L^{(k+1)}$ represents the temporal features extracted by the LSTM layer. Conv and Pool represent convolution operation and pooling operation, respectively, and the function f represents batch normalization operation and activation operation.

As shown in the structure described by the dashed box on the left of Figure 2, the input traffic flow data first passes through the convolutional layer to extract preliminary spatiotemporal features. The convolution kernel of size 3×3 is used to extract the spatial features of the local area. If a larger convolution kernel is used, although longer-distance spatial dependence can be extracted, it may also contain more invalid feature information, which may lead to larger deviations in the prediction results. Therefore, this structure is a more suitable choice, which allows the model to maintain a higher prediction accuracy without reducing the representation ability. A batch normalization operation is used between the convolution operation and the activation operation to alleviate the problems of vanishing gradient or overfitting that may occur in the model. In order to make up for the weak temporal feature extraction ability of CNN, the LTSM is added between the two convolutional layers. Note that LSTM can only process one-dimensional data sequences; a pooling layer needs to be used to reduce the spatial dimension (i.e., three-dimensional data is reduced to one-dimensional data) before extracting temporal features. Then, the sigmoid function is used to convert the time feature into a normal distribution in the range of [−1, 1], which is fused with the features initially extracted by the first convolutional layer, thereby enhancing the time feature information. Finally, a convolutional layer is used to obtain complete spatiotemporal features. As shown on the right of Figure 2, there are four Conv_LSTM units in the entire model, and each unit compresses the time dimension of the data to ensure that a single time step prediction data can be obtained at the end. In particular, the method of gradually compressing the time dimension makes the time information more visible to the subsequent modules and retains the time characteristics as much as possible.

4.4.2. Model Incremental Training. The deep learning model with multiple network layers cannot be applied directly to the short-term traffic flow prediction task, since it requires high computing resources and long training time due to its complex network structure and massive training data scale. Especially, if the new and historical data sets are directly fused as a new data set to train the model, the training and updating cost is too expensive to meet the short-term traffic prediction requirements. Therefore, incremental learning strategy is used in the proposed IL-TFNet model to perform the model training and update to ensure the real-time prediction performance. As shown in Figure 1, historical traffic flow data set is divided into multiple subtraining data sets and one testing data set. The iterative training is conducted on the subtraining data set to improve the model prediction performance gradually. In each round of training, the subtraining data set is further divided into training and validation data sets. The training data set is used for training, and the validation data set is used to verify current model performance. When the model performance of multiple iterations on the current subtraining data set is not improved continuously, the model is transferred to the next subtraining data set for training.

During model training, Adam optimization algorithm is applied to optimize the model parameters iteratively and dynamically until convergence [24]. The reason is that Adam can dynamically adjust the learning rate in the iterative process, speed up the model convergence process, and at the same time slow down the performance degradation [31]. It is possible to achieve better prediction performance if the model parameters are adjusted based only on the newly arrived data.

Suppose that X is the model input; Y and $f(X)$ represent the observed value and the prediction value, respectively; the squared loss is used as the loss function in the IL-TFNet model; $L2$ regularization represented by α (0.005) is specifically used to prevent the model from getting into a serious overfitting state; and $\Omega(\theta)$ represents all the parameters in the model. The loss function is defined as follows:

$$
L(Y, f(X)) = \sum_{i=1}^{n} (Y, f(X))^2 + \alpha\Omega(\theta). \tag{6}
$$

Based on the above analysis, the specific model training algorithm is shown in Algorithm 1. The model parameter initialization operation is completed in the initialization phase, where step (2) and step (3) are the parameter initialization settings of Adam optimization algorithm, wherein t is used to control the number of iterations, and learning rate (ε) is a hyperparameter that controls the speed of algorithm optimization. ρ_1 and ρ_2 are the exponential attenuation rate of the first and second moment estimations in the Adam optimization algorithm, which are used to calculate first-order momentum (s) and second-order momentum (r). The bias correction of s and r are used to update model parameters, and δ is used as the deviation value to ensure the stability of parameter updating and prevent drastic changes of parameters. In the incremental training phase, the IL-IFNet model uses the Adam optimization algorithm to optimize the parameters iteratively until convergence in step [4–8]. Incremental iterative training is performed on the subtraining data set to improve the prediction performance of the model gradually. Finally, the model parameters θ after training are output in step [10].

Input: $D = \{(x_t, y_t)\}_{t=1}^{N_0}$, N_0 represents the number of samples, $x_t = (X_t, E_t | X_t = \{X_i | X_i \in R^{C\times W\times H}\}_{i=t-n}^{t-1}, E_t \in R^{N\times W\times H})$, $y_t = X_t$.
Output: IL-TFNet model with certain predictive performance after training.
Phase 1: Initialization phase:
(1) Training and testing data set initialization: Divide traffic flow data set D into multiple subtraining data sets (D_1, D_2, ..., D_{n-1}) and one testing data set (Dn).
(2) Initialization iterations $t = 0$, Learning rate $\varepsilon = 0.001$, Attenuation rate $\rho_1 = 0.9$, $\rho_2 = 0.999$.
(3) Initialize first-order momentum and second-order momentum $s = 0$, $r = 0$ and the parameter to ensure numerical stability $\delta = 10^{-8}$.
(4) Initialize the CNN parameters of IL-TFNet model: $\theta(w, \beta)$, $w \sim G[0, \sqrt{2/n}]$, $\beta = 0$.
Phase 2: Incremental training phase:
(1) **while** current subtraining data set is not empty **do**
(2) **while** the termination condition is not reached (the model prediction performance is in a bottleneck or reaches a certain iteration round) **do**
(3) Sample the subtraining data set to obtain current batch training data set
(4) Calculate gradient based on formula (7): $g \leftarrow (1/m)\nabla_\theta \sum_i L(f(x^{(i)}; \theta), y^{(i)})$
(5) Update iterations $t \leftarrow t + 1$
(6) Update first-order momentum and second-order momentum: $s \leftarrow \rho_1 s + (1 - \rho_1)g$, $r \leftarrow \rho_2 r + (1 - \rho_2) g \otimes g$
(7) Calculate bias correction $\hat{s} \leftarrow s/1 - \rho_1^t$, $\hat{r} \leftarrow r/1 - \rho_2^t$
(8) Update model parameters $\theta \leftarrow \theta - \varepsilon(\hat{s}/\sqrt{\hat{r}} + \delta)$
end while
(9) Use the next subtraining data set to train the model in turn
end while
(10) **return** θ (Model parameters after training)

Algorithm 1: IL-TFNet model training algorithm.

4.5. Model Incremental Update and Prediction. Efficient and accurate short-term traffic flow prediction should consider historical traffic flow data, newly arrived data, and the related external features simultaneously to improve the prediction performance while meeting real-time requirement. However, the model update cost is expensive if the newly data is added to the historical data set to train the model directly. Especially, when the data set reaches a certain size, the model update speed may be low and makes it difficult to meet the requirement of real-time prediction. The online training, update, and prediction efficiency need to be improved to ensure the prediction stability and real-time performance of the prediction model. Thus, the newly generated data is regarded as a new subtraining data set, and the prediction model is updated through the above incremental learning method as well. Compared with directly fusing with historical data sets, only using a small-scale new data set to update the model will reduce model update time and improve modeling efficiency, thereby realizing real-time traffic flow prediction. Figure 3(a) shows the model update in normal case, the real-time traffic flow data is directly used as a subtraining data set to incrementally update the prediction model.

Note that the traffic flow data is quite different from the usual ones in some special cases, such as, bad weather, holidays, large gatherings, or traffic accidents. If the conventional traffic flow prediction model is still used directly for prediction, the prediction value obtained is greatly deviated from the observed value. However, the real-time and accurate prediction of traffic flow under special traffic conditions is of great significance to traffic control and dispatch. When a special traffic situation occurs, traffic flow data will be generated in real time, but the amount of data is relatively small. If the prediction model is incrementally updated based on only a small number of real-time data samples, it will cause serious overfitting and decrease prediction accuracy. Therefore, a method combining incremental learning and active learning is proposed, as shown in Figure 3(b), which combines typical samples, atypical samples, and real-time traffic flow data to fine-tune the prediction model, thereby improving the prediction accuracy in special cases without affecting the prediction efficiency.

Generally, the proportion of typical samples (the sample data occurs in special case) accounts for a small proportion of the entire data set in general, resulting in relatively low prediction accuracy in special cases. Considering that the contributions of different data samples to model training is different, if more of the most valuable sample data, such as typical samples and real-time data samples, are selected as the training data set for model update, the data set size can be reduced to decrease the model update time while meeting real-time prediction requirements. Extracting the most useful samples from large data set is a desirable way to reduce the training cost without reducing its performance. Thus, the proportion of typical sample data needs to be increased to improve the model generalization and applicability in this case. Based on the above analysis, as shown in Figure 3(b), the model incremental update process in special case is as follows: Firstly, the most useful unlabeled samples are queried through the active learning method, and then the experts mark them to obtain labeled samples as typical samples. The remaining data samples are called atypical samples. Then, the typical samples, some atypical samples obtained through random sampling, and real-time data are combined into the training data set to update the model

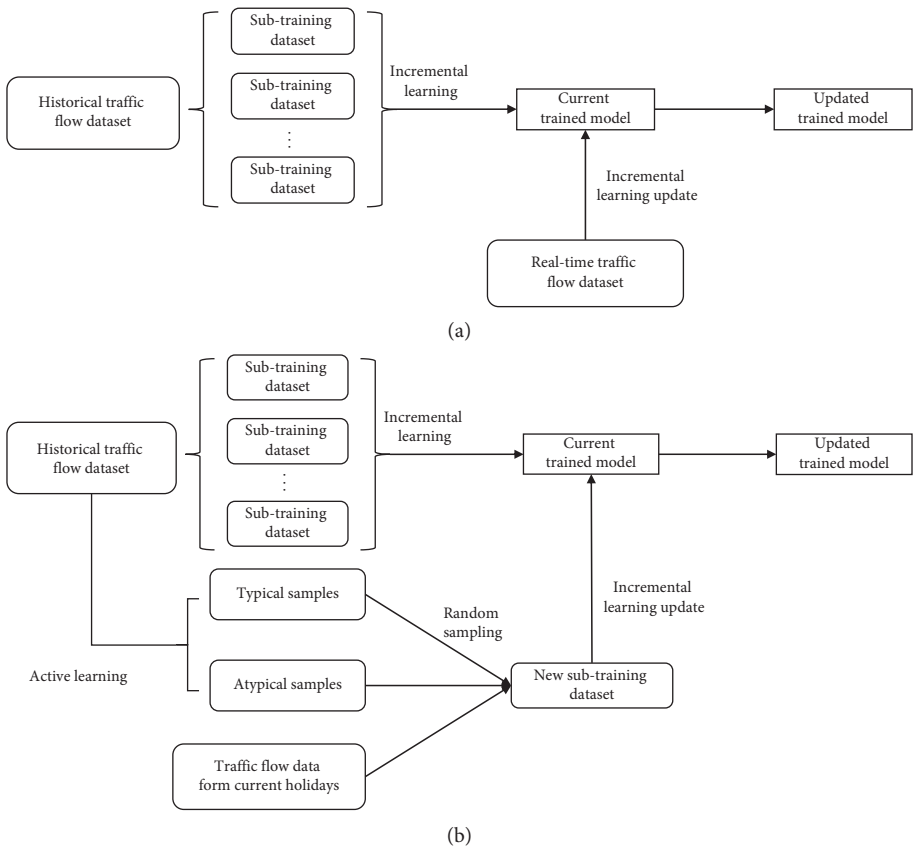

Figure 3: Model incremental update process in: (a) normal case and (b) special case.

through the incremental learning method. Now the typical samples, the newly generated traffic flow data, and a small number of atypical samples are merged to form a new subtraining data set for incremental update. In this way, mixing multiple types of data samples can consider the particularity and diversity of the data at the same time, thereby avoiding overfitting problems and improving the prediction accuracy in special cases.

Although the model adjustments will lead to longer prediction time, the combination of active learning and incremental learning will reduce the data set scale and reduce the model update time and will not affect the real-time performance of short-term traffic flow prediction. This is actually a method of converting the traffic flow prediction model that is universally applicable in all cases into a model for special cases while greatly improving the prediction accuracy in special cases.

5. Experiments and Discussion

5.1. Experimental Settings and the Related Definition

5.1.1. Experimental Settings. The experimental environment is shown in Table 1. The proposed model is verified in Python 3.6 and TensorFlow library.

The experimental data set is the same as the previous work [24], which is also the Beijing taxicab GPS trajectory data. The data set collects traffic flow data information, including input flow, output flow, external environmental features, and so on, of 32×32 grids in Beijing city every half an hour and obtains a total of 48 samples a day. Here, the data samples of the last 4 weeks are used as the test data sets, and the remaining are used as training data set. In the training data set, 10% of the samples are selected as validation data set for performance verification of each round of model training.

In this section, four groups of experiments are conducted to prove the feasibility and effectiveness of the proposed model. Since the overprediction performance analysis experiment includes a comparative experiment of adding traffic accident features, the first experiment is the performance analysis of different clustering algorithms, and the next is the overall prediction performance analysis. The third is incremental learning performance analysis, which is done by comparing the prediction accuracy and time cost of batch training and incremental training. Real-time prediction performance analysis in special cases is in the last group of experiment.

5.1.2. Performance Evaluation Indicators Definition. Three evaluation indicators, which are all commonly used in various research fields [32], are used for performance evaluation. Suppose that m is the total number of test samples, and y_i and $\hat{y}_i$ are the observed value and the predictive value of the i-th test sample, respectively. The evaluation indicators are defined as follows:

Table 1: Implementation environments.

Resource	Specification
Operating system	Window 10 bit professional version
CPU	Inter(R) Core i7-7700 3.60 GHz
RAM	16.00 GB
GPU	NVIDIA 1060 6G
Programming tools	Python 3.6 and TensorFlow 1.7.0
Programming language	Python

(1) Mean absolute error (MRE): The average of relative errors between the observed and predictive values:

$$\mathrm{MRE} = \frac{1}{m}\sum_{i=1}^{m}\frac{\left|y_i - \widehat{y}_i\right|}{y_i}. \tag{7}$$

(2) Mean relative error (MAE): The average of absolute errors between observed and predictive values:

$$\mathrm{MAE} = \frac{1}{m}\sum_{i=1}^{m}\left|y_i - \widehat{y}_i\right|. \tag{8}$$

(3) Root-mean-square error (RMSE): The arithmetic square root of mean square error:

$$\mathrm{RMSE} = \left[\frac{1}{m}\sum_{i=1}^{m}\left(y_i - \widehat{y}_i\right)^2\right]^{1/2}. \tag{9}$$

In addition, the clustering effect evaluation indicators are also defined for subsequent clustering algorithm evaluation:

(1) Silhouette coefficient (SC): The performance evaluation index evaluates the clustering effect by calculating the intra- and intercluster dissimilarity of each sample. The silhouette coefficient of each sample is obtained using the following formula:

$$s_i = \frac{b_i - a_i}{\max\left(a_i, b_i\right)}, \tag{10}$$

where a_i is the average distances between the sample and other samples in the cluster, and b_i is the minimum value of the average distances between the sample and the samples in other clusters. The average of the SC of all samples is used as the SC of the final clustering results. Generally, its value is in [−1, 1], and a value close to 1 means that its cohesion and separation are relatively better.

(2) Calinski–Harabasz index (CH): The performance evaluation index evaluates the performance of the clustering algorithm by intra- and intercluster covariance of each cluster:

$$s(k) = \frac{\mathrm{tr}\left(B_k\right)(m-k)}{\mathrm{tr}\left(W_k\right)(k-1)}, \tag{11}$$

where m is the number of samples in the training data set, and k is the number of categories. B_k is the intercluster covariance matrix; W_k is the intracluster covariance matrix; and tr is the trace of the matrix. The larger the value of CH and the closer the samples within a cluster are, the greater the difference between the samples of different clusters.

(3) Time cost (TC): The time required for the clustering algorithm from training to convergence is recorded as a time cost to evaluate the real-time performance.

5.2. Experiments and Performance Analysis

5.2.1. Clustering Algorithm Performance Analysis. This experiment compares the performance of the K-means clustering algorithm with two other commonly used clustering algorithms to prove its superiority in extracting traffic accident features. The contrast clustering algorithms are: (1) DBSCAN: the most commonly used density-based clustering algorithm that performs well on nonspherical cluster sample sets and (2) hierarchical clustering: the most commonly used hierarchical clustering algorithm. These three clustering algorithms are all classic and commonly used unsupervised clustering algorithms in the industry. The performance evaluation results on the exact same data set are shown in Table 2.

It can be seen from Table 2 that K-means is significantly better than DBSCAN and hierarchical clustering on SC and TC evaluation indicators but weaker than hierarchical clustering on CH. However, obtaining traffic accident features through clustering is an important step to build an accurate traffic flow prediction model. Meanwhile, traffic flow prediction task requires high real-time performance. If the clustering operation takes too long time, the real-time performance is reduced to some extent, so hierarchical clustering algorithm cannot meet the real-time requirement of short-term prediction. In addition, DBSCAN and hierarchical clustering need to adjust the K value continuously during the clustering process, which may make them more accurate, but the time consumption has increased exponentially. Meanwhile, there may be clusters exceeding the required number in the clustering process. In fact, the traffic accidents features are generally divided into no traffic accidents and different severity of traffic accident based on traffic flow fluctuation. There are obvious differences in the characteristics of each cluster, and the clustering situation is basically clear and limited. The k value can be preset, and the cluster analysis can be performed relatively quickly and accurately through the K-means algorithm. The experimental results also showed that the K-mean algorithm has the best overall performance in clustering performance and efficiency. Therefore, the K-means algorithm is used to

Table 2: Evaluation results of three clustering algorithms.

Clustering algorithm	SC	CH	TC (s)
K-means	**0.436**	11265.3	**0.31**
DBSCAN	0.329	250.5	1.54
Hierarchical clustering	0.366	**89567.0**	18.75

obtain traffic accident information in our work. Generally, the value of k affects the clustering performance [33]. In the process of adjusting parameters, the clustering performances of different k values are shown in Figures 4 and 5.

As can be seen from Figures 4 and 5, the K-means algorithm has an excellent clustering effect at $k = 3$, which is also consistent with the previous analysis and facts. In the physical sense, it indicates the presence or absence of traffic accidents and the impact level of traffic accidents on traffic conditions, including no or slight impact on traffic conditions, a large impact on traffic conditions, and a significant impact on traffic conditions. Therefore, k can be set to 3, which can meet the needs of extracting traffic accident features from traffic flow data. Meanwhile, the K-means algorithm can also obtain better prediction performance and efficiency.

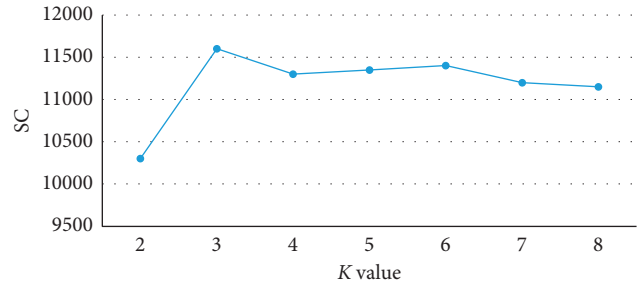

Figure 4: SC of K-means algorithm under different values of k.

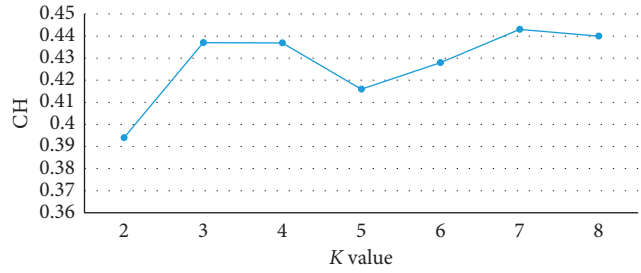

Figure 5: CH of K-means algorithm under different values of k.

5.2.2. Prediction Performance Analysis. Comparative experiments are made with statistical models (AR [4] and ARIMA [4]), machine learning models (kNN-5 [8], kNN-10 [8], and SVM [10]), deep learning models (DeepST [34] and ST-ResNet [17]), TF-Net [24], and the proposed IL-TFNet model to demonstrate its applicability and effectiveness. AR and ARIMA are statistical models with better prediction performance for early traffic flow prediction, and they are widely used as the benchmark model for performance comparison in traffic flow prediction. kNN and SVM are classical and commonly used machine learning algorithms for traffic flow prediction. The suffix of kNN indicates the number of neighbors closest to the sample features are used to predict. LSTM is a widely used deep learning algorithm for time-series prediction. This method can better extract the long- and short-term time characteristics of traffic flow data, thereby improving the prediction accuracy. DeepST is a deep neural network (DNN) based prediction model, which also uses CNN to capture the spatiotemporal features of traffic flow data, and can simulate the spatial distance dependence and temporal compactness. ST-ResNet is the model with best prediction performance proposed in Zhang's work [17]. This model focuses on analyzing the periodic characteristics of traffic flow data while dealing with temporal and spatial characteristics and has achieved good prediction performance. It is a prediction model with leading prediction accuracy in the existing domestic and foreign traffic flow prediction task. TF-Net model is an incremental CNN model that handles some uncertain features such as weather and holiday [24]. The listed are all classic models commonly used in early and recent traffic flow prediction tasks, which are suitable for use as benchmark models for performance comparison to prove the effectiveness of the proposed IL-TFNet model in this study.

The experimental result is shown in Table 3. The three IL-TFNet models listed in Table 3 only differ in the feature selection of model input. Here, IL-TFNet-1 only considers historical traffic flow data. IL-TFNet-2 adds two easily acquired external environmental features, namely, holiday and weather. IL-TFNet-3 introduces traffic accident feature derived by K-means clustering algorithm in addition to holiday and weather. The same data set and the same division method of training and testing data sets are used to ensure the experimental results of the above models are comparable.

As can be seen from Table 3, IL-TFNet model performs better than other prediction models in evaluation indicators, especially IL-TFNet-3 model has the best prediction performance in RMSE value. Compared with ST-ResNet, the value in RMSE is reduced to 8.51% ((16.69–15.27)/16.69). Compared with the TF-Net model, by modifying the prediction model structure and adding traffic accident features, the RMSE value is reduced by 2.61% ((15.68–15.27)/15.68. The smaller the value of RMSE, the more accurate the prediction. The IL-TFNet-2 model adds the above two external environment features, and its prediction performance surpasses the ST-ResNet model. It is worth noting that after adding traffic accident information, the prediction performance of the IL-TFNet-3 model has been further improved. There is a small decrease in the values of MRE and MAE, while the RMSE value is reduced by about 1.55% ((15.51–15.27)/15.51). This not only shows that the introduction of traffic accident features can improve the model performance but also indicates that the K-means-based traffic accident feature extraction method is effective. The K-means clustering algorithm can derive more comprehensive knowledge than CNN from the entire original data set through its unsupervised learning mechanism.

Table 3: Performance comparison results of different models.

Model	MRE	MAE	RMSE
AR	0.5816	30.82	28.24
ARIMA	0.4870	16.56	22.78
kNN-5	0.2832	12.43	17.26
kNN-10	0.2762	12.02	16.78
SVM	0.6749	18.81	23.43
LSTM	0.5538	18.20	23.41
DeepST	0.2998	12.36.	18.18
ST-ResNet	0.2677	12.31	16.69
TF-Net	0.2480	9.35	15.68
IL-TFNet-1	**0.2614**	**10.47**	**16.35**
IL-TFNet-2	**0.2541**	**9.31**	**15.51**
IL-TFNet-3	**0.2516**	**9.26**	**15.27**

A random area in the city is selected to show the prediction details. The actual observed and predicted results of input and output flows on the traffic flow test data set are shown in Figures 6 and 7, respectively, where X axis is the time step; Y axis is the traffic flow value; the red dotted line represents the actual observed data; and the blue curve represents the predicted data obtained by the prediction model. As can be seen from the figure, although most of the predicted values were slightly lower than the actual values in the peak flow period, the predicted values were roughly the same as the observed values in the remaining stages, and the whole predicted data curve was highly consistent with the observed data curve. Therefore, the prediction performance of IL-TFNet-3 model is relatively good from the comparison and specific prediction results.

5.2.3. Incremental Learning Performance Analysis. In this experiment, the change of the model performance during incremental training is observed to prove the feasibility of the incremental learning strategy, as shown in Table 4. During the incremental training process, the accuracy of model prediction increases continuously with the training rounds until the model converges.

Then, the comparison results of prediction accuracy and time cost between the normal and incremental models are given in Figures 8 and 9, respectively. Compared with batch learning methods, the prediction accuracy of the incremental learning method is slightly reduced, but the prediction efficiency is greatly improved, and the training and update time is greatly reduced. The reason for the slight decrease in prediction accuracy is: on the one hand, traditional batch training methods can obtain complete information of the entire training data set in a single training, while incremental learning methods can only use one subtraining data set for each subtraining, and the subsequent training sets are invisible to the model. Therefore, the feature information obtained by the model in a single training is incomplete. On the other hand, since only one subtraining set is trained at a time, after continuous iterative training, the previously learned feature information will gradually be replaced by the new subtraining set feature information, causing the model to forget intermittently what it has learned before. However, as the

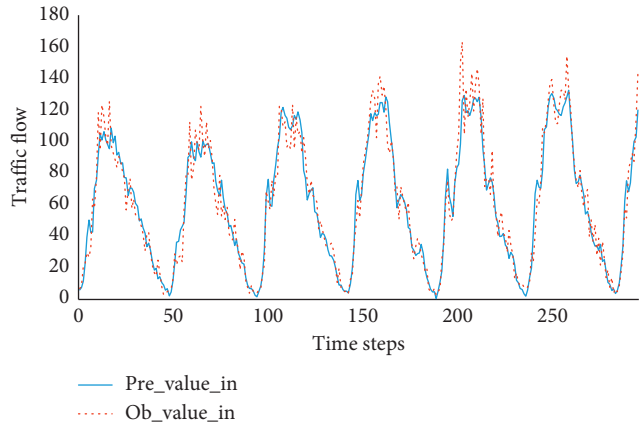

Figure 6: Model prediction results in input flow.

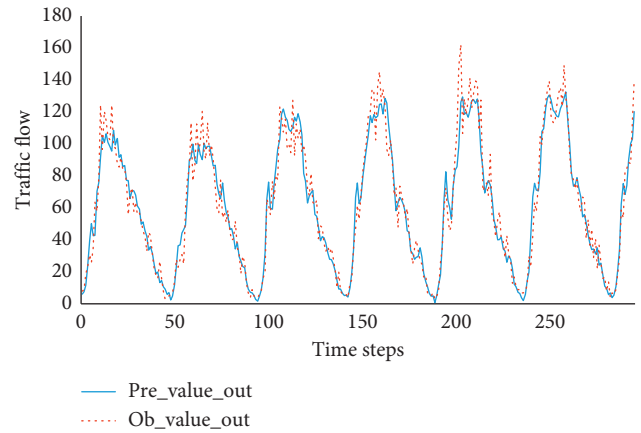

Figure 7: Model prediction results in output flow.

Table 4: Performance of incremental training.

Rounds	MRE	MAE	RMSE
1	0.6825	28.64	37.61
2	0.4037	14.39	21.03
3	0.2791	10.25	18.56
4	0.2648	9.87	16.14
5	0.2516	9.26	15.27

number of training increases, the phenomenon of model feature information forgetting will weaken. Although the traditional batch training method has a slightly higher prediction accuracy, its training cost and update cost are significantly increased compared with the incremental learning method. As can be seen from Figure 9, the training time of the model is reduced by 54.8%, and the updating time is reduced by 67.9% after the incremental learning training method is adopted. Therefore, compared with the significant reduction in model training and update costs, a small decrease in prediction accuracy of the incremental learning model is acceptable. This incremental learning strategy can meet the real-time requirements of short-term traffic flow prediction tasks while ensuring certain prediction accuracy.

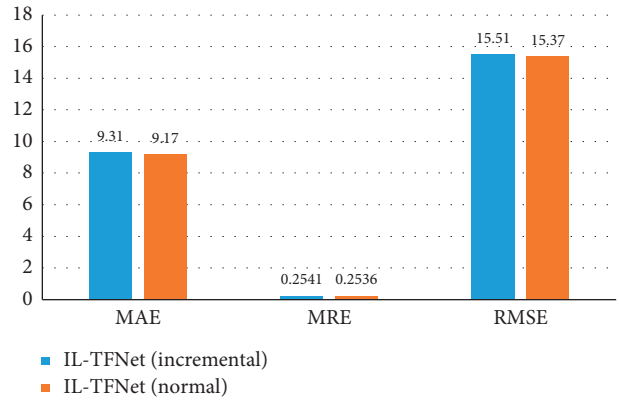

Figure 8: Model prediction performance comparison.

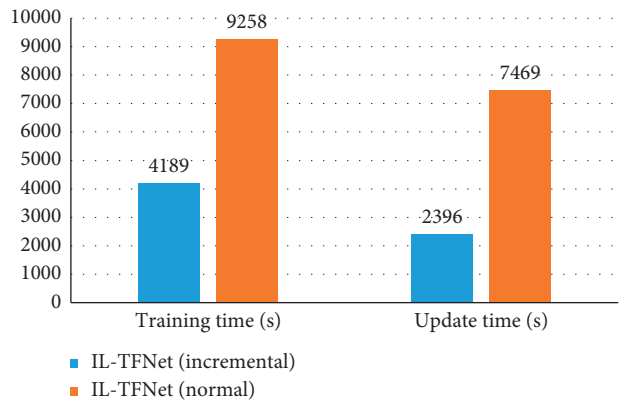

Figure 9: Model training cost comparison.

5.2.4. Real-Time Performance Analysis in Special Case. The real-time performance analysis in special case, that is, holiday is performed in the last experiment. Figure 10 shows the traffic flow during holidays and nonholidays in some region of the city. The traffic data trends of the two are the basically same but different from usual; the traffic flow during holidays is relatively small.

The experiment uses the model to predict the traffic flow during the next holiday period. For traffic flow prediction during holiday, typical samples are traffic flow data during holidays, and the rest are atypical samples. The samples produced in the same environment are more valuable and more helpful for improving model prediction performance. Generally, the samples generated during holidays account for a small proportion in the entire training data set, resulting in relatively low prediction accuracy. Therefore, more typical samples in the holiday are collected through active learning, so as to increase the proportion of typical samples to enable the updated model to learn more feature information under special traffic conditions. The subtraining data set was fused with the latest data, typical samples (samples during holidays), and atypical samples to fine-tune the model. As shown in Figure 11, all the three evaluation indicator values are slightly reduced after model fine-tuning. In particular, the value of RMSE is reduced from 15.44 to 15.27. Experiments show that the model can be fine-tuned

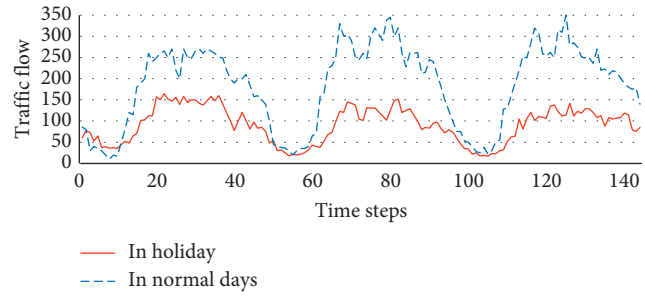

Figure 10: Traffic flow during holiday and nonholiday.

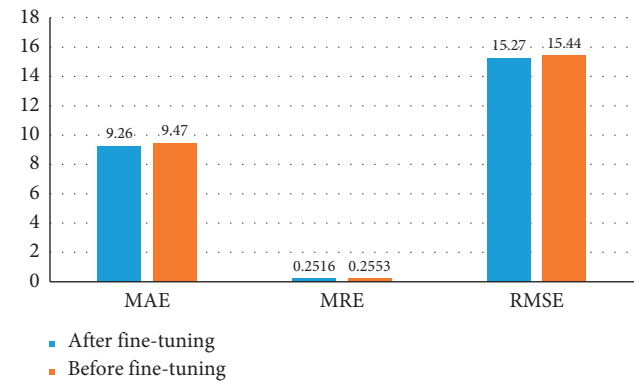

Figure 11: Performance comparison before and after model fine-tuning.

by using the typical sample training data set obtained by active learning to improve the prediction performance in special cases.

The comparison results of the prediction performance before and after model fine-tuning are shown in Figure 12. Overall, the prediction results after fine-tuning the model through the combination of incremental learning and active learning are better, and the trend of the predicted value curve is basically the same as the actual observation value.

Figure 13 shows the percentage of the predicted value and the actual value error of less than 20% under different typical sample ratios (10% and 30%). The abscissa represents the proportion of typical samples, and the ordinate represents the proportion of actual and predicted deviations less than 20%. It can be seen from the figure that when the proportion of typical samples increases, the prediction accuracy of the model is higher under special circumstances, and the model accuracy is slightly improved after model fine-tuning. When the typical sample ratio increases from 10% to 30%, the percentage of the actual and predicted deviations less than 20% after model fine-tuning increases from 55.10% to 64.70%. Even if the model is not fine-tuned, when the proportion of typical samples increases, the prediction accuracy increases slightly, and the percentage of actual and prediction deviations less than 20% increases from 49.70% to 53.60%. Therefore, increasing the proportion of typical samples and fine-tuning the model can improve the performance of traffic flow prediction under special circumstances.

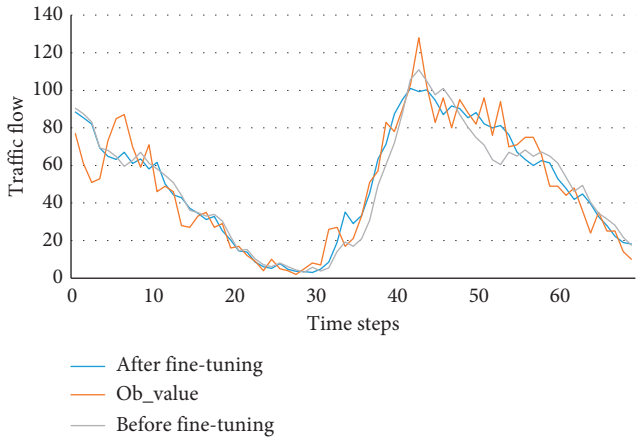

Figure 12: Comparison results of the prediction performance before and after fine-tuning.

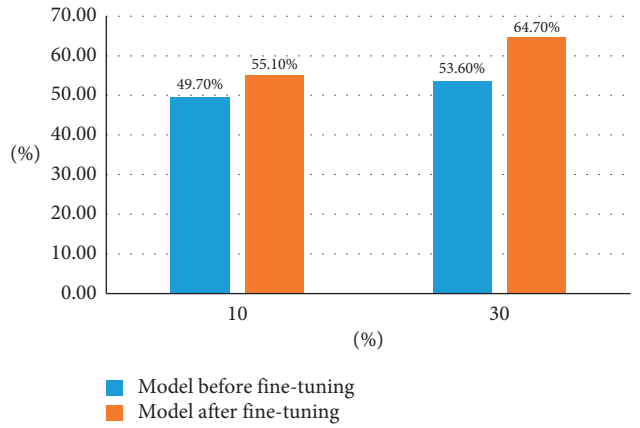

Figure 13: Comparison results of prediction deviation percentage before and after model fine-tuning under different typical sample ratios.

It should be noted that holiday feature has been introduced in IL-TFNet model, and the model without fine-tuning has a certain ability to fit the traffic flow samples during the holidays. The purpose of the fine-tuning strategy in the experiment is to continue to enhance the model expression ability in the case of holidays. However, if the model does not have the prior knowledge of traffic flow samples under special circumstances before fine-tuning, such as a large-scale parade in the city, that is, this special case is not one of the external environment features of the model input. In this case, the prediction performance comparison of the model before and after fine-tuning will be more obvious. Certainly, now the model performance after fine-tuning on the holiday testing data set is also acceptable.

6. Conclusions and Future Work

An incremental learning-based CNN-LTSM model, IL-TFNet, as a solution of mobile application is proposed for short-term traffic flow prediction in this study. A lightweight deep learning architecture, IL-TFNet model architecture is designed to process spatiotemporal features and external environmental features simultaneously to improve model prediction performance and reduce the model complexity and cost. Meanwhile, K-means clustering algorithm is applied to derived unknown and implicit traffic accident information as the uncertainty feature, which contributes greatly to model performance improvement. Furthermore, instead of traditional batch learning method, the incremental learning with high real-time performance and low computational resource overhead is applied for model training and update, thereby reducing training and update costs and improving prediction efficiency, while enabling the model suitable for mobile applications. The idea of combining incremental learning with active learning is deliberately proposed to fine-tune the prediction model to improve the prediction accuracy further under special situation while ensuring the real-time performance.

The experimental results demonstrate the excellent prediction accuracy and efficiency of the proposed IL-TFNet model for short-term traffic flow prediction. However, there remain some open problems. It is urgent and valuable to use a complete traffic flow data set containing various external environmental features to prove the model performance further. In addition, the missing data completion method should be considered in the future, which is important to improve the prediction accuracy as well.

Data Availability

The data will be available on the link https://pan.baidu.com/s/11ZOn_1Xzov_r4WKfwKXzJQ (password: hi9g).

Conflicts of Interest

The authors declare that they have no conflicts of interest to this work.

Acknowledgments

The authors appreciate the support from the Zhejiang Provincial Natural Science Foundation of China (LY20F020015), the National Science Foundation of China (61702517, 61972121, 61902345, and 61772525), the Defense Industrial Technology Development Program (No. JCKY2019415C001), and the Open Project Program of the State Key Lab of CAD & CG (Grant no. 2109), Zhejiang University.

References

[1] G. De La Torre, P. Rad, and K. K. R. Choo, "Driverless vehicle security: challenges and future research opportunities," *Future Generation Computer Systems*, vol. 108, 2018.

[2] N. Zhang et al., "DynaCAS: computational experiments and decision support for ITS," *IEEE Intelligent Systems*, vol. 23, p. 6, 2008.

[3] J. Zhang, F.-Y. Wang, K. Wang, W.-H. Lin, X. Xu, and C. Chen, "Data-driven intelligent transportation systems: a survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1624–1639, 2011.

[4] J. Abdi, B. Moshiri, B. Abdulhai, and A. K. Sedigh, "Short-term traffic flow forecasting: parametric and nonparametric approaches via emotional temporal difference learning," *Neural Computing and Applications*, vol. 23, no. 1, pp. 141–159, 2013.

[5] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: theoretical basis and empirical results," *Journal of Transportation Engineering*, vol. 129, no. 6, pp. 664–672, 2003.

[6] S. Jin, D.-h. Wang, C. Xu, and D.-f. Ma, "Short-term traffic safety forecasting using Gaussian mixture model and Kalman filter," *Journal of Zhejiang University Science A*, vol. 14, no. 4, pp. 231–243, 2013.

[7] Q.-J. Kong, Q. Zhao, C. Wei, and Y. Liu, "Efficient traffic state estimation for large-scale urban road networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 1, pp. 398–407, 2013.

[8] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: a deep learning approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 1–9, 2014.

[9] H. Gao, L. Kuang, Y. Yin, B. Guo, and K. Dou, "Mining consuming behaviors with temporal evolution for personalized recommendation in mobile marketing apps," *Mobile Networks and Applications*, vol. 25, no. 4, pp. 1233–1248, 2020.

[10] M. Castro-Neto, Y.-S. Jeong, M.-K. Jeong, and L. D. Han, "Online-SVR for short-term traffic flow prediction under typical and atypical traffic conditions," *Expert Systems with Applications*, vol. 36, no. 3, pp. 6164–6173, 2009.

[11] X. Jiang and H. Adeli, "Dynamic wavelet neural network model for traffic flow forecasting," *Journal of Transportation Engineering*, vol. 131, no. 10, pp. 771–779, 2005.

[12] Y.-S. Jeong, Y.-J. Byon, M. M. Castro-Neto, and S. M. Easa, "Supervised weighting-online learning algorithm for short-term traffic flow prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 4, pp. 1700–1707, 2013.

[13] E. I. Vlahogianni, M. G. Karlaftis, and J. C. Golias, "Optimized and meta-optimized neural networks for short-term traffic flow prediction: a genetic approach," *Transportation Research Part C: Emerging Technologies*, vol. 13, no. 3, pp. 211–234, 2005.

[14] S. Sun, C. Zhang, and G. Yu, "A Bayesian network approach to traffic flow forecasting," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 1, pp. 124–132, 2006.

[15] L. Li, B. Du, Y. Wang, L. Qin, and H. Tan, "Estimation of missing values in heterogeneous traffic data: application of multimodal deep learning model," *Knowledge-Based Systems*, vol. 194, Article ID 105592, 2020.

[16] H. Gao, W. Huang, and Y. Duan, "The cloud-edge-based dynamic reconfiguration to service workflow for mobile ecommerce environments," *ACM Transactions on Internet Technology*, vol. 21, no. 1, pp. 1–23, 2021.

[17] S. Dunne and B. Ghosh, "Regime-based short-term multivariate traffic condition forecasting algorithm," *Journal of Transportation Engineering*, vol. 138, no. 4, pp. 455–466, 2011.

[18] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," 2016.

[19] Z. Xue and Y. Xue, "Multi long-short term memory models for short term traffic flow prediction," *IEICE TRANSACTIONS on Information and Systems*, vol. E101.D, no. 12, pp. 3272–3275, 2018.

[20] Y. Tian, K. Zhang, J. Li, X. Lin, and B. Yang, "LSTM-based traffic flow prediction with missing data," *Neurocomputing*, vol. 318, pp. 297–305, 2018.

[21] T. Ma, Z. Zhou, and B. Abdulhai, "Nonlinear multivariate time-space threshold vector error correction model for short term traffic state prediction," *Transportation Research Part B: Methodological*, vol. 76, pp. 27–47, 2015.

[22] N. G. Polson and V. O. Sokolov, "Deep learning for short-term traffic flow prediction," *Transportation Research Part C: Emerging Technologies*, vol. 79, pp. 1–17, 2017.

[23] J. An, L. Fu, M. Hu, W. Chen, and J. Zhan, "A novel fuzzy-based convolutional neural network method to traffic flow prediction with uncertain traffic accident information," *IEEE Access*, vol. 7, pp. 20708–20722, 2019.

[24] F. Yu, J. Fang, B. Chen, and Y. Shao, "An incremental learning based convolutional neural network model for large-scale and short-term traffic flow," *International Journal of Machine Learning and Computing*, vol. 11, no. 2, 2021.

[25] V. Losing, B. Hammer, and H. Wersing, "Incremental on-line learning: a review and comparison of state of the art algorithms," *Neurocomputing*, vol. 275, pp. 1261–1274, 2018.

[26] F. QIN, N. Gao, Y. Peng, Z. Wu, S. Shen, and A. Grudtsin, "Fine-grained leukocyte classification with deep residual learning for microscopic images," *Computer Methods and Programs in Biomedicine*, vol. 162, pp. 243–252, 2018.

[27] C. Szegedy, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Boston, MA, USA, June 2015.

[28] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the Advances in Neural Information Processing Systems*, Lake Tahoe, NV, USA, December 2012.

[29] F.-W. Qin, J. Bai, and W.-Q. Yuan, "Research on intelligent fault diagnosis of mechanical equipment based on sparse deep neural networks," *Journal of Vibroengineering*, vol. 19, p. 4, 2017.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, June 2016.

[31] V. Lomonaco and D. Maltoni, "Comparing incremental learning strategies for convolutional neural networks," in *Proceedings of the IAPR Workshop on Artificial Neural Networks in Pattern Recognition*, Springer, Cham, Switzerland, September 2016.

[32] X. Yang, S. Zhou, and M. Cao, "An approach to alleviate the sparsity problem of hybrid collaborative filtering based recommendations: the product-attribute perspective from user reviews," *Mobile Networks and Applications*, vol. 25, no. 2, pp. 376–390, 2020.

[33] Y. Yin, Z. Cao, Y. Xu, H. Gao, R. Li, and Z. Mai, "QoS prediction for service recommendation with features learning in mobile edge computing environment," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 4, pp. 1136–1145, 2020.

[34] J. Zhang, Y. Zheng, D. Qi et al., "DNN-based prediction model for spatial-temporal data," 2016.