Laneformer: Object-Aware Row-Column Transformers for Lane Detection

Jianhua Han[1], Xiajun Deng[2], Xinyue Cai[1], Zhen Yang[1], Hang Xu[1], Chunjing Xu[1], Xiaodan Liang[2]*

[1]Huawei Noah's Ark Lab
[2]Shenzhen Campus of Sun Yat-sen University
{hanjianhua4, caixinyue1, yang.zhen, xu.hang, xuchunjing}@huawei.com,
dengxj9@mail2.sysu.edu.cn, liangxd9@mail.sysu.edu.cn

Abstract

We present **Laneformer**, a conceptually simple yet powerful transformer-based architecture tailored for lane detection that is a long-standing research topic for visual perception in autonomous driving. The dominant paradigms rely on purely CNN-based architectures which often fail in incorporating relations of long-range lane points and global contexts induced by surrounding objects (e.g., pedestrians, vehicles). Inspired by recent advances of the transformer encoder-decoder architecture in various vision tasks, we move forwards to design a new end-to-end Laneformer architecture that revolutionizes the conventional transformers into better capturing the shape and semantic characteristics of lanes, with minimal overhead in latency. First, coupling with deformable pixel-wise self-attention in the encoder, Laneformer presents two new row and column self-attention operations to efficiently mine point context along with the lane shapes. Second, motivated by the appearing objects would affect the decision of predicting lane segments, Laneformer further includes the detected object instances as extra inputs of multi-head attention blocks in the encoder and decoder to facilitate the lane point detection by sensing semantic contexts. Specifically, the bounding box locations of objects are added into *Key* module to provide interaction with each pixel and query while the ROI-aligned features are inserted into *Value* module. Extensive experiments demonstrate our Laneformer achieves state-of-the-art performances on CULane benchmark, in terms of 77.1% F1 score. We hope our simple and effective Laneformer will serve as a strong baseline for future research in self-attention models for lane detection.

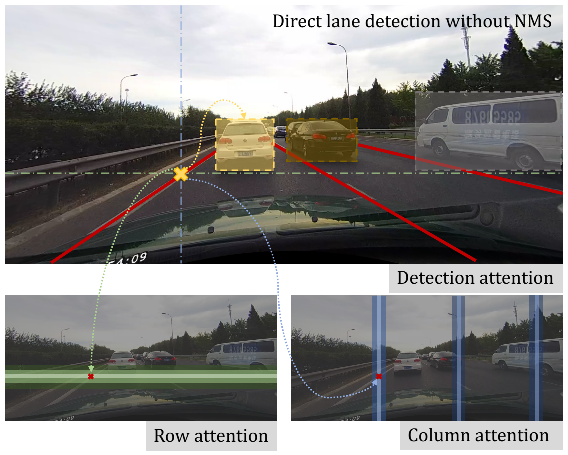


Figure 1: Sketch map of proposed detection attention and row-column attention in Laneformer. Given the detected person and vehicle instances, detection attention is performed to capture the implicit relationship between them and lanes, e.g., lanes are more likely to appear next to cars. Row attention is proposed to catch the information from the nearby rows since pixels in the same lane will not be far from each other between adjacent rows. On the other hand, noticed that different columns may across different lanes, the knowledge sharing of these columns may capture different lane features to construct better representations.

Introduction

Lane detection has been a long-standing task for visual perception in autonomous driving scenarios, targeting at precisely distinguishing lane segments from complicated road scenes(Hou et al. 2019; Wang, Shen, and Teoh 2000; Narote et al. 2018). It plays a crucial role towards a safe and reliable auto-driving system widely used in intelligent cars to assist drivers with modern technologies such as adaptive cruise control, lane departure warning, and traffic understanding.

The most state-of-the-art lane detection methods (Lee et al. 2021; Xu et al. 2020; Chen, Liu, and Lian 2019) take advantage of CNN architectures and exceed the traditional methods(Niu et al. 2016; Narote et al. 2018; Wang, Shen, and Teoh 2000) by a large margin. However, the existing CNN-based methods usually need complicated post-processing procedures like non-maximal suppression or clustering. In addition, the fixed receptive field of CNN architecture limits the ability to incorporate relations for long-range lane points, making them hard to capture well the characteristics of lanes since the shapes are conceptually long and thin. Several attention-based lane detection models (Lee et al. 2021; Tabelini et al. 2020b; Liu et al. 2021) have been also proposed to capture the long-range information. Nevertheless, the fixed attention routines can not adap-

*Corresponding authors

tively fit the shape characteristic of lanes. Besides, complicated road scenes including different light, weather conditions and occlusions of surrounding objects further require the model with a stronger perception of global contexts. In addition to the point-wise context information, it is reasonable to assume that objects on the road (e.g., pedestrians, vehicles) have some implicit relations with surrounding lanes. As the key sub-module (e.g., person-vehicle detection) of autonomous driving systems usually co-exist with lane detection, the sub-module outputs may promote the performance of lane detection and improves the system safety. However, none of the existing methods has considered incorporating semantics induced by detection results from a single and unified network view.

On the other hand, Transformer(Vaswani et al. 2017), a kind of encoder-decoder architecture, has shown surprisingly promising ability in dealing with tasks that require to capture global relations in nature language processing(Devlin et al. 2018; Radford et al. 2018; Young et al. 2018) and vision tasks such as image classification(Dosovitskiy et al. 2020), object detection(Carion et al. 2020; Zhu et al. 2020) and image segmentation(Zheng et al. 2020; Wang et al. 2020). Particularly, (Liu et al. 2021) proposed an LSTR model to predict lanes as polynomial functions using a transformer. Despite its benefit in providing rich global contexts in predicting lanes, the definition that regards lanes as polynomial functions has many drawbacks: 1) LSTR needs to formulate camera intrinsic and extrinsic parameters, which hinders the model from transferring to other datasets or train with combined datasets; 2) LSTR still lacks of explicitly modeling global semantic contexts into facilitating lane detection.

To tackle the above-mentioned issues, we move forwards to design a new Laneformer architecture to better capture the shape characteristics and global semantic contexts of lanes using transformers (shown in Figure 1). Our Laneformer defines lanes as a series of points. Then, the lane-specific row-column attentions are proposed to efficiently mine point context along with the lane shapes. Concretely, we take each feature row as a token and perform row-to-row self-attention and do the same for each feature column to perform column-to-column self-attention. Moreover, motivated by the appearing objects that would affect lanes' predictions, Laneformer further includes the detected object instances as auxiliary inputs of multi-head attention blocks in encoder and decoder to facilitate the lane point detection by sensing semantic contexts. To be specific, the bounding box locations of objects are added into the Key module to provide interaction with each pixel and query. At the same time, the ROI-aligned features are inserted into the Value module to provide detection information. Considering that bounding box locations and ROI-aligned features contain limited information about object instances, we use confident scores and predicted categories of the detected outputs to further improve the model's performance. Bipartite matching is adopted to ensure one-to-one alignment between predictions and ground truths, which makes the Laneformer architecture eliminate additional post-processing procedures.

Extensive experiments conducted on CULane(Pan et al. 2018) and TuSimple(TuSimple 2017) benchmarks show that our Laneformer as an early transformer attempt on lane detection, achieves state-of-the-art performance on CULane with 77.1% F1 score and superior 96.8% accuracy on Tusimple, with about 50 FPS(frames-per-second) inference speed on V100. Besides, visualization of the learned attention map in transformer demonstrates that our Laneformer can incorporate relations of long-range lane points and global contexts induced by surrounding objects.

To summarize, the main contributions can be listed:

- We design a new Laneformer architecture specified for lane detection, which accommodates the conventional transformers into capturing well the shape characteristics and semantic contexts of lanes.
- The lane-specific row-column self-attentions is proposed to mine point context along with the lane shapes with the prior that lanes are long and thin.
- The object instances by detection that usually co-exist in autonomous driving system are used as auxiliary inputs to sense object-wise semantic contexts.
- Experiments show that Laneformer achieves new state-of-the-art performance, in terms of 77.1% F1 score on CULane and superior 96.8% accuracy on Tusimple.

Related Work

Attention-free methods. Before the advent of deep learning, traditional lane detection methods are usually based on hand-crafted low-level features (Chiu and Lin 2005; Lee and Cho 2009; Gonzalez and Ozguner 2000). CNN architecture has then been adopted to extract advanced features in an end-to-end manner. Most lane detection methods follow pixel-level segmentation-based approach (Pizzati et al. 2019; Hou 2019; Mamidala et al. 2019; Zou et al. 2019). These approaches typically generate segmentation results with an encoder-decoder structure and then post-processing them via curve fitting and clustering. However, pixel-wise lane prediction methods usually require more computation and are also limited to the pre-defined number of lanes. On the other hand, several works (Chen, Liu, and Lian 2019; Li et al. 2019; Xu et al. 2020) follow traditional proposal-based diagrams by generating multiple point anchors and then predicting the relative distance between each lane point and the anchor point. However, these existing CNN-based methods usually need complicated post-processing procedures like non-maximal suppression or clustering. Besides, the fixed receptive field of CNN architecture limits the ability to incorporate relations for long-range lane points. Therefore, our Laneformer utilizes a transformer to capture context information and bipartite matching to eliminate additional post-processing procedures.

Attention-based methods. Several attention-based lane detection models(Lee et al. 2021; Tabelini et al. 2020b; Liu et al. 2021) have been proposed to capture the long-range information. (Lee et al. 2021) propose a self-attention mechanism to predict the lanes' confidence along with the vertical and horizontal directions in an image. (Tabelini et al. 2020b) proposes a novel anchor-based attention mechanism that aggregates global information named LaneATT. However, the

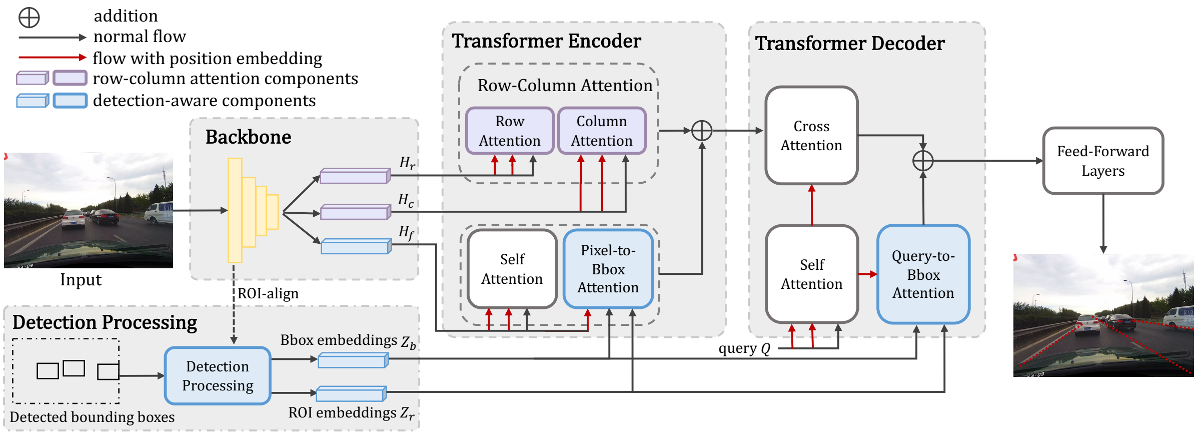


Figure 2: Overall Architecture. In Laneformer, backbone extracts backbone features H_f, row features H_r and column features H_c. Detection Processing module generates bounding box embeddings Z_b and ROI embeddings Z_r with detected bounding boxes and H_f. In the encoder, row attention and column attention are performed on H_r and H_c, added up with the pixel-to-Bbox attention performed on H_f, Z_b and Z_r as memory input for decoder. In the decoder, traditional cross-attention and query-to-Bbox attention are performed and outputs are added up for the feed-forward layers to predict lanes.

fixed attention routines can not adaptively fit the shape characteristic of lanes. Without special design for lane detection, (Xu et al. 2021) adopts Transformers with a multi-scale encoder/decoder strategy to perform line segment detection. Based on PolyLaneNet(Tabelini et al. 2020a), LSTR(Liu et al. 2021) is proposed to output polynomial parameters of a lane shape function by using a network built with a transformer to learn richer structures and context. Unlike LSTR, which assumes all lanes are parallel on the road and formulates it as a polynomial shape prediction problem, our Laneformer directly outputs each lane's points to adapt to more complex lane detection scenarios. Besides, detected object instances are further included as auxiliary inputs of multi-head attention blocks in Laneformer to facilitate the lane point detection by sensing semantic contexts.

Laneformer

Lane Representation

Similar to (Chen, Liu, and Lian 2019), we define lanes as a series of 2D-points that can adapt to all kinds of lanes. Specially, we formulate a lane as $l = (\mathbf{X}, s, e)$, where $\mathbf{X}$ stands for corresponding x-coordinate set for 72 equally-spaced y-coordinates and s, e denote for the start y-coordinate and end y-coordinate of the lane.

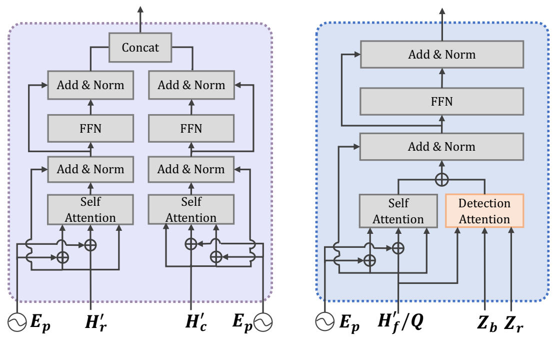
Architecture

The overall architecture of Laneformer is demonstrated in Figure 2. It mainly consists of four modules: a CNN backbone to extract basic features, a detection processing module to handle outputs from person-vehicle detection module, a specially designed encoder and a decoder for lane detection.

Given an RGB image as input, our model first extracts backbone features with a ResNet (He et al. 2016) backbone. We further get row features and column features by collapsing the column dimension and row dimension of backbone features, respectively. At the same time, detected bounding boxes and their predicted scores and categories from the input image are acquired through a trained person-vehicle detection module. We use the bounding box locations to crop ROI-aligned features from backbone features mentioned above, followed by a 1-layer perceptron with ReLU activation to transform the ROI-aligned features to one-dimensional embeddings. Similarly, another 1-layer perceptron is applied on the detected bounding boxes to get one-dimensional bounding box embeddings. In the encoder, row attention and column attention are performed on row features and column features respectively. Meanwhile, backbone features will perform self-attention and pixel-to-Bbox attention with bounding box embeddings and ROI embeddings. The outputs of row-column attention and pixel-to-Bbox attention are added up as the memory input for the following decoder. In the decoder, learnable queries first perform self-attention to get query features. Then the query features together with the input memory will perform cross attention. Meanwhile, with the bounding box embeddings and ROI embeddings, query-to-Bbox attention can be applied. Finally, outputs of the cross attention and query-to-Bbox attention are added up, and several feed-forward layers are utilized to predict lane points.

Detection Processing

After the acquisition of detected bounding boxes, predicted scores and predicted categories from a trained detector based on common Faster-RCNN(Ren et al. 2015) architecture, we propose a simple detection processing module to process them in order to better utilize the detection information. First, we crop ROI-aligned features from backbone features $\mathbf{H_f} \in R^{h \times w \times d}$ for the bounding boxes of the detected objects, where h, w and d are the spatial sizes and the corresponding dimension of backbone features. Since the features



(a) Row-column attention. (b) Detection attention.

Figure 3: Detailed operations of the row-column attention and detection attention. $\bigoplus$ stands for the addition and H'_f/Q denotes that the input can be either backbone feature H'_f in the encoder or query feature Q in the decoder.

we used are down-sampled, we need to rescale the bounding box locations by a specific ratio to crop out correct feature areas. Considering that objects with higher confidence scores may supply more robust information while objects with lower scores may be noisy, we also use predicted scores as weight coefficients to multiply the ROI-aligned features and get weighted ROI-aligned features for each object. After that, we pass through weighted ROI-aligned features into a 1-layer perceptron with output channel d' followed by ReLu activation to get final ROI embeddings $\mathbf{Z_r} \in R^{M\times d'}$. M denotes the number of used detected bounding boxes. If the number of detected bounding boxes is less than M, bounding boxes with random locations, categories and zero scores will be padded. On the other hand, bounding boxes with lower scores will be excluded if the number of detected bounding boxes is more than M.

In the meantime, with the prior that category information can help distinguish different objects, we further concatenate the predicted categories after bounding boxes. Specifically, we use a four-dimensional vector to represent a bounding box and a one-hot vector with length 7 (1 for padded box and 6 for categories) to represent the corresponding category, which will result in an 11-dimensional vector after concatenation. Similar to ROI features, a one-layer perceptron with ReLu activation is applied to get bounding box embeddings $\mathbf{Z_b} \in R^{M\times d'}$. Finally, $\mathbf{Z_r}$ and $\mathbf{Z_b}$ are sent to preform Pixel-to-Bbox attention in encoder and Query-to-Bbox attention in decoder.

Row-Column Attention

In the encoder of Laneformer, besides the traditional attention, we further propose the new row-column attention to efficiently mine point context along with the lane shapes. The row-column attention consists of row attention that captures the relations between rows and column attention that mine the relations between columns. As we all know, the traffic lane is a kind of object with unique shape characteristics. From the vertical view, lanes are long and thin, which means pixels in the same lane will not be far from each other between adjacent rows. From the horizontal view, since different columns may across different lanes, the knowledge sharing of these columns may capture different lane features to construct better representations.

Given row features $\mathbf{H_r} \in R^{h\times 1\times wd}$, where h, w and d are the spatial sizes and the the corresponding dimension of backbone features, we pass it through a linear transformation to reduce the last dimension to d', which we denoted as $\mathbf{H'_r} \in R^{h\times 1\times d'}$. Sinusoidal embeddings $\mathbf{E_p}$ is calculated according to the absolute positions to supply location information. Self-attention operations are applied with the inputs of $\mathbf{H'_r}$ and $\mathbf{E_p}$. Similarly, with the dimension-reduced column features $\mathbf{H'_c} \in R^{1\times w\times d'}$ and its position embeddings, self-attention are performed between columns. Finally, the outputs of row and column attention are added up as the memory input for decoder. Details are shown in Figure 3(a).

Detection Attention

Apart from row-column attention, in both the encoder and decoder of Laneformer, we propose detection attention to mine the valuable information from detected surrounding objects. It is straight-forward that lanes and objects on the road (e.g., pedestrians, vehicles) have some implicit relations with each other. For example, lanes are more likely to appear next to vehicles, while pedestrians are not supposed to walk between lanes. Besides, object detection module such as person-vehicle detection is always co-existed with lane detection in an auto-driving system, thus detected results can be acquired easily. The detection attention module consists of two parts: 1) a Pixel-to-Bbox attention module in encoder to excavate the relevance between each pixel token of feature map and each detected objects; 2) a Query-to-Bbox attention module in decoder to find out which object should be paid more attention in order to help predict corresponding lanes. Details are illustrated in Figure 3(b).

Pixel-to-Bbox Attention In Laneformer encoder, we propose pixel-to-Bbox attention. Pixel-to-Bbox attention is designed for digging out relations between feature pixels and detected objects. In pixel-to-Bbox attention, each pixel in dimension-reduced backbone features $\mathbf{H'_f}$ is considered as a query token. Detection information including bounding box embeddings $\mathbf{Z_b}$ and ROI embeddings $\mathbf{Z_r}$ make up the Key module and Value module, respectively. The pixel-to-Bbox attention is defined as follows:

$$\mathbf{O_{p2b}} = softmax(\frac{\mathbf{H'_f}\mathbf{Z_b}^T}{\sqrt{d'}}) \cdot \mathbf{Z_r} \tag{1}$$

Pixel-to-Bbox attention forces the model to learn which detected object a pixel should pay attention to so that the model can catch helpful context features. The output of pixel-to-Bbox attention $\mathbf{O_{p2b}}$ are added up with the above row-column attention output and act as the memory input for the decoder.

Query-to-Bbox Attention In Laneformer decoder, we propose query-to-Bbox attention. Query-to-Bbox attention on the other hand designed to mine relations between queries and detected objects. Query-to-Bbox attention is similar to Pixel-to-Bbox attention, while in here, queries are the learned embeddings $\mathbf{Q}$ with the size of $N \times d'$, where N is the number of learned embeddings. Bounding box embeddings $\mathbf{Z_b}$ and ROI embeddings $\mathbf{Z_r}$ still act as Key module and Value module here. Similarly, the query-to-Bbox attention is defined as follows:

$$\mathbf{O_{q2b}} = softmax(\frac{\mathbf{QZ_b^T}}{\sqrt{d'}}) \cdot \mathbf{Z_r} \tag{2}$$

This attention enables each query to focus on the instances near the lane it needs to predict. The output of query-to-Bbox attention $\mathbf{O_{q2b}}$ will be added up with the traditional cross-attention output, followed by several feed-forward features to get predicted lanes.

Loss Construction

Bipartite matching. After obtaining features from above-mentioned modules, Laneformer predicts a number of N lane set according to its number of queries, where N is set to be significantly larger than the maxinum number of lanes in the dataset. Therefore we need to pad the ground truth with non-lane to be a set of N objects first, denoted as $\mathbf{G} = \{g_n | g_n = (c_n, l_n)\}_{n=1}^{N}$, where $c_n \in \{0, 1\}$, 0 represents non-lane and 1 represents lane. Given predicted outputs as $\mathbf{P} = \{p_n | p_n = (\hat{p}_n, \hat{l}_n)\}_{n=1}^{N}$, where $\hat{p}_n(c_n)$ stands for the probability score that $\hat{l}_n$ belongs to the specific category c_n. We search for a permutation of N elements matching index δ to minimize the pair-wise distance function D between ground-truth lane g_n and predicted lane $p_{\delta(n)}$:

$$\hat{\delta} = \arg\min_{\delta} \sum_{n=1}^{N} D(g_n, p_{\delta(n)}). \tag{3}$$

The difference function D is defined as following:

$$D(g_n, p_{\delta(n)}) = -\omega_1 \hat{p}_{\delta(n)}(c_n) + \mathbb{1}(c_n = 1)\mathcal{L}_{loc}(l_n, \hat{l}_{\delta(n)}) \tag{4}$$

Where $\mathbb{1}(*)$ denotes an indicator function and ω_1 stands for the coefficient of classification term. $\mathcal{L}_{loc}$ is defined as follows:

$$\begin{aligned}\mathcal{L}_{loc}(l_n, \hat{l}_{\delta(n)}) = &\omega_2 L_1(X_n, \hat{X}_{\delta(n)}) + \omega_3 L_1(s_n, \hat{s}_{\delta(n)}) \\ &+ \omega_4 L_1(e_n, \hat{e}_{\delta(n)})\end{aligned} \tag{5}$$

where L_1 denotes the mean absolute error and $\omega_2, \omega_3, \omega_4$ indicate for the coefficients for point, start position and end position term respectively. Bipartite matching is adopted to ensure one-to-one alignment between predictions and ground truths, making the Laneformer an end-to-end architecture by eliminating additional post-processing procedures.

Total Loss The total loss of our model is calculated with matching index δ gained from bipartite matching, consisting of negative log-likelihood loss for classification prediction and $L1$-based location loss:

$$\mathcal{L}_{total} = \sum_{n=1}^{N} -\omega_1 log \hat{p}_{\delta(n)}(c_n) + \mathbb{1}(c_n = 1)\mathcal{L}_{loc}(l_n, \hat{l}_{\delta(n)}) \tag{6}$$

Where $\mathcal{L}_{loc}$ is calculated the same with Eq.(5) and $(n, \delta(n))$ is the optimal pair indexes that minimize Eq.(3). ω_1, ω_2, ω_3, ω_4 also adjust the effect of the loss terms and are set as same values with Eq.(4) and Eq.(5).

Experiment

Datasets and Evaluation Metrics

We conduct experiments on the two most popular lane detection benchmarks. **CULane** (Pan et al. 2018) is a large-scale traffic lane detection dataset that is collected by in-vehicle cameras in Beijing, China. It consists of 88,880 training images, 9675 validation images, and 34,680 test images. The test split is further divided into normal and 8 challenging categories. **TuSimple** (TuSimple 2017) is an autonomous driving dataset which specifically focuses on real highway scenarios, including 3626 images for training set and 2782 images for the test set.

Evaluation Metrics We use F1 score(abbreviated as F1 in the following section) to measure the model performance on CULane: $F_1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$, where $Precision = \frac{TP}{TP+FP}$ and $Recall = \frac{TP}{TP+FN}$. As for Tusimple, standard evaluation metrics including Accuracy, false positives(FP) and false negatives(FN) are adopted.

Implementation Details

The input resolution is set to 820×295 for CULane and 640×360 for TuSimple. Data augmentations are applied on the raw image, consisting of horizontal flipping, a random choice from color-shifting operations(e.g., gaussian blur, linear contrast) and position-shifting operations(e.g., cropping, rotate). Most of our experiments use ResNet50 as the backbone. We follow the setting of (Zhu et al. 2020) and utilize a deformable transformer as the plain transformer. The bipartite matching and loss term coefficients ω_1, ω_2, ω_3 and ω_4 are set as 2, 10, 10, 10, respectively. Both the number of encoder and decoder layers is set to 1. Moreover, we adopt 25 as the number of queries N and 10 as the number of used detected bounding boxes M. Eight V100s are used to train the model and the batch size is set to be 64. The learning rate is set to 1e-4 for the backbone and 1e-5 for the transformer. We train 100 epochs on CULane and drop the learning rate by ten at 80 epoch. On Tusimple, the total number of training iterations is set to 28k and the learning rate drops at 22k iteration. During inference, the single scale test is adopted with the score threshold set to 0.8. The trained detector used to obtain person-vehicle bounding boxes is based on common Faster-RCNN(Ren et al. 2015) architecture with ResNet-50 backbone and trained 12 epochs on BDD100K dataset(Yu et al. 2018) with 70k images.

Main Results

CULane. Table 1 shows the Laneformer's performance on CULane test set. Our Laneformer achieves state-of-the-art results on F1 of the total test split. In addition, our Lanformer with only ResNet-50 backbone even surpasses the results of LaneATT (Tabelini et al. 2020b) with a larger ResNet-122 backbone. Additionally, Laneformer outperforms all

Methods	Normal	Crowded	Dazzle	Shadow	No line	Arrow	Curve	Night	Cross	**Total**	MACs (G)
SCNN(Pan et al. 2018)	90.60	69.70	58.50	66.90	43.40	84.10	64.40	66.10	1990	71.60	/
ENet-SAD(Hou et al. 2019)	90.10	68.80	60.20	65.90	41.60	84.00	65.70	66.00	1998	70.80	/
PointLane(Chen, Liu, and Lian 2019)	88.00	68.10	61.50	63.30	44.00	80.90	65.20	63.20	1640	70.20	/
ERFNet-HESA(Lee et al. 2021)	92.00	73.10	63.80	75.00	45.00	88.20	67.90	69.20	2028	74.20	/
CurveLane-S(Xu et al. 2020)	88.30	68.60	63.20	68.00	47.90	82.50	66.00	66.20	2817	71.40	**9.0**
CurveLane-M(Xu et al. 2020)	90.20	70.50	65.90	69.30	48.80	85.70	67.50	68.20	2359	73.50	33.7
CurveLane-L(Xu et al. 2020)	90.70	72.30	67.70	70.10	49.40	85.80	**68.40**	68.90	1746	74.80	86.5
LaneATT(RN-18)(Tabelini et al. 2020b)	91.17	72.71	65.82	68.03	49.13	87.82	63.75	68.58	1020	75.13	9.3
LaneATT(RN-34)(Tabelini et al. 2020b)	**92.14**	75.03	66.47	**78.15**	49.39	**88.38**	67.72	70.72	1330	76.68	18.0
LaneATT(RN-122)(Tabelini et al. 2020b)	91.74	**76.16**	69.47	76.31	**50.46**	86.29	64.05	70.81	1264	77.02	70.5
Laneformer(RN-50)*	91.55	74.76	69.27	69.59	48.13	86.99	68.15	70.06	1104	76.04	26.2
Laneformer(RN-18)	88.60	69.02	64.07	65.02	45.00	81.55	60.46	64.76	25	71.71	13.8
Laneformer(RN-34)	90.74	72.31	69.12	71.57	47.37	85.07	65.90	67.77	26	74.70	23.0
Laneformer(RN-50)	91.77	75.41	**70.17**	75.75	48.73	87.65	66.33	**71.04**	**19**	**77.06**	26.2

Table 1: Comparison of F1-measure(%) and MACs(multiply–accumulate operations) on CULane testing set, where Laneformer* denotes for Laneformer without detection attention module. Our Laneformer achieves state-of-the-art performance.

other lane detection models on some challenging splits such as "Night", "Dazzle light" and "Cross". Among the above three splits, **Laneformer significantly improves the performance of "Cross" category, which achieves an extremely low FP(False Positive) number 19.** We observe that our model gets a result of 1104 FP on "Cross" category without detection attention, while with the detection attention, the performance makes a dramatic improvement as in Table 1. The promotion may come from the perception of vehicle-person global context on the particular cross-road scenario benefiting from the proposed detection attention layer.

TuSimple. Experimental results on TuSimple benchmark are summarized in Table 2. We achieve 96.8% accuracy, 5.6% FP and 1.99% FN with ResNet-50 backbone. It is shown that Laneformer achieves comparable accuracy with the state-of-the-art Line-CNN and 0.6% higher than another transformer-based method LSTR. Even with the smaller backbones (ResNet-18, ResNet-34), Laneformer can achieve a competitive accuracy compared with the state-of-the-art methods. Note that adding detection attention on Tusimple doesn't improve much due to the relatively simple highway driving scenes (e.g., few cars and straight lines).

Latency Comparison. In inference, Laneformer with ResNet-50 backbone achieves 53 and 48 FPS on one V100 for CULane and Tusimple benchmark. For latency comparison of different components in Laneformer, we conduct experiments on CULane testing split and illustrate the result in Table 3. After adding row-column attention and detection attention, there is only a 4.9%, 8.1% increment on inference FPS due to the efficient matrix multiplication.

Visualization. We visualize several attention maps in the transformer to find out the area that detection attentions and row-column attentions focus on, where brighter color denotes for more significant attention value. Shown in Figure 4(a), either the point or query pays more attention to the detected instances besides lanes it responsible for, especially when those instances are in occlusion with part of the lane. Moreover, observation can be made in Figure 4(b) that row attention mainly considers nearby rows, while the column attention focuses on the nearby representative column of each lane. These results demonstrate our assumption that the implicit relationship of traffic scenes can be obtained from proposed detection attention and row-column attention.

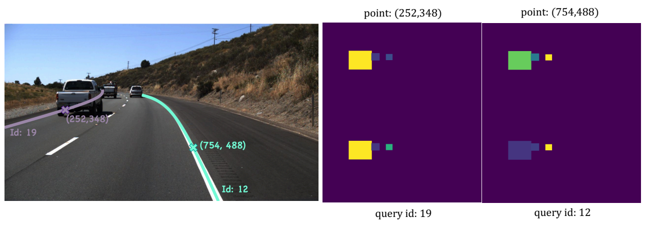
Method	Acc(%)	FP(%)	FN(%)
SCNN (Pan et al. 2018)	96.53	6.17	**1.80**
LSTR (Liu et al. 2021)	96.18	**2.91**	3.38
Enet-SAD (Hou et al. 2019)	96.64	6.02	2.05
Line-CNN (Li et al. 2019)	**96.87**	4.41	3.36
PolyLaneNet (Tabelini et al. 2020a)	93.36	9.42	9.33
PointLaneNet (Chen, Liu, and Lian 2019)	96.34	4.67	5.18
LaneATT (RN-18) (Tabelini et al. 2020b)	95.57	3.56	3.01
LaneATT (RN-34) (Tabelini et al. 2020b)	95.63	3.53	2.92
LaneATT (RN-122) (Tabelini et al. 2020b)	96.10	5.64	2.17
Laneformer(RN-50)*	96.72	3.46	2.52
Laneformer(RN-18)	96.54	4.35	2.36
Laneformer(RN-34)	96.56	5.39	3.37
Laneformer(RN-50)	**96.80**	5.60	1.99

Table 2: Comparison of different algorithms on the Tusimple testing benchmark, where Laneformer* denotes for Laneformer without detection attention module.

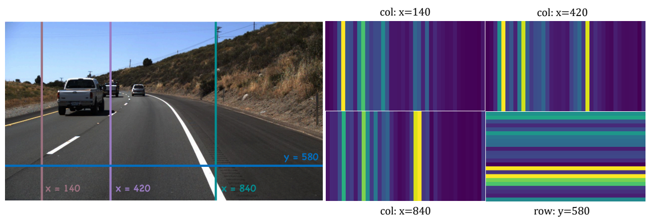
Ablation Study

Different components. As we can see in Table 3, without detection attention and row-column attention, our baseline (plain transformer) obtains 75.45% on F1. The adding of row-column attention improves the performance to 76.04%. What's more, simply introducing detected objects information can improve the performance of our model, but a little more extra information such as scores or categories will make it better. We can observe that making full use of detected objects with their scores and categories raises the F1 to 77.06%, which is the state-of-the-art results on CULane.

Score threshold of the bounding box. To find out the influences of detected objects with different confidence, we



(a) Visualization of detection attention.



(b) Visualization of row and column attention.

Figure 4: Visualization of proposed attentions on Tusimple dataset, where brighter color denotes more significant attention value. (a) shows the detection attention map for different points and queries in the transformer. We can observe either the pixel or the query focus on the detection bounding boxes near the corresponding lane. (b) shows that the row attention mainly considers the nearby rows, while the column attention focuses on the nearby representative column of each lane.

Model	F1(%)	Precision(%)	Recall(%)
Baseline(ResNet-50)	75.45	81.65	70.11
+ row-column attention	76.04	82.92	70.22
+ bounding box	76.08	85.30	68.66
+ score	76.25	83.56	70.12
+ category	77.06	84.05	71.14

Table 3: Ablation study results of different components of Laneformer on CULane.

conduct a series of experiments with different score thresholds on filtering detected bounding boxes. Table 4 shows that using detection outputs from different score thresholds has a slight impact on our results. Especially, results between threshold 0.6 to 0.8 are robust and the differences can be nearly neglect. When the threshold is lower than 0.6, there may be some noise in detected bounding boxes and therefore our performance gets slighted hurt. On the other hand, if the threshold is too high such as 0.9, only a few bounding boxes will be chosen so that a large number of padded boxes will interfere with the model's learning, which leads to a lower F1. Besides, we experiment on random bounding boxes to prove that Laneformer indeed makes use of the information of detected objects.

Number of bounding box. Apart from thresholds, we further explore the impact of using a different number of detected objects in Laneformer. The results in Table 4 show that too many detected objects lead to a performance drop, and our model reaches the best result under the setting of 10 bounding boxes. We speculate that the best setting of 10 bounding boxes is due to the average number of detected objects in each image under the bounding box threshold of 0.6, which is 9.84. So if we set a number much larger than 9.84, for example, 20 in our experiment, then too many useless padded boxes will be used, which may hurt the model's performance. On the other hand, if we use too few bounding boxes, the information of detected objects is not entirely utilized to reach the best performance.

Different categories. Vehicles and persons have different relations with lanes. To be specific, vehicles are on the road, beside the lanes, while persons usually far from lanes. So we also explore the impact of extra information with different categories of bounding boxes. Results in Table 4 show that both the adding of vehicles and persons can improve the model performance, and the result with vehicles is better than the one with persons. We suppose that vehicles share a more close relation with lanes in the perspective of locations. Experiments show that the model with all of the two categories reaches the best results.

		F1(%)	Pr(%)	Re(%)
Score threshold	0.4	76.34	85.41	69.01
	0.5	76.14	84.52	69.27
	0.6	**77.06**	84.05	**71.14**
	0.7	76.71	84.53	70.21
	0.8	76.85	84.51	70.46
	0.9	76.37	**85.54**	68.98
	random	75.99	84.50	69.03
Number of Bbox	5	76.90	84.30	70.69
	10	**77.06**	84.05	**71.14**
	20	76.51	**84.34**	70.01
Different categories	none	76.04	82.92	70.22
	person	76.40	**85.17**	69.27
	vehicle	76.79	84.44	70.41
	all	**77.06**	84.05	**71.14**

Table 4: Quantitative evaluation of different detection bounding box input settings on CULane testing split. 'Pr' and 'Re' denote for the Precision and Recall respectively.

Conclusions

In this work, we propose Laneformer, a conceptually simple yet powerful transformer-based architecture tailored for lane detection. Equipped with row and column self-attention module and semantic contexts provided by additional detected object instances, Laneformer achieves the state-of-the-art performance, in terms of 77.1% F1 score on CULane and superior 96.8% Accuracy on TuSimple benchmark. Besides, visualization of the learned attention map in transformer demonstrates that our Laneformer can incorporate relations of long-range lane points and global contexts induced by surrounding objects.

References

Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; and Zagoruyko, S. 2020. End-to-end object detection with transformers. In *European Conference on Computer Vision*, 213–229. Springer.

Chen, Z.; Liu, Q.; and Lian, C. 2019. PointLaneNet: Efficient end-to-end CNNs for Accurate Real-Time Lane Detection. In *IV*, 2563–2568. IEEE.

Chiu, K.-Y.; and Lin, S.-F. 2005. Lane detection using color-based segmentation. In *IV*, 706–711. IEEE.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Gonzalez, J. P.; and Ozguner, U. 2000. Lane detection using histogram-based segmentation and decision trees. In *ITSC2000. 2000 IEEE Intelligent Transportation Systems. Proceedings (Cat. No. 00TH8493)*, 346–351. IEEE.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*.

Hou, Y. 2019. Agnostic Lane Detection. *CoRR*, abs/1905.03704.

Hou, Y.; Ma, Z.; Liu, C.; and Loy, C. C. 2019. Learning lightweight lane detection cnns by self attention distillation. *arXiv preprint arXiv:1908.00821*.

Lee, J.-W.; and Cho, J.-S. 2009. Effective lane detection and tracking method using statistical modeling of color and lane edge-orientation. In *2009 Fourth International Conference on Computer Sciences and Convergence Information Technology*, 1586–1591. IEEE.

Lee, M.; Lee, J.; Lee, D.; Kim, W.; Hwang, S.; and Lee, S. 2021. Robust Lane Detection via Expanded Self Attention. *arXiv preprint arXiv:2102.07037*.

Li, X.; Li, J.; Hu, X.; and Yang, J. 2019. Line-CNN: End-to-End Traffic Line Detection With Line Proposal Unit. *IEEE Transactions on Intelligent Transportation Systems*.

Liu, R.; Yuan, Z.; Liu, T.; and Xiong, Z. 2021. End-to-end lane shape prediction with transformers. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 3694–3702.

Mamidala, R. S.; Uthkota, U.; Shankar, M. B.; Antony, A. J.; and Narasimhadhan, A. V. 2019. Dynamic Approach for Lane Detection using Google Street View and CNN. *CoRR*, abs/1909.00798.

Narote, S. P.; Bhujbal, P. N.; Narote, A. S.; and Dhane, D. M. 2018. A review of recent advances in lane detection and departure warning system. *Pattern Recognition*, 73: 216–234.

Niu, J.; Lu, J.; Xu, M.; Lv, P.; and Zhao, X. 2016. Robust lane detection using two-stage feature extraction with curve fitting. *Pattern Recognition*, 59: 225–233.

Pan, X.; Shi, J.; Luo, P.; Wang, X.; and Tang, X. 2018. Spatial as deep: Spatial cnn for traffic scene understanding. In *AAAI*.

Pizzati, F.; Allodi, M.; Barrera, A.; and García, F. 2019. Lane Detection and Classification using Cascaded CNNs. *CoRR*, abs/1907.01294.

Radford, A.; Narasimhan, K.; Salimans, T.; and Sutskever, I. 2018. Improving language understanding by generative pre-training. https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*.

Tabelini, L.; Berriel, R.; Paixao, T. M.; Badue, C.; De Souza, A. F.; and Oliveira-Santos, T. 2020a. PolyLaneNet: Lane estimation via deep polynomial regression. *arXiv preprint arXiv:2004.10924*.

Tabelini, L.; Berriel, R.; Paixão, T. M.; Badue, C.; De Souza, A. F.; and Olivera-Santos, T. 2020b. Keep your Eyes on the Lane: Attention-guided Lane Detection. *arXiv preprint arXiv:2010.12035*.

TuSimple. 2017. Tusimple lane detection challenge. In *CVPR Workshops*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *NIPS*.

Wang, H.; Zhu, Y.; Adam, H.; Yuille, A.; and Chen, L.-C. 2020. MaX-DeepLab: End-to-End Panoptic Segmentation with Mask Transformers. *arXiv preprint arXiv:2012.00759*.

Wang, Y.; Shen, D.; and Teoh, E. K. 2000. Lane detection using spline model. *Pattern Recognition Letters*, 21(8): 677–689.

Xu, H.; Wang, S.; Cai, X.; Zhang, W.; Liang, X.; and Li, Z. 2020. CurveLane-NAS: Unifying Lane-Sensitive Architecture Search and Adaptive Point Blending. *arXiv preprint arXiv:2007.12147*.

Xu, Y.; Xu, W.; Cheung, D.; and Tu, Z. 2021. Line segment detection using transformers without edges. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4257–4266.

Young, T.; Hazarika, D.; Poria, S.; and Cambria, E. 2018. Recent trends in deep learning based natural language processing. *ieee Computational intelligenCe magazine*, 13(3): 55–75.

Yu, F.; Xian, W.; Chen, Y.; Liu, F.; Liao, M.; Madhavan, V.; and Darrell, T. 2018. BDD100K: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*.

Zheng, S.; Lu, J.; Zhao, H.; Zhu, X.; Luo, Z.; Wang, Y.; Fu, Y.; Feng, J.; Xiang, T.; Torr, P. H.; et al. 2020. Rethinking Semantic Segmentation from a Sequence-to-Sequence Perspective with Transformers. *arXiv preprint arXiv:2012.15840*.

Zhu, X.; Su, W.; Lu, L.; Li, B.; Wang, X.; and Dai, J. 2020. Deformable DETR: Deformable Transformers for End-to-End Object Detection. *arXiv preprint arXiv:2010.04159*.

Zou, Q.; Jiang, H.; Dai, Q.; Yue, Y.; Chen, L.; and Wang, Q. 2019. Robust Lane Detection from Continuous Driving Scenes Using Deep Neural Networks. *CoRR*, abs/1903.02193.