Hindawi
Journal of Advanced Transportation
Volume 2022, Article ID 5134437, 13 pages
https://doi.org/10.1155/2022/5134437

WILEY | Hindawi

Research Article

Efficient Lane Detection Technique Based on Lightweight Attention Deep Neural Network

Zhiting Yao and Xiyuan Chen

Key Laboratory of Micro-Inertial Instrument and Advanced Navigation Technology, Ministry of Education, School of Instrument Science and Engineering, Southeast University, Nanjing 210096, China

Correspondence should be addressed to Xiyuan Chen; chxiyuan@seu.edu.cn

Received 16 November 2021; Revised 15 January 2022; Accepted 16 February 2022; Published 9 March 2022

Academic Editor: Carlos Guindel

For self-driving vehicles, detecting lane lines in changeable scenarios is a fundamental yet challenging task. The rise of deep learning in recent years has contributed to the thriving of autonomous driving. However, existing methods of lane detection based on deep learning have high requirements on computing environment, so their applicability is further restricted. This paper proposed an improved attention deep neural network (DNN), a lightweight semantic segmentation architecture catering for efficient computation in low memory, which contains two branches worked in different resolution. The proposed network integrates fine details captured by local interaction of pixels at high resolution into global contexts at low resolution, computing dense feature maps for prediction task. Based on the attributes of disparate feature resolution characteristics, different attention mechanisms are adopted to guide the network to effectively exploit the model parameters. The proposed network achieves comparable results with state-of-the-art methods on two popular lane detection benchmarks (TuSimple and CULane), with faster calculation efficiency at 259 frames-per-second (FPS) on CULane dataset, and the total number of model parameters only requires 1.57 M. This study provides a practical and meaningful reference for the application of lane detection in memory constrained devices.

1. Introduction

Lane detection is a research hotspot in autonomous driving, and it is a key technology of Advanced Driver Assistance System (ADAS) [1]. The forward-looking camera mounted behind the front windshield is used to collect the surrounding driving environment. The vision-based processing algorithm is embedded in the vehicle to detect lane lines from captured video clips. Then the results of lane detection will be applied to subsequent tasks including lane keeping [2], trajectory planning, and behavior prediction. Therefore, lane detection is an integral part of automatic driving and ADAS.

To extract lane markings from video clips, current studies have mainly investigated two key routes: traditional vision methods and deep learning methods. To be specific, traditional vision-based lane detection algorithm mainly includes two steps: extraction of hand-crafted features and fitting geometrical curves. Researchers distinguish lanes from the prior characteristics including lane shape, color feature, and edge texture information [3]. Then they use polynomial curves or splines to approach lane boundaries model with distinctive features. This traditional vision-based method can obtain satisfactory performance under the condition of clear lane markings and board vision. However, lane detection scenarios are subject to changes in illumination, weather conditions, traffic congestion, and other factors, which varies during driving and poses great challenges to lane estimation. These challenges are inevitable but can hamper the detection accuracy, especially when using traditional methods. Thus, we need a more robust method to meet high precision requirements of lane detection in challenging environments.

Deep learning, especially the convolutional neural network (CNN) [4], has rich feature representation capabilities, which greatly boosts the performance of computer vision

(CV) systems [5]. Recent studies on lane detection have shown that methods based on deep learning can deliver strikingly better results than traditional methods that depend on hand-crafted cues. Therefore, deep learning methods have become the preferred option in this field. Compared with traditional algorithms that highly rely on vision cues in specific environments, deep learning-based methods improve the network's scene perception ability by continuously optimizing neural network parameters, which attain higher robustness and applicability. Consequently, lane detection methods based on semantic segmentation and instance segmentation [6–9] have attracted extensive attention in recent years. However, some of these recent approaches have tried to directly adopt a classic segmentation network or its variants to segment lane markings. The results, although very encouraging, appear coarse in detail. The primary reasons come from two aspects: (1) In terms of accuracy, the segmentation of the boundary is less precise, especially when the lines are in distance and suffering occlusion. (2) In terms of computing resources, excessive computation and parameter overheads lead to large memory usage and deficient real-time performance, which limits the practicability of algorithm. In this paper, our motivation is to design an improved neural network architecture to compensate for the aforementioned dilemmas and execute a trade-off between high accuracy and low memory consumption. To achieve this goal, we designed a network framework with a two-branched structure. In this framework, we first utilize a lightweight downsampling module to squeeze the spatial dimension of the input feature map and forward them into two branches. One branch named global context embedding (GCE) focuses on capturing global information that can be used to deduce heavily occluded and blurred lane markings. Another branch explicit boundary regression (EBR) tries to exploit spatial attention mechanism (SAM) to aggregate boundary information at different locations, and a supervisory operation by label's edge information is attached at the end of EBR. In particular, SAM is integrated into the EBR module for better boundary regression, and channel attention mechanism (CAM) is placed behind the output of GCE encoder, so that channels with target objects can be assigned higher weights. Inspired by the effectiveness of MobileNet [10] series articles, we replace regular convolution layers with bottleneck units for network deepening operation, which contributes to the reduction of network parameters.

The main contributions of this study can be summarized as follows.

(1) This paper proposes a lightweight DNN framework to simultaneously address precision performance and memory overloads issues, which is better suited for the lane detection task.

(2) This paper designs an EBR module with SAM and auxiliary edge supervision to reinforce the consistency of semantic boundary. The lateral experimental results indicate these modules significantly improve the precision of lane boundaries.

(3) The experimental part elaborates on the details of the ablation study and compares the segmentation results on TuSimple and CULane datasets with other methods. Results show that the proposed model attains faster inference time with competitive performance compared with state of the art. And the robustness of the algorithm can tackle lane detection in dark, dazzled, and blurred environment.

The remainder of this paper proceeds organized as follows. Section 2 reviews the previous researches on lane detection. Section 3 introduces the proposed method. Section 4 demonstrates the experimental performance of the proposed method. Section 5 summarizes this article.

2. Related Works

The present work relies heavily on prior efforts in lane detection and attention mechanism areas.

2.1. Lane Detection. Before the rise of deep learning, methods on lane detection generally focus on feature extraction, model fitting, and lane tracking. Researchers use the color, boundary, and texture features of the road to realize lane detection and tracking. We conclude a general block diagram of the traditional lane detection methods in Figure 1. For more algorithm types and implementation details of image preprocessing, region of interest (ROI) selection, lane modeling, lane detection, and lane tracking in traditional methods, please refer to [11, 12].

Methods based on deep learning have gradually become the dominant algorithm in vision tasks relying on their powerful representation capabilities. In order to handle the complex situations in lane detection, [6] designed a hybrid deep architecture by combining RNN with CNN. Input consecutive frames into CNN for feature extraction and then exploit RNN to further learn extracted features and make lane prediction. In [7], they design a network named RS-Lane, which adds split attention and self-attention distillation on the basis of LaneNet to increase the reasoning ability and robustness of the proposed method. This work can detect lane lines without number limits. In [13], the authors improve you only look once (YOLO) object detector and realize the detection of yellow lane lines by means of object detection. Pan et al. [8] proposed a spatial CNN that aggregates the features of each pixel through slice-by-slice convolution in a layer, resulting in top-1 performance in the CVPR'17 TuSimple benchmark. However, to take advantage of spatial information sufficiently, this method gathers both horizontal and vertical information by shifting the sliced feature recurrently, which is conceivably time-consuming. Compared with SCNN, Tu et al. [9] employ the same spatial information utilization strategy, but on this basis, the algorithm is simplified by changing the information shifting strides. Qin et al. [14] realize fast lane detection by gridding the image and searching the lane grids row-by-row and column-by-column. Even though this method can benefit from the fast speed delivered by grid downsampling, it also cannot prevent reduced accuracy from low-resolution sparse

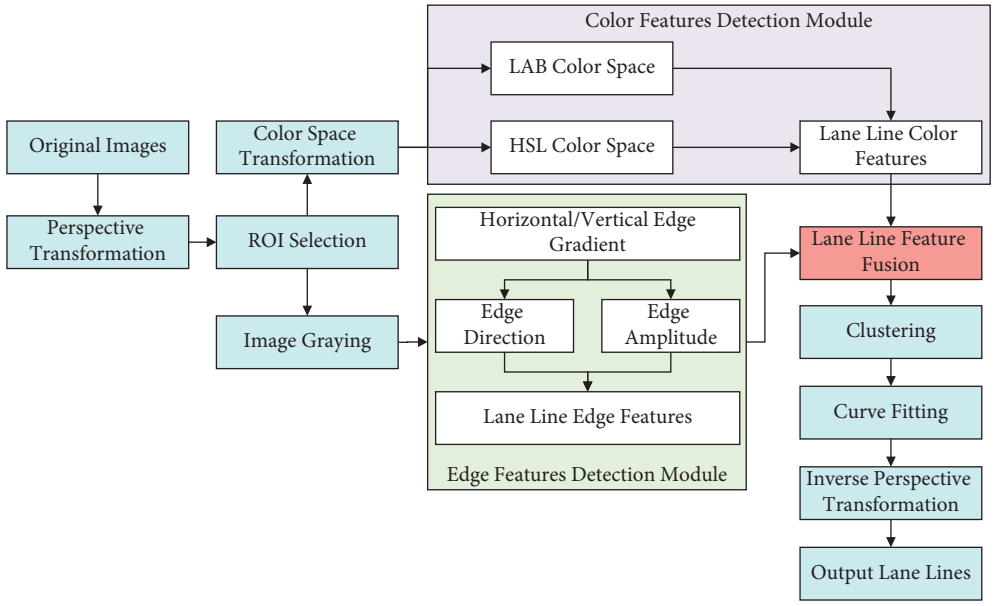

Figure 1: Framework of traditional vision-based lane detection technique.

feature map. With approach in [15], the authors proposed PolyLaneNet, which converts the lane estimation task into conjecturing polynomials that represent each lane in input image. This approach obtains higher real-time efficiency; however, the accuracy relies on the position of lane line starting points, so the performance drops significantly when the lanes suffer severe occlusion. Typical framework of deep learning-based lane detection is shown in Figure 2.

2.2. Attention Mechanism. Attention mechanism originated from the research of natural language processing (NLP) and was gradually applied to the field of CV recently. This mechanism can selectively focus on important features and suppress irrelevant features, thus improving the performance of DNN. SENet [16] proposes a "Squeeze-and-Excitation" block to model interdependencies between channels. By recalibrating the channelwise feature responses, the network produces significant performance improvements at negligible overheads. Different from SENet, approaches of [5, 17] both exploit semantic interdependencies in spatial and channel dimensions. In order to make best use of two complementary attention outputs, [5] propagates the channelwise output to spatialwise attention submodule in sequential arrangement, while [17] arranges two modules in parallel and sums the outputs to further improve feature representation. In this work, we utilize channel attention and spatial attention in different branches to enhance network representation power. The channel attention and spatial attention are similar to DANet [17] and CBAM [5], respectively. In [18], a self-attention distillation (SAD) approach is proposed to improve the representation learning of CNN-based lane detection models, which is a flexible plug-and-play module. For more articles based on deep learning to achieve lane line detection, please refer to [19].

More close to our work, [20] introduces channel attention module and self-attention module in parallel to obtain global contexts and channel dependencies of feature maps. However, successive dilated convolutions introduced in the subsampling process and large rectangular multiplication under the self-attention mechanism will incur expensive computations and memory burdens to the network. In our network, we utilize a lightweight downsampling module to gather low-stage feature maps and exploit spatial and channel attention based on a simple yet efficient architecture.

3. Methods

In this section, we first present a general architecture of our designed model and then elaborate the inner blocks used to capture the global context information and spatial edge information. Finally, we demonstrate how to aggregate the information for further enhancement of feature representation.

3.1. Architecture Design. Figure 3 is a pictorial description of our proposed network architecture. It is obvious that this architecture consists of four components and splits into two branches. The above branch encodes increasingly abstract feature representation with long-range context in deeper encoder output, while the bottom branch reserves spatial details in low-level high-resolution feature maps. Subsequently, we take the superiorities of two branches into consideration, so that elementwise summation is employed to fuse features. The following describes the details of each module.

3.2. Lightweight Downsampling. In lightweight downsampling module, we consider different strategies to subsample original input. The large kernel size (with kernel size = 7) is adopted in the first subsampling process to

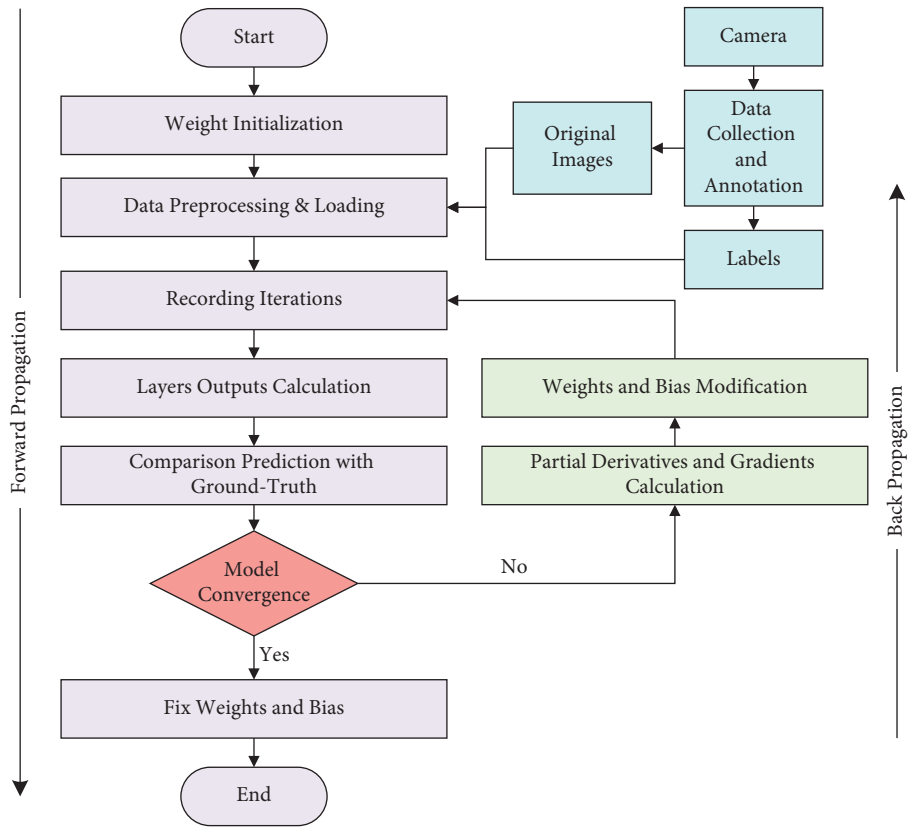

Figure 2: Framework of deep learning-based lane detection technique.

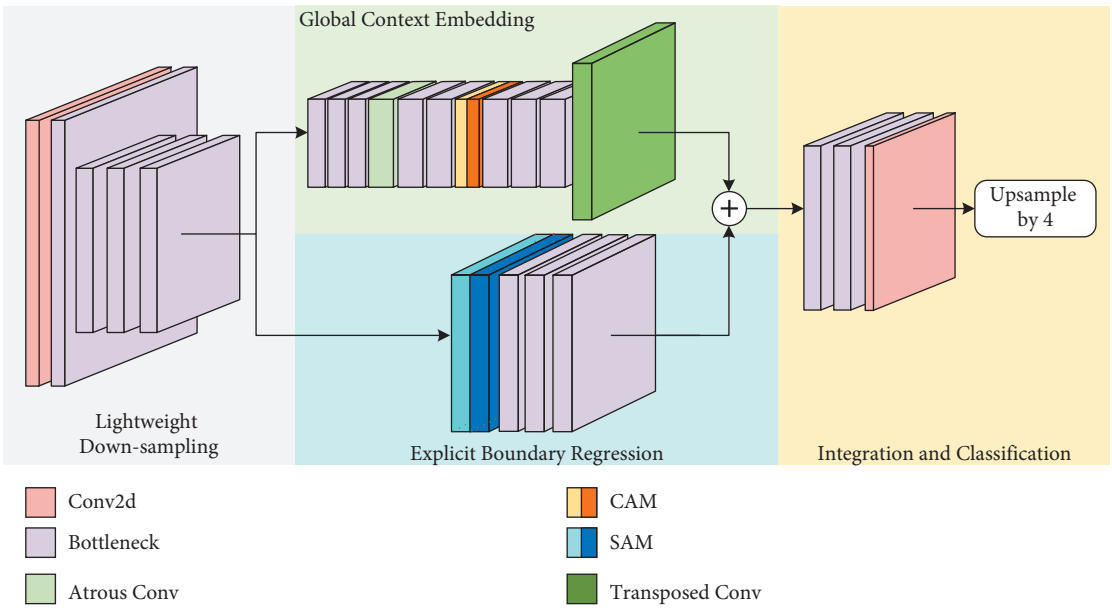

Figure 3: The two branches' architecture of proposed method and detailed structure diagram of bottleneck, CAM, and SAM are given afterwards. The symbol "⊕" represents elementwise summation (best viewed in color).

enhance dense connections between feature maps and per-pixel classifiers, strengthening the robustness to local disturbances and allowing the classifiers to handle variant distortion from cameras. Here we emphasize that subsampling with larger kernel size filter does not impose additional parameter burdens on the network, while reducing the parameters to some extent. The bottleneck layer unit [10] consists of two point convolutions and a group convolution with stride 2, which can further subsample the input and generate abstract feature maps in compact channel

dimension. Furthermore, we adopt the bottleneck with residual shortcut as a substitute for conventional convolution to reduce network overloads, as shown in Figure 4. These two feature extraction strategies are combined in the lightweight downsampling module to realize the aggregation of deep abstract features while reducing the number of parameters.

3.3. Global Context Embedding. The GCE module involves two parts laid out sequentially to encode long-range contexts. The first part is convolutional striding operation and dilated convolution, which capacitate high stage features to obtain richer global information. The other is the introduction of CAM. Since convolutional operation extracts informative features by blending cross-channel information with simple summation, the importance of different channels is ignored. However, we expect the network to selectively emphasize channels that contain lane line semantic features and restrain irrelevant ones. Channel attention provides a means of recalibration channelwise feature responses according to interchannel correlations, which stimulates the network sensitivity to crucial and informative features. CAM calculates the specific weight of each channel in the unit of feature map, so that the channels that cover the target feature receive more attention. Therefore, the precision of the network can be significantly improved while slightly increasing the computation, which is consistent with the theme of our proposed method.

Concretely, the second strategy in 3.2 is first introduced to subsample the input; then stacking the dilated convolutions is aimed at expanding receptive field of filters. Next, CAM is appended to aggregate the intraclass consistency and enhance robustness to local disturbance; the structure is illustrated in Figure 5(a). Here we describe the operation of CAM in detail below. Suppose that $X \in R^{C\times H\times W}$ is the output of the previous layer. Firstly, we reshape it to $X \in R^{C\times N}$, where N represents total pixel numbers. Then, we conduct a matrix multiplication between $X \in R^{C\times N}$ and transposed $X^T \in R^{N\times C}$.

$$Y = XX^T \in R^{C\times C}, \tag{1}$$

Next, the above multiplied resulting matrix is forwarded into a softmax layer to normalize the interchannel dependencies $Y \in R^{C\times C}$ between any two channel maps. Finally, we multiply the weighted channel attention map y_{ji} with corresponding channel X_i and perform summation with origin channel X_j to obtain the final feature maps O as follows:

$$\begin{aligned} y_{ji} &= \frac{\exp\left(X_i \cdot X_j\right)}{\sum_{i=1}^{C} \exp\left(X_i \cdot X_j\right)}, \\ O_j &= \varphi \sum_{i=1}^{C} \left(y_{ji} X_i\right) + X_j, \end{aligned} \tag{2}$$

y_{ji} denotes the influence exerted by channel i_{th} on j_{th}, where φ is a learnable parameter which is initialized as 0 and gradually learns to assign more weight.

3.4. Explicit Boundary Regression. The lane lines in the distance are inconspicuous and incomplete due to the influence of light and occlusion. If we merely perform ordinary convolution operation in high-resolution input to generate local feature map, the discriminability for these indistinct lane targets will be covered by other salient objects, resulting in misclassification and misdetection of semantic segmentation. To remedy the above mentioned issue, we first exert SAM to enhance the feature discriminant ability, which redistributes weight based on the interspatial relationship of features in a global view, as shown in Figure 5(b). We improve the SAM proposed in [5], which reallocates the weight of each pixel in feature map and assigns the pixel containing target cue to a higher weight, thus improving the representation ability of indistinct boundary features. Given the spatialwise refined features, we introduce an edge supervision to guide the network to learn the boundary characteristics. This supervision acts as an auxiliary boundary segmentation task, enabling the network to achieve EBR. Next, we expound the process of SAM and edge supervision.

As illustrated in Figure 5(b), given a local feature $I \in R^{C\times H\times W}$, we first feed it into point convolution, max-pooling, and average-pooling operations along the channel axis to generate feature descriptors I^s_{conv}, I^s_{max}, and I^s_{avg}, respectively, where $\left\{I^s_{conv}, I^s_{max}, I^s_{avg}\right\} \in R^{1\times H\times W}$. Then we concatenate the above three descriptors and forward them into a convolution layer with large kernel size of 7. After that we apply a sigmoid layer to calculate the 2D spatial feature map $F \in R^{H\times W}$. Finally, we perform an elementwise multiplication between I and feature map F to obtain the final output E as follows:

$$\begin{aligned} I^s_{conv} &= W_f I \\ I^s_{\max} &= MaxPool(I) \\ I^s_{avg} &= AvgPool(I) \\ E &= Sigmoid\left(W_h\left[I^s_{conv}; I^s_{\max}; I^s_{avg}\right]\right) \otimes I, \end{aligned} \tag{3}$$

where $W_f \in R^{1\times 1\times C\times 1}$ and $W_h \in R^{7\times 7\times 3\times 1}$ represent convolutional operation with kernel size of 1 and 7, respectively; $\otimes$ denotes elementwise multiplication.

To enhance the continuity and discriminability of the lane boundary, we employ Sobel edge extraction operator to filter the semantic labels and exploit them as supervision signal. It is worth noting that the lane lines occupy only a small proportion in the labels and the imbalance between background and foreground is detrimental to the segmentation performance. Thus, we employ focal loss [21], as an auxiliary loss function, to supervise the output of SAM; the equation is shown as follows:

$$FL(p_n) = -(1 - p_n)^{\gamma} \log(p_n), \tag{4}$$

where p_n means the probability of class n, $n \in \{1, 2, 3, \ldots, N\}$, N is the maximal number of labels, and γ is a modulating factor and is set to 2 here.

3.5. Integration and Classification. To embed rich semantic information into low-level features, we conduct transpose

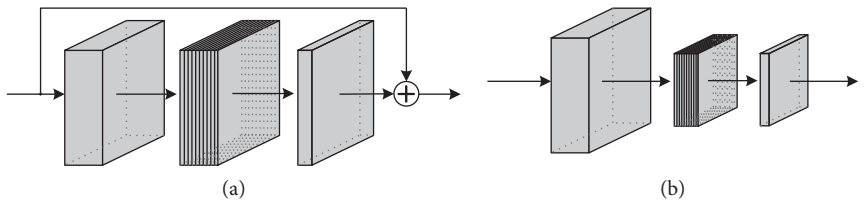

Figure 4: Structures of bottleneck layer unit with variant stride in group convolution. The first and the third convolutions in above structure are point convolution, while the second is group convolution. (a) Bottleneck with skip connection and stride of 1. The symbol "⊕" represents elementwise summation. (b) Bottleneck with stride of 2.

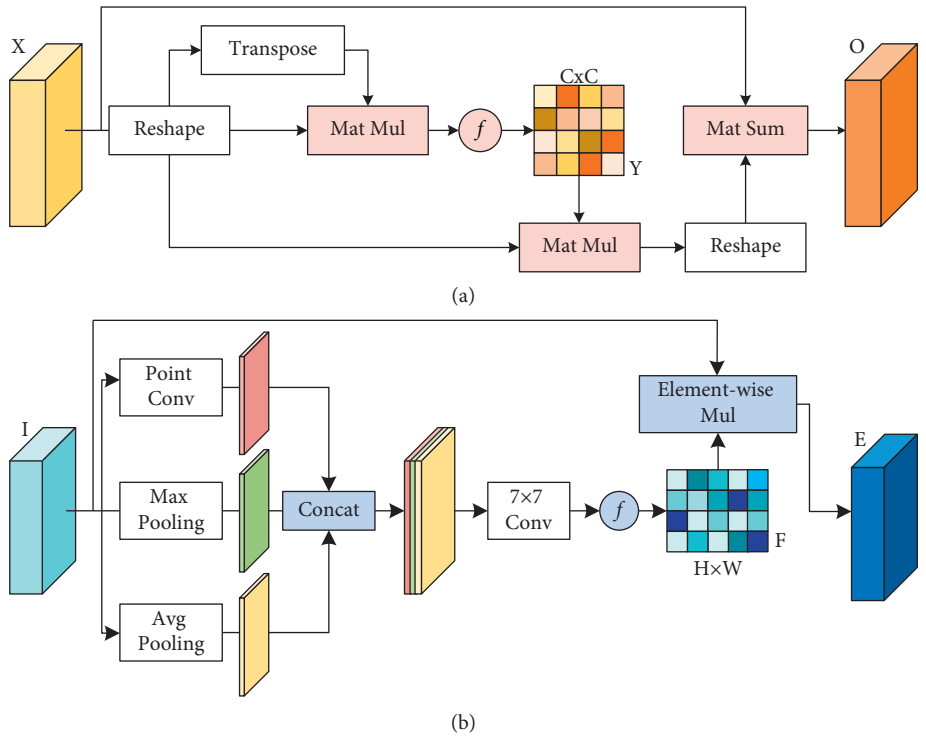

Figure 5: Schematic diagram of CAM (a) and SAM (b).

convolution with stride 2 to upsample the outputs of GCE and integrate them with the outputs of EBR by summation. Furthermore, from the perspective of feature fusion, the exploit of EBR can bridge the gap between low-level and high-level features as emphasized in [22]. Then two bottleneck layers with residual connection are followed to tightly aggregate features from aforementioned two branches. Finally, we convolve the fusion results and generate final prediction maps.

4. Experiment

We evaluate our network on two widely used lane detection benchmark datasets: TuSimple benchmark dataset [23] and CULane dataset [8]. We first introduce the datasets and report implementation details, then perform a series of ablation experiments, and present comparison results with other state-of-the-art approaches.

4.1. Datasets and Implementation Details

TuSimple: It is a well-known traffic lane detection benchmark which contains five annotated lane markings, involving 3626 images for training and 2782 images for testing. Because no validation set is given, we randomly split 368 images from train set as validation set, which are employed for preventing overfitting and validating the model during training.

CULane: The CULane dataset is a large scale challenging lane detection dataset which contains more than 55 hours of videos and extracts 133235 frames, involving 88880 frames for training, 9675 frames for validation, and 34680 frames for testing. This dataset comprises different conditions including urban, rural, and highway, which consists of normal and night challenging scenarios.

4.1.1. Implementation Details

Training: Our experiments are executed on PyTorch deep learning framework. Considering the excessive resolution of inputs is computationally expensive in training, thus we first resize the original images to 360 × 640 for TuSimple and 288 × 800 for CULane. Then we train our network using stochastic gradient decent (SGD) with momentum 0.9 and weight decay 0.0001. The batch size is set to 8 here. Influenced by DeepLab [24] success in semantic segmentation, we employ the similar "poly" learning rate policy where the initial learning rate $4e^{-3}$ is multiplied by $(1 - iter/\max\_iter)^{power}$. We adopt cross-entropy loss to measure the similarity between prediction mask and ground truth. Besides, a weighted focal loss served as an auxiliary loss function for precise boundary regression, as shown in

$$Loss = L_{CE} + \omega L_{FL}, \tag{5}$$

where L_{CE} represents the segmentation loss calculated by cross-entropy, L_{FL} reveals boundary regression loss by focal loss, and ω is a weight factor. The network training will be terminated when the epoch number exceeds 100, and the model with lowest validation loss will be selected as the final testing model.

Metrics: The evaluation metrics used to qualify the approach's segmentation performance consists of mean intersection-over-union (mIoU), accuracy (Acc), false positive (FP), false negative (FN), true positive (TP), and F1-measure (F1). With respect to speed evaluation metrics, we adopt the FPS.

The Acc metric defined in TuSimple benchmark is shown below:

$$Acc = \sum_i \frac{C_i}{N_i}, \tag{6}$$

where L_{CE} denotes the number of points detected correctly, and L_{FL} represents the total number of the ground truth points in ω clip.

The F1 metric defined in CULane dataset is shown below:

$$\begin{aligned} F1 &= \frac{2(precision^{*}recall)}{precision + recall}, \\ precision &= \frac{TP}{TP + FP}, \\ recall &= \frac{TP}{TP + FN}, \end{aligned} \tag{7}$$

4.2. Ablation Study. In this section, we stepwise disintegrate our network to investigate the effect of each component in proposed method. In the next experiments, we conduct comparative experiments on inner structure of the proposed network architecture on TuSimple dataset and use the framework without attention module and edge supervision as the baseline for ablation experiments.

4.2.1. Attention Mechanism. We use CAM in the GCE module to capture long-range dependencies for better lane marking inference. And SAM are used to break the situation where the position weights are identical, which is complementary to the channel attention. To verify the performance of the aforementioned components, we explore four different implementation combinations in Table 1. It can be seen from the table that the integration of SAM and CAM has significantly improved the segmentation accuracy of the network, especially the CAM. The SAM is placed on the branch of high-resolution feature map to optimize local position details by increasing the pixel weights containing lane features, thus improving the detection accuracy in a small range. The CAM is integrated into the high-semantic feature map branch to optimize the global classification accuracy of the network by enhancing the channel weights containing lane cues, resulting in an obvious boost in detection accuracy. These two submodules are arranged in parallel combination at different branches, so that the dependencies modeled by two attention modules do not affect each other. Therefore, it can be seen from the last line in Table 1 that the precision of the combined model has been further improved.

4.2.2. Boundary Regression. In order to better identify and locate boundary features, we employ label-based edge detection results as the supervision signal to strengthen the network representation ability. Then we attach the edge supervision to the output of SAM. This improves the continuity of lane boundary and contributes explicitly to regression of lane lines. We first quantitatively test the balance of bilateral losses, which further analyze the correlation between significance of boundary regression and final segmentation results; various values {0.05, 0.1, 0.2, 0.5, 0.75} are used to weight the focal loss. It is apparent from Figure 6, with the same setting, that mloU reaches the peak at 70.60% with w of 0.2. The initial climbing trend of the line chart shows that moderate edge supervision helps the network efficiently extract lane lines, while excessive reliance will lead to a rapid decline in network performance. Therefore, we set the turning point as the final weight factor.

We visualize the effect of auxiliary edge supervision in Figure 7. The first column and the second column represent the original image and ground truth, respectively. The last two columns exhibit the detection results with and without edge supervision. From the experiment results, the fourth column obviously outperforms the third column at better refined edge details.

4.3. Performance Evaluation on TuSimple. We carry out comparative experiments on TuSimple with other state-of-the-art methods to further evaluate the performance of our approach. The methods include RESA [9], Ultra-Fast [14], PolyLaneNet [15], SAD [18], FastDraw [25], and LaneNet

Table 1: Detailed performance of proposed network with and without SAM and CAM.

Method	SAM	CAM	mIoU (%)
Baseline			66.41
Baseline	√		67.36 (+0.95)
Baseline		√	68.83 (+2.42)
Baseline	√	√	69.52 (+3.11)

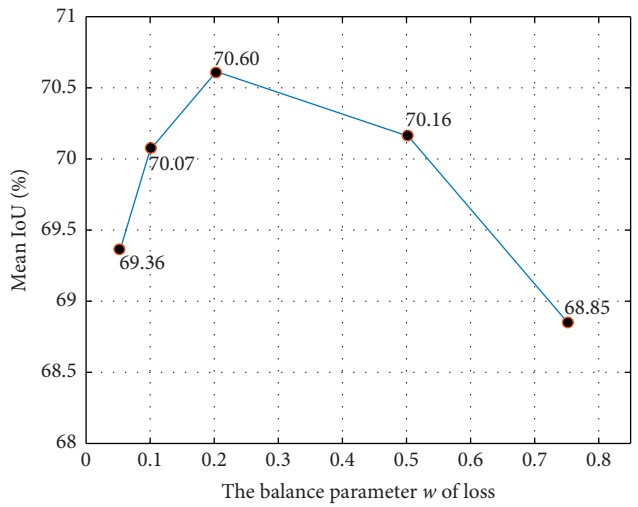

Figure 6: Performance of proposed network with different ω value.

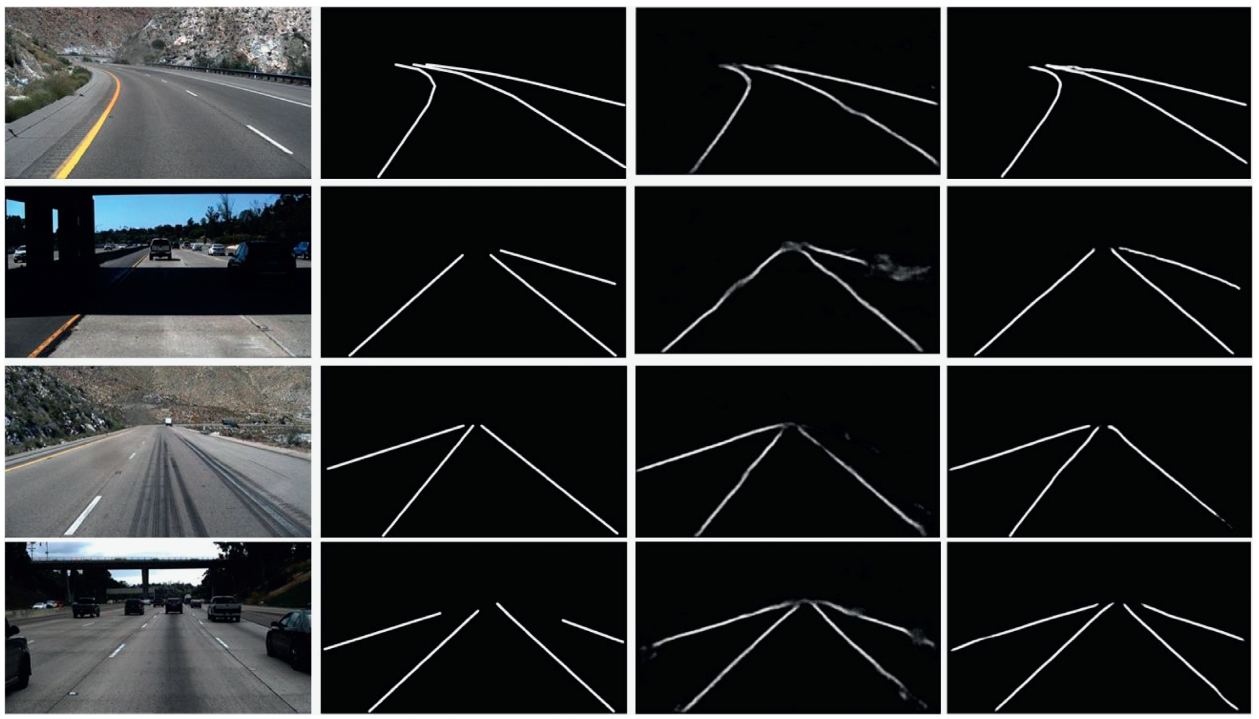

Figure 7: Lane prediction in white of proposed network with and without edge supervision. From left to right are the original pictures, labels, segmentation results without edge supervision, and segmentation results with edge supervision. The lane predicted with edge supervision is clearer and more continuous.

Table 2: Comparison with state-of-the-art methods on TuSimple dataset. Use PyTorch to test FPS and parameter at 3 * 360 * 640 resolution on NVIDIA RTX 2070s. Data with "*" mark is obtained from the original text.

Models	Accuracy	FP	FN	FPS	Parameter (M)
Res34-RESA	96.82	0.0363	0.0248	10	25.37
ENet-SAD	96.64	0.0602	0.0205	44	0.98
Res18-UFast	95.87	—	—	286	61.23
FastDraw	95.20	0.0760	0.0450	90*	—
LaneNet	93.38	0.0780	0.0224	—	—
PolyLaneNet	93.36	0.0942	0.0933	120	4.05
Proposed	95.11	0.0695	0.0490	244	1.57

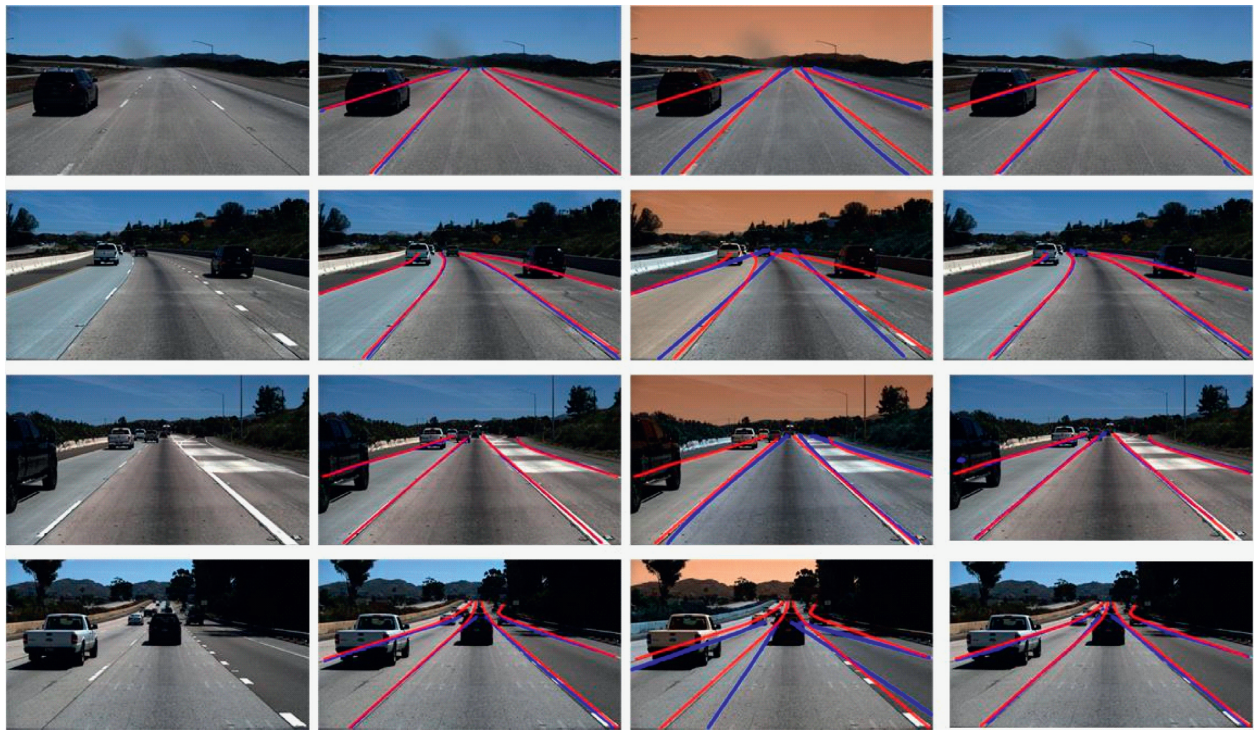

Figure 8: Visualization of lane detection results with comparative methods, the picture from left to right is the original pictures, results from RESA, results from PolyLaneNet, and results of proposed method. Red and blue lines denote ground truth and prediction lane, respectively.

[26]. The quantitative results are specifically reported in Table 2. The experimental results show that the proposed method outperforms the LaneNet and PolyLaneNet by 1.73% and 1.75% and achieves comparable performance with UFast. PolyLaneNet obtains the lane line detection results by predicting the polynomial parameters of the lane lines, while a slight deviation from the parameters will lead to a significant decrease in the accuracy of lane detection result. RESA gets excellent performance by recurrently convolving sliced feature maps vertically and horizontally to gather global information for each pixel. This approach introduces a large number of convolution operations, which significantly increases the number of parameters and inference time of the network. Figure 8 displays the intuitive comparison results. Because TuSimple's evaluation metrics allow the predicted points to bias from the true points within a certain threshold range, our quantitative accuracy index is only 1.75% higher than PolyLaneNet. However, it is apparent from the qualitative graph that our results are more consistent with real lines and the deviation value is smaller. In order to verify the lightweight structure and real-time performance of our model, further contrast experiments were carried out on inference time and parameters. We loop 100 times to calculate the run time of a single frame. The last two columns in Table 2 provide the results. The experimental results show that our network is 2.7×, 5.5×, and 24.4× faster than FastDraw, ENet-SAD, and RESA, respectively. In addition, the model parameters are only 1/16 and 1/39 of RESA and Res18-UFast. From the above comparison tests, although RESA and ENet-SAD achieve better results in terms of accuracy index, our method provides a more practical and comprehensive choice considering the high requirements for real-time performance in practical application scenarios and the limited memory of embedded devices.

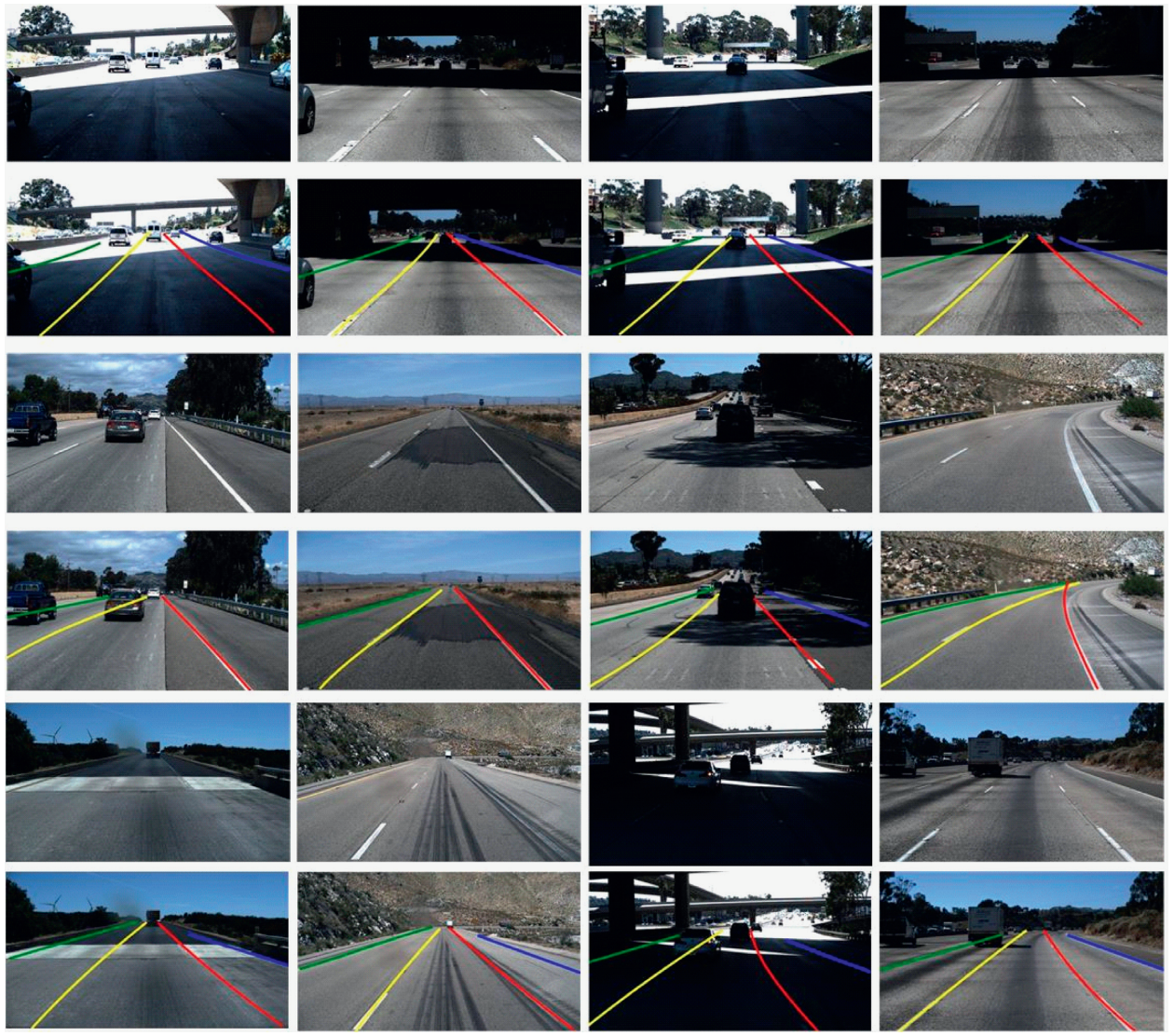

Figure 9: Robustness test for proposed method on TuSimple dataset. The odd lines display the original images, and the even lines present lane predictions overlapping on original images.

On the basis of completing the above ablation and comparative experiments, we also conduct further tests on the robustness of the algorithm. In the course of driving, the illumination of road surface often changes when being sheltered by shadows or passing under the viaduct. Furthermore, spurious marks on the road surface and significant changes in road color will also interfere with the lane line detection results. Therefore, the lanes detection algorithm needs to be robust enough to deal with the above situations. In Figure 9 we first display multiple road scenes with obvious challenges in road light, surface color, and spurious dirt traces. Then, the lane line clustering and polynomial fitting results are superimposed on the original map to get a more intuitive result displayed in the following rows. The experiment results show that the proposed algorithm can accurately identify the lanes, including scenarios with curves, road slope, and online.

4.4. Performance Evaluation on CULane. To verify the effectiveness and generality of the proposed method, we conduct comparative experiments on the CULane dataset. Several recent lane detection methods, including ENet-SAD [18], SCNN [8], FastDraw [25], Res18-Ultra [14], and Res18-VP [27], are used for comparison. Table 3 presents the results. Compared with TuSimple dataset, CULane dataset has a larger and richer training set and can better train the generalization performance of the model in different scenarios. It can be seen from Table 3 that the proposed method is comparable with Res18-VP and Res-Ultra in F1 metric and has outstanding performance in parameters, which denotes our method strikes a balance between accuracy, speed, and computation burdens. Compared with ENet-SAD, our model has 0.59 M more parameters. Considering that both models are lightweight network structures, the total network parameters are less than 2 M. Therefore, the influence of the 0.59 M parameter advantage on practical

Table 3: Comparison with state-of-the-art methods on CULane dataset with IoU threshold = 0.5. Use PyTorch to test FPS and parameters at 3 * 288 * 800 resolution on NVIDIA RTX 2070s. Data with "*" mark is obtained from the original text.

Category	Proportion	SCNN	ENet-SAD	Res18-VP	Res18-ultra	FastDraw	Proposed
Normal	27.74%	90.6	90.1	89.2	87.7	85.9	88.5
Crowded	23.39%	69.7	68.8	67.9	66.0	63.6	67.6
Night	20.27%	66.1	66.0	62.6	62.1	57.8	61.9
No line	11.73%	43.4	41.6	41.7	40.2	40.6	39.8
Shadow	2.68%	66.9	65.9	58.8	62.8	59.9	62.0
Arrow	2.57%	84.1	84.0	81.6	81.0	79.4	81.2
Dazzle light	1.40%	58.5	60.2	59.3	58.4	57.0	62.4
Curve	1.21%	64.4	65.7	60.8	57.9	65.2	56.9
Crossroad	9.00%	1990	1998	2919	1743	7013	2269
Total	—	71.6	70.8	69.1	68.4	—	68.8
FPS	—	8	48	42*	285	—	259
Parameter	—	20.72	0.98	16.07	61.23	—	1.57

Figure 10: Robustness test for proposed method on CULane test dataset. The odd lines display the original images, and the even lines present lane predictions overlapping on original images.

performance is not distinct. In terms of speed, the detection efficiency of proposed network is more than 5 times faster than that of ENet-SAD; this significant gap in real-time performance is more obvious in practical application, which can meet the detection speed requirements in various occasions. Then we select the challenging scenarios from the test set to intuitively verify the effectiveness of the proposed method. Figure 10 presents the visualizations.

5. Conclusion

In this paper, we propose an efficient lane detection method based on lightweight attention DNN, which is tailored for real-time lane detection task. Our method can effectually capture global context information to segment occluded lane lines and judge the classification of each pixel. Meanwhile, it retains high-resolution dense features to better conjecture

the inconspicuous lane boundaries. In order to generate semantically precise prediction maps and refine segmentation results along lane boundaries, we further incorporate attention and edge supervision mechanisms into the network. We evaluate effectiveness and generality of our proposed method on TuSimple and CULane datasets. Although the results show that state-of-the-art methods can obtain slightly higher accuracy, the model parameters and computational overheads far exceed our network. Thus, our proposed method strikes a balance between accuracy and computational costs. Extensive experiments demonstrate that our network can attain robust and effective performance under the challenging scenarios, which provides a reference for the implementation of lane detection in embedded devices. In the future, we will continue to explore how to improve the accuracy of the model and keep seeking a balance between accuracy and efficiency. In addition, we will deploy our algorithm on an embedded vehicle platform to guide subsequent obstacle avoidance and planning tasks.

Data Availability

The datasets are available from http://github.com/TuSimple/tusimple-benchmark and https://xingangpan.github.io/projects/CULane.html.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was partially supported by the Jiangsu Provincial Agricultural Science and Technology Independent Innovation Fund Project (no. CX(21)2025), National Natural Science Foundation of China (no. 61873064), and Suzhou Science and Technology Plan Project (no. SNG2020039).

References

[1] A. Ziebinski, R. Cupek, H. Erdogan, and S. Waechter, "A survey of ADAS technologies for the future perspective of sensor fusion," *Computational Collective Intelligence*, Springer, in *International Conference on Computational Collective Intelligence*, Springer, Cham, , pp. 135–146, 2016.

[2] J. Ni, J. Han, and F. Dong, "Multivehicle cooperative lane change control strategy for intelligent connected vehicle," *Journal of Advanced Transportation*, vol. 2020, no. 1, pp. 1–10, 2020.

[3] W Chen, W Wang, K Wang et al., "Lane departure warning systems and lane line detection methods based on image processing and semantic segmentation–a review," *Journal of Traffic and Transportation Engineering*, 2020.

[4] J Long, E Shelhamer, and T Darrell, "Fully convolutional networks for semantic segmentation," *Proceedings of the IEEE conference on computer vision and pattern recognition*, vol. 39, pp. 3431–3440, 2015.

[5] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: convolutional block attention module," *Computer Vision - ECCV 2018*, pp. 3–19, 2018.

[6] Q. Zou, H. Jiang, Q. Dai, Y. Yue, L. Chen, and Q. Wang, "Robust lane detection from continuous driving scenes using deep neural networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 1, pp. 41–54, 2020.

[7] R. Zhang, Y. Wu, W. Gou et al., "RS-lane: a robust lane detection method based on ResNeSt and self-attention distillation for challenging traffic situations," *Journal of Advanced Transportation*, vol. 7544355, p. 12, 2021 2021.

[8] X Pan, J Shi, P Luo et al., "Spatial as deep: spatial cnn for traffic scene understanding," *Thirty-Second AAAI Conference on Artificial Intelligence*, vol. 32, 2018.

[9] T Zheng, H Fang, Y Zhang et al., "RESA: recurrent feature-shift aggregator for lane detection," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, pp. 3547–3554, 2021.

[10] M Sandler, A Howard, M Zhu et al., "Mobilenetv2: inverted residuals and linear bottlenecks," *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4510–4520, 2018.

[11] P Sandipann, P. N Narote et al., "A review of recent advances in lane detection and departure warning system," *Pattern Recognition*, vol. vol 73, pp. 216–234, 2018.

[12] Y. Xing, C. Lv, L. Chen et al., "Advances in vision-based lane detection: algorithms, integration, assessment, and perspectives on ACP-based parallel vision," *IEEE/CAA Journal of Automatica Sinica*, vol. 5, no. 3, pp. 645–661, 2018.

[13] S. Hassan, T. Rahim, and S.-Y. Shin, "An improved deep convolutional neural network-based autonomous road inspection scheme using unmanned aerial vehicles," *Electronics*, vol. 10, no. 22, p. 2764, 2021.

[14] Z Qin, H Wang, and X Li, "Ultra fast structure-aware deep lane detection," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow*, Springer International Publishing, 2020, UK, 2020, Proceedings, Part XXIV 16.

[15] L Tabelini, R Berriel, T M Paixao et al., "Polylanenet: lane estimation via deep polynomial regression," in *25th International Conference on Pattern Recognition (ICPR)*, pp. 6150–6156, IEEE, 2020.

[16] J Hu, L Shen, and G Sun, "Squeeze-and-excitation networks," *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7132–7141, 2018.

[17] J Fu, J Liu, H Tian et al., "Dual attention network for scene segmentation," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3146–3154, 2019.

[18] Y Hou, Z Ma, C Liu et al., "Learning lightweight lane detection cnns by self attention distillation," *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1013–1021, 2019.

[19] J. Tang, S. Li, and P. Liu, "A review of lane detection methods based on deep learning," *Pattern Recognition*, vol. 111, p. 107623, 2021.

[20] D. Xiao, X. Yang, J. Li, and M. Islam, "Attention deep neural network for lane marking detection," *Knowledge-Based Systems*, vol. 194, p. 105584, 2020.

[21] T Y Lin, P Goyal, R Girshick et al., "Focal loss for dense object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 2999–3007, 2017.

[22] Z Zhang, X Zhang, C Peng et al., "Exfuse: enhancing feature fusion for semantic segmentation," *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 269–284, 2018.

[23] TuS. Benchmark, http://benchmark.tusimple.ai/, 2017.

[24] L C Chen, Y Zhu, G Papandreou et al., "Encoder-decoder with atrous separable convolution for semantic image segmentation," *Proceedings of the European conference on computer vision (ECCV)*, pp. 801–818, 2018.

[25] J. Philion, "Fastdraw: addressing the long tail of lane detection by adapting a sequential prediction network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*Long Beach, CA, USA, 2019.

[26] Z Wang, W Ren, and Q Qiu, "Lanenet: real-time lane detection networks for autonomous driving," *arXiv preprint arXiv*, p. 180701726, 2018.

[27] Y.-Bo Liu and M. Zeng, "Qing-hao Meng.Heatmap-based vanishing point boosts lane detection," *arXiv preprint arXiv*, 2020, 2007.