Agnostic Lane Detection

Yuenan Hou
The Chinese University of Hong Kong
hy117@ie.cuhk.edu.hk

***Abstract*—Lane detection is an important yet challenging task in autonomous driving, which is affected by many factors, e.g., light conditions, occlusions caused by other vehicles, irrelevant markings on the road and the inherent long and thin property of lanes. Conventional methods typically treat lane detection as a semantic segmentation task, which assigns a class label to each pixel of the image. This formulation heavily depends on the assumption that the number of lanes is pre-defined and fixed and no lane changing occurs, which does not always hold. To make the lane detection model applicable to an arbitrary number of lanes and lane changing scenarios, we adopt an instance segmentation approach, which first differentiates lanes and background and then classify each lane pixel into each lane instance. Besides, a multi-task learning paradigm is utilized to better exploit the structural information and the feature pyramid architecture is used to detect extremely thin lanes. Three popular lane detection benchmarks, i.e., TuSimple, CULane and BDD100K, are used to validate the effectiveness of our proposed algorithm.**

I. Introduction

Lane detection [1] plays a pivotal role in autonomous driving because lanes could serve as significant cues for constraining the maneuver of vehicles on roads. However, lane detection is challenging since it is affected by many factors, e.g., light conditions, occlusions caused by other vehicles, the existence of irrelevant markings on the road and the inherent long and thin property of lanes.

Conventional methods [2], [4] usually utilize hand-crafted features to extract lane segments and can perform quite well in the highway driving scenarios. However, these approaches need a good selection of features and have poor generalization ability. Therefore, they cannot be applied to scenarios with varying light conditions and road types. The emergence of deep learning has brought new insights into the task and Convolutional Neural Network (CNN) based methods begin to gain popularity [10], [13], [5], [3], [7], [8], [6]. The inherent and automatic feature extracting ability of CNN eases the complex feature selection process and partially solves the generalization problems. However, the CNN-based methods perform sub-optimally in urban roads where the lane markings are ambiguous or the lanes are severely occluded. Several schemes have been proposed to handle lane detection in urban roads, e.g., performing message passing to better exploit structural information [13] or utilizing vanishing points to guide the lane detection task [10]. These methods can work to some extent but cannot fully solve the problem as they ignore the inherent relationship between the different entities in the driving scenarios. For instance, the areas within two neighbouring lanes (i.e., drivable areas and alternative areas [16]) can serve as a strong indicator for the existence, shape and position of lanes. Besides, these models tend to fail when encountering an arbitrary number of lanes or lane

changing since they model lane detection as the semantic segmentation task and each lane is assigned a pre-defined class. Failing to achieve real-time performance is also a drawback of these approaches [13], [10].

Therefore, in this study, we propose to use a multi-task learning paradigm to better utilize the structural and contextual information of the driving scenarios. More specifically, besides the traditional lane detection branch, we also borrow the rich structural information from the drivable area detection task and the lane point regression task. The feature pyramid architecture [11] is also incorporated in our model to handle challenges of detecting extremely thin lanes. To fulfill the real-time requirement, we adopt the light-weight and efficient network, i.e., ENet [14], as our backbone. To detect conditions with unfixed number of lanes, we follow [12] and divide the lane detection task into two sub-tasks. The first one is to generate the binary segmentation map which only differentiates the lanes and the background. The second sub-task is to classify the lane pixels into different lane instances (i.e., treat each lane as an instance). Three popular benchmarks, i.e., TuSimple [15], CULane [13] and BDD100K [16], are selected to validate the effectiveness of our proposed algorithm. Since it is an on-going project, we only report preliminary experimental results on TuSimple and CULane.

II. Related Work

Lane detection is conventionally handled via using specialized and hand-crafted features to obtain lane segments. These segments are further grouped to get the final results [2], [4]. These methods are intuitive but have many shortcomings, e.g., requiring complex feature selection process, being lack of robustness and only applicable to relatively easy driving scenarios.

Recently, deep learning methods [10], [13], [5], [3] have been proposed to ease the selection of hand-crafted features as well as greatly improve the models' generalization ability. These approaches usually adopt the dense prediction formulation, i.e., treat lane detection as a semantic segmentation task, where each pixel in an image is assigned with a label to indicate whether it belongs to a lane or not. For example, Pan et al [13] propose SCNN, which combines spatial cues with CNN, to generate multi-channel probability maps where the number of channels equals to the number of lanes. However, these methods can only handle scenarios where the number of lanes is pre-defined and fixed, and they often fail when the vehicle is changing lanes. Another drawback is that these approaches could not achieve real-time performance, which impedes them from being used in the real world.

To overcome these shortcomings, we follow [12] and model lane detection as an instance segmentation task. More specifically, the lane detection task is divided into two sub-tasks. The first sub-task is generating a binary segmentation map which differentiates lanes and the background. The second sub-task is classifying each lane pixel into a lane instance. The light-weight network, i.e., ENet [14] is used as our backbone to achieve real-time performance. What's more, to utilize the structural and contextual information, we adopt a multi-task learning paradigm in which drivable area detection and lane point regression are incorporated into the original lane detection model. Moreover, the feature pyramid architecture is utilized to detect extremely thin lanes.

III. Methodology

In this section, we will give a detailed explanation of our framework as shown in Fig. 1. Our model is mainly composed of five components, i.e., the binary segmentation branch (B), the drivable area detection

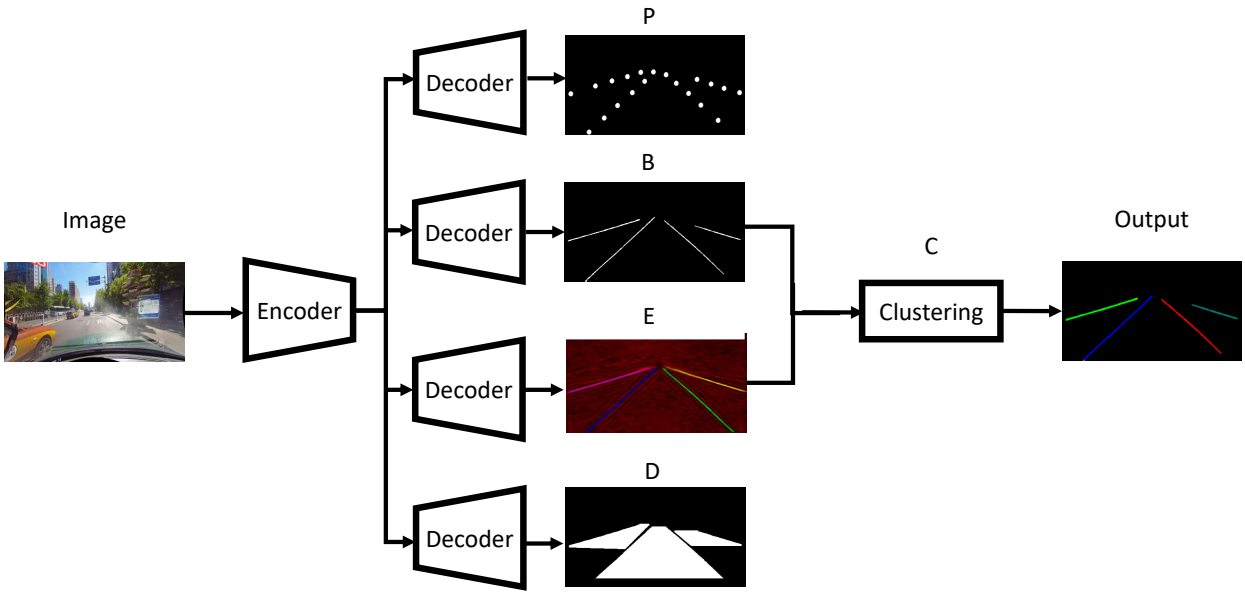

Fig. 1: An overview of our agnostic lane detection model.

branch (D), the lane point regression branch (P), the lane pixel embedding branch (E) and the clustering branch (C). The encoder and decoder of the first four branches are the same but only the encoder is shared.

A. Binary Segmentation

The objective of the binary segmentation branch is to generate a binary segmentation map, indicating whether each pixel in the original image belongs to the lanes or not. Since the ground-truth lane labels of three datasets are all lane points, we generate the final targets by connecting lane points into lines (see the final targets in Fig. 2). We use standard cross-entropy loss to train this branch. Besides, to solve the class imbalance of lane pixels and background pixels, the loss of background is multiplied by 0.4. Moreover, the feature pyramid architecture [11] is adopted to detect extremely thin lanes.

B. Drivable Area Detection

The target of the drivable area detection branch is to output a segmentation map, indicating which part of the road is drivable (we merge the original alternative areas into the drivable areas to provide denser targets). Standard cross-entropy loss is adopted to train this branch. This branch aims at using the boundary of drivable areas to refine the binary segmentation result via providing more structural information.

C. Lane Point Regression

The objective of this branch is to regress the position of each lane points. Since the lane points are relatively sparse, we use an 11 x 11 kernel to smooth the original lane point maps to get the final targets of this branch. L_2 loss is used to train this branch. This branch aims at refining the output of the binary segmentation branch.

D. Lane Pixel Embedding

The input of this branch is the lane pixels extracted from the binary segmentation maps. We treat each lane

in the image as an instance. The target of this branch is to classify the lane pixels into different lane instances. The core idea is that pixels belonging to the same lane instance should be close to each other while those belonging to different lane instances should be far from each other. We utilize the following equation to compute the clustering loss [12]:

$$L_{var} = \frac{1}{L}\sum_{c=1}^{L}\frac{1}{N_c}\sum_{i=1}^{N_c}[\|\mu_c - x_i\|_2^2 - \delta_v]_+^2, \quad (1)$$

$$L_{dist} = \frac{1}{L(L-1)}\sum_{c_A=1}^{L}\sum_{c_B=1,c_A\neq c_B}^{L}[\delta_d - \|\mu_{c_A} - \mu_{c_B}\|_2^2]_+^2, \quad (2)$$

where L denotes the number of lanes, x_i is the embedding of a pixel, N_c is the number of elements in cluster c, μ_c is the mean embedding of cluster c and $[x]_+ = max(0, x)$. The first loss term L_{var} is used to keep the distance between pixels belonging to the same lane instance closer than $2\delta_v$. The second loss term L_{dist} is used to keep the distance between different lane clusters farther than δ_d.

E. Clustering

The clustering branch is used to process the output of the lane pixel embedding branch. In the experiments, we set $\delta_d > 6\delta_v$. Therefore, given the output of the lane pixel embedding branch, we can randomly select a pixel as the starting point, and then label all pixels whose distance from the selected pixel is smaller than $2\delta_v$ as the same instance. This process is continued until all the lane pixels are assigned to a specific lane instance. Note that this branch does not have any learnable parameters.

F. Training strategy

Currently, we adopt a two-stage training strategy. In the first stage, we fix the parameters of branch E and train the branch P, B and D. In the second stage, we fix the parameters of branch P, B and D and train branch E.

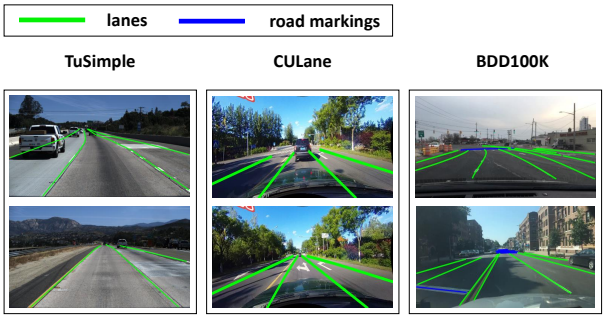

Fig. 2: Typical video frames of TuSimple, CULane and BDD100K datasets.

IV. Experiments

In this section, we will first give a brief introduction to three datasets used for evaluation. Then, preliminary experimental results are given.

A. Dataset

Table I records the basic information of three lane detection datasets. Note that the last column of Table I shows that TuSimple and CULane have no more than 5 lanes in a video frame while BDD100K typically has more than 8 lanes in a video frame. Besides, TuSimple is relatively easy while CULane and BDD100K are more challenging considering the total number of video frames and road types.

B. Evaluation Criterion

1) TuSimple: In TuSimple dataset, we use the official metric (accuracy) as the evaluation criterion. Besides, false positive (FP) and false negative (FN) are also listed. The following is the equation to compute accuracy [15]:

$$Accuracy = \frac{N_{pred}}{N_{gt}}, \quad (3)$$

TABLE I: A brief description about three lane detection datasets.

Name	# Frame	Train	Validation	Test	Resolution	Road Type	# Lane ≤ 5 ?
TuSimple	6, 408	3, 626	–	2, 782	1280 × 720	highway	$\surd$
CULane	133, 235	88, 880	9, 675	34, 680	1640 × 590	urban, rural and highway	$\surd$
BDD100K	80, 000	70, 000	–	10, 000	1280 × 720	urban, rural and highway	×

where N_{pred} is the number of correctly predicted lane points and N_{gt} is the number of ground-truth lane points.

2) CULane and BDD100K: To judge whether a lane is correctly detected, we treat each lane as a line with fixed pixel width (30 for CULane and 8 for BDD100K) and compute the intersection-over-union (IoU) between labels and predictions. Predictions whose IoUs are larger than 0.5 are considered as true positives (TP). Then, we use $F_1 - measure$ as the evaluation metric formulated as follows:

$$F_1 - measure = \frac{2 \times Precision \times Recall}{Precision + Recall}, \quad (4)$$

where $Precision = \frac{TP}{TP+FP}$ and $Recall = \frac{TP}{TP+FN}$.

C. Lane detection model

We choose ENet [14] as the backbone model (i.e., the encoder and decoder module in Fig. 1). Adam [9] is selected as the optimizer to train our model with an initial learning rate of 5×10^{-4}.

D. Preliminary results on TuSimple and CULane

Table II records the performance of some baselines and our algorithm in the testing set of TuSimple. Since TuSimple is relatively easy and our ENet model has much fewer parameters compared with SCNN (see Table IV), the performance of our model is satisfying. Table III records the performance of some baselines and our algorithms in the testing set of CULane. As can be seen in Table IV, in terms of the running time efficiency and the number of parameters, our algorithm obviously outperforms other baselines.

TABLE II: Performance of different algorithms on TuSimple testing set.

Algorithm	Accuracy	FP	FN
SCNN [13]	0.9653	0.0617	0.0180
LaneNet [12]	0.9638	0.0780	0.0244
EL-GAN [5]	0.9639	0.0412	0.0336
ENet (ours)	0.9629	0.0722	0.0218

TABLE IV: The running time and parameters of different algorithms on CULane testing set.

Indicator	**ENet**	ResNet-18	ResNet-101	SCNN
Running time (ms)	**13.4**	25.3	171.2	133.5
Parameter (M)	**0.98**	12.41	52.53	20.72

V. Conclusion

In this study, we first point out the value and main challenges of the lane detection task. Then, the strengths and weaknesses of both conventional and deep learning based methods are presented. To overcome the shortcomings of previous methods, our agnostic lane detection model is proposed, which utilizes a multi-task learning paradigm and the feature pyramid architecture to exploit structural and contextual information. We use three popular benchmarks, i.e., TuSimple, CULane and BDD100K, to validate the effectiveness of the proposed algorithm. Preliminary experimental results have shown that our model outperforms previous approaches in terms

TABLE III: Performance (F_1-measure) of different algorithms on CULane testing set. † indicates the results are copied from [13]. For crossroad, only FP is shown.

Category	**ENet**	ResNet-18	VGG-16†	ReNet†	DenseCRF†	MRFNet†	ResNet-50†	ResNet-101†	SCNN†
Normal	88.4	89.8	83.1	83.3	81.3	86.3	87.4	90.2	**90.6**
Crowded	67.0	68.1	61.0	60.5	58.8	65.2	64.1	68.2	**69.7**
Night	61.4	64.2	56.9	56.3	54.2	61.3	60.6	65.9	**66.1**
No line	42.9	42.5	34.0	34.5	31.9	37.2	38.1	41.7	**43.4**
Shadow	63.4	67.5	54.7	55.0	56.3	59.3	60.7	64.6	**66.9**
Arrow	81.9	83.9	74.0	74.1	71.2	76.9	79.0	84.0	**84.1**
Dazzle light	57.4	59.8	49.9	48.2	46.2	53.7	54.1	**59.8**	58.5
Curve	62.6	65.5	61.0	59.9	57.8	62.3	59.8	**65.5**	64.4
Crossroad	2768	1995	2060	2296	2253	**1837**	2505	2183	1990
Total	68.8	70.5	63.2	62.9	61.0	67.0	66.7	70.8	**71.6**

of the running time efficiency and the number of parameters. However, this is still an on-going project and more performance gains will be achieved via a good deployment of different components and a more rational training strategy.

REFERENCES

[1] Massimo Bertozzi and Alberto Broggi. Gold: A parallel real-time stereo vision system for generic obstacle and lane detection. *IEEE transactions on image processing*, 7(1):62–81, 1998.

[2] Amol Borkar, Monson Hayes, and Mark T Smith. A novel lane detection system with efficient ground truth generation. *IEEE Transactions on Intelligent Transportation Systems*, 13(1):365–374, 2012.

[3] Zhe Chen and Zijing Chen. Rbnet: A deep neural network for unified road and road boundary detection. In *International Conference on Neural Information Processing*, pages 677–687. Springer, 2017.

[4] Hendrik Deusch, Jürgen Wiest, Stephan Reuter, Magdalena Szczot, Marcus Konrad, and Klaus Dietmayer. A random finite set approach to multiple lane detection. In *Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on*, pages 270–275. IEEE, 2012.

[5] Mohsen Ghafoorian, Cedric Nugteren, Nóra Baka, Olaf Booij, and Michael Hofmann. El-gan: Embedding loss driven generative adversarial networks for lane detection. *arXiv preprint arXiv:1806.05525*, 2018.

[6] Yuenan Hou. Agnostic lane detection. *arXiv preprint arXiv:1905.03704*, 2019.

[7] Yuenan Hou, Zheng Ma, Chunxiao Liu, and Chen Change Loy. Learning to steer by mimicking features from heterogeneous auxiliary networks. *arXiv preprint arXiv:1811.02759*, 2018.

[8] Yuenan Hou, Zheng Ma, Chunxiao Liu, and Chen Change Loy. Learning lightweight lane detection cnns by self attention distillation. *arXiv preprint arXiv:1908.00821*, 2019.

[9] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[10] Seokju Lee, Junsik Kim, Jae Shin Yoon, Seunghak Shin, Oleksandr Bailo, Namil Kim, Tae-Hee Lee, Hyun Seok Hong, Seung-Hoon Han, and In So Kweon. Vpgnet: Vanishing point guided network for lane and road marking detection and recognition. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 1965–1973. IEEE, 2017.

[11] Tsung-Yi Lin, Piotr Dollár, Ross B Girshick, Kaiming He, Bharath Hariharan, and Serge J Belongie. Feature pyramid networks for object detection. In *CVPR*, volume 1, page 3, 2017.

[12] Davy Neven, Bert De Brabandere, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Towards end-to-end lane detection: an instance segmentation approach. *arXiv preprint arXiv:1802.05591*, 2018.

[13] Xingang Pan, Jianping Shi, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Spatial as deep: Spatial cnn for traffic scene understanding. *arXiv preprint arXiv:1712.06080*, 2017.

[14] Adam Paszke, Abhishek Chaurasia, Sangpil Kim, and Eugenio Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation. *arXiv preprint arXiv:1606.02147*, 2016.

[15] TuSimple. http://benchmark.tusimple.ai/#/t/1. Accessed: 2018-09-08.

[16] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*, 2018.