Lane detection for autonomous vehicles using Attention mechanism

1. Abstract

Motivation:
Lane detection is the process of identifying the position and orientation of the markings on the roads for accurate navigation. Lane detection being the basic feature required for trajectory prediction and motion planning, it is the most crucial tool in Autonomous vehicles. Precise detection of lanes will improve the efficient maneouver of cars, which in turn leads to a safer environment for humans.

Challenges in existing work:
• Recent models use deep network architecture(Torres et al., 2020)(Li et al., 2020) (Segmentation networks) for increasing the precision of lane detection algorithms. But the increase in complexity in turn decreases the computational speed (increasing the run time),thus making it void for real time applications.
• Contextual information is required for cases where the dynamic changes in environment hinders the lane detection. For example shown in Figure 1.a), the landmark is wrongly assumed to be lane due to loss of previous information about the landmark. In Figure 1.b), the lane is inaccurately determined due to the assumption of context that the road end is parallel to the lane markings.
• As shown in Figure 1.(c) and 1.(d), External factors like sunlight and occlusion of lane markings due to other vehicles severely affect the accuracy of detecting the lanes.

Summary of Implementation:
As mentioned above, lane detection has numerous challenges and hence it requires complex networks to solve the problems. In particular, the aim is to solve the real time issue of high computational complexity as well as increase the accuracy and robustness of lane detection. On further analysis of the three papers, the BeizerNet paper(git, b) appears to be time consuming (since the Beizer control points decoding is highly time consuming) and hence does not work in dynamic environment datasets. Hence, the aim is to improve the Lane attention mechanism (Tabelini et al., 2020) (Paper 1) to reach a higher level of accuracy, by implementing these approaches:

- Introduce initialization weights in the attention mechanism variables of laneatt to include the local contextual information.
- Modify the loss function to aggregate loss of all anchor points, in order to get accurate lanes, and reduce the number of erroneous lanes as network output.
- Add squeeze and excitation block in the attention mechanism to improve the attention module.

Summary of Results:
There is a slight increase of accuracy (from 95.63 to 95.67) by implementing Image normalization to induce brightness invariance in the images(details mentioned in Decsription). Accuracy is further improved (from 95.67 to 95.74) by initializing weights using xavier normal implementation and introducing squeeze and excitation network structure. The increase of 0.1 percent is significant since the problem statement is a real world application and 0.1 percent has a direct impact on saving the loss of hundreds of lives.

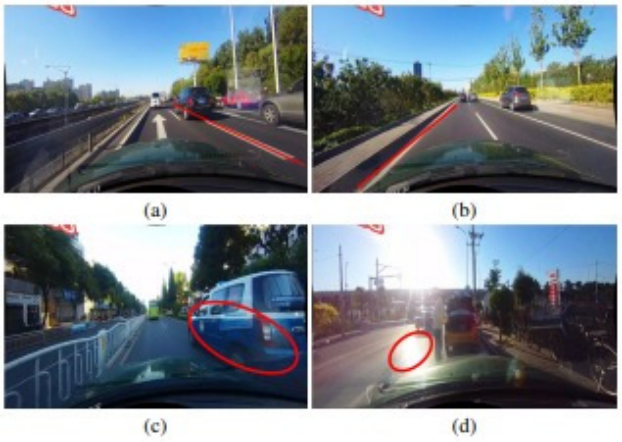

Figure 1. Figure a shows that the left lane marking is incorrectly detected due to wrong assumption of landmarks. Figure b shows that the lane detection is not accurate due to the association of high level features(assumes the lane marking to be parallel to the road end) Figure c shows that lack of detection due to the occlusion of another car Figure d shows effects of blur in image due to sunlight

2. Substantive review and critique

Paper1 (LaneATT):
Key points: Addresses three challenges faced in lane detection:

- Solves the real time issue of high computational com-

plexity by introducing features of lower dimensionality (feature pooling)

- Achieves higher accuracy by taking global features (allowing contextual information) in order to have a better prediction during loss of dynamic information

- Novel method of representing lanes through anchors (reference lines for lanes) which reduce the complexity of feature information

Framework: As shown by Figure 2, the following steps are used to detect and predict the predict the lanes of highest probability:

Step1:
Images are fed into a standard CNN network to get the feature map. This feature map is reduced dimensionally to a single channel (that is later compared with anchor line)

Step2:
The single channels are compared with a set of anchors predicted for each channel feature. Based on the number of intersection points, the anchor corresponds itself to an incoming feature channel by creating a pool of intersecting points. This step called as "feature pooling" extensively reduces computational costs.

Step3:
Further, this is fed to an attention network which concatenates the existing channel with global information(contextual information).

Step4:
The channel is fed into two Fully connected layers for classification and regression of the detected lanes respectively to predict the lanes of highest probability.

If the number of lanes are more than a certain threshold, NMS (Non maximal suppression) is applied.

Lane and anchor representation:
The paper suggests that, keeping the Y axis within a fixed interval (discrete finite steps), only X changes for defining a lane. Hence, only X can be considered as the variable among X & Y, along with the orientation specification of the lane. To define this with respect to a coordinate system, the term "anchor" is used, which is defined as a reference line in the image plane as a vector of origin with a direction.

STRENGTHS:
• Incorporates global context, which solves majority of the issues in images where the information is lost due to dynamic changes in the environment.

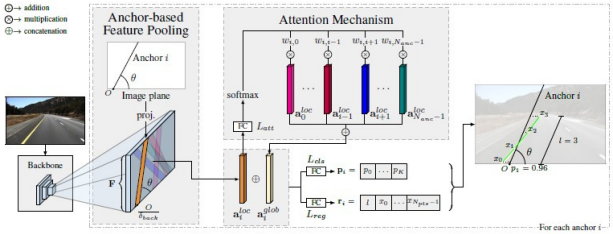

Figure 2. Backbone represents CNN network like ResNet, anchors are projected onto the feature channels, aggregate global information gets added through the attention networks, and finally the concatenated channel is fed into regression and classification networks.

• Reduction of a bounding box to an anchor line tremendously reduces the computational cost, and hence becomes the selling point of the paper.
WEAKNESS:
• Local context is not yet exploited well in this paper, CLRNet addresses this issue
• Higher postprocessing time, since there are many lanes detected and filtering is required.

Paper2 (CLRNet):

Key points:

- Considers the entire line as a segment (termed as "Line IoU") for calculating the loss function, instead of singular points, which improves the efficiency of contextual information being used.

- Utilizes global and local context as input information for lane detection.

- Novel method called "ROIGather" to use the global and local contextual information.

- Novel metrics implemented to compare with other benchmarked state-of-the-art techniques in lane detection.

Framework:
As stated above, CLRNet aims to fully utilize both high and level features using the feature pyramid network (FPN) shown in Figure 3(a). More specifically, it firsts performs detection in high semantic features to coarsely localize lanes and then performs refinement based on fine detail features to get more precise locations. Progressively refining the location of lane and feature extraction leads to

high accuracy detection results. Using ROIALign (He et al., 2017), lane prior is assigned to each feature but needs to extract more global contextual information to overcome the challenges of occlusion and extreme lighting conditions. ROIGather shown in Figure 3(b) adds convolutions along the lane prior thereby extracting features of nearby pixel Moreover, this paper also defines the IoU between lane lines and proposes the Line IoU (LIoU) loss as shown in Figure 3(c) to regress the lane as a whole unit and considerably improve the performance compared with standard loss, i.e., smooth-l1 loss.

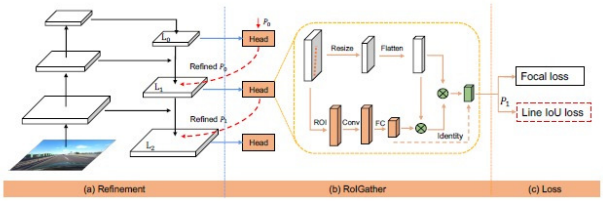

Figure 3. Refinement(a) represents networks to reduce dimensionality and create feature maps from the FPN structure. At each level, the features from low level to high level are extracted. RoIGather(b) represents the novel mechanism of aggregating the local and global information. Loss(c) calculates the loss function and gives the classification(focal) and regression(line IoU) loss.

STRENGTHS:
• One main difference between LaneATT and CLRNet is that LaneATT uses a predefined set of anchors whereas in CLRNet detect anchors with high-level semantics then perform refinement based on low level features.
• In the field of object detection ,the idea of using feature pyramid network (FPN) and assigning objects to different scales and detecting them sequentially improves the performance of the model (especially cascade RCNN)

WEAKNESS / DISADVANTAGES:
• Computationally expensive due to the heavier / deeper networks like FPN and ROIGather

Paper3 (BeizerNet):
Key points:

- This paper uses the basic idea of fitting lanes to a unique parametric curve called the Beizer Curve, unlike the other two papers which uses a large number of anchors to decode lane detection.
- Implements deformable convolution-based feature flip fusion to exploit the symmetric properties of lanes in driving scenes.
- Achieves low latency due to its lightweight backbones.
- Reduces computational time by avoiding post processing methods due to its end-to-end lane detection property.

Need for curve-based models:
The main advantage of using curve-based methods is that there is no need for post-processing (like Non-Maximum Suppression) steps and can detect varying numbers of lanes, unlike the previous two methods.

Beizer curve: These curves are controlled by n+1 control points where n is the order of curves. This paper uses cubic Bezier curves i.e controlled by 4 points. The paper presents Cubic order as a good trade-off between high degrees of freedom and instability. The network predicts the control points and later uses these points to predict lanes.

Framework:
CLRNet is an extension of LaneATT and uses the same lane representation. As shown in Figure 4.(a), BeizerNet uses ResNet as the backbone and replaces the last two layers with dilated blocks for better accuracy-latency tradeoff. The extracted feature is enriched using the feature flip fusion module shown in Figure 4.(b). This module fuses the features by calculating deformable convolution offsets between the original and the horizontally flipped version. These are then pooled using 1D and two 1D convolution layers as an alternative to non-maximal suppression (NMS) operation. Beizer curve sampling loss is used to measure the distance between two bezier curves and binary cross entropy loss (BCE) is used as classification loss.

STRENGTHS:
• Previous curve-based models used polynomial curves (Torres et al., 2020)(for example $x = ay^3 + by^2 + cy + d$) to fit the lanes. Although this is a simple way to fit lanes, these methods take a long time to converge and need high-latency backbone architectures for precise detection. The performances of these models were not comparable to other segmentation-based and point detection models. Beizer curve also helps in optimising these difficulties by reducing the computation and enhancing the stability of the lanes.
• Feature Flip fusion exploits the symmetric propertics of the lanes, which is a unique and useful information added to the network. This feature can be implemented in other networks to enhance the accuracy of lane detection.

WEAKNESS:
• The advantage of absence in post-processing steps is nullified by the time consumed in calculating the lane from control points

• Better for curves than straight lines but lanes most are straight than curved(data imbalance problem - bias between two types of data).

Overall Observation and Inference:

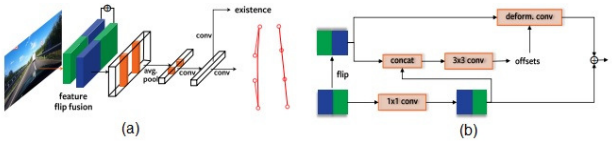

Figure 4. a) shows Resnet backbone with the last two layers of dilated blocks. b) represents the flip fusion module. This module fuses the features by calculating deformable convolution offsets between the original and the horizontally flipped version.

In particular, the aim is to solve the real time issue of high computational complexity as well as increase the accuracy and robustness of lane detection. On further analysis of the three papers, the BeizerNet paper(git, b) ap- pears to be time consuming (since the Beizer control points decoding is highly time consuming) and hence does not work in dynamic environment datasets.

3. Description of LaneATT Implementation & Improvements:

Implemented the code of LaneATT paper (git, a) with the following specifications:

Ran the training with Resnet32 model with TuSimple test and train datasets. Setup: On RTX 3090 conda environment with

- Python>= 3.5
- PyTorch == 1.6, tested on CUDA 10.2. The models were trained and evaluated on PyTorch 1.6
- CUDA, to compile the NMS code

Detailed description of LaneATT algorithm (along with new implementations) :

1. **Inputs:**

The input image (car's front camera images taken during maneouver)

2. **Expected Output :**

A lane proposal is predicted for each anchor and consists of three main components:

(i) K + 1 probabilities (K lane types (eg: solid or dashed) and one class for invalid proposal),

(ii) Npts offsets (the horizontal distance between the prediction and the anchor's line), and

(iii) the length l of the proposal (the number of valid offsets).

An anchor is a "virtual" line in the image plane defined by (i) an origin point $O = (xorig, yorig)(y\_origY)$ located in one of the borders of the image (except the top border) and (ii) a direction .

(i) is a classification problem output , (ii) and (iii) are regression outputs

3. **Network Structure:**

- Input image is convoluted with feature detectors to create feature maps.
- It is then passed through maxpooling (squeeze module) anf flattened to obtain pooled feature maps as one dimensional feature vector.
- These are the local features (obtained from the pixel values directly)
- In order to obtain the global values (information from inter dependency of these features with each other), the local features are passed through an attention mechanism to obtain global information (global features).
- These two types of features are then concatenated with each other and passed through two Fully connected layers (followed by Softmax layer) (excitation module and scale module), one acting as a classifier (to obtain the lane type) and other acting as a regression network (to obtain θ and origin point of the anchor).

Modifications:

- Implemented Image normalization to improve brightness using the formula:
  $Output\_channel = 255 * (Input\_channel - min)/(max - min)$
- Added zero-padding to check if there is any increase in accuracy of the existing model:
  $self.conv2d = nn.LazyConv2d(1, kernel\_size = 3, padding = 1)$
  along with existing stride and offsets
- Added Xavier_normal initialization of weights to have a better estimate of the first iteration parameters of attention weights. Optimal initialization of Attention weights significantly improve performance of the Global information parsed from the feature vectors.

- Added a part of the squeeze and excitation module. It basically consists of 3 components:
  - Squeeze Module
  - Excitation Module
  - Scale Module
- Since LaneATT already has Maxpool in its layers it is equivalent to having a Squeeze module
- Excitation module consists of a Multi Layer perceptron, which is nothing but Fully connected feed forward neural network (intermittently used after getting the Resnet backbone using Sequential neural network structure followed by Convolutional layers).
- Scale module is passing the excitation module through an activation layer (non linear function like sigmoid). In this paper, they have implemented Softmax after the excitation layer in order to tune the attention weights.
- The modules even though are not entirely implemented a a single network, it is inherently built in the code and hence an addition of the same as another block does not give a good trade off between accuracy and computational efficiency.

4. Evaluation Discussion & Results:

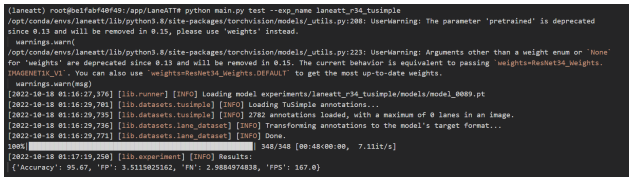

Figure 5. FP = Percentage of false positives, FN = Percentage of False negatives, FPS = Frames per second (speed of computation)

- There is a slight increase of accuracy (from 95.63 to 95.67) from the first two modifications mentioned above. The trade off between computational cost and accuracy is optimal, since the increase in computational complexity is negligible for the first two modifications.
- There is a considerable increase of accuracy (from 95.63 to 95.75) from the first two modifications on the initialization of attention mechanism mentioned as Point 3 in modifications. The trade off between computational cost and accuracy is optimal, since the increase in computational complexity is negligible for the first two modifications.
- Implemented Squeeze and excitation module, but the trade off between the computational complexity and accuracy was not optimal, since the accuracy increase was 0.02 percentage whereas the time taken for training increased by more than 1.5 hours. Hence, the modification is not carried forward to the final implementation.

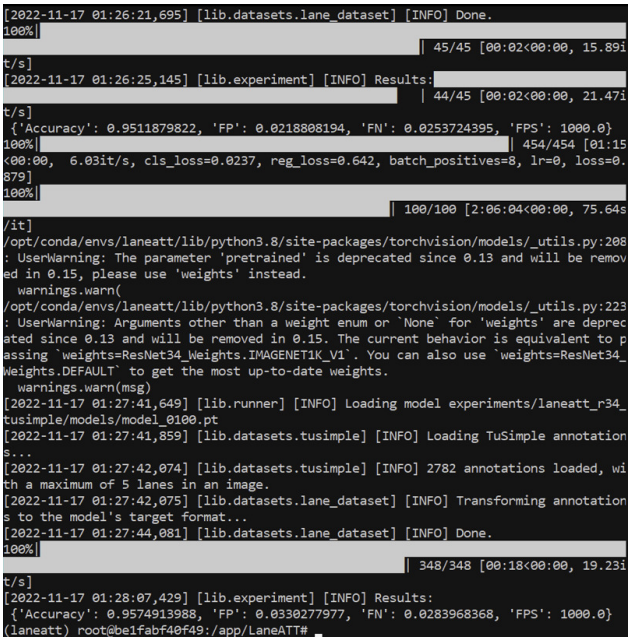

Figure 6. FP = Percentage of false positives, FN = Percentage of False negatives, FPS = Frames per second (speed of computation)

Future work:

- After using basic pre-processing techniques and some tweaks to improve the accuracy, apply different layers of gaussian filters and obtain Difference of Gaussian (DoG) to improve the global and local contextual information (Feature pyramid network). The output of this pyramid should be concatenated to the attention weights. Therefore, the feature (contextual information) at every level will be extracted.
- Secondly, the loss function of the network can be modified based on the fact that the instability in loss function (line IoU loss)(Zheng et al., 2022) occurs due to the norm calculation at each every point. This can be modified to model a line using the set of points, thereby considering the line as a whole unit to improve the localization accuracy.

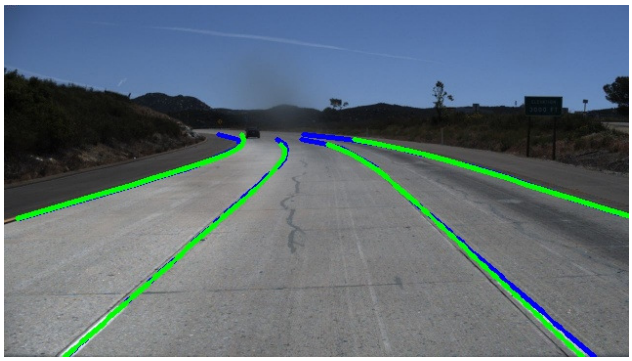

Figure 7. Sample 1: LaneATT qualitative results on TuSimple dataset (On improved accuracy model). Blue lines are ground-truth, while green and red lines are true-positives and false positives, respectively.

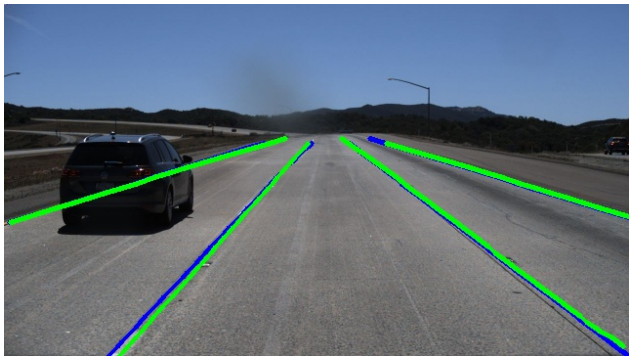

Figure 8. Sample 2: LaneATT qualitative results on TuSimple dataset (On improved accuracy model). Blue lines are ground-truth, while green and red lines are true-positives and false positives, respectively.

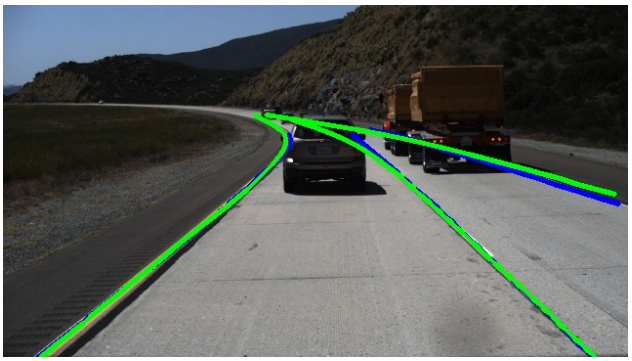

Figure 9. Sample 3: LaneATT qualitative results on TuSimple dataset (On improved accuracy model). Blue lines are ground-truth, while green and red lines are true-positives and false positives, respectively.

5. Papers

1. 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.

2. Tu Zheng, Yifei Huang, Yang Liu, Wenjian Tang, Zheng Yang, Deng Cai, Xiaofei He, Fabu, Zhejiang UniversityCLRNet: Cross Layer Refinement Network for Lane Detection In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022.

3. Zhengyang Feng, Shaohua Guo, Xin Tan, Ke Xu, Min Wang, Lizhuang Ma; Rethinking Efficient Lane Detection via Curve Modeling Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022, pp. 17062-17070

(Li et al., 2020) (Torres et al., 2020)

References

GitHub - lucastabelini/LaneATT: Code for the paper entitled "Keep your Eyes on the Lane: Real-time Attention-guided Lane Detection" (CVPR 2021) — github.com. https://github.com/lucastabelini/LaneATT, a. [Accessed 18-Oct-2022].

GitHub - mo-vic/BezierLaneNet: Exploring the use of bezier curves for lane detection —— A baseline model. — github.com. https://github.com/mo-vic/BezierLaneNet, b. [Accessed 18-Oct-2022].

Li, X., Li, J., Hu, X., and Yang, J. Line-cnn: End-to-end traffic line detection with line proposal unit. *IEEE Transactions on Intelligent Transportation Systems*, 21(1):248–258, 2020. doi: 10.1109/TITS.2019.2890870.

Tabelini, L., Berriel, R., Paixão, T. M., Badue, C., De Souza, A. F., and Oliveira-Santos, T. Keep your eyes on the lane: Real-time attention-guided lane detection, 2020. URL https://arxiv.org/abs/2010.12035.

Torres, L. T., Berriel, R. F., Paixão, T. M., Badue, C., Souza, A. F. D., and Oliveira-Santos, T. Polylanenet: Lane estimation via deep polynomial re-

gression. *CoRR*, abs/2004.10924, 2020. URL https://arxiv.org/abs/2004.10924.

Zheng, T., Huang, Y., Liu, Y., Tang, W., Yang, Z., Cai, D., and He, X. Clrnet: Cross layer refinement network for lane detection, 2022. URL https://arxiv.org/abs/2203.10350.