Notes on "End-to-end Lane Shape Prediction with Transformers"

1 Related Work

1.1 Segmentation based

- SCNN
- LaneNet

require heavy postprocessing due to heavy misalignment of representations, and suffer from locality of segmentation task (performs poorly under occulusion or extreme lighting conditions)

1.2 Point detection based

- Line-CNN
- LaneATT

have to design heuristic lane anchors, which highly depend on dataset statistics, and require NMS as postprocessing

1.3 Curve based

- PolyLaneNet
- LSTR

performance lags behind SOTA, slow convergence and high latency

2 BézierLaneNet

$$\mathcal{B} = \sum_{i=0}^{n} b_{i,n}(t)\mathcal{P}_i, t \leq t \leq 1 \tag{1}$$

where $\mathcal{P}_i$ is the i-th control point, and $b_{i,n}(t)$ is the Bernstein basis polynomial of degree n:

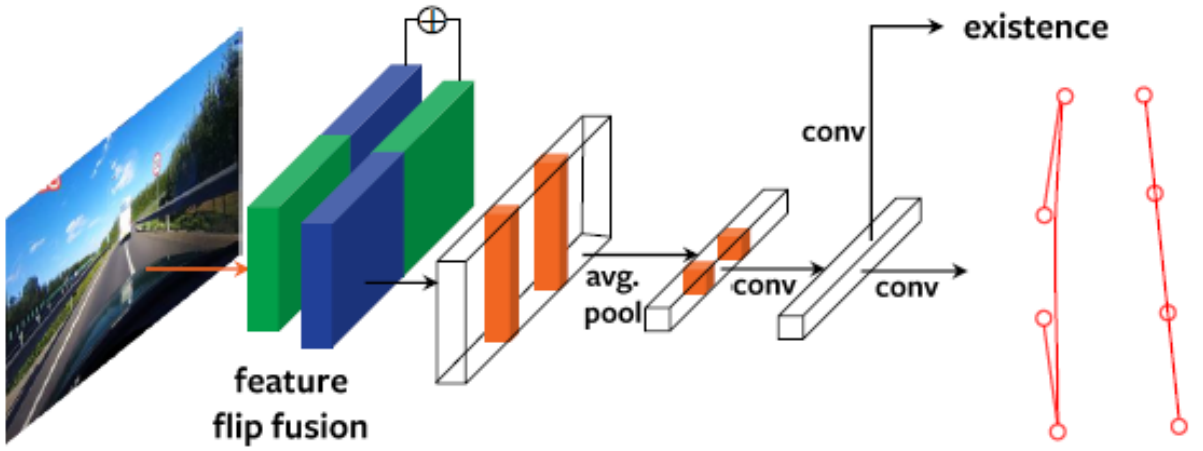
$$b_{i,n} = C_n^i t^i (1-t)^{n-i}, i = 0, ..., n \tag{2}$$

here $n = 3$ is used as it provides better ground truth fitting ability than 3rd order polynomial.

2.1 Architecture

2.1.1 Backbone

layer 3 feature of ResNet as backbone, but replace the dilation inside the backbone network by two dilated blocks outside with dilation rates.



2.1.2 Feature flip fusion module

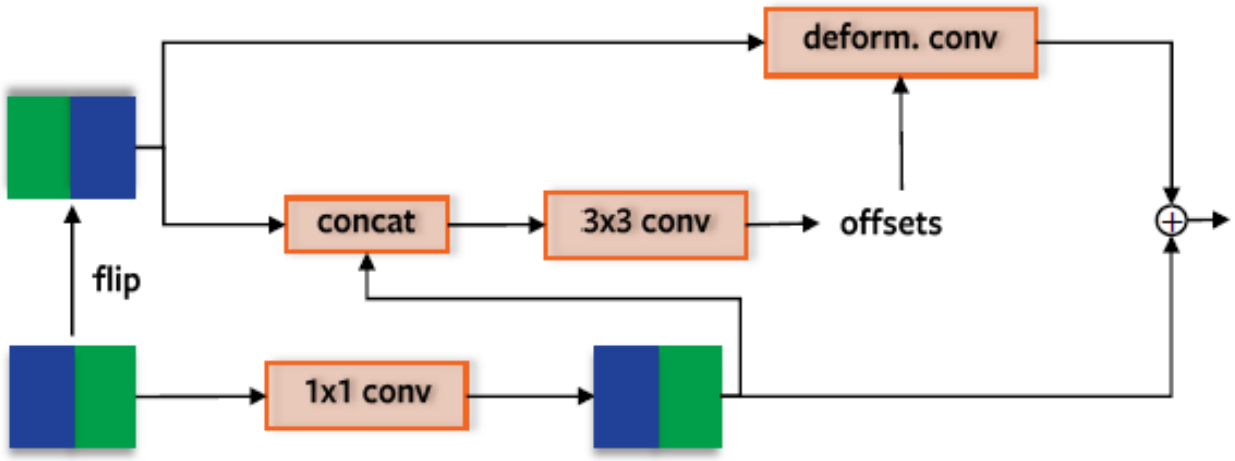
aggregate opposite lane features

2.1.3 Average pooling

enriched feature map ($C \times \frac{H}{16} \times \frac{W}{16}$) pooled to ($C \times \frac{W}{16}$), resulting in $\frac{W}{16}$ proposals, and 50 for CULane (800-pixel wide). Two 1×3 1D convolutions are used to transform the pooled features, replacing NMS (not required any more).

2.2 Final prediction

Outputs are $\frac{W}{16} \times 8$ for regression of 4 control points, and $\frac{W}{16} \times 1$ for existence of lane line object



2.2.1 Feature flip fusion

Fuse the feature map with horizontally flipped version: two separate convolution and normalization layers transform each feature map and then are added before ReLU.
Deformable convolution is used for learning offsets conditioned on the original feature map for feature alignment.
Auxiliary binary segmentation branch to enforce the learning of spatial details.

2.3 End-to-end fitting

2.3.1 Distance between Bézier curves

sample curves at a uniformly spaced set of t values

$$\mathcal{L}_{reg} = \frac{1}{n} \sum_{t \in T} ||\mathcal{B}(f(t)) - \mathcal{B}(\hat{f}(t))|| \tag{3}$$

usually $f(t) = t$ works well

2.3.2 Bézier ground truth generation

$$\begin{bmatrix} \mathcal{P}_0 \\ \mathcal{P}_1 \\ \vdots \\ \mathcal{P}_n \end{bmatrix} = \begin{bmatrix} k_{x_0} & k_{y_0} \\ k_{x_1} & k_{y_1} \\ \vdots & \vdots \\ k_{x_m} & k_{y_m} \end{bmatrix} \begin{bmatrix} b_{0,n}(t_0) & ... & b_{n,n}(t_0) \\ b_{0,n}(t_1) & ... & b_{n,n}(t_1) \\ \vdots & \ddots & \vdots \\ b_{0,n}(t_m) & ... & b_{n,n}(t_m) \end{bmatrix}^T \tag{4}$$

In which $\{t\}_{i=0}^{m}$ is uniformly sampled from 0 to 1.

2.3.3 Label and prediction matching

perform one-to-one assignment between G labels and N predictions (G ¡ N) using optimal bipartite matching. Find G-permutation of N predictions $\pi \in \Pi_G^N$:

$$\hat{\pi} = \underset{\pi \in \Pi_G^N}{\arg\max} \sum_{i}^{G} Q_{i,\pi(i)} \tag{5}$$

in which

$$Q_{i,\pi(i)} = (\hat{p}_{\pi(i)})^{1-\alpha} \cdot (1 - L_1(b_i, \hat{b}_{\pi(i)}))^{\alpha} \tag{6}$$

and represents matching quality of the i-th label with $\pi(i)$-th prediction, based on L_1 distance between curves $b_i, \hat{b}_{\pi(i)}$ and class score $\hat{p}_{\pi(i)} \cdot \alpha$ is set to 0.8 by default (solved by Hungarian algorithm)

2.3.4 Overall loss

classification loss:

$$\mathcal{L}_{cls} = -(y \log(p)) + w(1-y)\log(1-p)) \tag{7}$$

in which w is the wieght for negative samples, and is set to 0.4 in all experiments. Overall loss:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{reg} + \lambda_2 \mathcal{L}_{cls} + \lambda_3 \mathcal{L}_{seg} \tag{8}$$

in which $\lambda_1, \lambda_2, \lambda_3$ are set to 1, 0.1, 0.75 respectively.

2.4 Evaluation

$$F1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{9}$$