Lane2Seq: Towards Unified Lane Detection via Sequence Generation

Kunyang Zhou
Southeast University
kunyangzhou@seu.edu.cn

Abstract

In this paper, we present a novel sequence generation-based framework for lane detection, called Lane2Seq. It unifies various lane detection formats by casting lane detection as a sequence generation task. This is different from previous lane detection methods, which depend on well-designed task-specific head networks and corresponding loss functions. Lane2Seq only adopts a plain transformer-based encoder-decoder architecture with a simple cross-entropy loss. Additionally, we propose a new multi-format model tuning based on reinforcement learning to incorporate the task-specific knowledge into Lane2Seq. Experimental results demonstrate that such a simple sequence generation paradigm not only unifies lane detection but also achieves competitive performance on benchmarks. For example, Lane2Seq gets 97.95% and 97.42% F1 score on Tusimple and LLAMAS datasets, establishing a new state-of-the-art result for two benchmarks.

1. Introduction

Lane detection is a fundamental task in computer vision [31, 36, 40]. It aims to predict the location of the lane in a given image. Lane detection plays a crucial role in many applications, such as adaptive cruise control and lane keeping. Existing lane detection methods generally adopt a divide-and-conquer strategy, which decomposes the lane detection into multiple subtasks. Each subtask is accomplished by a task-specific head network. For instance, as shown in Fig. 1, segmentation-based methods [28,49] adopt a head network along with a task-specific module, such as the message-passing module [28], to predict per-pixel masks. Anchor-based methods [17, 36] utilize a classification head network for distinguishing lane instances and an anchor refinement network for regressing accurate lanes. Parameter-based methods [13, 23] use a network to predict the parameters of lanes and a vertical offset prediction network to locate the start points of lanes.

Although divide-and-conquer strategy has been proved an effectively way to address the certain subtask in the ex-

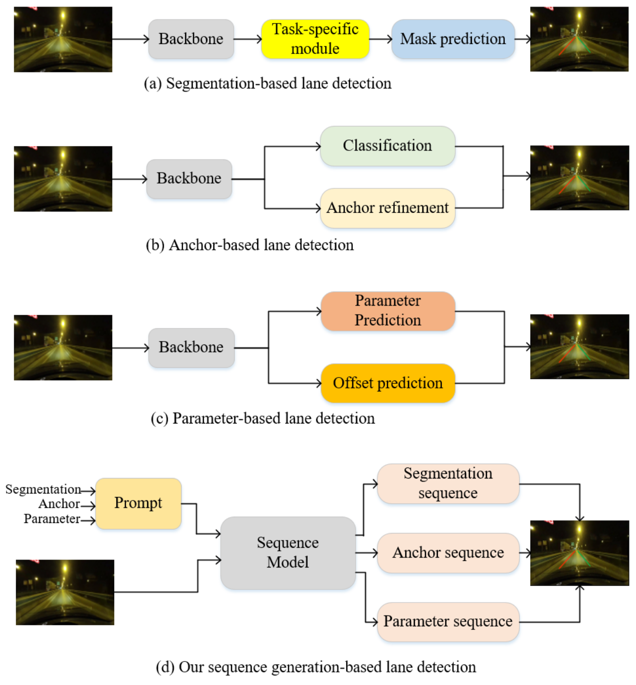

Figure 1. Comparison of different lane detection frameworks.

isting methods, there still exist several limitations. (1) Each subtask needs a customized task-specific head network, resulting in a complicated lane detection model. (2) Each task-specific head requires one or more loss functions, e.g., cross-entropy loss and Line-IOU loss [17], bringing out extra hyper-parameters.

In this paper, we present a novel sequence generation-based lane detecton (Lane2Seq) framework to tackle the aforementioned issues. By formulating the lane detection as a sequence generation task, Lane2Seq gets rid of customized head networks and task-specific loss function. It is based on the intuition that if the detection model knows where the target lane is, model can be simply teached how to read the location of lane out, instead of designing additional classification head or regression head by the divide-and-conquer strategy.

Therefore, we convert the outputs of different lane detection formats, i.e., segmentation-based, anchor-based, and parameter-based, into a sequence of discrete tokens and the model learns to generate this sequence token-by-token. As shown in Fig. 1 (d), to achieve a specific lane detection format, Lane2Seq uses a prompt to specify the detection format and the generated sequence adapts to the prompt so the model can produce format-specific output. By the format-specific prompt, Lane2Seq unifies different lane detection formats into a model.

While Lane2Seq does not contain task-specific components, task-specific knowledge contained in these components can help the model learn the features of lanes better. We propose a Multi-Format model tuning method based on Reinforcement Learning (MFRL) to incorporate the task-specific knowledge into the model without changing model's architecture. Inspired by Task-Reward [30], MFRL takes the evaluation metrics, which naturally integrates task-specific knowledge, as the reward and tunes Lane2Seq using REINFORCE [41] algorithm. However, evaluation metrics like F1 score cannot be used as the reward directly due to undecomposable as a sum of per-example rewards. In this paper, we propose three new evaluation metric-based rewards for segmentation, anchor, and parameter format, based on their task-specific knowledge.

Experimental results demonstrate that our Lane2Seq achieves competitive performance on three public datasets, Tusimple, CULane, and LLAMAS. For instance, Lane2Seq with ViT-Base encoder obtains 97.95% and 97.42% F1 score on Tusimple and LLAMAS, setting a new state-of-the-art result for two datasets. It should be noted that all existing lane detection methods heavily rely on well-designed task-specific head networks and the corresponding complicated loss functions. Instead, our Lane2Seq utilizes a plain transformer-based encoder-decoder architecture with a simple cross-entropy loss.

The main contributions of this paper are as follows.

- We propose a sequence generation-based method for lane detection, which casts the lane detection as a sequence generation task. To the best of our knowledge, we are the first to unify the lane detection through the sequence generation, which offers a new perspective on lane detection.

- We present a novel reinforcement learning-based multi-format model tuning, including three new evaluation metric-based reward functions, to incorporate the task-specific knowledge into the model.

- Experimental results show that our method achieves competitive performance on lane detection benchmarks. Remarkably, we establish a new state-of-the-art result on Tusimple and LLAMAS.

2. Related Work

Lane detection. Existing lane detection methods can be divided into three categories based on the representation of the lane: segmentation-based method, anchor-based method and parameter-based method. Segmentation-based methods [28, 47, 49] consider lane detection as a semantic segmentation task and performs pixel-wise prediction. SCNN [28] enhances the visual evidence by a message-passing module, which can capture spatial dependency for lanes. Anchor-based methods [20,31,36,51] predict the accurate lanes by refining the predefined lane anchors. UFLD [31] proposes a novel row anchor-based approach to detect lanes. Different from the segmentation-based and anchor-based methods, parameter-based methods [13, 23, 37] regard the lane detection as the parametric modeling and regress the parameters of the lane. PolyLaneNet [37] models a lane curve as a polynomial and regresses the parameters of the polynomial. In this paper, we treat the segmentation-based, anchor-based methods, and parameter-based metohds uniformly as the sequence generation task, getting rid of the complicate structure and task-specific modules, e.g., head network. It only adopts the cross-entropy loss and plain transformer architecture.

Sequence generation for vision tasks. Recently, sequence-to-sequence (seq2seq) method used in the natural language processing (NLP) has been applied for vision tasks. Seq2seq utilizes a basic transformer encoder and decoder architecture and accomplishes the task by making sequence predictions, rather than designing a model tailored to the vision task. Pix2seq [8] is the pioneering work to cast the object detection as the sequence generation task. It shows that object can be detected well without any task-specific modules like label assignment. Besides object detection, seq2seq has been extended to other vision tasks such as instance segmentation [9], keypoint detection [9], text spotting [19,29] and object tracking [10]. UniTAB [45] adopts the task prompt [33] to perform multi-task learning. Unified-IO [25] jointly trains various vision tasks by setting the unified input/output formats for all tasks. Besides, seq2seq becomes increasingly popular in the multi-modal model. Text-to-image models like DALL-E [34] and vision-language models like Flamingo [2] all use seq2seq to unify the multi-modal tasks. Nevertheless, how to perform the seq2seq in the lane detection to unify different detection formats is still unexplored.

Lane2Seq has a similar spirit with the Pix2seq and its successors [9, 10, 19, 29]. All of them view the vision tasks as the sequence generation. The main difference between these methods and Lane2Seq is the sequence format. Previous works use coordinates and category to construct the sequence, while Lane2Seq also adopts the parameter of the lane to construct the sequence.

Reinforcement learning in the computer vision. Many

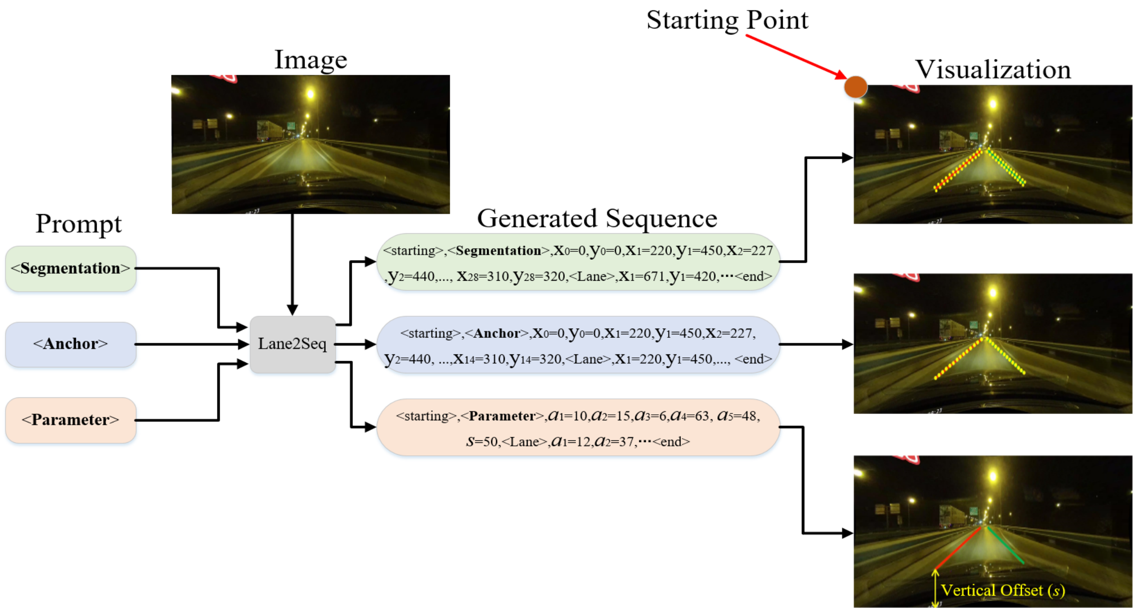
Starting Point
Image
Visualization
Prompt
Generated Sequence
<Segmentation>
<Anchor>
<Parameter>
Lane2Seq
<starting>,<Segmentation>,x0=0,y0=0,x1=220,y1=450,x2=227,y2=440,..., x28=310,y28=320,<Lane>,x1=671,y1=420,⋯<end>
<starting>,<Anchor>,x0=0,y0=0,x1=220,y1=450,x2=227,y2=440, ...,x14=310,y14=320,<Lane>,x1=220,y1=450,..., <end>
<starting>,<Parameter>,a1=10,a2=15,a3=6,a4=63, a5=48,s=50,<Lane>,a1=12,a2=37,⋯<end>
Vertical Offset (s)

Figure 2. Inference pipeline of Lane2Seq. The model perceives input image, prompt and generates format-specific tokens, which can be detokenized into required detection format for visualization.

previously researches introduce the reinforcement learning for the vision tasks, such as object detection [27], object tracking [26], image segmentation [21] and lane detection [48]. They generally focus on learning different parts of an image and refine the outputs iteratively. DQLL [48] localizes the lanes as a group of lanemarks and refine the location of lanemarks with the deep Q-learning. Recently, Task Reward [30] adopts a novel task risk-based reinforcement learning to tune the vision model without changing the model architecture. Reward functions in Task-Reward are designed for object detection, instance segmentation, colorization, and image captioning but not for lane detection. Our MFRL is the first attempt to design effective evaluation metric-based reward functions for different lane detection formats.

3. Method

In this section, we present the proposed sequence generation-based lane detection method, named Lane2Seq, in detail. The overall pipeline of Lane2Seq is illustrated in Fig. 2. Sec 3.1 depicts the unified interface for lane detection, Sec 3.2 details the model architecture and objective function, and Sec 3.3 introduces the proposed multi-format model tuning based on reinforcement learning.

3.1. Unified Interface for Lane Detection

As presented in Figure 1, lane detection formats in the existing method are diverse and have been formulated quite differently. Considering differences in the form of outputs, customized models with specialized head networks and loss functions are designed for different lane detection formats.

To integrate different lane detection formats into a model, we propose an unified sequence interface for lane detection, where both format transcription like segmentation and outputs are treated as sequences of discrete tokens. As shown in Fig. 2, the generated sequence is composed of four parts, a starting token <starting>, a format transcription token like <Segmentation>, a format-specific sequence, and an ending token <end>. The format-specific sequence for three detection formats can be constructed as follows.

Segmentation sequence. Instead of performing pixel-wise mask prediction, we predict the polygon [5] corresponding to the mask as a sequence of coordinates conditioned on a given lane instance. Then, we convert the polygon into a sequence by quantizing the coordinates of its points into discrete tokens. Specifically, x, y coordinates of a point are normalized to the width and height of the image, and then quantized to $[1, n_{bins}]$, where n_{bins} is the size of the vocabulary. Vocabulary will be described subsequently. A polygon sequence can be expressed as $[x_1, y_1, x_2, y_2, ..., x_{28}, y_{28}$,<Lane>], where <Lane> is the category token. If there are multiple lanes in an image, we concatenate the all polygon sequences. The segmentation sequence consists of a starting point and all polygon sequences, where starting point is the left-top point of the image $x_0 = 0, y_0 = 0$.

Anchor sequence. Since the essence of anchor-based methods is regressing the location of keypoints of lane anchors, we treat the anchor prediction as the keypoint sequence generation. Specifically, the

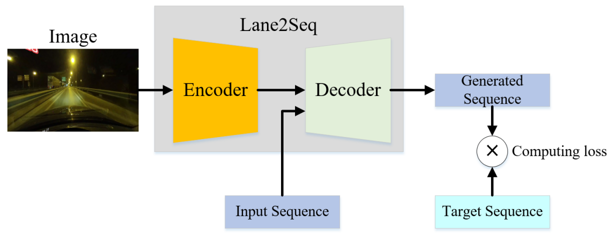

Figure 3. Training pipeline of Lane2Seq. The input sequence can be the segmentation sequence or anchor sequence or parameter sequence.

keypoint sequence of a lane can be expressed as $[x_1, y_1, x_2, y_2, ..., x_{14}, y_{14}$,<Lane>]. The normalization and quantization of keypoint coordinates are same to that of polygon point. The anchor sequence contains a starting point and all keypoint sequences, where starting point is still the left-top point of the image.

Parameter sequence. the sequence of the lane includes two parts: the parameter of polynomial function and vertical offsets. We set the polynomial degree as 5th. Specifically, the parameter sequence of a lane can be represented as $[a_1, a_2, a_3, a_4, a_5, s,$ <Lane>], where s is the vertical offset. s is normalized to the height of the image and quantized to $[1, n_{bins}]$. We use sigmoid to normalize a_i by $a_i = Sigmoid(a_i)$. Then a_i is quantized to $[1, n_{bins}]$. Parameter sequence does not require the starting point.

Vocabulary. We use a shared vocabulary for all formats and each integer between $[1, n_{bins}]$ can be regarded as a word in the vocabulary. Each word in the vocabulary corresponds to a learnable embedding.

3.2. Unified Architecture and Objective Function

We follow [10] and use a transformer-based encoder-decoder architecture to deal with image input and sequence output flexibly. As shown in Fig. 3, the image encoder takes the pixels as the input and outputs the corresponding image representations. We utilize the Vision Transformer (ViT) [11] to instantiate the image encoder. We adopt a transformer-based sequence decoder, which is widely used in the language modeling [32, 33], to generate the output sequence. The decoder generates one token at a time based on the preceding tokens and the image representations. Sequence decoder removes the well-designed task-specific heads for different detection formats.

Training Pipeline. We first tokenize annotation of each format into the corresponding sequence. Then, we construct a training batch using images and sequences from all detection formats. Finally, we compute loss between generated sequence and target sequence for each format. For a certain detection format, the input sequence is [<starting>, format transcription token, format-specific sequence] and the target sequence is [format transcription token, format-specific sequence, <end>]. For example, in the anchor format, the input sequence is [<starting>, <Anchor>, x_0, y_0, x_1, y_1,..., <Lane>, <end>] and the target sequence is [<Anchor>, x_0, y_0, x_1, y_1,..., <Lane>, <end>]. The starting token <starting> and ending token <end> are learnable embeddings, which tell the decoder when to begin and end the sequence generation. The sequence decoder perceives the image representations and input sequence and reconstructs the target sequence.

Objective Function. Similar to Pix2seq [8], we train the Lane2Seq with a simple cross-entropy loss. At each time stamp j, Lane2Seq aims to maximize the likehood of the target tokens conditioned on the image representations I and previously generated tokens $y_{1:j-1}$,

$$Loss_{obj} = -\sum_{j=1}^{N} w_j \log P(\hat{m}_j | I, m_{1:j-1}), \tag{1}$$

where m and $\hat{m}$ denote the input and target sequence, respectively. N is the length of sequence. w_j represents the weight for the j-th token, 0 is assigned to the format transcription token and 1 is assigned to other tokens to ensure the model is trained to predict the desired tokens but not the format transcription tokens.

Inference Pipeline. We take the format transcription token as the prompt to achieve the format-specific detection and the sequence decoder generates the rest of the sequence conditioned on the prompt and image representation. Once the whole sequence is generated, we de-quantize the sequence to obtain the location of lane. More details about de-quantization for each format can be found in the supplementary materials. Inference process is presented in Fig. 2.

3.3. Multi-Format Model Tuning Based on Reinforcement Learning

Lane2Seq get rids of the complicated task-specific components, such as task-specific head networks, via the sequence generation. However, this task-agnostic architecture inevitably lacks the task-specific knowledge of the lane detection, making the model ineffective in learning the features of lane. Exising lane detection methods [17, 49] usually design the specific modules to incorporate the task-specific knowledge into the model. But this way makes the model architecture complicate. We propose a novel Multi-Format model tuning method based on Reinforcement Learning (MFRL) to learn task-specific knowledge effectively without changing any components of the model.

Inspired by Task-Reward [30], MFRL takes the evaluation metric, which naturally contains the task-specific knowledge, as the reward function and adopts the reinforcement learning to tune the model, which is common practice in the modern language model [4, 12]. Specifically, it is difficult for the model to converge if model is trained with reinforcement learning method from scratch [30]. Hence, MFRL consists of two stages: the pretraining stage and the model tuning stage. In the pretraining stage, model is trained on the lane detection dataset with objective function in Eq. 1 to have a good weight initialization.

In the model tuning stage, we use the REINFORCE algorithm [35] to maximize the objective function as below,

$$Obj = E_{c \sim D}\left[E_{t \sim Q(\cdot|c,u)} R(t, g)\right], \tag{2}$$

where c, t, and g represent the input image, generated format-specific sequences, and ground truths, respectively. E and u denote the mathematic expectation and the model parameter. R, D, and Q stand for the reward function, data distribution of the dataset, and conditional distribution parameterized by u. REINFORCE algorithm estimates the gradient of the reward function by

$$\nabla_u E_{t \sim Q}[R(t, g)] = E_{t \sim Q}[R(t, g) \nabla_u \log Q(t|c, u)], \tag{3}$$

In practice, Eq. 3 is computed as an average of per-sequence gradient and reward function of per-sequence is not required to be differentiable. More details about REINFORCE algorithm can be found in the supplementary materials.

Reward Function. Evaluation metric F1 score and accuracy are unable to adopted as the reward directly, because thay cannot be decomposable as a sum of per-sequence rewards (see Sec 4.2). However, F1 score or accuracy is composed of false positives (FP), true positives (TP), and false negatives (FN). All three indicators can be computed per-sequence. We opt to use TP and FP to design the reward. Specifically, reward for three detection formats are constructed as follows.

Reward for the *segmentation* format (termed as R_{seg}). Segmentation-based lane detection includes two task-specific knowledge, i.e., knowledge of the segmentation and lane detection. For the knowledge of lane detection, we adopt a matched Line-IOU (LIOU) [50], which can demonstrate the quality of location prediction and shape prediction of the lane. For the knowledge of the segmentation, we adopt the matched Mean Intersection over Union (mIOU), which is a widely used metric in the semantic segmentation to evaluate the quality of segmentation, as the reward. Finally, we add both of them to compute r_{seg} as below,

$$\begin{aligned} R_{seg}(t, g) = & \frac{1}{K}\sum_{k=1}^{K}[LIOU(p_k, g) + mIOU(p_k, g)] \\ & - \lambda_1 FP_{seg}(t, g), \end{aligned} \tag{4}$$

where K and p_k represents the number of true positives in t and the k-th true positive. FP_{seg} is the false positive rate of the segmentation format and λ_1 is the weight to control the affect of FP_{seg}. We introduce FP_{seg} to penalize for false positives.

Reward for the *anchor* format (termed as R_a). Anchor-based lane detection contains the knowledge of the keypoint location and lane detection. For the knowledge of keypoint location, we simply adopt the matched Euclidean distance $d(p_k, g)$ between true positives and corresponding ground truths as the reward. Since MFRL needs to maximize the reward, we rescale the reward as $d_r(p_k, g) = 1 - \frac{d(p_k, g)}{H}$, where H is the height of image. For the knowledge of lane detection, we also utilize the matched LIOU as the reward. The formulation of r_a is:

$$R_a(t, g) = \frac{1}{K}\sum_{k=1}^{K}[LIOU(p_k, g) + d_r(p_k, g)] - \lambda_2 FP_a(t, g), \tag{5}$$

where FP_a and λ_2 stand for the false positives rate of the anchor format and the weight of FP_a.

Reward for the *parameter* format (termed as R_p). Parameter-based detection only has the knowledge of lane detection, hence we directly use the matched LIOU as the reward.

$$R_p(t, g) = \frac{1}{K}\sum_{k=1}^{K} LIOU(p_k, g) - \lambda_3 FP_p(t, g), \tag{6}$$

where FP_p and λ_3 stand for the false positives rate of the parameter format and the weight of FP_p.

Since different detection formats have different contributions, we weight the objective function of different formats. The final objective function of MFRL can be computed as,

$$Obj_{total} = \lambda_4 Obj_{seg} + \lambda_5 Obj_a + \lambda_6 Obj_p, \tag{7}$$

where Obj_{seg}, Obj_a, and Obj_p denote the objective function of segmentation, anchor, and parameter format, respectively. The objective function of three formats is the same as Eq. 2, except the format-specific sequence and reward function. λ_4, λ_5, and λ_6 are the scale factor.

4. Experiments

4.1. Datasets

We conduct experiments on three lane detection benchmarks: CULane [28], Tusimple [1], and LLAMAS [3].

CULane is a widely used large-scale dataset for lane detection. It contains a lot of challenging scenarios such as crowded roads. The CULane dataset consists of 88.9K images for training, 9.7K images in the validation set, and 34.7K images for the test. Image size is 1640×590.

Tusimple is a real highway dataset consisting of 3,626 training images and 2,782 testing images. All images have 1280×720 pixels.

LLAMAS is a recently released large-scale lane detection dataset with over 100K images. All lane markers are annotated with high-accurate maps. Image size is 1280×717.

4.2. Evaluation Metrics

For CULane and LLAMAS dataset, we adopt the F1 score to measure the performance: F_1 =

Table 1. Comparison of F1 score and MACs (multiply–accumulate operations) on CULane testing set. We only report the false positives for "Cross" category following [50]. FPS is tested on a single 2080Ti.

Methods	Encoder	FPS↑	Normal↑	Crowded↑	Dazzle↑	Shadow↑	No line↑	Arrow↑	Curve↑	Night↑	Cross↓	Total↑	MACs(G)↓
Segmentation based													
SCNN [28]	ResNet50	12	90.60	69.70	58.50	66.90	43.40	84.10	64.40	66.10	1900	71.60	8.4
RESA [49]	ResNet50	42.1	92.10	73.10	69.20	72.80	47.70	88.30	70.30	69.90	1503	75.3	7.6
AtrousFormer [44]	ResNet34	-	92.83	75.96	69.48	77.86	50.15	88.66	71.14	73.74	**1054**	78.08	14.22
LaneAF [44]	DLA34 [46]	-	91.80	75.61	71.78	79.12	51.38	86.88	72.70	73.03	1360	77.41	12.73
Lane2Seq (segmentation)	ViT-Base	19	**93.39**	**77.27**	**73.45**	**79.69**	**53.91**	**90.53**	**73.37**	**74.96**	1129	**79.64**	15.19
Anchor based													
LaneATT [36]	ResNet122	23	91.74	76.16	69.47	76.31	50.46	86.29	64.05	70.81	1264	77.02	70.5
O2SFormer [51]	ResNet50	44	93.09	76.57	72.25	76.56	52.80	89.50	69.60	73.85	3118	77.83	27.51
UFLD [31]	ResNet34	192	90.70	70.20	59.50	69.30	44.40	85.70	69.50	66.70	2037	72.30	3.3
CondLaneNet [22]	ResNet34	135	93.38	77.14	71.17	**79.93**	51.85	89.89	**73.88**	73.92	1387	78.74	19.6
ADNet [43]	ResNet34	-	92.90	77.45	71.71	79.11	52.89	89.90	70.64	74.78	1499	78.94	18.6
CLRNet [50]	ResNet34	112	**93.49**	**78.06**	**74.57**	79.92	**54.01**	**90.59**	72.77	75.02	1216	**79.73**	17.3
Lane2Seq (anchor)	ViT-Base	23	93.11	77.43	73.25	79.46	53.74	90.02	72.44	**75.12**	**1173**	79.27	15.19
Parameter based													
LSTR [23]	ResNet18	-	-	-	-	-	-	-	-	-	-	64.00	5.9
BezierLaneNet [14]	ResNet18	-	90.22	71.55	62.49	70.91	45.30	84.09	58.98	68.70	996	73.67	4.7
BSNet [6]	ResNet34	-	**93.75**	**78.01**	**76.65**	**79.55**	**54.69**	**90.72**	**73.99**	**75.28**	1445	**79.89**	17.2
Eigenlanes [18]	ResNet50	-	91.70	76.00	69.80	74.10	52.20	87.70	62.90	71.80	1509	77.20	18.7
Laneformer [15]	ResNet50	-	91.77	75.74	70.17	75.75	48.73	87.65	66.33	71.04	**19**	77.06	26.2
Lane2Seq (parameter)	ViT-Base	30	93.03	76.42	72.17	78.32	52.89	89.67	72.67	73.98	1319	78.39	15.19

$\frac{2\times Precision\times Recall}{Precision+Recall}$, where $Precision = \frac{TP}{TP+FP}$ and $Recall = \frac{TP}{TP+FN}$.

For Tusimple dataset, we use F1 score, accuracy, false positives, and false negatives to evaluate the model performance. Accuracy is defined as $Accuracy = \frac{\sum_{clip} C_{clip}}{\sum_{clip} S_{clip}}$, where C_{clip} represents the number of accurately predicted lane points and S_{clip} denotes the total number of lane points of a clip. A lane point is considered as correct if its distance is smaller than the given threshold $t_{pc} = \frac{20}{cos(a_{yl})}$, where a_{yl} denotes the angle of the corresponding ground truth.

4.3. Implementation Details

Model. We adopt the ViT-Base initialized with the MAE [16] pretrained parameters as the image encoder. The patch size is 16×16. The decoder is composed of 2 transformer blocks and its hidden size is 256. The number of attention heads is 8 and the hidden size of feed-forward network is 1024. n_{bins} is set to 1000 and the dimension of word embedding of the vocabulary is 256. Our Lane2Seq is implemented based on MMDetection toolbox [7]. λ_1, λ_2, λ_3, λ_4, λ_5, and λ_6 are set to 0.3, 0.3, 0.1, 0.2, 1, and 1.5, respectively.

Training. All input images are resized to 320×800. We utilize the AdamW [24] as the optimizer with initial learning rate 1e-4. The training epochs of the pretraining stage are 5, 20, and 15 for CULane, Tusimple, and LLAMAS. We set the training epochs of the model tuning stage to 15, 30, and 55 for CULane, Tusimple, and LLAMAS, respectively. For the data augmentation, we use the random horizontal flips and random affine transformation including scaling, rotation, and translation. All experiments are conducted on 8 A100 GPUs with total batch size 384.

Table 2. Comparison of the performance of different models on LLAMAS.

Methods	Encoder	F1(%)↑
PolyLaneNet [37]	EfficientNet-b0 [38]	90.20
BezierLaneNet	ResNet34	96.11
LaneATT	ResNet34	94.96
LaneATT	ResNet122	95.17
LaneAF	DLA34	96.90
CLRNet	ResNet18	96.96
CLRNet	DLA34	97.16
Lane2Seq (segmentation)	ViT-Base	**97.42**
Lane2Seq (anchor)	ViT-Base	97.05
Lane2Seq (parameter)	ViT-Base	96.74

4.4. Comparison with the State-of-the-art Methods

Performance on LLAMAS. Our Lane2Seq achieves a new state-of-the-art performance on LLAMAS dataset using segmentation format. In Table 2, Lane2Seq increases F1 score from 97.16% to 97.42% compare to the previous state-of-the-art method CLRNet. Compared with parameter-based state-of-the-art method BezierLaneNet, Lane2Seq performs better than BezierLaneNet (96.74% vs. 96.11%). The results manifest the detection strengths of Lane2Seq in the multi-lane scenario (The number of lanes $\geq$ 5), such as highways. The reason may be that transformer-based architecture has the advantages in capturing long-range dependency.

Performance on Tusimple. Table 3 presents the performance comparsion with state-of-the-art approaches on Tusimple. Due to small data scale and single scene, the performance gap between different models is small. Our Lane2Seq using segmentation format sets a new state-of-the-art F1 score of 97.95%. The results also validate the detection strengths of Lane2Seq in multi-lane scenario.

Table 3. Comparison of the performance of different models on Tusimple. Acc denotes accuracy.

Methods	Encoder	F1(%)↑	Acc(%)↑	FP(%)↓	FN(%)↓
SCNN	VGG16	95.97	96.53	6.17	**1.80**
RESA	ResNet34	96.93	96.82	3.63	2.48
PolyLaneNet	EfficientNet-bo	90.62	93.36	9.42	9.33
UFLD	ResNet34	88.02	95.86	18.91	3.75
LaneATT	ResNet122	96.06	96.10	5.64	2.17
GANet [39]	ResNet34	97.68	95.87	**1.99**	2.64
CondLaneNet	ResNet34	96.98	95.37	2.20	3.82
CondLaneNet	ResNet101	97.24	96.54	2.01	3.50
CLRNet	ResNet34	97.82	**96.87**	2.27	2.08
CLRNet	ResNet101	97.62	96.83	2.37	2.38
Lane2Seq (segmentation)	ViT-Base	**97.95**	96.85	2.01	2.03
Lane2Seq (anchor)	ViT-Base	97.86	96.72	2.21	2.05
Lane2Seq (parameter)	ViT-Base	96.59	96.00	2.23	3.54

Performance on CULane. We compare Lane2Seq with other state-of-the-art methods on CULane and results are shown in Table 1. For the segmentation-based methods, Lane2Seq improves F1 score from 78.08% to 79.64% compared to the previous state-of-the-art methods AtrousFormer. Compared with anchor-based methods, Lane2Seq outperforms most previous methods. For example, Lane2Seq achieves the better performance than existing row anchor-based methods, such as CondLaneNet (79.27% vs. 78.74%). Moreover, Lane2Seq achieves competitive performance compared to CLRNet (79.27% vs. 79.73%). One possible reason for the performance gap between CLRNet and Lane2Seq is that CLRNet adopts multi-scale detection, which can notably improve the detection performance, while Lane2Seq only uses the plain transformer architecture with single-scale detection. For the comparison of parameter-based methods, Lane2Seq surpasses the existing transformer-based methods. For example, Lane2Seq significantly increases F1 score from 64.00% to 78.39% compared with LSTR and outperforms Laneformer by 1.33% (78.39% vs. 77.06%). But Lane2Seq is ranked second after BSNet. As for the inference speed, we can see that Lane2Seq does not have a significant advantage in inference speed. We attribute that token-by-token prediction method slows the inference speed.

Based on the above results and analysises, we can conclude that the unified lane detection via the sequence generation without any well-designed task-specific components, combining our reinforcement learning-based multi-format model tuning, can achieve promising performance.

Qualitative Results. We display the qualitative results in Fig. 4. The results show that Lane2Seq can effectively detect lanes in the multi-lane scenario (see Fig 4 (a) and (c)). Even in the case of the night scene, Lane2Seq successfully discriminates the lanes (see Fig 4 (b)).

4.5. Ablation Study

We conduct the ablation experiments on CULane dataset to validate the effectiveness of each component. More ablation studies are in the supplementary materials.

Table 4. Ablation study on the multi-format model tuning based on reinforcement learning.

Methods	F1(%)↑	Precision(%)↑	Recall(%)↑
Lane2Seq (segmentation)	64.27	70.44	52.77
Lane2Seq (segmentation) + MFRL	**79.64 (+15.37)**	**85.26 (+14.82)**	**69.00 (+16.23)**
Lane2Seq (anchor)	66.23	71.36	55.75
Lane2Seq (anchor) + MFRL	**79.27 (+13.04)**	**84.72 (+13.36)**	**67.28 (+11.53)**
Lane2Seq (parameter)	68.14	73.27	57.89
Lane2Seq (parameter) + MFRL	**78.39 (+10.25)**	**83.68 (+10.41)**	**66.94 (+9.05)**

Table 5. Ablation study on the different reward functions. F1 score is adopted as the indicator.

R_{seg}	R_a	R_p	Segmentation	Anchor	Parameter
			64.27	66.23	68.14
✓			77.69 (+13.42)	66.23 (+0.0)	68.14 (+0.0)
	✓		64.27 (+0.0)	76.94 (+10.71)	68.14 (+0.0)
		✓	64.27 (+0.0)	66.23 (+0.0)	77.02 (+8.88)
✓	✓		78.72 (+14.45)	78.14 (+11.91)	68.14 (+0.0)
✓		✓	78.70 (+14.43)	66.23 (+0.0)	77.90 (+9.76)
	✓	✓	64.27 (+0.0)	78.20 (+11.97)	77.93 (+9.79)
✓	✓	✓	**79.64 (+15.37)**	**79.27 (+13.04)**	**78.39 (+10.25)**

Multi-Format Model Tuning Based on Reinforcement Learning. We first ablate the effectiveness of the proposed Multi-Format Model Tuning Based on Reinforcement Learning (MFRL). As shown in Table 4, MFRL significantly improves the performance of different detection formats. For the segmentation format, MFRL gains 15.37% F1 score improvement (79.64% vs. 64.27%). For the anchor and parameter format, MFRL increases 13.04% (79.27% vs. 66.23%) and 10.25% (78.39% vs. 68.14%) F1 score, respectively. Reasons of performance improvement can be attributed in two folds: (1) the vanilla Lane2Seq is a task-agnostic architecture and lacks the task-specific knowledge, leading to the ineffective feature learning of lane. MFRL tackles this problem by adopting the evaluation metric, which is designed based on the lane prior information like angle, as the reward to tune the model. (2) MFRL makes the model's predictions aligned with their intended usage, i.e., how to achieve a high performance. Optimizing the loss function is a common practice in the compute vision. However, this way indirectly optimizes for model's intended usage, because the lower loss does not mean the higher performance. Instead, MFRL views the maximizing the evaluation metric as the optimization objective, which is positively correlated with the model performance.

Effectiveness of Reward Function of Different Formats. We ablate the effectiveness of reward function of different formats and results are shwon in Table 5. It can be observed that our reward function can bring consistent performance improvement. However, we can also see that the performance will not be improved if the corresponding reward is not optimized. Besides, we can find that multi-reward optimization can achieves better results than single-reward optimization. Reason may be that multi-reward optimiza-

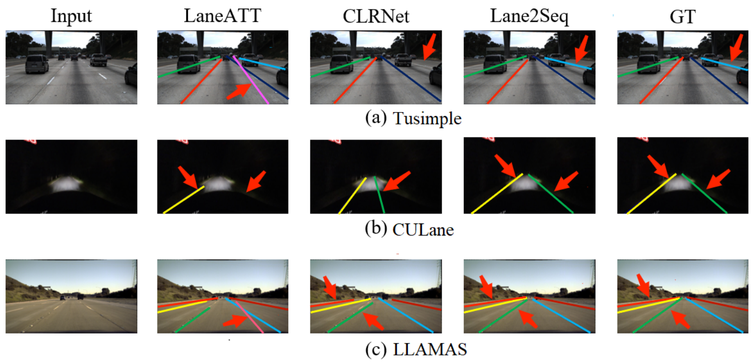

Figure 4. Visualization results of LaneATT, CLRNet, and Lane2Seq on Tusimple, CULane, and LLAMAS.

Table 6. Comparison between Lane2Seq and YOLOP.

Methods	F1(%)↑	Precision(%)↑	Recall(%)↑
YOLOP (segmentation)	75.23	79.46	60.58
Lane2Seq (segmentation)	**79.64 (+4.41)**	**85.26 (+5.8)**	**69.00 (+8.42)**
YOLOP (anchor)	76.44	80.99	62.15
Lane2Seq (anchor)	**79.27 (+2.83)**	**84.72 (+3.73)**	**67.28 (+5.13)**
YOLOP (parameter)	69.43	74.38	58.46
Lane2Seq (parameter)	**78.39 (+8.96)**	**83.68 (+9.30)**	**66.94 (+8.48)**

tion enables model to learn correlations between different detection formats and provides more supervised signals.

Comparison with other Multi-Task Methods. An alternative method to unify different detection formats is using different detection heads in a model. YOLOP [42] is a representative work for this method. We replace the heads of YOLOP with anchor-based head, segmentation-based head, and parameter-based head. The segmentation-based loss, anchor-based loss, parameter-based loss for YOLOP are same as that in SCNN [28], LaneATT [36], and PolyLaneNet [37]. From Table 6, we can observe that Lane2Seq surpasses YOLOP for all detection formats. It should be noted that YOLOP contains task-specific modules, such as detection head and loss function, but Lane2Seq does not. Results manifest the sequence generation combining with MFRL is a more simple and effective way to unify lane detection.

Ablation Study on the Scale Factor of Different Objective Functions. For the limitation of the space, we take the segmentation format as an example to present the influence of the scale factor of different objective functions in MFRL. Results of other two formats can be found in the supplementary materials. As shown in Table 7, assigning the same factor to different objective functions causes insuperior performance, indicating different detection formats have different contributions. We set λ_4, λ_5, and λ_6 to 0.2, 1, and 1.5 according to the model performance.

5. Conclusion

This paper presents a novel sequence generation-based lane detection framework, i.e., Lane2Seq, to unify lane detection, which casts lane detection as a sequence generation task. Lane2Seq gets rid of complicated task-specific modules and adopts a simple transformer-based encoder-decoder architecture. To incorporate the task-specific knowledge, we employ a multi-format model tuning based on reinforcement learning (MFRL). Extensive experiments show Lane2Seq is effective and achieves competitive results compared to the state-of-the-art methods.

Limitation. One major limitation of Lane2Seq is that sequence generation model is expensive for long sequence (mainly for inference). The inference speed is low when there are more than 5 lanes in an image despite its detection strength in multi-lane scenario. Therefore, future work is required to make it faster for real-time lane detection applications. Another limitation is that MFRL can only be applied to sequence generation-based model currently. Our ongoing work is applying MFRL to other vision models.

Table 7. Ablation study on different scale factors.

λ_4	λ_5	λ_6	F1(%)↑
1	1	1	73.23
0.7	1	1	74.96
0.4	1	1	76.27
0.2	1	1	77.35
0.1	1	1	77.02
0.2	1.1	1	77.03
0.2	1.2	1	76.75
0.2	0.9	1	77.09
0.2	0.8	1	77.00
0.2	1	0.9	76.87
0.2	1	1.2	77.37
0.2	1	1.4	79.46
0.2	1	1.5	**79.64**
0.2	1	1.6	79.39

References

[1] Tusimple dataset. https://github.com/TuSimple/tusimple-benchmark. Accessed on 11th August 2023. 5

[2] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736, 2022. 2

[3] Karsten Behrendt and Ryan Soussan. Unsupervised labeled lane markers using maps. In *Proceedings of the IEEE/CVF international conference on computer vision workshops*, pages 0–0, 2019. 5

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 4

[5] Lluis Castrejon, Kaustav Kundu, Raquel Urtasun, and Sanja Fidler. Annotating object instances with a polygon-rnn. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5230–5238, 2017. 3

[6] Haoxin Chen, Mengmeng Wang, and Yong Liu. Bsnet: Lane detection via draw b-spline curves nearby. *arXiv preprint arXiv:2301.06910*, 2023. 6

[7] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 6

[8] Ting Chen, Saurabh Saxena, Lala Li, David J Fleet, and Geoffrey Hinton. Pix2seq: A language modeling framework for object detection. *arXiv preprint arXiv:2109.10852*, 2021. 2, 4

[9] Ting Chen, Saurabh Saxena, Lala Li, Tsung-Yi Lin, David J Fleet, and Geoffrey E Hinton. A unified sequence interface for vision tasks. *Advances in Neural Information Processing Systems*, 35:31333–31346, 2022. 2

[10] Xin Chen, Houwen Peng, Dong Wang, Huchuan Lu, and Han Hu. Seqtrack: Sequence to sequence learning for visual object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14572–14581, 2023. 2, 4

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 4

[12] Yuqing Du, Olivia Watkins, Zihan Wang, Cédric Colas, Trevor Darrell, Pieter Abbeel, Abhishek Gupta, and Jacob Andreas. Guiding pretraining in reinforcement learning with large language models. *arXiv preprint arXiv:2302.06692*, 2023. 4

[13] Ruochen Fan, Xuanrun Wang, Qibin Hou, Hanchao Liu, and Tai-Jiang Mu. Spinnet: Spinning convolutional network for lane boundary detection. *Computational Visual Media*, 5:417–428, 2019. 1, 2

[14] Zhengyang Feng, Shaohua Guo, Xin Tan, Ke Xu, Min Wang, and Lizhuang Ma. Rethinking efficient lane detection via curve modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17062–17070, 2022. 6

[15] Jianhua Han, Xiajun Deng, Xinyue Cai, Zhen Yang, Hang Xu, Chunjing Xu, and Xiaodan Liang. Laneformer: Object-aware row-column transformers for lane detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 799–807, 2022. 6

[16] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022. 6

[17] Hiroto Honda and Yusuke Uchida. Clrernet: Improving confidence of lane detection with laneiou. *arXiv preprint arXiv:2305.08366*, 2023. 1, 4

[18] Dongkwon Jin, Wonhui Park, Seong-Gyun Jeong, Heeyeon Kwon, and Chang-Su Kim. Eigenlanes: Data-driven lane descriptors for structurally diverse lanes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17163–17171, 2022. 6

[19] Taeho Kil, Seonghyeon Kim, Sukmin Seo, Yoonsik Kim, and Daehee Kim. Towards unified scene text spotting based on sequence generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15223–15232, 2023. 2

[20] Xiang Li, Jun Li, Xiaolin Hu, and Jian Yang. Line-cnn: End-to-end traffic line detection with line proposal unit. *IEEE Transactions on Intelligent Transportation Systems*, 21(1):248–258, 2019. 2

[21] Xuan Liao, Wenhao Li, Qisen Xu, Xiangfeng Wang, Bo Jin, Xiaoyun Zhang, Yanfeng Wang, and Ya Zhang. Iteratively-refined interactive 3d medical image segmentation with multi-agent reinforcement learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9394–9402, 2020. 3

[22] Lizhe Liu, Xiaohao Chen, Siyu Zhu, and Ping Tan. Condlanenet: a top-to-down lane detection framework based on conditional convolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3773–3782, 2021. 6

[23] Ruijin Liu, Zejian Yuan, Tie Liu, and Zhiliang Xiong. End-to-end lane shape prediction with transformers. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 3694–3702, 2021. 1, 2, 6

[24] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 6

[25] Jiasen Lu, Christopher Clark, Rowan Zellers, Roozbeh Mottaghi, and Aniruddha Kembhavi. Unified-io: A unified

model for vision, language, and multi-modal tasks. *arXiv preprint arXiv:2206.08916*, 2022. 2

[26] Wenhan Luo, Peng Sun, Fangwei Zhong, Wei Liu, Tong Zhang, and Yizhou Wang. End-to-end active object tracking via reinforcement learning. In *International conference on machine learning*, pages 3286–3295. PMLR, 2018. 3

[27] Stefan Mathe, Aleksis Pirinen, and Cristian Sminchisescu. Reinforcement learning for visual object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2894–2902, 2016. 3

[28] Xingang Pan, Jianping Shi, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Spatial as deep: Spatial cnn for traffic scene understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. 1, 2, 5, 6, 8

[29] Dezhi Peng, Xinyu Wang, Yuliang Liu, Jiaxin Zhang, Mingxin Huang, Songxuan Lai, Jing Li, Shenggao Zhu, Dahua Lin, Chunhua Shen, et al. Spts: single-point text spotting. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 4272–4281, 2022. 2

[30] André Susano Pinto, Alexander Kolesnikov, Yuge Shi, Lucas Beyer, and Xiaohua Zhai. Tuning computer vision models with task rewards. *arXiv preprint arXiv:2302.08242*, 2023. 2, 3, 4

[31] Zequn Qin, Huanyu Wang, and Xi Li. Ultra fast structure-aware deep lane detection. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIV 16*, pages 276–291. Springer, 2020. 1, 2, 6

[32] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018. 4

[33] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020. 2, 4

[34] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR, 2021. 2

[35] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999. 5

[36] Lucas Tabelini, Rodrigo Berriel, Thiago M Paixao, Claudine Badue, Alberto F De Souza, and Thiago Oliveira-Santos. Keep your eyes on the lane: Real-time attention-guided lane detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 294–302, 2021. 1, 2, 6, 8

[37] Lucas Tabelini, Rodrigo Berriel, Thiago M Paixao, Claudine Badue, Alberto F De Souza, and Thiago Oliveira-Santos. Polylanenet: Lane estimation via deep polynomial regression. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 6150–6156. IEEE, 2021. 2, 6, 8

[38] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 6

[39] Jinsheng Wang, Yinchao Ma, Shaofei Huang, Tianrui Hui, Fei Wang, Chen Qian, and Tianzhu Zhang. A keypoint-based global association network for lane detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1392–1401, 2022. 7

[40] Ze Wang, Weiqiang Ren, and Qiang Qiu. Lanenet: Real-time lane detection networks for autonomous driving. *arXiv preprint arXiv:1807.01726*, 2018. 1

[41] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992. 2

[42] Dong Wu, Man-Wen Liao, Wei-Tian Zhang, Xing-Gang Wang, Xiang Bai, Wen-Qing Cheng, and Wen-Yu Liu. Yolop: You only look once for panoptic driving perception. *Machine Intelligence Research*, 19(6):550–562, 2022. 8

[43] Lingyu Xiao, Xiang Li, Sen Yang, and Wankou Yang. Adnet: Lane shape prediction via anchor decomposition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6404–6413, 2023. 6

[44] Jiaxing Yang, Lihe Zhang, and Huchuan Lu. Lane detection with versatile atrousformer and local semantic guidance. *Pattern Recognition*, 133:109053, 2023. 6

[45] Zhengyuan Yang, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Faisal Ahmed, Zicheng Liu, Yumao Lu, and Lijuan Wang. Unitab: Unifying text and box outputs for grounded vision-language modeling. In *European Conference on Computer Vision*, pages 521–539. Springer, 2022. 2

[46] Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. Deep layer aggregation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2403–2412, 2018. 6

[47] Jia-Qi Zhang, Hao-Bin Duan, Jun-Long Chen, Ariel Shamir, and Miao Wang. Houghlanenet: Lane detection with deep hough transform and dynamic convolution. *arXiv preprint arXiv:2307.03494*, 2023. 2

[48] Zhiyuan Zhao, Qi Wang, and Xuelong Li. Deep reinforcement learning based lane detection and localization. *Neurocomputing*, 413:328–338, 2020. 3

[49] Tu Zheng, Hao Fang, Yi Zhang, Wenjian Tang, Zheng Yang, Haifeng Liu, and Deng Cai. Resa: Recurrent feature-shift aggregator for lane detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 3547–3554, 2021. 1, 2, 4, 6

[50] Tu Zheng, Yifei Huang, Yang Liu, Wenjian Tang, Zheng Yang, Deng Cai, and Xiaofei He. Clrnet: Cross layer refinement network for lane detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 898–907, 2022. 5, 6

[51] Kunyang Zhou and Rui Zhou. End to end lane detection with one-to-several transformer. *arXiv preprint arXiv:2305.00675*, 2023. 2, 6