A Lightweight Attention-Based Network towards Distracted Driving Behavior Recognition

Yingcheng Lin 1, Dingxin Cao 1, Zanhao Fu 2, Yanmei Huang 1 and Yanyi Song 1,*

1 Chongqing Key Laboratory of Space Information Network and Intelligent Information Fusion, School of Microelectronics and Communication Engineering, Chongqing University, Chongqing 400030, China; lincy@cqu.edu.cn (Y.L.); cdx@cqu.edu.cn (D.C.); huangyanmei@cqu.edu.cn (Y.H.)
2 Chongqing University-University of Cincinnati Joint Co-op Institute, Chongqing University, Chongqing 400030, China; fuzo@mail.uc.edu
* Correspondence: songyanyi@cqu.edu.cn

Abstract: Distracted driving is currently a global issue causing fatal traffic crashes and injuries. Although deep learning has achieved significant success in various fields, it still faces the trade-off between computation cost and overall accuracy in the field of distracted driving behavior recognition. This paper addresses this problem and proposes a novel lightweight attention-based (LWANet) network for image classification tasks. To reduce the computation cost and trainable parameters, we replace standard convolution layers with depthwise separable convolutions and optimize the classic VGG16 architecture by 98.16% trainable parameters reduction. Inspired by the attention mechanism in cognitive science, a lightweight inverted residual attention module (IRAM) is proposed to simulate human attention, extract more specific features, and improve the overall accuracy. LWANet achieved an accuracy of 99.37% on Statefarm’s dataset and 98.45% on American University in Cairo’s dataset. With only 1.22 M trainable parameters and a model file size of 4.68 MB, the quantitative experimental results demonstrate that the proposed LWANet obtains state-of-the-art overall performance in deep learning-based distracted driving behavior recognition.

Keywords: distracted driving; behavior recognition; convolutional neural network; attention mechanism; lightweight deep learning network

1. Introduction

According to the Global Status Report on Road Safety [1] released by the World Health Organization (WHO) in 2018, 1.35 million people die from traffic accidents each year, and this number is still increasing. The Traffic Safety Facts [2] released by National Highway Traffic Safety Administration (NHTSA) in 2017 stated that there are at least 2994 fatal crashes caused by distracted driving. It is reported that with a proper driver monitoring system, the risk of distracted driving will be sharply reduced [3].

Distracted driving is defined as “a diversion of attention from driving because the driver is temporarily focusing on an object, person, task or event not related to driving” [4]. Physiological signals, vehicle data, and deep learning are the three main methods to classify distracted driving. Sahayad et al. [5] proposed a system that could detect inattention using an electrocardiogram (ECG) and surface electromyogram (sEMG) signals. K-nearest neighbor analysis (KNN), linear discriminant analysis (LDA), and quadratic discriminant analysis (QDA) were involved for classification. Wang et al. [6] studied the different EEG patterns in the driver’s lane-keeping behaviors. The accuracy was evaluated based on the support vector machine (SVM) and radial basis function (RBF) kernel function. Omerustaoglu et al. [7] fused the vehicle sensor data with vision data, and significantly improve the accuracy of detection tasks for distracted drivers. Li et al. [8] utilized the built-in accelerometer and gyro head inertial sensor to collect the driver’s head pose information. Although most proposed methods have achieved desirable performances, the
methods based on physiological signals exist physical contact with the driver. The collected information is part of personal privacy. Hence, these methods cannot be widely spread. The methods based on vehicle data require installing high-precision sensors, which increases the cost. It should be noted that methods involving physiological signals and vehicle data are not totally separated with deep learning. Instead, they work as complementary plans in these proposed methods.

With the rapid development of deep learning, due to the low cost and high accuracy, deep learning-based approaches of distracted driving behavior recognition will be mainstream [9]. Genetic algorithm [10], ensemble learning [11], skin segmentation [12], hybrid CNN [13], Long Short-Term Memory [14], Multiple-Granularity Middle Network [15], multi-stream CNN [16], faster R-CNN [17], and principal component analysis [18] have recognized some success in distracted driving detection. However, due to the limited resources on edge devices, a desirable real-time method should be with low computation cost while still maintaining high accuracy. It should also be with great adaptability to the environment, so it can remain excellent performance under various unknown circumstances.

Aiming at these requirements, this paper proposes a Lightweight Attention-based network (LWANet) and is tested in the field of distracted driving. The main contributions of this paper are summarized as follows:

1. We propose a lightweight Inverted Residual Attention Module (IRAM) towards the problem that the current lightweight network has relatively low accuracy. IRAM effectively improves the classification accuracy with almost no increase of trainable parameters and computation cost.
2. We embed the depthwise separable convolution into classic VGG16 and optimize the network structure. Regarding the problem that classic CNN cannot be implemented on edge devices due to the high model complexity, LWANet has very few trainable parameters and model size.
3. We utilize subtract mean filter in image preprocessing to further improve the model’s environmental adaptivity.
4. Compared with existing state-of-the-art deep learning-based methods, the proposed LWANet has fewer parameters and can be applied to various embedded real-time detection scenarios.

The rest of this paper is organized as follows. Section 2 introduces the related work. Section 3 discusses the involved methods. Section 4 presents the experimental results. Section 5 concludes the whole paper.

2. Related Work

In the field of deep learning, especially in image classification and object detection tasks, a significant amount of research focuses on reducing network complexity and computation cost rather than simply increasing the accuracy. An optimized network should train a model with the smallest size and the highest accuracy. Only such networks can be applied to industrial production and practices. These requirements challenge the overall performance of a network.

There are mainly two methods towards designing a lightweight convolutional neural network. One is compressing the existing network with excellent performance [19]. It refers to removing the redundant convolution layers and fully connected layers and cropping the convolution kernels. Another one is designing a new network module. Currently, most research focuses on the convolution operation. Howard et al. [20] utilized depthwise separable convolutions to establish lightweight deep neural networks. This method converted standard convolution to depthwise convolution and $1 \times 1$ pointwise convolution. Yang et al. [21] used asymmetric convolution blocks to replace the standard convolutional layers for object detection. The asymmetric convolution module had three parallel branches. The first branch included asymmetric convolution with kernel sizes of $1 \times 3$ and $3 \times 1$. The second branch included asymmetric convolution with a kernel size of $1 \times 3$. The third branch included asymmetric convolution with a kernel size of $3 \times 1$. The octave
convolution proposed by Chen et al. [22] was a plug-and-play convolutional unit that could greatly reduce memory and computation cost. The octave convolution divided the convolution feature maps into low-frequency and high-frequency groups and halved the size of the low-frequency feature maps to speed up the convolution operation.

In recent years, the attention mechanism has attracted researchers’ interests. Its essence is to simulate how a human observes an object. In cognitive science, due to the bottleneck of information processing, human beings will selectively pay attention to a part of the information and ignore the other visible information [23]. In the field of deep learning, attention mechanism is first applied to machine translation of natural language processing [24]. It spreads to object detection and image classification [25,26] later. The attention mechanism is usually divided into hard attention and soft attention. Since soft attention is differentiable, a neural network can compute the gradient and learn the weighted attention by forward propagation and backward feedback [27]. Hence, soft attention is widely utilized in deep learning. It can be further divided into channel domain, spatial domain, and mixed domain. Channel domain is related to the feature channels of an image. Spatial domain is related to the location of features in an image. The current mainstream of attention modules is SE [28], CBAM [29], EAC [30], and Triplet [31].

Towards image classification problem, with the aid of attention mechanism, the network performance improves significantly. He et al. [32] proposed a bilinear squeeze-and-excitation network (BiSENet) for tree species classification and achieved better performance than existing methods. Xie et al. [33] used the CBAM module for water scene recognition. The method had fewer parameters and could improve the accuracy of water scene recognition. Chen et al. [34] used an improved CBAM module for fly species recognition with an accuracy rate of 90%, which was better than the state-of-the-art methods. Wang et al. [35] used the triple attention method for 14 thoracic disease classification. According to the spatial location of pathological abnormalities, it could determine which feature channels provide discriminative information and which scale played a major role in diagnosis. Pande et al. [36] used a hybrid attention approach for hyperspectral image classification. It utilized 1D and 2D CNNs to enhance the spectral and spatial characteristics of the input image. In the application of classifying white blood cells from microscopy hyperspectral images, Wang et al. [37] used a 3D attention module to emphasize more important functions with an accuracy of 97.72%.

Some researchers have tried to embed the attention mechanism into driver action recognition. Hu et al. [38] proposed a multi-scale attention convolutional neural network (MSA-CNN). It included a multi-scale convolutional module, attention module, and classification module. Wang et al. [39] proposed an attention module including channel level attention and space level attention for driver behavior recognition. Experimental results showed that the introduction of an attention mechanism can effectively improve the accuracy. Jegham et al. [40] proposed a soft spatial attention network based on deep learning for driver action recognition. The network focused the attention on the driver’s silhouette and motion, and the accuracy reached 75%. Although the involvement of attention mechanism proved to be effective, these approaches utilized large-scale networks such as ResNet50. They did not consider the computation cost and the deployment on edge devices.

Motivated by the urgent needs of distracted driving detection, the development of lightweight network design, and attention mechanism, this paper introduces a novel network architecture LWANet (Lightweight Attention-based Network) to solve the trade-off between computation cost and overall accuracy.

3. Methods

In this section, we present the mainly utilized methods in detail. The subtract mean filter, depthwise separable convolution, and the proposed Inverted Residual Attention Module (IRAM) are introduced. Then, the whole network structure is further developed.
3.1. Subtract Mean Filter

Subtract mean filter is an effective method to reduce the noises in an image [41]. It is calculated as the gray value of each pixel in a certain area minus the average gray value in a certain area. The area is usually referred to a disk. Consider a disk area with radius \( r \), there are \( n \) pixels in the area and the gray value of the \( i \)th pixel is \( g_i \). After the subtract mean filter, the gray value of the \( i \)th pixel is defined in (1):

\[
g'_i = |g_i - \frac{\sum_{k=1}^{n} g_k}{H}|
\]  

(1)

According to (1), the grayscale image output from the subtract mean filter represents a relative pixel-wise grayscale level. The number of pixels \( n \) in a certain area is directly related to the disk area radius \( r \). A disk area with a larger radius will provide an output image with a clearer outline. Effective extraction of the driver’s outline can lead to a more accurate classification result.

3.2. Depthwise Separable Convolution

Depthwise separable convolution [20] can greatly reduce the computation cost and has been widely used in lightweight CNN design. Its essence is the decomposition of standard convolution within a feature channel domain.

Consider a feature map with size \( H \times W \times D_{\text{in}} \), where \( D_{\text{in}} \) is the number of input channels. The stride is set to be 1. When the feature map is convoluted by a standard convolution kernel with size \( k \times k \times D_{\text{in}} \times D_{\text{out}} \), where \( D_{\text{out}} \) is the number of output channels, the computation cost can be calculated as (2). After the convolution operation, the size of the output feature map is \((H - k + 1) \times (H - k + 1) \times D_{\text{out}}\). The schematic of standard convolution is shown in Figure 1.

\[
C_{\text{standard}} = H \cdot W \cdot k^2 \cdot D_{\text{in}} \cdot D_{\text{out}}
\]  

(2)

Depthwise separable convolution involves depthwise convolution and pointwise convolution. Depthwise convolution extracts feature on a single feature map. Pointwise convolution fuses these extracted features from different feature maps and outputs a final feature map. The feature map after the depthwise convolution is also called the intermediate feature map. The schematic of depthwise separable convolution is shown in Figure 2.

Figure 1. Schematic of standard convolution.
Currently, attention mechanism is a hot research topic. The presence of some effective and lightweight attention modules can improve the network performance with negligible computation cost. The channel domain in the soft attention mechanism is about “what to look for.” The spatial domain is about “where to look.” As for the real-time image classification of distracted driving, the camera is always fixed. This means that for most images, the spatial information is almost unchanged. An effective attention module in this scenario should focus on the channel domain features. The schematic of the proposed IRAM is shown in Figure 3.

Figure 2. Schematic of depthwise separable convolution.

According to Figures 1 and 2, the standard convolution and the depthwise separable convolution can output the same feature map. However, their computation cost is different. For the depthwise separable convolution, it can be calculated as (3):

\[
C_{\text{depthwise, separable}} = C_{\text{depthwise}} + C_{\text{pointwise}} = H \cdot W \cdot k^2 \cdot D_{\text{in}} + H \cdot W \cdot D_{\text{in}} \cdot D_{\text{out}}. \tag{3}
\]

We can compare the computation cost of both convolution methods:

\[
\frac{C_{\text{depthwise, separable}}}{C_{\text{standard}}} = \frac{H \cdot W \cdot k^2 \cdot D_{\text{in}} + H \cdot W \cdot D_{\text{in}} \cdot D_{\text{out}}}{H \cdot W \cdot k^2 \cdot D_{\text{in}} \cdot D_{\text{out}}} = \frac{1}{D_{\text{out}}} + \frac{1}{k^2} \tag{4}
\]

Normally, the number of output channels \(D_{\text{out}}\) is much larger than the convolution kernel size \(k\). It indicates that the computation cost of depthwise separable convolution is approximately \(\frac{1}{D_{\text{out}}}\) of the standard convolution. The model size will be correspondingly reduced. However, it has been reported [20] that the usage of too much depthwise separable convolution will decrease the accuracy. Hence, we need to balance the model complexity and the model accuracy.

3.3. Proposed Inverted Residual Attention Module (IRAM)

Currently, attention mechanism is a hot research topic. The presence of some effective and lightweight attention modules can improve the network performance with negligible computation cost. The channel domain in the soft attention mechanism is about “what to look for.” The spatial domain is about “where to look.” As for the real-time image classification of distracted driving, the camera is always fixed. This means that for most images, the spatial information is almost unchanged. An effective attention module in this scenario should focus on the channel domain features. The schematic of the proposed IRAM is shown in Figure 3.

Figure 3. Schematic of Inverted Residual Attention Module (IRAM).
Consider a feature map \( F_s \in \mathbb{R}^{H \times W \times C} \) is input into the IRAM. The channel attention module will infer a 1D channel-wise attention map \( M_c \in \mathbb{R}^{1 \times 1 \times C} \). The spatial attention module will infer a 2D spatial-wise attention map \( M_s \in \mathbb{R}^{H \times W \times 1} \). The whole attention mechanism process can be summarized as follows:

\[
F'_s = M_c(F_s) \otimes F_s, \quad (5)
\]

\[
F''_s = M_s(F'_s) \otimes F'_s, \quad (6)
\]

where \( \otimes \) denotes the element-wise dot product. The structure of channel-wise attention is inspired by the inverted residual module.

The process of a normal residual module can be concluded as “compression-convolution-expansion”. Comparatively, the process of an inverted residual module \([42]\) can be concluded as “expansion-convolution-compression”. Pointwise convolution is first used to expand feature channels and is finally used to compress them to the initial amount. The purpose is to extract more higher-level features. Hence, the output weighted channel-wise feature map can be more specific and precise.

For the channel attention module, the input feature map will first experience a global average pooling. The pooling value of each channel can be regarded with a global region of interest. Its calculation process is shown as:

\[
P_c = \text{GAP}(U_c) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} U_c(i,j),
\]

where \( P_c \) is the pooling value of the \( c \)th channel and \( U_c(i,j) \) is the value at position \((i,j)\) of the \( c \)th channel in the input feature map.

Pointwise convolution is performed after GAP to increase the channels. It can be represented as:

\[
P'_c = \sum_{i=1}^{C} W_c \times P_i,
\]

where \( P'_c \) is the pooling value of the \( c \)th channel, \( c \in [0, C \cdot r] \), \( r \) is the expand ratio, \( W_c \) is the weight of the \( c \)th filter, and \( P_i \) is the pooling value of the \( i \)th channel.

Depthwise convolution is performed after the first pointwise convolution to extract the features of each channel. Since depthwise convolution cannot change the feature channels, another pointwise convolution is performed to reduce the channels.

The spatial attention module is connected after the channel attention module. It utilizes the channel-wise weighted feature map as its input. It conducts the convolution with a standard convolution layer. We select a \( 3 \times 3 \) convolution kernel here to reduce the computation cost. It is tested that a \( 7 \times 7 \) convolution kernel produces a slightly higher accuracy. However, considering the increased model complexity, we still use a \( 3 \times 3 \) kernel. The process can be represented as:

\[
M_s(F'_s) = \delta \left( f^{3 \times 3}([\text{avgpool}(F'_s); \text{maxpool}(F'_s)]) \right),
\]

where \( \delta \) denotes a sigmoid function.

The attention module can be configured in three different configurations regarding the series or parallel connection. We will compare their performances in the experiment part. It should be noted that IRAM is designed as a lightweight plug-and-play attention module and can be easily combined with most classic CNN networks.

### 3.4. Proposed LWANet Network Structure

The classic VGG16 network \([43]\) includes 13 convolution layers and 3 fully connected layers. The entire network utilizes convolution kernels with size \( 3 \times 3 \) and max-pooling
with stride $2 \times 2$. Due to a large number of trainable parameters, VGG16 consumes a lot of computation resources. It is not feasible for its deployment on edge devices.

Based on the depthwise separable convolution and inverted residual attention module discussed above, we propose the Lightweight Attention-based network architecture. The depthwise separable convolution can greatly reduce the computational cost, making it possible to apply the model to the edge devices. Inverted residual attention module can focus the weights of the proposed network on meaningful pixels and channels, promote the effective features, and suppress the interference of noise. Therefore, the inverted residual attention module can effectively improve the accuracy. The network is designed to reduce the trainable parameters but remains a desirable model accuracy. The network schematic is shown in Figure 4.

**Figure 4.** Schematic of the Lightweight Attention-based network (LWANet).

LWANet takes an image of size $120 \times 120 \times 3$ as the input and produces a classification label as the output. The detailed structure of filter shapes and input and output shapes of each layer is shown in Table 1.

**Table 1.** A list of detailed network architecture.

| Layer Name | Input Shape | Output Shape | Filter Shape |
|------------|-------------|--------------|--------------|
| Conv_1     | $120 \times 120 \times 3$ | $120 \times 120 \times 32$ | $3 \times 3 \times 32$ |
|            |             |              | $1 \times 1 \times 32 \times 3$ |
|            |             |              | $3 \times 3 \times 96$ |
| IRAM_1     | $120 \times 120 \times 32$ | $120 \times 120 \times 32$ | $1 \times 1 \times 96 \times 32$ |
|            |             |              | $3 \times 3 \times 32$ |
|            |             |              | $3 \times 3 \times 32$ |
| DSWC_1     | $120 \times 120 \times 32$ | $120 \times 120 \times 32$ | $1 \times 1 \times 32 \times 32$ |
| Maxpool_1  | $120 \times 120 \times 32$ | $60 \times 60 \times 32$ | $3 \times 3 \times 32$ |
| Conv_2     | $60 \times 60 \times 32$ | $60 \times 60 \times 32$ | $3 \times 3 \times 32$ |
| Maxpool_2  | $30 \times 30 \times 32$ | $30 \times 30 \times 32$ | $3 \times 3 \times 64$ |
| Conv_3     | $30 \times 30 \times 32$ | $30 \times 30 \times 64$ | $1 \times 1 \times 64 \times 2$ |
|            |             |              | $3 \times 3 \times 3 \times 128$ |
| IRAM_2     | $30 \times 30 \times 64$ | $30 \times 30 \times 64$ | $1 \times 1 \times 128 \times 64$ |
|            |             |              | $3 \times 3 \times 3 \times 64$ |
|            |             |              | $3 \times 3 \times 3 \times 64$ |
| DSWC_2     | $30 \times 30 \times 64$ | $30 \times 30 \times 64$ | $1 \times 1 \times 64 \times 64$ |
| Maxpool_3  | $30 \times 30 \times 64$ | $15 \times 15 \times 64$ | $3 \times 3 \times 64$ |
| Conv_4     | $15 \times 15 \times 64$ | $15 \times 15 \times 64$ | $3 \times 3 \times 64$ |
| Maxpool_4  | $15 \times 15 \times 64$ | $8 \times 8 \times 64$ | $3 \times 3 \times 64$ |
| Conv_5     | $8 \times 8 \times 64$ | $8 \times 8 \times 128$ | $3 \times 3 \times 64$ |
| Maxpool_5  | $8 \times 8 \times 128$ | $4 \times 4 \times 128$ | $3 \times 3 \times 64$ |
| Fc_1       | $1 \times 1 \times 2048$ | $512$ | $1 \times 1 \times 2048$ |
| Fc_2       | $512$ | $10$ | $512$ |
Balancing the decreasing accuracy and computation lost, we utilize two depthwise separable convolution layers to replace the standard convolution layers. IRAM is connected after the first and third standard convolution layer. ReLU activation function and max-pooling are added after all the five standard convolution layers. Max pooling is involved to reduce the feature map sizes. Two fully connected layers reduce the last convolutional feature maps to 512-D and 10-D feature vectors.

4. Experimental Results

This section presents the experimental results using the proposed methods. Section 4.1 introduces the involved dataset. Section 4.2 describes implementation details. Section 4.3 describes the image processing results. Section 4.4 demonstrates the performance of the proposed network.

4.1. Dataset

We use two publicly available datasets to train and evaluate the performance of the network. The first dataset is the State Farm Distracted Driver Detection (SF3D) dataset released on Kaggle in 2016 [44]. It is taken by a constant-placed 2D dashboard camera of 640 × 480 pixels in RGB. The dataset contains 22,424 labeled pictures with ten classes: safe driving, texting-right, talking on the phone-right, texting-left, talking on the phone-left, operating the radio, drinking, reaching behind, hair and makeup, talking to passenger. We utilize 70% of pictures for training and 30% for testing. Figure 5 shows the pictures of ten prediction classes in the SF3D dataset.

![Figure 5. Some sample pictures from the SF3D dataset. (a) Safe driving (b), texting-right (c), talking on the phone-right (d), texting-left (e), talking on the phone-left (f), operating the radio (g), drinking (h), reaching behind (i), hair and makeup (j), talking to passenger.](image)

The second dataset was established by Abouelnaga et al. [10,45] and named the American University in Cairo Distracted Driver (AUC2D) dataset. It contains 44 participants from 7 different countries. The ten distracted classes are the same as the SF3D dataset. Some pictures are taken at different times of day, in different driving conditions, and wearing different clothes. The dataset includes 10,555 training images and 1123 testing images with 1920 × 1080 pixels in RGB. Figure 6 shows the pictures of ten prediction classes in the AUC2D dataset.

4.2. Implementation Details

To demonstrate the effectiveness of the LWANet proposed in this paper, we conducted a series of experiments on the SF3D and AUC2D datasets for verification. All the experiments were conducted on a computer with Intel(R) Xeon(R) Silver 4210 CPU @ 2.20 GHz, a GeForce RTX 2080, and 128 GB of RAM. The algorithms and network were developed in Python 3.6 with OpenCV 3.3.1 and TensorFlow 1.13.1. During the training process, we set the learning rate as 0.001, batch size of 50. The overall flowchart is shown in Figure 7.
Figure 6. Some sample pictures from the AUC2D dataset. (a) Safe driving (b), texting-right (c), talking on the phone-right (d), texting-left (e), talking on the phone-left (f), operating the radio (g), drinking (h), reaching behind (i), hair and makeup (j), talking to passenger.

Figure 7. Overall flowchart.

4.3. Image Preprocessing and Baseline Selection

The original size of images in the SF3D dataset is $640 \times 480 \times 3$, and $1920 \times 1080 \times 3$ in the AUC2D dataset. All the raw images in both datasets are resized to $120 \times 120 \times 3$ to speed up the CNN training process and decrease the computation cost.

It can be noticed that the images in the AUC2D dataset are taken at different times of the day. To avoid the impact of various illumination conditions, we utilize a subtract mean filter to extract the features. The disk radius is set as 10 for the SF3D dataset and 15 for the AUC2D dataset. The performance of the subtract mean filter is shown in Figure 8.

Figure 8. The performance of the subtract mean filter. (a) The original image. (b) The image after being filtered.

By applying the subtract mean filter, some important features of an image can be extracted. Other features, especially the unnatural luminance distribution, will be eliminated. It is expected that the images, after being preprocessed, will be easier to fit the network.
In our experiment, we select VGG16 as the baseline. VGG16 has recognized lots of success in the image classification field [46]. Compared to larger-scale networks, especially Resnet, VGG16 is with less time complexity and fewer trainable parameters. Compared to other lightweight networks, for example, MobileNet, ShuffleNet, and AlexNet, VGG16 is more flexible to embed some useful modules, such as the attention module. We compare the performance of networks with and without the subtract mean filter process. The results are summarized in Table 2.

| Item                              | SF3D Dataset          | AUC2D Dataset        |
|----------------------------------|-----------------------|----------------------|
|                                  | Accuracy   | Loss       | Accuracy   | Loss       |
| LWANet without subtract mean filter | 97.77 ± 1.53% | 0.120 ± 0.067 | 95.76 ± 2.44% | 0.216 ± 0.151 |
| LWANet with subtract mean filter  | 99.37 ± 0.22% | 0.026 ± 0.007 | 98.45 ± 0.28% | 0.089 ± 0.024 |

Besides the accuracy improving, the subtract mean filter can guarantee a stable result with less uncertainty. The confidence level of each training is, therefore, increased. Since the filter can eliminate some unexpected noise, it is proved that our model has greater environmental adaptivity.

4.4. Performance Evaluation of LWANet

4.4.1. Overall Model Performance

We test our model performance on both SF3D and AUC2D datasets. During the training process, we set the learning rate as 0.001, with a batch size of 50.

We present a detailed analysis on both datasets by training the model with the 70% training data and testing the model with the rest 30% testing data. The training and testing performance of both datasets are shown in Figure 9.

![Figure 9. Training and testing performance. (a) On the SF3D dataset. (b) On the AUC2D dataset.](image)

The training accuracy almost remains 100% at the end of the training process. However, the testing accuracy seems difficult to further improve and with the highest testing accuracy of 99.37 ± 0.22% on the SF3D dataset and 98.45 ± 0.28% on the AUC2D dataset. To find out which classes cause the wrong predictions, we perform a confusion matrix evaluation in Figure 10. We also list the number of correct and wrong classifications in Table 3.
Figure 9. Training and testing performance. (a) On the SF3D dataset. (b) On the AUC2D dataset.

The training accuracy almost remains 100% at the end of the training process. However, the testing accuracy seems difficult to further improve and with the highest testing accuracy of 99.37 ± 0.22% on the SF3D dataset and 98.45 ± 0.28% on the AUC2D dataset.

To find out which classes cause the wrong predictions, we perform a confusion matrix evaluation in Figure 10. We also list the number of correct and wrong classifications in Table 3.

Table 3. Classification accuracy of each class.

| Classes            | SF3D Dataset | AUC2D Dataset |
|--------------------|--------------|---------------|
|                    | Total | Correct | Accuracy | Total | Correct | Accuracy |
| Safe driving       | 771   | 763     | 98.96%    | 554   | 544     | 98.19%    |
| Texting-right      | 655   | 654     | 99.85%    | 364   | 364     | 100.00%   |
| Phone-right        | 702   | 699     | 99.57%    | 242   | 242     | 100.00%   |
| Texting-left       | 707   | 705     | 99.72%    | 209   | 207     | 99.04%    |
| Phone-left         | 697   | 695     | 99.71%    | 265   | 263     | 99.25%    |
| Radio              | 684   | 680     | 99.42%    | 222   | 222     | 100.00%   |
| Drinking           | 696   | 695     | 99.86%    | 218   | 206     | 94.50%    |
| Reaching behind    | 586   | 581     | 99.15%    | 211   | 208     | 98.58%    |
| Hair and makeup    | 575   | 566     | 98.43%    | 225   | 220     | 97.78%    |
| Talking to passenger | 627  | 620     | 98.88%    | 390   | 379     | 97.18%    |

The classification results from LWANet are generally desirable. However, in the AUC2D dataset, the class of drinking seems to be usually misclassified. It is predicted as texting-right or hair and makeup in most cases. This is because the training images labeled as drinking involve many different hand and head features. LWANet mainly focuses on these features when dealing with most tasks. The network may be confused with these features because some of them are similar to other classes. For example, in some images, drinking with the right hand is similar to texting using the right hand. Besides this class, LWANet performs excellently on the AUC2D dataset and achieves an overall accuracy of 98.45%.

4.4.2. Model Real-Time Performance

After the training process, the network model can be saved as a .pb file. It includes all the required parameters and can restore the network. It is easy to transfer the .pb file to some edge devices, for example, an Android phone. The file size will affect the operating speed. To further evaluate the model performance in real-time, we develop a related Android App and transfer the TensorFlow model. The schematic of the Android App development is shown in Figure 11. We conduct the experiment on an Android phone, “Xiaomi 11 Pro”, with a CPU of Qualcomm Snapdragon 888 and 12 GB of RAM. Table 4 summarizes the GPU and Android phone processing speed in units of frames per second. The results are summarized in Table 4.
Talking to passenger 627 620 98.88% 390 379 97.18%
The classification results from LWANet are generally desirable. However, in the AUC2D dataset, the class of drinking seems to be usually misclassified. It is predicted as texting or hair and makeup in most cases. This is because the training images labeled for drinking involve many different hand and head features. LWANet mainly focuses on these features when dealing with most tasks. The network may be confused with these features because some text in the training images may involve many different hand and head features.

4.4.3. Ablation Study

We perform ablation studies to evaluate the effectiveness of the lightweight structure and the IRAM. The structures are compared in terms of FLOPs (floating-point operations), trainable parameters, size of .pb model file, classification accuracy on both datasets, GPU processing speed, and Android phone processing speed.

For most convolutional neural networks, convolutional layers and fully connected layers cover a large proportion of the network FLOPs. The FLOPs of a convolutional layer are defined in (10).

\[
\text{FLOPS}_{\text{CONV}} = O \left( \sum_{i=1}^{D} M_i^2 K_i^2 C_{i-1} C_{i} \right),
\]

where \( D \) is the depth of the network, \( i \) is the \( i \)th convolutional layer, \( M \) is the length of the feature map for each convolutional kernel, \( K \) is the size of the convolutional kernel, and \( C_{i} \) is the number of channels of the \( i \)th convolutional layer.

For a fully connected layer, if the dimension of the input data is \((N, D)\), the weight dimension of a hidden layer is \((D, M)\) and the dimension of output data is \((N, M)\). The FLOPs of a fully connected layer are defined in (11):

\[
\text{FLOPS}_{\text{FC}} = (2D - 1)M
\]

Trainable parameters refer to the total amount of parameters in a network. In general cases, trainable parameters are mostly related to the weights and biases.

It is expected that the proposed network with an optimized structure and involvement of depthwise separable convolution will result in fewer FLOPs, trainable parameters, model file size, and faster processing speed. At the same time, the testing accuracy should remain the same or even higher. The results are shown in Table 5.

By comparing the standard VGG16 and the lightweight VGG without IRAM, the involvement of depthwise separable convolution and network compression results in 98.32% FLOPs reduction and 98.16% trainable parameters decrease. The model file size is reduced from 248 MB to 4.58 MB. The processing speed on Android phone increases by 1538.81%. The improvement of model complexity is accompanied by the sacrifice of model accuracy. The model accuracy decreases a bit on both datasets. This is not surprising...
because, in standard VGG16, there are 13 convolution layers. For the lightweight VGG without IRAM, there are only seven convolution layers and two of them are depthwise separable convolution layers. Fewer convolution layers will extract fewer high-level features. Nevertheless, according to this ablation study, we can conclude that the proposed lightweight network can maintain relatively high accuracy with very low computation cost.

Table 5. Ablation study evaluation.

| Items                   | Standard VGG16 | Lightweight VGG without IRAM | LWANet |
|-------------------------|----------------|-------------------------------|--------|
| FLOPs                   | 455,867,390    | 7666,141                      | 7715,347 |
| Trainable Parameters    | 65,120,350     | 1199,626                      | 1224,208 |
| Model File Size         | 248 MB         | 4.58 MB                       | 4.68 MB |
| Accuracy on SF3D        | 99.39 ± 0.13%  | 98.87 ± 0.26%                 | 99.37 ± 0.22% |
| Accuracy on AUC2D       | 98.58 ± 0.24%  | 97.32 ± 0.58%                 | 98.45 ± 0.28% |
| GPU Processing Speed    | 851 ± 75 FPS   | 1594 ± 92 FPS                 | 1485 ± 83 FPS |
| Android Phone Processing Speed | 0.67 ± 0.41 FPS | 10.98 ± 1.27 FPS | 10.77 ± 1.35 FPS |

The proposed IRAM is designed with two modules in series and channel attention first. There are another two configurations, which are indicated in Figure 12. We compare their accuracy on both datasets and summarize the results in Table 6. It demonstrates that the configuration of IRAM is the most desirable.

Figure 12. Three configurations of the attention module: series with channel first, series with spatial first, and parallel.

Table 6. Attention module configure evaluation.

| Items                   | Parallel     | Series-Spatial First | Series-Channel First |
|-------------------------|--------------|----------------------|----------------------|
| Accuracy on SF3D        | 99.31 ± 0.05%| 99.04 ± 0.32%        | 99.37 ± 0.22%        |
| Accuracy on AUC2D       | 98.11 ± 0.34%| 97.70 ± 0.52%        | 98.45 ± 0.28%        |

The ablation study of the IRAM is presented by comparing the lightweight VGG without IRAM with LWANet. The IRAM module embedded in the network causes 0.64% FLOPs increase, 2.05% trainable parameters increase, and 1.91% Android phone processing speed decrease. The model file size has increased by 0.1 MB. The model accuracy improved by 0.5% on the SF3D dataset and 1.13% on the AUC2D dataset. Suppose the tested Android phone is used in a vehicle and is capturing videos with 10 FPS. The 0.5%
to 1.13% improvement indicates that for about every 30 s, the model can correctly predict 1–3 images more. It should be noted that fatal crashes often happen several seconds right after distracted driving behaviors. The higher prediction accuracy may successfully prevent a traffic accident. Considering that the model complexity is with almost no increase, we can conclude that the proposed IRAM works as a lightweight module and can effectively improve the network performance. A conclusion can be further drawn that a lightweight network structure combining with the attention module can almost reach the performance of a classic large-scale network.

4.4.4. Model Comparison and Discussion

In this study, we mainly focus on three problems. The first problem is that most trained models from classic networks cannot be deployed on resource-limited edge devices due to the high model complexity and computation cost. The second problem is that most lightweight networks are with relatively low accuracy. The third problem is that for most methods, a jitter of light or unnatural illumination condition may cause a wrong prediction. Towards these problems, we replace standard convolution with the depthwise separable convolution, optimize the network structure, introduce an IRAM attention mechanism, and utilize subtract mean filter as part of image preprocessing.

In most papers, the proposed methods cannot balance the above three problems. In the past several decades, researchers have developed many techniques for deep learning-based distracted driving behavior recognition. They have improved a bit of accuracy, but the overall performance is still not desirable. Researchers have realized that computation cost is a bottleneck for model edge-oriented migration. They have reduced the model complexity significantly. However, for most vehicles, the onboard devices have much weaker microcontrollers than mobile phones. A trained model with a file size of more than 5 MB will result in a very low FPS on a mobile phone. Most proposed methods may not be able to run in real-time on a real vehicle device. To comparatively present the performance of LWANet, we compare our network with other state-of-the-art approaches and summarize the results in Table 7.

Table 7. A listed comparison of LWANet with other approaches.

| Author                 | Year | Network Baseline | Attention Module | Involved Dataset | Accuracy  | Trainable Parameters |
|------------------------|------|------------------|------------------|------------------|-----------|----------------------|
| Zhang et al. [47]      | 2016 | VGG16            | No               | SF3D             | 90.2%     | 140 M                |
|                        |      | VGG-GAP          |                  |                  | 91.3%     |                      |
|                        |      | Ensemble VGG16   |                  |                  | 92.6%     | >140 M               |
|                        |      | and VGG-GAP      |                  |                  |           |                      |
| Okon et al. [48]       | 2017 | AlexNet + Softmax| No               | SF3D             | 96.8%     | 63.2 M               |
|                        |      | AlexNet + Triplet Loss |          |                  | 98.7%     |                      |
| Hssayeni et al. [49]   | 2017 | ResNet           | No               | SF3D             | 85%       | 60 M                 |
| Abouelnaga et al. [10] | 2018 | AlexNet          | No               | AUC2D            | 93.65%    | 62 M                 |
|                        |      | Inception V3     |                  |                  | 95.17%    | 24 M                 |
|                        |      | Majority Voting  |                  |                  | 95.77%    | 120 M                |
|                        |      | Ensemble         |                  |                  |           |                      |
|                        |      | GA-Weighted Ensemble |          |                  | 95.98%    | 120 M                |
| Behera et al. [50]     | 2018 | DenseNet         | No               | AUC2D            | 94.2%     | 8.06 M               |
| Baheti et al. [51]     | 2018 | VGG16            | No               | AUC2D            | 94.44%    | 140 M                |
|                        |      | VGG16 + Regularization |      |                  | 96.31%    | 140 M                |
|                        |      | Modified VGG16   |                  |                  | 95.54%    | 15 M                 |
Table 7. Cont.

| Author          | Year | Network Baseline | Attention Module | Involved Dataset | Accuracy  | Trainable Parameters |
|-----------------|------|------------------|------------------|------------------|-----------|----------------------|
| Hu et al. [16]  | 2018 | VGG16            | No               | SF3D             | 86.6%     | 33.56 M              |
|                 |      |                  |                  | AUC2D            | 93.2%     |                      |
| Ai et al. [52]  | 2019 | VGG16-ones attention | Yes             | AUC2D            | 84.82%    | >140 M               |
|                 |      | VGG16-two-way attention |               |                 | 87.74%    | >140 M               |
| Janet et al. [53]| 2019 | Vanilla CNN      | No               | SF3D             | 97.05%    | 26.05 M              |
| Wang et al. [39]| 2019 | VGG16            | Yes              | SF3D             | 88.67%    | >65.12 M             |
|                 |      | Resnet50         |                  |                  | 92.45%    | >46.16 M             |
| Xing et al. [12]| 2019 | AlexNet          | No               | Self-Collected   | 87.5%     | 6.8 M                |
|                 |      | GoogLeNet        |                  |                  | 83.0%     | 46.16 M              |
| Huang et al. [13]| 2020| ResNet 50 + Inception V3 + Xception | No | SF3D             | 96.74%    | >72.3 M              |
| Dhakate et al. [11]| 2020| Faster R-CNN     | Yes              | SF3D             | 86.0%     | 6.53 M               |
|                 |      |                  |                  | SEU              | 92.1%     |                      |
| Baheti et al. [54]| 2020| VGG16            | No               | SF3D             | 99.75%    | 2.2 M                |
|                 |      |                  |                  | SEU              | 95.24%    |                      |
| Hu et al. [38]  | 2020 | Multi-scale CNN  | Yes              | SF3D             | 96.7%     | 44.06 M              |
|                 |      |                  |                  | S-DA             | 91.8%     |                      |
| Jegham et al. [40]| 2021| VGG16            | Yes              | MDAD             | 75%       | >65.12 M             |
| LWANet          | 2021 | VGG16            | Yes              | SF3D             | 99.37 ± 0.22% | 1.22 M           |
|                 |      |                  |                  | AUC2D            | 98.45 ± 0.28% |                      |

Among these approaches, Baheti et al. [54] is the only one focusing on lightweight network development. Compared to their work, we achieve a similarly competitive accuracy with only 1.2 M parameters and the aid of an attention mechanism. For some self-collected datasets which are not publicly available, we cannot further test the performance of our network.

Additionally, it should be noted that LWANet works as a general solution towards image classification problems in other fields, including but not limited to agriculture, emotion, and traffic. Based on the lightweight design principle, LWANet has the potential to fit any other datasets and output a model with small file size. It works as an attempt towards the future development of edge-oriented deep learning.
5. Conclusions

High accuracy and low computation cost distracted driving detection approach are with urgent needs. In this paper, we propose a lightweight attention-based VGG network. To reduce the computation cost, we embed the depthwise separable convolution in the original VGG16 network and optimize the redundant layers. Compared to the classic VGG16, the lightweight structure design earns a reduction of 98.32%, 98.16%, and 98.15% on FLOPs, trainable parameters, and model file size, respectively. To improve the model accuracy for a lightweight network, we propose an inverted residual attention module and embed it after convolution layers. Working as a plug-and-play lightweight attention module, the accuracy of LWAVGG is improved by at least 0.5% and achieved 99.37% on the SF3D dataset and 98.45% on the AUC2D dataset. Regarding the 0.1 MB model file size increase, the introduction of IRAM is proved to be cost-effective. We have compared the performance of LWANet with other state-of-the-art approaches. The results demonstrate the superiority of our network.

Limited by the experimental conditions, the model is not tested in the actual driving scene. In future work, we plan to utilize our network directly on video-based distracted driving detection. Compared to image-based detection, temporal context can be involved. Since the network may not need to work frame by frame, the computation cost can be further optimized while still maintaining a high classification accuracy.

Author Contributions: Conceptualization, methodology, software, validation D.C., Z.F. and Y.H.; Formal analysis, investigation D.C. and Z.F.; Data curation, visualization, writing—original draft D.C.; Resources, writing—review and editing, supervision, project administration, funding acquisition, Y.L. and Y.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Key Research and Development Program of China under grant 2021YFC3340500.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References
1. The World Health Organization. Global Status Report on Road Safety. 2018. Available online: https://www.who.int/publications/i/item/9789241565684 (accessed on 11 March 2021).
2. National Highway Traffic Safety Administration. Traffic Safety Facts. 2018. Available online: https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812806 (accessed on 11 March 2021).
3. Koesdwiady, A.; Soua, R.; Karray, F.; Kamel, M.S. Recent Trends in Driver Safety Monitoring Systems: State of the Art and Challenges. IEEE Trans. Veh. Technol. 2016, 66, 4550–4563. [CrossRef]
4. Regan, M.A.; Hallett, C.; Gordon, C.P. Driver distraction and driver inattention: Definition, relationship and taxonomy. Accid. Anal. Prev. 2011, 43, 1771–1781. [CrossRef] [PubMed]
5. Sahayadhas, A.; Sundaraj, K.; Murugappan, M.; Palaniappan, R. A physiological measures-based method for detecting inattention in drivers using machine learning approach. Biocybern. Biomed. Eng. 2015, 35, 198–205. [CrossRef]
6. Wang, Y.; Jung, T.-P.; Lin, C.-T. EEG-Based Attention Tracking During Distracted Driving. IEEE Trans. Neural Syst. Rehabil. Eng. 2015, 23, 1085–1094. [CrossRef]
7. Omerustaoglu, F.; Sakar, C.O.; Kar, G. Distracted driver detection by combining in-vehicle and image data using deep learning. Appl. Soft Comput. 2020, 96, 106657. [CrossRef]
8. Li, Y.; Li, J.; Jiang, X.; Gao, C.; Zhang, T. A Driving Attention Detection Method Based on Head Pose. In Proceedings of the 2019 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), Leicester, UK, 19–23 August 2019; pp. 483–490. [CrossRef]
9. Masood, S.; Rai, A.; Aggarwal, A.; Doja, M.; Ahmad, M. Detecting distraction of drivers using Convolutional Neural Network. Pattern Recognit. Lett. 2018, 139, 79–85. [CrossRef]
10. Abouelnaga, Y.; Eraqi, H.M.; Moustafa, M.N. Real-time distracted driver posture classification. arXiv 2017, arXiv:1706.09498. [CrossRef]
11. Dhakate, K.R.; Dash, R. Distracted Driver Detection using Stacking Ensemble. In Proceedings of the 2020 IEEE International Students’ Conference on Electrical, Electronics and Computer Science (SCEECS), Bhopal, India, 22–23 February 2020; pp. 1–5. [CrossRef]

12. Xing, Y.; Lv, C.; Wang, H.; Cao, D.; Velenis, E.; Wang, F.-Y. Driver Activity Recognition for Intelligent Vehicles: A Deep Learning Approach. *IEEE Trans. Veh. Technol.* 2019, 68, 5379–5390. [CrossRef]

13. Huang, C.; Wang, X.; Cao, J.; Wang, S.; Zhang, Y. HCF: A Hybrid CNN Framework for Behavior Detection of Distracted Drivers. *IEEE Access* 2020, 8, 109335–109349. [CrossRef]

14. Mase, J.M.; Chapman, P.; Figueroedo, G.P.; Torres, M.T. A Hybrid Deep Learning Approach for Driver Distraction Detection. In Proceedings of the 2020 International Conference on Information and Communication Technology Convergence (ICTC), Jeju, Korea, 21–23 October 2020; pp. 1–6. [CrossRef]

15. Tang, M.; Wu, F.; Zhao, L.-L.; Liang, Q.-P.; Lin, J.-W.; Zhao, Y.-B. Detection of Distracted Driving Based on MultiGranularity and Middle-Level Features. In Proceedings of the 2020 Chinese Automation Congress (CAC), Shanghai, China, 6–8 November 2020; pp. 2717–2722. [CrossRef]

16. Hu, Y.; Lu, M.; Lu, X. Driving behaviour recognition from still images by using multi-stream fusion CNN. *Mach. Vis. Appl.* 2018, 30, 851–865. [CrossRef]

17. Lu, M.; Hu, Y.; Lu, X. Driver action recognition using deformable and dilated faster R-CNN with optimized region proposals. *Appl. Intell.* 2019, 50, 1100–1111. [CrossRef]

18. Rao, X.; Lin, F.; Chen, Z.; Zhao, J. Distracted driving recognition method based on deep convolutional neural network. *J. Ambient Intell. Humaniz. Comput.* 2019, 12, 193–200. [CrossRef]

19. Han, S.; Mao, H.; Dally, W.J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv* 2015, arXiv:1510.00149. [CrossRef]

20. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv* 2017, arXiv:1704.04861. [CrossRef]

21. Yang, Z.; Ma, X.; An, J. Asymmetric Convolution Networks Based on Multi-feature Fusion for Object Detection. In Proceedings of the 2020 IEEE 16th International Conference on Automation Science and Engineering (CASE), Hong Kong, China, 20–21 August 2020; pp. 1355–1360. [CrossRef]

22. Chen, Y.; Fan, H.; Xu, B.; Yan, Z.; Kalantidis, Y.; Rohrbach, M.; Shuicheng, Y.; Feng, J. Drop an Octave: Reducing Spatial Redundancy in Convolutional Neural Networks with Octave Convolution. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27 October–2 November 2019. [CrossRef]

23. Henderson, J.M.; Hayes, T.R. Meaning-based guidance of attention in scenes as revealed by meaning maps. *Nat. Hum. Behav.* 2017, 1, 743–747. [CrossRef] [PubMed]

24. Zhang, B.; Xiong, D.; Su, J. Neural Machine Translation with Deep Attention. *IEEE Trans. Pattern Anal. Mach. Intell.* 2018, 40, 154–163. [CrossRef]

25. Nguyen, M.T.; Siritanawan, P.; Kotani, K. Saliency detection in human crowd images of different density levels using attention mechanism. *Signal Process. Image Communic.* 2020, 88, 115976. [CrossRef]

26. Deng, Z.; Jiang, Z.; Lan, R.; Huang, W.; Luo, X. Image captioning using DenseNet network and adaptive attention. *Signal Process. Image Communic.* 2020, 85, 115836. [CrossRef]

27. Liu, Y.; Wang, W.; Hu, Y.; Hao, J.; Chen, X.; Gao, Y. Multi-Agent Game Abstraction via Graph Attention Neural Network. *Proc. Conf. AAAI Artif. Intell. 2020*, 34, 7211–7218. [CrossRef]

28. Hu, J.; Shen, L.; Sun, G. Squeeze-and-Excitation Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* 2020, 42, 2011–2023. [CrossRef]

29. Woo, S.; Park, J.; Lee, J.-Y.; Kweon, I.S. CBam: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 3–19.

30. Jin, B.; Xu, Z. EAC-Net: Efficient and Accurate Convolutonal Network for Video Recognition. *Proc. Conf. AAAI Artif. Intell. 2020*, 34, 11149–11156. [CrossRef]

31. Misra, D.; Nalamada, T.; Arasanipalai, A.U.; Hou, Q. Rotate to attend: Convolutional triplet attention module. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Waikoloa, HI, USA, 3–8 January 2021; pp. 3139–3148.

32. He, Z.; He, D. Bilinear Squeeze-and-Excitation Network for Fine-Grained Classification of Tree Species. *IEEE Geosci. Remote Sens. Lett.* 2020, 17, 1139–1143. [CrossRef]

33. Xie, L.; Huang, C. A Residual Network of Water Scene Recognition Based on Optimized Inception Module and Convolutional Block Attention Module. In Proceedings of the 2019 6th International Conference on Systems and Informatics (ICSAI), Shanghai, China, 2–4 November 2019; pp. 1174–1178. [CrossRef]

34. Chen, Y.; Zhang, X.; Chen, W.; Li, Y.; Wang, J. Research on Recognition of Fly Species Based on Improved RetinaNet and CBAM. *IEEE Access* 2020, 8, 102907–102919. [CrossRef]

35. Wang, H.; Wang, S.; Qin, Z.; Zhang, Y.; Li, R.; Xia, Y. Triple attention learning for classification of 14 thoracic diseases using chest radiography. *Med. Image Anal.* 2020, 67, 101846. [CrossRef] [PubMed]

36. Pande, S.; Banerjee, B. Adaptive hybrid attention network for hyperspectral image classification. *Pattern Recognit. Lett.* 2022, 144, 6–12. [CrossRef]

37. Wang, Q.; Wang, J.; Zhou, M.; Li, Q.; Wen, Y.; Chu, J. A 3D attention networks for classification of white blood cells from microscopy hyperspectral images. *Opt. Laser Technol.* 2021, 139, 106931. [CrossRef]
38. Hu, Y.; Lu, M.; Lu, X. Feature refinement for image-based driver action recognition via multi-scale attention convolutional neural network. *Signal Processing Image Commun.* 2020, *81*, 115697. [CrossRef]

39. Wang, W.; Lu, X.; Zhang, P.; Xie, H.; Zeng, W. Driver Action Recognition Based on Attention Mechanism. In Proceedings of the 2019 6th International Conference on Systems and Informatics (ICSAI), Shanghai, China, 2–4 November 2019; pp. 1255–1259. [CrossRef]

40. Jegham, I.; Ben Khalifa, A.; Alouani, I.; Mahjoub, M.A. Soft Spatial Attention-Based Multimodal Driver Action Recognition Using Deep Learning. *IEEE Sens. J.* 2020, *21*, 1918–1925. [CrossRef]

41. Kuan, D.T.; Sawchuk, A.A.; Strand, T.C.; Chavel, P. Adaptive Noise Smoothing Filter for Images with Signal-Dependent Noise. *IEEE Trans. Pattern Anal. Mach. Intell.* 1985, *7*, 165–177. [CrossRef] [PubMed]

42. Jegham, I.; Ben Khalifa, A.; Alouani, I.; Mahjoub, M.A. Soft Spatial Attention-Based Multimodal Driver Action Recognition Using Deep Learning. *IEEE Sens. J.* 2020, *21*, 1918–1925. [CrossRef] [PubMed]

43. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* 2014, arXiv:1409.1556. [CrossRef]

44. State Farm. State Farm Distracted Driver Detection Dataset. 2016. Available online: https://www.kaggle.com/c/state-farm-distracted-driver-detection/overview (accessed on 11 March 2021).

45. 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, *2019*, 1–12. [CrossRef]

46. Hou, R.; Zhao, Y.; Hu, Y.; Liu, H. No-reference video quality evaluation by a deep transfer CNN architecture. *Signal Process. Image Commun.* 2020, *83*, 115782. [CrossRef]

47. Zhang, B. Apply and Compare Different Classical Image Classification Method: Detect Distracted Driver; Stanford CS 229 Project Reports; 2016. Available online: http://merrin5.mdpi.lab/public/tools/acs_final_check (accessed on 11 March 2021).

48. Okon, O.D.; Meng, L. Detecting Distracted Driving with Deep Learning. In Proceedings of the International Conference on Interactive Collaborative Robotics, Hatfield, UK, 12–16 September 2017; Springer: Berlin/Heidelberg, Germany; Volume 10459, pp. 170–179. [CrossRef]

49. Hssayeni, M.D.; Saxena, S.; Ptucha, R.; Savakis, A. Distracted Driver Detection: Deep Learning vs Handcrafted Features. *IS&T Int. Symp. Electron. Imaging* 2017, *29*, 20–26. [CrossRef]

50. Hssayeni, M.D.; Saxena, S.; Ptucha, R.; Savakis, A. Distracted Driver Detection: Deep Learning vs Handcrafted Features. *IS&T Int. Symp. Electron. Imaging* 2017, *29*, 20–26. [CrossRef]

51. Ai, Y.; Xia, J.; She, K.; Long, Q. Double Attention Convolutional Neural Network for Driver Action Recognition. In Proceedings of the 3rd International Conference on Electronic Information Technology and Computer Engineering (EITCE), Xiamen, China, 18–20 October 2019; pp. 1515–1519. [CrossRef]

52. Jamsheed, A.; Janet, B.; Reddy, U.S. Real Time Detection of driver distraction using CNN. In Proceedings of the 2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT), Tirunelveli, India, 20–22 August 2020; pp. 185–191. [CrossRef]

53. Baheti, B.V.; Talbar, S.; Gajre, S. Towards Computationally Efficient and Realtime Distracted Driver Detection with MobileVGG Network. *IEEE Trans. Intell. Veh.* 2020, *5*, 565–574. [CrossRef]