Hindawi
Journal of Computer Networks and Communications
Volume 2019, Article ID 4967261, 11 pages
https://doi.org/10.1155/2019/4967261

Research Article

Recognition of Transportation State by Smartphone Sensors Using Deep Bi-LSTM Neural Network

Hong Zhao,[1] **Chunning Hou**,[1] **Hala Alrobassy**,[1] **and Xiangyan Zeng**[2]

[1]*School of Computer and Communication, Lanzhou University of Technology, Lanzhou 730050, China*
[2]*Department of Mathematics and Computer Science, Fort Valley State University, Fort Valley, GA 31030, USA*

Correspondence should be addressed to Hong Zhao; 594286500@qq.com

Received 23 February 2018; Revised 5 November 2018; Accepted 26 November 2018; Published 3 January 2019

Academic Editor: Giovanni Pau

Smartphones have been used for recognizing different transportation states. However, current studies focus on the speed of the object, which only relies on the GPS sensor rather than considering other suitable sensors and actual application factors. In this study, we propose a novel method that considers these factors comprehensively to enhance transportation state recognition. The deep Bi-LSTM (bidirectional long short-term memory) neural network structure, the crowd-sourcing model, and the TensorFlow deep learning system are used to classify the transportation states. Meanwhile, the data captured by the accelerometer and gyroscope sensors of smartphone is used to test and adjust the deep Bi-LSTM neural network model, making it easy to transfer the model into smartphones and conduct real-time recognition. The experimental results show that this study achieves transportation activity classification with an accuracy of up to 92.8%. The model of the deep Bi-LSTM neural network can be used for other time-series fields such as signal recognition and action analysis.

1. Introduction

Using sensor data to recognize different types of motion state or physical activities has attracted great attention in recent years [1, 2]. Transportation state detection can be considered as a kind of state recognition task, and the transportation state data can be captured by smartphone sensors. Smartphones are equipped with powerful sensors such as accelerometer, gyroscope, pressure, and magnetometer. Owing to the popularity of smartphones, potentially huge volumes of data can be obtained and significant information can be retrieved from these sensors. In the process of training the neural network, the traditional data acquisition method cannot improve the recognition rate of the model because of the quantity and quality of data. To get more information, we propose a crowd-sourcing mode [3] for collecting the transportation state data. The crowd-sourcing mode can provide powerful, reliable, and enriched data so as to enhance the recognition rate.

In the transportation state recognition field, many studies have employed sensors such as GPS (Global Position System) and accelerometers and also used the GIS (Geographic Information System) to detect the modes of transportation. In many case studies, the GPS sensor is used as an integral part to capture the location and the speed. Although the results are encouraging, GPS is not an appropriate sensor for transportation state recognition. Because in many sites, there are tall buildings that affect the GPS signal and result in GPS positioning failure. Some situations are inevitable, such as people crossing the tunnel, entering the building, and taking the subway, which prevents the GPS sensor from working. In order to get much valid motion data, we use a gyroscope sensor and accelerometer sensor instead which can resolve the above problems of GPS.

The transportation recognition problem is tackled by using big data analysis, upgraded software-hardware technology, and cloud business models. Deep learning techniques for data analysis have excellent performance in many fields, such as image recognition, speech recognition, audio processing, and natural language processing. There are many open-source deep learning architecture models, such as Caffe, Theano, Thorch, and TensorFlow. TensorFlow is Google's second-generation AI (artificial intelligence) learning system [4]. It was designed from the ground up to be a good deep learning solution for mobile platforms and could be integrated into Android smartphones easily through Java interface.

There are many kinds of neural network models, built by deep learning techniques, such as autoencoder, restricted Boltzmann machine, deep belief networks, CNN (convolutional neural network), and RNN (recurrent neural network). This study examines some core problems in choosing suitable sensors, selecting various neural network structures and deep learning techniques. Subsequently, the application of the neural network model and crowdsourcing mode to the real-world is also considered.

The remainder of this study is organized as follows. In Section 2, the related works are presented. Section 3 establishes the neural network model and proposes the method for choosing a better neural network. Section 4 presents the experimental result, and Section 5 concludes this study and discusses its limitations.

2. Related Works

Many algorithms have been proposed for transportation state recognition such as SVM (support vector machine), decision trees, random forest, Bayesian belief network, and neural network.

Bolbol et al. [5] introduced a framework which was tested by using coarse-grained GPS data based on SVM classification. However, it took much time to compute the GPS data, get some parameters, and use SVM to classify the states. In addition, this method depended on the GPS data, and if the GPS signal was not stable, then GPS data would exert a negative influence on the classification results. The same issues were also presented in [6, 7]. Zhang et al. [6] used permutation entropy of speed as a feature for transportation state detection and used ELM (extreme learning machine) to distinguish different transportation. While the learning speed of ELM was faster than traditional forward feed network learning algorithms, experimental results were not better.

Shafique and Hato [8] compared the SVM, AdaBoost, decision tree, and random forests to classify the transportation state. Although these experimental results were better, the limitations of these papers were that the data were collected by the special facility and the device could not collect more data at one time. Stenneth et al. [9] proposed a method to infer a mode of transportation based on the GPS sensor on the mobile device and knowledge of the underlying transportation network. While this method used a GPS sensor and the speed of the object to classify the states, it could not recognize the motion state of the object.

Lari and Golroo [10] adapted random forest to analyze the collected data in order to distinguish transportation states. However, this method also relied on the GPS data, especially the speed of the object, if GPS tracking accuracy was not guaranteed, it does not perform well. Shafique and Hato [11] also adapted random forest to classifying the smartphone data among various modes. In that study, the GPS sensor and accelerometer sensor were used to collect data, and the age and gender of tested people were also used as features in classification. In spite of good recognition results, there was an existing problem of the trip segment.

Feng and Timmermans [12] used a Bayesian belief network model to infer transportation states and activity episodes simultaneously and compared the performance of three different groups of sensor data (GPS data only, accelerometer data only, and the combination of GPS and accelerometer data). In addition, it proposed the use of recorded real-time speed and distance in the case of missing data. Xiao et al. [13] identified travel modes using a Bayesian network, four features were extracted to construct the Bayesian network, and two targeted features were added to improve the mode identification performance. Byon and Liang [14] used neural network-based artificial intelligence to identify the mode of transportation, which detected the patterns of the distinct physical profile of each mode. It was also found that the route-specific neural network classifier performed better than the general neural network classifier.

According to the literature above, deep learning and machine learning techniques have been successfully applied in transportation state recognition. However, most of these studies used GPS sensor data as the only source of data to recognize states, which means that the speed of the object determines the result of classification, rather than considering other suitable sensors and real-world status. Based on the above analysis, we propose using the deep Bi-LSTM neural network for classification and collecting raw sensor data from the sensors of acceleration and the gyroscope built in a smartphone. Furthermore, we adopt deep learning methods that can automatically learn the features from a particular dataset that includes a training set and test set [15]. The TensorFlow system is used to construct the model and train and test the neural network architecture. The model is transferred into an Android smartphone for validation. The validation process optimizes the model structure and collects more information, which could improve the accuracy of experimental results.

3. Methodology

3.1. Experimental Setup. The experiments probed a smartphone attached to the waist to collect the sensor data during six common transport states. The main steps of transportation recognition are shown in Figure 1. First, two kinds of sensor data were collected by developing a smartphone application or using other applications in the App store. Then, the data were filtered to remove random noise during the preprocessing phase. After that, the data were divided into blocks of the same size so that various features could be extracted. It was vital to extract the most discriminative features that can distinguish different transport states. Next step, the neural network was trained until certain recognition accuracy was achieved. Finally, the model was transferred to the smartphone to test real-time situation by the smartphone application program.

In this study, six types of transport state are defined, namely, BIC (bicycling), BUS (bus), RUN (running), STI (statics), SUB (subway), and WAL (walking), as shown in Table 1. People performed six states wearing a smartphone on the waist, making sure the monitor of the smartphone is facing out; the steps are as shown in Figure 2. The sensor data,

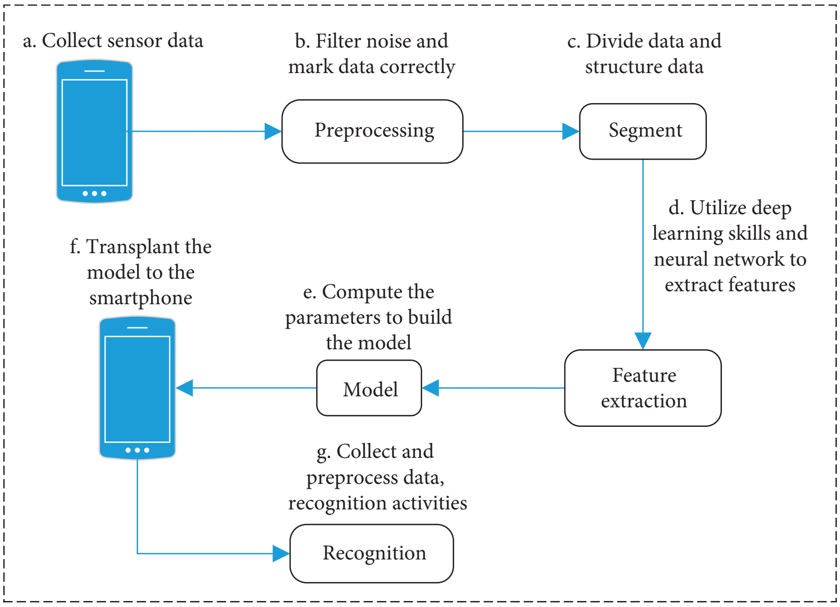

a. Collect sensor data
b. Filter noise and mark data correctly
Preprocessing
c. Divide data and structure data
Segment
d. Utilize deep learning skills and neural network to extract features
Feature extraction
e. Compute the parameters to build the model
Model
f. Transplant the model to the smartphone
g. Collect and preprocess data, recognition activities
Recognition

Figure 1: Flowchart of transportation recognition.

Table 1: Types of transport states.

No.	Transport state	Definition
S1	BIC	Riding a bicycle with a phone
S2	BUS	Taking a bus with a phone
S3	RUN	Running with a phone
S4	STI	Stilling with a phone
S5	SUB	Taking the subway with a phone
S6	WAL	Walking with a phone

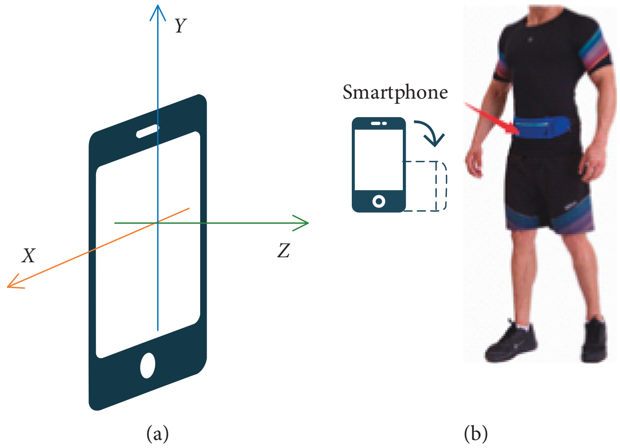


Figure 2: Wearing a phone on the waist. (a) Cell phone coordinate system; (b) how to put the smartphone.

which are the signals obtained by applying acceleration and angular velocity, were collected by a HUAWEI P9 smartphone and a XIAOMI smartphone at 50 Hz sampling rate.

3.2. Data Collection. In order to improve the accuracy, the crowd-sourcing mode was used to collect the data. Figure 3 shows collecting the transportation state using a smartphone. The crowd-sourcing mode enables to get and share more information. The data were collected by smartphones, and the various states of information were sent to the data centered by Internet. The data center will analyze these data and send the results to the users.

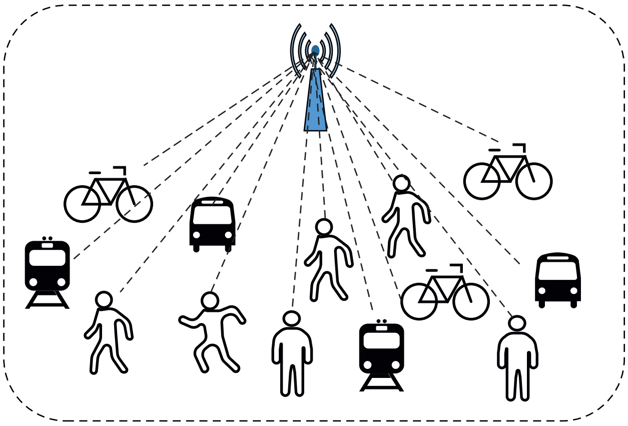
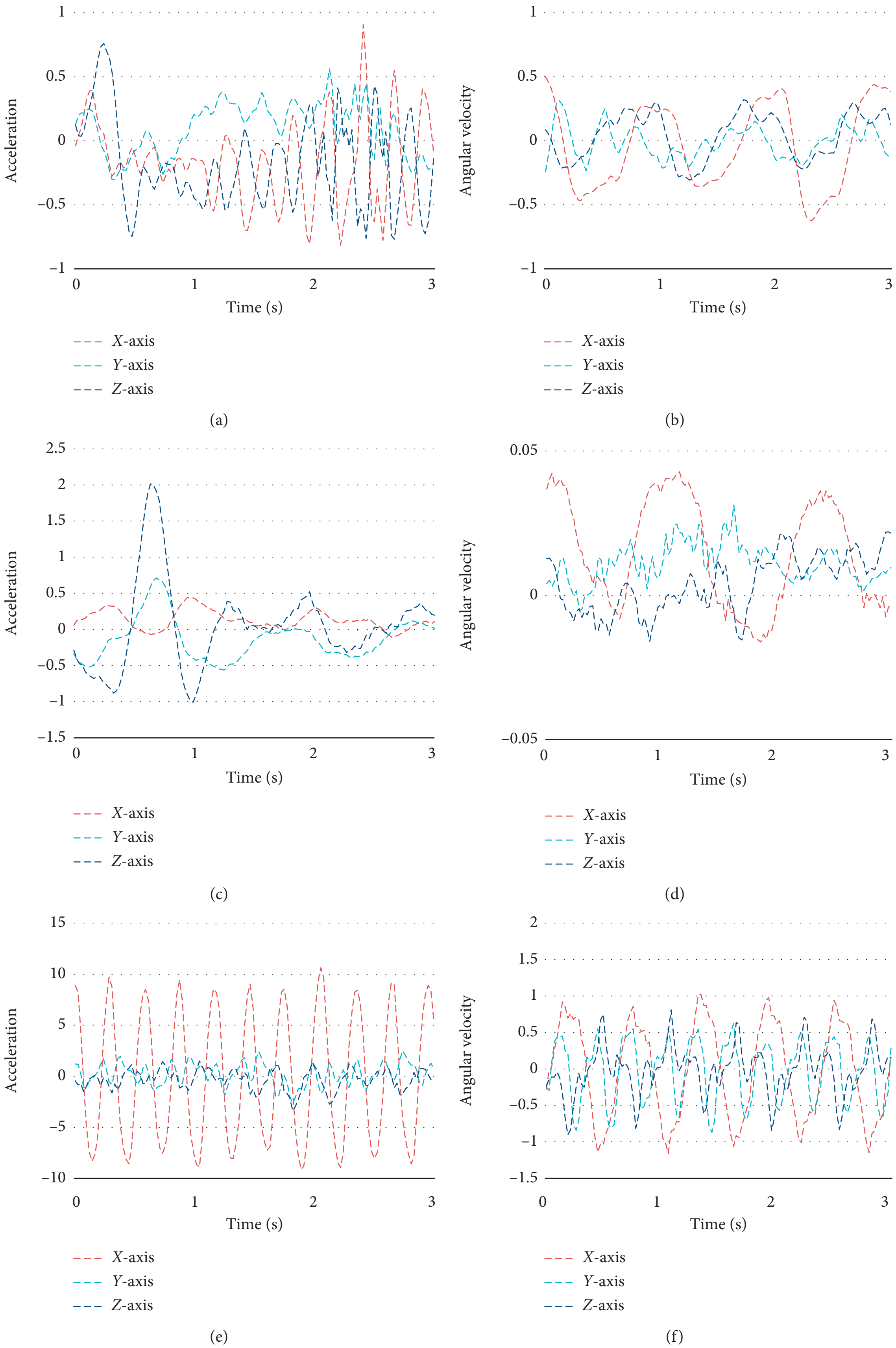


Figure 3: Using crowd-sourcing mode to collect transport states with the smartphone.

3.3. Transportation State Classification

3.3.1. Different Transport States. In this study, acceleration and angular velocity are used to describe different states. Figure 4 shows acceleration and angular velocity signals for the six transport states, in which the smartphone is placed on the waist as sown in Figure 2. Compared with other states, the signal of BIC is very sensitive to the road condition, and this is because the contact area of bicycle tires and ground is relatively small especially when the speed is fast and the road condition is poor. In order to reduce noise, the BIC signal was filtered. The acceleration signals of STI and SUB were also filtered. When the subway is running at a constant



(a)

(b)

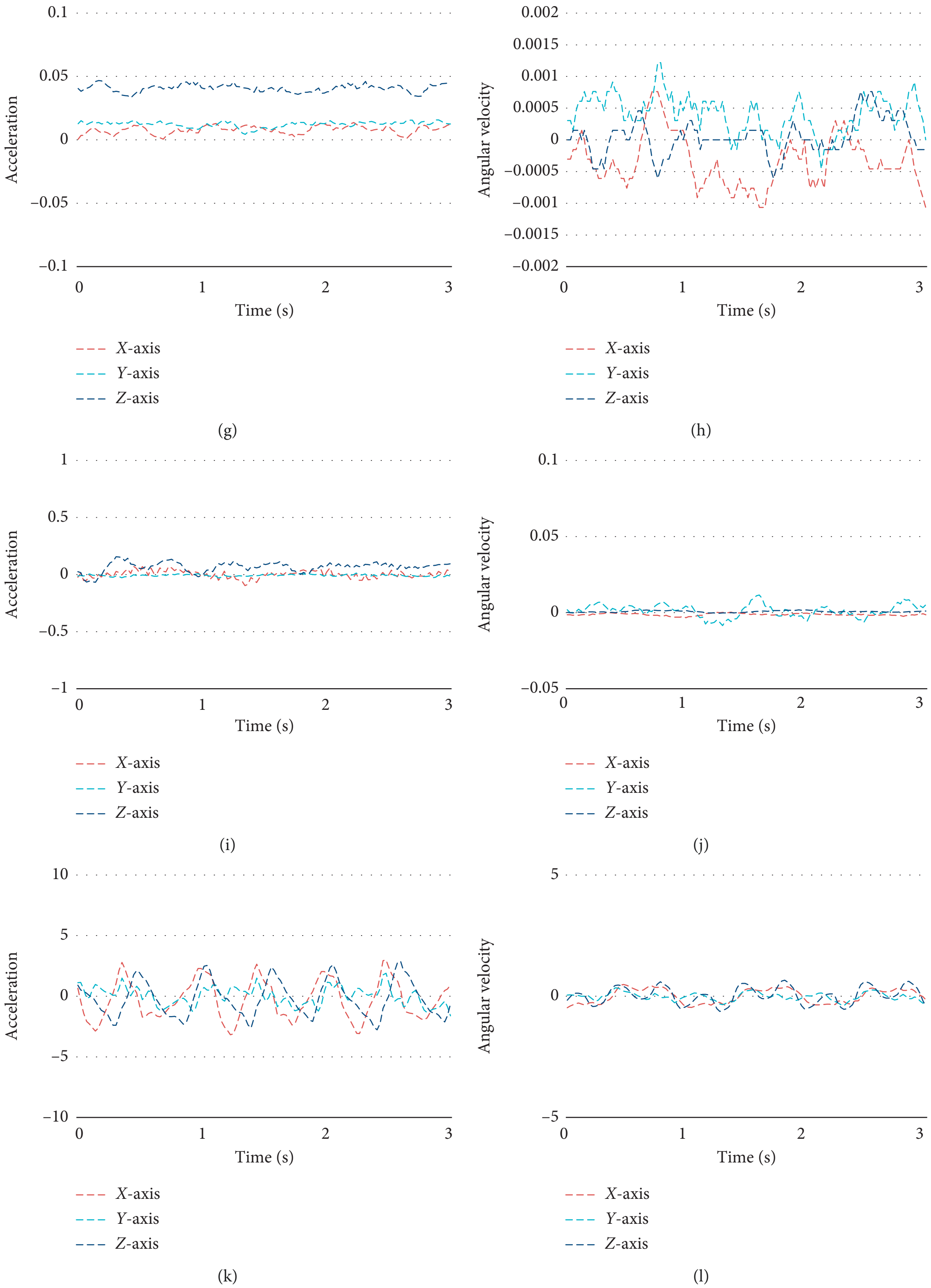
(c)

(d)

(e)

(f)

Figure 4: Continued.



(g)

(h)

(i)

(j)

(k)

(l)

Figure 4: Signals of acceleration and angular velocity for six transport states: (a) acceleration and (b) angular velocity of BIC; (c) acceleration and (d) angular velocity of BUS; (e) acceleration and (f) angular velocity of RUN; (g) acceleration and (h) angular velocity of STI; (i) acceleration and (j) angular velocity of SUB; (k) acceleration and (l) angular velocity of WAL.

speed, it is difficult to recognize it. Therefore, the angular velocity signal is used to identify it because the ups and downs of subway line lead to differences in angular velocity signals. The differences between the signals of RUN and WAL are more noticeable. Furthermore, every transport state has its own characteristic values of acceleration and angular velocity. Each state has at least one key feature to distinguish it from other states.

3.3.2. Feature Extraction. Before extracting features, the segment is needed for data preprocessing. Features are extracted from sensor data within a short time frame called a window that covers the full features of a transport state [16]. The duration of each state determines the size of the window. A smaller size of the window may not precisely contain the full characteristics of each state, while a larger one may bring noises in [17]. Typically, the time signals were then sampled in fixed-width sliding window of 2.56 sec and 50% overlap between them, optimized for the power of two vectors (2.56 sec × 50 Hz = 128 cycles) [18].

When using sensor signals to discriminate transport state, it is vital to consider the temporal dependence of nearby reading. LSTM neural network that utilizes data over time intervals is appropriate for this task. The discriminating transport state can be deemed to a sort of classification issue, where the input data set is time series signals and the output set is the transport state label. Figure 5 shows the transport state recognition process, consisting of a training phase and a test phase. The features are extracted from the raw time series data set and then utilized to train a perfect classification model until the stopping criteria are achieved during the training phase. In the test phase, features are extracted from the test set and the trained classification model is used to predict a transport state label.

Figure 5 also shows that various neural networks can be used to extract features. In this study, Bi-LSTM was utilized to extract features. In a classical RNN, the transmission of the state is one-way, from front to back, which limits the direction of propagation. In this study, the output of the current moment is not only related to the previous state, but also the state after it. So the Bi-LSTM is necessitated to deal with this condition. One primary purpose of the Bi-LSTM is to increase its available information.

3.3.3. Classification. The structure of Bi-LSTM is illustrated in Figure 6(a), and the structure of LSTM is illustrated in Figure 6(b). The Bi-LSTM structure has two directions which are the forward layer and backward layer, and the two-layers have no intersection in the running phase until the final output layer. In this study, based on features of sensor data which are time sequence, the output results of the current moment is not only related to the previous state, but also the state after it, so the Bi-LSTM is used to record the information of backward layer and forward layer and to increase expression of the neural network. However, the normal single-directional LSTM cannot get the information of the next state. The computation process of Bi-LSTM is defined as follows:

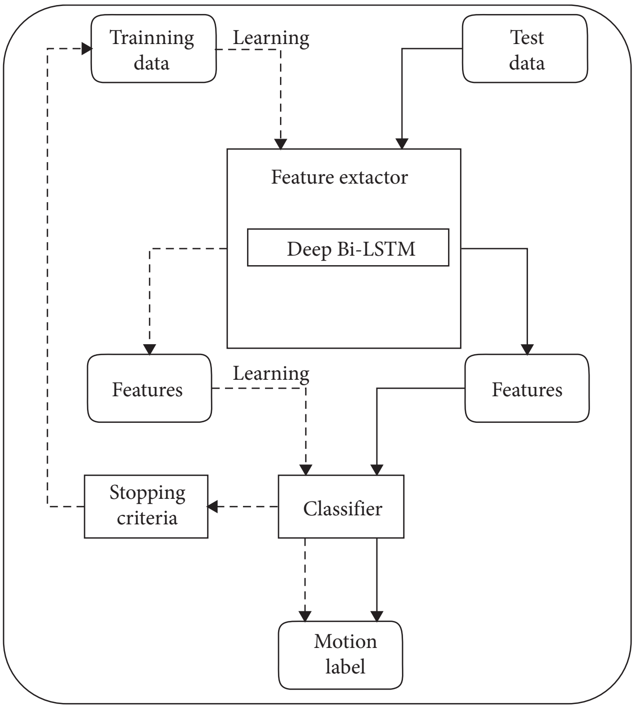


Figure 5: Feature extraction for transport state recognition.

$$\begin{aligned}\overrightarrow{\text{forward}_t} &= f\left(U_f X_t + W_f \overrightarrow{h_{t-1}} + \overrightarrow{b}\right),\\ \overleftarrow{\text{backward}_t} &= f\left(U_b X_t + W_b \overleftarrow{h_{t+1}} + \overleftarrow{b}\right),\\ \text{output}_t &= g\left(J\left[\overrightarrow{\text{forward}_t V_f} + \overrightarrow{d}\,; \overleftarrow{\text{backward}_t V_b} + \overleftarrow{d}\right] + c\right),\end{aligned} \tag{1}$$

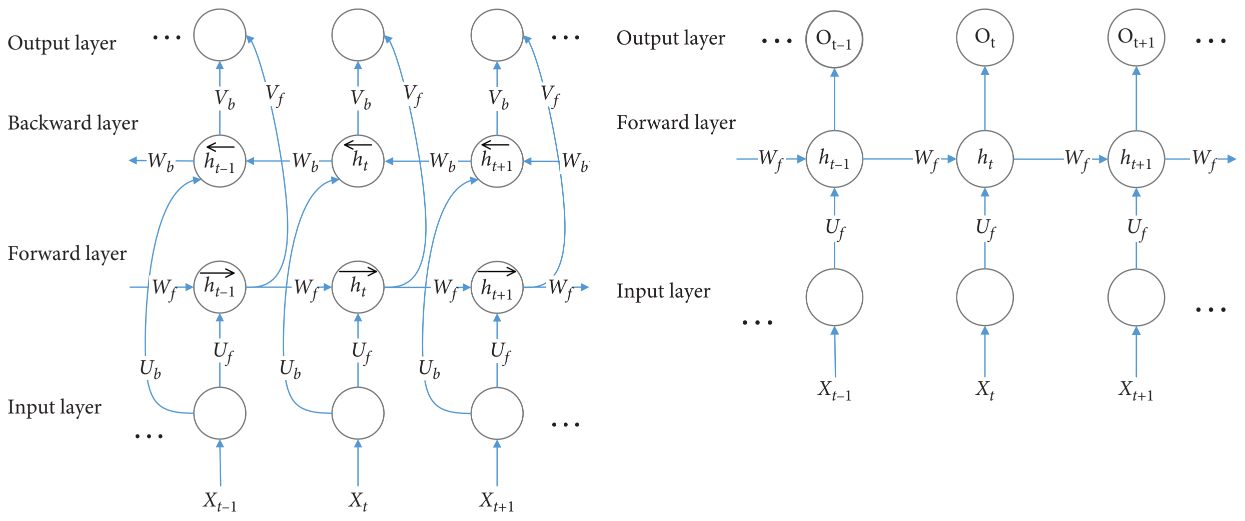
where $\overrightarrow{\text{forward}_t}$ is the result of the forward layer which is not a vector but is a real number; backward $_t$ is the result of the backward layer, which is a real number also; output$_t$ is the probability value generated by combining $\overrightarrow{\text{forward}_t}$ and $\overleftarrow{\text{backward}_t}$; f and g are nonlinear activation functions; J is the joint function; h is the values of the hidden layer; U, W, and V are the weight matrices; and b, d, and c are the bias values.

When feature extraction has been done, the next step is the classification. To normalize the probabilities of the output labels, the softmax layer is the inferred state class that is expressed as follows:

$$\text{soft max }(l_i) = \frac{\exp(l_i)}{\sum_{k=1}^{K}\exp(l_k)}, \tag{2}$$

where l_i is the ith value computed by multiplying the neuron values in the last layer with the weights connected to the ith output neuron, soft max (l_i) is the output probability, and $\sum_i^K \text{soft max }(l_i) = 1$.

3.3.4. Regularization. In spite of Bi-LSTM has superior performance in some fields, there are some weaknesses such as optimum problem and overfitting. Regularization can be used to compensate for these weaknesses. Weight decay and



(a) (b)

Figure 6: Structures of (a) Bi-LSTM and (b) LSTM.

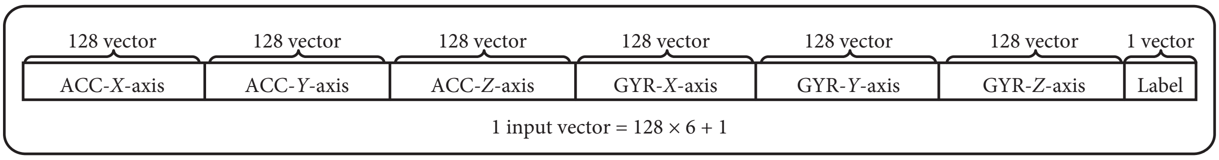


Figure 7: Format of input data.

L2 regularization are regularization methods that add an extravalue into the loss function to penalize the large weights, which can be expressed as follows:

$$C = C_0 + \frac{\lambda}{2n}\sum_{w} w^2, \tag{3}$$

where C is the value of regularization; C_0 is the cost function; λ is the regularization coefficient ($\lambda > 0$); and w is calculated as shown in the following equations according to the rule that regularization can choose a model with low risk and complexity at the same time:

$$w \longrightarrow w - \eta\frac{\partial C_0}{\partial w} - \frac{\eta\lambda}{n} w, \tag{4}$$

$$w = \left(1 - \frac{\eta\lambda}{n}\right) w - \eta\frac{\partial C_0}{\partial w}, \tag{5}$$

where η is the learning rate and $(1 - (\eta\lambda/n))$ is a factor which can adjust the weight and make weight smaller.

4. Experiments

In the experiment, 11 volunteer subjects performed these six transport states while carrying the smartphones on their waists. There are 8 subjects that are used for training, and the others are used for testing. The amount of training set is 16755 ∗ 128 ∗ 6, and the test set is 4239 ∗ 128 ∗ 6. The sensor data were sampled at a rate of 50 Hz and divided into windows of 128 values. Then, this study puts 6 axes (acceleration 3 axes and gyroscope 3 axes) in a raw input, and 1 input vector = 128 ∗ 6 + 1. The format of input data is shown in Figure 7.

Table 2: Experiment configuration.

Parameter	Value
Size of input vector	768
Size of batch	128
LRB (learning rate base)	0.5–0.005
LRD (learning rate decay)	0.90–0.95
Activation function	ReLU
RE (regularization rate)	0.5–0.005
HP (hidden points)	50–300
MAD (moving average decay)	0.99
Epochs	1000–1400
Probability of dropout	0.5–1

The experimental configuration is shown in Table 2. The size of batch expresses the size of data for every running process. The LRB (learning rate base) expresses basic learning rate and ranges from 0.5 to 0.005. The LRD (learning rate decay) expresses the decay rate. ReLU (rectified line unit), a kind of activation functions, is used to increase the nonlinear variation, which can make the neural network learn more features. To get a better solution and adjust the learning rate, the high learning rate is used in the beginning. Multiplied LRB and LRD are used to reduce the learning rate gradually. Then, the DLR (decayed learning rate) is defined as follows:

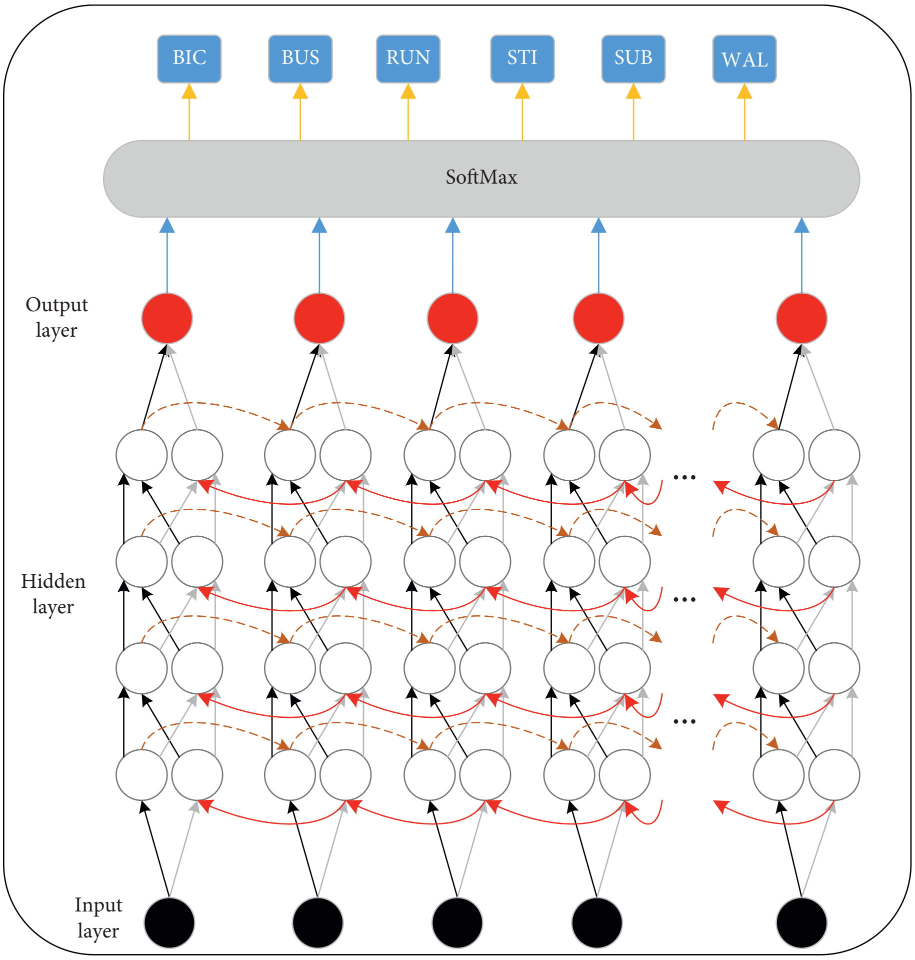


Figure 8: Structure of experimental neural network.

$$\mathrm{DLR} = \mathrm{LRB} \times \mathrm{LRD}^{(\mathrm{globalStep}/\mathrm{decaySteps})}, \quad (6)$$

where DLR is the learning rate that is used in each round of optimization, decaySteps is the speed of the decay rate, and globalStep is the global step's counter.

In this study, the moving mean model was adopted to make the model more robust. The MAD (moving average decay) determined the updating speed of the Bi-LSTM neural network model. When the value of the MAD is greater, where the reasonable values for decay is close to 1.0, the network model updates faster. An empirical value for the number of training epochs was set up from 1000 to 1400, which could be a stopping criterion for neural network training.

To increase the accuracy of Bi-LSTM network classification, different numbers of hidden layers were tested and four layers yielded the best results. The structure of the neural network is shown in Figure 8.

The experimental results are shown in Table 3. It was found that the accuracy is sensitive to hyperparameters. In order to get a better result in this study, different optimization methods were used. In the design of neural network structure, multihidden layers were used; in the process of neural network optimization, the LRD, the loss function of regularization, and the moving average model are used which are provided by TensorFlow.

The classification accuracy is shown in Figure 9. The blue curve expresses the accuracy of the training data, and the red curve shows the accuracy of the test data. In most cases, the accuracy of the training data is a little higher than that of the test data. Moreover, the red curve converges quickly, and the results are relatively stable, which demonstrates that the proposed model is robust and stable. To better evaluate the experimental results, the confusion matrix was utilized to illustrate the results, as shown in Table 4. The precision, recall, and F Score are defined in the following equations:

$$\mathrm{precision} = \frac{\mathrm{tp}}{(\mathrm{tp} + \mathrm{fp})}, \quad (7)$$

$$\mathrm{recall} = \frac{\mathrm{tp}}{(\mathrm{tp} + \mathrm{fn})}, \quad (8)$$

$$\mathrm{F1} = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}, \quad (9)$$

where tp is the true positives, fp is the false positives, and fn is the false negatives.

As demonstrated in Table 4, STI and SUB activities are the difficult ones for deep Bi-LSTM to discriminate. The reason is that the signals of subway and STI are similar

Table 3: Parts of the experiment results.

LRB	LRD	RE	HP	DR	Accuracy
0.03	0.90	0.01	250	0.75	0.812
0.03	0.90	0.01	300	1	0.874
0.015	0.90	0.01	50	1	0.845
0.1	0.90	0.05	50	1	0.564
0.05	0.90	0.03	50	1	0.743
0.025	0.90	0.015	50	1	0.845
0.025	0.90	0.015	100	1	0.843
0.015	0.95	0.005	50	1	0.856
0.0055	0.95	0.005	50	1	0.928
0.001	0.95	0.005	50	1	0.867
0.001	0.95	0.001	50	1	0.882

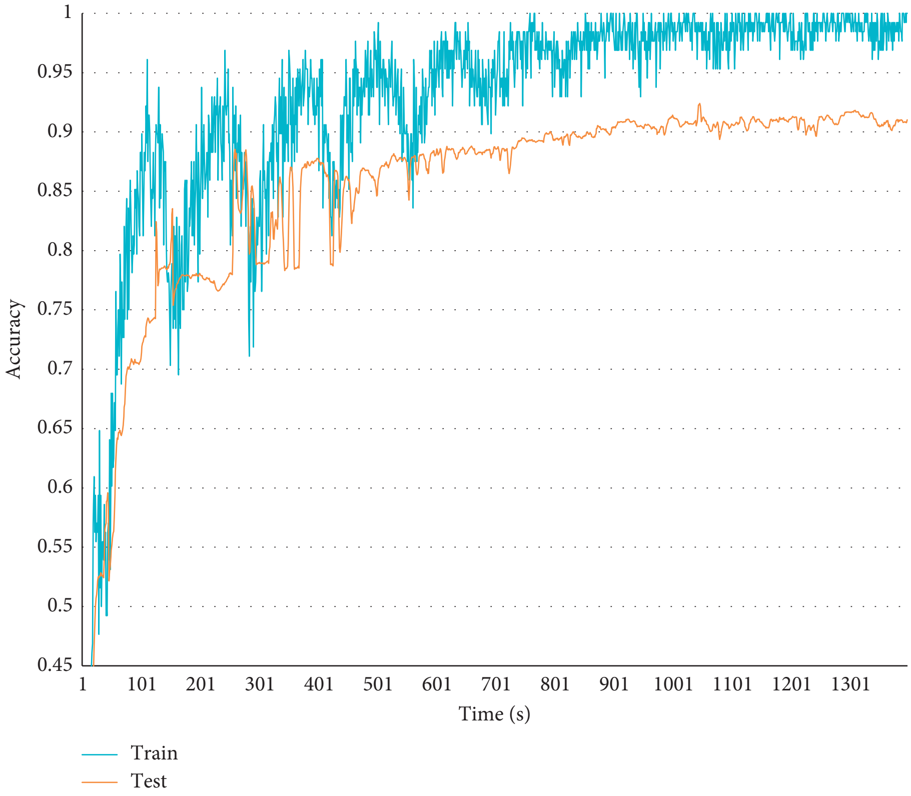
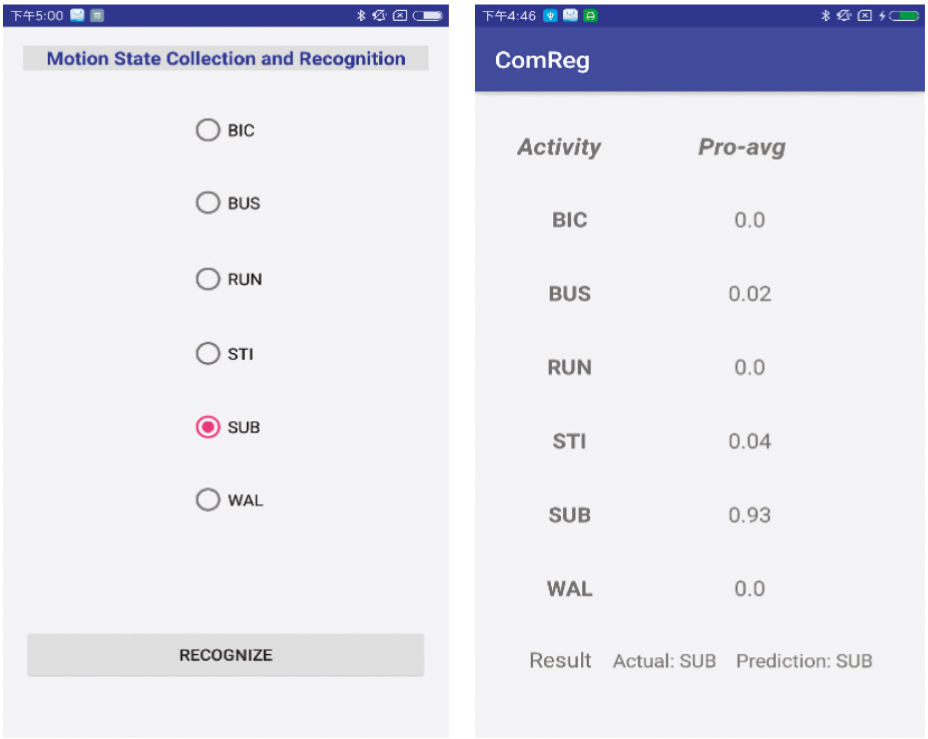


Figure 9: Performance of Bi-LSTM neural network.

Table 4: Results of confusion matrix.

		Predicted Class						
		BIC	BUS	RUN	STI	SUB	WAL	Recall
True class	BIC	999	36	13	0	3	6	0.945
	BUS	9	466	0	0	23	0	0.936
	RUN	0	0	758	0	0	0	1
	STI	0	0	0	548	184	0	0.749
	SUB	2	0	0	6	682	0	0.991
	WAL	23	3	0	0	0	480	0.949
	Precision	0.969	0.923	0.983	0.989	0.765	0.988	
	F1	0.957	0.929	0.991	0.852	0.864	0.968	**0.928**



Motion State Collection and Recognition

○ BIC

○ BUS

○ RUN

○ STI

● SUB

○ WAL

RECOGNIZE

Activity	Pro-avg
BIC	0.0
BUS	0.02
RUN	0.0
STI	0.04
SUB	0.93
WAL	0.0

Result Actual: SUB Prediction: SUB

(a) (b)

Figure 10: Performance of the model in the smartphone.

sometimes especially when stationary activities suddenly receive an external effect, and the state of the object changes slightly, which results in signal changing and making the state change.

In order to test the accuracy of the model in the actual situation, the neural network model was transferred to an Android phone. The transfer process is straightforward becuase TensorFlow has the Android interface. We developed an Android app to test the neural network model, which can collect the transport state data and recognize the transport state, as shown in Figure 10. In Figure 10(b), Pro-avg indicates the probability that the state is identified by the model as a different motion state. For example, if SUB is selected, then the SUB data are collected and submitted in Figure 10(a), and then the results are shown in Figure 10(b) which presents the probabilities of all the states; we can find that the SUB is 0.93 which presents the probability of SUB data being identified in the SUB state and gives the final results at the bottom of the page.

This study also compared the deep Bi-LSTM neural network with RNN and other variants of the RNN network as shown in Table 5. The results showed that deep Bi-LSTM outperforms other neural network models, which means the deep Bi-LSTM can deal with the recognition of the state than other variants of RNN. The internal structure of the LSTM unit can remember more information than the RNN unit. So it is why the result of the LSTM is better than RNN. The Bi-LSTM neural network changes the LSTM neural network structure which imports the backward layer and forward layer, which leads the Bi-LSTM neural network make the most of the date features. The key point is the structure of the deep Bi-LSTM which can increase the availability in the information of different states than other variants of RNN. For training the neural network model on a general personal computer, the deep Bi-LSTM neural network was recommended.

Table 5: Performance of deep Bi-LSTM compared to other algorithms.

Algorithm	Layer	LRB	Batch size	HP	Epochs	Accuracy
RNN	1	0.005	128	128	1200	0.504
LSTM	1	0.005	128	128	1200	0.774
Multi-LSTM	2	0.0075	128	128	1200	0.823
Bi-LSTM	1	0.005	128	128	1200	0.901
Deep Bi-LSTM	2	0.0025	128	64	1400	0.928

5. Conclusion

This study proposes a transport state detection method using a deep learning approach based on the deep Bi-LSTM neural network. Although most of the previous studies have adopted various methods, they overlooked some problems such as the attenuation of GPS signals in urban areas and tunnels. When smartphone cannot receive the GPS signal and the accelerometer sensor cannot record all signals of the transport state, this study uses the gyroscope sensor instead of the GPS sensor to record signals. In order to get more data, this study uses the crowd-sourcing mode to develop an app to collect more data and enhance the recognition accuracy.

Future work will be based enriching the dataset, adjusting the neural network structure, and improving the recognition rate.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Natural Science Foundation of China under Grant nos. 51668043 and 61262016 and the CERNET Innovation Project under Grant nos. NGII20160311 and NGII20160112.

References

[1] B. Ling and S. S. Intille, "Activity recognition from user-annotated acceleration data, pervasive computing," in *Proceedings of Second International Conference, PERVASIVE 2004*, pp. 1–17, Vienna, Austria, April 2004.

[2] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Activity recognition using cell phone accelerometers," *Acm SIGKDD Explorations Newsletter*, vol. 12, no. 2, pp. 74–82, 2011.

[3] J. Howe, "The rise of crowdsourcing," in *Convergence, Culture Where Old and New Media Collide*, H. Jenkins, Ed., pp. 1–5, NYU Press, New York City, NY, USA, 2006.

[4] F. Foerster, M. Smeja, and J. Fahrenberg, "Detection of posture and motion by accelerometry: a validation study in ambulatory monitoring," *Computers in Human Behavior*, vol. 15, no. 5, pp. 571–583, 1999.

[5] A. Bolbol, T. Cheng, I. Tsapakis, and J. Haworth, "Inferring hybrid transportation modes from sparse GPS data using a moving window SVM classification," *Computers, Environment and Urban Systems*, vol. 36, no. 6, pp. 526–537, 2012.

[6] L. Zhang, M. Qiang, and G. Yang, "Mobility transportation mode detection based on trajectory segment," *Journal of Computational Information Systems*, vol. 9, no. 8, pp. 3279–3286, 2013.

[7] A. Jahangiri and H. Rakha, "Developing a support vector machine (SVM) classifier for transportation mode identification by using mobile phone sensor data," in *Proceedings of Transportation Research Board, Meeting*, pp. 1430–1442, Washington, DC, USA, January 2014.

[8] M. A. Shafique and E. Hato, "Use of acceleration data for transportation mode prediction," *Transportation*, vol. 42, no. 1, pp. 163–188, 2014.

[9] L. Stenneth, O. Wolfson, P. S. Yu et al., "Transportation mode detection using mobile phones and GIS information," in *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 54–63, Chicago, IL, USA, November 2011.

[10] Z. A. Lari and A. Golroo, "Automated transportation mode detection using smart phone application via machine learning: case study mega city of Tehran," in *Proceedings of Transportation Research Board Conference*, pp. 144–153, Washington, DC, USA, January 2015.

[11] M. Shafique and E. Hato, "Travel mode detection with varying smartphone data collection frequencies," *Sensors*, vol. 16, no. 5, pp. 716–739, 2016.

[12] T. Feng and H. J. P. Timmermans, "Transportation mode recognition using GPS and accelerometer data," *Transportation Research Part C: Emerging Technologies*, vol. 37, no. 3, pp. 118–130, 2013.

[13] G. Xiao, Z. Juan, and C. Zhang, "Travel mode detection based on GPS track data and Bayesian networks," *Computers, Environment and Urban Systems*, vol. 54, pp. 14–22, 2015.

[14] Y.-J. Byon and S. Liang, "Real-time transportation mode detection using smartphones and artificial neural networks: performance comparisons between smartphones and conventional global positioning system sensors," *Journal of Intelligent Transportation Systems*, vol. 18, no. 3, pp. 264–272, 2014.

[15] M. Munoz-Organero and R. Ruiz-Blazquez, "Time-elastic generative model for acceleration time series in human activity recognition," *Sensors*, vol. 17, no. 2, pp. 319–336, 2017.

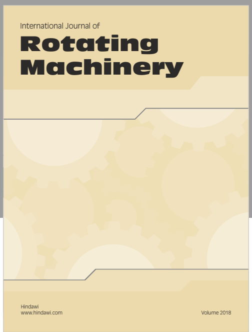
[16] F. Gu, A. Kealy, K. Khoshelham, and J. Shang, "User-independent motion state recognition using smartphone sensors," *Sensors*, vol. 15, no. 12, pp. 30636–30652, 2015.

[17] S. Reddy, M. Mun, J. Burke, D. Estrin, M. Hansen, and M. Srivastava, "Using mobile phones to determine transportation modes," *ACM Transactions on Sensor Networks*, vol. 6, no. 2, pp. 1–27, 2010.

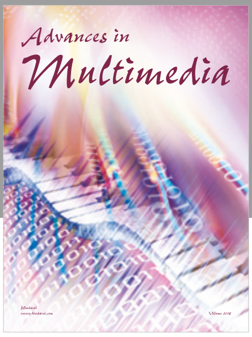
[18] D. Anguita, A. Ghio, L. Oneto et al., "A public domain dataset for human activity recognition using smartphones," in *Proceedings of ESANN*, pp. 437–442, Bruges, Belgium, April 2013.

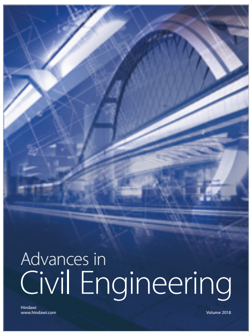
Journal of
Engineering

The Scientific
World Journal

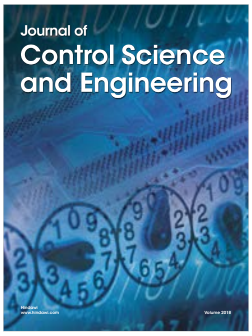

International Journal of
Rotating
Machinery

Journal of
Sensors

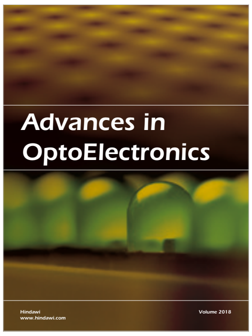

Advances in
Multimedia


Advances in
Civil Engineering

Journal of
Robotics

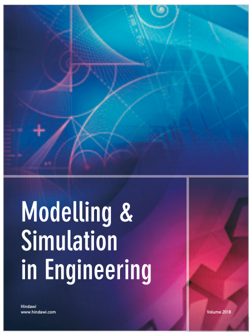

Journal of
Control Science
and Engineering

Journal of
Electrical and Computer
Engineering

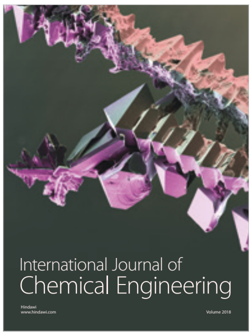

Advances in
OptoElectronics

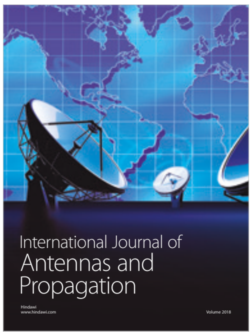
VLSI Design

International Journal of
Navigation and
Observation

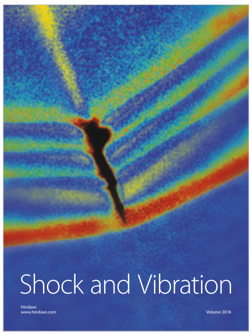

Modelling &
Simulation
in Engineering

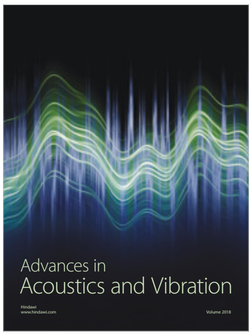
International Journal of
Aerospace
Engineering


International Journal of
Chemical Engineering


International Journal of
Antennas and
Propagation

Active and Passive
Electronic Components


Shock and Vibration


Advances in
Acoustics and Vibration