Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/mbe

MBE, 20(10): 18248–18266.
DOI: 10.3934/mbe.2023811
Received: 08 July 2023
Revised: 25 August 2023
Accepted: 12 September 2023
Published: 22 September 2023

Research article

Driver distraction detection based on lightweight networks and tiny object detection

Zhiqin Zhu[1], Shaowen Wang[1], Shuangshuang Gu[1], Yuanyuan Li[1], Jiahan Li[1], Linhong Shuai[2] and Guanqiu Qi[3,*]

[1] College of Automation, Chongqing University of Posts and Telecommunications, Chongqing 400065, China

[2] Intelligent Interaction R&D Department, Chongqing Lilong Zhongbao Intelligent Technology Co., Chongqing 400065, China

[3] Computer Information Systems Department, State University of New York at Buffalo State, Buffalo, NY 14222, USA

* **Correspondence:** Email: qig@buffalostate.edu; Tel: +1 (716) 878-4929.

Abstract: Real-time and efficient driver distraction detection is of great importance for road traffic safety and assisted driving. The design of a real-time lightweight model is crucial for in-vehicle edge devices that have limited computational resources. However, most existing approaches focus on lighter and more efficient architectures, ignoring the cost of losing tiny target detection performance that comes with lightweighting. In this paper, we present MTNet, a lightweight detector for driver distraction detection scenarios. MTNet consists of a multidimensional adaptive feature extraction block, a lightweight feature fusion block and utilizes the IoU-NWD weighted loss function, all while considering the accuracy gain of tiny target detection. In the feature extraction component, a lightweight backbone network is employed in conjunction with four attention mechanisms strategically integrated across the kernel space. This approach enhances the performance limits of the lightweight network. The lightweight feature fusion module is designed to reduce computational complexity and memory access. The interaction of channel information is improved through the use of lightweight arithmetic techniques. Additionally, CFSM module and EPIEM module are employed to minimize redundant feature map computations and strike a better balance between model weights and accuracy. Finally, the IoU-NWD weighted loss function is formulated to enable more effective detection of tiny targets. We assess the performance of the proposed method on the LDDB benchmark. The experimental results demonstrate that our proposed method outperforms multiple advanced detection models.

Keywords: driver distraction detection; lightweight network; tiny object detection

1. Introduction

Object detection plays a crucial role in monitoring driver distractions. Since the introduction of AlexNet in 2012 [1], deep convolutional neural networks have become widely adopted in computer vision [1, 2]. These networks have increasingly deeper depths, larger number of parameters and more complex structures, all in pursuit of higher accuracy in major rankings. However, this comes at the expense of increased network depth and computation. Today, one-stage object detectors [3, 4] are popular in real-time applications due to their excellent speed and accuracy tradeoffs. Currently, one-stage object detectors [3, 4] are preferred in real-time applications due to their exceptional balance of speed and accuracy. The YOLO series, particularly YOLOv5 [5], stands out as the most prominent architecture among one-stage detectors. It has achieved an excellent trade-off between accuracy and speed on the COCO dataset, making it widely adopted in the industry. In the context of driver distraction monitoring [6, 7], real-time computation is essential. The model must be deployed on in-vehicle edge devices, which have limited CPU performance and memory. To deploy real-time computation algorithms on these low-powered devices, their computing power must be fully optimized. Furthermore, the algorithms themselves need to be lightweight. The emergence of lightweight networks has focused attention on the trade-off between speed and accuracy. The challenge lies in making the network even more lightweight and compact for easy deployment, while still preserving accuracy. Additionally, as network optimization aims for acceleration, there is a need to improve the accuracy in detecting challenging objects like tiny ones.

We have contributed to addressing the aforementioned concern in the following ways:

1) We propose a deep learning-based method for monitoring driver distraction that optimizes the efficiency of detecting small targets while reducing the weight of the model and reducing computational complexity.

2) We propose a multidimensional adaptive feature extraction block (MAFEB) that is integrated into the network's four attention mechanisms across the kernel space using a parallel strategy to enhance its performance bounds. Additionally, we reduce the model parameters and storage space by combining convolutional and batch normalization layers, resulting in a lighter model and accelerated network inference.

3) We introduce the Lightweight Feature Fusion Block (LFFB), which consists of a CFSM module and an EPIEM module, to effectively reduce computational complexity and memory access without significant loss of accuracy.

4) We propose the IoU-NWD weighted loss function to address the issue of degraded detection capability caused by the sensitivity of the IoU metric to small targets. This loss function aims to improve the overall performance of the detector in detecting tiny targets.

The remaining portion of this paper is structured as follows: Section 2 provides an overview of the related work. Section 3 presents a detailed description of the proposed method. Section 4 presents the experimental results and corresponding discussion. Lastly, Section 5 concludes the paper.

2. Materials and methods

2.1. Related work

2.1.1. Driver distraction detection

Inspired by the advancements in deep learning for object detection, several convolutional neural network (CNN) models have been recently proposed for driver distraction detection. J. V. Abdul et al. [8] utilized a CNN-based approach to enhance the accuracy of classifying distracted drivers by studying driver actions from an image dataset. C. Huang et al. [9] introduced a CNN framework (HCF) that combined ResNet50, Inception V3 [10] and Xception [11] to achieve high accuracy in classifying distracted drivers' behavior through deep learning-based image feature processing. S. Faiqa et al. [12] explored the EfficientNet architecture for designing EfficientDet, utilizing EfficientNet as the backbone network for the initial section, BiFPN architecture for feature extraction and a characterization and location box prediction network for grouping and identifying diverted drivers. L. N. Duy et al. [13] proposed a lightweight CNN architecture for driver behavior identification. The architecture leverages standard convolutional and depth-separable convolutional operations, adaptive connectivity for feature extraction and a CBAM attention mechanism to focus on salient features. The network parameters are reduced, while improving task classification accuracy. Furthermore, Zhu et al. [38] introduced the MT-DTA model, which combines an autoencoder with an attention model in a cascade structure of CNNs.

2.1.2. Efficient CNNs

SqueezeNet [14] proposed Fire modules that consist of Squeeze and Expand operations in order to compress the model and achieve lightweighting of the network. MobileNet [15–17] incorporates deep separable convolutions instead of standard convolutions to maintain high accuracy while reducing the number of parameters. This approach provides guiding principles for the design of subsequent lightweight networks. ShuffleNet [18, 19] is specifically designed for mobile devices with limited computational power. To simplify the model and overcome the computational complexity of point convolutions, point group convolutions and channel shuffling techniques are employed. Xception [11] replaces the traditional Inception [10, 20–22] blocks with depthwise separable convolutions to achieve complete decoupling of cross-channel correlation and spatial correlation. The EfficientNet family [23, 24] introduces a new baseline network by scaling the three dimensions of depth, width and resolution with simple and efficient composite coefficients. GhostNet [25] introduces an inexpensive linear operation to generate more feature maps.

2.1.3. Tiny object detection

Object detection has been an important research direction in the field of computer vision and pattern recognition research, where tiny object detection is more challenging because small objects do not contain detailed information and may even disappear in the deep network. In order to solve the above problems, researchers have made some improvements to small target detection in terms of loss function. J. He et al. [26] designed a new formulation alpha-IoU that generalizes the existing IoU-based losses to a new family of power IoU losses. By modulating the power parameter alpha, alpha-IoU provides the flexibility to achieve different degrees of bbox regression accuracy when

training the target detector. On the other hand, researchers have also made some improvements on the feature pyramid network to detect small targets more accurately. C. Deng et al. [27] proposed the extended feature pyramid network (EFPN) with additional high-resolution pyramid layers, using the feature texture transfer (FTT) module to extract both super-resolution features and plausible region details. Some researchers have also designed detectors for small target detection. X. Yang et al. [28] designed a sampling fusion network incorporating multilayer features and effective anchor sampling to improve the sensitivity to small targets. Also, supervised pixel attention network and channel attention network were explored for small and cluttered target detection together by suppressing noise and highlighting target features. Zhu et al. [39] aim to achieve a fuller utilization of multimodal information based on the fusion of features that have special meaning and importance.

2.2. The proposed method

2.2.1. Overview

Generally, a convolutional neural network (CNN) detector comprises three major components: the backbone, the neck and the head. The backbone is responsible for extracting the input features, while the neck improves the assignment and fusion of these features before they are passed to the head. The head then detects objects using the input features from the neck. In Figure 1, we demonstrate the framework of our proposed driver distraction detection method. This framework mostly consists of a multidimensional adaptive feature extraction block and a lightweight feature fusion block. The multidimensional feature extraction block utilizes lightweight convolutional networks based on channel sparse convolution as the backbone. Additionally, we introduce the multidimensional adaptive module (MDAM) to provide the depth-separable convolutional kernel with dynamic properties. This module enables the dynamic extraction of feature information at varying scales and dimensions, effectively reducing computational effort.

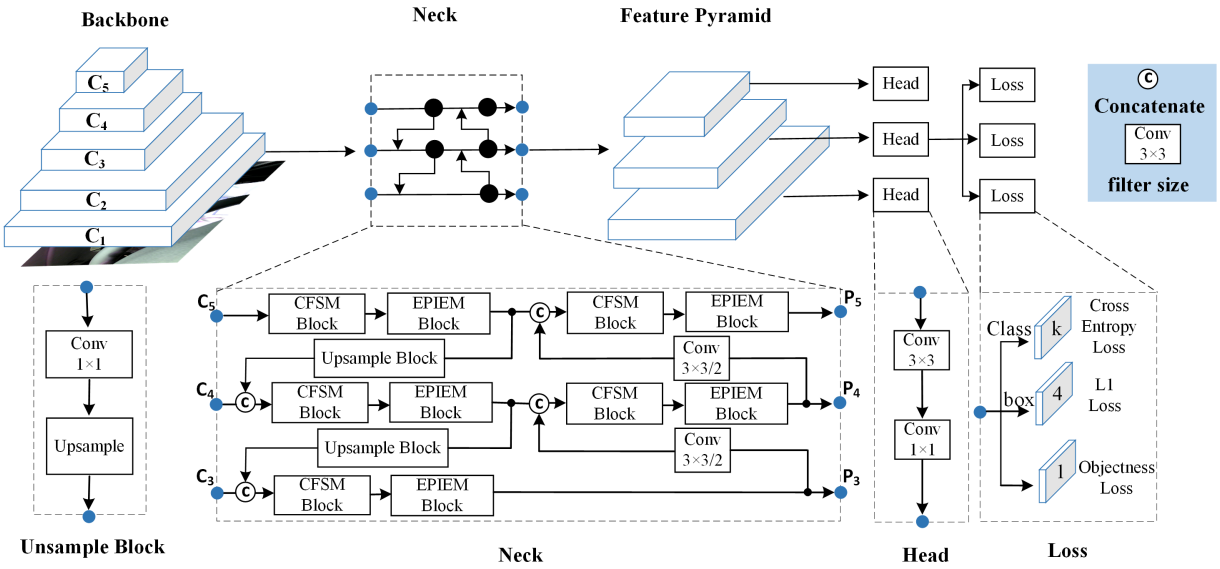


Figure 1. Overall Architecture of MTNet.

The Channel Feature Shuffle Module (CFSM) utilizes depth-separable convolution and channel shuffling operations to facilitate information interaction between channel-dense convolution and depth-separable convolution. This approach effectively reduces the model's weight while maintaining accuracy. The lightweight feature fusion block consists primarily of the CFSM and the Efficient Partial Information Extraction Module (EPIEM). These modules aim to decrease computational complexity and the number of memory accesses required for feature fusion across different scales. EPIEM optimizes costs by considering feature map redundancy. It employs partial convolution to reduce redundant computations and minimize frequent memory accesses, resulting in more efficient extraction of spatial features. In order to address the sensitivity of IoU to small deviations in target position, the NWD metric is introduced as a more appropriate measure for similarity between two bounding boxes. This, however, may negatively impact convergence speed. To mitigate this issue, a weighting factor is applied to NWD, which is then integrated into the loss function of the detector.

2.2.2. Multi-dimensional feature extraction block

We incorporate the improved multidimensional adaptive attention into a lightweight convolutional neural network to enhance feature extraction in the multidimensional feature extraction block. This enables us to inject features with multidimensional information to improve the detection task. The feature extraction architecture consists of multiple convolutional layers and residual blocks. Within these layers and blocks, the inclusion of MDAM extends the performance capabilities of lightweight CNNs. This is achieved by integrating attention mechanisms into the convolutional blocks, allowing for better adaptation to the detection of small targets.

The implementation details of MDAM are illustrated in Figure 2. This approach incorporates a novel multidimensional attention mechanism, which computes four complementary types of attention αSi, αCi, αfi and αwi of Wi along four dimensions of the kernel space. These dimensions include the number of convolutional kernels, the size of the convolutional kernel space, the number of input channels and the number of output channels. The parallel computation empowers the convolution kernel to capture diverse contextual information. Moreover, the adoption of a single kernel space structure in MDAM strikes a better balance between model accuracy and efficiency in comparison to existing dynamic convolution designs. MDAM utilizes a novel multidimensional attention mechanism with parallel strategies to learn the complementary attention of convolution nuclei along all four dimensions of the kernel space in any convolution layer. It is worth stating that the four kinds of attention learned are complementary to each other, and the order does not matter.

As denoted in Figure 3, the MDAM block can be defined as follows:

$$y = \left(\alpha_{w1} \odot \alpha_{f1} \odot \alpha_{c1} \odot \alpha_{s1} \odot W_1 + \cdots + \alpha_{wn} \odot \alpha_{fn} \odot \alpha_{cn} \odot \alpha_{sn} \odot W_n \right) * x, \tag{2.1}$$

where $\alpha_{\mathrm{wi}} \in R$ denotes the attention scalar of the convolution kernel wi; $\alpha_{\mathrm{si}} \in R_{k\times k}$, $\alpha_{\mathrm{si}} \in R_{Cin}$ and $\alpha_{\mathrm{si}} \in R_{Cout}$ denote the spatial dimension, input channel dimension and output channel dimension computations along the kernel space of the convolution kernel Wi, respectively, and $\odot$ denotes the multiplication operations along different dimensions of the kernel space.

We assume a convolutional kernel (W) for the convolutional layer. The convolution process involves performing a sliding window computation on the input feature map using the convolutional kernel (W). Assuming w as an element in the convolutional kernel (W) and x as an element in the input feature

map, the computation process for w and x is as follows:

$$y_{\text{conv}} = w \cdot x + b. \tag{2.2}$$

The BN layer requires calculating the mean and variance of the elements in a minibatch. Next, the mean value is subtracted from each element and then divided by the standard deviation. Finally, the BN output is obtained by performing an affine transformation using and γ, β as outlined below.

$$\mu_B \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i, \tag{2.3}$$

$$\sigma_B^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_B)^2, \tag{2.4}$$

$$\hat{x}_i \leftarrow \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}, \tag{2.5}$$

$$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv \mathrm{B}_{\gamma,\beta}(x_i). \tag{2.6}$$

Convolutional layers and BN layers can both be regarded as extensions of linear layers. However, they differ in terms of their specific implementation and mechanism of action. Consequently, these two aspects are combined:

$$BN_{\gamma,\beta}(x) = \gamma\frac{w \cdot x + b - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} + \beta = \frac{\gamma \cdot w}{\sqrt{\sigma_B^2 + \epsilon}} \cdot x + \frac{\gamma}{\sqrt{\sigma_B^2 + \epsilon}} \cdot (b - \mu_B) + \beta, \tag{2.7}$$

obtained as:

$$\hat{w} = \frac{\gamma \cdot w}{\sqrt{\sigma_B^2 + \epsilon}}, \tag{2.8}$$

$$\hat{b} = \frac{\gamma}{\sqrt{\sigma_B^2 + \epsilon}} \cdot (b - \mu_B) + \beta, \tag{2.9}$$

as a result:

$$BN_{\gamma,\beta}(x) = \hat{w} \cdot x + \hat{b}. \tag{2.10}$$

The amount of computation is reduced using operator fusion techniques, decrease the overall throughput of the process, enhance the localization of computation, consequently enhancing efficiency.

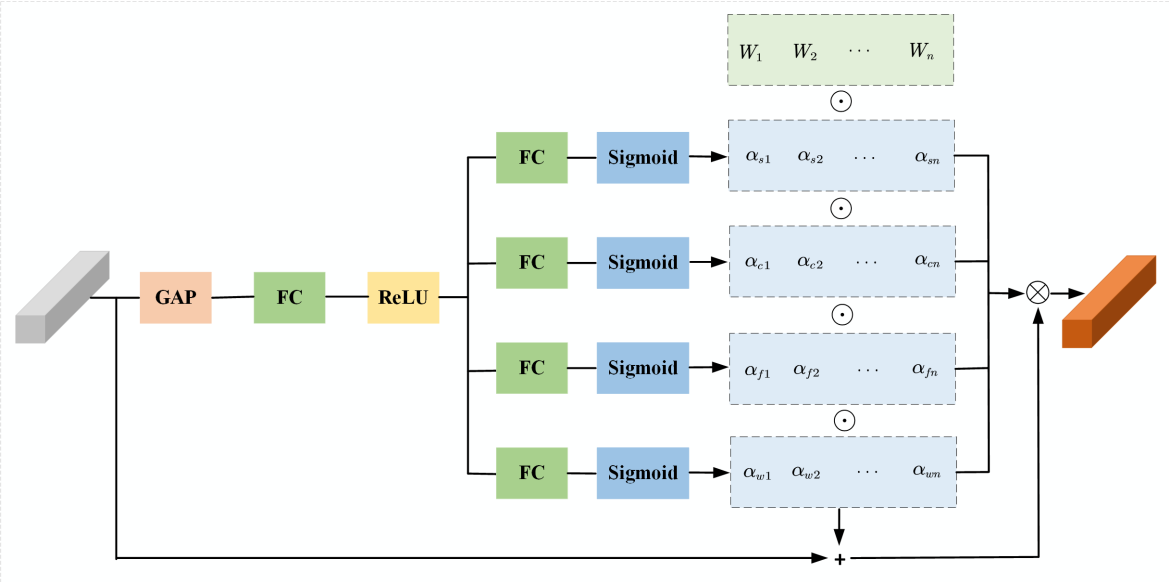


Figure 2. Architecture of Multidimensional Adaptive Modules (MDAM).

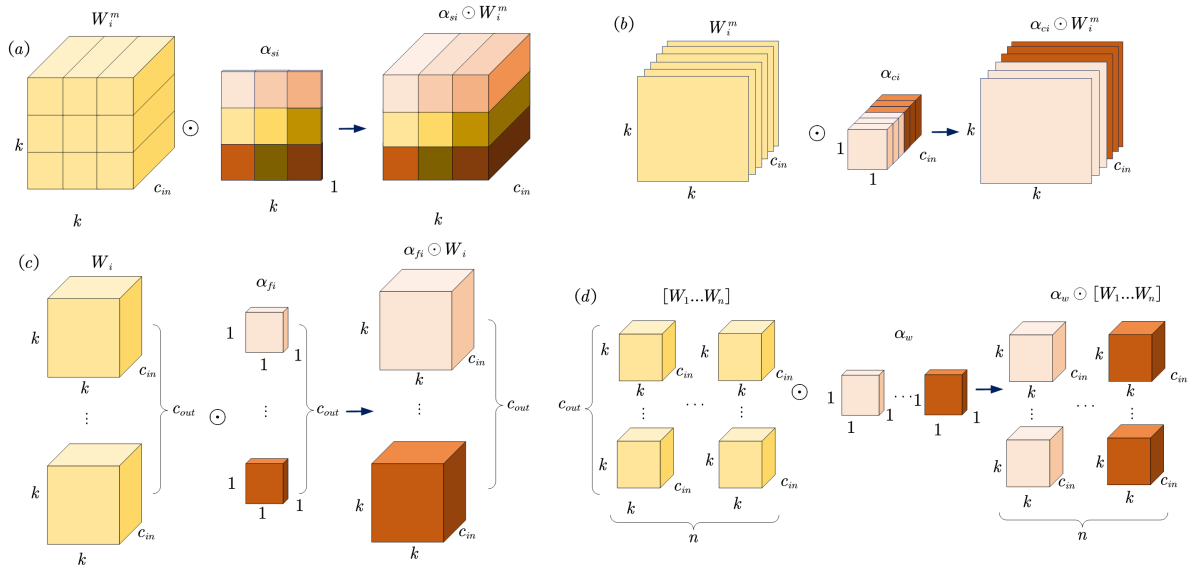


Figure 3. Four kinds of attention along the kernel dimension.

2.2.3. Lightweight feature fusion block

Studies in neuroscience have demonstrated that models with a larger number of neurons tend to exhibit stronger nonlinear representation. However, it is important to acknowledge that the human brain outperforms these models in terms of its superior information processing capabilities and low power consumption. Consequently, in the current stage of vision-based specific assisted driving systems, it is possible to optimize computational costs without introducing additional operations and achieve significant accuracy improvements through lightweight designs. Nonetheless, the utilization

of deep separable convolution (DSC) in lightweight networks, while common, results in the separation of channel information during computation. Over-reliance on DSC alone causes the feature extraction and fusion capabilities of the network to deteriorate, consequently reducing the overall performance of the model. Therefore, enhancing the interaction capability of inter-channel information is crucial to obtain richer features during the lightweight optimization of the network.

We introduce the Channel Feature Shuffle Module (CFSM), which comprises a channel-dense standard convolution (Conv) and a channel-information-separated Depthwise Convolution (DWConv). As depicted in Figure 4, the CFSM receives the image as input and facilitates the exchange of information between Conv and DWConv through shuffling. Shuffling is used to uniformly and effectively mix the information generated by Conv and DWConv. Initially, the input feature map is split into two groups, each with half the original number of channels. These two groups are then blended, with a 2-channel interval, and interleaved along the channel dimension to create a new feature map output. This approach enables Conv's information to enhance the model's representation and generalization performance by uniformly exchanging local feature information across channels and fully integrating it into the DSC output.

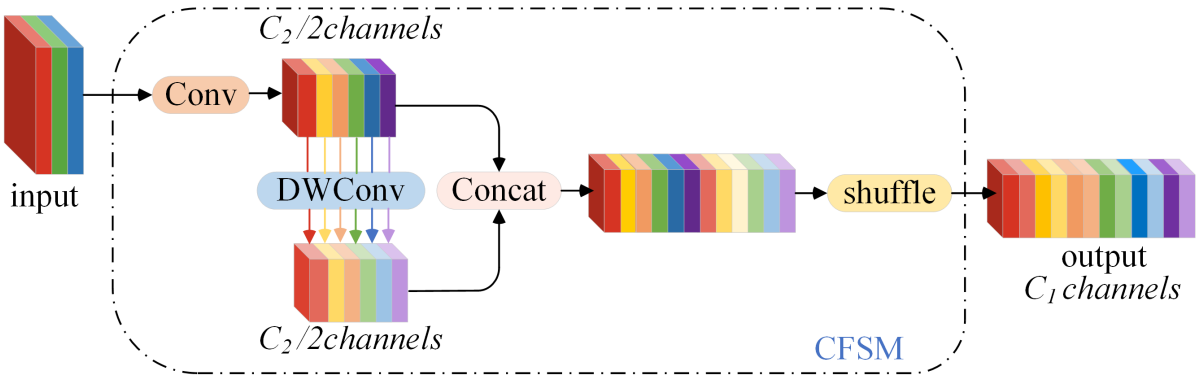


Figure 4. Design of Channel Feature Shuffle Module (CFSM).

The visualization in Figure 5 illustrates that the feature maps of the different channels exhibit a high degree of similarity during the network computation. Many other lightweight optimisation schemes [29, 30] address this redundant feature, but neglect to explore efficient and simple utilisation. We reduce both redundant computation and memory access by introducing the Efficient Partial Information Extraction Module (EPIEM) to improve the extraction of spatial features.

The module, as depicted in Figure 6, is implemented using the partial convolution (PConv) and pointwise convolution techniques. PConv performs standard convolutions (SC) on a subset of the input channel to extract spatial features, while disregarding interactions with other channels. The number of channels for SC is determined by the occupancy factor r = cp/c, typically set to 1/4 for balancing computational complexity and reusability of channel features. Consequently, PConv requires only 1/16 of the computational effort (FLOPs) compared to standard convolution, leaving the remaining channels unchanged. To fully and efficiently utilize information from all channels, a pointwise convolution (PWConv) is subsequently applied after PConv.

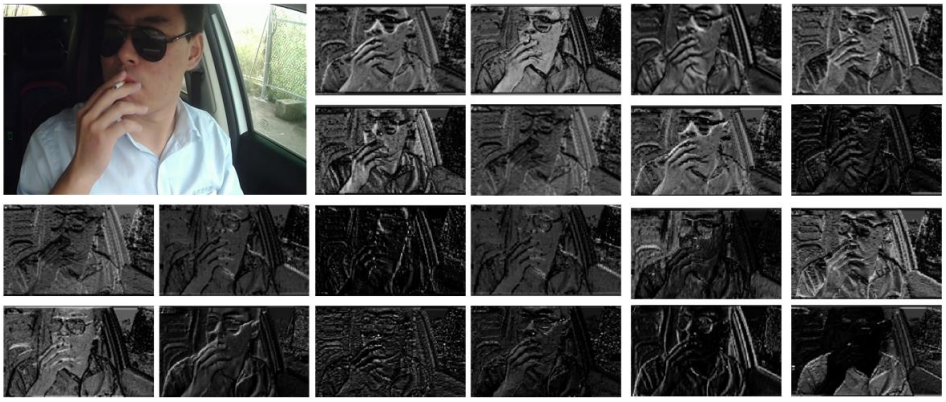


Figure 5. Visualization of redundant feature maps.

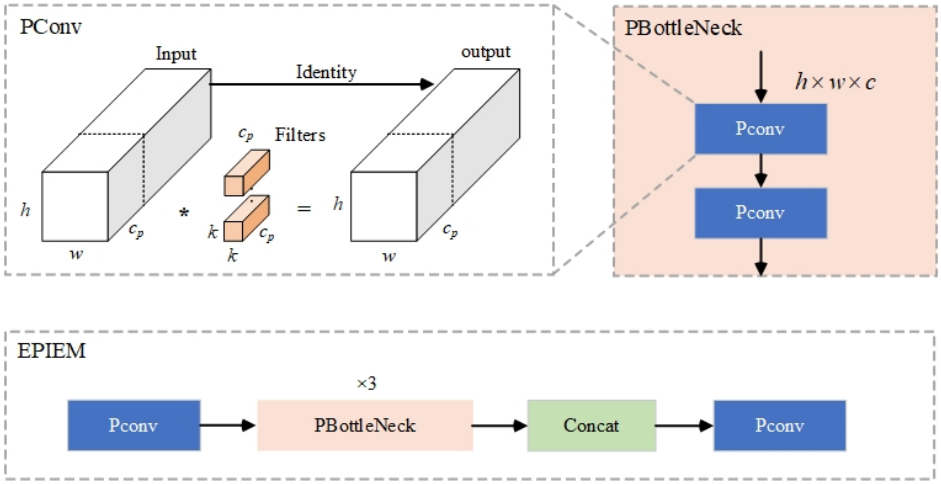


Figure 6. Design of Efficient Partial Information Extraction Module (EPIEM).

2.2.4. Weighted IoU-NWD loss function for tiny targets

The proposed detection network is trained by three loss functions, Lcls, Lobj and Lloc. Lcls is used to calculate the loss of classification, using the binary cross entropy loss (BCE), which calculates only the loss of classification of positive samples. Lobj calculates the CIOU of the target bounding box and the ground truth box (GT Box) of the network prediction, which calculates only the loss of all samples. Lloc is the loss of localization, using the CIOU Loss, which calculates only the loss of localization of positive samples. Here, $\lambda 1$, $\lambda 2$ and $\lambda 3$ are the balance coefficients.

$$\text{Loss} = \lambda_1 \text{ Lcls} + \lambda_2 \text{ Lobj} + \lambda_3 \text{ Lloc}. \tag{2.11}$$

Current intersection-over-union (IoU) based metrics, including IoU itself and its extensions, are highly susceptible to positional deviations in the context of small objects. Moreover, even a slight pixel deviation between small and large targets can result in substantial fluctuations in IoU values, thereby

negatively impacting the detection performance of anchor-based detectors. As depicted in Figure 7, each grid in the diagram corresponds to one pixel. Specifically, box A represents the true bounding box, whereas boxes B and C represent the predicted bounding boxes. When the deviation in position is identical, it has minimal impact on the intersection-over-union (IoU) value for normal-sized objects. However, with regard to tiny objects, even a slight position deviation can result in a notable decrease in the IoU value. Consequently, the IoU-based loss function is suboptimal for detecting small targets. Drawing inspiration from Wasserstein's novel metric for detecting small targets [31], we propose an IoU-NWD weighted loss function that aims to enhance the performance of small target detection. This is achieved by fine-tuning the weighting factor α, which is contingent upon the proportion of tiny targets present in the dataset.

$$\text{Lobj} = \alpha_1 L_{IoU} + \alpha_2 L_{NWD}, \tag{2.12}$$

$$\alpha_2 = 1 - \alpha_1. \tag{2.13}$$

The bounding box is initially represented as a 2D Gaussian distribution. The similarity between the resulting Gaussian distributions is then quantified using the proposed metric, known as Normalized Wasserstein Distance (NWD) [32]. NWD, unlike the IOU metric, takes into account the width-to-height ratio of the bounding box and is well-suited for handling boxes with different shapes. IOU, on the other hand, fails to properly account for boxes that vary significantly in their width-to-height ratio. Additionally, the NWD metric is robust to scale variations and can measure the similarity between two distributions even when the bounding boxes either do not overlap or have minimal overlap. As a result, NWD outperforms IOU significantly when it comes to detecting small objects. To account for the NWD metric's influence on convergence speed, we incorporate it into the component of the L_{obj} loss function, which improves the ability to detect small targets.

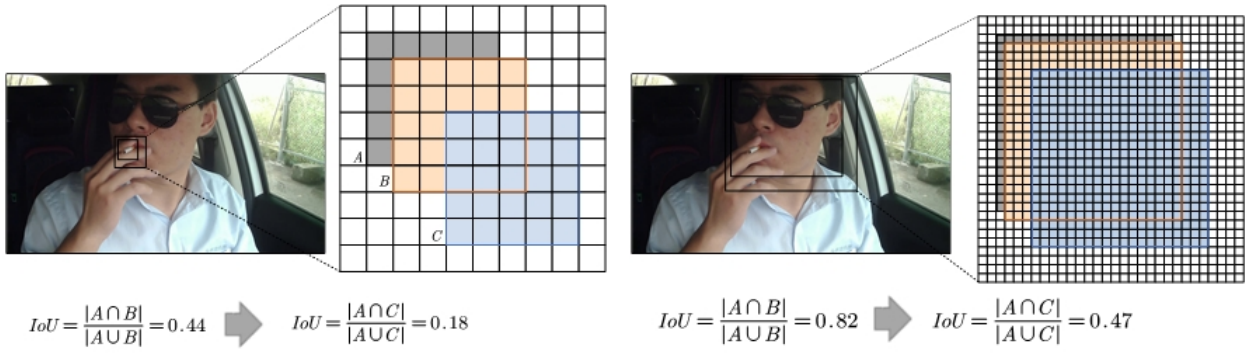


Figure 7. Sensitivity of the IoU metric to small target position offsets.

We use the Wasserstein distance from optimal transport theory to calculate the distribution distance. For two two-dimensional Gaussian distributions $\mu_1 = N(m_1, \Sigma_1)$ and $\mu_2 = N(m_2, \Sigma_2)$, the second-order Wasserstein distance between μ_2 and μ_2 is defined for:

$$W_2^2(\mu_1, \mu_2) = \|m_1 - m_2\|_2^2 + \text{Tr}\left(\Sigma_1 + \Sigma_2 - 2\left(\Sigma_2^{\frac{1}{2}}\Sigma_1\Sigma_2^{\frac{1}{2}}\right)^{\frac{1}{2}}\right). \tag{2.14}$$

It can be simplified as follows:

$$\mathrm{W}_2^2\left(\mu_1, \mu_2\right) = \left\|m_1 - m_2\right\|_2^2 + \left\|\Sigma_1^{\frac{1}{2}} - \Sigma_2^{\frac{1}{2}}\right\|_F^2, \tag{2.15}$$

where $\|\cdot\|_F$ is the Frobenius parametrization.

Furthermore, for the Gaussian distributions Na and Nb modeled by the bounding boxes $\mathrm{A} = (cx_a, cy_a, w_a, h_a)$ and $\mathrm{B} = (cx_b, cy_b, w_b, h_b)$, Eq 6 can be further simplified as.

$$\mathrm{W}_2^2\left(N_a, N_b\right) = \left\|\left(\left[cx_a, cy_a, \frac{w_a}{2}, \frac{h_a}{2}\right]^T, \left[cx_b, cy_b, \frac{w_b}{2} \cdot \frac{h_b}{2}\right]^T\right)\right\|_2^2. \tag{2.16}$$

However, $\mathrm{W}_2^2\left(N_a, N_b\right)$ is a distance metric and cannot be used directly as a similarity metric (i.e., values between $0 \sim 1$ as IoU). Therefore, we use its exponential form of normalization to obtain a new metric called Normalized Wasserstein Distance (NWD):

$$NWD\left(N_a, N_b\right) = \exp\left(-\frac{\sqrt{W_2^2\left(N_a, N_b\right)}}{C}\right), \tag{2.17}$$

where C is a constant that is closely related to the dataset. In the next experiments, we empirically set C to the average absolute size of the dataset and obtain the best performance.

To deal with the above problem, we design the NWD metric as a loss function L_{NWD} and apply it to L_{obj} .

$$L_{NWD} = 1 - NWD\left(N_a, N_b\right). \tag{2.18}$$

3. Experiments

3.1. Experiment environment

We employed the PyTorch framework to construct our models. The models were trained using a dual-card setup consisting of an NVIDIA TITAN RTX or an NVIDIA GeForce RTX 4090 running on the Windows 10 operating system. The hyperparameters used are as follows: the optimizer utilized is stochastic gradient descent; the batch size is set to 32; a linear decay learning rate scheduling strategy is implemented with an initial learning rate of 0.01 and a cyclic learning rate with the same value; the momentum and weight decay values are set to 0.937 and 0.0005, respectively. All validation experiments were conducted on the NVIDIA GeForce RTX 4090. In the tables and figures, we define AP0.5 as the average accuracy across all categories when evaluating accuracy with an Intersection over Union (IoU) threshold of 0.5. Additionally, AP represents the average accuracy with IoU thresholds ranging from 0.5 to 0.95 in increments of 0.05, calculated as a weighted average.

3.2. Dataset

We utilized two datasets in our study: the publicly available StateFarm dataset [33] and the non-public Lilong Distracted Driving Behavior (LDDB) dataset [34]. We selected the StateFarm dataset to assess the practical effectiveness of MDAM and various other lightweight architectures as a detector

feature extraction backbone network. This dataset, as illustrated in Figure 8, consists of a farm driver distraction detection dataset comprising ten categories. It contains 22,424 training images and 67,272 images enhanced using offline data augmentation techniques such as Gaussian blur, Gaussian noise and CutMix. On the other hand, the LDDB dataset comprises 14,808 videos captured by infrared cameras that record six driving behaviors of 2468 participants. Manual annotations were provided for these videos at a frame rate of five frames per second, resulting in a total of 287,804 images. To validate the module's efficacy, we compared it with mainstream object detection methods using the LDDB dataset.

Figure 8. Examples of the StateFarm's dataset.

3.3. Comparison experiments

To establish the superiority of our proposed method for detecting driver distraction, we compared it with several other detection methods using the LDDB dataset. These methods included YOLOv5s, as well as some lightweight networks (MobileNet, GhostNet and FasterNet) used as the backbone for YOLO. Additionally, we evaluated our proposed MTNet. The evaluation results for these different detectors are presented in Table 1. The best performance values are highlighted in bold.

Figure 9 displays the corresponding results for a more accurate comparison. Figure 9(a),(b) depict the performance of the various detectors in terms of accuracy and model weight. In each instance, the best performing method is indicated with an asterisk on the respective bar. Our proposed MTNet demonstrates exceptional performance in the detection of tiny objects on the LDDB dataset, achieving an average accuracy of 36.9%. Additionally, the computational complexity of the model surpasses that of other methods, boasting a level of 6.5 GFLOPs.

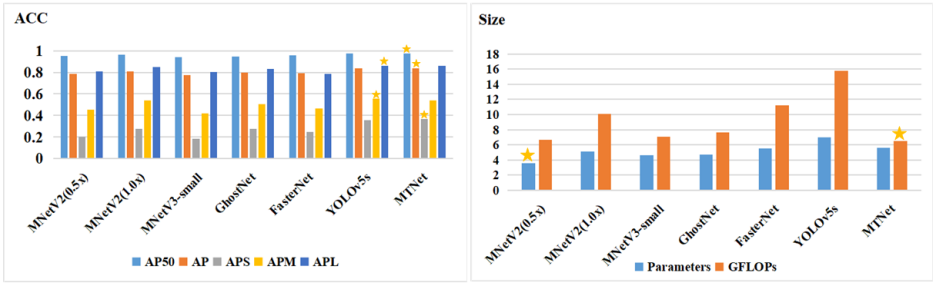
The detection performance comparison on the LDDB dataset is visualized in Figure 10. Other methods exhibit false detections and misses in detecting small objects like cigarette butts. In terms of small object detection efficiency, MTNet outperforms the other detection methods.

The proposed method demonstrated superior performance compared to other methods, as indicated by the results presented in the table and figure. The proposed method achieves higher accuracy than YOLOv5s on the LDDB benchmark, with values of 97.6%AP, 83.7%AP50 and 36.9%APS. Moreover, the proposed method reduces the number of model parameters by 1.38M and computation by 9.3 GFLOPs. The detection method exhibits a superior balance between speed and accuracy compared to other lightweight networks used as feature extraction backbone networks. The method outperforms in

all categories of distracted driving behavior, with notable improvements in small object detection and optimized model computation.

Table 2 presents the performance of our proposed multidimensional adaptive module (MDAM) in comparison to other lightweight classification networks on StateFarm's dataset. The results indicate that our proposed MDAM exhibits superior performance in terms of classification accuracy and computational complexity, simultaneously maintaining a smaller number of model parameters.

The corresponding metrics for accuracy, number of model parameters and computational complexity are depicted in Figure 11. The visualization of classification results and accuracy for certain categories is displayed in Figure 12. It is evident that alternative methods encounter challenges with misclassification or struggling to differentiate the correct category on the StateFarm dataset, as indicated by the highlighted red areas. In contrast, our proposed MDAM block achieves accurate classification in multiple categories.



(a) (b)

Figure 9. Comparison of the performance of different detectors on the LDDB dataset. The best performing methods are marked with an asterisk.

Table 1. Performance data for different detectors on the LDDB dataset.

Models	AP50	AP	APS	APM	APL	Parameters	GFLOPs
MNetV2(0.5x) [16]	0.951	0.784	0.202	0.453	0.807	**3.58**	6.7
MNetV2(1.0x) [16]	0.966	0.811	0.274	0.537	0.851	5.13	10.1
MNetV3-small [17]	0.942	0.775	0.181	0.418	0.805	4.60	7.1
GhostNet [25]	0.948	0.795	0.274	0.506	0.831	4.75	7.6
FasterNet [30]	0.956	0.789	0.248	0.465	0.784	5.55	11.2
YOLOv5s [5]	0.974	0.835	0.354	**0.556**	**0.86**	7.02	15.8
MTNet	**0.976**	**0.837**	**0.369**	0.538	0.859	5.64	**6.5**

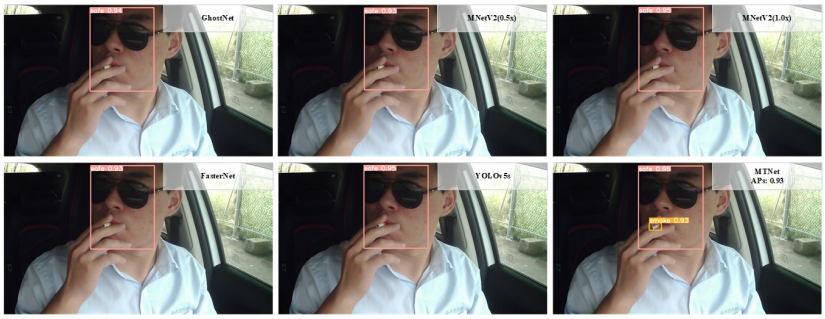


Figure 10. Visualization of the performance of different detectors on the LDDB dataset.

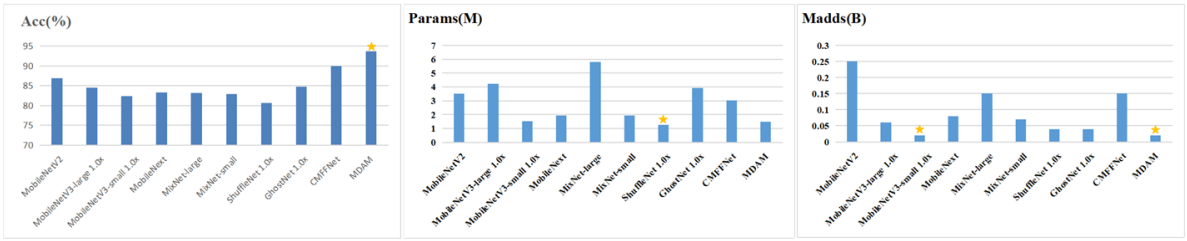


Figure 11. Performance comparison of different classification methods on the StateFarm dataset. The best performing method is marked with an asterisk.

Table 2. MDAM module performance comparison on Statefarm's benchmark.

Network	Acc(%)	Params(M)	Madds(B)
MobileNetV2 [16]	86.90	3.50	0.25
MobileNetV3-large 1.0x [17]	84.44	4.21	0.06
MobileNetV3-small 1.0x [17]	82.32	1.53	0.02
MobileNext [37]	83.23	1.94	0.08
MixNet-large [36]	83.20	5.81	0.15
MixNet-small [36]	82.88	1.94	0.07
ShuffleNet 1.0x [18]	80.64	**1.26**	0.04
GhostNet 1.0x [25]	84.77	3.91	0.04
CMFFNet [35]	89.95	3.03	0.15
MDAM	**93.70**	1.48	**0.02**

Figure 12. Performance visualization of different classification methods on the StateFarm dataset.

3.4. Ablation study

To further validate the effectiveness of each component in our method's modules, such as the multidimensional adaptive feature extraction block, lightweight feature fusion block and IoU-NWD weighted loss function, we conducted an ablation study in this subsection. Here, we aimed to assess the impact of these components on the overall performance.

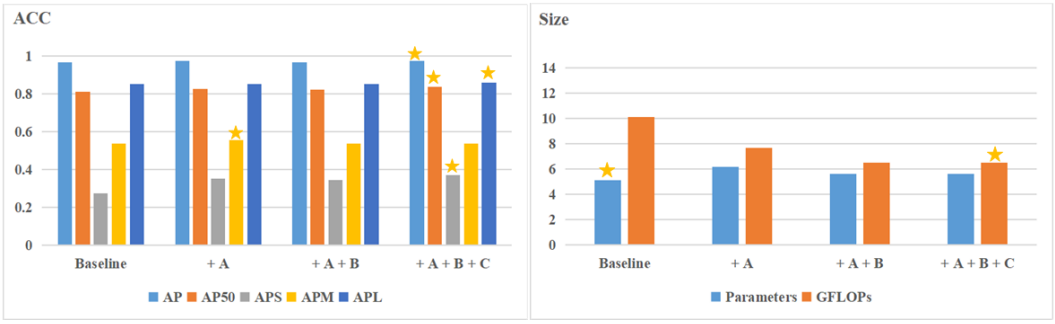
MobileNetV2 serves as the backbone network model for our baseline model. We incrementally add different components to assess their effectiveness and primarily compare the performance of the following four detection models:

- **Baseline:** We simply replace YOLOv5's feature extraction backbone network with lightweight MobileNetV2. This is the baseline model.
- **Baseline+A(MAFEB):** We incorporate the multi-dimensional adaptive feature extraction block into the backbone network of the detector architecture used for driver distraction behavior detection.
- **Baseline+A(MAFEB)+B(LFFB):** The lightweight feature fusion block is further enhanced based on the aforementioned model in order to reduce the model's overall weight.
- **Completed Model:** Our proposed comprehensive detector model.

Table 3 presents the utilization of Baseline and Baseline + A to showcase the efficacy of incorporating multidimensional adaptive feature extraction blocks into the model. Comparing Baseline + A with Baseline + A + B confirms the superior balance between speed and accuracy in the context of lightweight feature fusion blocks. Contrasting Baseline + A + B with the completed model demonstrates the benefits of using the IoU-NWD weighted loss function for detecting tiny targets.

Table 3 presents the detection performance of the various models. From the table, it is evident that each of the aforementioned components contributes to the improvement of the detection results. Notably, the inclusion of multi-dimensional adaptive feature extraction blocks and the utilization of IoU-NWD weighted loss function prove to be more effective in enhancing the detection accuracy of small targets. The performance comparison graph is depicted in Figure 13. Additionally, Figure 14

provides a visual representation of the impact of different models on the LDDB benchmark in our ablation study.



(a) (b)

Figure 13. Comparison of the performance of different detectors on the LDDB dataset in the ablation experiment. The best performing method is marked with an asterisk.

Table 3. Ablation study.

Models	AP	AP50	APS	APM	APL	Parameters	GFLOPs
Baseline	0.966	0.811	0.274	0.537	0.851	**5.13**	10.1
+A	0.973	0.828	0.354	**0.557**	0.851	6.2	7.7
+A+B	0.967	0.821	0.345	0.537	0.851	5.64	6.5
+A+B+C	**0.976**	**0.837**	**0.369**	0.538	**0.859**	5.64	**6.5**

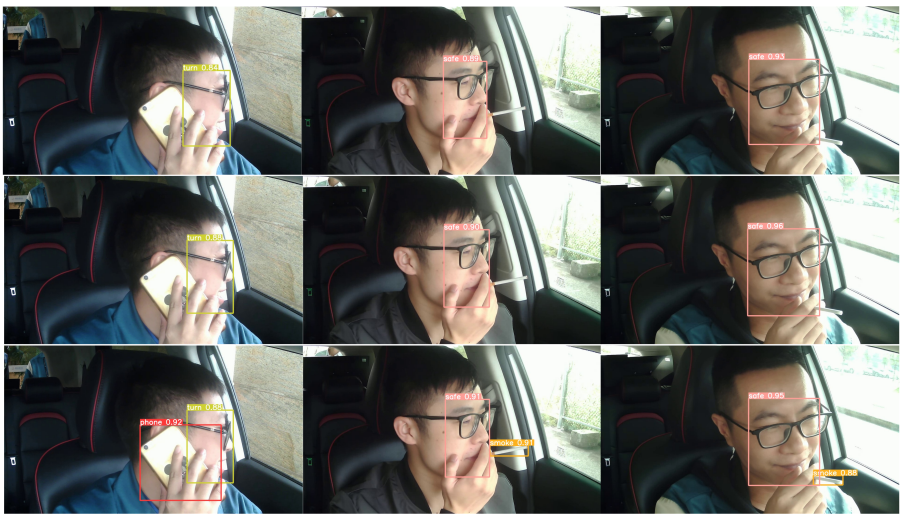


Figure 14. Performance visualization of different detectors in ablation experiments on the LDDB dataset.

4. Conclusions

In this paper, we propose a novel method for detecting driver distractions using deep learning. We combine a lightweight network architecture with optimization techniques for detecting tiny objects. To demonstrate the effectiveness of our approach, we have designed three functional blocks. First, we propose a feature extraction block that is based on an improved lightweight network. This block injects four adaptive attentions along the kernel space into the network using a parallel strategy. This enables the extraction of features with multidimensional information and improves efficiency through operator fusion operations. Second, we have designed a feature fusion block based on lightweight operator operations. The purpose of this block is to reduce computational complexity and memory access. Finally, we enhance the efficiency of detecting tiny targets by designing a weighted loss function based on the NWD metric. Our experimental results demonstrate the effectiveness of the key components in our method. Additionally, our proposed method outperforms some state-of-the-art methods on LDDB and StateFarm benchmarks. For future work, we plan to explore the feasibility of other lightweight optimizations.

References

1. A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, *Adv. Neural Inform. Process. Syst.*, **6** (2017), 84–90. https://doi.org/10.1145/3065386
2. K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, (2016), 770–778. https://doi.org/10.1109/CVPR.2016.90
3. J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, preprint, arXiv:1506.02640.
4. J. Redmon, A. Farhadi, Yolov3: An incremental improvement, preprint, arXiv:1804.02767.
5. Ultralytics, Yolov5, 2021. Available from: https://github.com/ultralytics/yolov5.
6. A. Misra, S. Samuel, S. Cao, K. Shariatmadari, Detection of driver cognitive distraction using machine learning methods, *IEEE Access*, **11** (2023), 18000–18012. https://doi.org/10.1109/ACCESS.2023.3245122
7. S. M. Iranmanesh, H. N. Mahjoub, H. Kazemi, Y. P. Fallah, An adaptive forward collision warning framework design based on driver distraction, *IEEE Trans. Intell. Trans. Syst.*, **19** (2018), 3925–3934. https://doi.org/10.1109/TITS.2018.2791437
8. A. Jamsheed V., B. Janet, U. S. Reddy, Real time detection of driver distraction using cnn, in *2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*, (2020), 185–191. https://doi.org/10.1109/ICSSIT48917.2020.9214233
9. C. Huang, X. Wang, J. Cao, S. Wang, Y. Zhang, Hcf: A hybrid cnn framework for behavior detection of distracted drivers, *IEEE access*, **8** (2020), 109335–109349. https://doi.org/10.1109/ACCESS.2020.3001159
10. C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, (2016), 2818–2826. https://doi.org/10.1109/CVPR.2016.308

11. F. Chollet, Xception: Deep learning with depthwise separable convolutions, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, (2017), 1251–1258. https://doi.org/10.1109/CVPR.2017.195
12. F. Sajid, A. R. Javed, A. Basharat, N. Kryvinska, A. Afzal, M. Rizwan, An efficient deep learning framework for distracted driver detection, *IEEE Access*, **9** (2021), 169270–169280. https://doi.org/10.1109/ACCESS.2021.3138137
13. D. L. Nguyen, M. D. Putro, K. H. Jo, Driver behaviors recognizer based on light-weight convolutional neural network architecture and attention mechanism, *IEEE Access*, **10** (2022), 71019–71029. https://doi.org/10.1109/ACCESS.2022.3187185
14. F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, K. Keutzer, Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size, preprint, arXiv:1602.07360.
15. A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, et al., Mobilenets: Efficient convolutional neural networks for mobile vision applications, preprint, arXiv:1704.04861.
16. M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L. C. Chen, Mobilenetv2: Inverted residuals and linear bottlenecks, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, (2018), 4510–4520. https://doi.org/10.1109/CVPR.2018.00474
17. A. Howard, M. Sandler, G. Chu, L. C. Chen, B. Chen, M. Tan, et al., Searching for mobilenetv3, in *Proceedings of the IEEE/CVF international conference on computer vision*, (2019), 1314–1324. https://doi.org/10.1109/ICCV.2019.00140
18. X. Zhang, X. Zhou, M. Lin, J. Sun, Shufflenet: An extremely efficient convolutional neural network for mobile devices, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, (2018), 6848–6856. https://doi.org/10.1109/CVPR.2018.00716
19. N. Ma, X. Zhang, H. T. Zheng, J. Sun, Shufflenet v2: Practical guidelines for efficient cnn architecture design, in *Proceedings of the European conference on computer vision (ECCV)*, (2018), 116–131. https://doi.org/10.1007/978-3-030-01264-9
20. C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, et al., Going deeper with convolutions, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, (2015), 1–9. https://doi.org/10.1109/CVPR.2015.7298594
21. S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, in *International conference on machine learning*, (2015), 448–456.
22. C. Szegedy, S. Ioffe, V. Vanhoucke, A. Alemi, Inception-v4, inception-resnet and the impact of residual connections on learning, preprint, arXiv:1602.07261.
23. M. Tan, Q. Le, Efficientnet: Rethinking model scaling for convolutional neural networks, in *International conference on machine learning*, (2019), 6105–6114.
24. M. Tan, Q. Le, Efficientnetv2: Smaller models and faster training, in *International conference on machine learning*, (2021), 10096–10106.
25. K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, C. Xu, Ghostnet: More features from cheap operations, in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, (2020), 1580–1589. https://doi.org/10.1109/CVPR42600.2020.00165

26. J. He, S. Erfani, X. Ma, J. Bailey, Y. Chi, X. Hua, Alpha-iou: A family of power intersection over union losses for bounding box regression, preprint, arXiv:2110.13675.
27. C. Deng, M. Wang, L. Liu, Y. Liu, Y. Jiang, Extended feature pyramid network for small object detection, *IEEE Trans. Multimedia*, **24** (2021), 1968–1979. https://doi.org/10.1109/TMM.2021.3074273
28. X. Yang, J. Yang, J. Yan, Y. Zhang, T. Zhang, Z. Guo, et al., Scrdet: Towards more robust detection for small, cluttered and rotated objects, in *Proceedings of the IEEE/CVF international conference on computer vision*, (2019), 8232–8241. https://doi.org/10.1109/ICCV.2019.00832
29. H. Li, J. Li, H. Wei, Z. Liu, Z. Zhan, Q. Ren, Slim-neck by gsconv: A better design paradigm of detector architectures for autonomous vehicles, preprint, arXiv:2206.02424.
30. J. Chen, S. h. Kao, H. He, W. Zhuo, S. Wen, C. H. Lee, et al., Run, don't walk: Chasing higher flops for faster neural networks, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (2023), 12021–12031. https://doi.org/10.1109/CVPR52729.2023.01157
31. V. M. Panaretos, Y. Zemel, Statistical aspects of wasserstein distances, *Annual Rev. Stat. Appl.*, **6** (2019), 405–431. https://doi.org/10.1146/annurev-statistics-030718-104938
32. J. Wang, C. Xu, W. Yang, L. Yu, A normalized gaussian wasserstein distance for tiny object detection, preprint, arXiv:2110.13389.
33. S. Farm, *State farm distracted driver detection*, Technical report, 2016. Available from:: https://www. kaggle. com//state-farm-distracted-driver-detection.
34. Z. Zhu, D. Liang, S. Zhang, X. Huang, B. Li, S. Hu, Traffic-sign detection and classification in the wild, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, (2016), 2110–2118. https://doi.org/10.1109/CVPR.2016.232
35. Y. Li, P. Xu, Z. Zhu, X. Huang, G. Qi, Real-time driver distraction detection using lightweight convolution neural network with cheap multi-scale features fusion block, in *Proceedings of 2021 Chinese Intelligent Systems Conference: Volume II*, Springer, (2022), 232–240.
36. M. Tan, Q. V. Le, Mixconv: Mixed depthwise convolutional kernels, preprint, arXiv:1907.09595.
37. A. Howard, C. Zhu, J. Chen, X. Wang, W. Wu, Y. He, et al., Mobilenext: Rethinking bottleneck structure for efficient mobile network design, preprint, arXiv:2003.10888.
38. Z. Zhu, Z. Yao, G. Qi, N. Mazur, P. Yang, B. Cong, Associative learning mechanism for drug-target interaction prediction, *CAAI Trans. Intell. Technol.*, (2023). https://doi.org/10.1049/cit2.12194
39. Z. Zhu, X. He, G. Qi, Y. Li, B. Cong, Y. Liu, Brain tumor segmentation based on the fusion of deep semantics and edge information in multimodal mri, *Inform. Fusion*, **91** (2023), 376–387. https://doi.org/10.1016/j.inffus.2022.10.022

AIMS Press