Automation and Machine Learning (2024)
Clausius Scientific Press, Canada

DOI: 10.23977/autml.2024.050107
ISSN 2516-5003 Vol. 5 Num. 1

Analysis of urban vehicle driving mode based on deep-learning method

Su Zhou[1,2,*], Yang Jiao[1], Jianhua Gao[1], Xiaoman Liu[1]

[1]College of Automotive Studies, Tongji University, Shanghai, China
[2]China-Germany Institute, Tongji University, Shanghai, China
[*]Corresponding author

Keywords: Driving behavior recognition, automatic driving, deep learning, neural network, model fusion

Abstract: In view of the data processing difficulties in the data collection of vehicle driving behavior, the neural network structures of FCN, ResNet, LSTM and Bi-LSTM are designed and optimized. Then, ResNet and Bi-LSTM with better performance were selected for model fusion, which improved the overall performance of the model. Finally, different degrees of error disturbance are added to the test set to verify the anti-interference and generalization ability of the fusion model. The results show that the classification accuracy of the model does not decline when there is 5% disturbance in the data, maintains a high level of performance, avoids the problem of high-dimensional features after the One-hot coding, realizes the same batch training of variable length samples, and improves the efficiency of model training.

1. Introduction

With the acceleration of automation in the transportation field, autonomous driving technology has gradually become an important part of transportation development. The driving behavior of vehicles in different scenarios has different characteristics. Urban scenes are the most common scenes in life. In this scenario, the driving behavior of the vehicle is related to its own driving state and the state of surrounding vehicles. Therefore, when performing driving behavior recognition it is necessary to consider both its own state behavior and its own driving behavior interacting with surrounding vehicles.

In recent years, many experts and scholars at home and abroad have conducted research on driving behavior. Mobyen Uddin Ahmed et al.[1] use artificial neural network (ANN), convolutional neural network (CNN), K nearest neighbor (KNN), hidden Markov model (HMM), random forest (RF) and other methods to identify vehicle driving modes. For driving mode, the results show that the CNN method has the highest accuracy, while the RF method can capture the correlation between data better than CNN. Lattanzi Emanuele et al.[2] trained and tested a support vector machine (SVM) and ANN with multiple features computed over 200 kilometers of travel. The classification results show that the average accuracy using SVM and ANN is about 88% and 90% respectively. Arati Gerdes et al.[3] use Bayesian networks to infer the probability of vehicle driving behavior from data with uncertainty. Xie et al.[4] proposed a method using multi-feature CNN to extract data features for real-time

dangerous driving behavior identification, which has improved accuracy compared with traditional neural networks. Chu et al.[5] proposed a driving behavior recognition model based on the tutor-student network, which uses a deeper network to extract higher-level abstract features, overcoming the limitations of traditional methods. Xue et al.[6] studied the identification method of dangerous driving behaviors such as car following and lane changing, proposed a modified collision margin to correct the trajectories of vehicles of different sizes, and used discrete wavelet transform and statistical methods to extract the key parameter input of the trajectory, and finally a lightweight gradient boosting machine is used to evaluate and identify driving behavior.

This article mainly studies driving behavior recognition based on deep learning, uses convolutional neural networks and recurrent neural networks to extract features required for training models, and studies the performance of strong feature extractors convolutional neural networks and recurrent neural networks on driving behavior recognition datasets, and conducts network optimization for this dataset, using model fusion methods to obtain a more stable model that is less susceptible to noise, achieving higher accuracy and anti-interference capabilities.

2. Dataset introduction and data preprocessing

This article mainly uses the INTERACTION dataset in the "DR USA Intersection EP0" file in HighD[7] for training and experiments. The dataset has 724 time series samples, 80% of the data is used as training set to train the model, and 20% of the data is used as test set to verify the performance of the model. The sequence length for each sample varies from 17 to 777 time steps. The INTERACTION dataset is collected by drones or fixed cameras in different areas and covers common complex scenes in urban scenes, such as roundabouts, unsignalized intersections, and signalized intersections.

The dataset provides information on the motion of the observed vehicle and all surrounding vehicles that may influence its behavior. As shown in Table 1, the features all carry time information and change with time, where track represents the ID of the vehicle, frame represents the frame in which the vehicle appears in the video, time represents the moment when the vehicle appears in the video, and psi is The heading angle of the vehicle in each frame, length represents the length of the vehicle, and width represents the width of the vehicle.

Table 1: Dataset format example.

track	frame	time	agent	x	y	vx	vy	psi	length	width
1	1	100	car	13.856	38.136	0.098	-0.583	-1.405	4.67	1.98
1	2	200	car	13.856	38.079	0.088	-0.566	-1.417	4.67	1.98
1	3	300	car	13.874	38.023	0.078	-0.547	-1.429	4.67	1.98
1	4	400	car	13.881	37.970	0.070	-0.527	-1.439	4.67	1.98
1	5	500	car	13.888	37.918	0.063	-0.505	-1.448	4.67	1.98
1	6	600	car	13.894	37.869	0.056	-0.483	-1.455	4.67	1.98
1	7	700	car	13.899	37.821	0.051	-0.462	-1.461	4.67	1.98
1	8	800	car	13.904	37.776	0.048	-0.445	1.775	4.67	1.98
1	9	900	car	13.909	37.732	0.049	-0.433	1.802	4.67	1.98
1	10	1000	car	13.914	37.689	0.052	-0.427	1.782	4.67	1.98
1	11	1100	car	13.919	37.646	0.060	-0.426	1.786	4.67	1.98
1	12	1200	car	13.926	37.603	0.069	-0.430	1.786	4.67	1.98
1	13	1300	car	13.933	37.560	0.080	-0.440	1.794	4.67	1.98

There are four main categories of category labels in the INTERACTION dataset as shown in Figure 1.

1) Vehicle state behavior, including 6 categories: acceleration, constant speed, deceleration, start, stationary, and parking. The starting behavior is the transition state from the vehicle parking state to the acceleration state, and the parking behavior is the transition state from the vehicle deceleration state to the stationary state.

2) Forward behavior mainly includes three categories: undetectable behavior, following the vehicle in front, and approaching the vehicle in front.

3) Lane behavior mainly includes three types of behaviors: undetectable, lane keeping, and lane changing.

4) Turning behavior, including no turning behavior, left turning, and right turning.

Vehicle state behavior		
Acceleration	Constant speed	Deceleration
Start	Stationary	Parking
Forward behavior(Interaction with front vehicles)		
Undetectable behavior	Following the vehicle in front	Approaching the vehicle in front
Lane behavior		
Undetectable behavior	Lane keeping	Lane changing
Turning behavior		
No turning behavior	Left turning	Right turning

Figure 1: INTERACTION dataset vehicle behavior category labels.

The following challenges exist when using datasets for vehicle driving behavior recognition.

1) Data imbalance: The number of labels of a certain type in the dataset is significantly less or more than the number of labels of another type. Majority behavior labels have a greater impact on the loss function, so the model will tend to reduce the loss for majority behavior labels while ignoring a smaller number of categories, thus showing poor classification performance.

2) Missing values: Missing values are data that were not collected or stored correctly when collecting data, so null values appear. This will affect the training of the model and introduce new errors.

3) Different data dimensions: In the dataset, different features usually have different sizes and size units. For example, the value range of the vehicle's x-axis position in the absolute coordinate system is 900-1000m, and the vehicle speed range is 0-10m/s. Since the neural network is optimized based on the gradient descent method, when the data is not normalized, the gradient will be optimized in the direction of larger values, which will cause the model to converge too slow.

4) High dimensionality caused by categorical features: There are some categorical features in both data sets, such as Lane ID and Road ID. One of the common methods to deal with these features is One-hot encoding [8,9]. The total number of ID categories of the lane ID feature in the dataset reaches 86, which means that if one-hot encoding is performed directly on this feature, an 86-dimensional sparse vector will be generated. As the number of dimensions increases, the search space of the model grows exponentially.

Data preprocessing is an important step in classification problems. The quality of data preprocessing determines the performance upper limit of the algorithm. The following approach is used in this article to overcome the previously mentioned challenges.

1) Solution to data imbalance:

In an imbalanced dataset, it is easy to be deceived by the results if accuracy is used as the only evaluation method. Therefore, more effective methods are needed to evaluate the performance of models on imbalanced datasets.

In addition to accuracy, precision, recall, F1-score, and confusion matrix can also be used as evaluation methods[10]. Precision, recall, and F1-score are widely used in binary classification problems. Although vehicle driving behavior recognition is a multi-category problem, we can regard a certain category as a positive category and other categories as negative categories, and then calculate the precision and recall of each category through Equations (1) and (2) respectively.

$$\text{Precision}=\frac{TP}{TP+FP} \tag{1}$$

$$\text{Recall}=\frac{TP}{TP+FN} \tag{2}$$

where, TP represents the number of positive samples that are correctly predicted as the positive class, and FP represents the number of negative samples that are incorrectly predicted as the positive class, FN represents the number of positive samples that are incorrectly predicted as the negative class.

The F1-score can be the harmonic average of the precision and recall, which can comprehensively evaluate the performance of the model. The closer the F1-score is to 1, the better the model performance is. The F1-score[11] can be calculated by Equation (3):

$$F1=2\frac{\text{Precision}\cdot\text{Recall}}{(\text{Precision}+\text{Recall})} \tag{3}$$

In addition to introducing more evaluation metrics, weighting the losses generated by different category labels is also a way to solve data imbalance [12]. Weighted cross-entropy loss refers to using the coefficient w to describe the importance of the sample category in the loss. After increasing the weight, the difference in the impact of different numbers of categories on the loss function can be reduced, and the sensitivity of the model to small category labels can be increased, thereby improving the prediction performance of the model. The expression of the weighted loss function is shown in Equation (4).

$$J=\sum_{i}^{n} w_i[y_i\cdot\log(p_i)+(1-y_i)\cdot\log(1-p_i)] \tag{4}$$

where, i represents different categories, p is the predicted probability of model output, y is the true label of the sample.

2) Solutions for different data ranges:

For problems of different feature dimensions, interval scaling is used to convert the feature values to between [-1,1] [13]. The main step is to scale the original data proportionally without changing the original distribution of the data, thus avoiding the introduction of new errors. The operation process is shown in Equation (5):

$$x_{norm}=\frac{x-x_{mean}}{x_{max}-x_{min}} \tag{5}$$

where, x is the feature value in the data, x_{norm} is the normalized feature value, x_{mean} is the mean

value of the feature, x_{max} and x_{min} are the maximum and minimum values of the feature respectively. After the data is normalized, the gradient descent process will be smoother and the descent speed will be faster.

3) Solution to missing values:

Missing values are common to be found in the dataset. For example, the forward behavior labels in the INTERACTION dataset have some missing values. Fill this part of missing values with -1.

The characteristics of data can be divided into numerical characteristics and discrete characteristics. For discrete features, if they are input directly into the network, the network will process them in the same way as numerical features, which will introduce new errors and have a negative impact on model performance. Therefore, before sending discrete features to model training, they need to be mapped to Euclidean space through Onehot encoding and converted into sparse vectors composed of 0 and 1.

4) Solution to High-dimensional problem:

Onehot coding [14][15] is one of the classic methods for processing discrete data. The feature lane ID in this paper's data is a categorical variable with 86 distinct values. In this way, the lane ID will be Onehot encoded to produce an 86-dimensional vector, where only one dimension is 1 and the other dimensions are 0. However, such high-dimensional sparse features are not friendly to model training, and these ID features have no practical significance. They only represent numbers and do not add new information, which makes model training more difficult and takes longer time to train. Therefore, in data processing, there is no need to feed the 86-dimensional ID vector directly into model training.

In this article, the ID features (road ID and lane ID) are binarized, and these category features are recoded, only to characterize whether the ID features have changed compared to the initial moment. 1 means the ID has changed, 0 means it has not changed. In this way, after Onehot encoding, the feature lane ID will be converted into a two-dimensional 0-1 vector, thus avoiding high-dimensional sparse features in model training.

The data preprocessing process is shown in Figure 2.

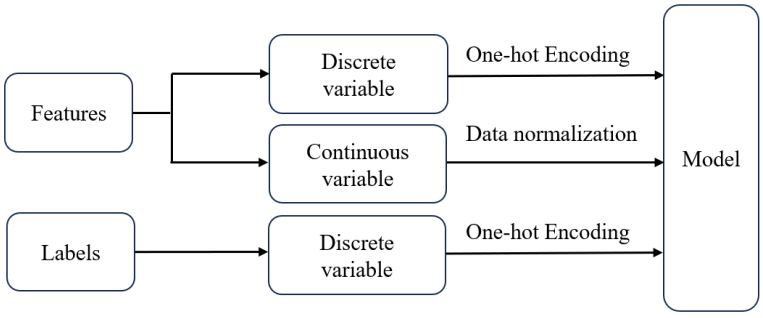


Figure 2: Data preprocessing flow chart.

3. Design and optimization of vehicle driving behavior model

The models have different structures and parameters, as well as different complexity and fitting capabilities. If the model structure is too simple, it will not be able to learn the rules of the data set, resulting in underfitting; on the contrary, if the structure is too complex, overfitting will occur, which will not only fail to improve the overall performance, but also lead to a waste of training time and computing resources. In order to determine the most suitable network structure on the vehicle driving behavior classification dataset, the following experiments were conducted.

3.1 Structural design and optimization of CNN and RNN

This section first conducts structural design and experiments on two neural network methods, FCN and ResNet. Taking FCN as an example, the F1-score is the harmonic mean of precision and recall, which can be used as an overall indicator to evaluate model performance. Table 2 lists the overall performance of the FCN network under the turning behavior label. There are three types of behavior labels for turning behavior: left turning, right turning and no turning behavior. The precisions, recalls, and F1-scores in the table are averages for each category.

It can be seen from Table 2 that as the receptive field increases, the F1-score of the model on the test machine generally shows an upward trend. Among them, the overall performance of the FCN model is the best when the receptive field is 29, reaching 94.7%. When the receptive field is 31, the model suffers from overfitting and the accuracy decreases. Therefore, the best receptive range of the model is determined to be 29.

Table 2: Performance of FCN model under different receptive fields.

Receptive field	Precision	Recall	F1-score	Accuracy
7	93.8%	92.2%	92.9%	97.3%
11	92.1%	93.8%	93.0%	96.7%
13	94.5%	93.6%	94.1%	97.6%
15	95.3%	92.7%	93.9%	97.5%
17	94.4%	92.7%	93.5%	97.4%
19	92.6%	93.7%	93.2%	97.1%
23	93.5%	93.4%	93.4%	97.3%
25	95.4%	93.2%	94.3%	97.7%
27	95.3%	93.4%	94.4%	97.8%
29	95.3%	94.1%	94.7%	97.9%
31	89.0%	87.7%	88.0%	94.3%

Table 3 shows the overall performance of the FCN model when the number of network layers is 3, 6, 9, and 12 respectively. The experiment is conducted under the same conditions: the receptive field is 29, the number of channels in the first and last layers is 128, the middle layer is 256, and 200 training batches are trained for each set of experiments. It can be seen from Table 3 that the accuracy of the model increases as the number of network layers increases, indicating that as the number of network layers deepens, the nonlinear fitting ability of the model is enhanced, and the F1-score continues to rise, indicating that the model's performance is getting better and better. The model reaches a peak when the number of layers is 9, and the F1-score drops when the number of layers is 12, indicating that a too deep network layer cannot improve model performance. Therefore, the optimal number of layers of the FCN network is 9 layers.

Table 3: Overall performance of the FCN model under different network layers.

Network layers	Precision	Recall	F1-score	Accuracy
3	95.3%	94.1%	94.7%	97.8%
6	95.8%	93.9%	94.8%	97.7%
9	96.7%	93.4%	95.0%	97.8%
12	94.3%	92.2%	93.2%	97.1%

To determine the number of convolutional layer channels, experiments were conducted under the same conditions: the receptive field is 29, the number of network layers is 9, and the training batch is 200. Experiments are divided into three sections. The standard group is named as 128/256.../128, which means that the number of channels of the convolutional layer representing the first and last

layers is 128 and the number of channels of the middle 7 layers is 256. The number of channels of the other two groups are respectively expanded by 2 times and reduced to a half. The experimental results are shown in Table 4. The F1-score of each group is selected as a measurement of the performance of the model under different channel numbers. It can be seen that the model performs best when the number of convolution kernel channels is 128/256.../128. Therefore, the optimal number of channels for the FCN model is determined as 128/256.../128 combination.

Table 5 shows the performance comparison between the optimized FCN network and the original FCN network structure. It can be seen from the table that the optimized network precision rate increased by 6%, the recall rate increased by 2.5%, and the F1-score increased by 4.2%, the accuracy increased by 2.2%. This shows that the performance of the optimized network is better than that of the original FCN network in all aspects.

Table 4: Overall performance of FCN model under different channel numbers.

Channel numbers	Precision	Recall	F1-score	Accuracy
64/128/…64	93.5%	91.7%	92.6%	96.8%
128/256…128	96.7%	93.4%	95.0%	97.8%
256/512…256	94.1%	92.2%	93.1%	97.4%

Table 5: Performance comparison of optimized FCN network and baseline FCN network.

Network	Precision	Recall	F1-score	Accuracy
Baseline FCN	90.7%	90.9%	90.8%	95.6%
Optimized FCN	96.7%	93.4%	95.0%	97.8%
Improvement	6.0%	2.5%	4.2%	2.2%

Later, the structure design and experiments are carried out on two neural network methods, LSTM and Bi-LSTM. Taking LSTM as an example, when conducting network layer experiments, the number of neurons in each layer is set to decrease by half, and the first layer is set to 128 neurons. When the number of network layers is set to 3, the F1-score of the model reaches its highest point. When the number of layers is too deep or too shallow, the model performance decreases. In order to determine the optimal number of neurons for the LSTM model on the INTERACTION dataset, the following experiments on the number of neuron channels are conducted when the number of network layers is 3. The experimental results are shown in Table 7. It can be seen that as the number of neurons increases, the precision, recall, F1-score and accuracy of the model are greatly improved. After two experiments in Table 6 and Table 7, it is finally determined that the network structure of LSTM consists of 1 layer of Masking layer and 3 layers of LSTM layer, and finally connect to the Softmax layer. The number of neurons in each layer is 1024/512/256.

Table 6: Overall performance of LSTM model under different network layers.

Network layers	Precision	Recall	F1-score	Accuracy
2	82.5%	81.4%	81.6%	91.1%
3	86.6%	81.1%	83.6%	92.6%
4	82.0%	80.2%	81.0%	91.1%

Table 7: Overall performance of LSTM model under different network layers.

Neuron numbers	Precision	Recall	F1-score	Accuracy
128/64/32	86.6%	81.1%	83.6%	92.6%
256/128/64	84.5%	87.0%	85.6%	92.8%
512/256/128	87.6%	87.1%	87.3%	93.8%
1024/512/256	90.1%	90.0%	90.2%	95.2%
2048/1024/512	90.0%	90.2%	90.1%	95.2%

Table 8: Performance comparison of optimized LSTM network and baseline LSTM network.

Network	Precision	Recall	F1-score	Accuracy
Baseline LSTM	90.9%	86.9%	88.8%	94.7%
Optimized LSTM	90.1%	90.0%	90.2%	95.2%
Improvement	-0.8%	3.1%	1.4%	0.5%

The performance of the optimized LSTM network is compared with the original LSTM network structure. As can be seen from Table 8, the accuracy of the optimized network has slightly decreased, the recall rate has increased by 3.1%, the F1-score has increased by 1.4%, and the accuracy has increased by 0.5%. This shows that the optimized network is slightly better than the original LSTM network in all aspects.

3.2 Model fusion

Convolutional neural networks and recurrent neural networks have their own advantages. CNNs can extract more advanced and abstract features, and RNNs are good at capturing dependencies in the time aspect. In the analysis of vehicle driving behavior, it is very important to mine the relationship between features and capture the dependence of behavior in the time aspect. In order to combine the advantages of the two networks to obtain a better and more stable model, model fusion is performed. As shown in Figure 3, ResNet, which performs better in convolutional neural networks, and Bi-LSTM, which performs better in recurrent neural networks, are selected for fusion. The data will be sent to two networks at the same time. The feature sequence processed by ResNet and the feature sequence processed by Bi-LSTM are integrated and spliced together, and finally connected to the softmax layer for classification. The merged network combines the advantages of both and is more stable.

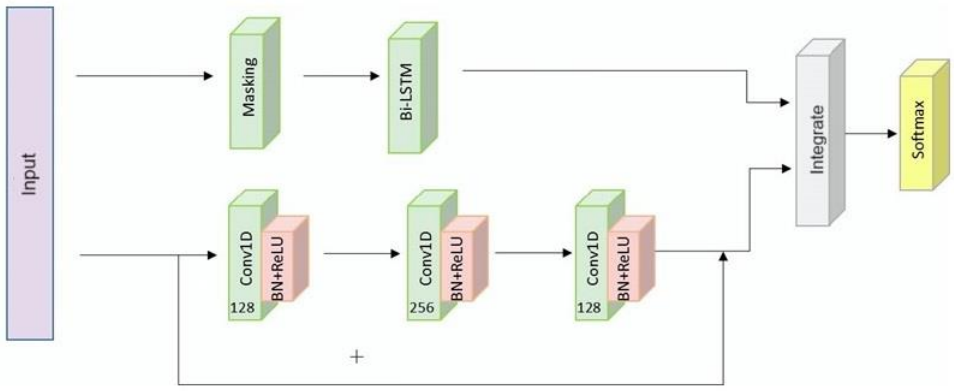


Figure 3: Network structure after model fusion.

Table 9: Performance comparison between the merged network and a single network.

Network	Precision	Recall	F1-score	Accuracy
ResNet	96.4%	94.3%	95.2%	97.9%
Bi-LSTM	94.4%	93.2%	93.8%	97.1%
Fused network	97.2%	96.0%	96.5%	98.3%

Table 9 is a comparison of the performance of the merged network and a single network. It can be seen from the table that the merged network has improved in various indicators. Among the indicators, the F1-score, which can be a more objective measure of the overall performance of the model, has increased, which shows that the performance of the fused model is better and more stable than before

the fusion.

In order to better verify the generalization ability and anti-interference ability of the model, 1%, 3%, and 5% perturbations were randomly added to different features of the test set samples. The perturbation range is determined based on the numerical range of the feature. According to Equation. 6, the characteristics after adding disturbance can be calculated:

$$X=x+(x_{max}-x_{min})\cdot Percent \cdot random.randn(0,1) \quad (6)$$

where, x represents the feature without added disturbance, x_{max} represents the maximum value of the feature, x_{min} represents the minimum value of the feature, Percent represents the increased percentage of error disturbance, and random.randn(0,1) represents a randomly generated value between 0 and 1 that satisfies the Gaussian distribution.

Table 10: Performance of the fusion model under different perturbations of the test set.

Pertubation	Precision	Recall	F1-score	Accuracy
0%	97.2%	96.0%	96.5%	98.3%
1%	97.2%	95.9%	96.5%	98.3%
3%	97.1%	95.9%	96.4%	98.2%
5%	97.0%	95.8%	96.3%	98.1%

Table 10 shows the performance of the fused model on the test set with different degrees of perturbation. It can be seen from the table that when adding 1% perturbation to the test set, the model performance basically does not change. When adding 3% perturbation and 5 % of the perturbation, the F1-score of the model dropped slightly, but it is still very close to that without perturbation, indicating that the fusion model has strong anti-interference ability

4. Vehicle Driving Behavior Model Evaluation Experiment

4.1 Experimental Procedure

Experiments are conducted under the following conditions. A single original sample in the INTERACTION data set has 14-dimensional features. After data preprocessing, a total of 17-dimensional features for each sample are sent to model training. Each model trained four models on "lane behavior", "forward behavior", "turning behavior" and "vehicle state behavior". The CNN model is trained for a total of 200 batches, and the RNN model is trained for a total of 200 batches. The learning rate is 0.0001, and the mini-batch method is used for training. The number of samples in a batch is 64. Precision, recall, F1-score, confusion matrix, and accuracy are used to evaluate the model classification performance, and the confusion matrix is calculated separately for each category.

4.2 Model evaluation experiment on the INTERACTION dataset

Table 11 shows the performance of four single models and fusion models under different driving behavior labels. It can be concluded from Table 11 that the fusion model performs stablest on the four types of behavior labels. In vehicle state behavior, there is a strong correlation between the order of behaviors. The previous state of the "stationary" behavior must be "parking", and the behavior state after "stationary" is "start". The indicators of the fusion model and the single Bi-LSTM model far exceed those of the other three single models. Both the fusion model and Bi-LSTM have the ability to extract contextual information to help predict behavior at the current moment, and are very suitable for processing vehicle state behavior that has strong correlation in both forward and backward time.

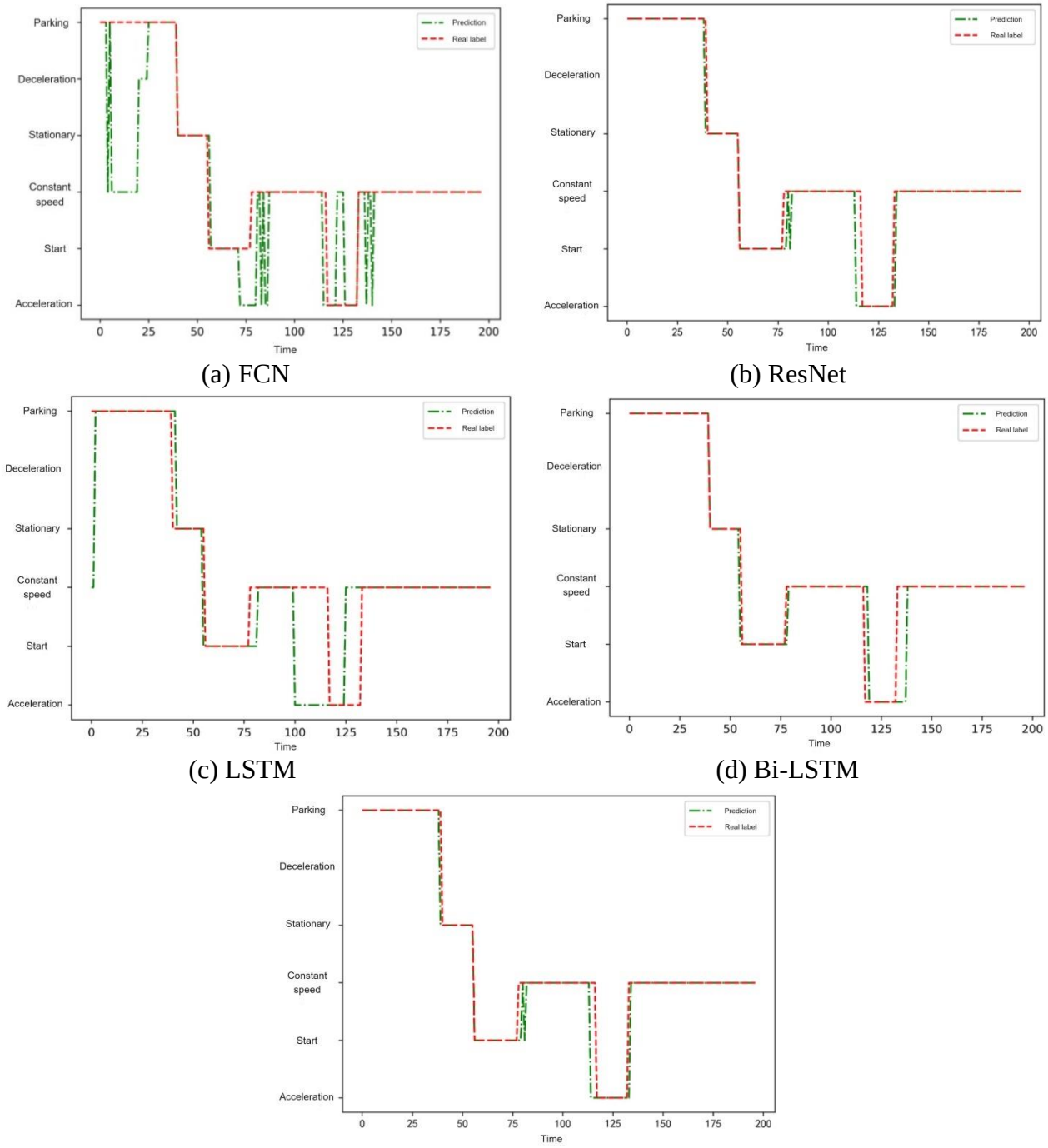
Since the fusion model combines ResNet and Bi-LSTM, it can not only capture the dependence in

the time aspect, but also extract deeper features through convolution, so the F1-score is the highest and the overall performance is the stablest. It shows that fusing convolutional neural network and recurrent neural network is a feasible method.

Table 11: Model performance in terms of vehicle state behavior.

Behavior label	Model	Precision	Recall	F1-score	Accuracy
Vehicle state behavior	FCN	70.7%	70.5%	70.3%	74.1%
	ResNet	72.4%	67.9%	69.6%	74.7%
	LSTM	68.3%	67.0%	66.9%	68.9%
	Bi-LSTM	90.9%	90.6%	90.7%	91.1%
	Fusion model	90.0%	90.7%	90.3%	90.7%
Forward behavior	FCN	91.8%	94.6%	93.1%	96.9%
	ResNet	90.5%	95.6%	93.3%	97.0%
	LSTM	90.9%	91.2%	91.1%	96.0%
	Bi-LSTM	93.1%	93.7%	93.2%	97.1%
	Fusion model	93.5%	95.3%	94.3%	97.2%
Lane behavior	FCN	66.8%	54.3%	58.8%	93.0%
	ResNet	73.2%	57.0%	62.8%	93.6%
	LSTM	63.2%	57.4%	59.9%	92.5%
	Bi-LSTM	74.9%	65.8%	69.6%	94.3%
	Fusion model	75.1%	66.4%	69.9%	94.5%
Turning behavior	FCN	96.7%	93.4%	95.0%	97.8%
	ResNet	96.4%	94.3%	95.2%	97.9%
	LSTM	90.1%	90.0%	90.2%	95.2%
	Bi-LSTM	94.4%	93.2%	93.8%	97.1%
	Fusion model	97.2%	96.0%	96.5%	98.3%

Figure 4 shows the performance differences of these five models under vehicle state behavior. In Fig. 4a and Fig. 4b, the results of the CNN model oscillate between maintaining vehicle speed and accelerating behavior (around the 75th time step and the 125th time step). There is a certain similarity in the data characteristics of these two behaviors during the transition because the CNN model cannot use contextual information to help prediction, and can only classify driving behaviors based on the data at the current moment. Therefore, it is susceptible to the influence of data disturbance and frequent oscillation occur. In the RNN model, although the prediction result of the LSTM in Fig. 4c is relatively stable, the overall performance is not as good as other models because of the serious "delayed prediction" phenomenon. In Fig. 4d, Bi-LSTM can capture the before-and-after dependencies in behavioral data effectively, and the "delay phenomenon" is not obvious. As can be seen from Fig. 4e, because the fusion model has the characteristics of both ResNet and Bi-LSTM, there is a certain "oscillation" phenomenon near the 75th time step, and a certain "delayed prediction phenomenon" near the 115th time step. Both phenomena are relatively minor, and the fusion model is relatively stable overall.



(a) FCN (b) ResNet

(c) LSTM (d) Bi-LSTM

(e) Fusion model

Figure 4: Comparison of the results of model predicted behavior labels and real behavior labels on vehicle status behavior

5. Conclusions

This paper studies deep learning methods that can be applied to vehicle driving behavior classification, using two research ideas: one is to use convolutional neural networks as strong feature extractors to classify by mining features; the other is to treat driving behavior data as For time series, a recurrent neural network is used to classify the time series. The conclusions are as follows:

1) This article solves the problem of class imbalance in the driving behavior dataset and the high-dimensional feature problem after one-hot encoding of ID class features. Variable-length samples can also be trained at the same time, improving the efficiency of model training.

2) This paper successfully applies deep learning methods to classify driving behaviors, and realizes the design and optimization of convolutional neural networks such as FCN and ResNet, and recurrent neural network structures such as LSTM and Bi-LSTM. The optimized model has greatly improved various evaluation indicators such as precision, recall and F1-score.

3) Combining the advantages of convolutional neural networks and recurrent neural networks, model fusion is performed to obtain a more powerful and stable model and improve the performance of the model.

4) The anti-interference ability and generalization ability of the fusion model are verified. The results show that the fusion model has strong anti-interference ability, indicating that fusion of convolutional neural network and recurrent neural network is a feasible method.

References

[1] Ahmed Mobyen Uddin,Begum Shahina. Convolutional Neural Network for Driving Maneuver Identification Based on Inertial Measurement Unit (IMU) and Global Positioning System (GPS) [J]. Frontiers in Sustainable Cities, 2020.

[2] Lattanzi Emanuele, Castellucci Giacomo,Freschi Valerio. Improving Machine Learning Identification of Unsafe Driver Behavior by Means of Sensor Fusion[J]. Applied Sciences, 2020, 10(18).

[3] Gerdes A. Driver manoeuvre recognition[C]//Proc. 13th World Congress on ITS. 2006.

[4] Xie F, Wang R J, Shen S B, Sun R, Zhang B, Liu X X. Detecting driving behaviors by smartphone inertial sensors based on multi-feature convolutional neural network[J]. Journal of Chinese Inertial Technology, 2019,27(03):288-294.

[5] Chu J H, Zhang S, Tang W H, Lyu W. Driving Behavior Recognition Method Based on Tutor-Student Network[J]. Laser & Optoelectronics Progress, 2020,57(06):211-218.

[6] Xue Q W, Jiang Y M, Lu J. Risky Driving Behavior Recognition based on Trajectory Data[J]. China Journal of Highway and Transport, 2020,33(06):84-94.

[7] Zhan W, Sun L, Wang D, et al. Interaction dataset: An international, adversarial and cooperative motion dataset in interactive driving scenarios with semantic maps [J]. arXiv preprint arXiv:1910.03088, 2019.

[8] Potdar K, Pardawala T S, Pai C D. A comparative study of categorical variable encoding techniques for neural network classifiers [J]. International journal of computer applications, 2017, 175(4): 7-9.

[9] Tan M S, Lyu X, Ding L, Li X J, Deep denoising autoencoder anomaly detection method based on elastic net[J]. Computer Engineering and Design, 2020,41(06):1516-1521.

[10] Goutte C, Gaussier E. A probabilistic interpretation of precision, recall and F-score, with implication for evaluation[C]//European conference on information retrieval. Springer, Berlin, Heidelberg, 2005: 345-359

[11] Powers D M W. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation [J]. arXiv preprint arXiv:2010.16061

[12] Kotsiantis S, Kanellopoulos D, Pintelas P. Handling imbalanced datasets: A review [J]. GESTS International Transactions on Computer Science and Engineering, 2006, 30(1): 25-36

[13] Sola J, Sevilla J. Importance of input data normalization for the application of neural networks to complex industrial problems [J]. IEEE Transactions on nuclear science, 1997, 44(3): 1464-1468.

[14] Chen Y C, Liu W. Opinion extraction and clustering of students' teaching evaluation text based on sentiment analysis[J]. Journal of Computer Applications, 2020,40(S1):113-117.

[15] Tao Z L, Song G G, Huang X L. Survey of Crucial Technologies on CTR Prediction[J]. Journal of Communication University of China (Science and Technology), 2019,26(06):72-75+79.