Transportation Research Part C 126 (2021) 103062

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Estimating traffic flow states with smart phone sensor data

Wenwen Tu [a], Feng Xiao [b,*], Lu Li [c], Liping Fu [d]

[a] School of Transportation & Logistics, Southwest Jiaotong University, PR China
[b] School of Business Administration, Southwestern University of Finance and Economics, PR China
[c] Business School, Sichuan University, PR China
[d] Department of Civil & Environmental Engineering, University of Waterloo, Waterloo, Ontario, Canada

ARTICLE INFO

Keywords:
Traffic flow state classification
Smartphone senor data
Deep Belief Network

ABSTRACT

This study proposes a framework to classify traffic flow states. The framework is capable of processing massive, high-density, and noise-contaminated data sets generated from smartphone sensors. The statistical features of vehicle acceleration, angular acceleration, and GPS speed data, recorded by smartphone software, are analyzed, and then used as input for traffic flow state classification. Data collected by a five-day experiment is used to train and test the proposed model. A total of 747,856 sets of data are generated and used for both traffic flow state classification and sensitivity analysis of input variables. After applying various algorithms to the proposed framework, the study found that acceleration and angular acceleration data can increase the accuracy of traffic flow classification significantly. When the hyper-parameters of the Deep Belief Network model are optimized by the Differential Evolution Grey Wolf Optimizer algorithm, the classification accuracy is further improved. The results have demonstrated the effectiveness of using smartphone sensor data to estimate the traffic flow states and shown that our proposed model outperforms some traditional machine learning methods in traffic flow state classification accuracy.

1. Introduction

Timely and accurate monitoring of traffic congestion is the foundation of an intelligent transportation system. It allows government administration to make data-driven decisions and provides real-time traffic information for travelers, lowering system costs and enhancing social efficiency.

Traffic flow state classification and assessment are essential to traffic congestion monitoring. Past traffic flow theory based on the Fundamental Graph divides the traffic flow into two phases: the low-density free flow and the high-density congested flow (Kerner, 1998, 1999). The three-phase traffic theory further divides the congested flow into another two phases, the synchronized flow and the wide moving jam (Kerner, 2004). The three-phase traffic flow theory has further enriched the traffic flow theory and has been widely applied in practice. Currently, one of the most widely used classification methods is to roughly divide the traffic flow into three states: the free-flow state, the steady flow state, and the congested flow state (Dong et al., 2011; Kerner et al., 2007; Zhu et al., 2012). Under the free-flow state, there are fewer vehicles on the road and lower traffic density, and the vehicles can easily reach the designed speed. Under the steady flow state, the traffic flow on the road reaches the road capacity, and the head-space between the vehicles is shortened, so the ability for the vehicles to operate and change lanes may be restricted to a certain extent. While under the congested

* Corresponding author.
E-mail address: xiaofeng@swufe.edu.cn (F. Xiao).

https://doi.org/10.1016/j.trc.2021.103062
Received 10 June 2019; Received in revised form 20 February 2021; Accepted 20 February 2021
Available online 13 March 2021

state, the vehicles on the road always follow the vehicles ahead, make frequent stops, and move forward slowly, hence, creating unstable driving conditions.

In the literature, most of the previous approaches have relied on time-stamped speed data derived from GPS position data, which may contain the following significant defects: First, when traveling underground or in urban canyons, GPS signals may not be received accurately or may even be lost entirely. As a result, GPS data could be incomplete or inaccurate. Second, speed information may not be enough to derive the accurate traffic flow states. For example, slow speed may be caused by other reasons than congestion. Especially on a local street when there are no enough floating cars, estimating the traffic state based on speed could vary and become unreliable. Zeng et al. (2011) found that the number of probe cars may change over time. If GPS data is insufficient, the accuracy of the estimation of the link speed may decrease sharply. Under such circumstances, it is necessary to introduce a new methodology that does not solely rely on GPS and speed. Modern smartphone build-in sensors provide many additional data that could be used for accurate traffic estimation. As technologies evolve, vehicle-based driving data can now be collected directly from smartphones. Collecting data from smartphones can address many limitations of traditional fixed sensors, such as complex installations, high maintenance costs, and insufficient coverages. Moreover, user penetration can be significantly improved as a result of the ubiquitous use of smartphones.

Thus, this paper is particularly concerned with applying data from smartphones for the estimation of traffic states. Nowadays, smartphone sensor data typically include GPS data, acceleration data, and angular acceleration data. Acceleration is defined as the ratio of the change in speed to the time taken. Angular acceleration describes the rate of change of angular speed magnitude and direction to time. When the smartphone is fixed in the vehicle, the acceleration and angular acceleration data can reveal more detailed information of a vehicle's behavior in the traffic. Besides, smartphone sensor data is less affected by cellular signals compared to GPS data. When the cellular signal is interrupted, smartphone sensor data can be stored temporarily in the phone memory and uploaded later after the signal is restored. However, GPS data will be lost permanently. Most previous studies used mobile phone acceleration data to analyze trip modes, drivers' aggressive and risky behaviors, and vehicle braking events. In contrast, this paper focuses on mining the sensor data of smartphones to classify the traffic flow states. The acceleration data and angular acceleration data of the smartphone sensor are regarded as additional inputs to the GPS data. After applying deep learning and other linear/nonlinear classification algorithms to the proposed framework, this study demonstrates that acceleration and angular acceleration can help further improve the traffic flow classification accuracy. The intrinsic mechanism is explained by analyzing the significant variables.

The massive and high-density data collected from smartphone sensors surely also contains noises. Hence, additional data processing and modeling are required to improve the accuracy of traffic flow state classification. In the past, traditional machine learning methods have been shown to have limitations in addressing big and noisy data (Farabet et al., 2013; Hinton et al., 2012; Krizhevsky et al., 2012; LeCun et al., 2015; Ma et al., 2015; Tompson et al., 2014). In contrast, deep learning models have achieved top-tier performance in many classification tasks in recent years (Chen et al., 2014b; Glorot et al., 2011; Lee et al., 2009b; Socher et al., 2012). The potential of deep learning algorithms is explored in this research. In this paper, traffic flow states are classified using 747,856 pieces of data collected from smartphone sensors. For comparison, the real traffic states are obtained from the Variable Message Signs (VMS) system, which are considered accurate and reliable. The traditional methods, e.g., VMS and other GPS-based systems, generally need a large number of shared users using GPS and the combination of massive multi-resource data to achieve accuracy. In contrast with these methods, the method we propose here can classify the traffic flow states accurately and derive the running condition by only analyzing the data of one car, which greatly reduces the data requirement and ultimately reduces the data collection cost. In our experiments, the nonlinear and linear classification algorithms have average accuracies of less than 0.6336 when only speed data is considered. By adding acceleration and angular acceleration data and applying nonlinear classification algorithms such as Deep Belief Network (DBN), Random Forest (RF), Support Vector Machine (SVM), and K-Nearest Neighbor (KNN) to the proposed framework, an average accuracy of 0.8422 is obtained. Among the nonlinear classification algorithms, the deep learning DBNb model obtained the highest average accuracy of 0.9607. Besides, the analysis of the significance of input variables shows that the most critical five variables are all related to acceleration and angular acceleration. We also show that the accuracy can be further increased by optimizing the hyperparameters of the model. For example, as the number of supervised iterative steps of the DBN model increases from 20 to 1000, the average classification error reduces from 9.91% to 2.44%. After using the DEGWO algorithm to optimize the hyperparameters of the DBN model, the classification error further decreases from 3.92% to 0.76% as the upper-bound of the number of supervised iterations increases from 50 to 1500.

2. Literature review

2.1. Research on classification of traffic flow states

The classification of urban road traffic states is an essential part of the intelligent transportation system. At present, the research on the classification of the traffic flow states can mainly be classified into two categories.

One is to study the variation of traffic flow characteristics such as speed, occupation, and flow, and combine the traffic flow theory to establish the threshold of characteristics and classify the traffic flow states. Jiang et al. (2006) analyzed the variation in traffic flow parameters and used traffic flow characteristics to realize real-time detection of regular and irregular congestion. Zhuang et al. (2006) proposed a relative incremental basis for traffic congestion based on flow and density indicators and achieved automatic traffic congestion inspection. Guan and He (2007) divided the traffic flow states into four levels according to the flow-density plane. Banaei-Kashani et al. (2011) analyzed the spatial and temporal traffic sensor data (the Los Angeles County dataset) and evaluated traffic flow patterns with the speed indicators. Treiber and Kesting (2012) proposed a quantitative approach describing congested traffic patterns based on the density metrics and the speed time series. Chen et al. (2014a) preprocessed the data, collected by the fixed detectors, on

typical roads in Qingdao city and established the identification model and thresholds for the traffic flow states. Although the identification method based on the predetermined threshold performs well in highway traffic flow, it may be insufficient to estimate urban traffic flow states accurately due to the traffic parameters thresholds for complex urban road system cannot be precisely calibrated (Liang et al., 2015).

The other branch is to study the traffic parameters and combine statistical analysis methods and machine learning algorithms to improve the performance of traffic flow state classification. Liu et al. (2008) proposed a traffic congestion evaluation model based on cumulative logistic regression. Ko and Guensler (2005) applied the Gaussian mixture distribution to identify traffic conditions between congested and non-congested states. Lozano et al. (2009) proposed a K-means clustering algorithm for congestion level recognition. Montazeri-Gh and Fotouhi (2011) proposed a K-means clustering method for traffic state recognition to identify the driving environment of autonomous vehicles. Antoniou et al. (2013) performed traffic flow state estimation and prediction based on data-driven calculation methods. They found that this method is suitable for traffic simulation models and can replace the traditional speed-density relationship method. Zhan et al. (2020) proposed a statistical modeling framework of link-based traffic state estimation. Based on this proposed framework, the License-plate recognition (LPR) data from a subset of intersections was used for signal timing inference and queue length approximation. Deng et al. (2020b) utilized the cloud model to evaluate the congestion features in the road section based on the speed data from floating vehicles. Yang and Qiao (1998) proposed a pattern recognition method based on the self-organizing neural network to achieve the division of highway traffic states. Under small-scale actual data tests, the experimental results showed that this method is feasible in practice. Habtie et al. (2016) proposed a real-time traffic flow estimation model for urban road networks based on the neural network model. The results showed that the proposed model could provide reliable traffic state information. Xu et al. (2020a) constructed the road traffic running characteristics reference sequences (RTRCRS) based on microwave sensor data and then proposed a traffic state prediction approach based on the Kernel K-nearest neighbor (Kernel-KNN) model. However, the accuracy of the constructed RTRCRS information will affect Kernel-KNN's choice of reference roads, therefore, influencing prediction accuracy. Zhang and Yang (2020) estimated the freeway traffic speed based on the detection vehicle and roadside sensor detector data. According to the calculation results of machine learning methods (SVM, RF, gradient-enhanced decision tree, XGBoost, and artificial neural network), a probe data system with statewide deployment could replace the roadside sensor system in traffic state estimation. Chikaraishi et al. (2020) researched the applicability of various machine learning models based on traffic volume and time occupancy from loop detectors during traffic network disruption. They pointed out that compared with the XGBoost, the deep neural network (DNN) had better stability and interpretability in short-term prediction.

Recently, graph-based methods have also been used for traffic state estimation. Jo et al. (2018) proposed an image-to-image traffic speed prediction approach based on a convolutional neural network (CNN). The average traffic speed data from taxi probes and the road network information were transformed as map images, which were used as CNN inputs to learn the spatio-temporal correlations between traffic states. However, a large amount of observed data was needed to enable the proposed approach to capture various traffic characteristics and identify correlations. Xu et al. (2019) proposed a novel graph embedding recurrent neural network (GERNN) framework based on the road network graph and the traffic flow data obtained from the loop detectors to predict traffic state. Uareemitr et al. (2019) processed video images by integration with time-spatial image (TSI) technology, and achieved space-mean-speed, flow and density estimation. Xu et al. (2020b) proposed a deep learning framework with graph embedding and the generative adversarial network. They used information collected from fixed detectors in adjacent links to estimate the missing speed and volume data of a single detector or multi-detectors. Zhang et al. (2021) developed a Dynamic Node-Edge Attention Network (DNEAT) to predict the ride-hailing OD demand. They captured time-varying OD flow patterns with a dynamic graph topology rather than static or pre-defined ones.

2.2. Traffic congestion monitoring with mobile phone GPS data

Technology based on mobile devices has proven its usefulness in collecting activity-based travel diary data (Bao and Intille, 2004). Such technology has been argued to be able to reduce both the respondent and the researcher's burdens. Moreover, the accuracy of the data would be better than those collected by conventional survey methods. Herrera et al. (2010) conducted a field experiment to estimate the feasibility of utilizing GPS-enabled mobile phones for highway traffic monitoring operations. Experimental results showed that when the penetration rate of GPS-enabled mobile phones reaches 2–3%, the measured value of traffic flow speed will eventually approach an accurate value. Händel et al. (2014) analyzed approximately 4500 driving hours of road vehicle traffic data collected during a ten-month-long project, i.e., the Berkeley Mobile Millennium Project. In this project, a framework was presented to deploy a smartphone-based measurement system for road vehicle traffic monitoring and usage-based insurance. Miranda-Moreno et al. (2015) utilized the GPS capabilities of Android and iOS smartphones in Québec City to collect data on more than 30,000 trips in three weeks. They used the average and difference of speed at the link layer to map traffic congestion based on these data. Yu and Gu (2019) proposed a Graph Convolutional Generative Autoencoder (GCGA) model to estimated traffic speed. By utilizing the GPS records of individual vehicle trips, they found that the GCGA model had the ability to learn spatial correlation from incomplete historical input. Chen et al. (2019) studied a high-order tensor Bayesian probability matrix factorization model for speed data interpolation task. After calculating the travel speed observations collected by smartphones GPS data, they showed that in the case of random loss and fiber loss, the third-order based on section, day, and time interval could get the best results. Liu et al. (2020) mapped GPS trajectory records of the online car-hailing into cells and proposed a fully convolutional model based on semantic segmentation technology to estimate the traffic speed.

2.3. Traffic congestion monitoring with accelerometer data and gyroscope data on mobile phones

The accelerometer data has recently been successfully used to detect transportation modes (Lee and Cho, 2014; Vlahogianni and Barmpounakis, 2017; Xu et al., 2011) and study drivers' aggressive and risky behavior (Predic and Stojanovic, 2015). Choudhury et al. (2017) calculated an initial position from a GPS receiver. In light of the mapping association, the position change was estimated by the GPS data and vertical vehicular motion data from the accelerometer. The authors utilized the GPS-and-accelerometer-derived location to implement vehicle navigation. Unlike such research objects, our article focuses on estimating the road congestion state from smartphone sensor data, including three-dimensional accelerations and angular accelerations.

In the field of traffic congestion estimation, Mohan et al. (2008) focused on the detection of bumps and potholes, braking, and honking, using the data from mobile smartphones equipped with an array of sensors (GPS, accelerometer, microphone). Pholprasit et al. (2015) deployed raw data from the accelerometer sensor. A pattern matching technique was proposed to identify driving events from driving patterns and sensor data. The driving events include brake, acceleration, turn left, turn right, lane change left, and lane change right. Zheng et al. (2014) described the vehicle mobility status specifies normal status, accelerating status, and decelerating status. A state machine was utilized to filter out the noise of acceleration from smartphones and recognize vehicle mobility status events. The traffic signal schedule can be generated based on the reported events from a set-off vehicle and the queue model. Hsieh et al. (2014) collected sensor data from 100 scooters and proposed a model of braking event detection. An accurate prediction of braking events can be evaluated 15 m ahead of the intersections. The trained model can provide a response time of about one second and about 91% accuracy in detecting red-light runners. Bhoraskar et al. (2012) proposed machine-learning algorithms to distinguish a bumpy road from a smooth one and detect braking events. The algorithms correctly identify 18 out of the 20 bump events by utilizing the features present in accelerometer data. No smooth road patch is incorrectly identified as a bump. Out of 37 braking events in the run, the algorithm accurately recognizes 29, with one false positive. In the above studies, the main research goals are to identify traffic events, such as braking, acceleration, and deceleration. However, the isolated traffic events of an individual vehicle cannot entirely and precisely characterize the traffic flow state of the road section. In our study, the congestion level of the road section is detected by the sensor data in a single-vehicle. Additionally, the above studies only used the accelerometer data (some studies may be limited by the smartphone technology at the time). We have further merged accelerometer data and the angular acceleration data of the gyroscope in the classification model.

Zong and Wen (2015) evaluated the traveling speed of a vehicle with the values of various characteristics, including time-domain characteristics, frequency-spectrum characteristics, and power-spectrum characteristics of acceleration and angular acceleration in advancing direction and vertical direction. The speed value is classified into 13 states. The proportion is over 80% when traveling in the first four states. The proportion succeeds by 60% when the traveling speed is between 4 m/s and 7 m/s. Petraki et al. (2020) used the smartphone sensors (accelerometer, gyroscope, magnetometer, and GPS) data to analyze the impact of road and traffic characteristics on driver behavior. The behavior examinations included the frequencies of harsh accelerations, harsh decelerations, and concentrations degree. They found that extreme events increased in road segments when the average traffic flow per lane increased in the respective areas. Li et al. (2020) used the acceleration sensor, gyroscope, and magnetic field sensor to recognize driver behavior such as speed change, harsh braking, and lane-changing. Mandal et al. (2020) estimated the road roughness based on the accelerometer sensor data from the carried smartphone by 2-wheelers travelers. The traffic flow state was estimated by the road roughness, bumps, WIFI hotspot density, and intersection density. In the best case, the 83.4% classification accuracy by the Gradient boosting approach was achieved.

In our study, we do not focus on estimating the speed of the test vehicle but on assessing the traffic flow state of the whole road section. To enhance classification accuracy, we utilize acceleration and angular acceleration data in three directions, not just in advancing and vertical direction. To extract the essential knowledge from the original data, it is usually necessary to utilize a complicated method (Reddy et al., 2008). Processing such raw data may become challenging, especially when the data size is significant. Therefore, advanced algorithms are required to take full advantage of mixed and massive mobile phone sensor data.

2.4. Traffic flow analysis with deep learning algorithms

In the past decades, many traditional machine learning methods were used in traffic state classification. However, with a considerable amount of noisy data from the mobile sensor, computing time and calculation accuracy were far from ideal (Schmidhuber, 2015). The challenge of predicting traffic flows comes from the sharp nonlinearities due to transitions between free flow, breakdown, recovery, and congestion. In recent years, DNN has become a great success in processing massive data (Bengio et al., 2006; LeCun et al., 2015; LeCun et al., 1989; Poultney et al., 2007). DNN is a type of Artificial Neural Networks (ANN), obeys the universal approximation theory. It ensures that a neural network has global approximation ability for any nonlinear function if enough hidden units are given. Furthermore, by using a deep structure, DNN overcomes the shortcomings of the exponential explosion and insufficient feature learning of traditional ANN such as Multi-layer Perceptron (MLP).

Recently, some researchers have explored this approach in several transportation tasks like traffic flow prediction and traffic sign classification. Lv et al. (2015) applied a deep architecture model using autoencoders and the model-built blocks to represent traffic flow features for prediction. The experiments proved that the proposed model for traffic flow prediction has good performance. Huang

et al. (2014) proposed a deep architecture for traffic flow prediction. The model consists of two parts, a DBN at the bottom and a multi-task regression layer at the top. The experiments demonstrated its excellent performance to learn traffic flow features. Many other past studies (Jiang et al., 2018; Shao et al., 2018; Uddin et al., 2017; Zhao et al., 2017) have shown that deep learning architectures are capable of handling the nonlinear Spatio-temporal traffic flow data effectively. Sekuła et al. (2018) studied applications of vehicle probe speeds estimated based on GPS data, automatic traffic recorder counts, and a fully-connected feedforward multi-layer ANN model to estimate hourly volumes. Gu et al. (2019) proposed a two-layer deep learning framework by combining the Long Short-Term Memory (LSTM) neural network with the Gated Recurrent Unit (GRU) neural network and utilized remote traffic microwave sensors to examine framework performance. Wang et al. (2019) proposed a path-based deep learning framework for predicting speed and used AVI data in empirical studies. Information is fused through multiple superimposed Bi-LSTM layers. Based on the fully connected layer and each path information, the network speed prediction is realized. In a series of scenarios, the proposed model was found to be superior to multiple benchmark methods. Wu et al. (2018) proposed a DNN-based traffic flow prediction model (DNN-BTF) based on traffic flow data. The model determines the importance of past traffic flow based on automatic learning of attention. The author visualizes how the DNN-BTF model understands traffic flow data. The calculation results based on PeMS data showed that the proposed method was superior to the most advanced methods. Jahanjoo et al. (2020) utilized the DBN model and the triaxial acceleration from personal smartphones to classify elderly falls and daily activity. Deng et al. (2020a) optimized the DBN model's parameters with a quantum-inspired differential evolution algorithm to enhance the robustness and global search capabilities. Based on rolling bearings' vibration data, the proposed model could reach 99.7% fault classification accuracy. Xing et al. (2021) considered the temperature parameter on the learning process of DBN to adjust the transfer functions and activation condition. The results showed that the optimal temperature was conductive to extracting relevant features and increasing the learning capability.

3. Data collection and analysis

To collect the data, we fixed the smartphone with built-in motion sensors in the center of the car to guarantee that the phone moves rigorously together with the vehicle. The y-axis is set to be parallel to the driving direction, and the z-axis is kept perpendicular to the ground. The positive x-axis is to the right of the driving direction. A smartphone app is developed for data acquisition and transmission. The acceleration data from accelerometers comprises three variables of AX, AY, and AZ, representing the acceleration values in the directions of the x-axis, y-axis, and z-axis. We define GX, GY, and GZ to represent the angular acceleration from triaxle gyroscopes in the three directions in the x-axis, y-axis, and z-axis. The GPS module provides time and space information by the latitude and longitude coordinates. The device speed can then be calculated accordingly.

The software installed in the smartphone retrieves the data from the accelerometer and the triaxle gyro module at a frequency of 50 Hz. Each data contains a timestamp and smartphone sensor records collected by the accelerometer and a three-axis gyroscope. The acquisition accuracies of the speed, acceleration, and angular acceleration are 0.01 $\mathrm{m/s}$ and 0.001 g and 0.001 $\mathrm{rad/s^2}$, respectively. In the meantime, the current speed is calculated every second. The software records fifty pieces of acceleration and angular acceleration data per second but only one piece of speed data per second. To match the sizes of acceleration and angular acceleration data, we copy the speed data to be the same size as fifty pieces per second.

In this study, the 2nd Ring Road in Chengdu is selected as the experimental site due to its simple environment and good road conditions. It is an expressway with no signal control. Thus, the probe car can keep running cyclically without interruption and encounter different traffic states. The benchmark of the traffic state is obtained from Variable Message Signs (VMS). There are 54 sets of VMS on the Chengdu Second Ring Road Expressway. VMS relies on the data acquisition system to provide real-time data and release traffic information. The average speed of the road section is calculated by combining floating car data (taxi GPS data) and high-definition video data. Based on the average speed thresholds, the system categorizes the road traffic condition into three types of traffic flow states: free flow when speed is faster than 40 km/h, steady flow when speed is between 20 and 40 km/h, and congested flow when speed is less than 20 km/h. Speed is also an input variable in our model. The difference is that, for VMS, the average speed is used, which contains the information of multiple cars, while in our model, only the speed of one car is used.

A total of 747,856 observations were obtained from this five-day-long experiment. Eight critical variables are selected in this experiment: accelerations in three-axis directions, AX, AY, AZ; angular accelerations in three-axis directions, GX, GY, GZ; vehicle speed, V; and the corresponding traffic flow states. 45.15% of the data is in the free-flow state, 30.04% of the data is in the steady flow state, and the remaining 24.81% is in the congested flow state. The distribution of the data in the three traffic flow states can be considered reasonably balanced. Table 1 illustrates the time of data collection, the proportion of each traffic flow state, skewness, and kurtosis of the state data each day.

Table 1
The information of collected data with corresponding traffic flow states.

Date	Start time	End time	Data volume	The percentage of each state			Skewness	Kurtosis
				Free flow	Steady flow	Congested flow		
The 1st day	08:25:00	09:32:51	203,538	19.59%	43.98%	36.43%	−0.03	1.69
The 2nd day	16:54:42	17:39:52	135,518	68.78%	31.22%	0%	0.81	1.66
The 3rd day	08:32:53	09:20:47	143,723	50.21%	20.73%	29.06%	0.53	1.57
The 4th day	08:22:00	09:03:27	124,276	65.11%	21.42%	13.47%	1.31	3.34
The 5th day	08:29:00	09:15:57	140,812	36.58%	25.87%	37.55%	0.11	1.34

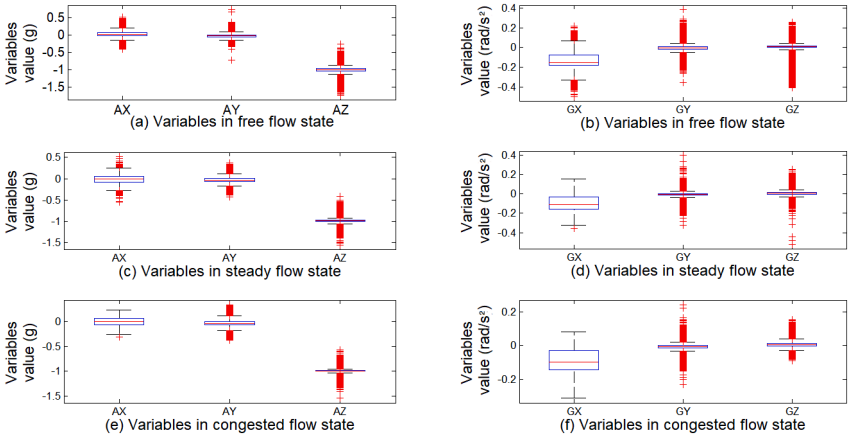

Fig. 1. Boxplot of AX, AY, AZ, GX, GY, and GZ data in different traffic flow states.

After calculating the boxplot of the variables, AX, AY, AZ, GX, GY, GZ, and V (Fig. 1), we find that the values of most variables are concentrated around 0. Only variable AZ is around the value of -1 (Here, the unit of AX, AY, and AZ is "g", which represents 1 unit of acceleration of gravity, i.e., $1\mathrm{g} = 9.8\ \mathrm{m/s^2}$). The reason is that the z-axis is perpendicular to the ground. Thus, there is always an offset caused by gravity. Compared with the variables along the x-axis, the variables in other directions contain more anomalies and fluctuations. The x-axis is the lateral direction, which is perpendicular to the vehicle's driving direction. Since the type of road in our experiment is a highway with a large turning radius, the variation of x-direction data is relatively small. As observed from the boxplots in Fig. 1, distributions of acceleration and angular acceleration are both similar in all three states. However, they can still help improve the classification accuracy in our framework.

4. A traffic flow state classification framework based on smart phone sensor data

In the proposed framework, the raw input data includes accelerations in three-axis directions from the accelerometer module of the smartphone and the angular acceleration in three directions from the three-axis gyroscope module, speed from the GPS module, and traffic flow status from VMS. The framework outputs the traffic flow state label of the test data. Traffic flow state includes free-flow state, steady flow state, and congested flow state. Framework 1 consists of five parts: smartphone sensor data preprocessing, model

Table 2
The selected statistical features and thresholds.

Variable Number	Statistical Feature	Variable	Threshold	Unit
1	Standard Deviation	AX	0.41	g
2	Mean	V	0.9	m/s
3	Quartile	AX	0.41	g
4	Quartile	AZ	0.51	g
5	Variance	AZ	0.2	g^2
6	Average Absolute Deviation	AX	0.4	g
7	Average Absolute Deviation	GX	0.25	rad/s^2
8	Coefficient of Variation	AZ	0.5	/
9	Range	AX	0.2	g
10	Range	AZ	0.2	g
11	Standard Deviation	AX	0.3	g
12	Standard Deviation	AZ	0.2	g
13	Standard Deviation	GX	0.1	rad/s^2
14	Standard Deviation	GY	0.15	rad/s^2
15	Mean	V	0.4	m/s
16	Quartile	AX	0.18	g
17	Quartile	AZ	0.26	g
18	Variance	AX	0.08	g^2
19	Variance	AZ	0.06	g^2
20	Average Absolute Deviation	AX	0.26	g
21	Average Absolute Deviation	AZ	0.25	g
22	Average Absolute Deviation	GX	0.2	rad/s^2
23	Coefficient of Variation	AZ	0.22	/

selection, model construction, model training, and traffic flow status classification based on test data.

Framework 1. traffic flow state classification framework based on smart phone sensor data

Input: the originally received smart phone sensor data.
Output: classification results of test data.
Training data and test data grouping and data preprocessing.
Choice of a classification model.
Classification model construction.
if the selected algorithm is a two-class classification algorithm **then**
  Convert the selected algorithm into a multi-class classification algorithm according to the one-versus-one algorithm.
end if
for each training cycle **do**
  Training of classification models based on training data set.
  if the parameter adjustment mechanism exists **then**
    Parameter optimization.
  end if
end for
Classification of traffic flow states based on test data.

The framework proposes a two-step feature calculation process for preprocessing analysis of training data and test data separately. A two-step feature calculation process is followed to extract the hidden information behind the data of each variable. The first step is to calculate new statistical features such as range, standard deviation, mean, quartile, variance, average absolute deviation, skewness, kurtosis, and coefficient of variation based on a moving window with width n_1. The window moves forward at a step size of m_1 and repeats until it reaches the end of the data; In the second step, to further extract the features in different states, we normalize the data and calculate the amount of these new statistical features that are greater than a certain threshold based on the results from the first step. The threshold is predetermined in Table 2. The computational procedure of step two is similar to step one, with window width n_2 and step size m_2. Here, the default values for n_1, m_1, n_2, m_2 are 100, 2, 1000, and 1.

As shown in Fig. 2, after the two-step feature calculation and data normalization, the new variables in different states show significantly different statistical characteristics. For example, the mean and the range of the variables gradually decrease from free-flow state, steady flow state, to congested flow state. The label on the x-axis of Fig. 2 corresponds to the variable number of Table 2. In our proposed framework, the size of the input variables is 23, as shown in Table 1.

5. Utilizing DBN model to solve the problem

5.1. The DBN model

DBN (Hinton et al., 2006) is a class of deep neural networks, composed of the stacking of numerous Restricted Boltzmann Machines (RBM) (Hinton and Sejnowski, 1986). RBM makes a joint distribution between data and labels based on the probability generating model, and it updates network parameters through the Contrastive Divergence (CD) algorithm. The learning of DBN parameters adopts the layer-by-layer training mode, involving unsupervised training and supervised fine-tuning. To realize the unsupervised learning of

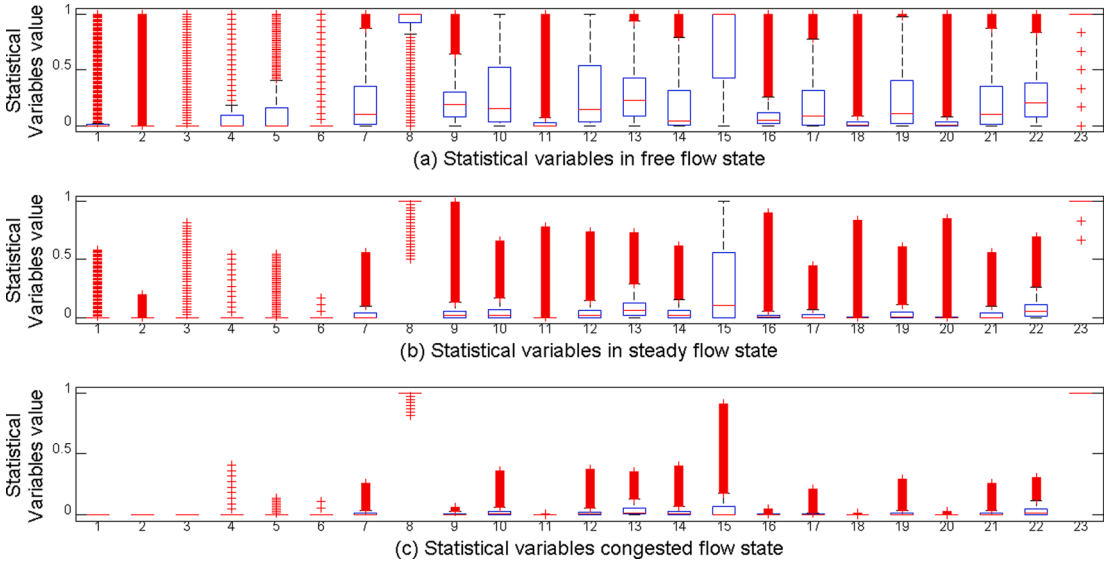

Fig. 2. Boxplot of statistical variables in different traffic flow states.

network parameters, DBN continuously stacks RBM and transmits information forward. During the supervised fine-tuning phase, DBN updates network parameters according to label information and the BP algorithm. Due to the relatively strong capability of extracting features from a large number of samples, DBN can handle classification tasks well (Lee et al., 2009a; Mohamed et al., 2012).

Especially, DBN has been shown the potential of learning generative and discriminative models on high-dimensional data. It is used in image classification (Tang et al., 2016), facial expression recognition (Bouchra et al., 2019; Liu et al., 2014), human emotion recognition (Hassan et al., 2019), disease detection (Abdel-Zaher and Eldeib, 2016), and mechanical failure diagnosis (Chen and Li, 2017; Shen et al., 2019; Wang et al., 2020). DBN is a probabilistic generative model with multi RBM layers. The learning procedure CD can effectively obtain the joint distribution of visible and hidden units, particularly in the high-dimensional feature space.

In neural network learning, the initial value of the weight will affect the learning process. The method of random initialization of parameters may cause slow convergence speed and fall into the local optimum. Besides, the back-propagation algorithm is not extensible. As the network becomes large, the performance may not be improved, accompanied by time issue problems (Dedinec et al., 2016). In the DBN model, the forward stacked RBMs learn in an unsupervised weight learning manner. It is equivalent to providing prior knowledge for DBN and the initialization of supervised learning parameters. It intends that each layer captures the previous layer's features, tracing from the lowest layer features (Hadsell et al., 2008). The known labels can be used to optimize the pre-trained DBN model. The model parameters of all layers are fine-tuned further, starting from the last layer. In the fine-tuning stage, the entire weight and bias are optimized by the back-propagation algorithm. DBN extracts features of the input data to form ideal features suitable for pattern classification. Compared with the random initialization method, the DBN makes the network's initial parameters closer to the optimal solution (Chen and Li, 2017). Contrasted with shallow learning models, the DBN can learn complex functions and represent more abstract features effectively and compactly.

The advantages that the DBN model has to deal with high dimensional data and prevent local optimums mentioned above, motivate us to utilize DBN to investigate the capability of the proposed framework. The illustration of the DBN model is shown in Fig. 3. The first layer at the bottom is the visible layer that receives the original feature data and acts as the input layer. This input layer is followed by three hidden layers. Input layer and the first hidden layer form the first Restricted Boltzmann Machines (RBM), and the first hidden layer and the second hidden layer form the second RBM, etc. The structure of each RBM is of full two-way connectivity between two layers. The last layer is the output layer associated with the classification results.

Like most neural network models, parameter selection of the DBN model could be challenging. When the hyper-parameters are not properly chosen, the model could converge to the locally optimal solution. Therefore, we use an optimization algorithm to determine hyper-parameters so that to improve classification accuracy. In this paper, the hyper-parameters of the DBN model are optimized by the hybrid optimization algorithm (DEGWO) based on the Differential Evolution (DE) algorithm and the Grey Wolf Optimizer (GWO) algorithm.

5.2. *Tuning the DBN model with hybrid optimization algorithm (DEGWO)*

For optimization algorithms, Mirjalili et al. (2014) proposed GWO as a new swarm intelligence heuristic algorithm that simulates the hunting of grey wolves in nature. To improve the optimization efficiency of the GWO algorithm and the ability to avoid being trapped into local optimum, El-Fergany and Hasanien (2015) proposed the DEGWO algorithm and developed the GWO algorithm by

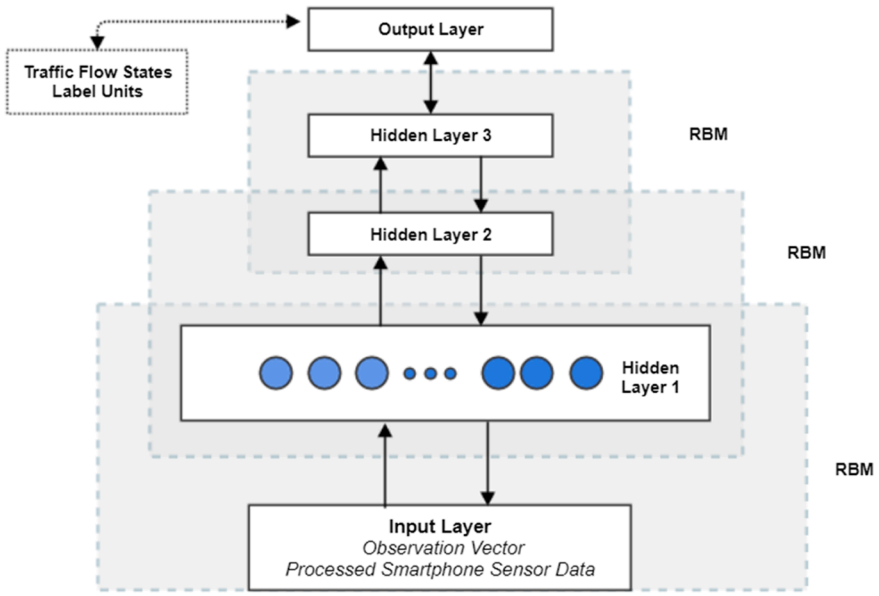

Fig. 3. The structure of the DBN traffic flow state classification model.

using the DE algorithm. DE (Storn and Price, 1997) is a stochastic model that simulates the evolution of living things. Individuals who adapt to the environment are preserved through repeated iterations. DE uses real number coding, differential-based simple mutation operation, and a one-to-one competitive survival strategy. Hence, DE can reduce the complexity of the genetic algorithm (Zhu et al., 2015). In this paper, we use DEGWO's global searching capabilities to optimize the hyper-parameters of the DBN model.

The GWO algorithm simulates the hierarchical system and hunting behavior of grey wolves in nature. The wolves are divided into four groups: α, β, δ, and ω. α, β, and δ are in turn the three individuals with the best fitness. They instruct other wolves (ω) to search toward the target.

We assume that k is the current iteration number; k_{max} is the maximum number of iterations; $\boldsymbol{A}_1$, $\boldsymbol{A}_2$, and $\boldsymbol{A}_3$ represent the adaptive parameter vectors of wolf α, wolf β, and wolf δ, respectively. $\boldsymbol{C}_1$, $\boldsymbol{C}_2$, and $\boldsymbol{C}_3$ represent the random parameter vectors of wolf α, wolf β, and wolf δ, respectively. The above parameter vectors obey the relationship: $\boldsymbol{A} = 2a\boldsymbol{r}_2 - a$, $\boldsymbol{C} = 2\boldsymbol{r}_1$,$a = 2 - 2(k/k_{max})$, where a decreases linearly from 2 to 0 with the number of iterations, and $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are random vectors in $[0,1]$. When $|\boldsymbol{A}| > 1$, the grey wolves spread as far as possible in each area and search for prey; When $|\boldsymbol{A}| \leqslant 1$, the grey wolves will focus on hunting for prey in one or more areas. $\boldsymbol{C}$ is a vector composed of random values in the interval $[0,2]$. The coefficient provides random weights for $\boldsymbol{X}(k)$ to increase ($|\boldsymbol{C}| > 1$) or decrease ($|\boldsymbol{C}| < 1$). $\boldsymbol{X}_\alpha(k)$, $\boldsymbol{X}_\beta(k)$, and $\boldsymbol{X}_\delta(k)$ represent the positions of wolf α, wolf β, and wolf δ, respectively. N is the population size of the grey wolf. As shown in Eqs. (1)–(3), $\mathbf{D}_{\boldsymbol{\alpha}}$, $\boldsymbol{D}_\beta$,and $\boldsymbol{D}_\delta$ denote the distances between wolf ω and the top three grey wolf individuals (Mirjalili et al., 2014).

$$\boldsymbol{D}_\alpha = |\boldsymbol{C}_1\boldsymbol{X}_\alpha(k) - \boldsymbol{X}(k)| \tag{1}$$

$$\boldsymbol{D}_\beta = |\boldsymbol{C}_2\boldsymbol{X}_\beta(k) - \boldsymbol{X}(k)| \tag{2}$$

$$\boldsymbol{D}_\delta = |\boldsymbol{C}_3\boldsymbol{X}_\delta(k) - \boldsymbol{X}(k)| \tag{3}$$

During the optimization process, $\boldsymbol{X}(k+1)$ are the updated positions of ω with the guidance of α, β, and δ, as shown in Eq. (4).

$$\boldsymbol{X}(k+1) = \frac{(\boldsymbol{X}_\alpha - \boldsymbol{A}_1 \cdot \boldsymbol{D}_\alpha) + (\boldsymbol{X}_\beta - \boldsymbol{A}_2 \cdot \boldsymbol{D}_\beta) + (\boldsymbol{X}_\delta - \boldsymbol{A}_3 \cdot \boldsymbol{D}_\delta)}{3} \tag{4}$$

When the maximum iteration number k_{max} is reached, the position of the wolf α, $\boldsymbol{X}_\alpha$, is the optimal solution. The detailed calculation procedure is shown by Algorithm 1.

Algorithm 1. GWO

Input: k_{max};N;
Output: The best solution $\boldsymbol{X}_{\boldsymbol{\alpha}}$
  Initialize the grey wolf population $\boldsymbol{X} = [\boldsymbol{X}_1, \boldsymbol{X}_2, ..., \boldsymbol{X}_N]$.
  Initialize a, $\boldsymbol{A}$, and $\boldsymbol{C}$.
  Calculate the fitness value of all individual.
  Calculate top three grey wolf individuals $\{\boldsymbol{X}_\alpha, \boldsymbol{X}_\beta, \boldsymbol{X}_\delta\}$.
  while $(k < k_{max})$
    for each individual wolf i **do**
      Calculate coefficient vectors $\boldsymbol{A}$, and $\boldsymbol{C}$.
      Calculate the distance variables $\{\boldsymbol{D}_\alpha, \boldsymbol{D}_\beta, \boldsymbol{D}_\delta\}$.
      Update the position of the ω, $\boldsymbol{X}(k+1)$.
    end for
    Update a, $\boldsymbol{A}$, and $\boldsymbol{C}$.
    Calculate the fitness value of all individual.
    Update $\{\boldsymbol{X}_\alpha, \boldsymbol{X}_\beta, \boldsymbol{X}_\delta\}$.
    $k \leftarrow k + 1$.
  end while
  return $\boldsymbol{X}_\alpha$

DE is an algorithm based on population evolution. DE adopts mutation operation, cross operation, and selection operation to approach the globally optimal solution. DE uses the differential mutation operation to generate new variant individuals. Cross individuals are obtained by crossing the mutated individual and the original individual. Finally, the greedy selection is operated to decide the next generation.

In the calculation procedure of the DBN_DEGWO model, we define $\boldsymbol{D}_{\text{train}} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), ..., (\boldsymbol{x}_{m'}, y_{m'})\}$ as the training set, and $\boldsymbol{D}_{\text{test}} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), ..., (\boldsymbol{x}_n, y_{n'})\}$ as the test set. $\boldsymbol{S} = (s_{\text{size}}, s_{\text{ban}}, r_{\text{learning}}, n_{\text{neuron.1}}, n_{\text{neuron.2}}, n_{\text{neuron.3}}, n_{\text{train}}, n_{\text{test}})$ is the parameter vector of the DBN model. s_{size} is the number of categories, s_{ban} is batch size, r_{learning} is the learning rate, and n_{train} is the number of unsupervised training iterations. r_{test} and r_{train} denote the error rates in test and learning procedure respectively. We assume that $z_{i,t}^j$ denotes the ith individual in the tth generation of the population, $\boldsymbol{\tau}_{i,t}^j$ denotes the mutation individuals, $\boldsymbol{\mu}_{i,t}^j$ denotes the crossover individuals, and F is an amplifying factor within $[0,1]$. The core operations of the DE algorithm mainly include mutation operation, cross operation, and selection operation. The DEGWO algorithm utilizes differentially evolved variant individuals to maintain population diversity and then uses the GWO algorithm and the DE algorithm to select individuals for global search. The specific procedures are shown in Al-

gorithm 2.

Algorithm 2. DBN_DEGWO

Input: $\boldsymbol{D}_{\text{train}}$; $\boldsymbol{D}_{\text{test}}$; $\boldsymbol{S}$; M; T; D;ε.
Output: r_{test},z_{best}

```
Initialize t, a, A,and C
for each individual i = 1 to M do
  for each dimension j = 1 to D do
    z_{i,t}^j ← z_{min}^j + rand(0,1)(z_{max}^j − z_{min}^j)
  end for
end for
Calculate the fitness value of all individual f(z_{i,t})
for each individual wolf k do
  f(z_{i,t}) ← DBN(D_train, D_test, z_{i,t})
end for
Sort f(z_{i,t}) and compute top three wolf individuals {z_α, z_β, z_δ}
while t⩽T do
  for each individual i = 1 to M do
    Calculate coefficient vectors A,and C
    Calculate the distance variables {D_α, D_β, D_δ}
    Update the position of the ω, z(t + 1)
  end for
  Update a, A,and C
  Update f(z_{i,t}) ← DBN(D_train, D_test, z_{i,t}) and {z_α, z_β, z_δ}
  for each individual i = 1 to M do
    for each dimension j = 1 to D do
      τ_{i,t}^j ← z_{i_1,t}^j + F · (z_{i_2,t}^j − z_{i_3,t}^j)
      μ_{i,t}^j = { τ_{i,t}^j,  rand(0,1)⩽ε
                  { z_{i,t}^j,  else
    end for
    Update f(μ_{i,t}) ← DBN(D_train, D_test, μ_{i,t})
    Update f(z_{i,t}) ← DBN(D_train, D_test, z_{i,t})
    z_{i,t} = { μ_{i,t},  f(μ_{i,t}) < f(z_{i,t})
              { z_{i,t},  f(μ_{i,t})⩾f(z_{i,t})
    if f(z_{i,t}) < f(z_best) then z_best ← z_{i,t}
    end if
  end for
 end while
Update S ← z_best
DBN (D_train,D_test,S)
 returnr_test,z_best
```

6. Experiments

6.1. Experimental setup

In the experiments, the computer configuration is as follows. The computer model is DELL ALIENWARE 15 R3, and the processor is Intel(R) Core (TM) i7-7700HQ CPU @ 2.80 GHz 2.80 GHz. The capacity of random-access memory is 8 GB with DDR4 type and 2400 MHz. The computer is equipped with a 1 TB memory hard drive with 7200 rpm, SATA interface, and 6 GB/s interface speed. The calculation processing software is MATLABR2016a.

For comparison, we utilize a variety of widely used linear and nonlinear classification models, which include DBN, RF, Ensemble method with adaptive boosting and Decision Tree, SVM, KNN, Discriminant analysis classifier (DA), Naive Bayes classifier (NB), Least Absolute Shrinkage, Selection Operator (Lasso), and polynomial logistic regression (PLR).

We build two DBN models with different parameter settings, namely the DBNa model and the DBNb model. The number of neurons in the three-layer unsupervised RBM layer in the DBN model is 300, and the activation function is the sigmoid function. In the unsupervised RBM layer, the learning rate is 1, the scale factor of the learning rate of each iteration is 0.5, and the final learning rate is the product of the learning rate and the scale factor of the learning rate. The momentum parameter is 0. The number of training iterations for the unsupervised layer is 30. The output layer is the NN layer, and its number of neurons is 3, which corresponds to the three states of traffic flow. The activation function of the output layer is the SoftMax function. In the output layer, the momentum parameter is 0.9, and the learning rate is 2. The scale factor of the learning rate of each iteration is 0.5. In addition to the general settings of the above parameters, the training iteration times of the output layer of the DBNa model and the DBNb model are 20 and 200, respectively.

To further improve the classification accuracy, we apply the DEGWO model to optimize the hyper-parameters in DBN. Here, the selected hyper-parameters are the number of neurons in each RBM network, the number of unsupervised training iterations, the learning rate of unsupervised training, and the number of supervised training iterations. In the experiment of DBN_DEGWO, the

selected ranges for the number of neurons in each RBM network ($n_{\text{neuron.1}}$, $n_{\text{neuron.2}}$, $n_{\text{neuron.3}}$) is 100 to 300, the unsupervised training iteration (n_{train}) is 100 to 200, and the learning rate of unsupervised training (r_{learning}) is 0.5 to 2, the scale factor of the learning rate of each iteration is 0.5. We set different values of 50, 100, 200, 300, 500, 1000, and 1500 as the upper-bound of supervised iterations (n_{test}) for different scenarios. In the unsupervised RBM layer, the activation function is the sigmoid function; the momentum parameter is 0. In the output NN layer, the number of neurons is 3; the activation function is the SoftMax function; the momentum parameter is 0.9. The scale factor of the learning rate of each iteration is 0.5. For comparison, we also provide the results of a non-optimized DBN model. For comparison, the same training and test data are used. We set the values of the hyper-parameters of the non-optimized DBN model to be the default value: the number of neurons per layer of RBM network is 300, the number of unsupervised training steps is 200, and the learning rate of unsupervised training is 1, the scale factor of the learning rate of each iteration is 0.5.

The models RFa, RFb, and RFc are constructed based on random forest algorithm. In the random forest model, the sample set is sampled with replacement to build the tree, and the out-of-bag samples are used to evaluate the quality of the model. The maximum depth of the decision tree is not limited. The minimum sample contained in the leaf node is 1. The minimum sample that can be divided is 2. There is no limit to the maximum number of leaf nodes and to the division criteria of the nodes. The impurity standard refers to the Gini index, and the information gain standard relates to entropy. In addition to the above general parameter settings, the numbers of decision trees for RFa, RFb, and RFc are 50, 200, and 500, respectively.

ABTa, ABTb, and ABTc models are constructed based on the combination of Adaboost.M2 and decision tree. The M2 learning rate is 1. We train a decision tree ensemble using AdaBoost with 100 learning cycles. The numbers of decision trees for ABTa, ABTb, and ABTc models are 50, 200, and 500, respectively.

For the SVM model applied to the proposed framework, the kernel function is Gaussian Radial Basis Function with scaling factor 1. Sequential Minimal Optimization (SMO) method implements the L1 soft-margin SVM classifier. Degree in kernel function is 3, the value of gamma in kernel function is 1/3, parameter C is 1 in the loss function.

The KNNa, KNNb, and KNNc models are constructed based on the KNN algorithm. The distance calculation method is the euclidean method. The numbers of nearest neighbors of the KNNa, KNNb, and KNNc models are 5, 6, and 7, respectively.

In the LDA model, we choose the linear classification and fit a multivariate normal density to each group with a pooled estimate of covariance. In NB, we utilize the congested flow as a reference class to classify the test data. Lasso model finds the coefficients of a regularized linear regression model using 10-fold cross-validation. The value of Alpha for the elastic net is 0.75. We use the largest Lambda value such that the mean squared error is within one standard error of the minimum MSE. PLR substitutes the test data into the classification probability function and selects the category with the highest probability as the test data classification result.

6.2. Analysis of classification results

To demonstrate the performance of the proposed framework, we conducted ten independent experiments for each algorithm. To ensure the fairness of the algorithm comparison, the test data and training data selected for each algorithm are the same. The Box-plot of the accuracy of the ten groups of tests for each algorithm is shown in Fig. 4.

To demonstrate the benefit of acceleration and angular acceleration data in traffic flow classification, we also carry out a scenario using only the characteristics of speed data. The classification accuracies of various algorithms are shown in Fig. 5.

Fig. 4(a) shows that most nonlinear classification models, such as DBNa, DBNb, RFa, RFb, RFc, SVM, KNNa, KNNb, and KNNc, achieve high classification accuracy. The highest average accuracy, 0.9607, is obtained by the DBNb model, and SVM has the lowest average accuracy of 0.8422. In Table 2, the DBN models also show better results compared with all the other models, which demonstrates the advantage of DBN in our task. As shown in Fig. 4(b), all linear classification models have relatively low accuracy. The

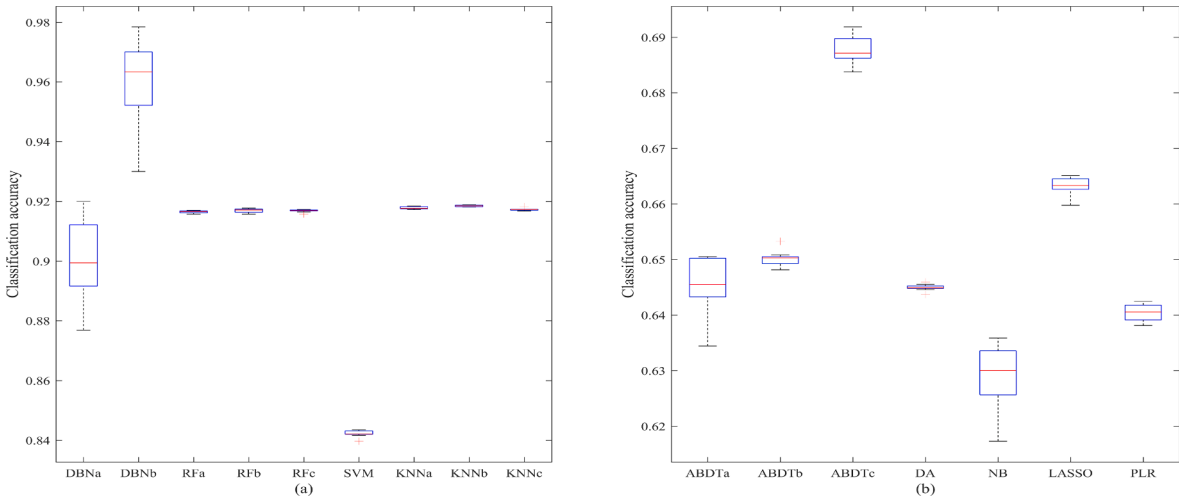

Fig. 4. Classification accuracies of DBN, RF, SVM, KNN, ABDT, DA, NB, LASSO and PLR.

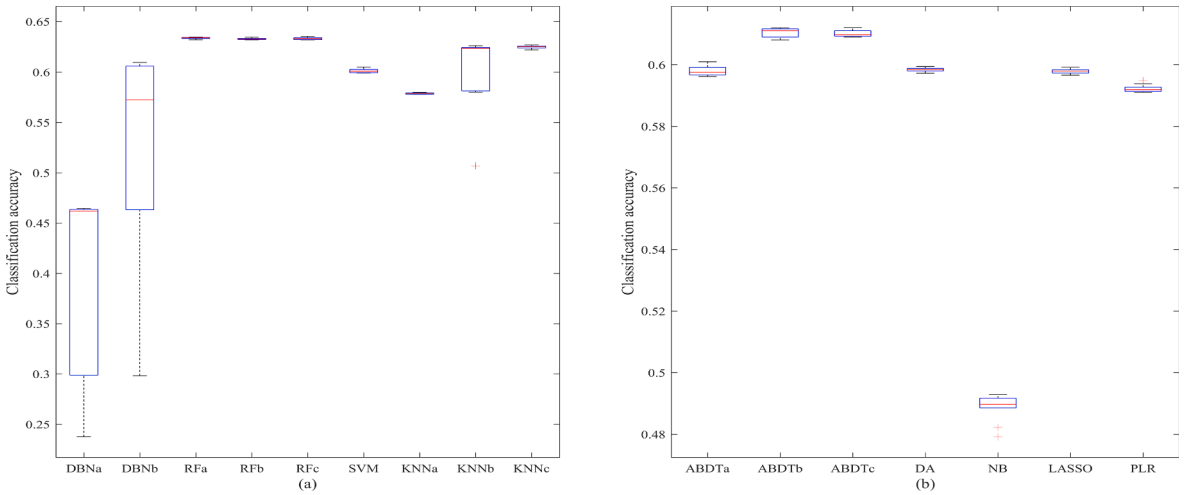

Fig. 5. Classification accuracies of DBN, RF, SVM, KNN, ABDT, DA, NB, LASSO and PLR with speed data only.

convergence of the DBN model is shown in Fig. 6.

As shown in Fig. 5 and Table 3, when only speed is used as the input variable in ten test groups, the highest average accuracy is 0.6336, which is significantly lower than the highest average accuracy (0.9607) obtained by including the acceleration and angular acceleration data. Furthermore, the average percentage errors for each state in the two cases are calculated in Table 4.

As shown in Table 4, in the speed-only case, the percentage error for the steady flow state is higher than those for the free flow and congested flow states. The reason is that the built-in sensors of smartphones, such as accelerometers and gyroscope, allow the perceptions of the motions of the car in more dimensions, which indirectly reflect the traffic environment around the vehicle. Thus, at the steady flow state, when the behavior of the vehicle is more diversified, acceleration and the angular acceleration data can help the most to improve the accuracy.

The classification error of the non-optimized DBN model and DBN_DEGWO model are shown in Table 5. When we increase the upper-bound of the number of supervised iterations from 50 to 1500, the optimized classification error of DBN_DEGWO is reduced from 3.92% to 0.76%. When the number of supervised iterations is set to 50, 100, 200, 300, 500, 1000, and 1500, the classification errors of the non-optimized DBN model are 5.01%, 4.81%, 4.77%, 4.76%, 3.91%, 2.28%, and 2.27%, respectively, which are both higher than the classification error value of the DBN_DEGWO of the corresponding groups. The DEGWO algorithm optimizes the hyperparameters of the DBN model, finds a better parameter collocation scheme, and reduces the classification error.

6.3. *Measuring the significance of the input variables*

In this section, we measure the significance of the input variables of the model through the permuted Out-of-Bag analysis (Bylander, 2002; Matthew, 2011) and Random Forest algorithm with 500 learners. For each tree t_{ob}, $t_{ob} \in \{1, \ldots, 500\}$, we first compute the Out-of-Bag error $\varepsilon_{t_{ob}}^{ob}$ of all input variables. Then for each variable to be estimated $x_{ob,\tau}$, $\tau \in \{1, \ldots, 23\}$, we use the Out-of-Bag observations containing the randomly permuted values of $x_{ob,\tau}$ to calculate the model error $\varepsilon_{t_{ob},\tau}^{ob}$ and the difference $d_{t_{ob},\tau}^{ob} = \varepsilon_{t_{ob},\tau}^{ob} - \varepsilon_{t_{ob}}^{ob}$. Finally, we

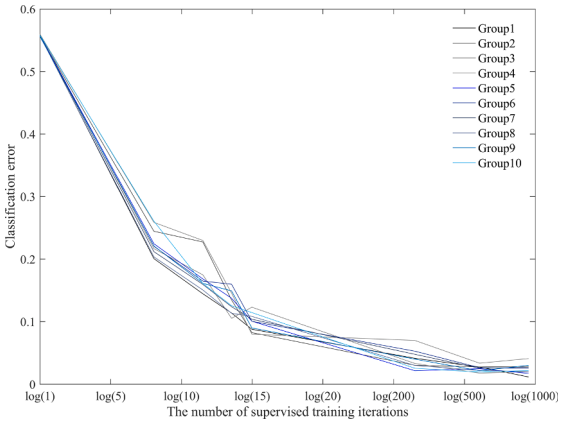

Fig. 6. Classification error of DBN model with ten test groups.

Table 3
Average classification accuracy of DBN, RF, SVM, KNN, ABDT, DA, NB, LASSO and PLR.

Classification model	Average accuracy with all data	Average accuracy calculation with only speed data	Percentage increased
DBNa	0.9009	0.3915	130.14%
DBNb	**0.9607**	**0.5179**	**85.49%**
RFa	0.9165	0.6336	44.64%
RFb	0.9169	0.6328	44.90%
RFc	0.9168	0.6333	44.78%
ABDTa	0.6456	0.5981	7.94%
ABDTb	0.6502	0.6104	6.52%
ABDTc	0.6878	0.6102	12.72%
SVM	0.8422	0.6011	40.10%
KNNa	0.9178	0.5788	58.58%
KNNb	0.9185	0.6038	52.12%
KNNc	0.9174	0.6248	46.83%
DA	0.6450	0.5984	7.78%
NB	0.6292	0.4885	28.79%
LASSO	0.6631	0.5979	10.91%
PLR	0.6404	0.5923	8.13%

Table 4
The average percentage error.

Classification model	Training with acceleration, angular acceleration, and speed data			Training with speed data only		
	Free flow	Steady flow	Congested flow	Free flow	Steady flow	Congested flow
DBNa	0.0893	0.1054	0.0716	0.1458	0.8620	0.5344
DBNb	0.0376	0.0402	0.0408	0.2166	0.4990	0.2927
RFa	0.0737	0.1518	0.0169	0.2063	0.7631	0.1823
RFb	0.0741	0.1510	0.0169	0.2070	0.7634	0.1811
RFc	0.0747	0.1504	0.0170	0.2093	0.7587	0.1817
KNNa	0.0718	0.1453	0.0228	0.4792	0.5761	0.3048
KNNb	0.0722	0.1446	0.0229	0.4417	0.6751	0.1663
KNNc	0.0720	0.1449	0.0235	0.4408	0.5741	0.3205
SVM	0.2034	0.5481	0.1963	0.2352	0.7936	0.2237
ABDTa	0.2135	0.6724	0.2186	0.2502	0.8359	0.1534
ABDTb	0.2137	0.6410	0.2485	0.2546	0.7123	0.2505
ABDTc	0.2137	0.6403	0.2484	0.2546	0.7128	0.2497

Table 5
The classification error of non-optimized DBN and DBN_DEGWO.

Upper-bound of n_{test}	$n_{\text{neuron.1}}$	$n_{\text{neuron.2}}$	$n_{\text{neuron.3}}$	n_{train}	r_{learning}	n_{test}	Classification error of DBN_DEGWO	Classification error of Non-optimized DBN
50	260	282	275	151	2	47	3.92%	5.01%
100	290	256	270	200	2	96	2.84%	4.81%
200	275	275	278	176	2	184	2.11%	4.77%
300	280	259	252	189	2	280	1.61%	4.76%
500	300	281	261	200	2	500	0.96%	3.91%
1000	266	300	286	188	2	999	0.78%	2.28%
1500	288	250	281	193	2	1499	0.76%	2.27%

calculate the standard deviation σ_τ of $d^{ob}_{t_{ob},\tau}$ over the learners t_{ob}, $\tau \in \{1, \ldots, 23\}$. A higher standard deviation indicates a higher significance of the input variable. The cumulative standard deviation errors of each input variable of ten test groups are shown in Fig. 7.

We can observe that the top five significant variables are the amount of GY Standard Deviations exceeding 0.15 (index 14), the amount of Average Absolute Deviations of GX exceeding 0.2 (index 22), the amount of AX Quartiles exceeding 0.18 (index 16), the amount of GX Standard Deviations exceeding 0.1 (index 13), and the amount of AZ range exceeding 0.2 (index 10).

Given the above significant variables, we analyze how acceleration and angular acceleration can help improve the accuracy of traffic flow state classification. AX and AZ represent the accelerations in the two directions perpendicular to the vehicle driving direction. Usually, in a congested situation with low speed, the values of AX are low because of the limited space around the vehicle. As a result, the AX quartile has a lower value in the congested state; hence, it can be useful for classifying congested flow and steady flow states. When the vehicle is operating at high speed, the range of AZ value is relatively high because of the vibration of the vehicle; thus, AZ is a crucial variable to distinguish between free flow and steady flow.

Compared to acceleration and speed, angular acceleration contains more detailed moving information. As shown in Fig. 1 and Fig. 2, when the vehicle is in a steady flow state or a congested flow state, the speed distributions are similar. But the angular

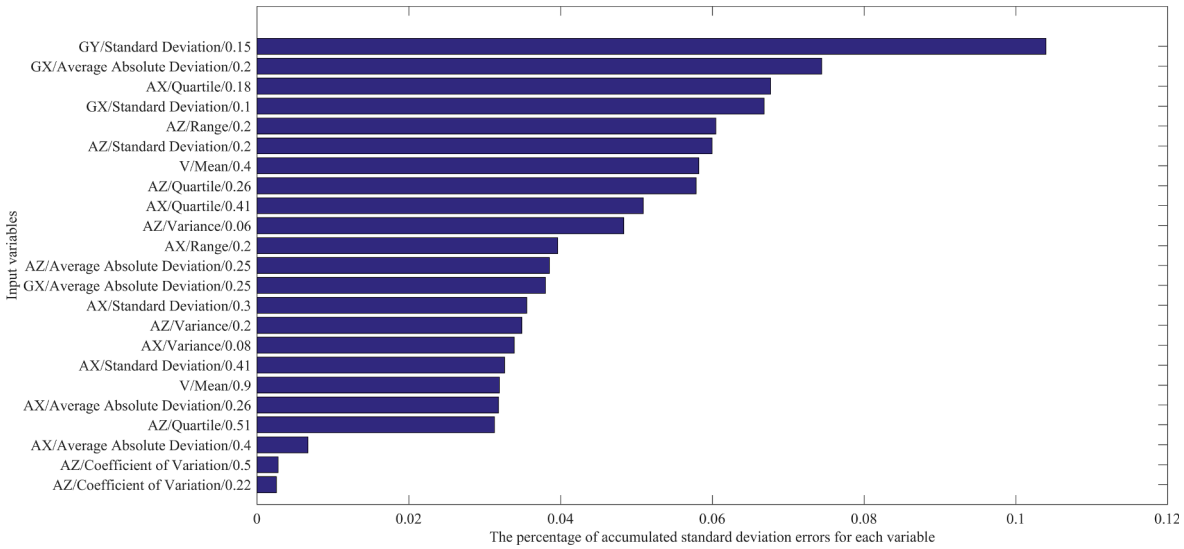

Fig. 7. The estimation of input variables importance.

acceleration has different variances and amplitude range changes. Furthermore, as shown in Fig. 1, acceleration may not change significantly in the free-flow state or steady flow state. However, the driving stability is different for the two states. The angular acceleration will have more valuable information in terms of its variance distribution as a result. For a similar reason, the amount of GY Standard Deviations exceeding 0.15 (index 14) and the amount of GX Standard Deviations larger than 0.1(index 13) can help indicate the difference between congested flow and steady flow. Moreover, the amount of Average Absolute Deviations of GX exceeding 0.2 (index 22) can help distinguish between free flow and steady flow.

All the top five significant variables are related to accelerations and angular accelerations, indicating that they are playing an important role in the traffic flow state classification.

6.4. *Sensitivity analysis*

In this section, computations are made to analyze the effect of the parameters of the feature calculation process on traffic flow state classification accuracy. Fig. 8 shows four groups of experimental results related to the parameters of the feature calculation process.

The parameter settings for each experimental group are as follows: in group (a), $n_1 = 100$, $m_2 = 1$, $n_2 = 1000$; in group (b), $m_1 = 2$, $m_2 = 1$, $n_2 = 1000$; in group (c), $m_1 = 2$, $n_1 = 100$, $n_2 = 1000$; in group (d), $m_1 = 2$, $n_1 = 100$, $m_2 = 1$. As shown in Fig. 8(a) and (b), in the first step feature calculation process, when the window width n_1 and the step size m_1 range from 2 to 200 and 10 to 1000 respectively, the classification accuracy of most groups remains above 88%. However, as shown in Fig. 8(c) and (d), in the second step feature calculation process, when the step size m_2 increases to 200, or the window width n_2 reduces to 10, the classification accuracy becomes less than 60%. The reason is that after the first step feature calculation process, data volume in the second step feature calculation process becomes smaller, and the data contains more feature information. In the second step feature calculation, when the number of step size m_2 is large, or the window width n_2 is small, it may lead to loss of some useful information and lower classification accuracy.

7. Discussion

In this paper, we propose a methodology to utilize smartphone sensor data in only one vehicle in the traffic flow to classify the traffic flow state. We prove that the classification accuracy can be enhanced by using acceleration and angular acceleration data in addition to speed data of the vehicle. The intrinsic reasons why acceleration and angular acceleration are crucial for the task of traffic flow state estimation are conjectured as below:

i) *The interactions between vehicles can be reflected by acceleration data*

In the free-flow state, drivers are rarely affected by other drivers on the road. When the leading vehicle is far away, drivers can reach their desired speed and freely change lanes at high speeds. When drivers keep a certain distance from the vehicle in front and operate forward at a relatively constant speed, drivers tend to maintain vehicle speed and unconsciously minimize the numbers of accelerations or decelerations. As the instant lane-occupancy increases, the interactions between vehicles gradually increase. Unlike the free-flow state, drivers create more decelerations and accelerations because of the car-following behavior in the steady and congested flow states.

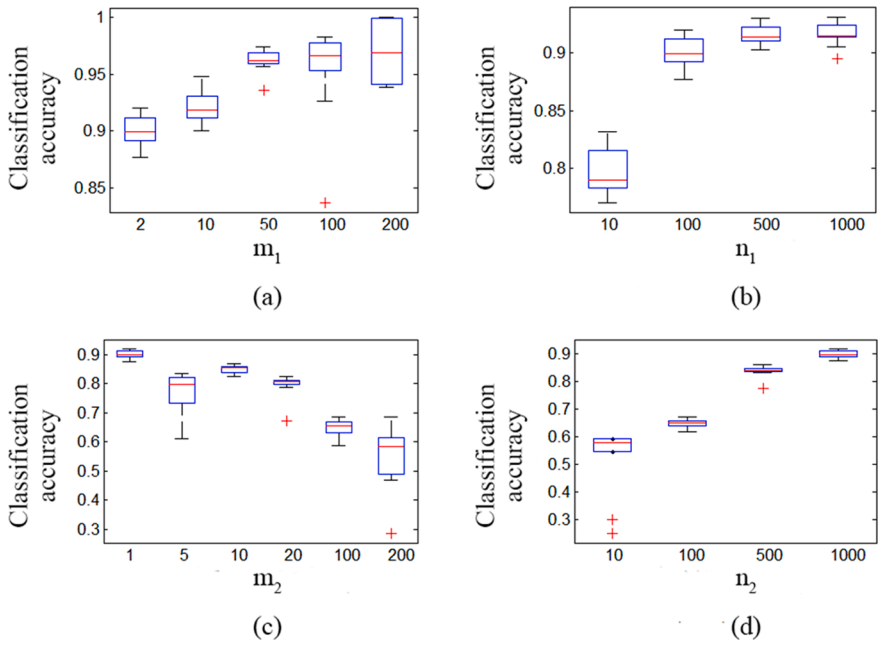

Fig. 8. Classification accuracy with different values of parameters m_1,n_1,m_2 and n_2.

ii) *The speed can be reflected by acceleration data when GPS signal is missing*

Accelerations at different directions can indicate how fast the vehicle moves. As the driving speed increases, the peak and variance of acceleration in the driving direction increase. Because of the uneven road surface, a faster speed also leads to vibrations with higher frequency when the vehicle is moving, which can be reflected by the vertical acceleration. Typically, the vertical acceleration rises when speed increases. The change of lateral acceleration reflects the driver's steering. When the vehicle speed is low, the large front wheel angle and angle change rate will increase the variance and peak of the lateral acceleration. Similarly, at high speed during lane switching or turning, the lateral acceleration will rise to a high peak in a short time.

iii) *The lane changing and turning behaviors can be reflected by angular acceleration data*

Angular acceleration shows the intensity of the driver's operations. The driver's behaviors of acceleration and deceleration, lane changing, and turning operation make the smartphone fixed in the vehicle generate additional angular acceleration fluctuations. For example, the angular acceleration in the driving and lateral directions reveal the operation of acceleration and deceleration, and lane changing and turning can be inferred by the angular acceleration in the driving and vertical directions. As a result, the angular acceleration data can thus indicate the states of the traffic flow, since as the road congestion increases, the lane changing and turning behavior decreases.

iv) *Vehicle's engine status can be reflected by angular acceleration data*

Like speed and acceleration, angular acceleration can also reveal the vehicle's engine status. Angular acceleration is more sensitive to weak abnormal vibrations. The change in angular speed per unit of time can be captured by angular acceleration, even with the small-amplitude vibration. As the vehicle speed decreases or the steering torque reduces, the smartphone torque rotation becomes less intense. The oscillation frequency and amplitude of the peak and variance will also be lower. Compared to the steady flow state, the angular accelerations in the congested flow state have apparently lower amplitudes and fewer high-frequency jitter peaks.

8. Concluding remarks

In this paper, we propose a general framework for traffic flow state classification based on smartphone sensor data. When nonlinear classification algorithms such as Deep Neural Network, random forest, SVM, and KNN algorithms are applied to the proposed framework, highly accurate classification results are obtained. We summarize the conclusions of the paper as below:

i) This framework only utilizes the smartphone sensor data of one car to complete the task of estimating the road traffic state. Thus, the proposed model can still work accurately in areas where there are no enough probe cars or lack of loop detectors or GPS signals.

ii) We show that by comparing with the results of using only the speed data, the participation of acceleration and angular acceleration data helps improve the classification accuracy. The reason is that, at the steady flow state, speed may contain both extremely high or low values, which lower the classification accuracy. Acceleration and angular acceleration information increase the accuracy by providing better perspectives of the surrounding of the probe car, not just the speed information itself.
iii) Based on the results of variable importance estimation, the characteristics of acceleration and angular acceleration play an important role in the task. In particular, GY Standard Deviations, GX Standard Deviations, and AX Quartiles are the three critical features that can help distinguish congested flow and steady flow. While Average Absolute Deviations of GX can help distinguish free flow and steady flow.
iv) 9 widely used algorithms are applied to the data from a five-day experiment, and the highest average classification accuracy, 96.07%, is obtained by the DBNb model. By introducing the DEGWO algorithm to optimize the hyper-parameters, we further reduced the classification error from 2.27% to 0.76% when the supervised iteration boundary is 1500.

This study demonstrates the potential of smartphone sensor data on investigating traffic flow states and driver behaviors. With the proposed method, traffic patterns and conditions can be better detected with less cost. Nevertheless, further research should be conducted to test the performance of the framework under a wider range of conditions with more massive datasets.

Acknowledgement

The work described in this paper was supported by the National Key Research and Development Program of China (2018YFB1600900), the National Science Fund for Distinguished Young Scholars (72025104), the National Natural Science Foundation of China (71861167001, 72001152), and the Department of Science and Technology of Sichuan Province, China (2018JY0254).

Author contribution

The authors confirm contribution to the paper as follows: study conception and design: Wenwen Tu, Feng Xiao; data preparation: Lu Li; analysis and interpretation of results: Wenwen Tu, Feng Xiao, Lu Li, draft manuscript preparation: Wenwen Tu, Feng Xiao, Liping Fu. All authors reviewed the results and approved the final version of the manuscript.

References

Abdel-Zaher, A.M., Eldeib, A.M., 2016. Breast cancer classification using deep belief networks. Expert. Syst. Appl. 46, 139–144. https://doi.org/10.1016/j.eswa.2015.10.015.
Antoniou, C., Koutsopoulos, H.N., Yannis, G., 2013. Dynamic data-driven local traffic state estimation and prediction. Transp. Res. Part C Emerg. Technol. 34 (9), 89–107. https://doi.org/10.1016/j.trc.2013.05.012.
Banaei-Kashani, F., Shahabi, C., Pan, B., 2011. Discovering patterns in traffic sensor data. In: Proceedings of the 2nd ACM SIGSPATIAL International Workshop on GeoStreaming. ACM, pp. 10–16. https://doi.org/10.1145/2064959.2064963.
Bao, L., Intille, S.S., 2004. Activity recognition from user-annotated acceleration data. In: International Conference on Pervasive Computing, Springer, pp. 1–17. https://doi.org/10.1007/978-3-540-24646-6_1.
Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H., 2006. Greedy layer-wise training of deep networks. In: Proceedings of the 19th International Conference on Neural Information Processing Systems (NIPS), MIT Press, pp. 153–160.
Bhoraskar, R., Vankadhara, N., Raman, B., Kulkarni, P., 2012. Wolverine: Traffic and road condition estimation using smartphone sensors. In: 2012 Fourth International Conference on Communication Systems and Networks (COMSNETS 2012), IEEE, pp. 1–6. https://doi.org/10.1109/COMSNETS.2012.6151382.
Bouchra, N., Aouatif, A., Mohammed, N., Nabil, H., 2019. Deep belief network and auto-encoder for face classification. Int. J. Interactive Multimedia Artificial Intelligence 5 (5), 22–29. https://doi.org/10.9781/ijimai.2018.06.004.
Bylander, T., 2002. Estimating generalization error on two-class datasets using out-of-bag estimates. Machine Learn. 48 (1–3), 287–297. https://doi.org/10.1023/A:1013964023376.
Chen, X., He, Z., Sun, L., 2019. A Bayesian tensor decomposition approach for spatiotemporal traffic data imputation. Transp. Res. Part C Emerg. Technol. 98, 73–84. https://doi.org/10.1016/j.trc.2018.11.003.
Chen, X.F., Liu, X., Han, F., Wang, D.L., Yin, J.J., 2014a. Models on real-time state identification for unban traffic based on fixed detector. Appl. Mech. Mater. 641–642, 818–823. https://doi.org/10.4028/www.scientific.net/AMM.641-642.818.
Chen, Y., Lin, Z., Zhao, X., Wang, G., Gu, Y., 2014b. Deep learning-based classification of hyperspectral data. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 7 (6), 2094–2107. https://doi.org/10.1109/jstars.2014.2329330.
Chen, Z., Li, W., 2017. Multisensor feature fusion for bearing fault diagnosis using sparse autoencoder and deep belief network. IEEE Trans. Instrum. Meas. 66 (7), 1693–1702. https://doi.org/10.1109/TIM.2017.2669947.
Chikaraishi, M., Garg, P., Varghese, V., Yoshizoe, K., Urata, J., Shiomi, Y., Watanabe, R., 2020. On the possibility of short-term traffic prediction during disaster with machine learning approaches: an exploratory analysis. Transp. Policy 98, 91–104. https://doi.org/10.1016/j.tranpol.2020.05.023.
Choudhury, R.R., Pal, J., Risbood, P., Matus, J., 2017. Method for accelerometer-assisted navigation. Google Patents. Patent No. 9733089.
Dedinec, A., Filiposka, S., Dedinec, A., Kocarev, L., 2016. Deep belief network based electricity load forecasting: an analysis of Macedonian case. Energy 115, 1688–1700. https://doi.org/10.1016/j.energy.2016.07.090.
Deng, W., Liu, H., Xu, J., Zhao, H., Song, Y., 2020a. An improved quantum-inspired differential evolution algorithm for deep belief network. IEEE Trans. Instrum. Meas. 69 (10), 7319–7327. https://doi.org/10.1109/TIM.2020.2983233.
Deng, Z., Huang, D., Liu, J., Mi, B., Liu, Y., 2020b. An assessment method for traffic state vulnerability based on a cloud model for urban road network traffic systems. IEEE Trans. Intell. Transp. Syst. 1–14. https://doi.org/10.1109/TITS.2020.3002455.
Dong, C., Shao, C., Xiong, Z., 2011. Identification of traffic states with optimized SVM method on urban expressway network. J. Beijing Jiaotong Univ. 35 (6), 13–16.
El-Fergany, A.A., Hasanien, H.M., 2015. Single and multi-objective optimal power flow using grey wolf optimizer and differential evolution algorithms. Electric Power Components Syst. 43 (13), 1548–1559. https://doi.org/10.1080/15325008.2015.1041625.
Farabet, C., Couprie, C., Najman, L., LeCun, Y., 2013. Learning hierarchical features for scene labeling. IEEE Trans. Pattern. Anal. Mach. Intell. 35 (8), 1915–1929. https://doi.org/10.1109/TPAMI.2012.231.
Glorot, X., Bordes, A., Bengio, Y., 2011. Domain adaptation for large-scale sentiment classification: a deep learning approach. In: Proceedings of the 28th International Conference on Machine Learning (ICML), Omnipress, pp. 513—520.

Gu, Y., Lu, W., Qin, L., Li, M., Shao, Z., 2019. Short-term prediction of lane-level traffic speeds: a fusion deep learning model. Transp. Res. Part C Emerg. Technol. 106, 1–16. https://doi.org/10.1016/j.trc.2019.07.003.
Guan, W., He, S., 2007. Phase identification of urban freeway traffic based on statistical properties. J. Transp. Syst. Eng. Inform. Technol. 7 (5), 42–50.
Habtie, A.B., Abraham, A., Midekso, D., 2016. A neural network model for road traffic flow estimation. In: Advances in Nature and Biologically Inspired Computing, Springer, pp. 305–314. https://doi.org/10.1007/978-3-319-27400-3_27.
Hadsell, R., Erkan, A., Sermanet, P., Scoffier, M., Muller, U., LeCun, Y., 2008. Deep belief net learning in a long-range vision system for autonomous off-road driving. In: 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, pp. 628–633. https://doi.org/10.1109/IROS.2008.4651217.
Händel, P., Ohlsson, J., Ohlsson, M., Skog, I., Nygren, E., 2014. Smartphone-based measurement systems for road vehicle traffic monitoring and usage-based insurance. IEEE Syst. J. 8 (4), 1238–1248. https://doi.org/10.1109/jsyst.2013.2292721.
Hassan, M.M., Alam, M.G.R., Uddin, M.Z., Huda, S., Almogren, A., Fortino, G., 2019. Human emotion recognition using deep belief network architecture. Inf. Fusion 51, 10–18. https://doi.org/10.1016/j.inffus.2018.10.009.
Herrera, J.C., Work, D.B., Herring, R., Ban, X.J., Jacobson, Q., Bayen, A.M., 2010. Evaluation of traffic data obtained via GPS-enabled mobile phones: The Mobile Century field experiment. Transp. Res. Part C Emerg. Technol. 18 (4), 568–583. https://doi.org/10.1016/j.trc.2009.10.006.
Hinton, G., Deng, L., Yu, D., Dahl, G., Mohamed, A.-R., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Kingsbury, B., 2012. Deep neural networks for acoustic modeling in speech recognition. IEEE Signal Process. Magazine 29 (6), 82–97. https://doi.org/10.1109/MSP.2012.2205597.
Hinton, G.E., Osindero, S., Teh, Y.-W., 2006. A fast learning algorithm for deep belief nets. Neural Comput. 18 (7), 1527–1554. https://doi.org/10.1162/neco.2006.18.7.1527.
Hinton, G.E., Sejnowski, T.J., 1986. Learning and Relearning in Boltzmann Machines, Parallel Distributed Processing: Explorations in the Microstructure of Cognition. MIT Press, Cambridge, MA, pp. 282–317.
Hsieh, C.-H., Tsai, H.-M., Yang, S.-W., Lin, S.-D., 2014. Predict scooter's stopping event using smartphone as the sensing device. In: Proceedings of the 2014 IEEE International Conference on Internet of Things(iThings), and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom). IEEE, pp. 17–23. https://doi.org/10.1109/iThings.2014.12.
Huang, W., Song, G., Hong, H., Xie, K., 2014. Deep architecture for traffic flow prediction: deep belief networks with multitask learning. IEEE Trans. Intell. Transp. Syst. 15 (5), 2191–2201. https://doi.org/10.1109/TITS.2014.2311123.
Jahanjoo, A., Naderan, M., Rashti, M.J., 2020. Detection and multi-class classification of falling in elderly people by deep belief network algorithms. J. Ambient. Intell. Humaniz. Comput. 11, 4145–4165. https://doi.org/10.1007/s12652-020-01690-z.
Jiang, G., Gang, L., Wang, J., 2006. Traffic congestion identification method of urban expressway. J. Traffic Transp. Eng. 6 (3), 87–91. https://doi.org/10.1016/S1872-2040(06)60039-X.
Jiang, M., Liang, Y., Feng, X., Fan, X., Pei, Z., Xue, Y., Guan, R., 2018. Text classification based on deep belief network and softmax regression. Neural Comput. Appl. 29 (1), 61–70. https://doi.org/10.1007/s00521-016-2401-x.
Jo, D., Yu, B., Jeon, H., Sohn, K., 2018. Image-to-image learning to predict traffic speeds by considering area-wide spatio-temporal dependencies. IEEE Trans. Veh. Technol. 68 (2), 1188–1197. https://doi.org/10.1109/TVT.2018.2885366.
Kerner, B.S., 1998. Experimental features of self-organization in traffic flow. Phys. Rev. Lett. 81 (17), 3797–3800. https://doi.org/10.1103/PhysRevLett.81.3797.
Kerner, B.S., 1999. Congested traffic flow: observations and theory. Transp. Res. Rec. 1678 (1), 160–167. https://doi.org/10.3141/1678-20.
Kerner, B.S., 2004. Three-phase traffic theory and highway capacity. Physica A 333, 379–440. https://doi.org/10.1016/j.physa.2003.10.017.
Kerner, B.S., Klenov, S.L., Hiller, A., 2007. Empirical test of a microscopic three-phase traffic theory. Nonlinear Dyn. 49 (4), 525–553. https://doi.org/10.1007/s11071-006-9113-1.
Ko, J., Guensler, R.L., 2005. Characterization of congestion based on speed distribution: a statistical approach using Gaussian mixture model. In: The Proceeding of the 82nd Annual Meeting of the Transportation Research Board, Transportation Research Board, pp. 105–113.
Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. In: Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS), Curran Associates Inc., pp. 1097–1105. https://doi.org/10.1145/3065386.
LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. Nature 521 (7553), 436. https://doi.org/10.1038/nature14539.
LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D., 1989. Backpropagation applied to handwritten zip code recognition. Neural Comput. 1 (4), 541–551. https://doi.org/10.1162/neco.1989.1.4.541.
Lee, H., Grosse, R., Ranganath, R., Ng, A.Y., 2009a. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In: Proceedings of the 26th Annual International Conference on Machine Learning, ACM, pp. 609–616. https://doi.org/10.1145/1553374.1553453.
Lee, H., Pham, P., Largman, Y., Ng, A.Y., 2009b. Unsupervised feature learning for audio classification using convolutional deep belief networks. In: Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems (NIPS) 2009, Neural Information Processing Systems Foundation, pp. 1096–1104.
Lee, Y.-S., Cho, S.-B., 2014. Activity recognition with android phone using mixture-of-experts co-trained with labeled and unlabeled data. Neurocomputing 126, 106–115. https://doi.org/10.1016/j.neucom.2013.05.044.
Li, D., Cui, J., Fan, Q., 2020. Research on abnormal monitoring of vehicle traffic network data based on support vector machine. Int. J. Vehicle Inform. Commun. Syst. 5 (2), 247–264. https://doi.org/10.1504/IJVICS.2020.108912.
Liang, S., Zhao, S., Ma, M., Liu, H., 2015. Analysis of traffic conditions in urban region based on data from fixed detectors. Discrete Dyn. Nat. Soc. 2015, 1–8. https://doi.org/10.1155/2015/184049.
Liu, M., Yu, L., Zhang, X., Guo, S., 2008. Cumulative logistic regression-based measurement models of road traffic congestion intensity. J. Beijing Jiaotong Univ. 32 (6), 52–56.
Liu, P., Han, S., Meng, Z., Tong, Y., 2014. Facial expression recognition via a boosted deep belief network. In: Proceedings of the IEEE conference on computer vision and pattern recognition, IEEE, pp. 1805–1812.
Liu, Y., Liu, Z., Vu, H.L., Lyu, C., 2020. A spatio-temporal ensemble method for large-scale traffic state prediction. Comput.-Aided Civil Infrastruct. Eng. 35 (1), 26–44. https://doi.org/10.1111/mice.12459.
Lozano, A., Manfredi, G., Nieddu, L., 2009. An algorithm for the recognition of levels of congestion in road traffic problems. Math Comput. Simul. 79 (6), 1926–1934. https://doi.org/10.1016/j.matcom.2007.06.008.
Lv, Y., Duan, Y., Kang, W., Li, Z., Wang, F.-Y., 2015. Traffic flow prediction with big data: a deep learning approach. IEEE Trans. Intell. Transp. Syst. 16 (2), 865–873. https://doi.org/10.1109/tits.2014.2345663.
Ma, J., Sheridan, R.P., Liaw, A., Dahl, G.E., Svetnik, V., 2015. Deep neural nets as a method for quantitative structure–activity relationships. J. Chem. Inf. Model. 55 (2), 263–274. https://doi.org/10.1021/ci500747n.
Mandal, R., Sonowal, P., Kumar, M., Saha, S., Nandi, S., 2020. Roadspeedsense: context-aware speed profiling from smart-phone sensors. EAI Endorsed Trans. Energy Web 7 (29). https://eudl.eu/doi/10.4108/eai.7-1-2020.162802.
Matthew, W., 2011. Bias of the Random Forest out-of-bag (OOB) error for certain input parameters. Open J. Statistics 1 (3), 205–211. https://doi.org/10.4236/ojs.2011.13024.
Miranda-Moreno, L.F., Chung, C., Amyot, D., Chapon, H., 2015. A system for collecting and mapping traffic congestion in a network using GPS smartphones from regular drivers. In: The Proceeding of the 94th Annual Meeting of the Transportation Research Board, Transportation Research Board, pp. 15-3792.
Mirjalili, S., Mirjalili, S.M., Lewis, A., 2014. Grey wolf optimizer. Adv. Eng. Softw. 69, 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007.
Mohamed, A.-R., Dahl, G.E., Hinton, G., 2012. Acoustic modeling using deep belief networks. IEEE Trans. Audio. Speech. Lang. Process. 20 (1), 14–22. https://doi.org/10.1109/tasl.2011.2109382.
Mohan, P., Padmanabhan, V.N., Ramjee, R., 2008. Nericell: rich monitoring of road and traffic conditions using mobile smartphones. In: Proceedings of the 6th Association for Computing Machinery conference on Embedded network sensor systems, ACM, pp. 323–336. https://doi.org/10.1145/1460412.1460444.

Montazeri-Gh, M., Fotouhi, A., 2011. Traffic condition recognition using the k-means clustering method. Sci. Iran. 18 (4), 930–937. https://doi.org/10.1016/j.scient.2011.07.004.
Petraki, V., Ziakopoulos, A., Yannis, G., 2020. Combined impact of road and traffic characteristic on driver behavior using smartphone sensor data. Accid. Anal. Prev. 144, 105657 https://doi.org/10.1016/j.aap.2020.105657.
Pholprasit, T., Choochaiwattana, W., Saiprasert, C., 2015. A comparison of driving behaviour prediction algorithm using multi-sensory data on a smartphone. In: 2015 IEEE/ACIS 16th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD). IEEE, pp. 1–6. https://doi.org/10.1109/SNPD.2015.7176249.
Poultney, C., Chopra, S., Cun, Y.L., 2007. Efficient learning of sparse representations with an energy-based model. In: Advances in Neural Information Processing Systems. MIT Press, pp. 1137–1144.
Predic, B., Stojanovic, D., 2015. Enhancing driver situational awareness through crowd intelligence. Expert Syst. Appl. 42 (11), 4892–4909. https://doi.org/10.1016/j.eswa.2015.02.013.
Reddy, S., Burke, J., Estrin, D., Hansen, M., Srivastava, M., 2008. Determining transportation mode on mobile phones. In: Wearable Computers, 2008, ISWC 2008, 12th IEEE International Symposium, IEEE, pp. 25–28. https://doi.org/10.1109/ISWC.2008.4911579.
Schmidhuber, J., 2015. Deep learning in neural networks: an overview. Neural Netw. 61, 85–117. https://doi.org/10.1016/j.neunet.2014.09.003.
Sekuła, P., Marković, N., Vander Laan, Z., Sadabadi, K.F., 2018. Estimating historical hourly traffic volumes via machine learning and vehicle probe data: a Maryland case study. Transp. Res. Part C Emerg. Technol. 97, 147–158. https://doi.org/10.1016/j.trc.2018.10.012.
Shao, H., Jiang, H., Zhang, H., Liang, T., 2018. Electric locomotive bearing fault diagnosis using novel convolutional deep belief network. IEEE Trans. Ind. Electron. 65 (3), 2727–2736. https://doi.org/10.1109/TIE.2017.2745473.
Shen, C., Xie, J., Wang, D., Jiang, X., Shi, J., Zhu, Z., 2019. Improved hierarchical adaptive deep belief network for bearing fault diagnosis. Appl. Sci. 9 (16), 3374. https://doi.org/10.3390/app9163374.
Socher, R., Huval, B., Bhat, B., Manning, C.D., Ng, A.Y., 2012. Convolutional-recursive deep learning for 3D object classification. In: Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS), Curran Associates Inc., pp. 656–664.
Storn, R., Price, K., 1997. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. J. Glob. Optim. 11 (4), 341–359. https://doi.org/10.1023/a:1008202821328.
Tang, B., Liu, X., Lei, J., Song, M., Tao, D., Sun, S., Dong, F., 2016. Deepchart: Combining deep convolutional networks and deep belief networks in chart classification. Signal Processing 124, 156–161. https://doi.org/10.1016/j.sigpro.2015.09.027.
Tompson, J., Jain, A., LeCun, Y., Bregler, C., 2014. Joint Training of a Convolutional Network and a Graphical Model for Human Pose Estimation. arXiv:1406.2984.
Treiber, M., Kesting, A., 2012. Validation of traffic flow models with respect to the spatiotemporal evolution of congested traffic patterns. Transp. Res. Part C Emerg. Technol. 21 (1), 31–41. https://doi.org/10.1016/j.trc.2011.09.002.
Ua-areemitr, E., Sumalee, A., Lam, W.H., 2019. Low-cost road traffic state estimation system using time-spatial image processing. IEEE Intelligent Transp. Syst. Magazine 11 (3), 69–79. https://doi.org/10.1109/MITS.2019.2919634.
Uddin, M.Z., Hassan, M.M., Almogren, A., Alamri, A., Alrubaian, M., Fortino, G., 2017. Facial expression recognition utilizing local direction-based robust features and deep belief network. IEEE Access 5, 4525–4536. https://doi.org/10.1109/ACCESS.2017.2676238.
Vlahogianni, E.I., Barmpounakis, E.N., 2017. Driving analytics using smartphones: algorithms, comparisons and challenges. Transp. Res. Part C Emerg. Technol. 79, 196–206. https://doi.org/10.1016/j.trc.2017.03.014.
Wang, J., Chen, R., He, Z., 2019. Traffic speed prediction for urban transportation network: a path based deep learning approach. Transp. Res. Part C Emerg. Technol. 100, 372–385. https://doi.org/10.1016/j.trc.2019.02.002.
Wang, Y., Pan, Z., Yuan, X., Yang, C., Gui, W., 2020. A novel deep learning based fault diagnosis approach for chemical process with extended deep belief network. ISA Trans. 96, 457–467. https://doi.org/10.1016/j.isatra.2019.07.001.
Wu, Y., Tan, H., Qin, L., Ran, B., Jiang, Z., 2018. A hybrid deep learning based traffic flow prediction method and its understanding. Transp. Res. Part C Emerg. Technol. 90, 166–180. https://doi.org/10.1016/j.trc.2018.03.001.
Xing, H., Wang, G., Liu, C., Suo, M., 2021. PM2. 5 concentration modeling and prediction by using temperature-based deep belief network. Neural Netw. 133, 157–165. https://doi.org/10.1016/j.neunet.2020.10.013.
Xu, D., Dai, H., Wang, Y., Peng, P., Xuan, Q., Guo, H., 2019. Road traffic state prediction based on a graph embedding recurrent neural network under the SCATS. Chaos: an Interdisciplinary. J. Nonlinear Sci. 29 (10), 103125 https://doi.org/10.1063/1.5117180.
Xu, D., Song, G., Gao, P., Cao, R., Nie, X., Xie, K., 2011. Transportation modes identification from mobile phone data using probabilistic models. In: The 7th International Conference on Advanced Data Mining and Applications, Springer, pp. 359–371. https://doi.org/10.1007/978-3-642-25856-5_27.
Xu, D., Wang, Y., Peng, P., Beilun, S., Deng, Z., Guo, H., 2020a. Real-time road traffic state prediction based on kernel-KNN. Transportmetrica A: Transp. Sci. 16 (1), 104–118. https://doi.org/10.1080/23249935.2018.1491073.
Xu, D., Wei, C., Peng, P., Xuan, Q., Guo, H., 2020b. GE-GAN: A novel deep learning framework for road traffic state estimation. Transp. Res. Part C Emerg. Technol. 117, 102635 https://doi.org/10.1016/j.trc.2020.102635.
Yang, H., Qiao, F., 1998. Neural network approach to classification of traffic flow states. J. Transp. Eng. A. Syst. 124 (6), 521–525. https://doi.org/10.1061/(ASCE)0733-947X(1998)124:6(521).
Yu, J.J.Q., Gu, J., 2019. Real-time traffic speed estimation with graph convolutional generative autoencoder. IEEE Trans. Intell. Transp. Syst. 20 (10), 3940–3951. https://doi.org/10.1109/TITS.2019.2910560.
Zeng, W., He, Z., She, X., 2011. Data repair method for real time urban link speed estimation. In: 11th International Conference of Chinese Transportation Professionals (ICCTP): Towards Sustainable Transportation Systems, ASCE, pp. 1733–1744.
Zhan, X., Li, R., Ukkusuri, S.V., 2020. Link-based traffic state estimation and prediction for arterial networks using license-plate recognition data. Transp. Res. Part C Emerg. Technol. 117, 102660 https://doi.org/10.1016/j.trc.2020.102660.
Zhang, D., Xiao, F., Shen, M., Zhong, S., 2021. DNEAT: a novel dynamic node-edge attention network for origin-destination demand prediction. Transp. Res. Part C Emerg. Technol. 122, 102851 https://doi.org/10.1016/j.trc.2020.102851.
Zhang, Z., Yang, X., 2020. Freeway traffic speed estimation by regression machine-learning techniques using probe vehicle and sensor detector data. J. Transp. Eng. A. Syst. 146 (12), 04020138. https://doi.org/10.1061/JTEPBS.0000455.
Zhao, Z., Jiao, L., Zhao, J., Gu, J., Zhao, J., 2017. Discriminant deep belief network for high-resolution SAR image classification. Pattern Recognit. 61, 686–701. https://doi.org/10.1016/j.patcog.2016.05.028.
Zheng, J., Cao, J., He, Z., Liu, X., 2014. iTrip: Traffic signal prediction using smartphone based community sensing. In: 17th International IEEE Conference on Intelligent Transportation Systems (ITSC). IEEE, pp. 2944–2949. https://doi.org/10.1109/ITSC.2014.6958162.
Zhu, A., Xu, C., Li, Z., Wu, J., Liu, Z., 2015. Hybridizing grey wolf optimization with differential evolution for global optimization and test scheduling for 3D stacked SoC. J. Syst. Eng. Electron. 26 (2), 317–328. https://doi.org/10.1109/JSEE.2015.00037.
Zhu, G., Liu, K., Qiao, L., 2012. Modeling and simulation for traffic states with ANFIS based on T-S model. J. Beijing Jiaotong Univ. 36 (6), 96–101.
Zhuang, B., Yang, X., Li, K., 2006. Criterion and detection algorithm for road traffic congestion incidents. China J. Highway Transp. 19 (3), 82–86.
Zong, X., Wen, X., 2015. A new approach to estimate real-time traveling speed with accelerometer. Int. J. Distrib. Sens. Netw. 11 (10), 928168 https://doi.org/10.1155/2015/928168.