Received September 7, 2019, accepted September 24, 2019, date of publication September 30, 2019, date of current version October 11, 2019.

*Digital Object Identifier 10.1109/ACCESS.2019.2944686*

# Toward Transportation Mode Recognition Using Deep Convolutional and Long Short-Term Memory Recurrent Neural Networks

**YANJUN QIN[1], HAIYONG LUO[2], (Member, IEEE), FANG ZHAO[1], CHENXING WANG[1], JIAQI WANG[3], AND YUEXIA ZHANG[3]**

[1]School of Software Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China
[2]Research Center for Ubiquitous Computing Systems, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100876, China
[3]School of Information and Communication Engineering, Beijing Information Science and Technology University, Beijing 100876, China

Corresponding authors: Haiyong Luo (yhluo@ict.ac.cn) and Fang Zhao (zfsse@bupt.edu.cn)

This work was supported in part by the National Key Research and Development Program under Grant 2018YFB0505200, in part by the Action Plan Project of the Beijing University of Posts and Telecommunications supported by the Fundamental Research Funds for the Central Universities under Grant 2019XD-A06, in part by the Special Project for Youth Research and Innovation, Beijing University of Posts and Telecommunications, supported by the Fundamental Research Funds for the Central Universities under Grant 2019PTB-011, in part by the National Natural Science Foundation of China under Grant 61872046, in part by the Key Research and Development Project from Hebei Province under Grant 19210404D, in part by the Open Project of the Beijing Key Laboratory of Mobile Computing and Pervasive Device, and in part by the Blue Fire Plan (Huizhou) Industry-University Joint Innovation Project of Ministry of Education under Grant CXZJHZ201729.

**ABSTRACT** With the rapid development of mobile Internet techniques, using the sensor-rich smartphones to sense various contexts attracts much attention, such as transportation mode recognition. The transportation mode information can help to improve urban planning, traffic management and journey planning. Though much work has been done on the transportation mode recognition using classic machine learning algorithms, the performance of these methods is not reasonable and heavily relies on the effectiveness of handcrafted features. In this paper, we leverage the strong representation ability of deep learning method and present a deep-learning-based algorithm for transportation mode recognition, namely CL-TRANSMODE, which is capable of accurately detecting multiple transportation modes. The algorithm first uses a convolutional neural network (CNN) to learn appropriate and robust feature representations for transportation modes recognition. Then, an LSTM network performs a further learning of the temporal dependencies characteristics on the feature vectors of CNN output. To further enhance the accuracy of transportation mode recognition, several artificial segments and peak features are extracted from the raw sensor measurements. These features characterize the transportation modes over a much long period of time (minutes or hours). By combining the CNN-extracted features and handcrafted features, our proposed CL-TRANSMODE transportation mode recognition algorithm can accurately differentiate eight transportation modes, i.e., walking, running, bicycling, driving a car, riding a bus, taking a metro, taking a train, or being stationary. Extensive experiments on both the SHL and HTC datasets demonstrate that use our proposed CL-TRANSMODE transportation mode recognition algorithm which outperforms the state-of-the-art comparative algorithms. On the SHL dataset, which contain barometric data, the accuracy using the CL-TRANSMODE algorithm can reaches 98.1%.

**INDEX TERMS** Mobile sensing, context-aware, transportation mode recognition, CNN, LSTM.

## I. INTRODUCTION

With the ever-increasing computing and perception capabilities of smartphones, fine-grained activity recognition attracts much attention. As a special kind of activity, the transportation mode recognition can accurately differentiate various transportation patterns, which helps to enhance urban planning, traffic management, individual carbon footprint analysis, travel time estimation, and journey planning.

Though much work has been conducted on transportation mode recognition, most work uses classic machine learning algorithms to determine transportation modes, such as tree-based algorithms [1], [2] support vector machine (SVM), and Adaboost [3]–[5]. The performance of these algorithms is not

The associate editor coordinating the review of this manuscript and approving it for publication was Rashid Mehmood.

reasonable and mainly depends on the validity of handcrafted features, which is time-consuming and requires quality expert knowledge [6], [7].

In recent years, deep learning has drawn much attention due to its remarkable capability to automatically extract features from large-scale raw data and strong representation capability [8], [9]. The convolutional neural network (CNN) [10] can automatically create a hierarchy of abstract features by stacking several convolutional operators. The long short-term memory (LSTM) recurrent neural network can reasonably model temporal dependencies in time series problems. In this study, we combine a CNN network and an LSTM network into a unified framework to detect transportation mode (CL-TRANSMODE). By making advantage of the strong representation capability of CNN and the captured time dependencies by LSTM, our proposed algorithm can accurately recognize the transportation modes.

The main contributions of this paper are summarized as follows:

1) Integrating the advantages of features automatically extracted by CNN for a short time interval and handcrafted segment and peak features for a much longer time interval (such as several minutes or hours). These segment features and peak features can reflect the corresponding transportation modes covering a much longer period of time, which cannot be learnt by the CNN. Furthermore, we use LSTM layers to learn the temporal dependencies of various features.
2) Different from single activity recognition or the transportation mode recognition, our proposed CL-TRANSMODE algorithm can accurately recognize both complex human activities and transportation modes simultaneously, including walking, running, bicycling, driving a car, riding a bus, taking a metro, taking a train, or being stationary.
3) We performed extensive evaluation of our proposed algorithm and other comparative algorithms on two large public datasets, i.e., Sussex-Huawei and the HTC dataset. The experimental results demonstrated that our proposed algorithm can recognize transportation modes in various complex scenarios with high accuracy, outperforming the state-of-the-art baseline methods.

## II. RELATED WORK

Based on our literature reviews of transportation mode recognition, we categorized related works into two categories: (1) traditional machine learning based transportation mode identification; (2) deep learning based transportation mode identification.

Many traditional machine-learning algorithms are used for transportation mode recognition. Dang-Nhac *et al.* [11] employed SVM to differentiate six transportation modes (driving a car, riding a bus, motorcycling, bicycling, walking and being stationary) on the HTC dataset. Zhou *et al.* [12] introduced a chained random forest (RF) to detect travel modes, which automatically classifies smartphone data into different travel modes. Guvensan *et al.* [13] proposed a segment-based transportation mode detection architecture. To improve the accuracy of traditional classification algorithms (RF, k-nearest neighbor, and naïve Bayes), a post-processing step (the healing algorithm) was added to correct the misclassification results. To avoid the effect of training sample imbalance, Chang *et al.* [14] combined the kNN classifier and the synthetic minority over-sampling technique (SMOTE) algorithm. MonoSense [15] leverages the phone serving cell information to detect transportation mode using a traditional decision tree classifier. In general, the accuracy of the aforementioned methods is not reasonable and heavily relies on the expert knowledge of extracting discriminative features. These machine learning models only have a shallow architecture to capture features.

Deep learning models inversely use multilayer architecture where different layers capture features from different perspectives. Considering the strong automatic feature extraction and feature representation abilities, several studies attempted using deep learning methods to determine transportation modes. Fang *et al.* [16] adopted the deep neural network (DNN) to recognize and differentiate five transportation modes (still, walk, run, bike, and vehicle). Liang and Wang [17] introduced a convolutional neural network to detect transportation modes by only using the lightweight accelerometer embedded in commodity smartphones. Vu *et al.* [18] used a recurrent neural network (RNN) to detect transportation mode on a mobile phone. The Mago system [19] uses a two-layer classifier to recognize transportation modes. The first layer uses a random forest classifier to distinguish the still state and non-still state. The second layer uses a neural network model to recognize the six transportation modes. The DeepTransport system [20] builds a deep LSTM learning architecture, which combines global positioning system data, road structure, and point-of-interest information to determine human mobility and transportation patterns. Liu and Lee [21] leveraged a bidirectional LSTM classifier to identify transportation modes.

Different from the above-mentioned methods, in this paper, we combined CNN and LSTM to form a unified framework for recognizing complex transportation modes. This architecture takes advantage of the automatic feature extraction ability of convolutional operations and the time-dependent feature extraction ability of LSTM. To further improve the accuracy of transportation mode identification, some handcrafted semantic features (i.e., segment and peak features) covering a much longer time interval are used and concatenated with the features learnt by the CNN model. Extensive experimental results demonstrated that our proposed CL-TRANSMODE can accurately recognize various transportation modes and significantly outperforms other traditional methods.

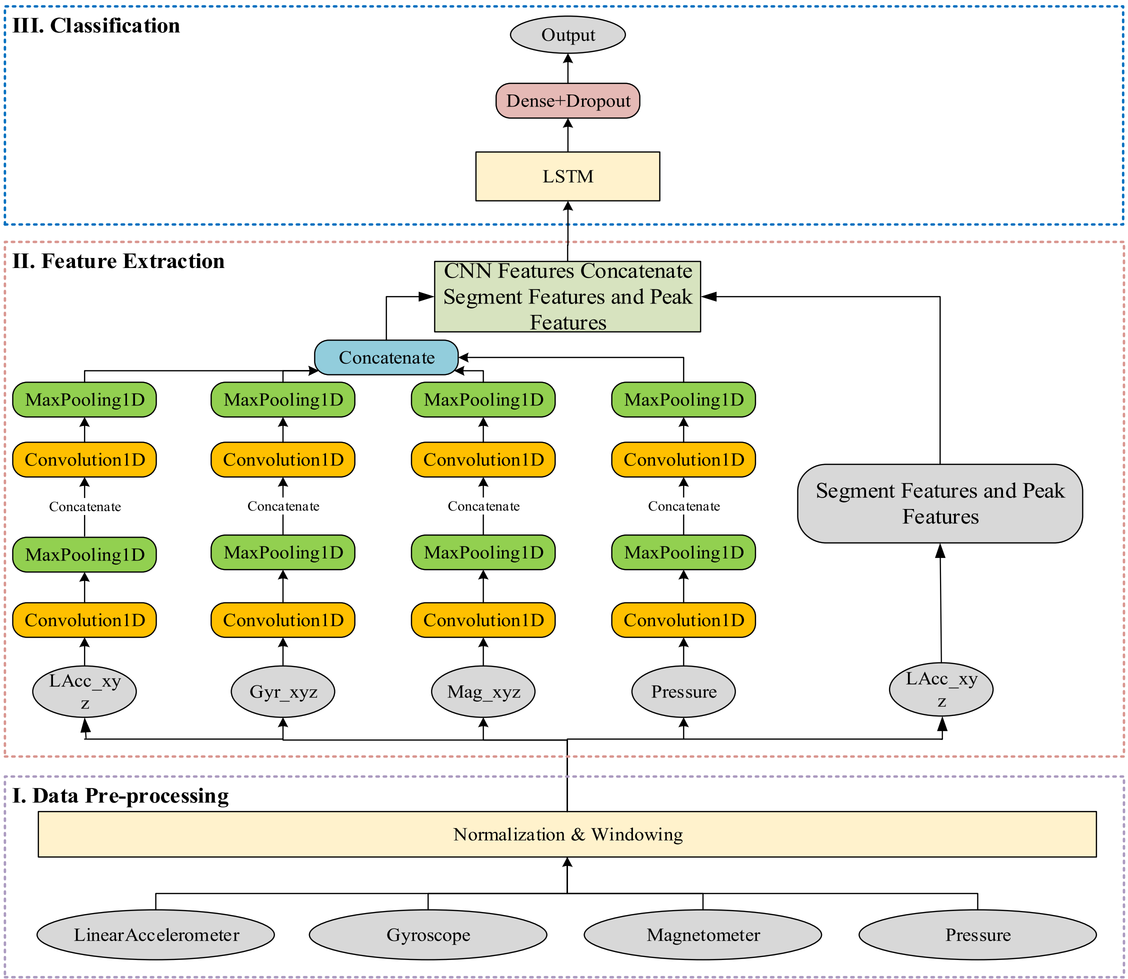



**FIGURE 1.** **The architecture of our proposed CL-TRANSMODE transportation mode recognition algorithm, which comprises of three layers. The bottom is raw data preprocessing layer. The middle layer is a convolutional neural network (CNN) that automatically extracts features and some segment features and peak features. The top layer includes long and short memory neural network (LSTM), connection layers, and dropout layers, which are united to perform the classification of eight transportation modes.**

## III. MATERIALS AND METHODS

### A. SYSTEM ARCHITECTURE

Our proposed transportation mode recognition algorithm CL-TRANSMODE consists of three main modules: data pre-processing, feature extraction, and classification, as shown in Fig. 1.

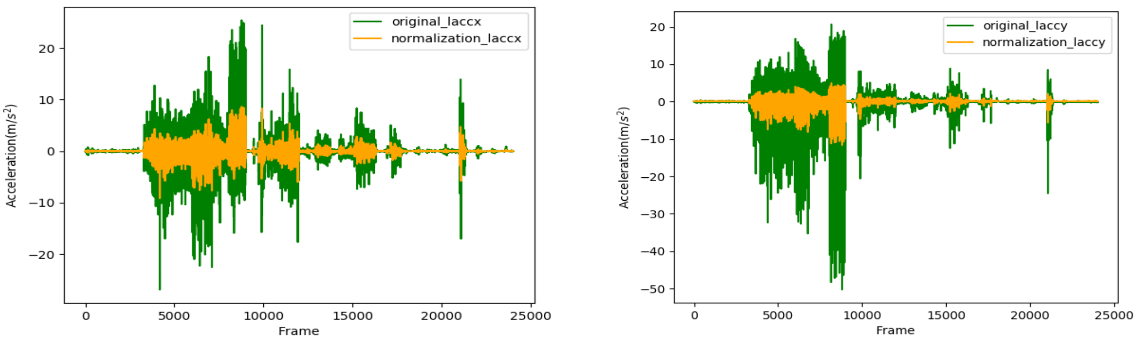
In the data preprocessing layer, the raw sensor data consisting of accelerometer, gyroscope, barometer, and magnetometer are normalized and segmented with a sliding window. In the middle layer, we use a CNN network to automatically acquire suitable features from the preprocessed data over a short-scale time interval. To further improve recognition accuracy, we also handcrafted several segment features and peak features, which can reflect the corresponding transportation mode over a much longer period of time (minutes or hours). In the top layer, we employ an LSTM unit to learn the temporal dependencies of the features extracted by the CNN in small-scale time interval and the handcrafted features in a large-scale time interval. Above the LSTM layer, connection layers are used to increase the learning ability of the model. Dropout layers are applied to prevent overfitting. At last, a SoftMax activation function is used to output the transportation mode recognition result.

### B. DATA PRE-PROCESSING

#### 1) NORMALIZATION

To balance the value range of heterogeneous sensor data, a normalization operation is performed for the CNN. In this paper, we used the Z-Score normalization method [22] to normalize various sensor data, which can guarantee stable convergence of weight and biases for our proposed CL-TRANSMODE model.

We compared the raw data with the normalized data in Fig. 2. The green lines denote the raw data, and the orange lines represent the normalized data. It can be seen from




(a) linear acceleration x-axis data

(b) linear acceleration y-axis data

**FIGURE 2.** Normalization of raw different sensor data.

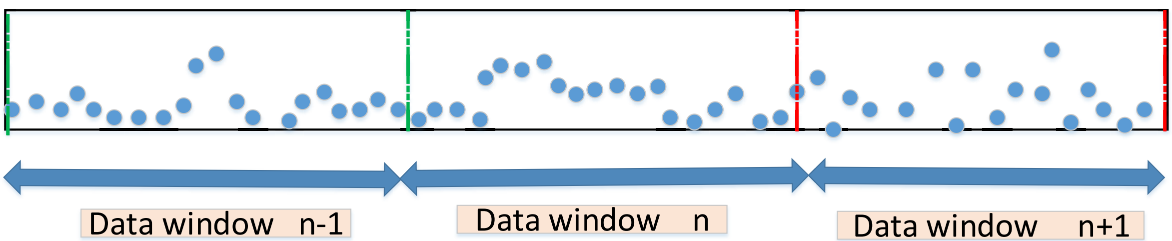



**FIGURE 3.** Data segmentation using a sliding and fixed-size window. Each segmented data are used as an input unit for transportation recognition.

Fig. 2 that the range of the normalized data for different input becomes similar after the Z-Score normalization operation.

### 2) SEGMENTATION OF DATA

The collected raw sequence data from different sensors contains various noises. To reduce the influence of various noises and obtain a steady feature representation, we used a fixed-length sliding window to segment the raw data. We regard each window as a "sequence", which is the initial input of the network. The raw data were then segmented into a series of fixed-length sequences, as Fig. 3 shows.

The selection of sliding window size influences the feature representation and the prediction accuracy of transportation modes. Some useful features may be ignored if a much small window is chosen. Conversely, if the features are extracted from the raw data with an extremely large window, the first prediction delay will be long and the change of transportation transfer will not be responded in time, which will be evaluated in the later experimental parts.

### 3) CNN ARCHITECTURE FOR FEATURE EXTRACTION

In this section, a CNN [23] is used to learn deep features. We use one-dimensional convolutions operations [24] on the raw sensor data. The kernel acts as a filter in the one-dimensional domain. The kernel can remove outliers, filter data, and extract features. The $l^{th}$ feature map using a one-dimensional convolution operation is calculated as Equation (1) shows,

$$conv_u^{l,v,p}(x_u^{l-1}) = f(\sum_{c=1}^{p} w_{u+c-1}^{l-1} x_{u+c-1}^{l-1} + b_v) \tag{1}$$

where $l$ is the sequence number of convolution layer. $p$, $x$, $w$, $C$ denote the kernel size, the input data, the weight, and the index of the filters, respectively. $v$, $u$ represent the sequence number in the kernels and the unit index of convolution layers. $f$ is the rectified linear activation function ReLU. b denotes the bias of the convolution layers.

Fig. 4 shows the convolutional neural network used in our CL-TRANSMODE system. There are three CNN layers and three maxpooling layers in the CL-TRANSMODE system. Convolution operations and pooling operations are performed on each element of different sensor data, i.e., 3-axis accelerometer, 3-axis gyroscope, 3-axis magnetometer and pressure data. $300 \times 1$ raw components are fed into the first convolution layer in per epoch. The optimal parameters of the CNN and maxpooling layers are shown in the black boxes of Fig. 4. The three numbers of the black boxes in convolution layers represent the number of one-dimensional convolution, kernel size and step length, respectively. The three numbers of maxpooling layers represent the number of input data, pool size and stride size, respectively. Experimental results showed that this CNN architecture can extract features with reasonable discrimination ability.

To visually illustrate the distinguishability of different transportation modes using the features extracted by the CNN, we use the stochastic neighbor embedding (T-SNE) method to reduce the dimension of features. As Fig. 5 shows, the extracted features of different transportation modes are aggregated, which verifies that the extracted features are rather differentiable among different traffic patterns.

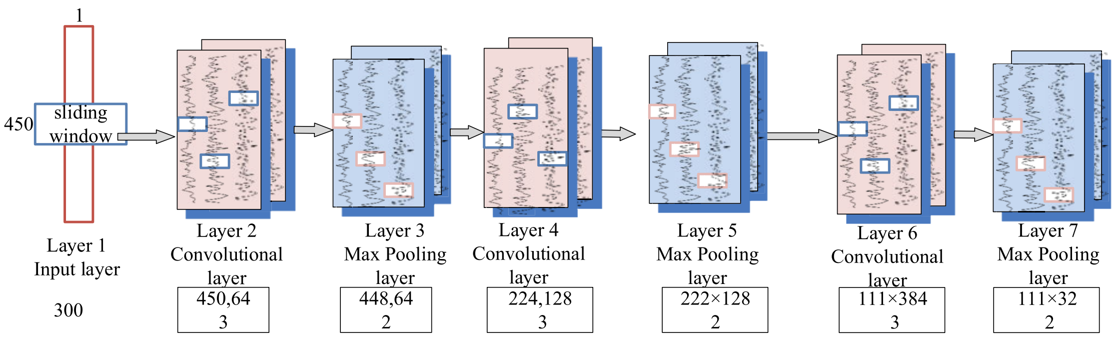



**FIGURE 4.** The convolutional neural network used in our system.

**TABLE 1.** Segment features and peak features used for transportation recognition in acceleration data.

| Domain | Features | Instructions |
|---|---|---|
| peak | volume | integral of the peaks at time t |
| | intensity | the maximum value of the all peaks |
| | length | the duration of acceleration and deceleration |
| | kurtosis | measuring kurtosis of a peak |
| | skewness | degree of asymmetric distribution of a peak |
| segment | variance of peak features (10 features) | the variance is calculated by 10 peak features |
| | peak frequency (2 features) | the frequency in 2 features |
| | stationary duration | the stationary duration in a station in a segment time |
| | stationary frequency | the frequency of still over a period of time |

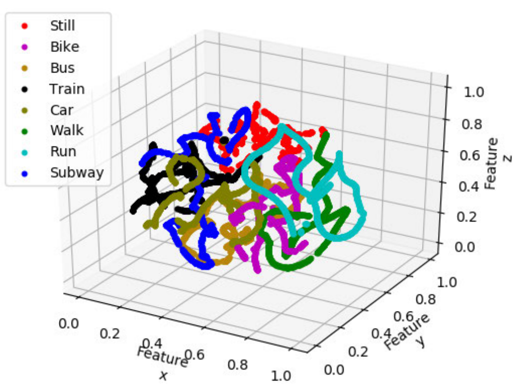



**FIGURE 5.** Using the T-SNE method to visually show feature distinguishability of different transportation modes extracted by the CNN.

### C. LONG AND SHORT MEMORY NEURAL NETWORK ARCHITECTURE

Considering that the same transportation patterns have high temporal correlation, we make advantage of LSTM to capture time dependencies from sequential features extracted by CNN and handcraft to improve the transportation mode recognition accuracy.

By combining gate structures, LSTM ensures that the transportation mode information is retained to learn the temporal dependencies of features; it can also retain deviations for back propagation in network layers. The network structure includes one layers LSTM and four layers one-dimension fully connected layers. The number of cell is 128 in LSTM layer.

### D. SEGMENT FEATURES AND PEAK FEATURES

Though extending the aforementioned CNN network can learn features of different transportation modes, the heavy calculation and storage cost will limit its use on the smartphones. To improve the accuracy of transportation mode recognition, we introduced some peak features and segment features as listed in Table 1. These handcrafted features cover longer time dependencies (minutes or hours).

Peak features: peak features represent the acceleration and deceleration behavior of different transportation modes, as Fig.6 shows. Two pre-defined thresholds are selected to identify peak areas in acceleration. Once the modulus of acceleration is bigger than the threshold (e.g., threshold = 0.6 m/s2), the starting boundary of the peak area is marked. The threshold for identifying the end boundary of the peak area is set to a specific value on trial and error, e.g. 0.2 m/s$^2$ in this paper. As listed in Table 1, other features are also extracted for each peak.

- Volume: Volume is the integral of three-axis accelerometer for a peak.
- Intensity: Intensity represents the maximum modulus of acceleration or deceleration for a peak.
- Length: Length represents the duration of acceleration or deceleration for a peak.
- Kurtosis: Kurtosis is a descriptor of the modulus of acceleration or deceleration distribution. Kurtosis is a measure of the "tailedness" of the probability distribution as Formula (2) shows. $n$ donates the number of samples in a peak. $x$ donates the value of acceleration

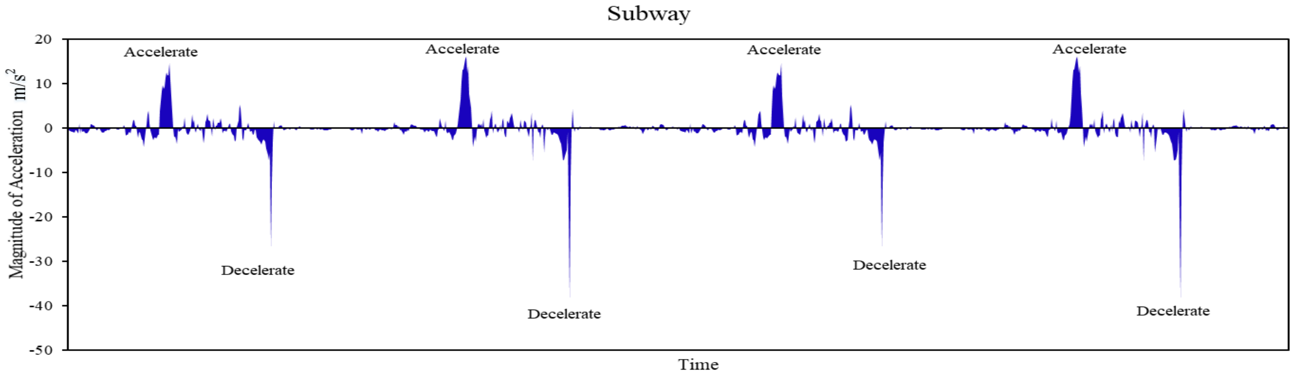



**FIGURE 6.** Peaks appear at the speed-up and speed-down periods in a subway.

or deceleration. $\overline{x}$ donates the mean value of acceleration or deceleration in a peak.

$$\text{kurtosis} = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})}{\left(\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})\right)^2} - 3 \tag{2}$$

- Skewness: Skewness measures the asymmetry of the acceleration or deceleration distribution, as Formula (3) shows. The variables in Formula (3) have the same definition as Formula (2).

$$\text{skewness} = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^3}{\left(\left(\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})\right)^2\right)^{\frac{3}{2}}} \tag{3}$$

Segment features: The segment features are used to characterize acceleration and deceleration for longer periods. It is assumed that the patterns of acceleration and deceleration is similar for the same transportation mode. For example, the Beijing subway always stops for two minutes at each station. The variance of peak features (ten features), the frequency of peak or stationary and duration of the intermittent stationary periods are similar for the same transportation mode, but are distinctive for different transportation modes. Segment features are calculated from peak features and provide fine-grained information for the specific transportation modes.

Though these segment features and peak features bring about some statistical calculation cost, the distinguishability of these segment and peak features help detest various transportation modes. When a gas pedal or a brake pedal is depressed, the vehicle speed will increase or decrease. The magnitudes of three dimensional acceleration also change over time. We use the magnitude of three dimensional acceleration to identify peak areas. Fig.6 shows that the acceleration values increase or decrease sharply during the speed-up and speed-down phases of a subway.

The cumulative distribution functions of these features are shown in Fig.7, which demonstrates the possibility of using these segment and peak features for transportation mode recognition. Fig.7a compares the distribution of stationary frequency feature of eight kinds of transportation modes. The remarkable differences among eight kinds of transportation modes confirm that the stationary frequency feature can help identify transportation modes. Fig.7b, Fig.7c and Fig.7d show the cumulative distribution functions of Stationary Duration / Kurtosis / Length features, respectively.

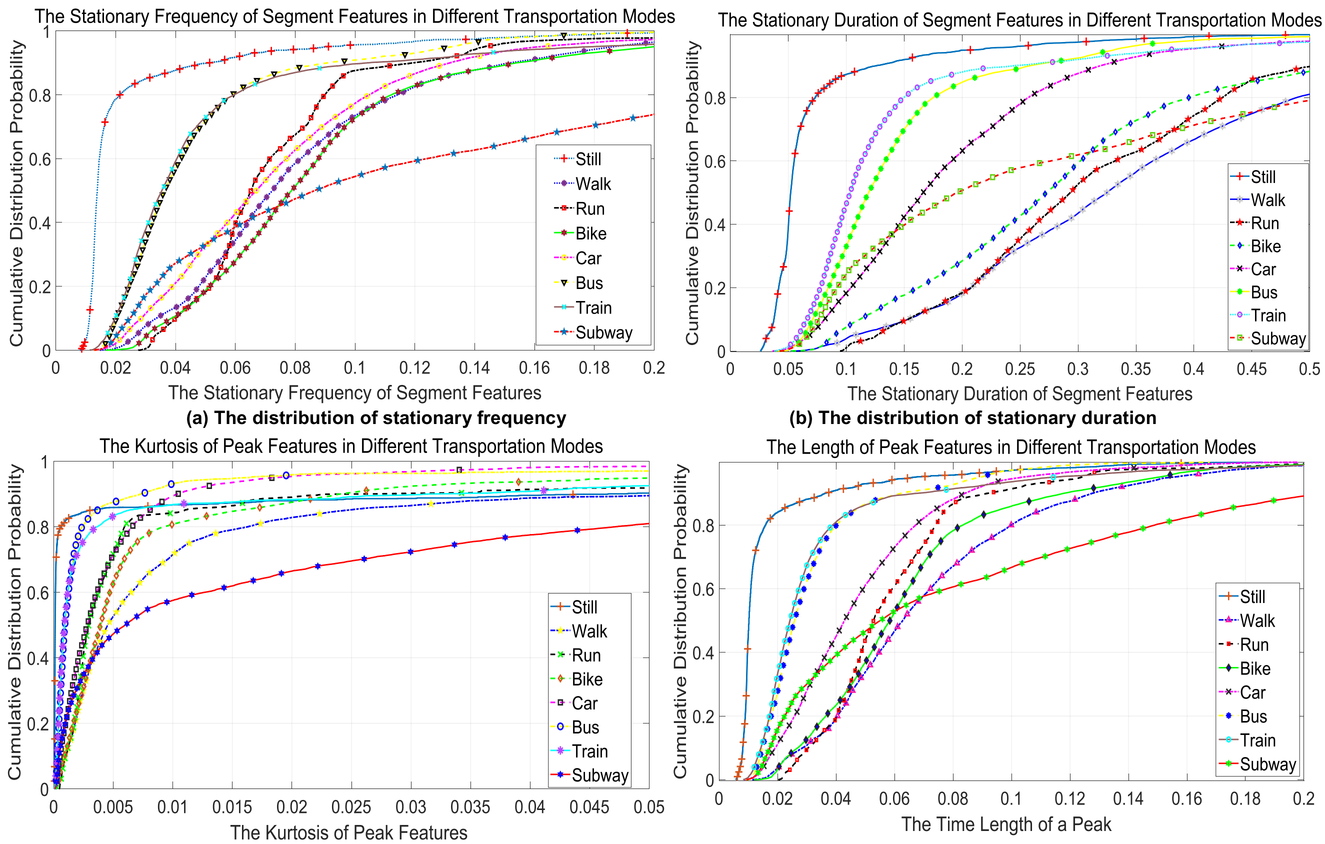
We use an $m \times n$ tensor to represent short period features obtain from CNN layers. The one-dimensional tensor is converted to $q \times n$ of peak and segment features in long time interval. If the number of peak and segment features is not enough, we add zero into the tensor of peak and segment features. Then the short period features are combined with peak and segment features in the architecture.

## IV. EVALUATION AND ANALYSIS

We evaluated our proposed CL-TRANSMODE algorithm on two large-scale public datasets, i.e., SHL dataset [25] and HTC dataset [26]. As comparisons, several other state-of-the-art algorithms are also implemented. We also evaluate the influence of different parameters on the CL-TRANSMODE algorithm, such as different window size, and the segment and peak features, etc.

### A. DATASETS

To evaluate the CL-TRANSMODE algorithm, the popular SHL dataset (SHL) and HTC dataset are used. The SHL dataset was collected by three volunteers in the UK in 2017. It contains eight types of transportation modes. All samples in SHL dataset was collected with Huawei Mate 9 smartphones, which were placed in different positions, i.e., in bag, in hand, strapped to chest or in pocket. The HTC Dataset was collected by one hundred and fifty volunteers in 2012 using HTC smartphones. It contains ten types of transportation modes. Each sample contains accelerometer, gyroscope and magnetometer data. The sampling rate of SHL and HTC datasets are both set to 100Hz. To keep the types of different transportation modes in consistence between SHL dataset (SHL) and HTC dataset, the data of motorcycle and high-speed rail in HTC Dataset were abandoned in our experiments.




(a) The distribution of stationary frequency (b) The distribution of stationary duration

(c) The kurtosis distribution of the acceleration and deceleration (d) The distribution of duration of the acceleration and deceleration

**FIGURE 7.** The segment and peak feature distribution of acceleration sensor for different transportation mode.

**TABLE 2.** The distribution OF SHL and HTC dataset For different transportation patterns.

| Transportation patterns | SHL dataset | HTC dataset |
|---|---|---|
| Still | 114h | 1857h |
| Walk | 113h | 1384h |
| Run | 22h | 914h |
| Bike | 80h | 141h |
| Motorcycle | 0h | 1817h |
| Car | 92h | 767h |
| Bus | 103h | 1317h |
| subway | 78h | 384h |
| Train | 100h | 334h |
| High Speed Rail | 0h | 163h |

The detailed transportation pattern distributions of the SHL dataset and HTC dataset are listed in Table 2.

Considering that the HTC dataset does not contain barometric data, we use it to evaluate the generality of our proposed CL-TRANSMODE without barometric sensor.

### B. EXPERIMENTAL PLATFORM

All experiments are run on a node of the DAWN supercomputer. A lightweight deep learning framework PyTorch is used to build the CL-TRANSMODE network. The detailed software and hardware configurations of the experimental platform are listed in Table 3.

**TABLE 3.** The software and hardware configurations of the experimental platform.

| Hardware and Software | Description |
|---|---|
| CPU | Intel(Intel Corporation, California, US) E5-2680 2.4 |
| Operating System | Centos (US) 7.4 |
| Memory Size | 64 GB |
| GPU | Tesla (Nvidia Corporation, , California, US) K80 12GB ×2 |
| Development Language | Python |
| Operation Environments | Python 3.6.5 |
| Deep Learning Framework | PyTorch 0.4.1 |

### C. EXPERIMENTAL RESULTS ON THE SHL DATASET

#### 1) INFLUENCE OF WINDOW SIZE

Fig.8 shows the accuracy of the CL-TRANSMODE algorithm under different window sizes. In general, the accuracy of CL-TRANSMODE algorithm first goes up and then goes down with the window size increase from 100 to 750.

**TABLE 4.** The identification accuracy of eight transportation modes with different window sizes.

| Window/Overlap | Still | Walk | Run | Bike | Car | Bus | Train | Subway |
|---|---|---|---|---|---|---|---|---|
| Win = 100 | 98.0% | 98.2% | 99.2% | 97.7% | 98.6% | 97.3% | 94.9% | 92.4% |
| Win =200 | 98.1% | 98.2% | 99.6% | 97.9% | 99.4% | 98.0% | 94.7% | 918% |
| Win = 300 | 98.4% | 97.9% | 99.7% | 97.7% | 99.3% | 97.3% | 94.7% | 93.3% |
| Win = 450 | 97.9% | 98.2% | 99.7% | 97.8% | 98.7% | 97.9% | 94.9% | 90.5% |
| Win = 750 | 97.5% | 98.2% | 99.8% | 97.3% | 99.1% | 96.8% | 93.7% | 90.4% |

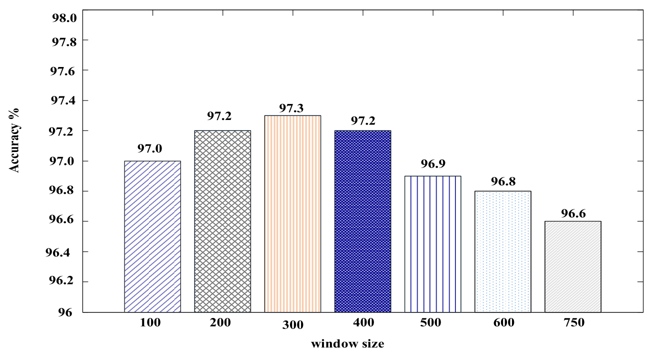



**FIGURE 8.** The classification accuracy using different window sizes.

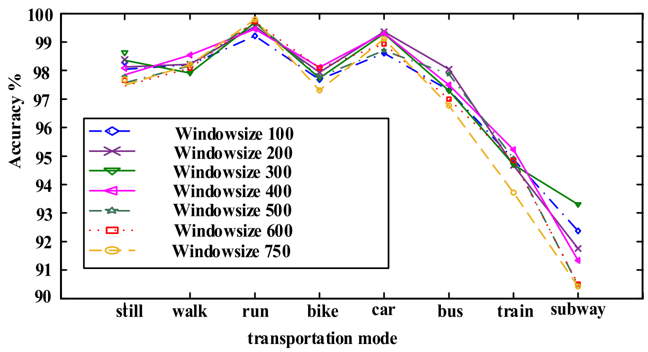



**FIGURE 9.** The classification accuracy under different window sizes (Win).

When window size is set to 300, the performance of our proposed algorithm is the best. Fig.9 compares the performance with different window sizes for each transportation mode. The accuracies of recognizing still, walk, run, bike, car and bus modes are higher than 95%. However, nearly all of the accuracy of recognizing train and subway modes are under 95%. Because train and subway modes are smooth and steady on the road, they are easily misclassified. However, the accuracy of identifying train and subway is the best (94.7% and 93.3%) with the window size equal to 300. The detailed accuracies for walk, bike, car, bus, train, and subway are shown in Table 4.

### 2) PERFORMANCE EVALUATION OF THE SEGMENT AND PEAK FEATURES

In this section, an experiment is conducted to verify the feasibility of the segment and peak features for transportation modes. Fig.10 illustrates the detection results of

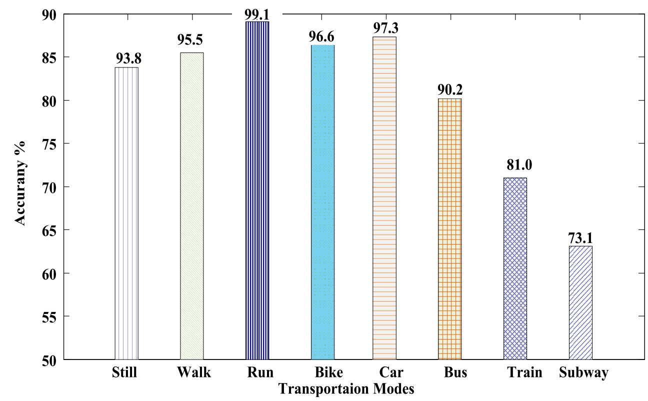



**FIGURE 10.** The identification accuracy of transportation modes only using CNN-extracted features and without using segment and peak features.

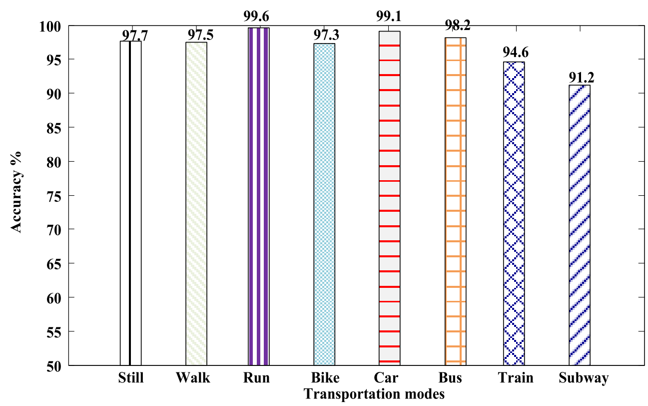



**FIGURE 11.** The identification accuracy of transportation modes using CNN-extracted features and handcrafted segment and peak features.

transportation modes by only using the features without segment and peak features. As a comparison, Fig. 11 illustrates the detection results of transportation modes by both using the CNN-extracted and the handcrafted features. From these two figures, we can find that both using the CNN-extracted and the handcrafted features can get better accuracy than that only using the CNN-extracted features. By adding the handcrafted segment and peak features, the metro and train identification accuracy was dramatically improved. The accuracy of train increases from 81.0% to 94.6%, and the accuracy of subway increases from 73.1% to 91.2%. It confirms that the handcrafted segment and peak features are efficient to represent the transportation modes over a much long period

**TABLE 5.** Different parameter configuration of the CL-TRANSMODE algorithm for evaluation.

| Parameter Group | First Conv Layer | Second Conv Layer | Third Conv Layer (after concatenating) | Pressure Conv | Pooling Layer MaxPool |
|---|---|---|---|---|---|
| a | kernel_num= 64, kernel_size = 2 | kernel_num= 128, kernel_size = 3 | kernel_num= 32, kernel_size = 3 | kernel_num=128, kernel_size = 3 | pool_size=2, stride = 2 |
| b | kernel_num=128, kernel_size = 3 | kernel_num=128, kernel_size = 3 | kernel_num=32, kernel_size = 3 | kernel_num=128, kernel_size =3 | pool_size=2, stride = 2 |
| c | kernel_num=128, kernel_size = 3 | kernel_num=128, kernel_size =3 | kernel_num=32, kernel_size = 3 | kernel_num=128, kernel_size = 3 | pool_size=3, stride = 3 |
| d | kernel_num=64, kernel_size = 4 | kernel_num=128, kernel_size = 3 | kernel_num=32, kernel_size = 3 | kernel_num=128, kernel_size = 3 | pool_size=3, stride = 3 |
| e | kernel_num=128, kernel_size = 2 | kernel_num=128, kernel_size = 2 | kernel_num=32, kernel_size = 3 | kernel_num=128, kernel_size = 3 | pool_size=3, stride = 3 |
| f | kernel_num=64, kernel_size = 3 | kernel_num=128, kernel_size = 3 | kernel_num=32, kernel_size = 3 | kernel_num=128, kernel_size = 3 | pool_size=2, stride = 2 |

**TABLE 6.** The recall rates of recognizing different transport modes using different super parameters.

| Parameters | Still | Walk | Run | Bike | Car | Bus | Train | Subway | Accuracy |
|---|---|---|---|---|---|---|---|---|---|
| a | 95.9% | 95.4% | 99.0% | 97.2% | 97.2% | 94.8% | 84.0% | 81.3% | 93.1% |
| b | 91.6% | 97.0% | 98.9% | 96.0% | 96.6% | 91.4% | 78.5% | 78.6% | 91.0% |
| c | 95.0% | 97.7% | 99.0% | 96.7% | 98.6% | 95.2% | 87.6% | 81.7% | 93.9% |
| d | 93.2% | 96.1% | 98.3% | 96.9% | 97.4% | 95.6% | 89.4% | 78.7% | 93.3% |
| e | 91.1% | 96.2% | 99.2% | 96.4% | 98.6% | 92.5% | 83.1% | 81.5% | 92.3% |
| f | 98.3% | 98.5% | 99.6% | 98.5% | 99.5% | 98.6% | 97.2% | 93.9% | 98.1% |

of time (minutes or hours) and help improve accuracy of transportation mode recognition.

### 3) PARAMETER OPTIMIZATION IN CL-TRANSMODE ALGORITHM

The hyper-parameters optimization is very important to improve the performance of the CL-TRANSMODE algorithm. We optimize the hyper-parameters of the convolution layers and the pool layers by trials. The detailed hyper-parameter configuration is listed in Table 5. The accuracies with different hyper-parameters are listed in TABLE 6. From TABLE 6, we can see that using the hyper-parameter of Group "f" can obtain the best accuracy.

Fig. 12(a)–(f) detail the confusion matrices of transportation mode recognition results using different hyper-parameters, respectively. From the five hyper-parameter configurations, we can see that among the eight transportation modes, the train and subway modes are prone to confuse with each other. That is caused by the similar driving pattern, i.e., smoothly and steadily running on the railroad tracks.

### 4) ACCURACY COMPARISON OF DIFFERENT ALGORITHMS ON SHL DATASET

Fig.13 compares the accuracy using our proposed CL-TRANSMODE algorithm and other methods on the SHL dataset. From Fig.13, we can see that using our proposed deep learning algorithms can obtain the best accuracy (98.1%), while only using the CNN or LSTM can only achieve 93.5% and 74.9% accuracy, respectively. This experimental result shows that by introducing the LSTM into our proposed algorithm, our proposed CL-TRANSMODE algorithm can learn time dependencies of sensor data and obtain higher accuracy than only using the CNN based algorithm.

It is worth to mention that using Adaboost and XGboost [27] algorithms can obtain better accuracies than other algorithms except for our proposed algorithm. The reasonable accuracies of Adaboost and XGBoost algorithms heavily rely on the labor-intensive feature extraction. For the transportation modes with larger shaking and remarkable feature differences, such as walking, running and bicycling, the XGBoost slightly outperforms the artificial neural networks algorithms. For the transportation modes with less shaking and similar features, such as car, bus, train and subway, artificial neural networks can reasonably differentiate them and obtain higher recognition accuracy. To recognize all the eight transportation modes with good balance and high accuracy, the CL-TRANSMODE algorithm pays a little cost with slight accuracy decrease of identifying walking, running and bicycling compared with the XGBoost. The XGBoost cannot accurately the transportation modes with smooth and stable movement, such as train and subway. On the contrary, the CL-TRANSMODE algorithm can learn deep features by CNN and LSTM layers to get good performance in transportation modes, especially in car, bus, train and subway

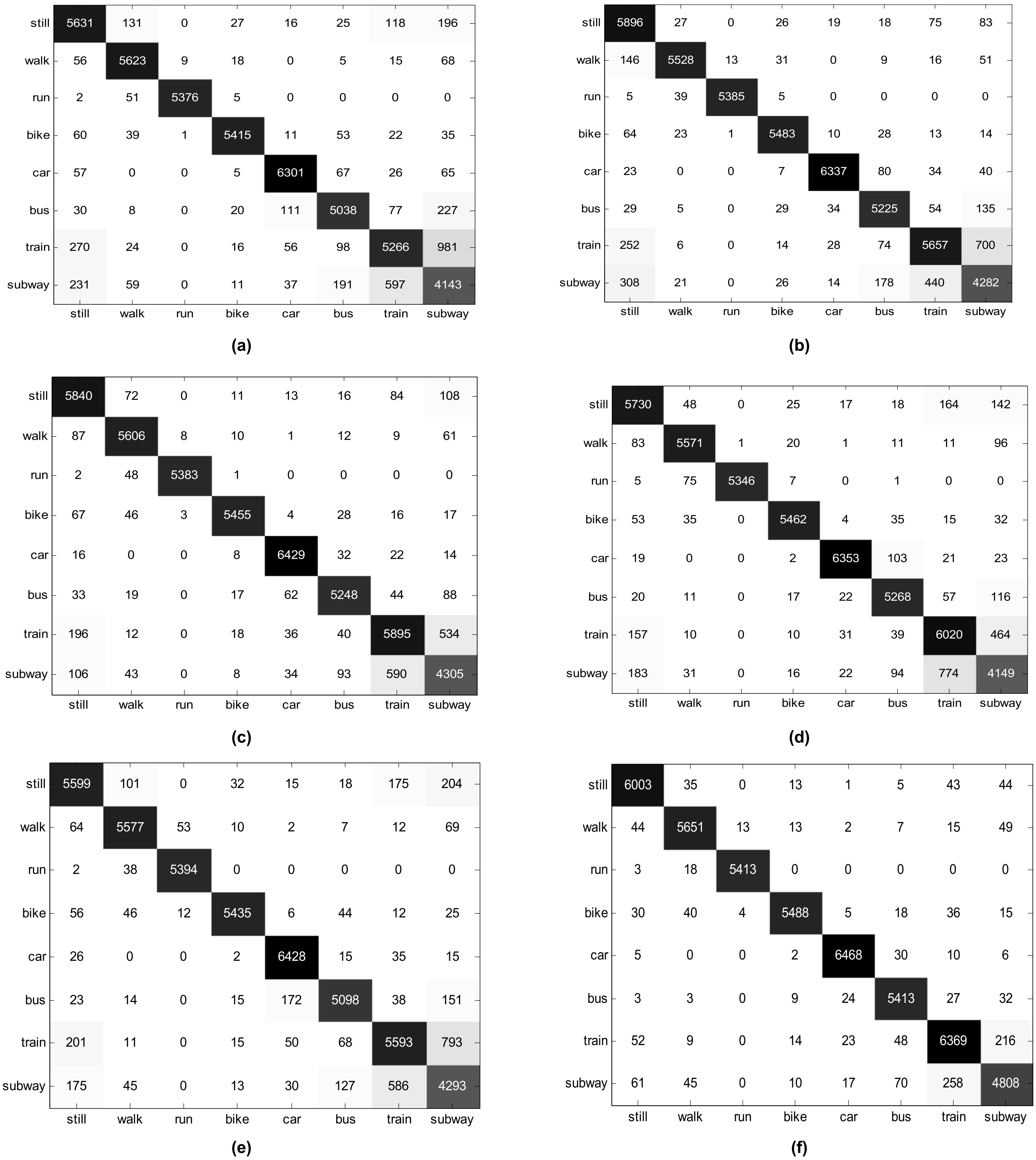



**FIGURE 12. Confusion matrixes of the CL-TRANSMODE algorithm using different super-parameter configurations.**

Experimental results show that using CL-TRANSMODE algorithm can obtain 98.1% recognition accuracy for the eight transportation modes and the recognition performance shows good robustness. It also demonstrates that extracting features by CNN is more effective than by hand-crafted method.

Fig. 14 shows the accuracy using different algorithms to identify eight types of transportation patterns. We find that CL-TRANSMODE algorithm can accurately recognize all the eight transportation patterns. Our proposed CL-TRANSMODE algorithm outperforms DNN, CNN, RNN, LSTM, Decision tree, Random Forest, Adaboost and XGBoost for identifying all the eight traffic modes.

Table 7 and Table 8 detail the main parameters of the ten algorithms, including the hyper-parameters in each layers of DNN, CNN, LSTM, RNN, CNN+RNN, ResNet[28],VGG Net[29], Google Net[30], CL-TRANSMODE, Decision Tree, Random Forest, Adaboost, and XGBoost. These hyper-parameters include dropout rate, hidden unit number, kernel number, kernel size, constitutive of Max Depth, Min Sample Split and Min Sample Leaf etc.

**TABLE 7.** Network configuration of the DNN/CNN/LSTM/RNN/CNN+RNN/CL-TRANSMODE.

| Mode | DNN | CNN | LSTM | RNN | CNN+ RNN | ResNet | VGG Net | Google Net | CL-TRANSMODE |
|---|---|---|---|---|---|---|---|---|---|
| Dropout Rate | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 |
| Full-Connected Network Hidden Unit Number | Layer1:128<br>Layer2:256<br>Layer3:512<br>Layer4:1024 | Layer1:128<br>Layer2:256<br>Layer3:512<br>Layer4:1024 | Layer1:128<br>Layer2:256<br>Layer3:512<br>Layer4:1024 | Layer1:128<br>Layer2:256<br>Layer3:512<br>Layer4:1024 | Layer1:128<br>Layer2:256<br>Layer3:512<br>Layer4:1024 | Layer1:1000 | Layer1:4096<br>Layer2:4096<br>Layer3:1000 | Layer1:128 | Layer1:128<br>Layer2:256<br>Layer3:512<br>Layer4:1024 |
| Convolution Layer Kernel Number | - | Layer1:64<br>Layer2:128<br>Layer3:32 | - | - | Layer1:64<br>Layer2:128<br>Layer3:32 | Layer1:64<br>Layer2:64<br>Layer3:128<br>Layer4:256<br>Layer5:512 | Layer1:64<br>Layer2:128<br>Layer3:256<br>Layer4:512<br>Layer5:512 | Layer1:32<br>Layer2:32<br>Layer3:64<br>Layer4:64<br>Layer5:80<br>Layer6:129 | Layer1:64<br>Layer2:128<br>Layer3:32 |
| Convolution Layer Kernel Size | - | Layer1:(3,1)<br>Layer2: (3,1)<br>Layer3: (3,1) | - | - | Layer1:(3,1)<br>Layer2: (3,1)<br>Layer3: (3,1) | Layer1:<br>(1,1)<br>Layer2:<br>(3,3)<br>Layer3:<br>(1,1) | Layer1:<br>(3,3,3)<br>Layer2:<br>(3,3,64)<br>Layer3:<br>(3,3,128)<br>Layer4:<br>(3,3,256)<br>Layer5:<br>(3,3,512) | Layer1-<br>Layer6:<br>(3,3) | Layer1：<br>(3,1)<br>Layer2:<br>(3,1)<br>Layer3:<br>(3,1) |
| Convolution Layer Stride | - | 1 | - | - | 1 | 2 | 1 | 2 | 1 |
| Activition Function | ReLU | ReLU | - | - | ReLU | ReLU | ReLU | ReLU | ReLU |
| MaxPooling Layer Size | - | Layer1:(2,1)<br>Layer2: (2,1)<br>Layer3: (2,1) | - | - | Layer1:(2,1)<br>Layer2: (2,1)<br>Layer3: (2,1) | Layer1:(3,3) | Layer1:(2,2)<br>Layer2:(2,2)<br>Layer3:(2,2)<br>Layer4:(2,2)<br>Layer5:(2,2) | Layer1:(3,3)<br>Layer2: (3,3) | Layer1:(2,1)<br>Layer2:<br>(2,1)<br>Layer3:<br>(2,1) |
| MaxPooling Layer Stride | - | 2 | - | - | 2 | 2 | 2 | 2 | 2 |
| RNN Hidden Unit Number | - | - | 128 | 128 | 128 | - | - | - | 128 |
| RNN Activation Function | - | - | tanh | tanh | tanh | - | - | - | tanh |

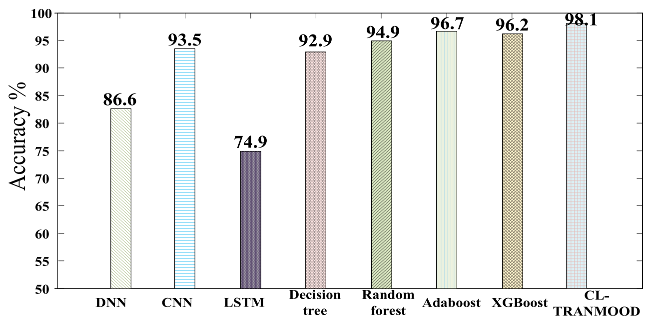



**FIGURE 13.** The accuracy comparison of transportation mode identification using different classification algorithms on the SHL dataset.

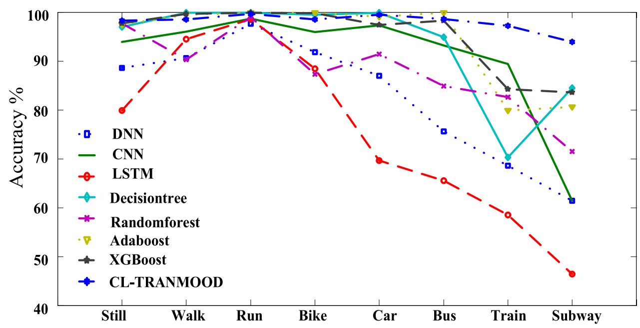



**FIGURE 14.** Accuracy comparison using different algorithms to identify eight types of transportation patterns.

The detailed accuracy of identifying each transportation mode using different algorithms are denoted in Table 9. The modes of ResNet, VGG Net, Google Net are by far the state-of-the-art convolutional neural network models. ResNet features special skip connections and a heavy use of batch normalization. The architecture is also missing fully connected layers at the end of the network. VGGNet introduces deep network to enhance performance. GoogleNet develops an inception module that dramatically reduced the number of parameters in the network). However, these modes are so deep that are not suitable for our temporal dependencies dataset.

### 5) EVALUATION OF CL-TRANSMODE SYSTEM'S PERFORMANCE ON DIFFERENT DATASETS

In this section, we perform a performance comparison on HTC dataset and SHL dataset to evaluate the generalization

**TABLE 8.** The super-parameters of decision Tree/Random Forest/Adaboost/XGBoost.

| Mode | Decisiontree | Random Forest | Adaboost | XGBoost |
|---|---|---|---|---|
| Decision Tree Criterion | Gini | - | - | - |
| Max Depth | 2 | 2 | 100 | 3 |
| Min Sample Split | 2 | 2 | 2 | - |
| Min Sample Leaf | 1 | 1 | 1 | - |
| Random Forest Estimator Number | - | 10 | - | - |
| Random Forest Criterion | - | Entropy | - | - |
| Adaboost min weight fraction leaf | - | - | 0 | - |
| Adaboost max_leaf_nodes | - | - | none | - |
| Xgboost Shrinkage ETA | - | - | - | 0.001 |

**TABLE 9.** The accuracy of identifying eight transportation modes using ten different algorithms.

| Mode | DNN | CNN | LSTM | RNN | CNN+ RNN | ResNet | VGG Net | Google Net | Decision tree | Random Forest | Adaboost | XGBoost | CL-TRANS MODE |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Still | 88.6% | 93.9% | 79.9% | 41.5% | 78.4% | 57.4% | 85.9% | 58.0% | 97.0% | 97.7% | 97.5% | 97.7% | 98.3% |
| Walk | 90.6% | 96.0% | 94.5% | 67.4% | 91.9% | 96.1% | 85.5% | 96.3% | 99.9% | 90.3% | 99.5% | 99.6% | 98.5% |
| Run | 97.6% | 98.7% | 98.6% | 71.1% | 97.5% | 95.3% | 96.9% | 97.1% | 99.9% | 99.0% | 99.9% | 99.9% | 99.6% |
| Bike | 91.8% | 95.9% | 88.4% | 60.4% | 88.9% | 88.2% | 89.4% | 91.8% | 99.5% | 87.3% | 99.9% | 99.8% | 98.5% |
| Car | 87.0% | 97.3% | 69.6% | 71.8% | 77.5% | 87.5% | 86.9% | 92.9% | 99.9% | 91.4% | 99.1% | 97.4% | 99.5% |
| Bus | 75.6% | 93.2% | 65.5% | 72.9% | 71.6% | 81.7% | 76.0% | 79.5% | 94.9% | 84.9% | 99.9% | 98.3% | 98.6% |
| Train | 68.6% | 89.4% | 58.5% | 66.8% | 74.1% | 55.5% | 69.4% | 53.6% | 70.3% | 82.6% | 79.9% | 84.2% | 97.2% |
| Subway | 61.4% | 61.4% | 46.4% | 61.2% | 71.1% | 47.4% | 58.6% | 51.4% | 84.5% | 71.5% | 80.6% | 83.6% | 93.9% |

**TABLE 10.** Performance of the CL-TRANSMODE algorithm on The SHL and the HTC datasets.

| | Mode | precision | recall | F-Score |
|---|---|---|---|---|
| Use Sussex-Huawei Locomotion Dataset Accuracy is 98.1% | Still | 98.3% | 98.3% | 98.3% |
| | Walk | 98.5% | 98.5% | 98.5% |
| | Run | 99.6% | 99.6% | 99.7% |
| | Bike | 98.5% | 98.5% | 98.6% |
| | Car | 99.5% | 99.5% | 99.4% |
| | Bus | 98.6% | 98.6% | 98.4% |
| | Train | 97.2% | 97.2% | 96.6% |
| | Subway | 93.9% | 93.9% | 95.1% |
| Use HTC Dataset Accuracy is 96.3% | Still | 97.8% | 97.9% | 97.9% |
| | Walk | 95.6% | 96.9% | 96.2% |
| | Run | 99.3% | 98.6% | 98.9% |
| | Bike | 97.2% | 96.8% | 97.0% |
| | Car | 96.4% | 97.2% | 96.8% |
| | Bus | 95.1% | 89.7% | 92.4% |
| | Train | 94.7% | 95.8% | 95.2% |
| | Subway | 94.0% | 93.9% | 94.0% |

ability of our proposed CL-TRANSMODE algorithm in recognizing transportation modes. Table 10 lists the classification performance of our proposed algorithm with precision, recall and F1-score. Overall, the precision on HTC dataset is slightly lower than that on the SHL dataset. The F-scores are more than 95% on the SHL dataset. However, the F-scores on the HTC dataset are about 4% lower than the F-scores on the SHL dataset. That is because the HTC dataset does not contain barometric data. In general, using our proposed CL-TRANSMODE algorithm can accurately identify transportation modes on these two datasets. In particular, the accuracy of recognizing still, walk, run, bike, car, and bus are excellent.

### 6) ACCURACY COMPARISON WITH THE STATE-OF-THE-ART METHODS

In this section, we compare the proposed CL-TRANSMODE model with several state-of-the-art methods. Those methods employ different structures and features. Slight modifications have been made to accommodate the test datasets to those methods and to ensure the fairness of these experiments.

**TABLE 11. Comparison with state-of-the-art transportation modes recognition methods.**

| Mode | DNN-based [16] | CNN [17] | CGRNN [18] | Mago [19] | DeepTransport [20] | Bi-LSTM [21] | CL-TRANS MODE |
|---|---|---|---|---|---|---|---|
| Still | 84.8% | 86.5% | 82.8% | 66.3% | 92.1% | 21.5% | 98.3% |
| Walk | 92.7% | 93.0% | 94.1% | 93.2% | 96.1% | 82.6% | 98.5% |
| Run | 94.1% | 98.4% | 98.8% | 94.5% | 98.9% | 98.0% | 99.6% |
| Bike | 87.7% | 91.1% | 93.6% | 86.4% | 94.9% | 87.3% | 98.5% |
| Car | 86.2% | 79.2% | 87.8% | 88.3% | 92.5% | 56.4% | 99.5% |
| Bus | 78.1% | 65.1% | 75.4% | 79.5% | 85.2% | 59.2% | 98.6% |
| Train | 67.3% | 57.3% | 66.8% | 69.4% | 80.1% | 34.9% | 97.2% |
| Subway | 61.1% | 38.1% | 53.4% | 68.2% | 75.4% | 42.5% | 93.9% |

These methods are trained by the same Adam optimizer and the same batch size.

- DNN-based transportation mode classification [16]: several features are extracted form the data of accelerator, gyroscope and magnetometer sensors. These features include average of the accelerometer's magnitude, standard deviation of the accelerometer's magnitude, average of the gyroscope's value, average of the gyroscope's value, average of magnetic instantly changes and so on. These features are feed into DNN model to recognize transportation modes. The DNN model is trained by 3 hidden layers. The accuracy is 81.6%.
- CNN transportation mode detection architecture [17]: only the raw data of accelerator are used. In the preprocessing stage, we use the similar data processing method as CL-TRANSMODE. The architecture includes six convolutional, six max pooling and four fully connected layers. This method achieves 76.2%.
- CGRNN [18]: The CGRNN method redesigns the typical recurrent unit by using a control gate and deleting the reset gate. Only the raw data of accelerator are used. CGRNN reach 81.7% accuracy.
- Mago [19]: For Mago, a two-layer model is designed, and the first layer uses the random forest (RF) model to recognize the mode of still. The maximum depth of RF is 8 and the leaf nodes is 100. The second layer is a neural network (NN) model with a hidden layer containing 64 hidden units. We extract 198 features from the raw data of magnetometer and accelerometer. These features are listed in the paper [19]. The transportation mode recognition accuracy of Mago reaches 87.9%.
- DeepTransport [20]: DeepTransport consists of 4 LSTM layers. The same raw data are used as the CL-TRANSMODE. The DeepTransport reaches 89.4% accuracy.
- Bi-LSTM transportation model detection architecture [21]: The Bi-LSTM model includes 3 Bi-LSTM layers and the hidden unit number is 64, 100 and 300, respectively. We test the Bi-LSTM model using the same raw data as the CL-TRANSMODE. The accuracy of Bi-LSTM is only 59.4%.

Table 11 compares the accuracy using different deep learning methods for transportation mode recognition. Among them, our proposed CL-TRANSMODE method works best especially for identifying the transportation modes of bus, car, train and subway. The results also show that the CL-TRANSMODE algorithm can accurately recognize all the eight transportation modes with reasonable balance.

**TABLE 12. The power consumption of smartphone on the tested stage (Huawei Mate 10).**

| operation on smartphone | Sensor data collection | Sensor data collection and transmission to the cloud |
|---|---|---|
| Power consumption | 289.7mW | 532.5mW |

### 7) ENERGY CONSUMPTION AND CALCULATION COMPLEXITY

We evaluate the energy consumption of CL-TRANSMODE system with two different stages. The first stage is the training phase and the second stage is the testing phase. On the training phase, the CL-TRANSMODE algorithm is run on a server. Since training phase is only needed to run once, its energy consumption is not an issue. On the test stage, the CL-TRANSMODE algorithm is run on the smartphone, which includes sensor data collection and sensor data transmission. The energy consumption is evaluated on a Huawei Mate 10 as Table 12 listed.

By only using lightweight sensors, our proposed CL-TRANSMODE algorithm is energy efficient for transportation identification. Considering that the deep learning platform will soon be supported on the smartphone, our algorithm will be run locally without need to send the sensor data to the cloud, which will decrease the power consumption of wireless transmission remarkably.

**TABLE 13. The calculation complexity of our system on the server (vCPUs: 2Core(s)).**

| operation on the server | Calculation time on the training stage | Transportation calculation time on the test stage |
|---|---|---|
| Time of period | 49800s | Less than 1ms |

We also evaluate the calculation complexity of our proposed algorithm on a server as Table 13 shows. Although it consumes much time on the training stage on large-scale training data, transportation calculation time on the test stage is less than 1ms, which can meet the real-time requisite for transportation identification application.

## V. CONCLUSION

In this paper, we proposed a unified framework composed of CNN and LSTM for recognizing complex transportation modes. Our algorithm takes advantage of the automatic feature extraction ability of CNN and the time-dependent feature extraction ability of LSTM. To further improve the accuracy of transportation mode identification, several handcrafted semantic features (i.e., segment and peak features) covering a much longer time interval are also used and concatenated with the features learnt by the CNN model. Extensive experimental results on two public large-scale datasets demonstrated that using our proposed CL-TRANSMODE algorithm can recognize eight transportation modes with 98.1% accuracy and significantly outperforms other state-of-the-art methods.

## REFERENCES

[1] S. Reddy, J. Burke, D. Estrin, M. Hansen, and M. Srivastava, "Determining transportation mode on mobile phones," in *Proc. 12th IEEE Int. Symp. Wearable Comput.*, Sep./Oct. 2008, pp. 25–28.
[2] X. Zhibin, W. Yang, F. Kun, and W. Fan, "Identifying different transportation modes from trajectory data using tree-based ensemble classifiers," *ISPRS Int. J. Geo-Inf.*, vol. 6, no. 2, pp. 57–79, Feb. 2017.
[3] A. Bolbol, T. Cheng, I. Tsapakis, and J. Haworth, "Inferring hybrid transportation modes from sparse GPS data using a moving window SVM classification," *Comput., Environ. Urban Syst.*, vol. 36, no. 6, pp. 526–537, Nov. 2012.
[4] S. Hemminki, P. Nurmi, and S. Tarkoma, "Accelerometer-based transportation mode detection on smartphones," in *Proc. ACM Conf. Embedded Netw. Sensor Syst.*, Nov. 2013, pp. 11–15.
[5] X. Su, H. Caceres, H. Tong, and Q. He, "Travel mode identification with smartphones," *Sensors*, vol. 15, no. 16, pp. 4871–4889, 2015.
[6] M. A. Shafique and E. Hato, "Classification of travel data with multiple sensor information using random forest," *Transp. Res. Procedia*, vol. 22, no. 22, pp. 144–153, Jan. 2017.
[7] T. Feng and H. J. P. Timmermans, "Comparison of advanced imputation algorithms for detection of transportation mode and activity episode using GPS data," *Transp. Planning Technol.*, vol. 39, no. 2, pp. 180–194, 2016.
[8] D. Cireşan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *Proc. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2012, vol. 4, no. 12, pp. 3642–3649.
[9] B. Shi, X. Bai, and C. Yao, "An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 11, pp. 2298–2304, Nov. 2017.
[10] Y. Gong, F. Zhao, S. Chen, and L. Haiyong, "A convolutional neural networks based transportation mode identification algorithm," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Sep. 2017, pp. 1–7.
[11] L. Dang-Nhac, N. Duc-Nhan, N. Thi-Hau, and N. Ha-Nam, "Vehicle mode and driving activity detection based on analyzing sensor data of smartphones," *Sensors*, vol. 18, no. 4, p. 1036, Mar. 2018.
[12] X. Zhou, W. Yu, and W. C. Sullivan, "Making pervasive sensing possible: Effective travel mode sensing based on smartphones," *Comput. Environ. Urban Syst.*, vol. 58, pp. 52–59, Jul. 2016.
[13] M. A. Guvensan, B. Dusun, B. Can, and H. Turkmen, "A novel segment-based approach for improving classification performance of transport mode detection," *Sensors*, vol. 18, no. 1, p. 87, Dec. 2017.
[14] X. Chang, W. Yingguan, B. Xinghe, and L. Fengrong, "Vehicle classification using an imbalanced dataset based on a single magnetic sensor," *Sensors*, vol. 18, no. 6, pp. 1690–1706, May 2018.
[15] A. M. Abdelaziz, "MonoSense: An energy efficient transportation mode detection system," in *Proc. 14th Annu. Int. Conf. Mobile Syst., Appl., Services Companion*, Sep. 2016, p. 3.
[16] S.-H. Fang, Y.-X. Fei, Z. Xu, and Y. Tsao, "Learning transportation modes from smartphone sensors based on deep neural network," *IEEE Sensors J.*, vol. 17, no. 18, pp. 6111–6118, Sep. 2017.
[17] X. Liang and G. Wang, "A convolutional neural network for transportation mode detection based on smartphone platform," in *Proc. IEEE 14th Int. Conf. Mobile Ad Hoc Sensor Syst. (MASS)*, Oct. 2017, pp. 338–342.
[18] T. H. Vu, D. Le, and J. C. Wang, "Transportation mode detection on mobile devices using recurrent nets," in *Proc. 24th ACM Int. Conf. Multimedia*, Oct. 2016, pp. 392–396.
[19] K.-Y. Chen, R. C. Shah, J. Huang, and L. Nachman, "Mago: Mode of transport inference using the Hall-effect magnetic sensor and accelerometer," *Proc. ACM Interact., Mobile, Wearable Ubiquitous Technol.*, vol. 1, no. 2, Jun. 2017, Art. no. 8.
[20] X. Song, H. Kanasugi, and R. Shibasaki, "Deep transport: Prediction and simulation of human mobility and transportation mode at a citywide level," in *Proc. Int. Joint Conf. Artif. Intell.*, Jul. 2016, pp. 2618–2624.
[21] H. Liu and I. Lee, "End-to-end trajectory transportation mode classification using Bi-LSTM recurrent neural network," in *Proc. 12th Int. Conf. Intell. Syst. Knowl. Eng. (ISKE)*, Nov. 2017, pp. 1–5.
[22] C. Saranya and G. Manikandan, "A study on normalization techniques for privacy preserving data mining," *Int. J. Eng. Technol.*, vol. 5, no. 3, pp. 2701–2704, Jun. 2013.
[23] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," in *The Handbook of Brain Theory and Neural Networks*, M. A. Arbib, Ed. vol. 3361, no. 10, 1998, pp. 255–258.
[24] M. Zeng, L. T. Nguyen, B. Yu, O. J. Mengshoel, J. Zhu, P. Wu, and J. Zhang, "Convolutional neural networks for human activity recognition using mobile sensors," in *Proc. 6th IEEE Int. Conf. Mobile Comput., Appl. Services (MobiCASE)*, Nov. 2014, pp. 197–205.
[25] H. Gjoreski, M. Ciliberto, L. Wang, F. J. O. Morales, S. Mekki, S. Valentin, and D. Roggen, "The University of Sussex-Huawei locomotion and transportation dataset for multimodal analytics with mobile devices," *IEEE Access*, vol. 6, pp. 42592–42604, 2018.
[26] M.-C. Yu, T. Yu, S.-C. Wang, C.-J. Lin, and E. Y. Chang, "Big data small footprint: The design of a low-power classifier for detecting transportation modes," *Proc. VLDB Endowment*, vol. 7, no. 13, pp. 1429–1440, 2014.
[27] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM Sigkdd Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 785–794.
[28] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.
[29] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2014, pp 1–14.
[30] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 2818–2826.

**YANJUN QIN** is currently pursuing the Ph.D. degree with the School of Software Engineering, Beijing University of Posts and Telecommunications, China. Her current main interests include location-based services, pervasive computing, convolution neural networks, and machine learning. She mainly involved in traffic pattern recognition-related project research and implementation.

**HAIYONG LUO** received the B.S. degree from the Department of Electronics and Information Engineering, Huazhong University of Science and Technology, Wuhan, China, in 1989, the M.S. degree from the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, China, in 2002, and the Ph.D. degree in computer science from the University of Chinese Academy of Sciences, Beijing, China, in 2008. He is currently an Associate Professor with the Institute of Computer Technology, Chinese Academy of Sciences (ICT-CAS), China. His main research interests include location-based services, pervasive computing, mobile computing, and the Internet of Things.

**FANG ZHAO** received the B.S. degree from the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China, in 1990, and the M.S. and Ph.D. degrees in computer science and technology from the Beijing University of Posts and Telecommunications, Beijing, China, in 2004 and 2009, respectively, where she is currently a Professor with the School of Software Engineering. Her research interests include mobile computing, location-based services, and computer networks.

**CHENXING WANG** is currently pursuing the master's degree with the School of Software Engineering, Beijing University of Posts and Telecommunications, China. His current main interests include pattern recognition, deep learning, and machine learning.

**JIAQI WANG** is currently pursuing the bachelor's degree with the School of Information and Communication Engineering, Beijing Information Science and Technology University, China. His current main interests include transportation mode recognition and machine learning.

**YUEXIA ZHANG** received the Ph.D. degree in computer science and technology from the Beijing University of Posts and Telecommunications, Beijing, China, in 2005 and 2008, respectively. She is currently a Professor with the School of Information and Communication Engineering, Beijing Information Science and Technology University, China. Her current research interest includes the key technologies of mobile communication.