Article

Lightweight Deep Learning for Road Environment Recognition

Han Liang and Suyoung Seo *

Department of Civil Engineering, Kyungpook National University, Daegu 41566, Korea; shto613@knu.ac.kr
* Correspondence: syseo@knu.ac.kr; Tel.: +82-539-505-613

Featured Application: The system proposed in this paper provides a new idea for the recognition algorithm of road environments in safety-assisted driving systems, and subsequently improves the existing algorithm to reduce the computational cost and improve accuracy.

Abstract: With recent developments in the field of autonomous driving, recognition algorithms for road environments are being developed very rapidly. Currently, most of the network models have good recognition rates, but as the accuracy rate increases, the models become more complex and thus lack real-time performance. Therefore, there is an urgent need to propose a lightweight recognition system for road environments to assist autonomous driving. We propose a lightweight road environment recognition system with two different detection routes based on the same backbone network for objects and lane lines. The proposed approach uses MobileNet as the backbone network to acquire the feature layer, and our improved YOLOv4 and U-Net allows the number of parameters of the model to be greatly reduced and combined with the improved attention mechanism. The lightweight residual convolutional attention network (LRCA-Net) proposed in this work allows the network to adaptively pay attention to the feature details that need attention, which improves the detection accuracy. Finally, the object detection model and the lane line detection model of this lightweight road environment detection system evaluated on the PASCAL VOC dataset and the Highway Driving dataset show that their mAP and mIoU reach 93.2% and 93.3%, respectively, achieving excellent performance compared to other methods.

Keywords: object detection; lane detection; attention mechanism; lightweight network

1. Introduction

The fields of autonomous driving (AD) and safety driving assistance systems (SDAS) have seen a significant amount of research in recent years. As the deep learning field has grown in recent years, the field of autonomous driving has moved at a breakneck pace. It is well known that deep learning algorithms often perform better than classical algorithms in these aspects of image processing due to variations in lighting conditions, shadows, road breaks and occlusions, and different camera settings, which often result in classical algorithms not performing well. Many works can be found in the literature based on deep learning for road image processing, and invariably the two most important challenges are the localization and classification of targets encountered while driving, and the segmentation of roads taken by the driver.

In general, the goal of object detection is to locate the studied object in each image using a rectangular prediction frame and output the class and confidence level of the object. The evolution of object detection algorithms is divided into two phases: one is the traditional feature-based solution, and the other is the deep learning algorithm. Before 2013, the mainstream detection algorithms were traditional feature-optimized detection methods, which usually consisted of three parts, the first being the selection of the detection window, the second being the design of the features, and the third being the design of the classifier. The sliding window is used to traverse the whole image and extract the features from the window, and then the classifier is used to detect them. The common methods...
used in the feature extraction stage are Haar, histogram of oriented gradient (HOG), local binary pattern (LBP), aggregated channel features (ACF), and other operators, and the common classifiers are SVM, boosting, random forest, and so on. With the Adaboost-based face detection method [1], the object detection algorithm has experienced the traditional framework of manually designed features plus shallow classifiers and reached the pinnacle of traditional object detection. However, the traditional sliding window technique used for object detection needs to handle thousands of windows and has low performance without optimization strategies. Secondly, because of the manual design of features, it cannot express the characteristics of the object in more detail, resulting in a lower recognition rate. So, after 2013, the whole academic and industrial communities gradually started using convolutional neural network (CNN) to do object detection.

On the other hand, classical image processing and computer vision approaches typically divide lane line recognition algorithms into four distinct steps. A considerable amount of background noise and extra pixel information may be removed by first establishing an orthogonal system concerning the body and the road surface. Then, the lane detection zone can be calculated. Image enhancement techniques such as smoothing, sharpening, and other methods are used in the second step of lane line feature extraction to improve the quality of the image. Examples include image-enhancing algorithms for the night, fog, or shadow photos [2–4]. The extraction of image features is the third step in the procedure. Image-based algorithms for lane line detection rely primarily on picture characteristics such as lane line shapes, pixel gradients, and color cues to identify lane lines [5–7]. In the end, some straight lines or curves are used to match the lane lines [8–10]. To employ these classic methods, the filtering operator must be de-tuned, and the algorithm’s parameters manually adjusted based on the features of the street scene that is being targeted by the algorithm. In addition, the recognition of lane lines fails when the driving environment undergoes major changes. To meet the ever-increasing demands for lane line recognition precision, the procedure becomes more complex. Therefore, image processing and computer vision technologies will be phased out in favor of semantic segmentation methods, which have only recently begun to be researched.

2. Related Work

Most of the early object detection is based on deep learning using the sliding window approach for window extraction, which is essentially an exhaustive method R-CNN [11]. Later, regional window extraction algorithms such as selective search were proposed, where instead of using a sliding window to scan the image for a given image, some candidate windows are “extracted”, and the number of candidate windows can be controlled to be in the range of thousands or hundreds, provided that an acceptable recall is obtained for the object to be detected [12]. The number of candidate windows can be limited to a few thousand or a few hundred, provided that an acceptable recall is obtained for the detected target.

The SPP layer [13] solves this problem well by first dividing the whole image into 4 equal parts and extracting features of the same dimension in each part, then dividing the image into 16 equal parts, and so on. The extracted dimensional data are consistent regardless of the image size so that they can be sent to the fully connected layer uniformly. Although R-CNN and SPP have made great advances in detection, the duplicate computation problems they bring are problematic, and Fast R-CNN emerged to solve these problems.

Fast R-CNN uses a simplified SPP layer called the region of interest (RoI) pooling layer. Fast R-CNN also uses SVD to decompose the parameter matrix of the fully connected layer, compressing it into two much smaller fully connected layers [14]. Faster R-CNN uses region proposal networks (RPN) to compute candidate frames directly, which takes a picture of arbitrary size as input and outputs a batch of rectangular regions, each corresponding to a target score and location information [15]. Image object detectors are usually divided into two types: one is a two-stage detector, which is characterized by high detection
accuracy. R-CNN, Fast R-CNN, and Faster R-CNN are two-stage detection algorithms, and YOLO [16] and SSD [17], which consider object detection as a regression problem, are one-stage detection algorithms.

According to prior research, it seems that even though the two-stage detector has a high identification rate, it does not perform particularly well when employed in real-time applications [18]. In road environment recognition, the ability to detect obstacles in real-time is crucial, not only to achieve high recognition rates but also to have fast processing speed. The YOLO family of algorithms [16,19–22] and the single shot multibox detector (SSD) algorithm [17] are among the one-stage algorithms, these types of algorithms directly regress the class confidence and coordinate values of the object, and the detection speed is very fast and very suitable for real-time detection.

Deeper neural networks are frequently required to achieve higher accuracy in lane semantic segmentation models, making segmentation models more complex and slower, and in some recent approaches, segmentation networks have become increasingly complex, resulting in models that require large GPU resources and are slow.

For example, the SCNN method [23] for extracting lane lines is effective for slender lane line detection by sequentially convolving in a certain direction compared to the traditional network that convolves directly between layers but is slow at only 7.5 FPS. For the lane curve part, the CurveLanes-NAS [24] aims to solve the curved lane line detection problem by capturing the global coherence features and local curvature features of the lane lines from the perspective of network search for long lane lines. Although state-of-the-art results have been achieved, they are computationally very time-consuming.

To address the speed issue, there is also some recent research focusing on segmentation speed improvement with SwiftNet for road driving image segmentation. SwiftNet [25] uses a lightweight general framework with horizontal connectivity and a resolution pyramid approach based on shared parameters to increase the perceptual field of the model. The segmentation speed of this model is faster, but its accuracy is not high.

Yu et al. proposed BiSeNet [26], a bidirectional segmentation network. One module deals with spatial information and the other with contextual information, and then proposed a new module for fusing features. A 68.4% IOU has been achieved in 2048 × 2024 high-resolution access and 105 FPS in NVIDIA Titan X. The structure of BiSeNet is relatively simple and the segmentation speed is fast, but the segmentation accuracy is still slightly lacking. Currently, the difficulties in semantic segmentation stem from the loss of spatial information and too small acceptance fields. Due to these problems, the accuracy of image semantic segmentation suffers. Even though these lightweight models focus on improving segmentation speed, they produce poor segmentation results and fail to trade-off well.

To improve target detection and lane semantic segmentation in road environments, this paper proposes a single-stage target detection algorithm and a multi-feature fusion semantic segmentation algorithm based on the same lightweight backbone network and combined with a modified attention mechanism for the improved single-stage target detection algorithm. Its main contributions are provided as follows.

A modified attention mechanism module that combines spatial and the channel is proposed. By improving the channel attention module in CBAM [27] and using 1D convolution to replace the fully connected layer in the original module, which not only avoids dimensionality reduction and effectively captures the information of cross-channel interactions but also greatly reduces the number of parameters. In addition, a residual block is added to the whole attention mechanism to solve the gradient dispersion problem caused by the sigmoid function. It enhances the representation of features, focuses on important features, and suppresses unimportant features, thus improving network accuracy.

The object detection algorithm replaces the backbone network in YOLOv4 [21] with the lightweight network MobileNet [28] and modifies some convolutions in the YOLOv4 feature fusion network to reduce the number of parameters in the network. This makes the whole network lighter while ensuring that accuracy is not compromised.
Based on the decoder-encoder end-to-end architecture model, MobileNet is used as the backbone feature extraction network for lane detection, and the extracted multiple sets of features are decoded through a series of upsampling and downsampling and connected to the corresponding feature layers to finally obtain pixel probability results for each category. It has the features of low parameters of the lightweight model, fast convergence, and high accuracy.

Experimental results show that our proposed detection system is effective and maintains high detection quality with a smaller detection model on the PASCAL VOC and freeway driving datasets. Thus, the computational cost of our method is much lower than state-of-the-art methods.

3. Proposed Method

In this paper, we propose a lightweight deep learning system for object detection and lane detection as shown in Figure 1. Firstly, we propose the lightweight residual convolution attention network, which makes the network pay attention to the detailed features it needs and suppresses the interference of other useless information, to be applied in the object detection and lane semantic segmentation network to improve the network performance. Secondly, we propose an object detection network by replacing the YOLOv4 backbone network with a lightweight network MobileNet and modifying the normal convolution in the feature fusion network with a depthwise separable convolutional layer, which combines the network attention mechanism to make the network more efficient while greatly reducing the number of parameters. Third, a lane semantic segmentation network is proposed, based on the lightweight network MobileNet as the backbone network, using the extracted feature layers and the extended path method of U-Net [29] as the decoder, which can increase the local perceptual field and collect multi-scale information without reducing the dimensionality. Additionally, the feature representation can be further enhanced by inserting an attention mechanism in the feature layer fusion process. Such an approach can effectively utilize the dataset and improve the segmentation accuracy of the network.

Figure 1. Lightweight system for object detection and lane semantic segmentation.

3.1. Lightweight Residual Convolutional Attention Network

The central focus of the attention mechanism is to get the network to pay attention to the features it needs to pay more attention to. When we use convolutional neural networks to process images, it is impossible for us to manually adjust what needs attention, and this is when it becomes extremely important to make the convolutional neural network...
pay attention to important objects adaptively. The attention mechanism is one way to achieve adaptive attention of the network. The lightweight residual convolutional attention network (LRCA-Net) proposed in this work is an improved attention mechanism proposed in this study for improving accuracy.

Early on, attentional mechanisms were analyzed from brain imaging mechanisms, using the winner-takes-all [30] mechanism to study how to model attention. In deep learning, it is now more important to build neural networks with attention mechanisms, because they can focus on more detailed information about the target and suppress the interference from other useless information. In convolutional neural networks, visual attention is usually divided into two forms: channel and spatial attention. The CBAM [27] consists of a serial connection between the channel attention module and the spatial attention module. It can calculate the attention map of the feature map from both channel and spatial dimensions, and then multiply the attention map with the input feature map to perform adaptive learning of features.

The LRCA-Net proposed in this paper is an improved module based on CBAM. The overall structure is shown in Figure 2. First, the idea of residuals is added to the network architecture of the CBAM module, and the original features \( F \) and the features \( F'' \) after CBAM are directly summed and fused. Second, the fully connected layer of the channel attention module is replaced by using 1D convolution.

Figure 2. The structure of LRCA-Net.

It can be seen that the input feature map \( F \) has a shape of \( H \times W \times C \) after the channel attention module \( A_c \) to get the attention weight \( A_c(F) \) with the shape of \( 1 \times 1 \times C \). Then, \( A_c(F) \) and \( F \) are multiplied to get the feature \( F' \), as shown in Equation (1). \( F' \) then goes through a spatial attention mechanism \( A_s \) to the attention weight \( A_s(F') \) with the shape of \( H \times W \times 1 \). Then, \( A_s(F') \) and \( F' \) are multiplied to get the final feature \( F'' \), the shape of \( F'' \) is \( H \times W \times C \), as shown in Equation (2).

After a series of processing, the shape of the feature does not change, so this attention mechanism can be inserted after any feature, the network does not need to make changes. Finally, the output feature \( F''' \) is obtained by summing \( F \) and \( F'' \) using the residual idea, as shown in Equation (3).

\[
F' = F \times A_c(F), \tag{1}
\]

\[
F'' = F' \times A_s(F'), \tag{2}
\]

\[
F''' = F'' + F, \tag{3}
\]

where \( F \) is the input feature map, \( F'' \) is the spatial-refined feature, and \( F''' \) is the output feature, \( A_c \) is the channel attention module, and \( A_s \) is the spatial attention module.

In convolution, there will be multiple channels of feature outputs, and some channel features will have a greater impact on the final target, so it is necessary to focus attention on these channels better, and a common practice is global pooling. Two types of pooling—global maximum pooling and global average pooling—are used in the original
CBAM. As shown in the original channel attention module in Figure 3, the input feature maps are reshaped into (1, 1, C) after max-pooling and averaging pooling, respectively. To capture the nonlinear cross-channel interactions, the original channel attention module uses two fully connected layers with nonlinearity. The two results of maximum pooling and average pooling of shape (1, 1, C) are summed after two full-connected operations, and then the channel attention weight \( A_c \) of shape (1, 1, C) is obtained by the sigmoid function.

![Figure 3. Improved Channel Attention Module and Spatial Attention Module.](image)

According to the experiments of [31] Wang Q.L. et al., it can be seen that two fully connected layers have side effects on channel attention prediction and capturing the dependencies between all channels is inefficient and unnecessary. Therefore, in this study, we modified the original CBAM by exploiting the local cross-channel interaction strategy of the ECA-Net module to improve the channel attention module of CBAM. The modification was performed by replacing the fully connected layer of the channel attention module with an adaptive selection of the size of the 1D convolutional kernel. As shown in the modified channel attention module in Figure 3, the two pooling results are reshaped into (1, 1, C), respectively, by 1D convolution operation, and then the summed results are sigmoid to obtain the weights \( A_c \). Such a modified module can effectively capture the information of cross-channel interactions, thus achieving an improvement in the overall attention mechanism.

For the spatial attention module there is no modification in this study, and for the input incoming feature layer, the maximum and average values are taken over the channels, which are shaped as (H, W, 1). After that the two results are concatenated into (H, W, 2) and the number of channels is adjusted using a convolution with an input channel number of 2 and an output channel number of 1. Then, the sigmoid is taken, at which point we get the spatial attention weight \( A_s \).

Moreover, as shown in Figure 2, we modified the original CBAM by adding residual blocks to the entire attention mechanism to solve the gradient dispersion problem caused by the sigmoid function since both the channel and spatial attention modules use the sigmoid function to generate the weights.

The structural analysis of CBAM and LRCA-Net can be obtained from Table 1, assuming that the input feature shape is (26, 26, 512) it can be seen by analyzing the CBAM structure that the number of parameters is mainly concentrated in the two fully connected layers of the channel attention module, and the overall number of parameters can be made to plummet after using 1D convolutional substitution. Assuming that the input feature
shape is (26, 26, 512) it can be seen by analyzing the CBAM structure that the number of parameters is mainly concentrated in the two fully connected layers of the channel attention module, and the overall number of parameters can be made to plummet after using 1D convolutional substitution.

Table 1. Structure and parameters of CBAM and LRCA-Net.

| Module | Layer | Output Shape | Parameters | Total Parameters |
|--------|-------|--------------|------------|------------------|
| CBAM   | Input | 26 × 26 × 512 | 0          |                  |
|        | Global Average Pooling (Input) | 1 × 1 × 512 | 0          |                  |
|        | Global Max Pooling (Input) | 1 × 1 × 512 | 0          |                  |
|        | Channel Attention_FC1 (GAP, GMP) | 1 × 1 × 64 | 32,768     |                  |
|        | Channel Attention_FC2 (FC1(GAP), FC1(GMP)) | 1 × 1 × 512 | 32,768     |                  |
|        | Add (CA_FC2) | 1 × 1 × 512 | 0          |                  |
|        | Activation (Sigmoid)_1 (Add) | 1 × 1 × 512 | 0          |                  |
|        | Multiply_1 (Input, Activation_1) | 26 × 26 × 512 | 0          |                  |
|        | Average Pooling (Multiply_1) | 26 × 26 × 1 | 0          |                  |
|        | Max Pooling (Multiply_1) | 26 × 26 × 1 | 0          |                  |
|        | Concatenate (AP, MP) | 26 × 26 × 2 | 0          |                  |
|        | Spatial Attention (Concatenate) | 26 × 26 × 1 | 98         |                  |
|        | Activation (Sigmoid)_2 (SA) | 26 × 26 × 1 | 0          |                  |
|        | Multiply_2 (Multiply_1, Activation_2) | 26 × 26 × 512 | 0          |                  |
|        | LRCA-Net | 26 × 26 × 512 | 0          |                  |
|        | Global Average Pooling (Input) | 1 × 1 × 512 | 0          |                  |
|        | Global Max Pooling (Input) | 1 × 1 × 512 | 0          |                  |
|        | Channel Attention_1DConv 1 (GAP) | 1 × 1 × 512 | 5          |                  |
|        | Channel Attention_1DConv 2 (GMP) | 1 × 1 × 512 | 5          |                  |
|        | Add_1 (Channel Attention_1DConv 1, Channel Attention_1DConv 2) | 1 × 1 × 512 | 0          |                  |
|        | Activation (Sigmoid)_3 (Add_1) | 1 × 1 × 512 | 0          |                  |
|        | Multiply_3 (Input, Activation_3) | 26 × 26 × 512 | 0          |                  |
|        | Average Pooling (Multiply_3) | 26 × 26 × 1 | 0          |                  |
|        | Max Pooling (Multiply_3) | 26 × 26 × 1 | 0          |                  |
|        | Concatenate (AP, MP) | 26 × 26 × 2 | 0          |                  |
|        | Spatial Attention (Concatenate) | 26 × 26 × 1 | 98         |                  |
|        | Activation (Sigmoid)_4 (SA) | 26 × 26 × 1 | 0          |                  |
|        | Multiply_4 (Multiply_3, Activation_4) | 26 × 26 × 512 | 0          |                  |
|        | Add_2 (Input, Multiply_4) | 26 × 26 × 512 | 0          |                  |

3.2. Object Detection Network

YOLOv3 has made some enhancements over its predecessors, YOLOv1 and YOLOv2. This algorithm is generally used to enhance category prediction, multi-scale prediction, and bounding box prediction, as well as multi-label classification, among other tasks. To achieve CSPDarknet53, YOLOv4 enhances the backbone network, which contains 29 convolutions, 725 × 725 receptive fields, and 27.6 million parameters.

As shown in Figure 4, the YOLOv4 architecture consists of the following components: CSPDarknet53 + SPP + PANet + YOLO Head. After resizing the original picture to 416 × 416 resolution as input, the algorithm employs up-sampling and feature fusion
operations to divide the original image into \( S \times S \) grids, where \( S \) can be 13, 26, or 52, depending on the scale of the feature map, to forecast on the feature image of several scales. With the use of three anchor boxes, each size grid can estimate the object border. Finally, YOLO Head will display the bounding box position and size \((x, y, h, w)\), as well as the object’s category, with confidence.

**Figure 4.** Structure of YOLOv4.

Advanced network structures can achieve good accuracy with a smaller number of parameters. This is because having too many parameters in a network structure can eventually lead to slow training. The convergence speed can be greatly accelerated by reducing the number of parameters. It has been a challenge to make the network less computationally intensive while ensuring accuracy.

Reference [32] found that most neural networks are over-parameterized, and after their study found that excessive training weights have little to no effect on overall accuracy, and in many networks, it is even possible to remove 80–90% of the network weights with little loss in accuracy. So, after we choose the object detection model architecture as YOLOv4, we do some work to make the network lighter and with fewer parameters, which makes the network more efficient.

First, we will use MobileNet to replace YOLOv4’s backbone network. MobileNet is a lightweight network designed for mobile terminals or embedded devices, which has been developed into v1 [28], v2 [33], and v3 [34] versions. The MobileNet model is built around depthwise separable convolutions, which are a type of factorized convolution that factorizes a standard convolution into a depthwise convolution and a \( 1 \times 1 \) convolution known as a pointwise convolution. A standard convolution filters and combines inputs into a new set of outputs in a single step, as illustrated in Figure 5a. Figure 5b depicts the depthwise separable convolution that divides this into two layers, one for filtering and one for combining. This factorization has the effect of reducing computation and model size significantly.

The iterative concatenation of \( 3 \times 3 \) depthwise separable convolutions can be used to form the backbone feature extraction network of MobileNetv1 as shown in Figure 6 to replace the backbone network of the original YOLOv4. We take out the effective feature layers of the last three shapes of MobileNetv1 for the subsequently enhanced feature extraction.

As shown in Figure 7a, MobileNetv2 is an upgraded version of MobileNetv1, which has a very important feature of Inverted Resblock, the whole MobileNetv2 is composed of Inverted Resblock. The left side is the backbone part, which first uses \( 1 \times 1 \) convolution for upscaling, then \( 3 \times 3 \) depth separable convolution for feature extraction, and then \( 1 \times 1 \) convolution for downscaling. The right side is the residual edge part, where the input and output are directly connected. As can be seen in Figure 7b, MobileNetv3 is a combination of ideas from the following three models: MobileNetv1’s depthwise separable
convolutions, MobileNetv2’s the inverted residual with linear bottleneck, and the attention mechanism b-neck structure is introduced on top of it, which works by adjusting the weights of each channel. The HardSwish activation function is also introduced to reduce the number of operations and improve performance. The backbone network of YOLOv4 is replaced by MobileNetv2 and MobileNetv3 in the same way as in Figure 6, so that we have three object detection networks, MobileNetv1-YOLOv4, MobileNetv2-YOLOv4, and MobileNetv3-YOLOv4. The total number of parameters of each network is calculated and compared with the total parameters of the original YOLOv4 network to obtain the results shown in Figure 8.

![Figure 5. The difference between standard convolution and depthwise separable convolution. (a) Standard convolution; (b) Depthwise separable convolution.](image)

![Figure 6. MobileNetv1 replaces the original YOLOv4 backbone network.](image)
The parameters of each network is calculated and prove performance. The backbone network of YOLOv4 is replaced by MobileNetv1 and MobileNetv3. Table 1 shows the total parameters of the original YOLOv4 network, MobileNetv1-YOLOv4, MobileNetv2-YOLOv4, and MobileNetv3-YOLOv4 are directly connected. As can be seen in Figure 7b, MobileNetv3 is a combination of ideas from the following three models: MobileNetv1, MobileNetv2, and MobileNetv3. After replacing the backbone network, the number of parameters in each network is significantly reduced compared to the original YOLOv4, but still has a huge number of about 40 million. Therefore, this work replaces the standard convolutions in the SPP and PANet with the depthwise separable convolutions as shown in Figure 9.

![Figure 7](image.png)

**Figure 7.** The basic modules that make up (a) MobileNetv2 and (b) MobileNetv3.

![Figure 8](image.png)

**Figure 8.** Comparison of parameters before and after replacing only the backbone network.

After replacing the backbone network, the number of parameters in each network is significantly reduced compared to the original YOLOv4, but still has a huge number of about 40 million. Therefore, this work replaces the standard convolutions in the SPP and PANet with the depthwise separable convolutions as shown in Figure 9.

![Figure 9](image.png)

**Figure 9.** The depthwise separable convolutions are used to replace the standard convolutions in SPP and PANet.
This replacement can significantly reduce the number of weights of the network. As shown in Figure 10, the number of parameters of the fully improved network is only one-sixth of the original YOLOv4 network. Finally, to further improve the detection accuracy, as in Figure 11 after the feature fusion network layer, our proposed attention mechanism is added before outputting the results.

**Figure 10.** The parameters of each improved model are compared with the original model.

**Figure 11.** The proposed object detection network.

### 3.3. Lane Detection Network

The semantic segmentation network is a neural network and a model based on deep convolutional networks. Its most basic task is to classify different types of pixel points in an image and to aggregate the same type of pixel points to distinguish different target objects in the image [35–37]. One of the U-Net model structures can increase the local perceptual field and collect multi-scale information without reducing the dimensionality. Deep learning networks usually require many datasets for training. This training method allows more efficient use of datasets and enables the network to perform more accurate segmentation even with a small number of training images, so the U-Net model is often
used in medical image processing. Continuing this idea of U-Net, this work proposes three lightweight road semantic segmentation models using the MobileNet series as the backbone network, as shown in Figure 12, and the models are divided into four parts.

The first part is MobileNetv1 as the backbone network is the encode process, i.e., the feature extraction process. It consists of many depthwise separable convolutions iteratively strung together. According to its characteristics mentioned before, we can understand that this can significantly reduce the number of network parameters. In this way, we can use the backbone to obtain five feature layers, whose shapes are (208, 208, 64), (104, 104, 128), (52, 52, 256), (26, 26, 512), and (13, 13, 1024). Since we only need to single extract the safe lane that the driver is driving in, with fewer classification items, the first feature layer (208, 208, 64) and the last layer (13, 13, 1024) are discarded from use. This way we will use the middle three effective feature layers for feature fusion.

The second part is feature layer fusion, which enhances the diversity of feature extraction. The backbone network outputs feature layers (26, 26, 512), and after the ZCB module (ZeroPadding+Conv2D+BN) the output is combined with the attention mechanism module proposed in this study for upsampling and then superimposed with the feature layer (52, 52, 512) of the backbone network output, and this process is repeated to finally obtain a valid feature layer that fuses all features.

The third part is the attention mechanism. Adding the LRCA-Net module to the overlay process of the feature fusion layer enables the network to focus its attention on the effective features and ensures the normal convergence during training.

The fourth part is the prediction part, where we use the final effective feature layer obtained in the second part to classify each feature point by softmax, which is equivalent to classifying each pixel point. The semantic segmentation model thus designed has the advantage of both a lightweight backbone model with a small number of parameters and combines the features of encode-decode structure and skip connection.

4. Experimental Results

4.1. Experimental Dataset and Parameters Setting

The object detection network proposed in this paper is evaluated on the datasets Pascal VOC 2007 and 2012. The dataset includes 11,180 annotated images, where the training set, the validation set are randomly divided into 10,062 and 1118 according to the ratio of 9:1. In our experiments, five targets commonly found in road environments are used for
detection, i.e., bus, bicycle, motorbike, people, and car. Figure 13 shows the percentage of ground truth in the dataset for each target.

Figure 13. Percentage of ground truth for each target.

The lane detection network was evaluated on highway driving dataset for semantic video segmentation from KAIST. The database consists of a total of 20 sequences of 60 frames at a frame rate of 30 Hz. Each video clip was taken at a fixed location in the vehicle’s black box. For each sequence a manually annotated sequence is provided, i.e., each provided frame is densely annotated. The input image resolution is 1080 × 720, and there are 1200 annotated images, of which the training set and validation set are randomly divided into 1080 and 120 according to the ratio of 9:1. We selected seven classes in the dataset and the details of the selected classes are shown in Table 2.

Table 2. Details of mark labels in the dataset.

| Category       | Color |
|----------------|-------|
| Lane           |       |
| Fence          |       |
| Construction   |       |
| Traffic Sign   |       |
| Car            |       |
| Truck          |       |
| Vegetation     |       |

The experimental setting of this paper is shown in Table 3. The TensorFlow framework is used to build the experimental model’s training, validation, and testing, and the CUDA kernel is used to calculate the results. The hardware consists primarily of a high-performance workstation host. The workstation is outfitted with an Intel(R) Core(TM) i7-870 processor and a GTX 1050Ti graphics card.
Table 3. The experimental environment.

| Items            | Description                      |
|------------------|----------------------------------|
| H/W              |                                  |
| CPU              | Intel(R) Core(TM) i7-10700        |
| RAM              | 32 GB                            |
| SSD              | Samsung SSD 512 GB               |
| Graphics Card    | NVIDIA GeForce GTX 1650          |
| S/W              |                                  |
| Operating System | Windows 10 Pro, 64bit            |
| Programming Language | Python 3.6                |
| Learning Framework | TensorFlow 1.13.2           |
| API              | Keras 2.1.5                      |

4.2. Metrics

To validate the accuracy of object detection and lane detection, we used the following metrics to evaluate each model. Object detection mainly uses precision, recall, and average precision as shown in Equations (4)–(7). Lane detection mainly uses mean intersection over union (mIoU) and mean pixel accuracy (mPA). T/F denotes true/false, which indicates whether the prediction is correct, and P/N denotes positive/negative, which indicates a positive or negative prediction result.

\[
\text{Precision} = \frac{TP}{TP + FP}, \quad (4)
\]

\[
\text{Recall} = \frac{TP}{TP + FN}. \quad (5)
\]

However, in some data sets with unbalanced distribution, if the number of negative samples is very small it will lead to precision close to perfection and recall close to zero, obviously this situation alone using any one indicator is not able to fully evaluate the system, so average precision is introduced. The AP is calculated using the difference-average accuracy measure, which is the area under the precision-recall curve, as shown in Equation (6).

\[
\text{AP} = \left( \frac{1}{n} \sum_{r=\frac{1}{n}}^{\frac{n-1}{n}} p_{\text{interop}}(r) \right)^2, \quad (6)
\]

where \(n\) denotes the number of detection points and \(p_{\text{interop}}(r)\) represents the value of the accuracy at a recall of \(r\).

Based on the AP, mAP (Mean Average Precision) can be calculated as shown in Equation (7).

\[
\text{mAP} = \frac{1}{n} \sum \text{AP}. \quad (7)
\]

In the evaluation metric of lane detection, IoU is used to represent the ratio of the intersection of the predicted result and the true value for a category to the merged set, and pixel accuracy (PA) represents the ratio of the number of pixels correctly predicted for a category to the total number of pixels, as shown in Equations (8) and (9).

\[
\text{mIoU} = \frac{TP}{TP + FP + FN} = \frac{1}{n+1} \sum_{i=0}^{n} \frac{p_{ij} - p_{ij}'}{\sum_{j=0}^{n} p_{ij} + \sum_{j=0}^{n} p_{ji} - p_{ii}'}, \quad (8)
\]

\[
\text{mPA} = \frac{TP + TN}{TP + FP + FN + TN} = \frac{1}{n+1} \sum_{i=0}^{n} \frac{p_{ii}}{\sum_{j=0}^{n} p_{ij}}, \quad (9)
\]

where \(n\) represents the number of classes. \(i\) represents the true value, \(j\) represents the predicted value. Then, \(p_{ij}\) denotes TP, \(p_{ij}\) denotes TN, \(p_{ij}\) denotes FP, and \(p_{ij}\) denotes FN.
4.3. Experimental Result and Analysis

Figure 14 shows the loss value plot during the training of the model and the validation loss function plot obtained by using the validation dataset to validate the model during the training process. The pre-trained MobileNet model is used to initialize the weight parameterization of the underlying shared convolutional layer. For the training of the object detection and lane detection models, the loss functions use YOLO loss and Focal loss, respectively and the training batch size was set to 8 and 4, respectively, with initial learning rates of 0.001 and 0.0001, respectively, and a total training iteration count of 100. The loss function completed convergence at 57 and 74 epochs, respectively.

![Model loss function graph.](image)

Firstly, to verify the optimization ability of the attention mechanism LRCA-Net proposed in this paper, two sets of control experiments were conducted with the network without adding the attention mechanism and the network with CBAM as the attention mechanism for object detection and lane detection, respectively, as shown in Figure 15. The results of the two quantitative experiments for object detection and lane detection are given in Tables 4 and 5. Taking the backbone network Mobilenetv1 as an example, mAP and mPA are 90% and 90.3%, respectively, when no attention mechanism is added. Additionally, the proposed method in this paper mAP, mPA can reach 93.2% and 93.1%, which also improves 1.8% and 1.7%, respectively compared to CBAM, while the number of parameters gets reduced to some extent. It can be no matter for object detection or lane detection, the attention mechanism method proposed in this paper can effectively improve the performance of the algorithm, and certain optimization improvements compared to CBAM. Note that our LRCA-Net obtains higher accuracy while having less model complexity.

Table 6 gives a comparison of the quantitative experimental results using the same dataset and model training methods. Here, the proposed object detection algorithm proposed in this paper is compared with SSD, Faster RCNN, YOLOv4, YOLOvX series, EfficientDet series [38]. Our proposed method can achieve 93.2%, 92%, 91.8% mAP depending on the backbone network, where the model with the backbone network of MobileNetv1 is the optimal model, which is an improvement compared with other detection methods. As shown in Figure 16, the mAP is improved by 5.5% and 2.3%, respectively, compared with YOLOX-S and EfficientDet-D3, which have a similar number of parameters, for the optimal model. The volume reduction was 77%, 87.4%, and 64.8% compared to YOLOX-L, YOLOX-X, and EfficientDet-D5 with mAP at the same level. For the time factor, compared to YOLOX-L, YOLOX-X, and EfficientDet-D5, which achieve the same level of mAP but
only 6, 3, and 11 fps, respectively, our method is stable at around 30 fps, which is a good improvement for processing speed. This is because many network topologies have too many unnecessary parameters, which not only cause slow convergence during model training but also may overfit and affect the accuracy of detection results. Our proposed network structure can achieve higher accuracy with a smaller model size, then our prediction results are not only accurate, but also have more predictions with the same arithmetic power compared to other algorithms, i.e., have stronger real-time performance.

Firstly, to verify the optimization ability of the attention mechanism LRCA-Net proposed in this paper, two sets of control experiments were conducted with the network without adding the attention mechanism and the network with CBAM as the attention mechanism for object detection and lane detection, respectively, as shown in Figure 15.

The results of the two quantitative experiments for object detection and lane detection are given in Tables 4 and 5. Taking the backbone network Mobilenetv1 as an example, mAP and mPA are 90% and 90.3%, respectively, when no attention mechanism is added. Additionally, the proposed method in this paper mAP, mPA can reach 93.2% and 93.1%, which also improves 1.8% and 1.7%, respectively compared to CBAM, while the number of parameters gets reduced to some extent. It can be no matter for object detection or lane detection, the attention mechanism method proposed in this paper can effectively improve the performance of the algorithm, and certain optimization improvements compared to CBAM. Note that our LRCA-Net obtains higher accuracy while having less model complexity.

![Figure 14. Model loss function graph.](image)

![Figure 15. Comparison of attention modules (i.e., no attention module, CBAM, LRCA-Net) in terms of accuracy and network parameters. (a) mAP metric for object detection; (b) mPA metric for lane detection.](image)

**Table 4.** Comparison of no attention module, CBAM module, and LRCA-Net in the object detection model.

| Method               | Backbone   | AP     | mAP   | Parameters (Millions) |
|----------------------|------------|--------|-------|-----------------------|
|                      |            | Bus    | Bicycle | Motorbike | Person | Car       |            |
| No attention mechanism | Mobilenetv1 | 90.3   | 92.8   | 88.5 | 88.2 | 90.2 | 90 | 12.8 |
|                      | Mobilenetv2 | 89.7   | 91.7   | 91.1 | 89.1 | 90.6 | 89 | 10.9 |
|                      | Mobilenetv3 | 89.4   | 90.2   | 92.3 | 88.4 | 89.7 | 90 | 11.8 |
| CBAM                 | Mobilenetv1 | 92     | 91     | 88    | 90 | 90.2 | 91.4 | 13 |
|                      | Mobilenetv2 | 93.6   | 91.7   | 92.2 | 89.2 | 87.3 | 90.8 | 11.1 |
|                      | Mobilenetv3 | 92.5   | 92.3   | 90    | 90.1 | 91.7 | 92 | 12 |
| LRCA-Net             | Mobilenetv1 | 95.4   | 94.7   | 92.6 | 92.1 | 90.2 | 93.2 | 12.8 |
|                      | Mobilenetv2 | 94.3   | 93.2   | 93.4 | 91.7 | 87.4 | 92 | 10.9 |
|                      | Mobilenetv3 | 94.5   | 92.6   | 92.7 | 91.1 | 89.1 | 91.8 | 11.8 |

**Table 5.** Comparison of no attention module, CBAM module, and LRCA-Net in the lane detection model.

| Method               | Backbone   | PA     | mPA   | Parameters (Millions) |
|----------------------|------------|--------|-------|-----------------------|
|                      |            | Lane   | Background | mPA | Parameters (Millions) |
| No attention mechanism | Mobilenetv1 | 92.4  | 88.2   | 90.3 | 12.3 |
|                      | Mobilenetv2 | 91.7  | 87.3   | 89.5 | 10.5 |
|                      | Mobilenetv3 | 92.8  | 87.4   | 90.1 | 11.5 |
Table 5. Cont.

| Method       | Backbone     | PA Lane | mPA | Parameters (Millions) |
|--------------|--------------|---------|-----|-----------------------|
| CBAM         | MobileNetv1  | 93.3    | 91.4| 12.6                  |
|              | MobileNetv2  | 91.5    | 90.8| 10.8                  |
|              | MobileNetv3  | 93.7    | 92.1| 11.8                  |
| LRCA-Net     | MobileNetv1  | 95.5    | 93.1| 12.3                  |
|              | MobileNetv2  | 92.9    | 92   | 10.5                  |
|              | MobileNetv3  | 95.1    | 92.7| 11.5                  |

Table 6. Comparison of the VOC on the Pascal VOC 2007 and 2012 testing set, where bold numbers indicate the highest mAP in the column.

| Method            | Input Size | Backbone   | mAP  | Parameters (Millions) | FPS |
|-------------------|------------|------------|------|-----------------------|-----|
| SSD               | 300 × 300  | VGG16      | 82.4 | 26.3                  | 15  |
| Faster            |            |            |      |                       |     |
| Faster-RCNN-VGG16 | 600 × 600  | VGG16      | 78.9 | 136.8                 | 3   |
| Faster-R-CNN-ResNet50 | 600 × 600 | ResNet50   | 84.1 | 28.4                  | 13  |
| YOLOX-Tiny        | 640 × 640  | Modified CSP | 85.2 | 5.1               | 72  |
| YOLOX-S           | 640 × 640  | Modified CSP | 87.7 | 8.9               | 41  |
| YOLOX-M           | 640 × 640  | Modified CSP | 89.3 | 25.3              | 14  |
| YOLOX-L           | 640 × 640  | Modified CSP | 92.6 | 54.2              | 6   |
| YOLOX-X           | 640 × 640  | Modified CSP | 93.7 | 99.1              | 3   |
| YOLOv4            | 416 × 416  | CSPDark-53 | 90.7 | 64.4              | 5   |
| EfficientDet-D0   | 512 × 512  | Efficient-B0 | 76.9 | 3.9             | 96  |
| EfficientDet-D1   | 640 × 640  | Efficient-B1 | 83.3 | 6.6            | 70  |
| EfficientDet-D2   | 768 × 768  | Efficient-B2 | 89.5 | 8.1            | 42  |
| EfficientDet-D3   | 896 × 896  | Efficient-B3 | 90.9 | 12            | 30  |
| EfficientDet-D4   | 1024 × 1024| Efficient-B4  | 91.5 | 20.7          | 18  |
| EfficientDet-D5   | 1280 × 1280| Efficient-B5  | 92.8 | 33.7          | 11  |
| Our Approach-1    | 416 × 416  | MobileNetv1 | 93.2 | 12.8            | 29  |
| Our Approach-2    | 416 × 416  | MobileNetv2 | 92  | 10.9            | 34  |
| Our Approach-3    | 416 × 416  | MobileNetv3 | 91.8 | 11.8          | 32  |

To verify the actual effectiveness of our model, the actual detection results of the proposed model in this paper are given in Figure 17. We acquired real-time road pictures of the road environment in Daegu, South Korea, through an in-vehicle camera, and most people, cars, motorcycles, and bicycles are detected with sufficient accuracy and few missed detections. Only one detection frame is output for each target, and the detection capability is good for both near and far objects, which further validates the effectiveness of our algorithm.

Tables 7 and 8 give the comparison of quantitative experimental results using the same lane dataset and model training method. Here, the lane detection algorithm proposed in this paper is compared with SegNet [39], PSPNet [40], DeepLabv3+ [41], and U-Net. Our method is shown to achieve 96.4%, 94.8%, and 96.1% mPA depending on the backbone network. as shown in Figure 18, and compared to other models, our method achieves the highest accuracy rate with a much-reduced model size. Compared with PSPNet-ResNet50, DeepLabv3+, and U-Net-ResNet50, which are high-accuracy models, it not only improves the mPA by 2.9%, 4%, and 2.6%, respectively, but also reduces the volume by 73.4%, 70%, and 71.9%, and it can be seen by the FPS that the processing speed of our proposed method is 3 times faster than these three high-accuracy models. This is due to our use of a lightweight backbone network, which greatly reduces the network size, and the addition of LRCA-Net in the network feature fusion part to improve the model performance. As can
be seen in Table 8, our method produced the IoU 93.7% for lane lines and produces higher quality compared to other methods.

Figure 16. Model Size vs. mAP. Details are in Table 6.

Figure 17. Object detection results of the proposed model in real situations. (a) detection results of various vehicles on the road, (b) detection results of pedestrians and bicycles on the road, and (c) detection results of motorcycles and surrounding vehicles while the vehicle is moving.
Table 7. Comparison of the quantitative experimental results on the highway driving testing set, where bold numbers indicate the highest mPA in the column.

| Method           | Input Size | Backbone   | mPA   | Parameters (Millions) | FPS |
|------------------|------------|------------|-------|-----------------------|-----|
| SegNet-VGG16     | 416 × 416  | VGG16      | 79.7  | 11.6                  | 32  |
| SegNet-ResNet50  | 416 × 416  | ResNet50   | 83.1  | 14.9                  | 24  |
| PSPNet-ResNet50  | 473 × 473  | ResNet50   | 93.5  | 46.8                  | 8   |
| PSPNet-MobileNetv2 | 473 × 473 | MobileNetv2 | 90.2  | 2.4                   | 150 |
| DeepLabv3+      | 512 × 512  | Xception   | 92.4  | 41.3                  | 9   |
| DeepLabv3+-MobileNetv2 | 512 × 512 | MobileNetv2 | 89.9  | 2.8                   | 130 |
| UNet-VGG16      | 512 × 512  | VGG16      | 91.6  | 24.9                  | 15  |
| UNet-ResNet50   | 512 × 512  | ResNet50   | 93.8  | 44                    | 8   |
| Our Approach-1  | 512 × 512  | MobileNetv1 | 96.4  | 12.3                  | 30  |
| Our Approach-2  | 512 × 512  | MobileNetv2 | 94.8  | 10.5                  | 35  |
| Our Approach-3  | 512 × 512  | MobileNetv3 | 96.1  | 11.5                  | 33  |

Table 8. Segmentation results of the multi-task model, where bold numbers indicate the highest IoU in each column.

| Method           | Lane   | Fence  | Construction | Traffic Sign | Car   | Truck  | Vegetation | mIoU |
|------------------|--------|--------|--------------|--------------|-------|--------|------------|------|
| SegNet-VGG16     | 62.2   | 71.2   | 83           | 69.7         | 79    | 84.6   | 91.4       | 77.3 |
| SegNet-ResNet50  | 73.4   | 70.5   | 89.2         | 68.1         | 82.9  | 89.4   | 92.8       | 80.9 |
| PSPNet-ResNet50  | 89.4   | 93.6   | 84.3         | 95.1         | 90.7  | 94.5   | 90.8       | 91.2 |
| PSPNet-MobileNetv2 | 87.2  | 93.7   | 85.6         | 90           | 89.5  | 90.4   | 82.4       | 88.4 |
| DeepLabv3+      | 86.1   | 92.1   | 81.6         | 84.9         | 86.4  | 92.1   | 86.5       | 87.1 |
| DeepLabv3+-MobileNetv2 | 86.5 | 91.4   | 79.5         | 82.1         | 83.5  | 89.2   | 94.7       | 86.7 |
| UNet-VGG16      | 87.3   | 88.5   | 90.7         | 88.1         | 84.4  | 89     | 94.3       | 88.9 |
| UNet-ResNet50   | 89.6   | 89.2   | 91.3         | 88.7         | 89.7  | 90.5   | 94.1       | 91.5 |
| Our Approach-1  | 93.7   | 94.3   | 88.4         | 90.4         | 93.8  | 95     | 97.5       | 93.3 |
| Our Approach-2  | 91.5   | 92.8   | 87.1         | 92.7         | 91.9  | 92.6   | 98.9       | 92.5 |
| Our Approach-3  | 92.8   | 94.4   | 91           | 91.7         | 93.5  | 94.3   | 83.5       | 91.6 |

Figure 18. Model Size vs. mPA. Details are in Table 7.
The visual data of different methods are shown in Figure 19. The visual display results show that our model still performs well when dealing with variable road types, complex environmental backgrounds, and variable weather, which confirms that our model is robust.

![Figure 19](image.png)

**Figure 19.** Example results compared with other methods. (a) Original images; (b) DeepLabv3+; (c) PSPNet; (d) U-Net; (e) Our approach.

5. Conclusions

In this paper, we propose a novel lightweight detection system that has two different detection routes combined with an improved attention mechanism based on the same
backbone network divided into object detection and lane detection applied in a safe driving assistance system. Firstly, to improve the detection accuracy, an attention mechanism module is used to capture the information of cross interactions efficiently while greatly reducing the number of parameters. Secondly, the YOLOv4 backbone network is replaced by the lightweight network MobileNet, and the ordinary convolution in the feature fusion network is modified to a depthwise separable convolutional layer, which is combined with the network attention mechanism to make the network more efficient. Third, using the feature layer extracted by the backbone network and U-Net’s extended path method as a decoder, the local perceptual field can be increased, and multi-scale information can be collected without reducing the dimensionality. Additionally, the features can be further enhanced by inserting an attention mechanism in the feature layer fusion process. Such an approach can effectively utilize the dataset and improve the segmentation accuracy of the network.

The proposed algorithm in this paper was evaluated on the object detection dataset PASCAL VOC and highway driving dataset. mAP and mIoU reached 93.2% and 93.3%, respectively, achieving high performance compared to other methods.

Author Contributions: Conceptualization, H.L. and S.S.; methodology, H.L.; software, H.L.; writing—original draft preparation, H.L.; writing—review and editing, S.S.; visualization, H.L.; supervision, S.S.; project administration, S.S.; funding acquisition, S.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2016R1D1A1B02011625).

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. Viola, P.; Jones, M. Rapid object detection using a boosted cascade of simple features. In Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Kauai, HI, USA, 8–14 December 2001; Volume 1.

2. Yoo, H.; Yang, U.; Sohn, K. Gradient-enhancing, conversion for illumination-robust lane detection. IEEE Trans. Intell. Transp. Syst. 2013, 14, 1083–1094. [CrossRef]

3. Küçükmansıa, A.; Tarım, G.; Urhan, O. Real-time. illumination and shadow invariant lane detection on mobile platform. J. Real-Time Imaging Processing 2019, 16, 1781–1794. [CrossRef]

4. Kortli, Y.; Marzougui, M.; Bouallegue, B.; Bose, J.S.C.; Rodrigues, P.; Atri, M. A novel illumination-invariant lane detection system. In Proceedings of the 2017 2nd International Conference on Anti-Cyber Crimes (ICACC), Abha, Saudi Arabia, 26–27 March 2017; pp. 166–171.

5. Liang, H.; Seo, S. Lane Model Extraction Based on Combination of Color and Edge Information from Car Black-box Images. J. Korean Soc. Surv. Geod. Photogramm. Cartogr. 2021, 39, 1–11.

6. Zhou, S.; Jiang, Y.; Xi, J.; Gong, J.; Xiong, G.; Chen, H. A novel lane detection based on geometrical model and gabor filter. In Proceedings of the 2010 IEEE Intelligent Vehicles Symposium, La Jolla, CA, USA, 21–24 June 2010; pp. 59–64.

7. Joshy, N.; Jose, D. Improved detection and tracking of lane marking using hough transform. IJCSMC 2014, 3, 507–513.

8. Jung, C.R.; Kelber, C.R. Lane following and lane departure using a linear-parabolic model. Image Vis. Comput. 2005, 23, 1192–1202. [CrossRef]

9. Niu, J.; Lu, J.; Xu, M.; Lv, P.; Zhao, X. Robust lane detection using two-stage feature extraction with curve fitting. Pattern Recognit. 2016, 59, 225–233. [CrossRef]

10. Chen, W.; Wang, W.; Wang, K.; Li, Z.; Li, H.; Liu, S. Lane departure warning systems and lane line detection methods based on image processing and semantic segmentation: A review. J. Traffic Transp. Eng. 2020, 7, 748–774. [CrossRef]

11. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.

12. Uijlings, J.R.; Van De Sande, K.E.; Gevers, T.; Smeulders, A.W. Selective search for object recognition. Int. J. Comput. Vis. 2013, 104, 154–171. [CrossRef]
