Combining Residual and LSTM Recurrent Networks for Transportation Mode Detection Using Multimodal Sensors Integrated in Smartphones

Chenxing Wang, *Member, IEEE*, Haiyong Luo, *Member, IEEE*, Fang Zhao, and Yanjun Qin

***Abstract*— In recent years, with the rapid development of public transportation, the ways people travel has become more diversified and complicated. Transportation mode detection, as a significant branch of human activity recognition (HAR), is of great importance in analyzing human travel patterns, traffic prediction and planning. Though many works have been devoted to transportation mode detection, there remains challenge for accurate and robust transportation pattern identification. In this paper, we propose a residual and LSTM recurrent networks-based transportation mode detection algorithm using multiple light-weight sensors integrated in commodity smartphones. Feature representation learning is adopted separately on multiple preprocessed sensor data using deep residual and LSTM network, which can enhance the identification accuracy and support one or more sensors. Residual units are introduced to accelerate the learning speed and enhance the accuracy of transportation mode detection. Furthermore, we also leverage the attention model to learn the significance of different features and different timesteps to enhance the recognition accuracy. Extensive experimental results on three datasets indicate that using our proposed model can achieve the best recognition accuracy for eight transportation modes including being stationary, walking, running, cycling, taking a car, taking a bus, taking a subway and taking a train, which outperforms other benchmark algorithms.**

***Index Terms*— Context awareness, transportation pattern recognition, multimodal sensors, activity recognition.**

Manuscript received May 17, 2019; revised October 11, 2019 and February 13, 2020; accepted April 9, 2020. 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 through 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, in part 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 and Grant 61761038, in part by the Joint Research Fund for Beijing Natural Science Foundation and Haidian Original Innovation under Grant L192004, in part by the Key Research and Development Project from Hebei Province under Grant 19210404D, and in part by the Open Project of the Beijing Key Laboratory of Mobile Computing and Pervasive Device. The Associate Editor for this article was X. Si Ma. *(Corresponding authors: Haiyong Luo; Fang Zhao.)*

Chenxing Wang, Fang Zhao, and Yanjun Qin are with the School of Software Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: wangchenxing@ict.ac.cn; zfsse@bupt.edu.cn; qinyanjun@ict.ac.cn).

Haiyong Luo is with the Research Center for Ubiquitous Computing Systems, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 830011, China (e-mail: yhluo@ict.ac.cn).

Digital Object Identifier 10.1109/TITS.2020.2987598

I. Introduction

IN RECENT years, smartphones have gradually become a necessity in people's daily life. Equipped with variety of sensors, smartphones can provide plentiful sensor data. These data are able to obtain more practical information beyond the meaning of data itself [1], such as human activity recognition (HAR). By identifying users' daily activities with HAR technology, smartphones begin to become an assistant tool to provide various intelligent services.

Transportation mode detection (or transportation pattern identification), as an important branch of HAR, aims at identifying the current transportation mode of the user. Transportation mode detection can be used in many application scenarios, such as traffic prediction and planning, the carbon footprint calculation [2], the population flow pattern mining [3] and wireless network optimizations [4].

The common transportation modes include: being stationary, walking, cycling, motorcycling, taking a car, taking a bus, taking subways and taking a train. To achieve a high level of recognition accuracy, many studies have aggregated vehicle-related models into one category. In this way, the transportation modes are simplified into four groups: being stationary, walking, cycling and taking vehicles [5]. Different from this simplified transportation classification, in this paper, we still use the fine-grained transportation modes: being stationary, walking, running, cycling, taking a car, taking a bus, taking a subway and taking a train, which can provide more information for the practical applications.

Many researches have been conducted on transportation mode detection (Refer to Section II). Most work employs the hand-crafted features from GPS or motion sensors with classic machine learning algorithms to identify transportation modes [3], [6]–[9]. Some work attempts to use Wi-Fi signals or sound data for transportation mode detection. Recently, several researchers introduced the deep learning technology into the transportation mode detection, such as recurrent neural network [10], convolutional neural network [11], long-short-term memory recurrent neural network [12] and their combination (e.g., Deep Convolutional LSTM Network [13]). Nevertheless, existing researches are only evaluated on small dataset with less transportation modes, and the accuracy cannot meet the application requirements.

The main contributions of this paper are summarized as follows:

(1) We propose a deep learning framework based on residual and LSTM for transportation mode detection (called MSRLSTM) using multiple light-weight sensors integrated in smartphones. The feature learning for individual sensors separately enhances the accuracy of transportation pattern identification and increases the scalability of the MSRLSTM for different combination of multiple sensors.

(2) We adopt the residual techniques to the transportation feature representation learning. With the introduction of residual units into the deeper neural networks, the training process is accelerated and the accuracy of transportation pattern recognition is improved.

(3) We leverage the attention model to learn the significance of different features and different timesteps from the LSTM based networks, which assigns larger weights for more important features and timesteps. Better recognition accuracy for the similar transportation modes is achieved, such as train and subway.

(4) Our proposed algorithm is evaluated on two large public datasets, i.e., SHL dataset [14] and HTC dataset [15], as well as our collected data. Experimental results demonstrated that the MSRLSTM is obviously superior to the baseline algorithms including MLP, CNN, RNN, LSTM, DeepConvLSTM, Decision Tree [16], Random Forest [17], and Adaboost [18] based transportation mode recognition method [19]. The reasonable scalability of the MSRLSTM is also confirmed.

The remaining of this paper is organized as follows: Section II reviews some related works for transportation mode detection using different sensor data and machine learning or deep learning methods. Section III introduces the architecture of the MSRLSTM algorithm, followed by the design details of the proposed algorithm. Section IV describes the experimental setup and datasets for evaluation. Then, the evaluation results are presented and discussed in this section. Finally, Section V concludes this work.

II. Related Work

In recent years, many works have been conducted on transportation mode detection from various perspectives. Several researchers use the GPS data for transportation mode detection. Feng Tao *et al.* [3] adopted Bayesian Belief Network to classify eight transportation modes (walking, cycling, running, motorcycling, taking bus, car, metro and tram). Three approaches, i.e., using GPS only, using accelerometer only, and using GPS with accelerometer were evaluated. Stenneth *et al.* [9] also used GPS data to identify stationary, bike, walk, car, bus and train with random forest algorithm and obtained 93.7% accuracy on approximately six hours GPS data. Though using GPS data can obtain reasonable accuracy, the high-power consumption and non-availability in the GPS-denied environments limit its wide usage.

To identify transportation with low-power consumption, some researchers resorted to light-weight sensors. Hemminki *et al.* [7] proposed a hierarchical classifier to distinguish multiple transportation modes including train, bus, stationary, metro, tram and car. Many hand-crafted features are extracted from accelerometer data. Online gravity estimation and peak-segment feature extracting methods are adopted to obtain better feature representations. To reduce the influence from different orientations of smartphones, vertical and horizontal features are extracted after removing the influence of gravity. After that, HMM is combined with Adaboost classifier to forecast transportation pattern. Ashqar *et al.* [6], [8] proposed a hierarchical classifier based on KNN, SVM, RF algorithms to identify five transportation patterns (i.e., cycling, taking a car, walking, running and taking a bus) using features of time and frequency domains. A same work conducted by Liang *et al.* [20], employed the magnitude of accelerometer observation vector for transportation pattern recognition. In general, the accuracy of these classic machine learning based transportation mode recognition methods strongly rely on the effectiveness of hand-crafted feature extraction. The hand-crafted feature extraction from original sensor data in most of these works may lead to heavy burden on computing devices or manpower. The accuracy and robustness of current transportation mode recognition methods using the light-weight sensors still cannot meet the application requirements.

Recently, considering the automatic feature learning and high representation ability of deep learning technology, some researchers attempted using deep learning to detect the transportation mode. Toan H *et al.* [10] introduced a control gate into typical recurrent neural network and adopted it for transportation pattern recognition using the magnitude of accelerometer observation vector. Dabiri S *et al.* [11] utilized a convolutional neural network to learn features from four-channel GPS segment, including speed, acceleration, jerk and bearing rate for transportation pattern recognition. Hong *et al.* [12] introduced a LSTM recurrent neural network, which combines feature learning on timestamps, latitudes and longitudes obtained from GPS sensors. Francisco *et al.* [13] presented a hybrid network combining a deep convolutional and an LSTM network for activity recognition. Four convolutional layers and two LSTM layers were adopted for feature learning with all the sensor data concatenated together to produce the predicted label for activity pattern. Chen, *et al.* [21] adopted a single-layer feedforward neural network with hand-crafted features to assist a deep LSTM network for human activity recognition. The deep LSTM network is not only used to learn temporal dependencies from raw sensory data, but also is used to learn from single-layer feedforward neural network to mimic how it generalizes. Other researches attempted using new sensors, such as Wi-Fi and sound signal for transportation mode recognition. Kalatian *et al.* [22] designed a residual network using Wi-Fi signals obtained from smartphones for transportation mode detection. Instead of using convolutional layers as the weight layers in residual unit, they chose to use fully connected layers for identification. Three transportation modes including walking, cycling and driving are fed into the proposed network for classification. The total accuracy of mode detection reaches 84.1%. Chen, *et al.* [23] adopted the attention based BLSTM with Wi-Fi CSI data to distinguish different activities including lie down, fall, walk, run, sit down and stand up. An attention model based BLSTM network is

adopted for feature learning and it can achieve an average recognition accuracy of 97%. Wang *et al.* [24] adopted sound data in SHL dataset to recognize eight transportation modes using several baselines including classic machine learning, i.e. NB, KNN, DT, RF and SVM, and deep learning techniques using two convolutional layers and three fully connected layers. Both of them use sound data preprocessed by STFT and normalization techniques to reduce the environmental noises. The accuracy reaches approximately 86.6% of all the eight transportation modes. However, the Wi-Fi or sound data-based method may suffer from the availability of Wi-Fi network and consume more power.

Our previous work [25] combines the handcrafted features (i.e., peak and segment features) and the automatically-learnt features from CNNs and LSTMs for travel pattern recognition. Although we obtained reasonable evaluation results, there are still some challenges: (1) a pre-defined threshold is required to calculate peak and segment features, which limits the generalization ability. (2) A typical peak and segment feature extraction task require a much longer sliding window (about 120 seconds) than the sliding window adopted in MSRLSTM (no more than 4.5 seconds). It would be better when less predicting time is required for once transportation mode prediction in practical applications. (We published our proposed MSRLSTM algorithm at the following website: https://github.com/morningstarwang/TMDMobileNG). (3) It takes much longer time to train the model in our previous work, which is caused by the peak and segment feature extraction. Dataset in real world is frequently updated overtime and it is meaningful to decrease the training and predicting time for mode recognition. The training time decreases from 49800 seconds in our previous work to 37890 seconds in our proposed MSRLSTM in this work, which does not contain the peak and segment feature extraction task.

Different from the aforementioned methods, in this paper, we proposed a residual attention-based LSTM for transportation mode detection using multiple light-weight sensors integrated in commodity smartphones. The introduction of residual unit into deep neural network not only accelerates the training process [26], but also reduces the degradation influence of multiple hidden layers. We leverage the attention scheme to learn the significance of different feature vectors and timesteps from LSTM based network to enhance the recognition accuracy for distinguishing train and subway transportation modes using raw sensor data only compared to the previous work. The recognition accuracy for train and subway increases from 81.0% and 73.1% (previous work without peak and segment features) to 97.04% and 96.08%.

III. Algorithm

A. Architecture

The proposed deep learning algorithm for transportation mode detection includes three main steps: data preprocessing, MSRLSTM model training, and transportation mode predicting.

In the data preprocessing step, all datasets are transformed into uniform matrices, and then are processed with batch normalization and segmentation using a sliding window to reduce the influence of environmental noises. In the MSRLSTM model training step, preprocessed sensor data from the selected datasets are used as training data and fed into the MSRLSTM model to train the model. To speed up the feature learning process and reduce the degradation influence of multiple hidden layers, convolutional operations in residual units are applied to each element of the individual sensor immediately after preprocessing. Then the feature representations of each element coming from the same sensor are merged into one tensor, and a convolutional operation is applied on each merged feature representation. All these merged features are concatenated together and fed into LSTM layer for feature learning with time dependencies. Then, an attention model is adopted to learn the importance of different features and timesteps from the LSTM layer. At last, representations with time dependencies are passed through several fully connected layers and finally produce the transportation mode estimation using Softmax function. In the transportation mode predicting step, testing data separated from the selected dataset are adopted to evaluate the performance of our proposed MSRLSTM model using accuracy, precisions, recalls and F1-scores evaluation indices.

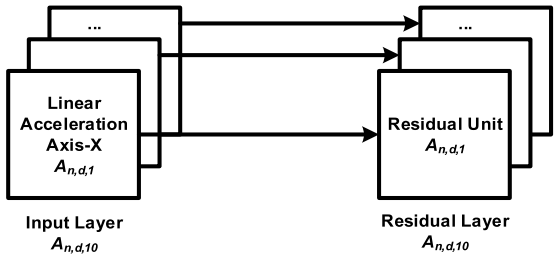

Fig. 1. Input layer structure.

B. MSRLSTM Model for Transportation Mode Detection

In this section, we detail the architecture of our proposed MSRLSTM model. The overall architecture of our proposed MSRLSTM model is shown in Fig. 1.

1) Multimodal Input Layer: All preprocessed sensor data are fed into the model from the input layer and defined as Tensor $\boldsymbol{A}_{n,d,k}$, where n is the number of total samples, d is the length of the selected sliding window, and k is the total element number of all sensors. In this paper, $k = 10$ represents linear acceleration axis-X, Y, Z, gyroscope axis-X, Y, Z, magnetometer axis-X, Y, Z, and barometric pressure. d represents the length of the selected sliding window (450 samples corresponding to 4.5s sampling period with 100Hz sampling frequency). And then $\boldsymbol{A}_{n,d,10}$ is transformed into ten tensors $\boldsymbol{A}_{n,d,1}$ and they are fed into the residual layer as Fig.2 depicts.

2) Residual Layer: The residual layer consists of ten residual units. As the first feature learning on the sensor data, the residual layer influences the trend of feature learning of the entire network. The advantages of using residual network is that it can speed up the training process and improve the accuracy. Each of them is fed with the tensor from input layer: $\boldsymbol{A}_{n,d,1}$. The convolutional and max-pooling [27] operations are

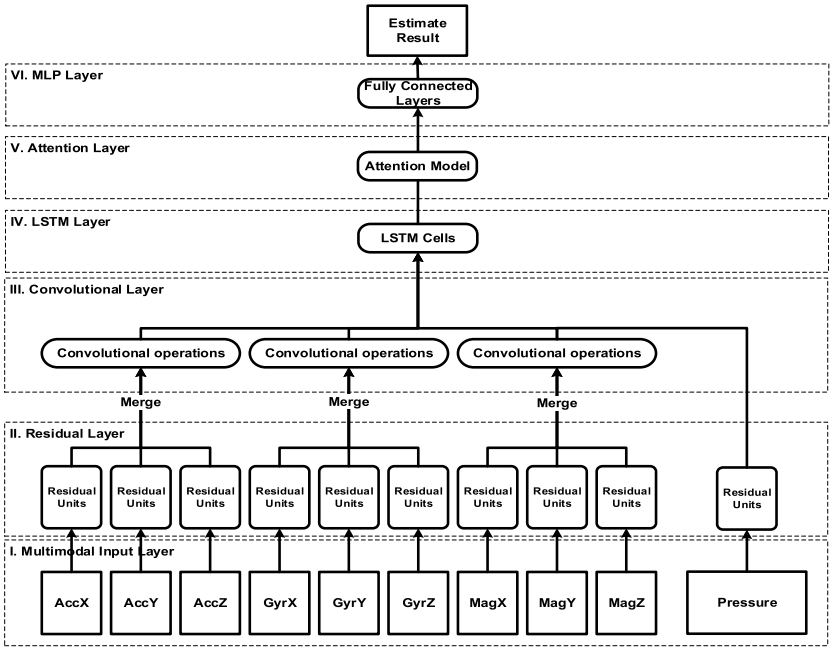

Fig. 2. The architecture of our proposed MSRLSTM model.

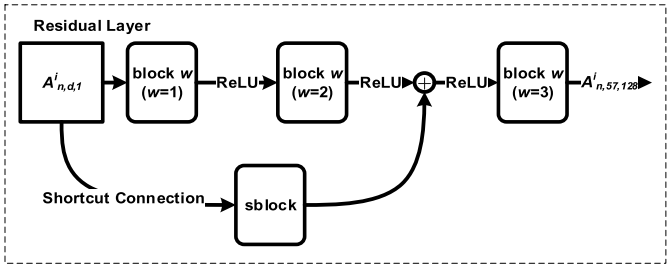

Fig. 3. The i^{th} residual unit architecture.

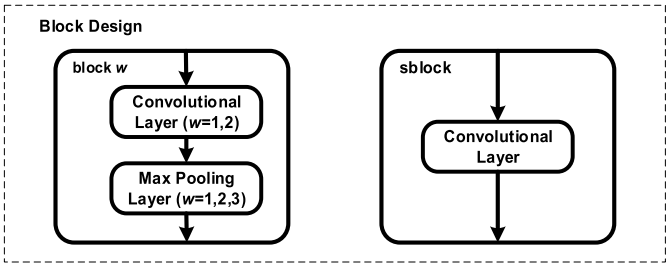

Fig. 4. The block design for residual unit.

defined as the follows. In this paper, the output of the v^{th} feature map on the u^{th} unit of the l^{th}convolutional layer is defined as Equation (1) shows.

$$conv_u^{l,v,p}\left(\boldsymbol{x}_u^{l-1}\right) = f\left(\sum_{c=1}^{p} \boldsymbol{w}_c^v \boldsymbol{x}_{u+c-1}^{l-1} + \boldsymbol{b}_v\right) \quad (1)$$

where f is the activation function ReLU [28], p is the kernel size, $\boldsymbol{w}_c^v$ is the weight of the v^{th}feature map and c^{th} filter, and $\boldsymbol{b_v}$ is the bias of the v^{th}feature map. The chosen activation function can obviously accelerate the training process [29]. The output of the v^{th}feature map on the u^{th}unit of the l^{th}max-pooling layer is$mp_u^{l,v}\left(\boldsymbol{x}_u^{l-1}\right)$, in which the max pooling method is applied to down-sample the input representation and the kernel size of max pooling is set to 2 with stride equal to 2.

Several blocks are designed for the residual network and the i^{th} residual unit architecture is shown in Fig.3, where block$w$ (w= 1, 2, 3) is designed to learn features for transportation mode detection, and sblock is designed to transform the shape of input tensor for short connection.

As illustrated in Fig.4, these blocks comprise of several convolutional or max pooling layers.

According to the descriptions above, the output of the i^{th} residual unit is defined as follows.

$$\boldsymbol{A}_{n,57,128}^{i} = block w_3 \quad (2)$$

$$sblock = conv_{u_s}^{1,v_s,p_s}\left(\boldsymbol{A}_{n,d,1}^{i}\right) \quad (3)$$

$$block w_1 = mp_{u_1}^{1,v_1}\left\{f\left[conv_{u_1}^{1,v_1,p_1}\left(\boldsymbol{A}_{n,d,1}^{i}\right)\right]\right\} \quad (4)$$

$$block w_2 = mp_{u_2}^{1,v_2}\left\{f\left[conv_{u_2}^{1,v_2,p_2}\left(block w_1\right)\right]\right\} \quad (5)$$

$$block w_3 = mp_{u_3}^{1,v_3}\left[f\left(block w_2 + sblock\right)\right] \quad (6)$$

TABLE I
THE DETAILED PARAMETERS FOR RESIDUAL LAYER DEFINITIONS

Parameter	Value
w_1, w_2, w_3	1, 2, 3
v_s, v_1, v_2, v_3	1, 2, …, 128; 1, 2, …, 64; 1, 2, …, 128; 1, 2, …, 128
p_s, p_1, p_2	4, 3, 3
u_s, u_1, u_2, u_3	1, 5, …, $\left\lceil\frac{d}{4}\right\rceil$; 1, 2, …, d; 1, 2, …, $\frac{d}{2}$; 1, 2, …, $\left\lceil\frac{d}{4}\right\rceil$

The detailed parameters in Equation (2-6) are listed in Table I.

3) Convolutional Layer: The CNN layer is used to reduce the size of data and speed up the training process and improve the model fitting. The convolution operation is performed again on the combined axis-X, Y, Z residual features of

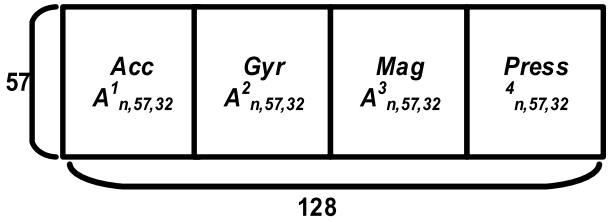

Fig. 5. The output of the CNN layer.

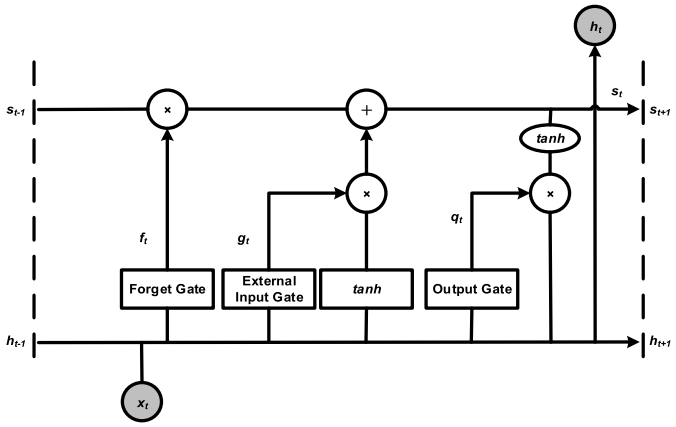

Fig. 6. The LSTM Cell Architecture [32].

all heterogeneous sensors including the barometric pressure. MSRLSTM model consists of four convolutional layers that correspond to feature learning operations of acceleration, gyroscope, magnetism and barometric pressure data. The initial inputs of the convolutional layer are four tensors defined as $\boldsymbol{A}^1_{n,57,384}$, $\boldsymbol{A}^2_{n,57,384}$,$\boldsymbol{A}^3_{n,57,384}$ and $\boldsymbol{A}^4_{n,57,128}$. And the output of the i^{th} convolutional layer is defined as Equation (7) shows,

$$\boldsymbol{A}^i_{n,57,32} = conv^{1,v_c,p_c}_{u_c}\left(\boldsymbol{A}^i_{n,57,384/128}\right) \tag{7}$$

where $v_c = 1, 2, \ldots, 32, p_c = u_c = 1, 2, \ldots, 57$.

Therefore, as illustrated in Fig.5, the output of the whole CNN layer is $\boldsymbol{A}^i_{n,57,128}$.

4) LSTM Layer: The LSTM [30] layer is adopted to further learn long-term feature of sensor data within a window. The LSTM cell [31] structure is selected for the MSRLSTM model. We attempt to analyze the data inflow and outflow relationship of the same LSTM cell at time step $t-1$, time step t, and time step $t+1$ because the LSTM layer contains time series data.

As illustrated in Fig.6, $\boldsymbol{x}_t$ is the input vector at time step t, and $\boldsymbol{h}_t$ is the hidden layer vector of time step t, which contains the outputs of all the LSTM cells.

With the control of the forget gate (f), external input gate (g), and output gate (q), long-term time dependencies of sensor data observation for the transportation pattern recognition can be reasonably learnt from LSTM layer. Further explanations for these gates are as follows: (1) forget gate calculates the weights and biases of $\boldsymbol{x}_t$ and $\boldsymbol{h}_{t-1}$ with sigmoid function to determine what features should be or not be forgotten; (2) external input gate, similar to forget gate, determines whether the inputs from $tanh$ channel shown in the bottom of Fig.8 should be included to update the state at time step t; (3) output gate, however, determines whether the updates of

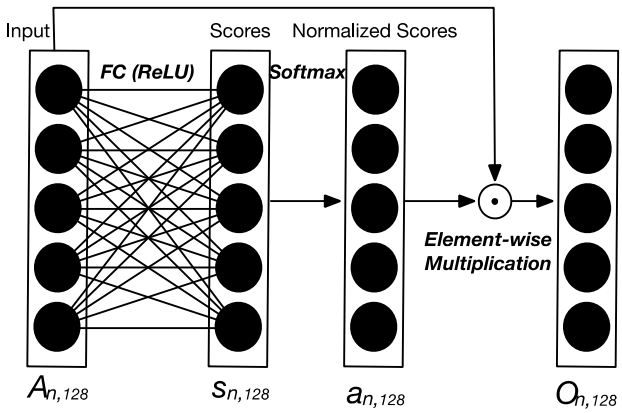

Fig. 7. The architecture of the attention layer.

outputs at time step t ought to be delivered to next cell of this layer.

For our proposed MSRLSTM model, the number of LSTM cell is 128, and the final output of the LSTM layer with time step 57 will be$\boldsymbol{A}_{n,128}$. These optimal parameter values are obtained based on practical experiments.

5) Attention Layer: As is shown in Fig. 7, the attention layer is adopted to learn the importance of features and time steps from the LSTM layer and assign larger weights to more important features and time steps for better recognition performance [23].

The learnt sequential features obtained by the LSTM layer are adopted as inputs for attention layer, which is known as self-attention. The score $\boldsymbol{s}$to evaluate the importance of feature vector $\boldsymbol{A}_{n,i}$is defined as follows:

$$\mathrm{s_i} = Activation(\boldsymbol{W} \cdot \boldsymbol{A}_{n,i} + \boldsymbol{b}) \tag{8}$$

where $\boldsymbol{W}$ is the weight of the hidden layer and $\boldsymbol{b}$ is the bias. The activation function we choose here is the ReLU activation function. Once the score of each feature vector is obtained, we normalize it by applying softmax function and the normalized score $\boldsymbol{a}$ of each feature vector is:

$$\boldsymbol{a_i} = Softmax(s_i) \tag{9}$$

We apply element-wise multiplication to the learnt features and attention matrix, which modifies the feature matrix with the attention matrix. And the final output $O$of the attention model is shown as follow:

$$\boldsymbol{O} = \boldsymbol{a_i} * \boldsymbol{A}_{n,i} \tag{10}$$

where $i = 128$.

6) MLP Layer: The MLP layer is used for the last feature learning process from the LSTM layer output. Consisting of five fully connected networks, the MSRLSTM model can reach better accuracy than only using one layer with Softmax activation function. To avoid overfitting problem [33], Dropout [34] method is adopted during the MLP layer. The output of the i^{th} fully connected layer is defined as follows:

$$o_i = Activation(\boldsymbol{W} \cdot \boldsymbol{x} + \boldsymbol{b}) \tag{11}$$

where $\boldsymbol{W}$ is the weight of the hidden layer and $\boldsymbol{b}$ is the bias.

For our proposed MSRLSTM model, the number of units for all hidden layers are 128, 256, 512, 1024, 8, and the activation functions are: ReLU, ReLU, ReLU, ReLU and Softmax, respectively. Dropping out is done on all hidden layers except the last hidden layer. The probability of dropping out units in each hidden layer is set to 0.2 to avoid overfitting. Softmax is used as an activation function in the last hidden layer for output.

IV. Experimental Setup And Evaluation

In this section, we conducted extensive experiments to evaluate the performance of our proposed MSRLSTM model on three datasets. The details of dataset and experimental setup are introduced firstly. Then the performance of the MSRLSTM algorithm is evaluated compared to other state-of-art baseline algorithms. The detailed settings for several baseline algorithms are listed. The calculation complexity, i.e., the training time and prediction time of the MSRLSTM algorithm and the comparative algorithms are also evaluated. Lastly, we discussed the impact of residual networks.

A. Datasets

We chose three datasets including two public datasets, i.e., SHL Dataset [14] and HTC Dataset [15], and our collected dataset (ICT Dataset) to evaluate the performance of the MSRLSTM algorithm. All sensor data in these datasets are transformed into $n \times m$ matrixes before being fed into the MSRLSTM model, where n represents the total number of collected samples, m represents the total element number of different sensor observations, i.e., $m = 10$ for SHL Dataset and ICT Dataset and $m = 9$ for HTC Dataset, which does not contain barometric sensor data. After all the data preprocessing steps as described in Section III(B) were thoroughly performed on selected SHL Dataset, HTC Dataset and ICT Dataset, the preprocessed data were separated into two parts: training part (80%) and testing part (20%).

- **SHL Dataset:** The SHL Dataset was collected over seven months in 2017 by three volunteers in the UK. Eight types of transportation modes are labelled during their daily traffic transfer. Each sample contains ambient light, temperature, GPS, Wi-Fi and motion data. These samples were collected with Huawei Mate 9 smartphones, which were placed in different positions, such as in bag, in hand, strapped to chest or in pocket. In this paper, to recognize transportation mode with low power consumption, only the light-weight sensor data (i.e., accelerometer, gyroscope, magnetometer and baroceptor) are used to evaluate our proposed MSRLSTM model. Considering most practical application, only the sensor data collected in hand and in pocket are selected for performance evaluation. We selected about 272 hours' sensor data to train and test the MSRLSTM model. These data were collected by a same volunteer over four months. All the sensor data were sampled at 100Hz. To make advantage of the time dependencies in our experiments, we rearranged the SHL data in time order.

TABLE II
The Distribution of HTC Dataset for Different Transportation Patterns

Transportation patterns	Internal Program	University Program (Selected Dataset)
Still	107h	1,750h
Walk	121h	1,263h
Run	61h	88h
Bike	78h	63h
Motorcycle	134h	1,683h
Car	209h	558h
Bus	69h	1,248h
Metro	95h	289h
Train	67h	267h
High Speed Rail	91h	72h

TABLE III
Detailed Configurations of Smartphones Used in ICT Dataset

Type	Configurations	Number of volunteers
Huawei Mate 8	Accelerometer: LSM6DS3 Gyroscope: LSM6DS3 Magnetometer: AKM09911 Barometric Pressure: BM1383	8
Huawei Mate 9	Accelerometer: LSM6DSM Gyroscope: LSM6DSM Magnetometer: AKM09911 Barometric Pressure: LPS22BH	11
Huawei Mate 10	Accelerometer: LSM6DSM Gyroscope: LSM6DSM Magnetometer: YAS537 Barometric Pressure: BMP380	2

- **HTC Dataset:** To evaluate the scalability of the MSRLSTM model, a large-scale dataset without barometric pressure called HTC Dataset is adopted. HTC Dataset have been collected since 2012. There are 8311 hours' data with 100GB, which were collected by 150 volunteers using HTC smartphones. Each sample contains accelerometer, gyroscope and magnetometer data. The distribution of the HTC dataset for different transportation pattern are listed in Table II. To keep in consistence among the three evaluation datasets, the data of motorcycle and high-speed rail in HTC Dataset were abandoned and different sensor data with timestamp difference less than 0.1 second were defined at the same time.
- **ICT Dataset:** To further evaluate the generalization ability of our proposed MSRLSTM algorithm, we also used our collected ICT Dataset, which was collected by twenty-one volunteers around eleven cities in China using three types of Huawei smartphones (See Table III for more detailed configurations). There are about 70 hours of sensor data in ICT Dataset. Each sample contains accelerometer, gyroscope, magnetometer and barometric pressure data. The sampling rate is also set to 100Hz. The distribution of our ICT Dataset is illustrated in Table IV.

TABLE IV
THE DISTRIBUTION OF ICT DATASET FOR DIFFERENT TRANSPORTATION PATTERNS

Transportation patterns	Number of samples (4.5 seconds per sample)
Still	12166
Walk	12162
Car	8485
Bus	14790
Metro	4838
Train	3462

B. Data Preprocessing

The raw sensor data contain various noises and errors, and the value ranges of different sensor data are different. To provide MSRLSTM model with clean and normalized data, dirty data removing, and data normalizing operations are conducted before being fed into the MSRLSTM model, which can effectively improve the training and predicting accuracy.

1) Dirty Data Removing: Considering that the size of public dataset is very large, we adopted a low-cost removing operation for those samples with incomplete sensor vector elements, i.e., a three-dimensional sensor data without one or two elements. For the small-scale dataset, interpolation processing may be more appropriate.

2) Normalization: To handle the large value range differences of heterogeneous sensor data, the Z-Score normalization operation [35] is applied to each element of sensor vector data as Equation (12) shows,

$$\boldsymbol{x}' = \frac{\boldsymbol{x} - \mu}{\sigma} \tag{12}$$

where μ is the average of each element of the sensor data, σ is the standard deviation of each element of the sensor data.

3) Sliding Window-Based Segmentation: It is important to select appropriate length of sensor data sequence for accurate transportation pattern recognition. Using small sequence length of sensor data will cause many recognition errors because feature representations for different transportation patterns cannot be accurately learnt when the length of the window is too short. Though using large sequence length of sensor data helps learn reasonable feature representations for different transportation patterns, too large length of sliding window will cause large delay for the first transportation pattern prediction and heavy burden of computation.

To obtain high accuracy with low computation cost and low prediction delay, we adopted a fixed length sliding window to segment the long-term sequences of sensor data into short-term sequences for feature learning. The optimal window length is selected based on the practical experiments.

Furthermore, to improve prediction accuracy and decrease prediction delay, partial overlapping between two successive windows is adopted as Fig. 8 illustrates. The window slides in the direction indicated by the top arrow. The solid black bar represents the tail of the previous window, the diagonal bar represents the window head (the latest data of the window). Three windows are illustrated in Fig. 8, namely theM^{th}, the$(M+1)^{th}$, and the$(M+2)^{th}$windows. Two window overlaps are formed during the sliding process. That is, the $(M+1)^{th}$window contains partial historical information of the M^{th} window, and the $(M+2)^{th}$window also contains partial historical information of $(M+1)^{th}$window.

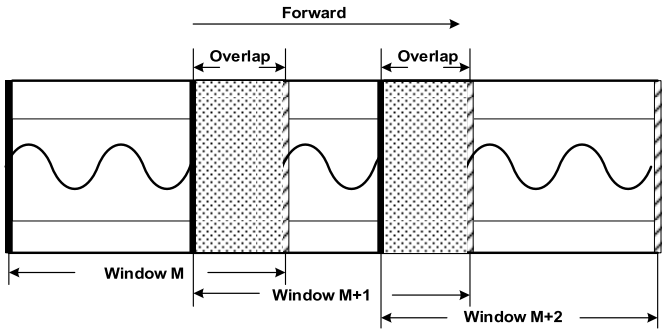

Fig. 8. Sliding window based sensor data segmentation.

C. Baselines and Benchmark

In additional to evaluate our proposed MSRLSTM model, as comparisons, we also implemented several other algorithms as baselines and benchmark, including classic machine learning algorithms (decision tree, random forest and Adaboost [18]), deep learning algorithms (CNN, LSTM, MLP, DeepConvLSTM [13] and DeepConvLSTM 2.0). Among these baselines, MLP, CNN and LSTM are parts of our proposed MSRLSTM model. Decision tree and random forest-based transportation mode recognition algorithms are implemented using Weka machine learning tools [36]. To evaluate the influence of adopting convolutional and pooling operations on each element of each sensor, we proposed an augmentation version of DeepConvLSTM (we called it DeepConvLSTM 2.0), which adopts convolutional and pooling operations on each element of each sensor first. For the DeepConvLSTM, the convolutional operations are performed on all concatenated elements of all sensors and there is no pooling operation. The detailed parameters of all baselines are listed in Table V.

We adopt four metrics to evaluate the proposed algorithm, i.e., model accuracy, precision, recall, and F1 Score which is defined in Equation (13).

$$\mathrm{F1} = 2^{*}\frac{Precision^{*}Recall}{Precision + Recall} \tag{13}$$

D. Experimental Setup

We used the Keras deep learning framework to train the MSRLSTM model with the preprocessed SHL dataset. Adam [37] optimizer and cross-entropy loss function are leveraged with learning rate=0.001, beta1=0.9, beta2=0.999. The parameter epoch is set to 100 and the batch size is set to 1024 samples. The feeding order is shuffled to avoid overfitting. We conducted the model training on a node of the Dawn supercomputer with GPU support. The detailed configuration of the node is listed in Table VI.

TABLE V
THE DETAILED PARAMETERS OF BASELINES

Name	Architecture
MLP	FC (128)-FC (256)-FC (512)-FC (1024)-Softmax
CNN	Each Element [C (64)-P (2)-C (128)-P (2)]-C (32)-P (2)-MLP-Softmax
LSTM	LSTM (128)-MLP-Softmax
DeepConvLSTM	C (64)-C (64) -C (64) -C (64)-LSTM (128)-Softmax
DeepConvLSTM 2.0	Each Element [C (64)-C (64) -P(2)]-LSTM (128)-Softmax
DT	Pruning confidence: 0.25, minimum number of instances per leaf: 2
RF	Size of bag: all training data, number of iterations: 100, number of attributes to randomly investigate: 0, minimum number of instances per leaf: 1, minimum variance for split
Adaboost	Number of estimators:50, learning_rate:0.001, algorithm: SAMME.R

Note: FC is fully connected layer; C is Convolutional 1D layer with kernel size:3, stride:1; P is MaxPooling 1D layer with kernel size:2, stride:2

TABLE VI
THE CONFIGURATION OF A NODE IN DAWN SUPERCOMPUTER

Name	Detail
CPU	Intel E5-2680 2.4GHz x 28
Memory	64GB
GPU	Tesla K80 12GB x 2
Operating System	CentOS 7.4
Python Environment	3.6.5
Development Framework	Keras

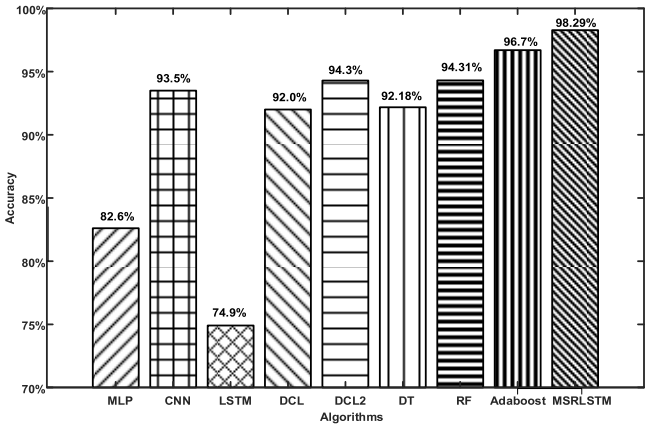

Fig. 9. The transportation recognition accuracy using different classification algorithmns on the SHL dataset.

E. Accuracy of Different Algorithms on Different Datasets

The experimental results on SHL dataset, HTC dataset and ICT dataset were illustrated in Fig. 9, Fig. 10, Fig. 11, Table VII, Table VIII and Table IX, respectively. We can observe that the proposed MSRLSTM algorithm significantly outperforms all the baselines. The accuracies of the MSRLSTM model achieve 98.29%, 96.22% and 98.44% on the SHL dataset, HTC dataset and ICT dataset, respectively. The recognition precisions of all eight transportation patterns are over 95%. The accuracies of CNN, DeepConvLSTM and Adaboost based algorithms are all over 90% as well. Nevertheless, these baselines cannot distinguish patterns between train and subway with high precisions. High-level features or time dependencies may not be learnt using these baselines.

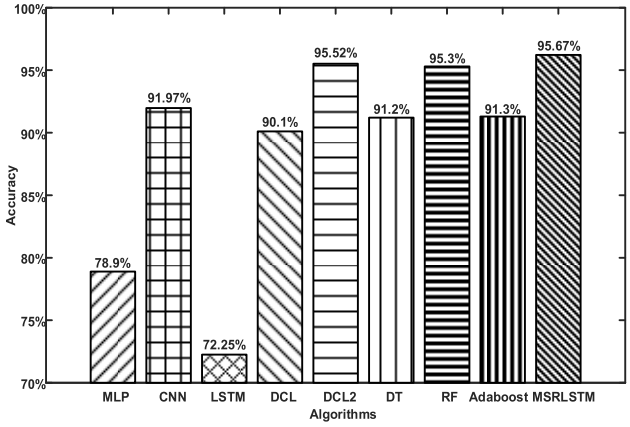

Fig. 10. The transportation recognition accuracy using different classification Algorithmns on the HTC dataset.

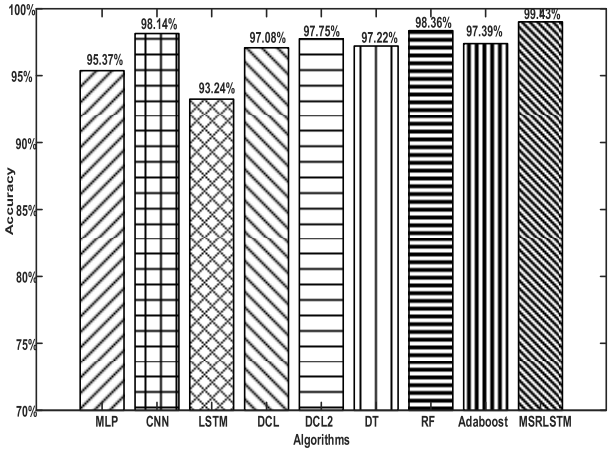

Fig. 11. The transportation recognition accuracy using different classification Algorithmns on the ICT Dataset.

The accuracy of recognizing train and subway using DeepConvLSTM [13] is lower than that using our proposed MSRLSTM, we think there are two reasons: (1) the approaches of dealing with the input sensors are different between DeepConvLSTM and our proposed MSRLSTM. Instead of learning features on each element of individual sensor at first, the DeepConvLSTM attempts to learn features from all concatenated elements of all sensors. The comparative experimental results of DeepConvLSTM and DeepConvLSTM 2.0 verified that learning features on each element of individual sensor outperforms learning features from all concatenated elements of all sensors, as Table VII shows; (2) DeepConvLSTM only

TABLE VII

THE PRECISIONS OF DIFFERENT TRANSPORTATION MODE RECOGNITION USING DIFFERENT ALGORITHMS ON THE SHL DATASET

	MLP	CNN	LSTM	DCL	DCL2	DT	RF	Adaboost	MSRLSTM
Still	88.60%	93.90%	79.90%	94.50%	94.24%	97.03%	97.84%	97.50%	98.64%
Walk	90.60%	96.00%	94.50%	96.80%	96.62%	97.51%	99.76%	99.50%	97.82%
Run	97.60%	98.70%	98.60%	98.90%	99.78%	99.31%	98.95%	99.90%	99.55%
Bike	91.80%	95.90%	88.40%	96.00%	98.16%	99.55%	99.43%	99.90%	99.03%
Car	87.00%	97.30%	69.60%	96.10%	97.57%	96.74%	99.71%	99.10%	99.52%
Bus	75.60%	93.20%	65.50%	92.50%	95.54%	99.77%	93.60%	99.90%	98.49%
Train	68.60%	98.40%	58.50%	82.80%	88.36%	60.87%	70.73%	79.90%	97.04%
Subway	61.40%	61.40%	46.40%	78.30%	84.23%	74.91%	83.72%	80.60%	96.08%

Note: DCL is short for DeepConvLSTM, and DCL2 is short for DeepConvLSTM 2.0; Number 1-8 represent Still, Walk, Run, Bike, Car, Bus, Train and Subway.

TABLE VIII

THE PRECISIONS OF DIFFERENT TRANSPORTATION MODE RECOGNITION USING DIFFERENT ALGORITHMS ON THE HTC DATASET

	MLP	CNN	LSTM	DCL	DCL2	DT	RF	Adaboost	MSRLSTM
Still	84.23%	94.24%	70.40%	93.24%	96.74%	98.03%	98.67%	98.31%	96.87%
Walk	87.18%	93.87%	88.16%	94.38%	95.55%	93.10%	94.93%	92.03%	95.70%
Run	97.28%	98.08%	97.18%	98.60%	98.54%	95.91%	97.38%	96.00%	98.21%
Bike	78.78%	93.53%	80.44%	95.20%	97.23%	87.05%	97.97%	87.51%	96.75%
Car	73.80%	91.86%	64.16%	89.89%	96.46%	87.95%	92.63%	87.72%	96.31%
Bus	56.26%	86.13%	76.42%	83.08%	93.27%	80.39%	89.72%	81.54%	92.08%
Train	66.66%	87.89%	47.99%	77.37%	94.49%	88.43%	94.58%	91.89%	94.60%
Subway	82.50%	88.51%	76.93%	86.73%	91.14%	92.96%	94.07%	93.04%	93.27%

Note: DCL is short for DeepConvLSTM, and DCL2 is short for DeepConvLSTM 2.0; Number 1-8 represent Still, Walk, Run, Bike, Car, Bus, Train and Subway.

TABLE IX

THE PRECISIONS OF DIFFERENT TRANSPORTATION MODE RECOGNITION USING DIFFERENT ALGORITHMS ON THE ICT DATASET

	MLP	CNN	LSTM	DCL	DCL2	DT	RF	Adaboost	MSRLSTM
Still	94.96%	98.75%	91.53%	97.49%	96.51%	97.73%	97.58%	97.46%	99.79%
Walk	98.49%	98.95%	97.78%	97.85%	99.52%	99.93%	99.32%	99.92%	99.83%
Subway	88.03%	91.88%	87.10%	87.85%	89.60%	82.69%	91.46%	86.54%	96.38%
Train	90.23%	95.30%	91.94%	98.51%	98.28%	92.59%	99.87%	92.34%	98.27%
Bus	95.83%	98.17%	91.87%	97.13%	99.31%	98.71%	98.10%	99.90%	99.50%
Car	94.96%	98.75%	91.53%	97.49%	96.51%	95.91%	97.58%	94.46%	99.72%

Note: DCL is short for DeepConvLSTM, and DCL2 is short for DeepConvLSTM 2.0; Number 1-8 represent Still, Walk, Run, Bike, Car, Bus, Train and Subway.

TABLE X

THE PRECISIONS OF DIFFERENT TRANSPORTATION MODE RECOGNITION USING DIFFERENT SENSORS ON THE SHL DATASET

	LA	LAG	LAGM	LAGMP
Still	77.91%	79.09%	88.64%	98.64%
Walk	96.42%	97.15%	96.65%	97.82%
Run	99.83%	99.74%	99.54%	99.55%
Bike	98.04%	98.51%	97.45%	99.03%
Car	94.34%	95.96%	97.85%	99.52%
Bus	89.52%	94.84%	95.77%	98.49%
Train	75.98%	86.09%	90.08%	97.04%
Subway	68.12%	77.62%	83.45%	96.08%

Note: LA is short for Linear Accelerometer, LAG is short for Linear Accelerometer + Gyroscope, LAGM is short for Linear Accelerometer + Gyroscope + Magnetometer and LAGMP is short for Linear Accelerometer + Gyroscope + Magnetometer + Barometric Pressure.

contains convolutional layers and LSTM layers. Instead of building the whole network with convolutional layers and LSTM layers only, the MSRLSTM is built with residual units containing convolutional and pooling operations, LSTM, attention layer and fully connected layers, which not only accelerates the training and predicting process, but also enhances the overall performance of transportation mode detection.

From Table VII, Table VIII and Table IX, we can see that the classic machine learning algorithms can accurately

TABLE XI
PRECISIONS, RECALLS AND F1-SCORES OF THE MSRLSTM MODEL ON THE SHL DATASET

	Precision	Recall	F1-Score
Still	98.64%	98.46%	98.55%
Walk	97.82%	98.69%	98.25%
Run	99.55%	99.61%	99.58%
Bike	99.03%	98.34%	98.69%
Car	99.52%	99.28%	99.40%
Bus	98.49%	98.46%	98.47%
Train	97.04%	97.17%	97.11%
Subway	96.08%	96.15%	96.12%

Still	13212	54	1	24	5	7	66	49
Walk	39	12431	35	27	0	3	15	46
Run	3	39	11741	4	0	0	0	0
Bike	41	64	17	11898	4	16	42	17
Car	9	1	0	8	14077	51	18	15
Bus	4	32	0	27	25	11776	44	52
Train	45	28	0	19	19	40	13984	256
Subway	41	59	0	7	15	64	241	10676

Fig. 12. The confusion matrix of the MSRLSTM on SHL dataset.

recognize most of the transportation patterns, i.e., still, walk, run, bike, car, bus. However, the recognition accuracy of train and subway is low, which indicates that the handcrafted feature may not well distinguish the train with subway patterns. On the contrary, using our proposed MSRLSTM algorithm can obtain reasonable representation of all the eight transportation modes and achieve the best accuracy among all the aforementioned baselines.

Furthermore, Table XI shows the precisions, recalls and F1-scores of the MSRLSTM model. The confusion matrix of the MSRLSTM Model on SHL Dataset, HTC Dataset and ICT Dataset are illustrated in Fig. 12, Fig. 13 and Fig. 14, respectively. To investigate the improvement made by using attention scheme, we also conduct ablation study on the MSRLSTM model without attention scheme. The evaluation results are shown in Table XII. We can observe that the precisions, recalls and F1-scores are improved by introducing the attention scheme. The main goal of using attention scheme is to learn the importance for different time steps and features. By "Score" the importance of each time step or feature vector, the metrics for distinguishing the train and subway transportation modes have been improved, which demonstrates the similar characterization in motion patterns [7].

The influence of the length and overlap ratio of the sliding window on the MSRLSTM model is evaluated, as illustrated in Fig.15. The proposed MSRLSTM algorithm achieves the

Still	5778	17	17	8	38	11	10	34
Walk	33	4892	22	40	37	21	3	31
Run	1	23	2359	3	0	0	0	1
Bike	10	36	0	3693	37	6	3	14
Car	38	61	3	30	9129	70	50	69
Bus	18	42	0	18	77	2185	18	78
Train	23	6	1	10	57	27	3100	91
Subway	64	35	0	15	104	53	93	4408

Fig. 13. The Confusion Matrix of the MSRLSTM on HTC dataset.

Still	2324	0	0	0	1	0
Walk	0	2372	0	0	1	0
Subway	1	0	639	5	9	3
Train	0	1	18	341	0	0
Bus	4	1	2	1	2774	1
Car	0	2	4	0	3	1423

Fig. 14. The confusion matrix of the MSRLSTM on ICT dataset.

TABLE XII
PRECISIONS, RECALLS AND F1-SCORES OF THE MSRLSTM MODEL WITHOUT ATTENTION SCHEME ON THE SHL DATASET

	Precision	Recall	F1-Score
Still	97.47%	98.29%	97.87%
Walk	98.25%	98.11%	98.18%
Run	99.66%	99.60%	99.63%
Bike	98.95%	98.05%	98.50%
Car	99.44%	99.27%	99.35%
Bus	97.15%	98.78%	97.96%
Train	96.77%	96.14%	96.45%
Subway	95.51%	95.00%	95.26%

best performance when the length and overlap ratio of the selected sliding window are set to 300 and 0.3. It can be seen that different sliding window sizes and overlap rates affect the recognition accuracy remarkably. Small sliding window size may not contain enough features learnt by the MSRLSTM model to distinguish travel patterns, while large sliding window may cause overfitting problems. The similar influences of different overlap rates also exist.

F. Discussion About Scalability and Generalization Ability

- **Scalability:** The plentiful configuration of sensors in commodity smartphones required reasonable scalability

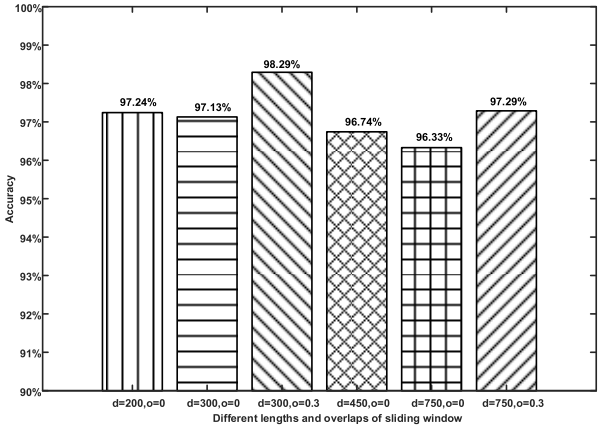

Fig. 15. Accuracy of different lengths and overlap ratios of sliding window using the MSRLSTM model on the SHL dataset.

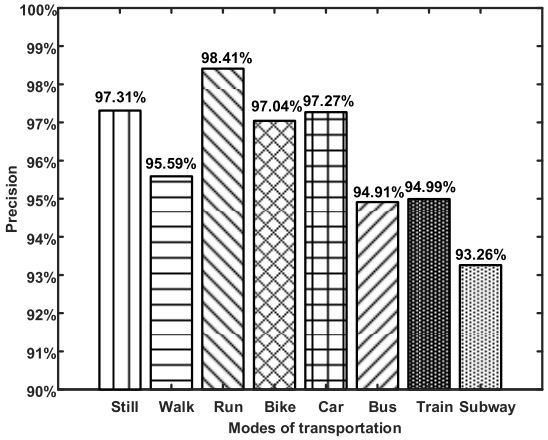

Fig. 16. The precisions of Different Modes using the MSRLSTM model on the HTC dataset.

of transportation recognition algorithm. The separate feature learning architecture of our proposed MSRLSTM model can easily fit flexible types of the input data. Considering that barometric sensor is not integrated in many commodity smartphones, such as Huawei P20, we evaluate the MSRLSTM model on the HTC Dataset by removing the barometric pressure related part of the whole model. The experimental results illustrated that using the MSRLSTM model still can obtain 96.22% accuracy, as Fig.16 depicts.

Furthermore, we investigated the recognition effect using different sensor variables to confirm the scalability of the MSRLSTM model. As is shown in Table X, LA, LAG, LAGM and LAGMP can achieve an average recognition accuracy of 87.33%, 90.83%, 93.67% and 98.29%, respectively. The accuracy of the model is improved by approximately 3% to 5% by adding a sensor. We can observe that the MSRLSTM may learn more features when more sensor variables are used, especially the barometric pressure sensor.

- **Generalization Ability:**To further evaluate the generalization ability of our MSRLSTM model, we also tested the MSRLSTM model on our collected ICT Dataset in addition to the SHL Dataset and HTC Dataset, as is illustrated in Fig.17. The precisions on six different

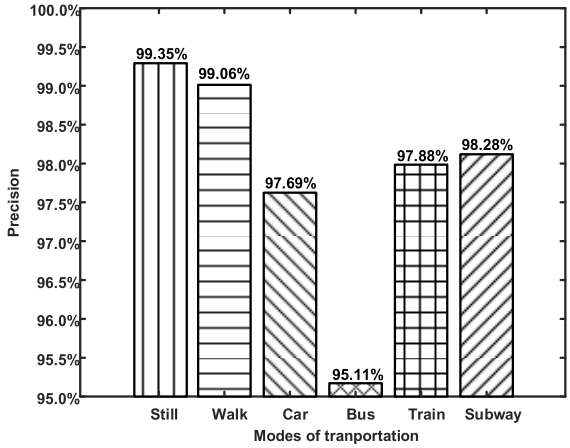

Fig. 17. The precisions of different modes using the MSRLSTM model on the ICT dataset.

TABLE XIII
PRECISIONS, RECALLS AND F1-SCORES OF DIFFERENT MODES USING THE MSRLSTM MODEL ON THE HTC DATASET AND THE ICT DATASET

Dataset	Mode	Precision	Recall	F1-Score
HTC Dataset	Still	96.87%	97.72%	97.29%
	Walk	95.70%	96.32%	96.01%
	Run	98.21%	98.83%	98.52%
	Bike	96.75%	97.21%	96.98%
	Car	96.31%	96.60%	96.46%
	Bus	92.08%	89.70%	90.87%
	Train	94.60%	93.51%	94.05%
	Subway	93.27%	92.37%	92.82%
ICT Dataset	Still	99.79%	99.96%	99.87%
	Walk	99.83%	99.96%	99.89%
	Car	96.38%	97.26%	96.82%
	Bus	98.27%	94.72%	96.46%
	Train	99.50%	99.68%	99.59%
	Subway	99.72%	99.37%	99.55%

transportation modes all exceed 95%. And the total accuracy is 97.92%.

More detailed evaluation results are shown in Table XIII. The results demonstrate that the scalability and generalization ability of the proposed MSRLSTM model are strong enough for flexible types of input data and datasets collected under different circumstances, which varies from the Europe to Asia. With the help of feature learning on separate sensor data, possibilities for more flexible input data are greatly increased. The residual network adopted in the MSRLSTM model also accelerates the training process of fitting a fresh new dataset in new regions around the world.

G. Calculation Complexity

To further analysis the calculation complexity, the training and predicting time are calculated as Table XIII depicts. The training time for all selected samples and predicting time for each sample are calculated on in SHL Dataset running on a computing node of the Dawn supercomputer and each sample contains data with 4.5 second window. The training time

TABLE XIV
THE TRAINING AND PREDICTING TIME IN DIFFERENT ALGORITHMS

Algorithm	Platform Type	Training Time	Predicting Time
MLP	GPU	1937s	0.11ms
CNN	GPU	19315s	0.22ms
LSTM	GPU	23467s	2.68ms
DCL	GPU	52132s	4.58ms
DCL2	GPU	49012s	10.5ms
DT	CPU	541.7s	0.04ms
RF	CPU	895.35s	0.07ms
Adaboost	CPU	7292s	121.7ms
MSRLSTM	GPU	37890s	0.68ms

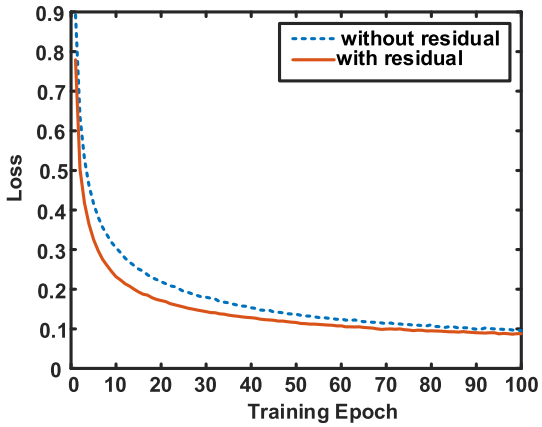

Fig. 18. The training loss value with different epoches during the training process.

for MLP, CNN, LSTM, DCL and MSRLSTM is calculated for one hundred epochs.

As Table XIV depicts, by introducing the residual convolutional layers into the LSTM layers and the MLP layers, the training time of MSRLSTM algorithm is less than that of DeepConvLSTM, which does not contain any residual unit. The reason of low calculation complexity of our MSRLSTM algorithm is introduction of residual networks and pooling operations.

H. Impact of Residual Networks

We introduced the residual networks into the MSRLSTM model to accelerate the training and predicting process and enhance the overall accuracy, which is confirmed by calculation complexity analysis and model convergence rate analysis.

The evaluation result of calculation complexity was shown in Table XIV. The training time and predicting time of DCL2, which represents the version of our model without residual networks, are 49012s and 10.5ms. However, after introducing the residual networks into the MSRLSTM model, the training time and predicting time are reduced to 37890s and 0.68ms, respectively.

The model convergence rate was also evaluated, as shown in Fig. 18. The convergence rate with residual networks is always faster than that without residual networks, which indicates that the convergence rate can be efficiently increased by introducing residual networks.

V. CONCLUSION

In this paper, we present a novel transportation mode detection algorithm, MSRLSTM. By combining the residual, LSTM and attention scheme networks, the accuracy of transportation pattern identification is improved, and the training process is accelerated. The MSRLSTM algorithm is also energy-efficient, which only uses multiple light-weight sensors integrated in smartphones to detect transportation mode. By learning features on individual sensors separately, the MSRLSTM can support more heterogeneous sensors to enhance the identification accuracy. Extensive experimental results on three datasets confirm that our proposed MSRLSTM model outperforms other baseline algorithms, including MLP, CNN, LSTM, DeepConvLSTM, Decision Tree, Random Forest, and Adaboost based transportation pattern algorithms.

In the future, a cloud-based Client/Server framework will be developed to provide robust, lightweight and accurate transportation mode predicting service using the pretrained MSRLSTM model and the generalization ability of the MSRLSTM model will be evaluated under more environments.

REFERENCES

[1] L. Runhem, "Resource efficient travel mode recognition," Ph.D. dissertation, School Comput. Sci. Commun., KTH Roy. Inst., Stockholm, Sweden, 2017.
[2] O. Lorintiu and A. Vassilev, "Transportation mode recognition based on smartphone embedded sensors for carbon footprint estimation," in *Proc. IEEE 19th Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2016, pp. 1976–1981.
[3] T. Feng and H. J. P. Timmermans, "Transportation mode recognition using GPS and accelerometer data," *Transp. Res. C, Emerg. Technol.*, vol. 37, pp. 118–130, Dec. 2013.
[4] S. Lee, J. Lee, and K. Lee, "VehicleSense: A reliable sound-based transportation mode recognition system for smartphones," in *Proc. IEEE 18th Int. Symp. A World Wireless, Mobile Multimedia Netw. (WoWMoM)*, Jun. 2017, pp. 1–9.
[5] M. Han, J. Bang, C. Nugent, S. McClean, and S. Lee, "A lightweight hierarchical activity recognition framework using smartphone sensors," *Sensors*, vol. 14, no. 9, pp. 16181–16195, 2014.
[6] H. I. Ashqar, M. H. Almannaa, M. Elhenawy, H. A. Rakha, and L. House, "Smartphone transportation mode recognition using a hierarchical machine learning classifier and pooled features from time and frequency domains," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 1, pp. 244–252, Jan. 2019.
[7] S. Hemminki, P. Nurmi, and S. Tarkoma, "Accelerometer-based transportation mode detection on smartphones," in *Proc. 11th ACM Conf. Embedded Networked Sensor Syst.*, 2013, p. 13.
[8] A. Jahangiri and H. A. Rakha, "Applying machine learning techniques to transportation mode recognition using mobile phone sensor data," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 5, pp. 2406–2417, Oct. 2015.
[9] L. Stenneth, O. Wolfson, P. S. Yu, and B. Xu, "Transportation mode detection using mobile phones and GIS information," in *Proc. 19th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.*, 2011, pp. 54–63.
[10] T. H. Vu, L. Dung, and J.-C. Wang, "Transportation mode detection on mobile devices using recurrent nets," in *Proc. ACM Multimedia Conf.*, 2016, pp. 392–396.
[11] S. Dabiri and K. Heaslip, "Inferring transportation modes from GPS trajectories using a convolutional neural network," *Transp. Res. C, Emerg. Technol.*, vol. 86, pp. 360–371, Jan. 2018.
[12] 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.
[13] F. Ordóñez and D. Roggen, "Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 1, p. 115, 2016.
[14] H. Gjoreski *et al.*, "The university of sussex-huawei locomotion and transportation dataset for multimodal analytics with mobile devices," *IEEE Access*, vol. 6, pp. 42592–42604, 2018.

[15] 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, Aug. 2014.
[16] S. L. Salzberg, "C4.5: Programs for machine learning by J. Ross Quinlan. Morgan kaufmann publishers, Inc., 1993," *Mach. Learn.*, vol. 16, no. 3, pp. 235–240, Sep. 1994.
[17] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.
[18] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, Aug. 1997.
[19] Q. Yanjun, J. Mengling, Y. Weichao, C. Shaomeng, and L. Haiyong, "Transportation mode recognition algorithm based on Bayesian voting," in *Proc. 5th Int. Conf. Enterprise Syst. (ES)*, Sep. 2017, pp. 260–269.
[20] 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.
[21] Z. Chen, L. Zhang, Z. Cao, and J. Guo, "Distilling the knowledge from handcrafted features for human activity recognition," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, pp. 4334–4342, Oct. 2018.
[22] A. Kalatian and B. Farooq, "A semi-supervised deep residual network for mode detection in Wi-Fi signals," 2019, *arXiv:1902.06284*. [Online]. Available: http://arxiv.org/abs/1902.06284
[23] Z. Chen, L. Zhang, C. Jiang, Z. Cao, and W. Cui, "WiFi CSI based passive human activity recognition using attention based BLSTM," *IEEE Trans. Mobile Comput.*, vol. 18, no. 11, pp. 2714–2724, Nov. 2019.
[24] L. Wang and D. Roggen, "Sound-based transportation mode recognition with smartphones," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 930–934.
[25] Y. Qin, H. Luo, F. Zhao, C. Wang, J. Wang, and Y. Zhang, "Toward transportation mode recognition using deep convolutional and long short-term memory recurrent neural networks," *IEEE Access*, vol. 7, pp. 142353–142367, 2019.
[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[27] J. Nagi *et al.*, "Max-pooling convolutional neural networks for vision-based hand gesture recognition," in *Proc. IEEE Int. Conf. Signal Image Process. Appl. (ICSIPA)*, Nov. 2011, pp. 342–347.
[28] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. 14th Int. Conf. Artif. Intell. Statist.*, 2011, pp. 315–323.
[29] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017.
[30] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[31] I. Goodfellow, Y. Bengio, and A. Courville, "Sequence modeling: Recurrent and recursive nets," *Deep Learn.*, vol. 10, pp. 367–415, May 2016.
[32] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
[33] I. V. Tetko, D. J. Livingstone, and A. I. Luik, "Neural network studies. 1. comparison of overfitting and overtraining," *J. Chem. Inf. Model.*, vol. 35, no. 5, pp. 826–833, Sep. 1995.
[34] J. Xiong, K. Zhang, and H. Zhang, "A vibrating mechanism to prevent neural networks from overfitting," in *Proc. 15th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, Jun. 2019, pp. 1929–1958.
[35] 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, 2013.
[36] I. H. Witten, E. Frank, L. E. Trigg, M. A. Hall, G. Holmes, and S. J. Cunningham, "Weka: Practical machine learning tools and techniques with Java implementations," Dept. Comput. Sci., Univ. Waikato, Hamilton, New Zealand, Tech. Rep. 99/11, 1999.
[37] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980

Chenxing Wang (Member, IEEE) received the B.S degree from the Department of Software Engineering, Northeast Petroleum University, Heilongjiang, China. He is currently pursuing the M.S with the School of Software Engineering, Beijing University of Posts and Telecommunications, Beijing, China. His current main interests include transportation mode detection and traffic flow prediction based on spatial–temporal data and methods using deep learning techniques.

Haiyong Luo (Member, IEEE) 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, the Beijing University of Posts and Telecommunication, 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 Science (ICT-CAS), China. His main research interests are 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 Telecommunication, Beijing, China, in 2004 and 2009, respectively. She is currently a Professor in School of Software Engineering, Beijing University of Posts and Telecommunication. Her research interests include mobile computing, location-based services, and computer networks.

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 is mainly engaged in traffic pattern recognition related project research and implementation.