*electronics*

*Article*

# Research on Lightweight-Based Algorithm for Detecting Distracted Driving Behaviour

**Chengcheng Lou [1] and Xin Nie [1,*]**

School of Computer Science and Engineering, Wuhan Institute of Technology, Wuhan 430079, China; 22107010076@stu.wit.edu.cn
* Correspondence: niexin@whu.edu.cn

**Abstract:** In order to solve the existing distracted driving behaviour detection algorithms' problems such as low recognition accuracy, high leakage rate, high false recognition rate, poor real-time performance, etc., and to achieve high-precision real-time detection of common distracted driving behaviours (mobile phone use, smoking, drinking), this paper proposes a driver distracted driving behaviour recognition algorithm based on YOLOv5. Firstly, to address the problem of poor real-time identification, the computational and parametric quantities of the network are reduced by introducing a lightweight network, Ghostnet. Secondly, the use of GSConv reduces the complexity of the algorithm and ensures that there is a balance between the recognition speed and accuracy of the algorithm. Then, for the problem of missed and misidentified cigarettes during the detection process, the Soft-NMS algorithm is used to reduce the problems of missed and false detection of cigarettes without changing the computational complexity. Finally, in order to better detect the target of interest, the CBAM is utilised to enhance the algorithm's attention to the target of interest. The experiments show that on the homemade distracted driving behaviour dataset, the improved YOLOv5 model improves the mAP@0.5 of the YOLOv5s by 1.5 percentage points, while the computational volume is reduced by 7.6 GFLOPs, which improves the accuracy of distracted driving behaviour recognition and ensures the real-time performance of the detection speed.

**Keywords:** distracted driving behaviour; YOLOv5; Ghostnet; GSConv; soft-NMS; CBAMs



**Citation:** Lou, C.; Nie, X. Research on Lightweight-Based Algorithm for Detecting Distracted Driving Behaviour. *Electronics* **2023**, *12*, 4640. https://doi.org/10.3390/electronics12224640

Academic Editors: Yu-Chen Hu and Manohar Das

Received: 20 September 2023
Revised: 20 October 2023
Accepted: 8 November 2023
Published: 14 November 2023



## 1. Introduction

With the boom in the vehicle industry and the world economy, the number of vehicles is increasing and a wide variety of traffic accidents are occurring. Around one-fifth of these crashes are caused by distracted driver behaviour, such as smoking, drinking and using mobile phones while driving. Therefore, how to quickly and accurately detect distracted driving behaviour has become the focus of traffic safety issues.

Currently, target recognition algorithms are mainly classified into two categories. For example, Mask-RCNN [1], Faster-RCNN [2], Fast R-CNN [3], R-CNN [4] and so on are two-stage recognition algorithms based on candidate regions; the other is a single-stage algorithm based on edge regression, such as SSD [5–8], YOLO [9–12] and so on. Various detection methods exist for distracted driving behaviours that occur in drivers. Nilufar V et al. [13] derived a novel set of features for identifying distracted behaviour of drivers from physiological and thermal signals of the human body. Luo G et al. [14] proposed a distracted driving behaviour detection method based on transfer learning and model fusion. Peng P et al. [15] proposed a method using a causal And–or graph called C-AOG and a spatio-temporal bilinear DL network called TSD-DLN for driver distracted driving behaviour recognition. Yingcheng L et al. [16] proposed a lightweight attention module called IRAM to simulate human attention to extract more specific features of driver distraction behaviour. Xia Zhao et al. [17] proposed a ViT model based on deep convolution and token dimensionality reduction for real-time driver distraction behaviour

detection. Libo Cao et al. [18] used NanoDet, a lightweight target detection model, for distracted driving behaviour detection in response to low detection accuracy as well as poor real-time performance. Bin Zhang et al. [19] proposed a class-spacing-optimization-based model training method for driver distraction detection, which improves the model's classification performance for distracted driving behaviour categories through end-to-end model training. Tianliu Feng and others used the recognition judgement of human facial triangular features to derive whether a driver is exhibiting distracted driving behaviour or not [20]. Ding Chen et al. [21] proposed a framework for identifying driver distraction behaviour based on an ensemble model. Mingqi Lu and others gave a pose-guided model to identify driving behaviours in a single image using keypoint action features [22]. Omid Dehzangi et al. [23] proposed a minimally invasive, wearable physiological sensor that can be used on smartwatches to quantify skin conductance (SC), called galvanic skin response (GSR), to characterise and identify distractions during natural driving. Furkan Omerustaoglu and others proposed the integration of sensor data into a vision-based distracted driving recognition model, which improves the generalisation of the system [24]. Lei Zhao et al. [25], through the use of an adaptive spatial attention mechanism, proposed a novel driver behaviour detection system. Md. Uzzol Hossain et al. [26] propose a CNN-based model to detect distracted drivers and determine the reason for the distraction. In order to detect drivers' distracting behaviours, such as making phone calls, eating, texting, etc., Yuxin Zhang and others proposed a deep unsupervised multimodal fusion network called UMMFN [27]. Hiteshwar Singh et al. [28] used a data-centric approach with enhancements to support vector machines that resulted in significant improvements. Abeer. A. Aljohani combined artificial deep learning and machine learning models with genetic algorithms to detect driver behaviour [29]. Weichu Xiao and others proposed an attention-based deep neural network approach called ADNet for driver behaviour recognition [30]. Mingqi Lu and others identified driving behaviour by detecting specific parts of the action [31]. A dilated bald R-CNN method called DL-RCNN was proposed. Cammarata, A. et al. [32] proposed a novel approach to derive rigid and interpolation multipoint constraints by exploiting the role of the reference conditions used in the Floating Frame of Reference Formulation to cancel rigid body modes from the elastic field.

Although the above methods achieve the detection of driver distraction behaviour from various directions and angles, many of them are limited. Some methods start from the direction of physiological monitoring, and the results may be very good, but the implementation of the driver's physiological adaptation needs to be considered, and these methods use sensors and other items that are likely to cause interference to the driver in the driving process. There are also methods that are very innovative but are not suitable for universal use, and the engineering results would not be ideal. This paper proposes a lightweight YOLOv5-based target recognition algorithm to detect common distracted driving behaviours, which is suitable for universal use, does not cause interference to the driver during the detection process, and ensures improved accuracy of the detected targets while dramatically increasing the detection speed.

## 2. YOLOv5 Target Detection Algorithm

YOLOv5 is a more advanced and popular target detection algorithm, which belongs to a class of YOLO series algorithms. It improves the robustness of the network by using a series of convolutional kernels and feature fusion modules in the feature extraction network and realises an organic combination of data and features in the neck network. YOLOv5s is the smallest and fastest variant of YOLOv5, and in view of the high requirements of distracted driving detection on detection speed, the YOLOv5s network is adopted as the base model. The structure of YOLOv5s is shown in Figure 1.

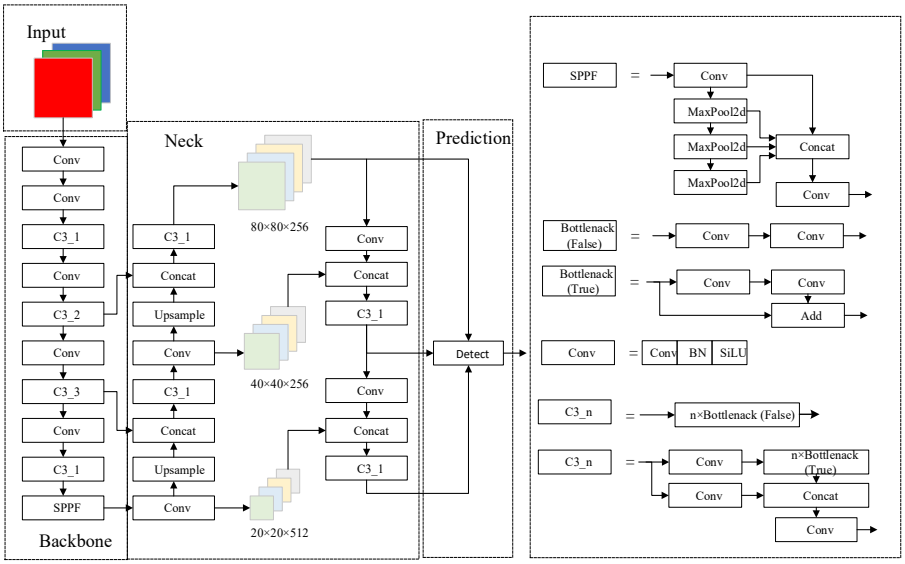


**Figure 1.** Structure of YOLOv5s.

YOLOv5s contains four important parts: prediction, neck, backbone and input. The input performs image scaling, adaptive anchor frame computation and mosaic data enhancement to enrich the data volume of the dataset and enhance the performance of the neural network. The backbone focus and CSP (Cross Stage Partial) structures are used to reduce the feature map and extract the corresponding image features. The neck uses an FPN (Feature Pyramid Network) + PAN (Perceptual Adversarial Network) structure to fuse the features extracted by input. The prediction detecting head uses different scales corresponding to the a priori frames to make predictions and classifications.

## 3. The Algorithm in This Paper

### *3.1. Overall Network Structure*

In order to solve the current distracted driving behaviour detection problems, such as slow recognition speed and low recognition accuracy, and to enhance the recognition performance of the model under the influence of various types of complex environments, based on YOLOv5s, a lightweight model is proposed. The improved model discards the original backbone network and part of the C3 network structure, utilises a combination of Ghostnet and GSConv to realise the lightweight of the model, which greatly reduces the computational and parametric quantities of the model, and enhances the recognition speed of the model; meanwhile, the incorporation of SoftNMS and the CBAMs ensures an improvement in the model's accuracy. The improved model network structure is shown in Table 1.

**Table 1.** Improved network structure of the model.

| Layers | From | Parameters | Module |
|---|---|---|---|
| 0 | −1 | 224 | GSConv |
| 1 | −1 | 476 | Gbneck |
| 2 | −1 | 2668 | Gbneck |
| 3 | −1 | 3120 | Gbneck |
| 4 | −1 | 7482 | Gbneck |
| 5 | −1 | 12,318 | Gbneck |
| 6 | −1 | 15,684 | Gbneck |
| 7 | −1 | 10,608 | Gbneck |
| 8 | −1 | 149,404 | Gbneck |
| 9 | −1 | 150,452 | Gbneck |
| 10 | −1 | 295,880 | Gbneck |

**Table 1.** *Cont.*

| Layers | From | Parameters | Module |
|---|---|---|---|
| 11 | −1 | 91,880 | Gbneck |
| 12 | −1 | 553,880 | Gbneck |
| 13 | −1 | 91,880 | Gbneck |
| 14 | −1 | 553,880 | Gbneck |
| 15 | −1 | 854,978 | C3CBAM |
| 16 | −1 | 122,160 | GSConv |
| 17 | −1 | 65,152 | GSConv |
| 18 | −1 | 0 | Upsample |
| 19 | [−1,9] | 0 | Concat |
| 20 | −1 | 310,784 | C3 |
| 21 | −1 | 18,240 | GSConv |
| 22 | −1 | 0 | Upsample |
| 23 | [−1,5] | 0 | Concat |
| 24 | −1 | 77,568 | C3 |
| 25 | −1 | 75,584 | GSConv |
| 26 | [−1,21] | 0 | Concat |
| 27 | −1 | 296,448 | C3 |
| 28 | −1 | 298,624 | GSConv |
| 29 | [−1,16] | 0 | Concat |
| 30 | −1 | 1,297,408 | C3 |
| 31 | [24,27,30] | 21,576 | Detect |

### *3.2. Lightweight Model Ghostnet*

The Ghostnet lightweight convolutional network focuses its attention on the large number of redundant feature pictures generated after convolution compared to other lightweight networks. In a well-trained deep neural network, a large or even redundant number of feature images are usually included to ensure a comprehensive understanding of the input data. While Ghostnet generates more feature maps using fewer parameters to ensure the richness of the feature maps, Ghostnet builds the Ghost module, which can be utilised to generate a large number of redundant pictures faster with less arithmetic requirements.

The process of extracting target features using the Ghost module is shown in Figures 2 and 3, which are divided into two parts. The first part does the standard convolution operation, using the given size of the convolution kernel to operate on the input image to obtain the feature maps of each channel of the detection target; the second part linearly transforms the feature maps generated in the first part to obtain similar feature maps, which produces a high-dimensional convolution effect and reduces the computational and parametric quantities of the model; finally, the two parts are spliced together to obtain the complete output layer.

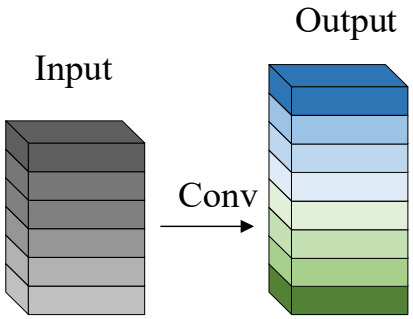


**Figure 2.** Standard convolution.

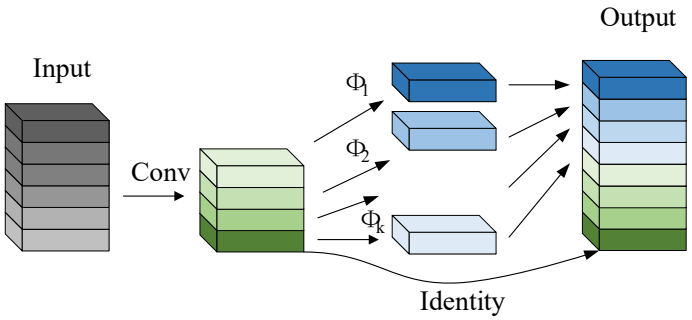


**Figure 3.** Ghost module convolution.

For standard convolution, given the input data $X \in R^{h \times w \times c}$, where $w$ and $h$ are the width and height of the input feature map, respectively, and $c$ is the amount of channels of the input, the resulting operation for generating an arbitrary convolutional layer of $n$ feature maps is shown in Equation (1).

$$Y = X * f + b \tag{1}$$

where $b$ is the bias term, $Y \in R^{(h' \times w' \times n)}$ is the output feature map with $n$ output channels and $f \in R^{c \times k \times k \times n}$ is the convolution kernel of this feature layer. In this convolution process, the number of channels, $c$, and the number of convolution kernels, $n$, can be very large, which makes the computational $FLOPs$ increase. The formula for $FLOPs$ is shown in Equation (2).

$$FLOPs = n \times h' \times w' \times c \times k \times k \tag{2}$$

where $k$ is the size of the convolution kernel and $w'$ and $h'$ are the width and height of the output feature map, respectively.

The Ghost module performs a convolution operation on a portion of the existing feature maps, which is performed using one standard convolution for $m$ original output feature maps, $Y' \in R^{h' \times w' \times m}$, where $m \leq n$. The procedure is shown in Equation (3).

$$Y' = X * f' \tag{3}$$

where $f' \in R^{c \times k \times k \times m}$ is the convolution kernel used in this feature layer, omitting the presence of the bias term.

In order to further obtain the required $n$ feature maps, a series of simple linear operations are used on the obtained $m$-dimensional feature maps to generate $s$ similar feature maps, which is shown in Equation (4).

$$y_{ij} = \varnothing_{ij}(y'_i) \ \forall \ i = 1, \ldots, \ m, \ j = 1, \ldots, s \tag{4}$$

where $y'_i$ is the ith original feature map in $Y'$. $\varnothing_{ij}$ is the $j$th linear computation for generating the $j$th similar feature map, $y_{ij}$. This part of the computational $FLOPs$ is shown in Equation (5).

$$FLOPs = \frac{n}{s} \times h' \times w' \times c \times k \times k + (s-1) \times \frac{n}{s} \times h' \times w' \times d \times d \tag{5}$$

where $d$ is the average kernel size for each linear operation. The theoretical speedup after using the Ghost module is derived from Equations (2) and (5), as shown in Equation (6).

$$r_s = \frac{n \times h' \times w' \times c \times k \times k}{\frac{n}{s} \times h' \times w' \times c \times k \times k + (s-1) \times \frac{n}{s} \times h' \times w' \times d \times d} = \frac{c \times k \times k}{c \times k \times k \times \frac{1}{s} + \frac{s-1}{s} \times d \times d} \approx \frac{s \times c}{s + c - 1} \approx s \tag{6}$$

Since the magnitude of $d \times d$ is similar to $k \times k$ and $s \ll c$, similarly, the parameter compression ratio $r_c$ is approximately equal to $r_s$, resulting in $s$. By calculation, the overall compression ratio of the model is approximately $s$ times.

### *3.3. Lightweight Convolution Method GSConv*

The lightweight convolution GSConv is a new convolution method based on deeply separable convolution (DSC), standard convolution (SC) and channel blending operations; the idea is to separate the input channels into multiple groups, perform an independent depth-wise separable convolution for each group and then recombine them by means of channel blending, the structure of which is shown in Figure 4.

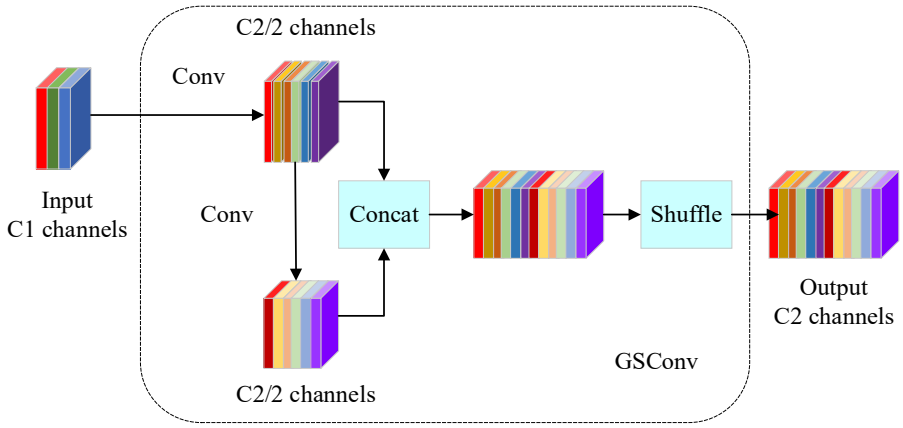


**Figure 4.** Structure of GSConv.

When the convolution size $f \times g$, $x \times y$ is the size of the output, $M$ is the number of input channels and $N$ is the number of channels of the output, then the standard convolution, $SC$, produces the computation shown in Equation (7).

$$SC = f \times g \times x \times g \times M \times N \tag{7}$$

The computational effort generated by the depth-wise separable convolutional, $DSC$, is shown in Equation (8).

$$DSC = f \times g \times x \times y \times M + x \times y \times M \times N \tag{8}$$

This results in a compression ratio between the $DSC$ and $SC$, as shown in Equation (9).

$$\frac{DSC}{SC} = \frac{f \times g \times x \times y \times M + x \times y \times M \times N}{f \times g \times x \times g \times M \times N} = \frac{1}{N} + \frac{1}{f \times g} \tag{9}$$

In the GSConv lightweight convolution, each convolution kernel is divided into two parts: the Ghost Convolution Kernel and the Main Convolution Kernel. The Main Convolution Kernel acts as the core bit and performs the main computation, while the Ghost Convolution Kernel is used to complement the computation of the Main Convolution Kernel, thus achieving the effect of grouped convolution and greatly reducing the amount of computation. Its calculation formula is shown in Equation (10).

$$(f * g)(n) = \sum_{i=1}^{n-1} f(i) g(n \oplus i \oplus r(i)) \tag{10}$$

where $\oplus$ denotes the dissimilarity operation, $n$ represents the length of the input and $r(i)$ denotes the $i$th element of the input, which in a convolution corresponds to the amount

of random shifts. The time complexity of GSConv is low due to the fact that it retains the channel-dense convolution, which is shown in Equation (11).

$$time_{GSConv} \sim O\left[x \cdot y \cdot f \cdot g \cdot \frac{C_2}{2}(C_1 + 1)\right] \tag{11}$$

The time complexity formulas for the standard convolution, *SC*, and the depth-wise separable convolution, *DSC*, are shown in Equations (12) and (13).

$$time_{SC} \sim O(x \cdot y \cdot f \cdot g \cdot C_1 \cdot C_2) \tag{12}$$

$$time_{DSC} \sim O(x \cdot y \cdot f \cdot g \cdot 1 \cdot C_2) \tag{13}$$

where $C_2$ is the number of channels in the output and $C_1$ is the number of channels in the convolution kernel. In this convolution operation, the Main Convolution Kernel and the Ghost Convolution Kernel are in different positions horizontally and vertically, and it adopts the strategy of channel mixing and shuffling, grouping the input channels and letting the channels in each group be cross-computed, which enhances the performance of the network and reduces the number of parameters in the model.

### *3.4. CBAM*

Incorporating the CBAM on the basis of YOLOv5 aims to improve the accuracy of the algorithm by means of feature weighting. The CBAM, as a comprehensive and powerful attention mechanism, takes advantage of its ability to fuse spatial and channel attention, focuses on the spatial and channel regions of some features, and then enriches the feature information by combining the MaxPool operation and the AvgPool operation. The model's structure is shown in Figure 5.

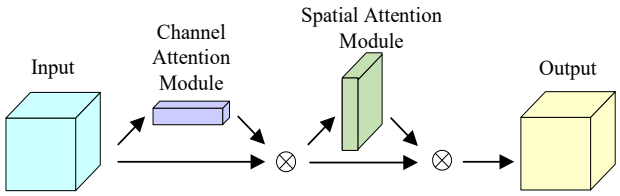


**Figure 5.** The structure of the CBAM.

In the CBAM, the input, firstly, passes the channel attention module, which performs an element-by-element multiplication operation between the output one-dimensional convolution of the channel attention and the input, so as to obtain the input enhanced by the channel attention, and then, passes the spatial attention module, which performs an element-by-element multiplication operation between the output two-dimensional convolution of the Spatial Attention and the input feature map of the Module again. The final output feature map of the CBAM is obtained.

The network structure of the channel attention module in the CBAM is shown in Figure 6.

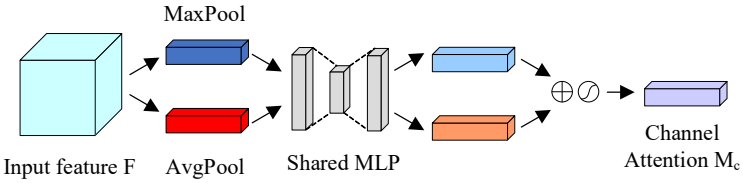


**Figure 6.** Channel attention module in the CBAM.

Firstly, the input feature map is passed through the MaxPool and AvgPool layers to aggregate the spatial information of the input feature maps to obtain the MaxPool and AvgPool features, respectively; then, it is passed through a shared network, which consists of a multilayer perceptron (MLP) and a hidden layer, respectively, and then the output features from the shared network are summed element-by-element, and finally a sigmoid activation operation is performed to produce a one-dimensional convolution of the channel attention. The computational procedure of the channel attention module is shown in Equations (14) and (15). $F$ denotes the initial input feature map, $M_c(F)$ denotes the generated channel attention one-dimensional convolution, $F'$ denotes the output feature map after the channel attention convolution, $MLP$ denotes multilayer perceptron, $\odot$ denotes the element-by-element multiplication operation and $\sigma$ denotes sigmoid activation function.

$$M_c(F) = \sigma[MLP(AvgPool(F)) + MLP(MaxPool(F))] \tag{14}$$

$$F' = M_c(F) \odot F \tag{15}$$

The structure of the Spatial Attention Module of the CBAM is shown in Figure 7.

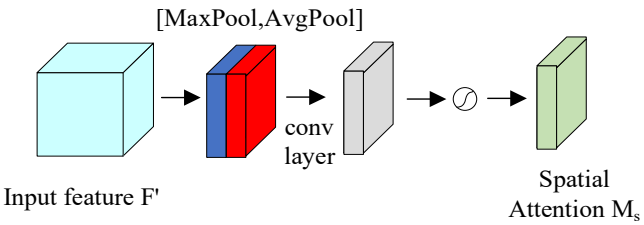


**Figure 7.** Spatial attention module in the CBAM.

The input to the spatial attention module is the feature map enhanced by the channel attention module. The feature map is first aggregated by MaxPool and AvgPool to aggregate the channel information of the feature map, thus generating two two-dimensional feature maps, which represent the MaxPool and AvgPool features of the feature map in the channel, respectively; then, the standard convolutional layer is used to perform the connectivity as well as convolutional operations; and finally, the spatial attention two-dimensional convolution is generated by the sigmoid activation function. The computational process of the Spatial Attention Module is shown in Equations (16) and (17). $F'$ represents the feature map obtained after processing by the channel attention module, $M_s(F')$ denotes the resulting two-dimensional convolution of the spatial attention, $F''$ denotes the resulting output feature map after the convolution of spatial attention, $\odot$ denotes the element-by-element multiplication operation, $\sigma$ denotes the sigmoid activation function and $f^{7\times7}$ represents the convolution operation with dimensions of $7 \times 7$.

$$M_s(F') = \sigma\left(f^{7\times7}([AvgPool(F')]; [MaxPool(F')])\right) \tag{16}$$

$$F'' = M_s(F') \odot F' \tag{17}$$

### *3.5. The Soft-NMS Algorithm*

The main idea of the NMS (non-maximum suppression) mechanism is to sort the anchor frame scores in the score set from highest to lowest, traverse the calculation of the proportion of overlap area between each anchor frame and the highest-scoring candidate frame, and if the calculation result is larger than the threshold, the anchor frame is directly removed. However, this method may lead to the detection failure of the target appearing in the overlapping area; in addition, if the threshold value is taken to be too small, it may cause the neighbouring targets to miss the detection phenomenon, and if the threshold value is too large, it may also lead to a target with multiple detection frames.

In order to avoid missing or re-detection in detecting the target object, the Soft-NMS algorithm is introduced; the core idea of this algorithm is that when the overlap area ratio of two anchor frames is greater than the threshold value, it will not directly return the score to zero but will reduce the score of the anchor frame, and the higher the overlap area ratio is, the more the score will be reduced. Compared with the traditional NMS algorithm, Soft-NMS only needs to make minor changes to the traditional NMS, does not add extra parameters, does not need extra training, is easy to implement, the complexity of the algorithm is the same as that of the traditional NMS, and is highly efficient to use. The flow of the Soft-NMS algorithm is shown in Algorithm 1.

**Algorithm 1:** Flow of Soft-NMS algorithm.

**Inputs:** Initial Anchor Frame Set $B$, Score set $S$, Overlap Ratio Threshold $\pi_0$
**Output:** Updated Anchor Box Set $B'$ and Score Set $S'$

$B \leftarrow \{b_i\}, S \leftarrow \{s_i\}, \pi(b_u, b_v) \leftarrow \pi_0$
$B', S' \leftarrow \varnothing$
while $B \neq \varnothing$ do
  $s^* \leftarrow max\{S_i\}$
  $b^* \leftarrow s^*$
  $B' \leftarrow B' \cup b^*$
  $S' \leftarrow S' \cup s^*$
  $B \leftarrow B - b^*$
  $S \leftarrow B$
  for $b_i \in B$

$$s_i \leftarrow \begin{cases} s_i & \pi(b^*, b_i) < \pi_0 \\ s_i(1 - \pi(b^*, b_i)) & \pi(b^*, b_i) \geq \pi_0 \end{cases}$$

  end
end

## 4. Experimental Results and Analysis

### *4.1. Dataset Production*

Due to the protection of the privacy of the driver's driving behaviour, the relevant datasets are almost never made public, and most of the similar datasets are self-made. At present, the most common public dataset is SFD3 (State Farm Distracted Driver Detection), but the data in this dataset have a single orientation, which cannot meet the needs of multi-directional detection of drivers, and it does not detect some special distracted driving behaviours, such as smoking in the process of driving. In this paper, based on the SFD3 dataset, some data captured in different directions and under different conditions are added to enrich the data in the dataset. Then, a labelling annotation tool was used to annotate the target objects one by one, and a total of 8222 image data points were produced. The dataset is randomly divided in the ratio of 7:3, in which there are about 5755 picture data points in the training set and 2467 picture data points in the validation set. The label distribution of the dataset is shown in Figure 8.

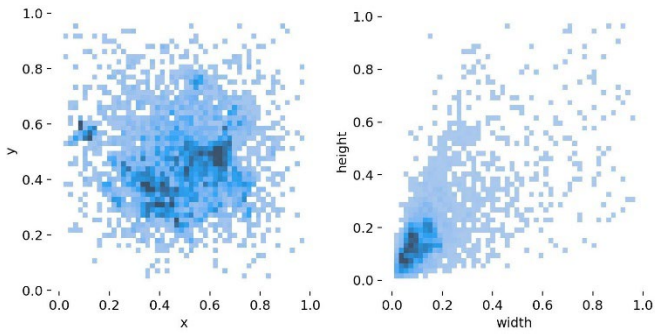


**Figure 8.** Distribution of labels in the dataset.

x and y denote the horizontal and vertical coordinates of the centre point of the label box, and the width and height denote the width and height of the label box, respectively. From the data in the figure, it can be seen that the labels are uniformly distributed and belong to multi-scale targets, which are suitable for the detection of distracted driving behaviour by drivers in real-life scenarios.

### *4.2. Configuration of Parameters and Evaluation of Indicators*

In this paper, YOLOv5s is chosen as the base algorithm and is improved based on it. In the experimental environment, NVIDIA RTX 3090 is used for the GPU to satisfy the arithmetic support. The training input image data size is 640 × 640, the batch size is 16, the data enhancement method adopts Mosaic data enhancement, the quantity of workers is set to 12, and the number of epochs is 200.

In this paper, *P*, *R*, GFLOPs, parameters and mAP@0.5 are used as the evaluation metrics of model performance. *P* is the ratio of correct predictions; *R* is the ratio of selected predictions. The formulas of *P* and *R* are shown in Equations (18) and (19).

$$P = \frac{TP}{TP + FP} \tag{18}$$

$$R = \frac{TP}{TP + FN} \tag{19}$$

where, when the IOU threshold is 0.5, *TP* denotes the quantity of detected frames with the IOU outweighing 0. 5, which is the number of correct detections; *FP* denotes the number of detected frames with the IOU less than or equal to 0.5, which is the quantity of misdetections; and *FN* represents the quantity of targets with missed detections. *mAP* is computed as shown in Equations (20) and (21).

$$AP = \int_0^1 p(R)dR \tag{20}$$

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{21}$$

*N* represents the quantity of categories of target objects, mAP@0.5 denotes the average *AP* when the IOU threshold is 0.5 and *mAP* is the area under the P–R curve. mAP@0.5 is often used to reflect the detection ability of the model, which can show the performance of the model well.

### *4.3. Experimental Results and Analysis*

In order to verify the validity of the relevant modules, a number of experiments were carried out on each of the modules to compare their effects on the performance of the original model under the same hardware conditions. The mAP@0.5 comparison between the final model and YOLOv5 is shown in Figure 9.

As can be seen from the figure, after a period of training, the improved model has a more obvious improvement in accuracy compared with YOLOv5. While the accuracy is improved, the model also has good performance for phenomena such as error checking and rechecking, as shown in Figures 10 and 11.

Since the improvement in the accuracy of the improved algorithm depends on the joint action of the modules, it is necessary to verify the role of each module in the model and to calibrate its effectiveness. In this paper, under the same experimental parameters and hardware environment conditions, five sets of ablation experimental results are derived on the homemade dataset, and the experimental results are shown in Tables 2 and 3.

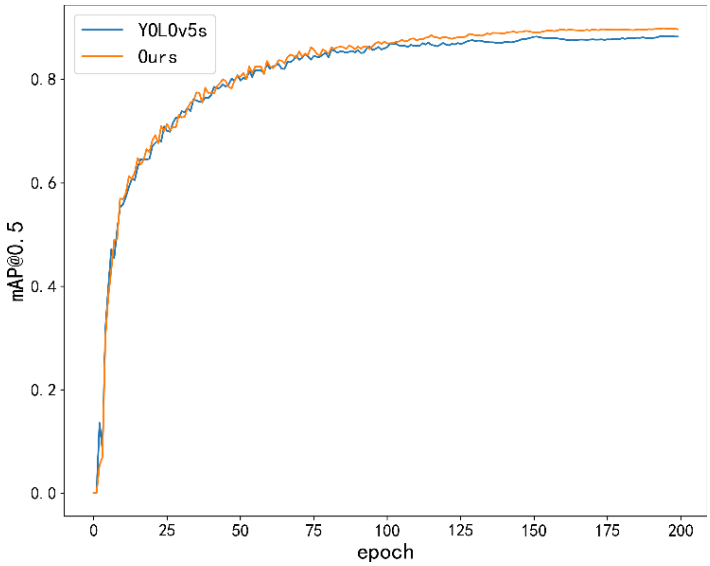


**Figure 9.** mAP@0.5 comparison between YOLOv5s and the algorithm in this paper.


**Figure 10.** Original model detection map.


**Figure 11.** The detection map of the improved model.

**Table 2.** Results of ablation experiments (1).

| Model | CBAM | Ghostnet | GSConv | Soft-NMS | P (%) | R (%) | mAP@0.5 (%) |
|---|---|---|---|---|---|---|---|
| YOLOv5s | × | × | × | × | 88. 8 | 83.7 | 88.3 |
| Module A | √ | × | × | × | 90.8 | 85.6 | 89.1 |
| Module B | × | √ | × | × | 91.6 | 83.3 | 87.3 |
| Module C | × | × | √ | × | 90.6 | 86.5 | 89.7 |
| Module D | × | × | × | √ | 92.1 | 84.1 | 89.7 |
| This paper | √ | √ | √ | √ | 91.4 | 83.8 | 89.8 |

**Table 3.** Results of ablation experiments (2).

| Model | CBAM | Ghostnet | GSConv | Soft-NMS | Params ($10^6$) | GFLOPs | Inference (ms) |
|---|---|---|---|---|---|---|---|
| YOLOv5s | × | × | × | × | 7.0 | 15.8 | 11.3 |
| Module A | √ | × | × | × | 7.0 | 15.8 | 9.9 |
| Module B | × | √ | × | × | 4.9 | 7.8 | 10.7 |
| Module C | × | × | √ | × | 6.6 | 15.2 | 9.0 |
| Module D | × | × | × | √ | 7.0 | 15.8 | 7.3 |
| This paper | √ | √ | √ | √ | 5.4 | 8.2 | 11.2 |

In the table, "√" indicates that the module is introduced and "×" indicates that the module is not introduced. As shown by the results of ablation experiments (a) and (b), Module A, the CBAM, improves the inference time by 1.4 ms, P and R by 2% and 1.9%, respectively, and mAP@0.5 by 0.8% while keeping the Params and GFLOPs almost the same, and the data show that the CBAM reduces the false detection rate to some extent. Module B, Ghostnet, has a relatively good effect on the model, compared with YOLOv5s. Although mAP@0.5 is reduced by 1%, all other data quantities are improved to different degrees, and the reduction in the Params, GFLOPs, and inference makes the model better able to meet the needs of industrialisation. Module C, GSConv, on the other hand, balances the coordination between each of the data, with a good improvement in mAP@0.5 while ensuring that the Params, GFLOPs and inference have been reduced, further meeting the industrialisation requirements. Module D, the SoftNMS algorithm, substantially improves the inference time while keeping the Params and GFLOPs almost the same, reducing the inference by 4 ms compared with YOLOv5s and improving the real-time requirements of the model. Compared with YOLOv5s, the improved model has different degrees of improvement in each data quantity. p and r are improved by 2.6% and 0.1%, respectively; mAP@0.5 is improved by 1.5%; the Params are reduced by 1.6 unit quantities; and the GFLOPs are reduced by 7.6. Clearly, the algorithmic model in this paper has superior performance enhancement in detecting and identifying the distracted driving behaviour of drivers compared with YOLOv5s.

To further validate the superiority of the algorithmic model in this paper, a series of data comparisons with other algorithms of the same type were performed on a home-made dataset. The results of the comparison experiments are shown in Table 4.

**Table 4.** Comparison of experimental results.

| Model | P (%) | R (%) | mAP@0.5 (%) | ModelSize (M) | Params ($10^6$) | GFLOPs | Inference (ms) |
|---|---|---|---|---|---|---|---|
| YOLOv5s | 88.8 | 83.7 | 88.3 | 14.2 | 7.0 | 15.8 | 11.3 |
| SSD | 87.4 | 72.7 | 80.2 | 96.7 | 24.2 | 274.2 | 85.0 |
| YOLOv5x | 90.5 | 86.0 | 89.2 | 165.0 | 86.2 | 203.6 | 29.9 |
| YOLOv7 | 90.1 | 87.4 | 89.1 | 72.1 | 37.2 | 105.0 | 24.8 |
| This paper | 91.4 | 83.8 | 89.8 | 19.8 | 5.4 | 8.2 | 11.2 |

Compared with YOLOv5s, SSD, YOLOv5x and YOLOv7, the SSD model is faster in operation. mAP@0.5 is able to outperform some of the two-phase detection algorithms, while the SSD model maintains fast operation. However, its Params and ModelSize are a non-negligible influencing factor. The mAP@0.5 improvement in YOLOv5x and YOLOv7 is more obvious, but the growth of its Params and GFLOPs deepens the complexity of the model, and the huge ModelSize of YOLOv5x has a big impact on engineering. The model in this paper has a significant advantage in terms of its Params and GFLOPs. mAP@0.5 has a nice boost and its Inference is not inferior at all. Although ModelSize has a slight increase, it hardly affects the engineering deployment. Therefore, compared with other models, this paper's model is more suitable for detecting drivers' distracted driving behaviour.

## 5. Conclusions

In this paper, based on the traffic safety problems related to driver distracted driving behaviours as the research background, we detect and identify the common distracted driving behaviours produced by drivers in the driving process and propose an improved target recognition algorithm based on the YOLOv5s. The purpose of the algorithm in this paper is to solve the problems of slow recognition, high computation and low recognition accuracy. YOLOv5s is used as the base model, and Ghostnet is used to lighten the model and reduce the computational and parametric quantities of the model. The GSConv module is used to balance accuracy and speed while lightweighting, so that its accuracy will not be drastically reduced. The Soft-NMS algorithm and the CBAM enable the model to have an increase in speed while the accuracy can be improved after lightweighting. The final improved model is compared with the YOLOv5s model. The Params are reduced by about $1.6 \times 10^6$, the GFLOPs are reduced by 7.6 GFLOPs, and mAP@0.5 is improved by 1.5%. In summary, the improved lightweight model provides superior performance in detecting and identifying distracted driving behaviour.

**Author Contributions:** Writing—original draft preparation, C.L.; writing—review and editing, C.L. and X.N.; supervision, X.N. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by a grant from the Hubei Key Laboratory of Intelligent Robot of China (Grant No. HBIRL202009).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The authors can provide the raw data in this work upon reasonable request.

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

## References

1. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.
2. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards realtime object detection with region proposal networks. In Proceedings of the 2015 Advances in Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; pp. 91–99.
3. Girshick, R. Fast RCNN. In Proceedings of the 2015 IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
4. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
5. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single shot multibox detector. In Proceedings of the 2016 European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; pp. 21–37.
6. Jeong, J.; Park, H.; Kwak, N. Enhancement of SSDby concatenating feature maps for object detection. *arXiv* **2017**, arXiv:1705.09587.
7. Fu, C.Y.; Liu, W.; Ranga, A.; Tyagi, A.; Berg, A.C. DSSD: Deconvolutional single shot detector. *arXiv* **2017**, arXiv:1701.06659.
8. Li, Z.; Zhou, F. FSSD: Feature fusion single shot multibox detector. *arXiv* **2017**, arXiv:1712.00960.
9. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, realtime object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
10. Redmon, J.; Farhadi, A. Yolo9000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7263–7271.
11. Redmon, J.; Farhadi, A. YOLOv3: Anincremental improvement. *arXiv* **2018**, arXiv:1804.02767.
12. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. YOLOv4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934.
13. Vosugh, N.; Bahmani, Z.; Mohammadian, A. Distracted driving recognition based on functional connectivity analysis between physiological signals and perinasal perspiration index. *Expert Syst. Appl.* **2023**, *231*, 120707. [CrossRef]
14. Luo, G.; Xiao, W.; Chen, X.; Tao, J.; Zhang, C. Distracted driving behaviour recognition based on transfer learning and model fusion. *Int. J. Wirel. Mob. Comput.* **2023**, *24*, 159–168. [CrossRef]
15. Ping, P.; Huang, C.; Ding, W.; Liu, Y.; Chiyomi, M.; Kazuya, T. Distracted driving detection based on the fusion of deep learning and causal reasoning. *Inf. Fusion* **2023**, *89*, 121–142. [CrossRef]

16. Lin, Y.; Cao, D.; Fu, Z.; Huang, Y.; Song, Y. A Lightweight Attention-Based Network towards Distracted Driving Behavior Recognition. *Appl. Sci.* **2022**, *12*, 4191. [CrossRef]
17. Zhao, X.; Li, C.; Fu, R.; Ge, Z.; Wang, C. Real-time detection of distracted driving behaviour based on deep convolution-Tokens dimensionality reduction optimized visual transformer. *Automot. Eng.* **2023**, *45*, 974–988+1009. [CrossRef]
18. Cao, L.; Yang, S.; Ai, C.; Yan, J.; Li, X. Deep learning based distracted driving behaviour detection method. *Automot. Technol.* **2023**, *06*, 49–54. [CrossRef]
19. Zhang, B.; Fu, J.; Xia, J. A training method for distracted driving behaviour recognition model based on class spacing optimization. *Automot. Eng.* **2022**, *44*, 225–232. [CrossRef]
20. Feng, T.; Wei, L.; Wenjuan, E.; Zhao, P.; Li, Z.; Ji, Y. A distracted driving discrimination method based on the facial feature triangle and bayesian network. *Balt. J. Road Bridge Eng.* **2023**, *18*, 50–77. [CrossRef]
21. Chen, D.; Wang, Z.; Wang, J.; Shi, L.; Zhang, M.; Zhou, Y. Detection of distracted driving via edge artificial intelligence. *Comput. Electr. Eng.* **2023**, *111*, 108951. [CrossRef]
22. Lu, M.; Hu, Y.; Lu, X. Pose-guided model for driving behavior recognition using keypoint action learning. *Signal Process. Image Commun.* **2022**, *100*, 116513. [CrossRef]
23. Dehzangi, O.; Sahu, V.; Rajendra, V.; Taherisadr, M. GSR-based distracted driving identification using discrete & continuous decomposition and wavelet packet transform. *Smart Health* **2019**, *14*, 100085.
24. 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]
25. Zhao, L.; Yang, F.; Bu, L.; Han, S.; Zhang, G.; Luo, Y. Driver behavior detection via adaptive spatial attention mechanism. *Adv. Eng. Inform.* **2021**, *48*, 101280. [CrossRef]
26. Hossain, M.U.; Rahman, M.A.; Islam, M.M.; Akhter, A.; Uddin, M.A.; Paul, B.K. Automatic driver distraction detection using deep convolutional neural networks. *Intell. Syst. Appl.* **2022**, *14*, 200075. [CrossRef]
27. Zhang, Y.; Chen, Y.; Gao, C. Deep unsupervised multi-modal fusion network for detecting driver distraction. *Neurocomputing* **2021**, *421*, 26–38. [CrossRef]
28. Singh, H.; Sidhu, J. Smart Detection System for Driver Distraction: Enhanced Support Vector Machine classifier using Analytical Hierarchy Process technique. *Procedia Comput. Sci.* **2023**, *218*, 1650–1659. [CrossRef]
29. Aljohani, A.A. Real-time driver distraction recognition: A hybrid genetic deep network based approach. *Alex. Eng. J.* **2023**, *66*, 377–389. [CrossRef]
30. Xiao, W.; Liu, H.; Ma, Z.; Chen, W. Attention-based deep neural network for driver behavior recognition. *Future Gener. Comput. Syst.* **2022**, *132*, 152–161. [CrossRef]
31. Lu, M.; Hu, Y.; Lu, X. Dilated Light-Head R-CNN using tri-center loss for driving behavior recognition. *Image Vis. Comput.* **2019**, *90*, 103800. [CrossRef]
32. Cammarata, A.; Sinatra, R.; Maddio, P.D. Interface reduction in flexible multibody systems using the Floating Frame of Reference Formulation. *J. Sound Vib.* **2022**, *523*, 116720. [CrossRef]