Signal, Image and Video Processing (2020) 14:617–624
https://doi.org/10.1007/s11760-019-01589-z

ORIGINAL PAPER

Distracted driver classification using deep learning

Munif Alotaibi[1] · **Bandar Alotaibi**[2]

Received: 12 November 2018 / Revised: 5 October 2019 / Accepted: 21 October 2019 / Published online: 6 November 2019
© Springer-Verlag London Ltd., part of Springer Nature 2019

Abstract
One of the most challenging topics in the field of intelligent transportation systems is the automatic interpretation of the driver's behavior. This research investigates distracted driver posture recognition as a part of the human action recognition framework. Numerous car accidents have been reported that were caused by distracted drivers. Our aim was to improve the performance of detecting drivers' distracted actions. The developed system involves a dashboard camera capable of detecting distracted drivers through 2D camera images. We use a combination of three of the most advanced techniques in deep learning, namely the inception module with a residual block and a hierarchical recurrent neural network to enhance the performance of detecting the distracted behaviors of drivers. The proposed method yields very good results. The distracted driver behaviors include texting, talking on the phone, operating the radio, drinking, reaching behind, fixing hair and makeup, and talking to the passenger.

Keywords Distracted drivers · Deep learning · Convolutional neural network · Inception

1 Introduction

As the amount of traffic density increases, the number of car crashes is expected to further increase. The World Health Organization (WHO) issued the 2015 Global Status Report, which indicated that nearly 1.25 million deaths are estimated to have occurred globally each year because of car accidents [1,2]. Hazardous and risky driving behavior causes the deaths of more than a million people and 50 million serious injuries globally every year [3,4]. The National Highway Traffic Safety Administration (NHTSA) states that, in 2015, car accidents involving distracted driving caused the deaths of 3477 people and 391,000 people were seriously injured. The main reason for the reported car crashes was texting or talking on the phone while driving [5]. Distracted driving is defined by NHTSA as "any activity that diverts attention from driving". Distracted driving includes: eating, drinking, talking to passengers, texting, talking on the phone, and adjusting the stereo, drowsy driving, navigation system, or entertainment [5,6].

A clearer definition of distracted driving is provided by the Centers for Disease Control and Prevention (CDC), which categorizes distracted driving into three classes: visual (i.e., looking around, and not concentrating visually on the road), cognitive (i.e., looking at the road, but not concentrating mentally on the road), and manual (i.e., taking the driver's hands off the car's wheel) [2]. The amount of car accident reduction caused by distracted driving and the improvement in traffic safety using smart vehicles equipped with distracted driver's postures detectors have become the first priority of many governments and car manufacturers. Additionally, to increase road safety, police officers or radar cameras have to be supplied for use with such distracted driving detectors to penalize the law offender.

In this research, we consider the manual category in which the distractions are in the forms of texting, talking on the phone, eating or drinking, reaching behind, adjusting the stereo, entertainment, or GPS, and fixing hair and makeup. We improved the performance of distracted driving using an ensemble of convolutional neural networks architectures and a hierarchical recurrent neural network (HRNN). The rest of the paper is arranged as follows. Section 2 reviews the related work. Section 3 explains deep learning algorithms. Section 4

Thanks to the title.

✉ Munif Alotaibi
munif@su.edu.sa

Bandar Alotaibi
balotaibi@ut.edu.sa

[1] College of Computing and Information Technology, Shagra University, Shagra, Saudi Arabia

[2] College of Computer Science and Information Technology, University of Tabuk, Tabuk, Saudi Arabia

presents the datasets information. Section 5 presents our proposed method. Section 6 introduces the experimental results. Section 7 discusses our results. Section 8 concludes the paper.

2 Related work

Distracted driver classification is classified into two main categories. The first category uses wearable sensors to measure physiological and biomedical signals such as brain activity, vascular and muscular activities, and heart rate. These methods [7,8] have some disadvantages such as hardware cost and user involvement. The second category to classify distracted drivers employs a camera. This category consists of three vision-based techniques to monitor distracted behaviors in real-time, namely, head pose [9–11] gaze detection [12,13], fatigue cues extraction from the driver's face [14–16], and body postures characterization (e.g., arms, feet, and hands positions) [17,18].

Most of the vision-based approaches employ a two-step structure in which the features are extracted from the raw data using hand-crafted methods and classifiers are fitted based on the hand-crafted features. Approaches that follow the two-step architecture cannot attain an optimal trade-off between the robustness of the trained classifier and the distinctively hand-crafted features. In the last two decades, vision-based approaches to detect distracted drivers that are based on support vector machines (SVMs) and decision trees have dominated the research area. Lately, with the great success of deep learning models, particularly the convolutional neural networks (CNNs) in computer vision [19–22], natural language processing [23], and speech recognition [24,25], these deep learning models have become the dominant approach to solve the distracted driving problem as well.

Yan et al. [4] proposed an approach to recognize and detect driving posture based on deep CNN. The approach applies local neighborhood operations and trainable filters to select meaningful features automatically. The advantage of this approach is the ability to learn meaningful features with minimal domain knowledge, which could provide the model with an improved performance over models that used the hand-crafted features employed in previous works. The authors also pre-trained the filters using a sparse filter [26] to accelerate the training for faster convergence and better generalization. The authors tested some activation functions and pooling operations and found that the best activation function is the rectifier linear unit (ReLU) and the best pooling operation is the max-pooling technique.

Aboulenaga et al. [2] proposed a method to recognize manual driver distractions, including: texting while driving, using the cell phone, drinking, eating, adjusting the radio, reaching behind, and fixing hair and makeup. The authors proposed technique combines two well-known CNN architectures, namely, AlexNet [19] and Inception V3, into a genetically weighted ensemble. The inputs to the model are raw images, face images, hands images, face and hands images, and skin-segmented images. The model was then trained on the input images to recognize the distraction behavior. The authors pre-trained the model using transfer learning (i.e., the ImageNet model) for faster convergence. Then, they obtained the final class distribution by evaluating the weighted sum of all networks' outputs. They also developed a genetic algorithm to evaluate the weights. Table 1 lists state-of-the-art methods, the dataset used with each method, and the overall accuracy of each method.

Table 1 Summary of state-of-the-art methods

Method	Dataset	Classes	Accuracy (%)
Baseline [27]	SEU	4	90.63
CNN [4]	SEU	4	99.78
Alex net [28]	AUC	10	94.29
GA-weighted ensemble [28]	AUC	10	95.98

3 Deep learning background

Recently, deep learning has dominated visual interpretation tasks and has shown outstanding performance. Particularly, the convolutional neural network (CNN) deep learning algorithm has achieved significant progress on image recognition tasks. Finding the perfect CNN architecture is still a very difficult task. Thus, there have been many architectures proposed in the past, such as GoogLeNet (i.e., Inception), AlexNet, VGGNet, and most recently, the deep residual network (i.e., ResNet). On the other hand, the recurrent neural network (RNN) is a well-known algorithm that obtains impressive results on time series problems as well as language tasks such as speech recognition and machine translation.

3.1 ResNet model

The ResNet model [21] was invented by Microsoft researchers in 2016. The model has achieved the state-of-art result of 96.4% in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC). The network is very deep, consisting of 152 layers. Furthermore, the ResNet model introduced unique residual blocks in which the identity skip connections are used to address training a very deep architecture approach. The purpose of the residual blocks is to copy and carry out the inputs of a specific layer to the next layer. The vanishing gradients issue is overcome by the identity skip connection step, which guarantees that the next layer trains on some-

thing other than the input that the layer is familiar with. In addition to the ResNet success in the ILSVRC, ResNet has shown impressive results on many computer vision tasks.

3.2 Inception model (GoogleNet)

The GoogleNet model [29] is a deep CNN network that was proposed in 2014 by Google researchers, which eventually achieved top-5 accuracy of 93.3% in the ILSVRC. GoogLeNet architecture is deep, consisting of 22 layers. GoogLeNet architecture was created upon a novel building block called the Inception model. This architecture uses a network in the network layer instead of using the typical sequential process. The architecture uses parallel computing to compute a large convolutional layer, a small convolutional layer and a pooling layer. The architecture performs a one-by-one convolution operation to reduce the dimensionality of the features. The number of parameters and operations is reduced significantly because of the dimensionality reduction used in this architecture and the parallelism that has been introduced; therefore, these features save memory and minimize the computational cost [30].

3.3 Hierarchical multiscale recurrent neural network

The hierarchical multiscale recurrent neural network (HM-RNN) was proposed by University of Montreal researchers in 2017 [31]. HM-RNN utilizes temporal data to learn the hierarchical multiscale structure (i.e., it does not use explicit boundary information). The model adaptively designates adequate update rates identical to the layers abstraction levels, instead of assigning fixed update times. The authors suggest to use a binary detector at each layer for coarse timescales and fine timescales. For high-level layers, the model learns coarse timescales, and for low-level layers, it learns fine timescales. At the time step in which the segment of the corresponding abstraction level is totally executed, the boundary detector is turned on. Otherwise, during the segment execution, the boundary detector remains off. The authors introduced three operations utilizing the hierarchical boundary states (i.e., COPY, UPDATE, and FLUSH). One of these operations can be used at each time step. The UPDATE operation is different from update rule of the long short-term memory (LSTM) [32] because it is sparsely processed based on the detected boundaries.

4 Dataset information

One of the first available datasets for driver distraction classification is the StateFarm[1] dataset. The dataset was released on Kaggle[2] in 2016. The dataset consists of ten classes to be classified, namely, normal driving, texting while driving using the right hand, talking to someone on the phone using the right hand, texting while driving using the left hand, talking to someone on the phone using the left hand, reaching for the dashboard to operate the radio, drinking or eating, reaching behind, fixing hair and makeup, and talking to a

Fig. 1 Some samples of the dataset that represent the ten classes

[1] An insurance company; its headquarters are located in Bloomington, IL, USA

[2] A platform for data science and predictive models competitions

Table 2 Summary details of the 1st dataset

Class	Description	Images
0	Normal driving	2489
1	Texting while driving using right hand	2267
2	Talking to somebody on the phone using right hand	2317
3	Texting while driving using the left hand	2346
4	Talking to somebody on the phone using left hand	2326
5	Reaching the dashboard to operate the radio	2312
6	Drinking or eating	2325
7	Reaching behind	2002
8	Fixing hair and makeup	1911
9	Talking to passenger	2129
Sum		33,636

passenger. The submission to the StateFarm competition was evaluated using the multiclass logarithmic loss. A true target is attached to each image, and the goal is to submit a set of predicted probabilities for each image (Fig. 1).

The original published dataset has 2 folders, which are the testing and the training data, and we use only the data in the training folder to evaluate our method since the data in the testing folder is unlabeled. Thus, we used 33,636 images; more details about the dataset that we used are shown in 2.

To further evaluate the strength and generalization performance of our approach, we also used another recent dataset [28] from the American University in Cairo (AUC). The AUC dataset has 44 individuals, 29 males and 15 females, from seven countries: the USA, Egypt, Germany, Uganda, Canada, Morocco, and Palestine. The videos were taken at different times of day, in different cars, with different driving conditions, and with the drivers wearing different clothes. The dataset consists of 14,478 frames distributed over 10 classes, as shown in Table 3. The dataset is divided into 75% for training and 25% for testing.

5 Proposed method

In this paper, we take advantage of three advanced deep learning models, namely, the residual network (ResNet), the hierarchical recurrent neural network (HRNN), and the Inception architecture by combining them into one model. In our model shown in Fig. 3, there is one ResNet block and two HRNN layers integrated with the Inception module, and they are followed by two dense layers and finally by the softmax classifier. The ResNet has two convolutional layers. The details of each of these entities are explained in the following sections.

Table 3 Summary details of the AUC dataset

Class	Description	Size (frames)
0	Safe driving	2986
1	Phone right	1256
2	Phone left	1320
3	Text right	1718
4	Text left	1124
5	Adjusting radio	1123
6	Drinking	1076
7	Hair or makeup	1044
8	Reaching behind	1034
9	Talking to passenger	1797
Sum		14,478

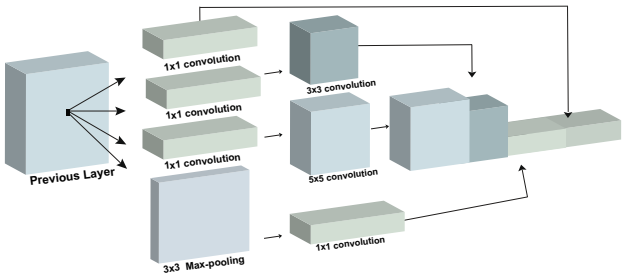


Fig. 2 The original inception module

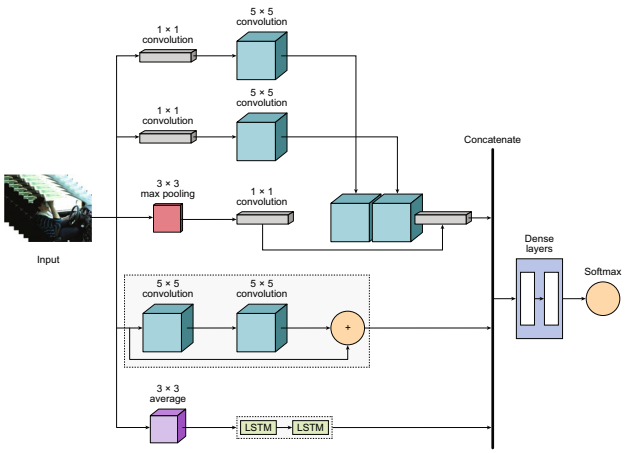


Fig. 3 Our modified version of the inception module

5.1 Inception module

We use one Inception module that is similar to the original Inception module used in the GoogLeNet model. However, in addition to the entities of the original Inception module as shown in Fig. 2, we add two entities: one ResNet block and two LSTM layers (the HRNN network) as shown in Fig. 3. The max-pooling in our Inception has a size of 3×3 with a stride of 1. The figure also shows the size of each filter (kernel) in each convolutional layer.

Some convolutional layers in the Inception performs a one-by-one convolution operation to reduce the dimensionality of the features. The size of the filters in the other two convolutional layers is 5×5. Each convolutional layer has seven kernels. The output of each one of the convolutional kernels has the same length as the original input (same padding). Each input image has RGB color channels. Thus, we are using 2D convolutional layers.

5.2 Our ResNet

There is one ResNet block that has two convolutional layers. The raw data of the image is fed to the first convolutional layer, which convolves the input volume with three filters; each one of them has a size of 5×5. The output feature maps, which have the same length as the original input (same padding), are fed into the second layer, which has the same settings as the first layer. Then, we sum up the output features with the input and feed them to the next step. The operation of the filter in each convolutional layer is summarized in Eq. 1:

$$X^i = (\phi(W^i \Theta X^{i-1} + \beta^i)) \tag{1}$$

where X^{i-1} is the input, and X^i is the output feature map. Note that each convolutional layer will output more than one feature map. b is the bias term β, φ is the activation function, and Θ is the convolution operator. The ReLU activation function φ applies an elementwise operation on the input data x. It is defined as the following equation:

$$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$$

Then, the output of the residual block is fed to the average pooling layer. We apply the average pooling of size 2 with a stride of 2 to the data.

The operation of the residual block has two convolutional layers as defined in Eq. 2

$$\begin{aligned} x^1 &= F(x^0) \\ x^2 &= F(x^1) \\ x^3 &= (x^0 + x^2) \end{aligned} \tag{2}$$

where F(x) represents the operation of each convolutional layer, and x^i represents the input and output feature maps.

5.3 Our HRNN

Before feeding the data to the HRNN, we use average pooling to reduce the dimensionality of the data. In the hierarchical RNN, there are two long short-term memory (LSTM) layers. The first one will encode every column of the data that has a shape of (26, 80) to a column vector with a shape of (80). The second one will encode these 80 column vectors of shape (80, 80) to a vector that represents all of the data. More details of the structure of our HRNN are shown in Table 4.

Table 4 Summary details of the HRRN

Layer	Outputtensorsize	Parameters number
1stLSTMlayer	(26, 80)	26880
2edLSTMlayer	(80)	51520

Finally, we concatenate all the outputs of the three entities and feed them to two dense layers, each with 80 neurons. Then, we use the softmax function, which is performed on the obtained output of the dense layers to classify the input image to one of the ten classes.

We use a batch size of 80 and set the number of epochs to 30. We use the Adam optimizer for our model. We set the initial adaptive learning rate to 0.001, beta_1 to 0.9, and beta_2 to 0.999. For every layer in our model, we use glorot uniform (Xavier uniform) [33] to initialize the weights matrix. We initialize all the bias terms of the convolutional layers to zero.

The proposed model was implemented using the Keras library [34]. It is trained and tested using a computer with an Intel core (TM) i7 CPU @ 2.00 GHz, 16.0 GB of RAM and a 64-bit Windows 10 operating system.

6 Experiment and results

To evaluate the performance of our proposed method in detecting distracted actions of drivers, we used the State Farm Distracted Driver Detection dataset as provided by Kaggle. We measure and evaluate the performance using the overall accuracy. The overall accuracy is calculated as the number of all images that are classified correctly divided by the total number of all samples in the test set.

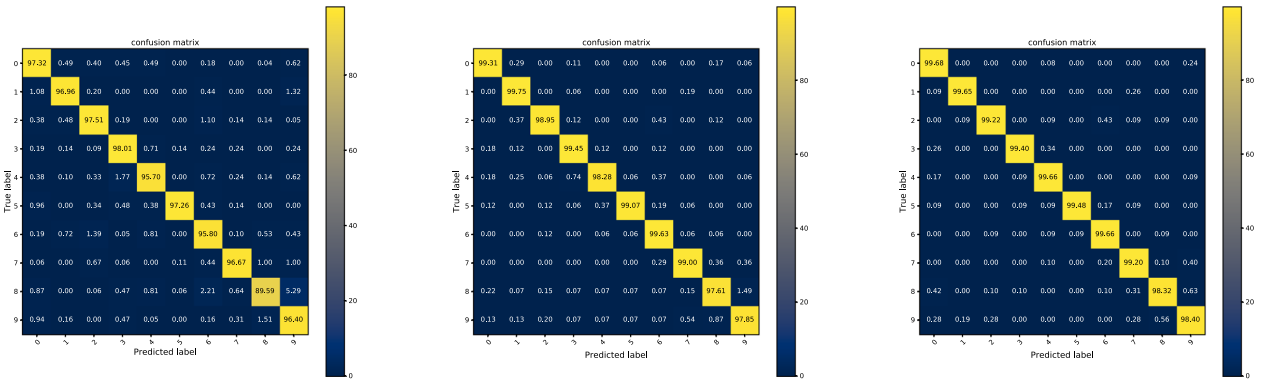
We used several percentages to divide the data into testing and training sets. We used 10%, 20%, and 30% for the training and the remaining for the testing. The overall accuracies of our method when applied on the dataset using 10%, 20%, and 30% of the data as training are shown in Table 5.

The confusion matrix of our proposed deep learning architecture when using 10% of data for training is shown in Fig. 4a. Figure 4b shows the confusion matrix of our proposed deep learning architecture when using 30% of the data for training. The accuracy increased when we increased the size of the training data. Figure 4c shows the confusion matrix of our proposed deep learning architecture when using 50% of the data for training. The accuracy is 99.30%.

Our model can smoothly converge to the local minima within a few epochs. Figure 5a shows how our model mini-

Table 5 Overall accuracies of our method when applied on the dataset using different training split percentages

Training (%)	Training samples	Testing samples	Accuracy (%)
10	2242	20,182	96.23
30	6727	15,697	98.92
50	11,212	11,212	99.30

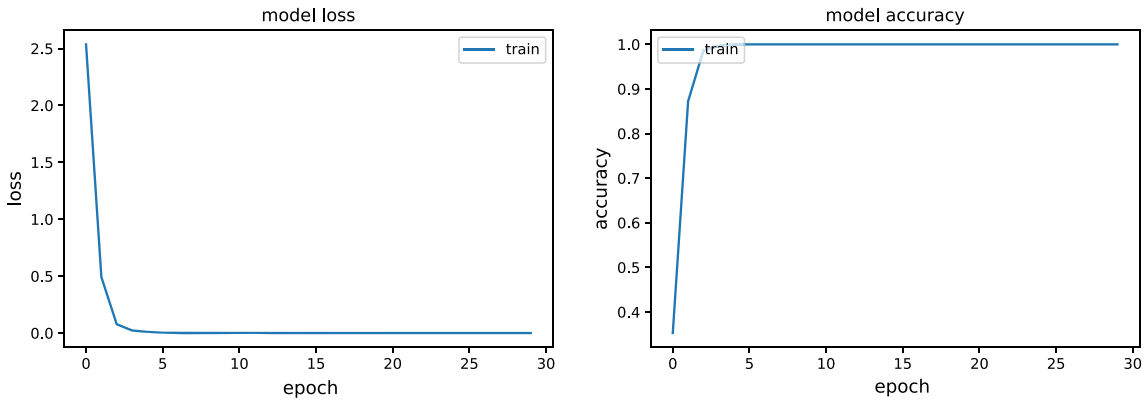


(a) 10% of the data as training and the rest for testing

(b) 30% of the data as training and the rest for testing

(c) 50% of the data as training and the rest for testing

Fig. 4 The confusion matrix of our proposed deep learning architecture using three different percentages to divide the State Farm Distracted Driver dataset into testing and training sets



(a) Loss

(b) Accuracy

Fig. 5 **a** Loss and **b** accuracy during the convergence of our model

mizes the loss function during the training phase in order to optimize the parameters of the network. In addition, Fig. 5b shows the accuracy of the training data during the convergence steps of our model.

Moreover, We also applied our method to the AUC distracted driver dataset [28]. We re-sized each image to 67×120 pixels. We used 75% of the dataset for training and the remainder for testing. The results are shown in Table 6. Furthermore, Table 6 shows that our model has promising results.

7 Discussion

From the experiment, we can conclude that our model is capable of learning richer representations with a small number of parameters. It was able to achieve very accurate results particularly when there is sufficient training data.

The motivation for the proposed model is to find an easier and more efficient model by combining several techniques that have already been proven to be very effective. Moreover, we compare the performance of the proposed method with the ResNet model as shown in Table 6. In this table, we use State Farm Distracted Driver dataset. We use two residual

Table 6 Overall accuracies of our method when applied on to State Farm distracted driver and AUC datasets

Method	Database	Training	Testing	Accuracy (%)
Our method	StateFarm	2242	20,182	96.23
ResNet [21]	StateFarm	2242	20,182	95.31
HRNN [31]	StateFarm	2242	20,182	98.34
ResNet + HRNN	StateFarm	2242	20,182	91.72
Our method	AUC	12,977	4331	92.36
ResNet [21]	AUC	12,977	4331	88.52
HRNN [31]	AUC	12,977	4331	84.85

Table 7 Computation time during the testing phase (AUC dataset)

Method	Time (ms)
Ours	114
ResNet	62
HRNN	71

blocks, followed by the average pooling layer. We also list the result of the HRNN alone without any convolutional layer. We use 10% of the data for training and the remaining 90% for testing. We also list the result of one ResNet Block followed by the HRNN.

We noted that the larger architectures such as Xception, Inception, VGG and ResNet50 can not be optimized easily on this dataset. For example, the Inception requires fine-tuning techniques (i.e., transfer learning) . We found that using a few convolutional layers is very optimal for our model. When we increase the number of layers, the accuracy decreases.

When we apply our method to the AUC distracted driver dataset, the result shows that our method can perform better than both ResNet and HRNN. Our method obtained 92.36%, ResNet obtained 88.52%, and HRNN achieved 84.85%. Our method is better than ResNet by around 3.80% and better than HRNN by around 7%.

Table 7 shows the average computation time for all three models. ResNet has less computation time, with one sample taking only 62 ms during the testing phase. For HRNN, it only takes 71 ms for one image. For our model, it takes 114 ms to process one sample. Thus, our model requires more computation time when compared to both ResNet and HRNN, and this is due to our model's components. However, the results demonstrate that all the methods can maintain real-time performance.

8 Conclusions

The automatic recognition of the driver's behavior is one of the challenging problems in ITS. In the last decade, as smartphones have become prevalent worldwide, many car crashes caused by distracted drivers have also occurred. We investigate the distracted driver posture as a part of human action recognition to recognize the driver's behavior. Our objective is to improve the accuracy in the distracted driver classification problem. We propose a method that combines three of the most advanced models in deep learning, namely, the residual network, the Inception module and the hierarchical recurrent neural network to improve the performance of detecting a distracted driver's behavior. We test our approach using two datasets (i.e., State Farm's dataset on the Kaggle platform and AUC dataset). The images in the datasets were taken using a dashboard camera to detect distracted drivers from the 2D images. The proposed method achieved promising results.

References

1. World Health Organization: World Health Organization. Management of Substance Abuse Unit. Global Status Report on Alcohol and Health, 2014. World Health Organization, Geneva (2014)
2. Abouelnaga, Y., Eraqi, H.M., Moustafa, M.N.: Real-time distracted driver posture classification. arXiv preprint arXiv:1706.09498 (2018)
3. Peden, M.: World Report on Road Traffic Injury Prevention. World Health Organization, Geneva (2004)
4. Yan, C., Coenen, F., Zhang, B.: Driving posture recognition by convolutional neural networks. IET Comput. Vis. **10**(2), 103–114 (2016)
5. National Highway Traffic Safety Administration.: 2015 motor vehicle crashes: overview. In: Traffic Safety Facts Research Note, pp. 1–9 (2016)
6. Resalat, S.N., Saba, V.: A practical method for driver sleepiness detection by processing the EEG signals stimulated with external flickering light. Signal Image Video Process. **9**, 1751–1757 (2015)
7. Craye, C., Karray, F.: Driver distraction detection and recognition using RGB-D sensor. arXiv preprint arXiv:1502.00250 (2015)
8. Fernández, A., Usamentiaga, R., Carús, J.L., Casado, R.: Driver distraction using visual-based sensors and algorithms. Sensors **16**(11), 1805 (2016)
9. Watta, P., Lakshmanan, S., Hou, Y.: Nonparametric approaches for estimating driver pose. IEEE Trans. Veh. Technol. **56**(4), 2028–2041 (2007)
10. Murphy-Chutorian, E., Trivedi, M.M.: Head pose estimation and augmented reality tracking: an integrated system and evaluation for monitoring driver awareness. IEEE Trans. Intell. Transp. Syst. **11**(2), 300–311 (2010)

11. Teyeb, I., Jemai, O., Zaied, M., Amar, C.B.: A drowsy driver detection system based on a new method of head posture estimation. In: International Conference on Intelligent Data Engineering and Automated Learning, pp. 362–369, September 2014. Springer, Cham (2014)
12. Doshi, A., Trivedi, M.M.: On the roles of eye gaze and head dynamics in predicting driver's intent to change lanes. IEEE Trans. Intell. Transp. Syst. **10**(3), 453–462 (2009)
13. Teyeb, I., Jemai, O., Zaied, M., Amar, C.B.: A novel approach for drowsy driver detection using head posture estimation and eyes recognition system based on wavelet network. In: The 5th International Conference on Information, Intelligence, Systems and Applications, IISA 2014, pp. 379–384, July 2014. IEEE (2014)
14. Bergasa, L.M., Nuevo, J., Sotelo, M.A., Barea, R., Lopez, M.E.: Real-time system for monitoring driver vigilance. IEEE Trans. Intell. Transp. Syst. **7**(1), 63–77 (2006)
15. Jemai, O., Teyeb, I., Bouchrika, T.: A novel approach for drowsy driver detection using eyes recognition system based on wavelet network. Int. J. Recent Contrib. Eng. Sci. IT (iJES) **1**(1), 46–52 (2013)
16. Lei, J., Han, Q., Chen, L., Lai, Z., Zeng, L., Liu, X.: A novel side face contour extraction algorithm for driving fatigue statue recognition. IEEE Access **5**, 5723–5730 (2017)
17. Cheng, S.Y., Park, S., Trivedi, M.M.: Multi-spectral and multi-perspective video arrays for driver body tracking and activity analysis. Comput. Vis. Image Underst. **106**(2–3), 245–257 (2007)
18. Tran, C., Doshi, A., Trivedi, M.M.: Modeling and prediction of driver behavior by foot gesture analysis. Comput. Vis. Image Underst. **116**(3), 435–445 (2012)
19. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
20. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
21. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
22. Soon, F.C., Khaw, H.Y., Chuah, J.H., Kanesan, J.: Vehicle logo recognition using whitening transformation and deep learning. Signal Image Video Process. **13**, 111–119 (2019)
23. Hu, B., Lu, Z., Li, H., Chen, Q.: Convolutional neural network architectures for matching natural language sentences. In: Advances in Neural Information Processing Systems, pp. 2042–2050 (2014)
24. Abdel-Hamid, O., Mohamed, A.R., Jiang, H., Penn, G.: Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition. In: 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4277–4280, March 2012. IEEE (2012)
25. Abdel-Hamid, O., Mohamed, A.R., Jiang, H., Deng, L., Penn, G., Yu, D.: Convolutional neural networks for speech recognition. IEEE/ACM Trans. Audio Speech Lang. Process. **22**(10), 1533–1545 (2014)
26. Ngiam, J., Chen, Z., Bhaskar, S.A., Koh, P.W., Ng, A.Y.: Sparse filtering. In: Advances in Neural Information Processing Systems, pp. 1125–1133 (2011)
27. Zhao, C.H., Zhang, B.L., He, J., Lian, J.: Recognition of driving postures by contourlet transform and random forests. IET Intell. Transp. Syst. **6**(2), 161–168 (2012)
28. Eraqi, H.M., Abouelnaga, Y., Saad, M.H., Moustafa, M.N.: Driver distraction identification with an ensemble of convolutional neural networks. J. Adv. Transp. (2019). https://doi.org/10.1155/2019/4125865
29. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–9 (2015)
30. Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., Garcia-Rodriguez, J.: A review on deep learning techniques applied to semantic segmentation. arXiv:1704.06857 (2017)
31. Chung, J., Ahn, S., Bengio, Y.: Hierarchical multiscale recurrent neural networks. arXiv preprint arXiv:1609.01704 (2016)
32. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
33. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the 13th International Conference on Artificial Intelligence and Statistics, pp. 249–256 (2010)
34. Chollet, F., et al.: Keras. https://keras.io (2015). Accessed 8 Aug 2018

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.