Research on Facial Micro-Expressions Based on Deep Learning for Driver Fatigue Detection

Wentao Wang *

College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao City, Shandong Province, 266590, China

* Corresponding Author Email: fangkefan@ldy.edu.rs

Abstract. Detecting driver sleepiness is crucial for preventing accidents. The necessity for more sophisticated solutions is underscored by the fact that conventional detection methods frequently depend on physical indicators or self-reporting, which may be intrusive or inaccurate. This research employs deep learning architectures such as EfficientNetB0, VGG16 and ResNet50 to identify driver weariness using the Driver Drowsiness Dataset (DDD). The DDD is a comprehensive dataset of facial photographs taken under diverse real-world driving scenarios, offering a solid basis for model training and assessment. The study workflow encompasses data augmentation, image preprocessing, facial micro-expression extraction and model training. According to the comparative analyses of models, EfficientNetB0 achieves high accuracy, ensuring its reliability in detecting driver fatigue states. Additionally, the model's lightweight architecture guarantees deployment in embedded systems and in-vehicle platforms. EfficientNetB0 strikes an optimal balance among accuracy, inference duration and model size, rendering it suitable for real-time applications. This study not only demonstrates the effectiveness of EfficientNetB0 in driver fatigue detection but also underscores the importance of model efficiency in safety-critical applications. These findings highlight its potential use for detecting driver weariness in real-world scenarios, providing essential technological assistance to improve road safety.

Keywords: Deep Learning; Driver Fatigue Detection; EfficientNetB0; Real-Time Applications.

1. Introduction

Driver fatigue, an important factor leading to road traffic accidents, results in significant deaths and economic losses globally [1]. Timely interventions may be advised to mitigate these dangers solely if the driver's tiredness is identified promptly. Conventional methods for identifying driver tiredness often rely on physiological signs such as the eye closure ratio, head posture measures and facial features. However, these treatments typically induce discomfort in individuals. Moreover, under certain lighting conditions and driving behaviors, they fail to adequately record the nuances of facial expressions resulting from driver weariness.

In recent years, deep learning methodologies, particularly convolutional neural networks, have demonstrated considerable potential for addressing this issue by providing high accuracy and fast detection speed for driver fatigue detection [2, 3]. Convolutional neural networks are successful in the detection of fatigue. This study used them for facial features and can obtain real-time identification without invasion with higher accuracy [4--6]. EfficientNet, which was introduced in 2019, received a good amount of attention as it can reach state-of-the-art accuracy while maintaining low computational costs, which becomes very significant for embedded systems or real-time applications in vehicles. EfficientNetB0 was proposed as a model to effectively balance accuracy and efficiency, in contrast to popular architectures like VGG16 or ResNet50. Thus, it is an ideal architecture solution for driver monitoring systems with deployment in resource-constrained environments.

This study evaluated the efficiency level of three selected deep learning architectures, which refer to EfficientNetB0, VGG16 and ResNet50, in detecting driver fatigue on the Driver Drowsiness Dataset. In the end, this research found that the EfficientNetB0 model has the best performance and can provide a reliable solution that can reduce road accidents resulting from driving fatigue.

The content of this paper is structured as follows. Section 1 examines conventional methods for detecting driver weariness as well as contemporary approaches utilizing deep learning. Section 2

delineates the dataset, data augmentation, preprocessing procedures and model training. Section 3 presents the results of the tests and evaluation measures for the experiments. Section 4 addresses the findings, limits of the study and avenues for future research. Section 5 presents a conclusion that encapsulates the principal findings and ramifications of this research.

2. Method

2.1. Overall Framework and Workflow

This study employs the EfficientNetB0 deep learning model to detect fatigue driving states using the Driver Drowsiness Dataset (DDD). The workflow comprises several key stages, including image preprocessing, face detection, facial micro-expression feature extraction, model training and fatigue state recognition [7, 8]. The workflow is illustrated in Fig. 1.



Fig. 1 Workflow of the Fatigue Driving Detection System

2.1.1 Data Augmentation and Image Preprocessing

The DDD dataset comprises a substantial collection of facial photographs of drivers in both drowsy and alert conditions. The input images are resized to 224×224 pixels and normalized to match the input requirements of the EfficientNetB0 model. Data augmentation techniques like random rotation, translation, zooming, shearing and horizontal flipping are used in the training set to generate additional variations from existing data, which can improve generalization ability. This process is created to mimic different situations in the real world, making sure that the model can maintain stability when facing changes in lighting, orientation of face and other factors. For the validation and test sets, only standardization is applied.

2.1.2 Face Detection

Since all subsequent analyses will be based on facial regions, face detection really matters. The OpenCV's Cascade Classifier is used to detect the human faces in this study. When a face is detected, it is cropped and saved for further processing. If no face is found, the central area of the image is regarded as a fallback. This approach makes sure that whether a face is present or clear in an image, the input images will always have a consistent facial region. Only in this way, effective features can be extracted and fatigue can be accurately detected. Consistently having the facial area in view enables the model to concentrate on specific properties for detecting driver fatigue, like whether their eyes are closed or performing mouth movement.

2.1.3 Facial Micro-Expression Feature Extraction

Following preprocessing and face identification, this research utilizes EfficientNetB0 as a feature extractor for identifying micro-expressions in facial images. In order to allow the network to transform the input images into high-dimensional feature vectors, the top layer of the pre-trained EfficientNetB0 is removed. The vectors may encapsulate details like eyelids dropping or yawns, all of which reveal if a driver is drowsy. EfficientNetB0's architecture balances depth, width and resolution, which makes it a strong performer in detecting complex patterns from facial expressions.

2.1.4 Model Training

A binary classification model with fully connected layers, dropout layers and sigmoid activation function is built, which is based on the feature vectors extracted by EfficientNetB0. The purpose of this model is to differentiate between tired and awake drivers.

The training procedure is executed in batches of 32 over the course of 30 epochs. A performance evaluation is conducted on the validation set following each epoch. The model is optimized via the Adam optimizer with a starting learning rate of 0.001. The loss function employed is binary cross-entropy, which is highly effective for binary classification tasks and managing imbalanced datasets. This study employed early stopping and learning rate reduction techniques to enhance model performance and prevent overfitting. The early stopping method halts model training if validation loss fails to improve during 10 consecutive epochs and reverts to the weights of the optimal model. It employs the learning rate reduction method that decreases the learning rate by 50% if there is no reduction in validation loss over the last three epochs.

2.1.5 Fatigue State Recognition

Fresh images from the testset are subjected to a fatigue state prediction using the trained model. It is necessary to compare each prediction with the true label and assess the model's performance using measures such as F1-score, accuracy, precision and recall. The true and predicted label distributions are visualized using confusion matrices, which aids in the analysis of the model's classification performance. This stage is crucial to confirm the model's practical usefulness in real-world situations where quick and accurate driver tiredness identification is important.

2.2. Fatigue Driving Classification Overview and Dataset

Facial photographs of drivers in real-world driving situations make up the Driver Drowsiness Dataset (DDD), which was utilized in this study. The photos in the collection are mostly categorized into two groups:

One group is Drowsy. The driver is experiencing fatigue, as evidenced by signs such as eye closure, yawning, or drooping eyelids. The motorist is typically depicted as inattentive or drowsy in these images.

The other group is non-Drowsy. With wide eyes and a serious countenance, the motorist appears to be in a perfectly normal, alert condition. The motorist appears to be paying close attention and aware in these photographs.

A 70% training set, a 20% validation set and a 10% test set make up the dataset. To be more precise, there are 29,254 photos in the training set, 8,358 in the validation set and 4,181 in the test set. This division guarantees a fair assessment of the robustness and performance of the model.

2.3. Fatigue Driving Detection Model Based on EfficientNetB0

This research makes use of EfficientNetB0 as the foundation for its tired driving detection model. The model's general architecture is shown as Fig. 2. An RGB image with dimensions of 224×224 pixels are fed into the model. The EfficientNetB0 network receives the preprocessed images as input. The EfficientNetB0 network then uses a cascade of convolutional layers to extract features with high dimensionality. As the number of feature channels increases, these layers progressively decrease the input's spatial dimensions. It takes the input picture and uses it to build a global feature vector that contains all the important details.

Following feature extraction, a 256-neuron fully connected layer with ReLU activation is incorporated. To aid the suggested model in differentiating between drowsy and non-drowsy phases, this layer is intended to learn the intricate nonlinear interactions among the characteristics that have been extracted. During training, neurons are randomly deactivated using a 0.5-rate dropout layer to avoid overfitting, which enhances the model's capacity for generalization.

The last layer is a dense classification layer including a single neuron and a sigmoid activation function. This layer transforms the processed feature vector into a probability score ranging from 0 to 1, indicating the likelihood of the driver being in a drowsy state. This design is tailored for binary categorization. A threshold of 0.5 is established to classify the driver as drowsy or non-drowsy.

All things considered, the architecture makes use of fully connected and dropout layers along with EfficientNetB0's effective feature extraction capabilities to provide a strong model that can accurately identify driver weariness.

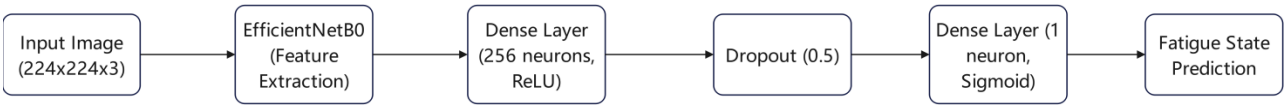


Fig. 2 Architecture for Fatigue Driving Detection Based on EfficientNetB0

3. Results

3.1. Experimental Setup and Hyperparameters

Table 1 lists the hardware and software environment configurations used in this study. The NVIDIA GeForce RTX 3070 GPU and Intel i7 processor provide sufficient computational resources for this research to ensure effective training and inference of the deep learning model. The operating system and software versions used in this study ensure consistency and compatibility within the environment. Additionally, Jupyter Notebook provides a more flexible and efficient development environment.

Table 1. Experimental Environment

Hardware/Software	Specification
CPU	Intel(R) Core(TM) i7-10700K @ 3.80GHz
GPU	NVIDIA GeForce RTX 3070
GPU Memory	8 GB
RAM	16 GB DDR4
Operating System	Windows 10 Professional 64-bit
Python Version	3.8.5
CUDA Version	11.4
cuDNN Version	8.2
Other Tools	TensorFlow 3.8.5, Keras 2.8.0, OpenCV 4.5.3, NumPy 1.19.5, Matplotlib 3.4.2

Table 2 presents the hyperparameters of the model training, which directly influence the training outcomes and performance of the model. The batch size is configured to 32 to enable the model to process a defined number of samples during each training iteration, hence optimizing training performance and memory utilization. Employing a learning rate of 0.001 is essential for facilitating smooth training and rapid convergence. The learning rate reduction technique is set with a reduction factor of 0.5 and a patience of 3 epochs, ensuring that the learning rate decreases when training performance plateaus. It facilitates improved convergence in subsequent training phases. The dropout rate is established at 0.5. It randomly deactivates 50% of neurons with each forward pass, which is crucial for mitigating overfitting. This strategy significantly enhances the model's generalization capability, particularly when training on relatively few samples or complex structures. A random seed of 42 is employed to initialize model weights, which is crucial for guaranteeing experimental reliability.

Table 2. Experimental Hyperparameters

Hyperparameter	Value
Batch Size	32
Epochs	30
Learning Rate	0.001
Learning Rate Reduction	factor = 0.5, patience = 3
Dropout Rate	0.5
Random Seed	42

3.2. Evaluation Metrics

3.2.1 True Positive (TP), True Negative (TN), False Positive (FP), False Negative (FN)

TP is the number of correct predictions where the driver is drowsy. For example, in the validation set, the model correctly predicted 4468 drowsy instances.

TN is the number of correct predictions where the driver is not drowsy. For instance, the model correctly identified 3888 non-drowsy instances in the validation set.

FP is the number of incorrect predictions where the model predicted drowsy but the driver was actually not. There was 1 false positive in the validation set.

FN is the number of incorrect predictions where the model predicted non-drowsy but the driver was drowsy. There was also 1 false negative in the validation set.

These values are useful in finding where the model is predicting wrongly and can also be used to calculate other evaluation metrics.

3.2.2 Recall

Recall, sometimes referred to as True Positive Rate or Sensitivity, gauges how well the model can detect positive occurrences or sleepy drivers. It is computed as the formula (1).

$$Recall = \frac{TP}{TP+FN} \tag{1}$$

3.2.3 Precision

Precision is the proportion of accurately anticipated positive observations to the total predicted positives. It is computed as:

$$Precision = \frac{TP}{TP+FP} \tag{2}$$

3.2.4 F1 Score

The F1 Score represents the harmonic mean of precision and recall. It is typically considered an indicator of a test's precision. The equation is:

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision+Recall} \tag{3}$$

Where recall is shown in equation (1) and precision is shown in equation (2).

3.2.5 Accuracy

Accuracy is the ratio of correctly classified cases to the total number of instances. It is computed as:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{4}$$

3.2.6 Inference Time

Inference time refers to the duration required for a model to generate predictions based on new data. It is essential for real-time applications. This metric is useful in evaluating the efficiency of the model in practical deployment scenarios.

3.3. Experimental Results

In this paper, the model performance is compared between three different models considering the following metrics: f1 score, recall, accuracy, misclassification count, inference speed, model size, parameter count and memory usage. The summarized results are shown in Table 3.

Table 3. Performance Comparison of Driver Drowsiness Detection Models

Model	F1 Score (%)	Recall (%)	Accuracy (%)	Misclassification Count	Inference Speed (sec/sample)	Model Size (MB)	Parameter Count	Memory Usage (MB)
VGG16	99.9	99.9	99.9	1	0.0482	129	21137729	1138
ResNet50	98.4	98.4	98.4	67	1.7996	148	25686913	1065
EfficientNetB0	99.9	99.9	99.9	1	1.0663	16.96	4377764	2131

Table 3 provides a comprehensive analysis of the performance metrics for each model. The VGG16 and EfficientNetB0 models both achieved high accuracies of 0.999, while ResNet50 reached an accuracy of 0.984. The misclassification count for ResNet50 was 67, whereas both VGG16 and EfficientNetB0 had only 1 misclassification. In terms of model size, VGG16 was 129 MB, ResNet50 was 148 MB, and EfficientNetB0 was the smallest at 16.96 MB. Inference speed per sample varied, with VGG16 taking 0.0482 seconds, ResNet50 taking 1.7996 seconds, and EfficientNetB0 taking 1.0663 seconds. The parameter counts also differed significantly among the models, with VGG16 having 21,137,729 parameters, ResNet50 having 25,686,913 parameters, and EfficientNetB0 having 4,377,764 parameters.

3.4. Model Performance Analysis

3.4.1 Confusion Matrix Comparison

The confusion matrices shown in Fig. 3 are more intuitive summaries of classification performance for all the models. The EfficientNetB0 model shows the best performance with only one misclassification on the testset, which indicates strong generalization ability. VGG16 is just as accurate with tiny errors. However, due to its larger model size and longer inference time, it takes away from its applicability for real-time applications. There are more misclassifications in the ResNet50 model, with 67 errors on the testset. Thus, it is less recommended for some tasks requiring high precision.

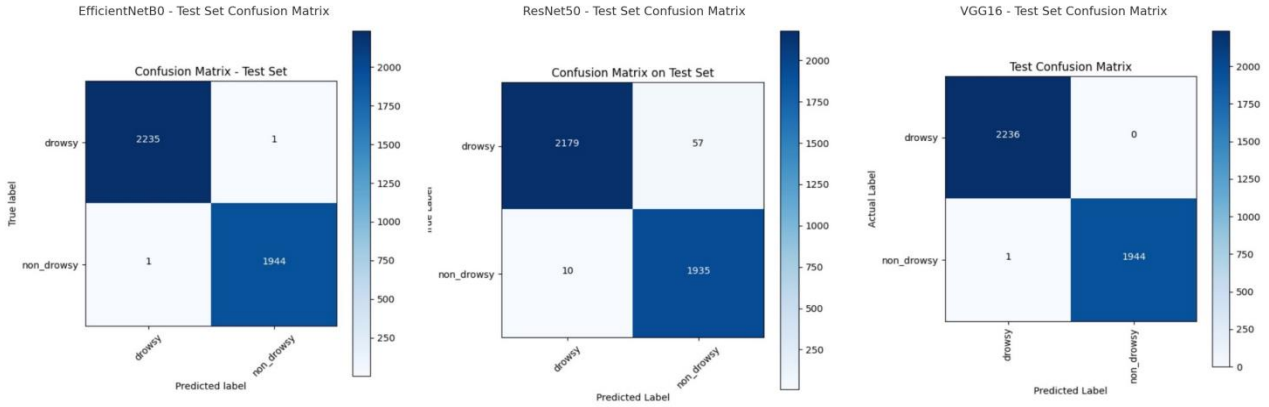


Fig. 3 Test Set Confusion Matrix

3.4.2 Accuracy and Classification Performance

The VGG16 model performed almost perfectly on the testset all its results being 0.999 for F1-score, recall and accuracy. Fig. 3 shows that there is only 1 misclassification in the VGG16 model. However, the large size of the VGG16 model (129 MB) and slower inference speed make it difficult for a resource-constrained environment to deploy.

The EfficientNetB0 model came to 0.999 in F1-score, recall and accuracy as well. As shown in Fig. 3, it has just one misclassification on the testset, indicating strong generalization ability. Furthermore, compared to VGG16, its 16.96 MB model size is substantially less, which makes it a more effective option for real-time applications, particularly in settings with restricted resources.

The ResNet50 model performed worse than the other two models, with an accuracy of 0.984 on the testset and 67 misclassifications. Because of its larger model size (148 MB) and slower inference speed (1.7996 seconds/sample), the ResNet50 model might not be as suitable for real-time systems as the EfficientNetB0 model.

3.4.3 Inference Speed and Resource Demand

EfficientNetB0 has an inference speed of 1.0663 seconds/sample, which is significantly faster than ResNet50 (1.7996 seconds/sample). Additionally, EfficientNetB0's model size is 16.96 MB, making it ideal for deployment in resource-limited systems.

VGG16 and ResNet50 have larger parameter counts and model sizes, at 21.1 million and 25.7 million parameters respectively, which may be unsuitable for resource-limited environments.

The EfficientNetB0 model has significantly fewer parameters, with only 4.3 million, and a model size of 16.96 MB, making it a more efficient choice.

4. Discussion

4.1. Interpretation of Results

This study conducted an experiment to test the performance of driver drowsiness detection using three deep learning models (VGG16, EfficientNetB0 and ResNet50) on the DDD dataset. Out of them, EfficientNetB0 showed the best balance between classification performance and resource efficiency. It outperformed the other models in terms of achieving 99.9% in F1-score, recall and accuracy while having a smaller model size and faster inference speed. On the other hand, VGG16 reached equally high accuracy but had a much bigger model size and worse inference speed, which made it less ideal for real-time or resource-constrained environments.

4.2. Limitations of the Study

One limitation of this study is that the models were only tested on the DDD dataset, which may limit the model's adaptability to other driving or fatigue datasets. The models may perform differently on the datasets that have different driving situations or face appearances that the DDD dataset does not include. Another limitation is that the models may become victims of overfitting. In different use cases or when using more diverse test cases, models like ResNet50 and VGG16 may not work as well as they have done on the validation set. Although the inference time of EfficientNetB0 is sufficient for many real-time applications, it may still not be fast enough for some time-sensitive applications.

4.3. Future Research Directions

Future research can mitigate the limitations mentioned before by testing the models more on some other diverse datasets, which would make them better applied to real-world driving situations. Additionally, there are some model optimization techniques that could further reduce the inference time of EfficientNetB0 and keep its high accuracy. Lastly, future research could improve the system's ability to detect driver fatigue by incorporating multimodal data, such as physiological signals like heart rate or EEG [9, 10].

5. Conclusion

This study assessed three deep learning models that refer to VGG16, EfficientNetB0 and ResNet50 to detect driver drowsiness using the DDD dataset. After comparing the three models through critical metrics like accuracy, F1-score, recall, inference speed and model size, it was found that EfficientNetB0 is best suited for this task. EfficientNetB0 achieved a high accuracy of 99.9% on both validation and test sets while maintaining a small model size of 16.96 MB and an inference speed of 1.0663 seconds per sample. So, it is very suitable for real-time applications in resource-constrained environments like in-vehicle monitoring systems.

However, there is a limitation lying in that the study is carried out in only one single dataset, the DDD dataset. Future work should test the performance of the proposed model on more datasets to ensure its robustness and applicability to different driving conditions and fatigue detection scenarios.

To sum up, EfficientNetB0 gives a great balance between performance and resource efficiency, which makes it well-suited for real-time fatigue detection systems. Possible future work includes improving the inference time of the model and applying it to a wider range of datasets and multimodal inputs in order to extend its real-world utility.

References

[1] Tefft, B.C. (2014) Prevalence of motor vehicle crashes involving drowsy drivers, United States, 2009–2013. AAA Foundation for Traffic Safety, Washington, DC.

[2] Chen, X., Su, L., Qin, M. 2020. Fatigue detection based on multi-feature fusion of fatigue behavior. In: Journal of Physics: Conference Series. Xiamen, China. 853(1), 012056.

[3] Danishvar, S. et al. (2023) Deep Learning for Detecting Multi-Level Driver Fatigue Using Physiological Signals: A Comprehensive Approach. Sensors, 23(19), 8171.

[4] Safarov, F. et al. (2023) Real-Time Deep Learning-Based Drowsiness Detection: Leveraging Computer-Vision and Eye-Blink Analyses for Enhanced Road Safety. Sensors, 23(14), 6459.

[5] Huang, R. et al. (2022) RF-DCM: Multigranularity deep convolutional model based on feature re-calibration and fusion for driver fatigue detection. IEEE Transactions on Intelligent Transportation Systems, 23: 630-640.

[6] Li,X. et al. (2023) Computer vision-based driver fatigue detection framework with personalization threshold and multi-feature fusion. Signal, Image and Video Processing, 18: 505-514.

[7] Ge,Q., Wang,J. 2022. Research on fatigue driving detection methods. In: International Conference on Smart Transportation and City Engineering. Chongqing, China. 75.

[8] Lu,P., Xue,R., Li,D., Ma,Z. 2023. Fatigue driving detection method based on deep learning. In: International Conference on Optics and Machine Vision. Changsha, China. 35.

[9] Wang, W. et al. 2023. Fatigue detection based on multiple visual features in virtual driving system. In: IECON 2023- 49th Annual Conference of the IEEE Industrial Electronics Society. Singapore, Singapore. pp.1-6.

[10] Liu, W., Tang, M., Wang, C., Zhang, K., Wang, Q., Xu, X. 2021. Attention-guided dual enhancement train driver fatigue detection based on MTCNN. In: International Academic Exchange Conference on Science and Technology Innovation (IAECST). Guangzhou, China. pp.1324-1329.