ICIC Express Letters
Volume **14**, Number **12**, December **2020**
 ISSN 1881-803X
pp. **1169–1176**

AN IMPROVED ALGORITHM FOR INSTANCE SEGMENTATION LANE DETECTION OF TWO BRANCHES

Cong Wang[1], Long Zhang[1] and Hailong Zhu[2]

[1]Department of Computer Science and Information Engineering
Harbin Normal University
No. 1, South of Shida Road, Limin Development District, Harbin 150025, P. R. China
15776846546@163.com

[2]Department of Computer and Information Engineering
Tianjin Normal University
No. 393, Binshuixidao, Xinqing District, Tianjin 300387, P. R. China
zhanglong@tjnu.edu.cn

Received March 2020; accepted June 2020

Abstract. *Aiming at the problems of the existing two branches lane line instance segmentation detection algorithms that accuracy is greatly affected by batches and high losses caused by the imbalance of positive and negative samples, we use the switchable normalization function to effectively solve the problem that the accuracy of the original algorithm in network training is greatly affected by batches. We use the Focal Loss function to solve the original algorithm's high losses when the positive and negative samples are not balanced. We use the traditional Stochastic Gradient Descent optimizer to optimize the entire model, the numerical instability problem of the original algorithm in the experimental process is effectively solved. We verify the effectiveness of the algorithm on the TuSimple lane dataset. Compared with the original algorithm, the performance of the improved algorithm improved significantly, the accuracy increased from 96.4% to 98.6%, relative increase of 2.28%, and the loss was reduced to 0.0158.*
Keywords: Deep learning, Two branches instance segmentation, Lane detection, Switchable normalization, Focal Loss

1. **Introduction.** Lane detection plays a very important role in intelligent driving that is widely used in intelligent assistance systems and lane departure warning systems. With the development of artificial intelligence, research on end-to-end intelligent lane instance segmentation detection algorithms is a hot spot in the field of intelligent driving [1,2].

There are two main methods for lane detection: one is the traditional hand-crafed feature recognition lane method, and the other is based on deep learning method.

Traditional lane detection methods rely on hand-crafed features to identify lane segments, including color based features [3], the structure tensor [4], the bar filters [5], ridge features [6], etc. After identifying the lane segments, post-processing techniques are employed to filter out misdetections and group segments together to form the final lanes. These traditional lane detection methods have many problems such as slow detection speed, low accuracy and poor robustness.

In recent years, deep networks have been the most popular in the fields of image classification, object detection and image instance segmentation [7]. Compared with traditional image processing methods using Convolutional Neural Network (CNN) models, the lane detection accuracy can be increased from 80% to 90% [8]. Li et al. [2] propose the use of a multi-task deep convolutional network that focuses on finding geometric lane attributes, together with a Recurrent Neural Network (RNN). Gurghian et al. [10] propose another deep CNN method that uses two lateral cameras to detect lane markings, the method

DOI: 10.24507/icicel.14.12.1169

recognizes the position of lateral lanes through an end-to-end detection process, using real and synthetic images to train the model. Neven et al. [11] aimed at the problem that the binary lane segmentation map of the above single branch needs to be separated into different lane instances and can only detect fixed lanes. They propose an end-to-end two branches instance segmentation lane detection algorithm, and it includes LaneNet and H-Net network models. First, the lane segmentation branch of the LaneNet has two output classes, background or lane, while the lane embedding branch further disentangles the segmented lane pixels into different lane instances. Then H-Net predicts the transformation matrix and remodels all pixels that belong to a lane. At last, the lane embedding branch, which is trained using a clustering loss function, assigns a lane id to each pixel from the lane segmentation branch while ignoring the background pixels. By splitting the lane detection problem into the aforementioned two tasks, that can fully utilize the power of the lane segmentation branch without having to assign different classes to different lanes.

However, the Neven et al.'s algorithm has the problems that the accuracy is greatly affected by batches, and the imbalance of positive and negative samples causes high losses. In this paper, we go beyond the aforementioned limitations. Our contributions can be summarized to the following. 1) We use the Switchable Normalization (SN) function in instance segmentation to solve the problem that the accuracy is greatly affected by batches. 2) We use Focal Loss function in LaneNet training neural network to solve the problem of high loss caused by the imbalance of positive and negative samples. 3) The traditional SGD (Stochastic Gradient Descent) optimizer is used to optimize the entire model to solve the inefficiency during experiment process.

The remainder of the paper is organized as follows. Section 2 describes the frame principle of the instance segmentation lane detection algorithm based on two branches. Section 3 introduces the optimization algorithm of ours. It replaces the normalization function, loss function and optimizer to optimize the original algorithm. Experimental analysis is in Section 4. By using the TuSimple lane data set for comparative experiments, the effectiveness of the algorithm in this paper is verified and the experimental results are analyzed. Finally, Section 5 is the conclusion, which summarizes our work.

2. Instance Segmentation Lane Detection Algorithm Based on Two Branches.

2.1. Algorithm principle.

Two branches instance segmentation lane detection algorithm proposed by Neven et al., treats lane detection as an instance segmentation problem [12], and performs the lane detection task in real time. The algorithm structure is shown in Figure 1. The segmentation branch of LaneNet (see Figure 1, top branch) is trained to output a binary segmentation map, indicating which pixels belong to a lane and which does not. The segmentation network is trained with the standard cross-entropy loss function. It has two output categories (background and lane), the white represents lane, and the black represents background. Lane embedding branch (see Figure 1, bottom branch) uses a one-shot method based on distance metric learning, one-shot method can be integrated with standard feed-forward networks and specifically designed for real-time applications. The pixel embeddings of the same lane will cluster together, forming unique clusters per lane. Then use the predicted binary image to cover the instance image and train a network, H-Net, for generating perspective transformation matrix coefficients. At last, the algorithm uses H-Net for fitting a third-order polynomial, which converts the image to a bird's-eye view to estimate the parameters of the ideal perspective transformation and reprojects the lanes onto the image to output the final. As a result, this method is not constrained on the number of lanes it can detect and is able to cope with lane changes.

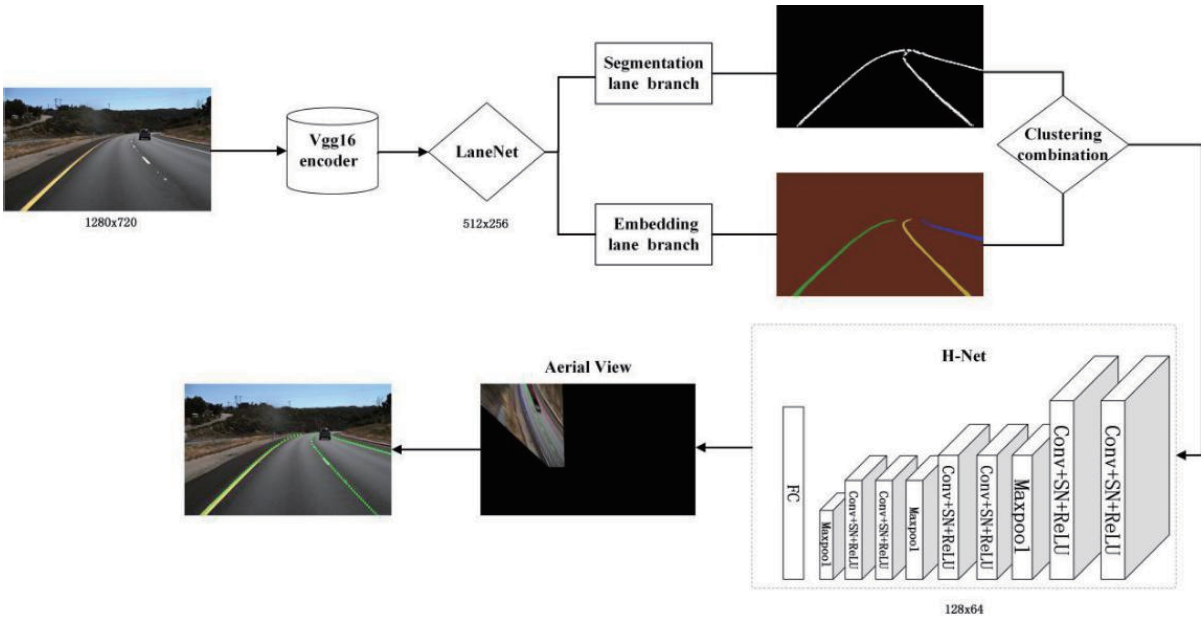

FIGURE 1. Structure of a two branches lane segmentation detection algorithm

2.2. **Normalization function.** The normalization function is to limit the data to be processed to a certain range after processing, so that the distribution of each batch of training data is the same. The purpose is to facilitate the subsequent data processing and ensure faster convergence when the program runs.

The normalization function of Neven et al.'s detection algorithm uses Batch Normalization (BN) [13], which is normalized in the dimension of batch and is independent of batch. The calculation formula of BN is

$$y = \frac{\gamma}{\sqrt{Var[x] + \varepsilon}} \cdot x + \left(\beta - \frac{\gamma E[x]}{\sqrt{Var[x] + \varepsilon}} \right) \tag{1}$$

where γ and β are learnable parameters, $\varepsilon > 0$ is a very small constant, $E[x]$ represents the average value of each batch of training data, $Var[x]$ represents the variance of each batch of training data, and y is the normalized output value.

2.3. **Calculation of loss function.** The loss function is used to represent the degree of inconsistency between the predicted value and the true value of the model, and it is one of the important parameters determining the network effect. The loss function of Neven et al.'s segmentation network is trained with the standard cross-entropy loss function, and the calculation formula is

$$FL(p_t) = -\log(p_t) \tag{2}$$

where p_t is the prediction probability.

3. Improved Instance Segmentation Lane Detection Algorithm.

3.1. **Improvement of the normalization function.** We use the Switchable Normalization (SN) function to effectively solve the problem that the accuracy of the original algorithm in network training is greatly affected by batches. SN [14] trains and learns to select different normalizers for different normalization layers of deep neural networks, and combines three types of statistical information that are Instance Normalization (IN) [15], Layer Normalization (LN) [16], and BN. These types of statistical information are respectively estimated through channel, layer, and mini-batch methods to learn their importance weights in an end-to-end manner in deep neural networks, thereby switching

between BN, IN, LN, and then selecting the appropriate attribution for the network. Suppose the data is represented as four dimensions (N, C, H, W) of the input feature map, and each dimension represents the number of samples, the number of channels, the height of the channel, and the width of the channel. The normalized pixel value calculation formula for SN is

$$\hat{h}_{ncij} = \gamma \frac{h_{ncij} - \sum_{k\in\Omega} w_k \mu_k}{\sqrt{\sum_{k\in\Omega} w'_k \sigma_k^2 + \varepsilon}} + \beta \tag{3}$$

The mean μ and variance σ^2 of SN are weighed and averaged by selecting an appropriate normalization method in a set Ω that includes BN, IN and LN. The weight coefficients corresponding to the statistics are w_k and w'_k, and h_{ncij} represents every pixel. The structural model of SN is shown in Figure 2. W_{bn}, W_{in}, W_{ln} represent the weights of BN, IN, LN respectively.

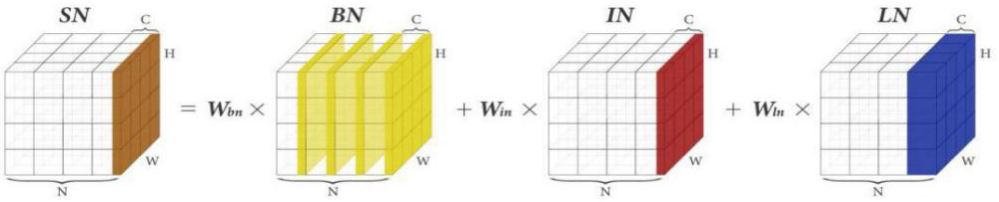

FIGURE 2. Switchable normalization model

3.2. **Improvement of loss function.** Since the lane detection is mostly simple and easily separable negative samples (samples that belong to the background), the training process cannot fully learn the information of category samples, and the negative samples are too many, which may mask the role of other existing samples category. These simple and easily separable negative samples will produce a certain degree of loss, and then they will play a major role in loss and dominate the update direction of the gradient and conceal important information. The cross-entropy loss is used by Neven et al., the prediction result will change with the change of P_t, which will cause poor performance and unsatisfactory loss of the lane detection module in the experiment.

In this paper, Focal Loss is used instead of the standard cross-entropy loss function to guide the network parameter learning by backpropagating the errors generated by the prediction samples and the true sample labels, fully learning the positive sample parameters of the lanes that account for a small number of the total samples. For simple samples, P_t will be larger, and the weight will naturally decrease. For difficult samples, P_t will be smaller, and the weight will be larger. Focal Loss adds a modulation factor $(1-p_t)^\gamma$ to the standard cross-entropy loss, it uses adjustable parameters γ and α to balance the positive and negative proportions in Focal Loss, where γ has the same meaning as γ in Equation (1). Focal Loss is calculated as:

$$FL(p_t) = -\alpha_t (1-p_t)^\gamma \log(p_t) \tag{4}$$

The modulation factor added in the formula changes dynamically, which will gradually make the complex samples better, update the loss function to reduce the proportion of simple examples, and finally make the impact of negative samples gradually decrease.

3.3. **Optimizer improvements.** Aiming at the instability problem in the training process caused by Neven et al. using Adam (Adaptive Moment Estimation) optimizer, we use a traditional SGD optimizer to optimize the entire model and train the network until convergence. SGD can automatically escape the saddle point, escape the local best advantage, and can perform well on dataset that has not been seen, which can solve the instability in the training process caused by the original algorithm using Adam optimizer

problem. It has been experimentally verified that the SGD optimizer is more stable during training than before, and does not fall into NaN errors as easily as when using Adam.

4. Experiments and Analysis.

4.1. **Dataset.** We use the TuSimple dataset, which is a large scale dataset for testing deep learning methods on the lane detection task. It consists of 3626 training and 2782 testing images, under good and medium weather conditions. They are recorded on 2-lane/3-lane/4-lane or more highway roads, at different daytimes. For each image, they also provide the 19 previous frames, which are not annotated. The annotations come in a json format, indicating the x-position of the lanes at a number of discretized y-positions. On each image, the current (ego) lanes and left/right lanes are annotated and this is also expected on the test set.

4.2. **Evaluation criteria.** C_{im} denotes the number of correct points and S_{im} denotes the number of ground-truth points. A point is correct when the difference between a ground-truth and predicted point is less than a certain threshold. The accuracy is calculated as the average correct number of points per image:

$$acc = \sum_{im} \frac{C_{im}}{S_{im}} \tag{5}$$

4.3. **Setup.** Experiments are performed on Ubuntu 16.04 (×64), python3.5, Cuda-9.0, cudnn-7.0, and TensorFlow 1.10.0. Vgg 16 is selected as the basic encoder. The size of the TuSimple original image is 1280 × 720, rescale the images to 512 × 256 in LaneNet and rescale the images to 128 × 64 in H-Net. The training period is set to 10000, the batch size is 8, and the initial learning rate is 0.0005.

4.4. **Experiments.** The experiment results are shown from Figure 3 to Figure 6. Three representative pictures were selected for testing. The first is a picture of three straight lanes in windy weather, the middle part of the picture is covered by wind and sand, the second is a picture of four straight lanes with good weather, and the third is three lanes with curved part being blocked image. It can be seen from the experiment results that the lane segment instance detection and our algorithm based on the two branches has a good fit of the lane, that can identify multiple lanes and block lanes accurately.

FIGURE 3. The original images

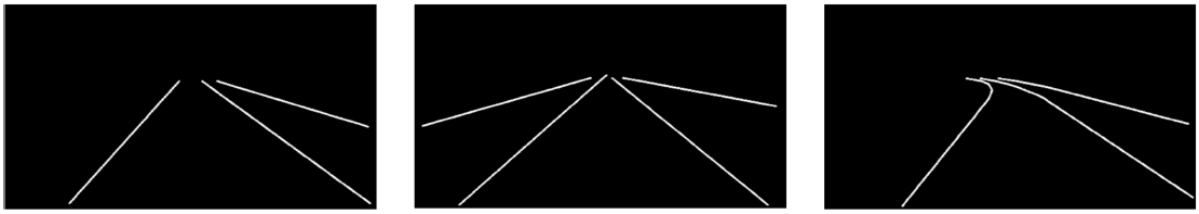

FIGURE 4. Picture of binary lane

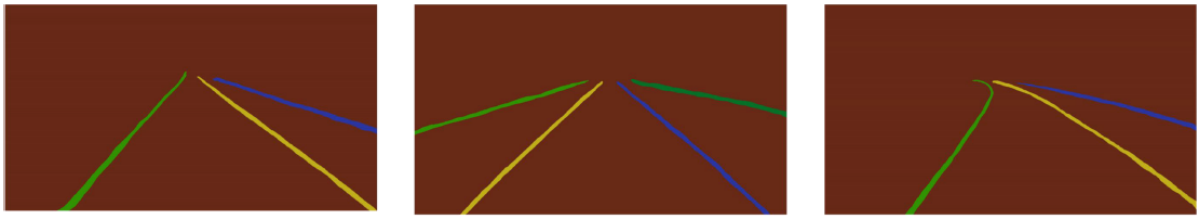

FIGURE 5. LaneNet output pictures

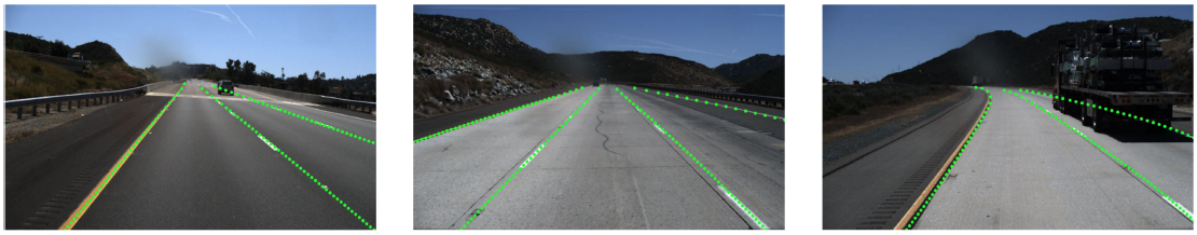

FIGURE 6. Final predictions

TABLE 1. Comparison of different algorithms for accuracy and loss

	Accuracy	Loss
Leonardoli	96.9%	0.0197
Xingang Pan	96.5%	0.0180
Aslarry	96.5%	0.0260
Neven et al.	96.4%	0.0244
Ours	98.6%	0.0158

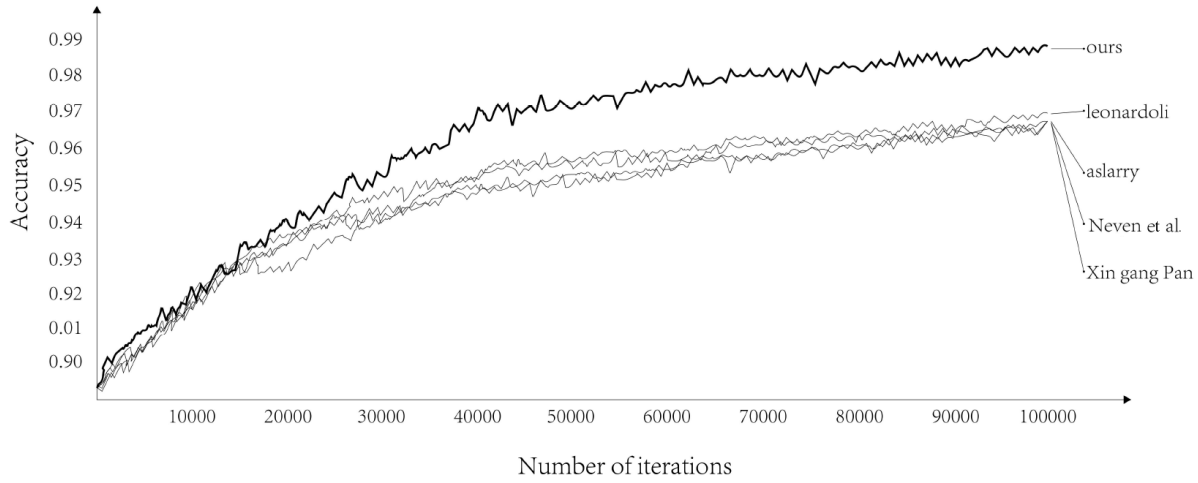

FIGURE 7. Testing accuracy in different algorithms

We compare the experiments with the top four algorithms in the TuSimple2017 challenge [17]. The experiment results are shown in Table 1, Figure 7 and Figure 8. Compared with other algorithms, our algorithm has significantly higher accuracy than other algorithms. Compared with Neven et al.'s algorithm, the accuracy rate is improved by 2%. The accuracy rate at the beginning of training is about 0.89. As the number of training iterations increases, the value remains stable, and after iteration to 100000 times the accuracy rate can be maintained at more than 98%. Compared with other algorithms, ours has a significantly reduced loss. The value of our algorithm at the beginning of training

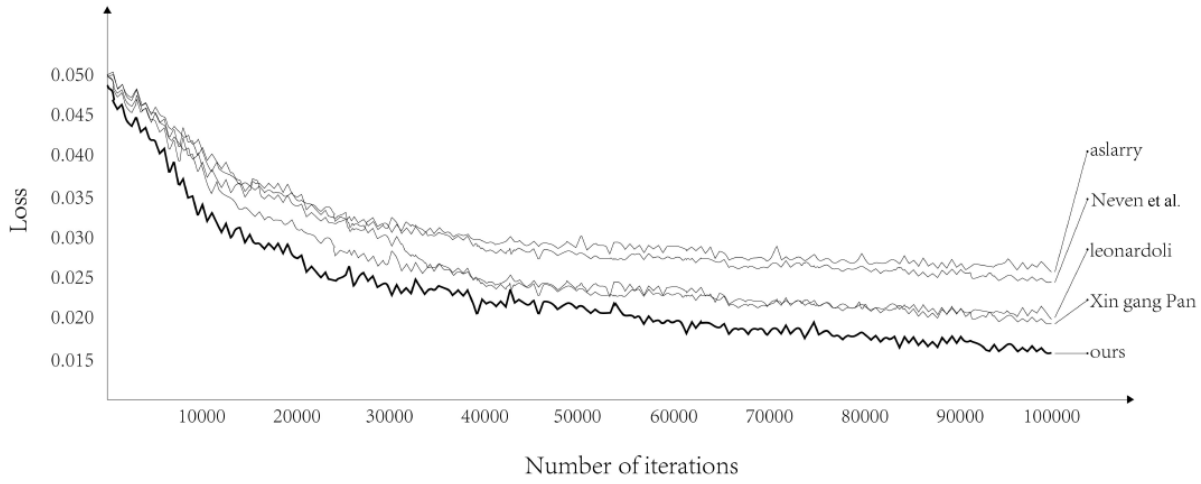

FIGURE 8. Testing loss in different algorithms

is about 0.050, with the increase of the number of training iterations, the loss value gradually decreases and remains stable. When iterating to 100000 times the loss value can be reduced less than 0.016.

5. **Conclusions.** This paper studies the existing end-to-end lane instance segmentation detection algorithm based on two branches, we propose an optimization algorithm to make up for the shortcomings of high loss caused by the imbalance of positive and negative samples, and large accuracy affected by batches. Experiment results show that compared with other related deep learning algorithms, our algorithm improves the detection accuracy, reduces the loss, and recognize multi-lanes and block lanes.

REFERENCES

[1] B. He, R. Ai and Y. Yan, Accurate and robust lane detection based on dual-view convolutional neutral network, *Proc. of the Intelligent Vehicles Symposium*, Sweden, pp.1041-1046, 2016.

[2] J. Li, X. Mei, D. Prokhorov and D. Tan, Deep neural network for structural prediction and lane detection in traffic scene, *Neural Networks and Learning Systems*, vol.28, no.3, pp.690-703, 2017.

[3] K. Y. Chiu and S. F. Lin, Lane detection using color based segmentation, *Intelligent Vehicles Symposium*, pp.706-711, 2005.

[4] H. Loose, U. Franke and C. Stiller, Kalman particle filter for lane recognition on rural roads, *IEEE Intelligent Vehicles Symposium*, pp.60-65, 2009.

[5] Z. Teng, J. H. Kim and D. J. Kang, Real-time lane detection by using multiple cues, *Control Automation and Systems*, pp.2334-2337, 2010.

[6] A. Lopez, J. Serrat and C. Canero, Robust lane markings detection and road geometry computation, *International Journal of Automotive Technology*, vol.11, no.3, pp.395-407, 2010.

[7] Y. Omae, M. Mori, T. Akiduki and H. Takahashi, A novel deep learning optimization algorithm for human motions anomaly detection, *International Journal of Innovative Computing, Information and Control*, vol.15, no.1, pp.199-208, 2019.

[8] B. He, R. Ai and Y. Yan, Lane marking detection based on convolution neural network from point clouds, *IEEE Trans. Intelligent Transportation Systems*, pp.2475-2480, 2016.

[9] S. Sato, M. Hashimoto and M. Takita, Multilayer libar-based pedestrian tracking in urban environments, *IEEE Intelligent Vehicles Symposium (IV)*, pp.849-854, 2010.

[10] A. Gurghian, T. Koduri and S. V. Bailur, DeepLanes: End-to-end lane position estimation using deep neural networks, *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Las Vegas, pp.38-45, 2016.

[11] D. Neven, B. D. Brabandere, S. Georgoulis, M. Proesmans and L. V. Gool, Towards end-to-end lane detection: An instance segmentation approach, *IEEE Intelligent Vehicles Symposium (IV)*, pp.286-291, DOI: 10.1109/IVS.2018.8500547, 2018.

[12] K. He, G. Gkioxari and P. Dollar, Mask R-CNN, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2017.

[13] S. Ioffe and C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, *International Conference on Machine Learning (ICML)*, 2015.
[14] P. Luo, J. Ren and Z. Peng, Differentiable learning to normalize via switchable normalization, *International Conference on Learning Representations (ICLR)*, 2018.
[15] D. Ulyanov, A. Vedaldi and V. Lempitsky, Instance normalization: The missing ingredient for fast stylization, *CVPR*, 2016.
[16] J. L. Ba, J. R. Kiros and G. E. Hinton, Layer normalization, *CVPR*, 2016.
[17] X. Pan, J. Shi and P. Luo, Spatial as deep: Spatial CNN for traffic scene understanding, *AAAI Conference on Artificial Intelligence*, 2018.