Mixer-Based Lidar Lane Detection Network and Dataset for Urban Roads

Donghee Paek, Seung-Hyun Kong, Kevin Tirta Wijaya
{donghee.paek, skong, kevin.tirta}@kaist.ac.kr

Abstract

Accurate lane detection under various road conditions is a critical function for autonomous driving. Generally, when detected lane lines from a front camera image are projected into a bird’s-eye view (BEV) for motion planning, the resulting lane lines are often distorted. And convolutional neural network (CNN)-based feature extractors often lose resolution when increasing the receptive field to detect global features such as lane lines. However, Lidar point cloud has little image distortion in the BEV-projection. Since lane lines are thin and stretch over entire BEV image while occupying only a small portion, lane lines should be detected as a global feature with high resolution. In this paper, we propose Lane Mixer Network (LMN) that extracts local features from Lidar point cloud, recognizes global features, and detects lane lines using a BEV encoder, a Mixer-based global feature extractor, and a detection head, respectively. In addition, we provide a world-first large urban lane dataset for Lidar, K-Lane, which has maximum 6 lanes under various urban road conditions. We demonstrate that the proposed LMN achieves the state-of-the-art performance, an F1 score of 91.67%, with K-Lane. The K-Lane, LMN training code, pre-trained models, and total dataset development platform are available at [github].

1 Introduction

Lane detection in an autonomous vehicle is to detect the accurate positions and curvature of the ego lane and neighboring lanes. Since an autonomous vehicle plans the local path using the lane detection results, the lane detection should be robust to various conditions (night time, day light, crowded, etc.) and situations including lane line occlusion. The existing image processing-based lane detection techniques are vulnerable to situations when multiple lane lines are occluded, because the heuristic techniques rely on line fitting with detected pixels (Schreiber, Alefs, and Clabian 2005, Dong et al. 2012).

Recently, lane detection has been significantly improved with deep learning, in which the deep learning network extracts the features of lanes, produces the lane detection results, and is trained to acquire the same results as the given ground-truth data. Therefore, when a large dataset with accurate label is given, a deep learning techniques such as convolutional neural network (CNN)-based lane detection techniques show a superior performance than heuristic lane detection techniques. Besides the large-volume lane datasets that have recently been published (e.g., CULane (Pan et al. 2018), TuSimple (TuSimple 2017)), thanks to the development of the CNN technology (He et al. 2016), the CNN-based camera lane detection techniques show excellent performance in terms of detection accuracy and real-time operation (Ibelini et al. 2021, Liu et al. 2021).

However, CNN-based camera lane detection techniques have a few inherent problems. First, the camera sensors suffer from poor lighting conditions, such as dark and harsh lights (Pan et al. 2018). Second, it is often necessary to project front camera images into 2-dimensional (2D) bird’s eye view (BEV), since motion planning for autonomous driving is generally performed with a BEV image, which causes lane mark distortions (Bai et al. 2018). Third, when detections are obtained near the vanishing line of an image from a front camera (Wang et al. 2014), it is harder to accurately project to BEV, which may limit the motion planning to a shorter distance.

On the other hand, lane detection from a Lidar point cloud does not suffer from the three above stated problems. Lidar is not affected by lighting conditions and illumination changes. And when BEV image is obtained by the projection of a Lidar point cloud, the image is neither distorted nor affected by a vanishing line. Although Lidar has multiple advantages over camera in lane detection, little studies have been introduced in the literature, which may be because of the following two reasons: Firstly, unstructured nature of point cloud data does not allow direct application of traditional CNN models. Recent solutions employ PointNet (Qi et al. 2017) for a feature encoder from a point cloud, or voxelize the point clouds and scatter them into grid coordinates as in Point Pillars (Lang et al. 2019) before applying CNN. Another is the lack of large public dataset for Lidar lane detection.

In fact, the lane lines are a global feature in the BEV image, such that they are thin and stretch over the entire BEV image, while only occupying a small portion of the image. This characteristics of the lane lines is not fully exploited by CNN-based lane detection networks for both camera and Lidar. In this paper, we propose Lane Mixer Network (LMN) that fully exploits the special characteristics of the lane lines by utilizing an MLP-Mixer (Tolstikhin et al. 2021)-based
global feature extractor. The proposed LMN consists of a BEV encoder to project the Lidar point cloud into a horizontal plane, MixSegNet as a global feature extractor, and a detection head, where the BEV encoder can be any of the two most popular BEV encoders introduced in the literature, the Point Pillars (Lang et al. 2019) and a 2D projector with a CNN based point projector (Simony et al. 2018). And we demonstrate that the proposed MixSegNet achieves a significant improvement in performance for both of the two different BEV encoders. The proposed LMN also exhibits a fast inference time, by producing lane detection results for every point cloud scan from Lidar. We also provide a large multi-modal urban lane dataset for Lidar lane detection, called K-Lane dataset that contains Lidar point cloud for various (normal and sever) urban conditions and scenarios. The performance demonstration of the proposed LMN is carried out for K-Lane, and we show that the proposed LMN achieves a superior performance in all road conditions, which is different from other lane detection techniques, tested with CULane (Pan et al. 2018), that shows significant degradation for various uncommon road conditions.

This paper is organized as follows. Section 2 introduces prior studies related to this paper and the topic of this paper, Section 3 introduces the large multi-modal urban lane dataset for Lidar lane detection provided with this paper, and the proposed LMN in detail. Section 4 shows the experiment setup and the results of the experiments. We draw our conclusion in Section 5, and introduce the performance evaluation metric and the ablation studies in Appendix.

2 Related Works

Deep Learning-based Lane Detection Techniques for Camera As deep learning techniques are studied and precisely labeled large camera dataset (Pan et al. 2018) for various road environments become available, there have been a significant advancement in the lane detection techniques for camera images. Compared with the early rule-based techniques (Dong et al. 2012; Deng and Wu 2018), CNN-based lane detection techniques are more adaptive to various weather changes and show less performance deterioration by occlusion. In these techniques, lanes are predicted by a lane detection head based on local features extracted by CNN (He et al. 2016), and the performance is improved with development of lane detection heads that exploit the features of lane lines; Segmentation-based techniques such as (Pan et al. 2018) detect lanes by assigning classes (e.g., lanes, and backgrounds) to each predicted pixel, which may cause discontinuous lane lines and marks and additional clustering is introduced to compensate (Abualasbed et al. 2021). Anchor-based techniques detect lanes through the regression of coordinate change of anchors that are designated initially. By defining anchors as segments of various lengths and angles, (Tabelini et al. 2021) accomplishes an excellent performance with several datasets. Row-wise detection techniques divide the entire image into grids, and recognize lanes in each row of the grids. (Qin, Wang, and Li 2020) improves the lane detection performance by implementing the domain knowledge, that lane lines continue across the rows, into a cost function. (Liu et al. 2021) proposes a two-stage lane detection technique that combines the row-wise detection with a conditional convolution (Yang et al. 2019) in the detection head and achieves a state-of-the-art (SOTA) performance in several datasets. However, camera lane detection techniques have some inherent problems as they rely on camera. The CULane benchmark of most of the camera lane detection techniques show significant performance drop (about 20%) for night time and dazzling light conditions from their daytime performance (Liu et al. 2021; Qin, Wang, and Li 2020).

Early Lane Detection Techniques for Lidar In early studies, lane points are detected by thresholding the measured intensity (or reflectivity). (Ogawa and Takagi 2006) fuses several frames of Lidar scans containing point intensities and estimate the lane parameters using a Kalman filter. (Lindner et al. 2009) uses a fixed polar grid map to store point intensities and filter the lane candidates with thresholding along azimuth angles. (Hernandez, Hoang, and Jo 2014) introduces a clustering approach, where the filtered lane points are clustered using DBSCAN (Ester et al. 1996) to predict lane marks. However, these heuristic techniques rely on several pre-defined thresholding parameters, and, therefore, it is not very adaptable to different environments.

Deep Learning-based Lane Detection Techniques for Lidar Deep learning-based lane detection studies for Lidar have not been actively conducted due to the absence of large open datasets, and only some studies with their own partial Lidar datasets are introduced in the literature. (Bai et al. 2018) proposes a deep learning-based technique that combines 2D BEV images developed with the Lidar point cloud and the front camera image for lane detection. And (Martinek et al. 2020) proposes a CNN-based technique that uses BEV images from point clouds to detect ego-lanes, and tests the technique in an uncrowded highway.

MLP-Mixer Recently, ViT (Dosovitskiy et al. 2021) and MLP-Mixer (Tolstikhin et al. 2021) greatly improve the image classification performance through Transformer (Vaswani et al. 2017) implementing attention mechanism. ViT (Vision Transformer) divides input image is into unit patches and applies Transformer encoder to each patch for image classification. However, ViT employs three independent MLPs (Multilayer Perceptrons) for each attention mechanism, for which high computational cost and large model size are necessary. On the other hand, MLP-Mixer implements attention mechanism with one MLP, which results in a fast inference with a small model size and achieves comparable performance to ViT. However, MLP-Mixer and ViT can achieve the highest performance in the presence of very large dataset, such as JFT-300M (Sun et al. 2017), due to its low model inductive bias.

3 K-Lane Dataset and Proposed Technique

In this section, we first introduce the large multi-modal urban lane dataset, K-Lane, which contains Lidar point clouds for various (normal, uncommon, severe) urban conditions and scenarios. Subsection 3.2 introduces the proposed Lidar lane detection technique based on the Mixer network, LMN (Lane Mixer Network), in detail.
3.1 K-Lane: Lidar Lane Dataset for Various Urban Roads

K-Lane is the first large open LiDAR lane dataset that consists of Lidar point cloud and their corresponding RGB images collected for various roads in urban environments.

The K-Lane presented with this paper is collected using an Ouster OS2-64 Lidar sensor (Ouster 2020) that has 64 channels and a maximum range of 240m. The ground truth labels are produced by projecting the Lidar point cloud into a BEV image, thresholding the intensity measurements, and drawing a one-pixel-wide line for each lane line on the road. There are total 3205 data frames that split into 2798 frames for the training set, and 407 frames for the testing set, where each set contains various conditions and challenging urban scenarios, for example:

- different lighting conditions including daytime, night time, cloudy, and clear skies;
- crowded traffic with lane occlusions by other vehicles; and
- converging, diverging, and curve lanes of up to six lanes.

As a result, compared to the conventional lane detection datasets, K-Lane has multiple advantages:

- K-Lane is collected under various urban roads and conditions as stated above, while KITTI (Fritsch, Kuehnl, and Geiger 2013) includes only daytime urban roads and TuSimple (TuSimple 2017) only includes a daytime highway.

- K-Lane distinguishes lane instances (lane class) and labels with precise lane location (pixel level), whereas KiTITI has labels for ego-lane or drivable path without distinction between lane classes. For example, the ego-lane has lane lines in yellow and green as shown in Figure 1. We depict with orange and red colors left to the yellow line, and blue and purple colors right to the green line, as shown in Figure 1, and

- K-Lane has larger number of labeled lanes (e.g., maximum 6 lanes), while KITTI, and CULane (Pan et al. 2018) have only up to 4 lanes.

In addition to K-Lane, we provide a total dataset development platform that has easy to use GUI-based annotation, evaluation, and visualization tools as shown in Appendix B. This is for the research community to expand the dataset easily regardless of the Lidar sensor models.

To standardize the evaluation of the network being developed, we provide the F1 metric in the platform, where F1 metric represents a harmonic mean between precision and recall. Details of mathematical definition of F1 metric are in Appendix C.

3.2 The Proposed Lane Mixer Network

As shown in Fig. 2, the proposed LMN consists of three modules to detect lanes directly from BEV images required for path planning; BEV Encoder, Global Feature Extractor (GFE), and Lane Detection Head.

3.2.1 BEV Encoder

BEV encoder projects 3D (3-dimensional) point cloud into a horizontal plane to produce 2D pseudo-BEV image. A large number of heuristic path planning algorithms, such as A* (Hart, Nilsson, and Raphael 1968), RRT* (Karaman and...
Frazzoli (2011), and End-to-End autonomous driving algorithms such as Chen et al. (2020) require lane lines expressed on 2D BEV images. Therefore, the proposed LMN uses two most common 2D BEV encoders, as shown in Fig. 2, and explained below, but do not use voxel encoder (Zhou and Tuzel 2018) that requires computationally expensive 3D convolutions.

One is the point projector (Simony et al. 2018; Ku et al. 2018) that projects point clouds into xy-horizontal plane and produces pseudo-BEV image using CNN. In this case, three additional information (x, intensity, and reflectivity) other than x and y of the point cloud is used to generate three channels of the produced pseudo-BEV image. And in order not to lose lane information while improving the real-time speed, we use only to a depth of the CNN where the feature map becomes the 1/64 of the pseudo-BEV image. Specifically, we may use the first 14, 28, and 41 convolutional layers of the ResNet-18, ResNet-34, and ResNet-50, respectively. Note that we denote the partial ResNet’s as ResNet14, ResNet28, and ResNet41 in the ablation studies in Appendix E, and that ResNet41 is the one used for the proposed LMN.

The other is the pillar encoder based on Point Pillars that has relatively small network size (Lang et al. 2019). Pillar encoder has slightly lower performance than the CNN-based point projector, but has a faster speed. Therefore, in this paper, pillar encoder is presented as an alternative to the point projector for an improved real-time performance. As shown in Fig. 2 the pillar encoder aligns the point cloud in each grid of the horizontal plane to generate stacked pillars of size $N_g \times N_c \times N_p$, where $N_g$ is the total number of grids, $N_c$ is the point feature components, and $N_p$ is the maximum number of points present on the grid. Then, a simplified version of PointNet (Lang et al. 2019) consisting of shared MLP’s of size $N_c \times C$ is applied to each grid to extract pseudo-BEV image of size $H_{bev} \times W_{bev} \times C$. In this paper, considering that the lane has a width of about 16cm and is long in the longitudinal direction of the road, the grid size is set to 32cm in the longitudinal direction and 16cm in the transverse direction.

3.2.2 MixSegNet for GFE

In general, the characteristics of features to be extracted should be considered in the design of neural networks (He et al. 2016), and CNN has been effective feature extractors for image classification (Krizhevsky, Sutskever, and Hinton 2012), and object detection (Liu et al. 2016).

In general, the lane is a global feature that occupies a very small portion of the BEV image but is thin and long stretched over the entire image, whereas each object (such as Car, Cyclist, Pedestrian, and etc.) on the road occupies a noticeable portion of the image and often separated or occluded. Therefore, it is necessary to calculate the correlation between distant grids within the feature map for lane detection throughout the road even in situations of occlusion. Thus, a reliable lane detection requires a GFE capable of calculating global correlations with high resolution.

As shown in Fig. 3(b), CNN feature extractors generally process through multiple convolution layers to find correlations between distant grids (or patches) shown in Fig. 3(a). As a result, CNN detects an object in a global feature with low resolution ($N_2 \ll N_0$ in (b) of Fig. 3) from a deep layer. Therefore, many CNN-based lane detection techniques (Pan et al. 2018) utilize similar ideas to the Feature Pyramid Network (Lin et al. 2017) which concatenates feature maps at different depths for improved detection. However, in FPN, a high resolution feature map from a lower layer has only local features, which is not enough to express the global feature with high resolution. Subsection 4.1 shows heatmaps as the basis for this hypothesis.

Conversely, the proposed MixSegNet, a Mixer-based GFE, divides the pseudo-BEV image into small patches and finds global correlation between all patches through channel mixing (i.e., channel-wise MLP) and token mixing (i.e., patch-wise MLP). Therefore, as shown in Fig. 3(c), MixSegNet is useful for detecting global features such as lane lines, since it repeatedly performs global correlation from high-resolution feature maps without reducing the feature map ($N_1 = N_0$) size. The proposed MixSegNet has high attention performance, too. Since Mixer realizes self-attention mechanism using MLP, it requires a large training data for its low model inductive bias in the image classification tasks. However, the proposed MixSegNet, despite calculating global correlation with MLPs, can perform reliable lane detection with a relatively small number of data (e.g., 2904 frames) and a fast learning time, which is discussed (and compared to CNN-based GFE) in detail in subsection 4.1.

Figure 4 shows the details of the MixSegNet structure; Function (A) reshapes the pseudo-BEV image into a 2D tensor for global correlation. Function (B) performs patch-wise linear transform, and functions (C) and (D) perform...
global correlation through token mixing and channel mixing, respectively. Function (E) reshapes the Mixer block output to the size required for the lane detection head. Since the channel size of the reshaped output image depends on the hidden dimension \( N_h \) of the Mixer block, it can be smaller than that of the input BEV image, \( C \). This may cause Bottleneck (Szegedy et al. 2015), so function (F) applies 1x1 convolution and produces the final output feature map for the detection head. Note that ablation studies in Appendix E shows lane detection performance for various change in hyper-parameters of the MixSegNet, where Projector41-MixSegNet3 (i.e., \( N_D=3 \)) and Pillars-MixSegNet5 (i.e., \( N_D=5 \)) are found as the best performing networks for point projector and Point Pillars, respectively.

### 3.2.3 Detection Head and Loss Function

To design the detection head, we formulate the lane detection problem as a multi-class segmentation problem, where each pixel is assigned a class and a confidence score. This is because there are multiple advantages with the multi-class segmentation-based approaches over other approaches; First, binary-class (i.e., lane or background) segmentation-based approaches (Abualsaud et al. 2021) need additional post-processing to assign a new lane class for a new prediction, while multi-class segmentation-based approaches assign a class directly to a prediction. Second, anchor-based approaches use fixed-shaped anchors that limits the lane shape detection. However, multi-class segmentation approaches can predict any shape of lane lines (e.g., converging, diverging lanes) because the predictions are done at the pixel or grid levels. Note that lane classes and various shapes are important for motion planning, for which the ego vehicle can plan inter-lane motions or recognize lane merging and separation.

We design two segmentation heads in the LMN detection head, for which we develop two different losses to supervise each head. In the K-Lane dataset and others, the number of lane samples are significantly fewer compared to the number of background samples on each frame. Seeing the imbalance between the lane samples and the background samples, we consider the soft dice (Sudre et al. 2017) loss for the confidence loss. For the classification head, we choose the grid-wise cross-entropy loss (Ronneberger, Fischer, and Brox 2015) that has been widely used in multi-class classification problems, leading the network to learn to maximize the probability of the correct lane class during training. As a result, the total loss function is the summation of both the soft-dice loss and the cross-entropy loss. Details including mathematical expressions and network structure are in Appendix D.

## 4 Experiments and Comparison

In this section, we compare MixSegNet to the conventional CNN-based GFE in subsection 4.1, the proposed LMN to the conventional heuristic Lidar lane detection technique in subsection 4.2, and the proposed LMN to the recent camera lane detection techniques in subsection 4.3.

### Experiments Setup

We use RTX3090 GPUs for training the networks on the K-Lane dataset for 120 epochs using Adam optimizer (Kingma and Ba 2017) with a batch size 4 and a learning rate of 0.0002. All training and evaluations are implemented with PyTorch 1.7.1 (Paszke et al. 2019) on an Ubuntu 18.04 machine. All of the codes for the networks and the total development platform are available on [github].

### 4.1 MixSegNet vs. CNN for GFE

Figure 5: ResNetFPN is a CNN-based GFE compared with the proposed MixSegNet. There are 5 residual blocks composed of 3, 5, 5, 5, and 5 convolutional layers in the ResNet side. Each block produces a feature map that is 4 times smaller than the input feature map, and the Feature Pyramid Network (FPN) concatenates the feature maps from each block to produce the final output feature map.

In this subsection, we compare the performance of Projector41-MixSegNet3 and Pillars-MixSegNet5 as the
Table 1: F1-score comparison of MixSegNet to ResNetFPN for various network depths.

| Model                     | Total | Daytime | Night | Crowded | Uncrowded | Curve | Occlusion | Size (MB) | Speed (ms) |
|---------------------------|-------|---------|-------|---------|-----------|-------|-----------|-----------|------------|
| Projector41-MixSegNet1    | 91.15 | 91.56   | 93.26 | 87.40   | 93.84     | 88.78 | 84.70     | 72.5      | 93.73      |
| Projector41-MixSegNet3    | 91.67 | 92.15   | 93.47 | 88.34   | 94.07     | 89.85 | 85.29     | 96.0      | 94.24      |
| Projector41-ResNet8FPN    | 85.09 | 86.02   | 89.27 | 77.54   | 90.50     | 81.20 | 72.20     | 38.3      | 85.85      |
| Projector41-ResNet13FPN   | 85.29 | 85.95   | 89.52 | 77.87   | 90.60     | 80.94 | 74.07     | 42.9      | 87.82      |
| Projector41-ResNet23FPN   | 85.40 | 86.08   | 90.20 | 77.84   | 90.79     | 79.87 | 74.10     | 103.7     | 95.10      |
| Pillars-MixSegNet1        | 84.42 | 85.18   | 86.56 | 79.22   | 88.14     | 77.80 | 78.78     | 37.8      | 69.26      |
| Pillars-MixSegNet5        | 89.78 | 90.41   | 91.68 | 86.07   | 92.45     | 86.70 | 85.32     | 84.9      | 71.00      |
| Pillars-ResNet8FPN        | 64.41 | 65.65   | 68.78 | 56.15   | 70.36     | 57.77 | 49.45     | 3.6       | 60.61      |
| Pillars-ResNet13FPN       | 72.41 | 73.22   | 76.45 | 64.95   | 77.78     | 60.98 | 59.70     | 8.2       | 62.11      |
| Pillars-ResNet23FPN       | 74.22 | 75.19   | 78.70 | 66.19   | 79.98     | 61.44 | 65.47     | 69.0      | 69.95      |

4.2 The Proposed LMN vs. Heuristic Lidar Lane Detection

In the heuristic Lidar lane detection technique, we first project pointcloud into a BEV image and apply a thresholding operation to filter out low-intensity points as in (Hernandez, Hoang, and Jo 2014). The remaining points are then clustered using, for example, DBSCAN (Ester et al. 1996) and then fitted by a first order polynomial to create smooth lane lines.

During the experiments, we observe multiple instances where the heuristic technique are unreliable; First, when a strong source of light illuminates a spot on the ground plane, as shown in Fig. 7 (b), it results in false positives (FPs). Second, when lane marks are occluded by an object, the heuristic Lidar lane detection cannot infer the presence of lane marks, leading to a significant number of false negatives (FNs) as shown in Fig. 7 (d). However, the proposed...
| Technique | Model | Total | Normal | Night | Crowded | Uncrowded | Curve | Occlusion |
|-----------|-------|-------|--------|-------|---------|-----------|-------|-----------|
| Camera    | SCNN (Pan et al. 2018) | 71.6  | 90.6  | 66.1  | 69.7    | -  | 64.4     | - |
| Lane      | UFLD-ResNet-34 (Qin, Wang, and Li 2020) | 72.3  | 90.7  | 66.7  | 70.2    | -  | 69.5     | - |
| Detection | CurveLane-NAS-L (Xu et al. 2020) | 74.8  | 90.7  | 68.9  | 72.3    | -  | 68.4     | - |
|           | LaneATT-ResNet-122 (Tabelini et al. 2021) | 77.0  | 91.7  | 76.2  | 75.0    | -  | 64.1     | - |
|           | LaneAF-DLA-34 (Abualsaud et al. 2021) | 77.4  | 91.8  | 73.0  | 75.6    | -  | 72.7     | - |
|           | CondLaneNet-Large (Liu et al. 2021) | 79.5  | 93.5  | 74.8  | 77.4    | -  | 75.2     | - |
| Lidar     | LMN (Projector41-MixSegNet3) | 91.7  | 92.2  | 93.5  | 88.3    | 94.1 | 89.9  | 85.3   |
| Lane      | LMN (Pillars-MixSegNet5) | 89.8  | 90.4  | 91.7  | 86.1    | 92.5 | 86.7  | 85.3   |
| Detection | Heuristic Lane Detection | 29.8  | 29.7  | 33.7  | 21.5    | 35.6 | 15.7  | 22.2  |

Table 2: Lane detection performance comparison of Camera and Lidar including the proposed LMN.

LMN can produce reliable lane detection results for the two scenarios. As the LMN learns global context features of the scene, a bright illuminated road spot or partial occlusion of lane marks hardly deteriorate the lane detection performance.

**Figure 7:** Comparison between the proposed LMN (Projector41-MixSegNet3) (in (a) and (c)) and heuristic Lidar lane detection (in (b) and (d)). When a strong source of illumination appears on the scene, (b) the heuristic method fails, but (a) the proposed LMN is not affected. When lane marks are occluded, (d) the heuristic method cannot infer the lanes, but (c) the proposed LMN is able to infer the occluded lane lines.

### 4.3 LMN vs. Camera Lane Detection

Most state-of-the-art lane detection techniques in the literature employ camera images. To make a reasonable comparison between the proposed LMN technique, Projector41-MixSegNet3, to the available state-of-the-art, we divide the K-Lane test dataset into several categories based on the road conditions. This division approach is similar to the widely-used CULane (Pan et al. 2018) dataset. We divide our dataset into day, night, crowded, uncrowded, occluded, and curvy road conditions.

In the camera-based lane detection techniques, we observe a significant drop in performance for the challenging conditions from a normal condition. For example, CondLaneNet (Liu et al. 2021) has a 93.5 F1-score for normal condition but shows only a 74.8 F1-score for the night condition. In contrast, the proposed LMN, Projector41-MixSegNet3, can maintain similar performance for both day and night times as shown in Table 2. This shows an inherent advantage of using Lidar as the primary sensor for the lane detection. Next, the performance of the detection techniques in crowded traffics is considered, where the obstruction of lane marks often occurs, resulting in no measurements on a large portion of the road. As shown in Table 2, camera-based lane detection techniques reveal a significant performance drop for the crowded traffic condition, where CondLaneNet shows 16.1 points drop compared to the normal condition. However, the proposed LMN only shows at most 3.9 points drop for the crowded traffic condition.

In the K-Lane dataset, there is an additional condition for occlusion, where at least four of the existing lanes are obscured by other objects on the road. In such a scenario, the proposed LMN still produce a reliable detection performance, with only 6.9 points drop compared to the uncrowded traffic condition. This result demonstrates that the proposed LMN is robust to harsh conditions such as the lane mark obstructions.

### Conclusion

In this paper, we have proposed Lane-detection Mixer Network (LMN), a Mixer-based lane detection deep neural network, which extracts 2D (2-dimensional) bird-eye-view (BEV) image from Lidar point cloud and detects lane lines with high accuracy and robustness. The proposed LMN has been realized in two different basic forms, Projector-MixSegNet and Pillars-MixSegNet, to accommodate two widely used 2D BEV generation models, point projector and pillar encoder, respectively. In conjunction with the proposed LMN, this paper provides K-Lane, a large urban road dataset for Lidar lane detection containing various road conditions, and have demonstrated that the lane detection performance of the two proposed networks shows high accuracy and robustness under the various urban road conditions such as severe illumination change, crowded and uncrowded traffics, curved lane lines, and lane occlusion. This performance under the various road conditions is by far the superior than those of other lane detection deep neural networks for camera image and Lidar point cloud, which testifies that the proposed LMN has achieved the state-of-the-art (SOTA) performance.

### References

Abualsaud, H.; Liu, S.; Lu, D.; Situ, K.; Rangesh, A.; and Trivedi, M. M. 2021. LaneAF: Robust Multi-Lane Detection with Affinity Fields. arXiv:2103.12040.

Bai, M.; Mattyus, G.; Homayounfar, N.; Wang, S.; Lakshmikanth, S. K.; and Urtasun, R. 2018. Deep multi-sensor
lane detection. In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 3102–3109. IEEE.

Chen, D.; Zhou, B.; Koltun, V.; and Krähenbühl, P. 2020. Learning by cheating. In Conference on Robot Learning, 66–75. PMLR.

Deng, G.; and Wu, Y. 2018. Double lane line edge detection method based on constraint conditions Hough transform. In 2018 17th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES), 107–110. IEEE.

Dong, Y.; Xiong, J.; Li, L.; and Yang, J. 2012. Robust lane detection and tracking for lane departure warning. In 2012 International Conference on Computational Problem-Solving (ICCP), 461–464. IEEE.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; Uszkoreit, J.; and Houlsby, N. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. arXiv:2010.11929.

Ester, M.; Kriegel, H.-P.; Sander, J.; Xu, X.; et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In kdd, volume 96, 226–231.

Fritsch, J.; Kuehnl, T.; and Geiger, A. 2013. A new performance measure and evaluation benchmark for road detection algorithms. In 16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013), 1693–1700. IEEE.

Hart, P. E.; Nilsson, N. J.; and Raphael, B. 1968. A formal basis for the heuristic determination of minimum cost paths. IEEE transactions on Systems Science and Cybernetics, 4(2): 100–107.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, 770–778.

Hernandez, D. C.; Hoang, V.-D.; and Jo, K.-H. 2014. Lane surface identification based on reflectance using laser range finder. In 2014 IEEE/SICE International Symposium on System Integration, 621–625. IEEE.

Karaman, S.; and Frazzoli, E. 2011. Sampling-based algorithms for optimal motion planning. The international journal of robotics research, 30(7): 846–894.

Kingma, D. P.; and Ba, J. 2017. Adam: A Method for Stochastic Optimization. arXiv:1412.6980.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In Advances in Neural Information Processing Systems, volume 25.

Ku, J.; Mozifian, M.; Lee, J.; Harakeh, A.; and Waslander, S. L. 2018. Joint 3d proposal generation and object detection from view aggregation. In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 1–8. IEEE.

Lang, A. H.; Vora, S.; Caesar, H.; Zhou, L.; Yang, J.; and Beijbom, O. 2019. Pointpillars: Fast encoders for object detection from point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 12697–12705.

Lin, T.-Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; and Belongie, S. 2017. Feature pyramid networks for object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, 2117–2125.

Lindner, P.; Richter, E.; Wanielik, G.; Takagi, K.; and Isogai, A. 2009. Multi-channel lidar processing for lane detection and estimation. In 2009 12th International IEEE Conference on Intelligent Transportation Systems, 1–6. IEEE.

Liu, L.; Chen, X.; Zhu, S.; and Tan, P. 2021. CondLaneNet: a Top-to-down Lane Detection Framework Based on Conditional Convolution. arXiv:2105.05003.

Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; and Berg, A. C. 2016. Ssd: Single shot multibox detector. In European conference on computer vision, 21–37. Springer.

Martinek, P.; Pucea, G.; Rao, Q.; and Sivalingam, U. 2020. Lidar-based Deep Neural Network for Reference Lane Generation. In 2020 IEEE Intelligent Vehicles Symposium (IV), 89–94. IEEE.

Ogawa, T.; and Takagi, K. 2006. Lane recognition using on-vehicle lidar. In 2006 IEEE Intelligent Vehicles Symposium, 540–545. IEEE.

Ouster. 2020. OS2-64. https://ouster.com/products/os2-lidar-sensor/ Accessed: 2021-09-08.

Pan, X.; Shi, J.; Luo, P.; Wang, X.; and Tang, X. 2018. Spatial as deep: Spatial cnn for traffic scene understanding. In Thirty-Second AAAI Conference on Artificial Intelligence.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. 2019. Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems, 32: 8026–8037.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, 652–660.

Qin, Z.; Wang, H.; and Li, X. 2020. 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, 276–291. Springer.

Ronneberger, O.; Fischer, P.; and Brox, T. 2015. U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention, 234–241. Springer.

Schreiber, D.; Alefs, B.; and Clabian, M. 2005. Single camera lane detection and tracking. In Proceedings. 2005 IEEE Intelligent Transportation Systems, 2005., 302–307. IEEE.

Simony, M.; Milzy, S.; Amendey, K.; and Gross, H.-M. 2018. Complex-yolo: An euler-region-proposal for real-time 3d object detection on point clouds. In Proceedings of the European Conference on Computer Vision (ECCV) Workshops, 0–0.
Sudre, C. H.; Li, W.; Vercauteren, T.; Ourselin, S.; and Cardoso, M. J. 2017. Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. Springer.

Sun, C.; Shrivastava, A.; Singh, S.; and Gupta, A. 2017. Revisiting unreasonable effectiveness of data in deep learning era. In Proceedings of the IEEE international conference on computer vision, 843–852.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition, 1–9.

Tabelini, L.; Berriel, R.; Paixao, T. M.; Badue, C.; De Souza, A. F.; and Oliveira-Santos, T. 2021. 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, 294–302.

Tolstikhin, I.; Houlsby, N.; Kolesnikov, A.; Beyer, L.; Zhai, X.; Unterthiner, T.; Yang, J.; Steiner, A.; Keysers, D.; Uszkoreit, J.; Lucic, M.; and Dosovitskiy, A. 2021. MLP-Mixer: An all-MLP Architecture for Vision. arXiv:2105.01601.

TuSimple. 2017. Tusimple benchmark. [https://github.com/TuSimple/tusimple-benchmark](https://github.com/TuSimple/tusimple-benchmark) Accessed: 2021-09-08.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need. In Advances in neural information processing systems, 5998–6008.

Wang, C.; Zhang, H.; Yang, M.; Wang, X.; Ye, L.; and Guo, C. 2014. Automatic parking based on a bird’s eye view vision system. Advances in Mechanical Engineering, 6: 847406.

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. In Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XV 16, 689–704. Springer.

Yang, B.; Bender, G.; Le, Q. V.; and Ngiam, J. 2019. Cond-Conv: Conditionally Parameterized Convolutions for Efficient Inference. In Wallach, H. M.; Larochelle, H.; Beygelzimer, A.; d’Alché Buc, F.; Fox, E. A.; and Garnett, R., eds., Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada, 1305–1316.

Zhou, Y.; and Tuzel, O. 2018. Voxelnet: End-to-end learning for point cloud based 3d object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, 4490–4499.

Appendix

We provide detailed descriptions of K-Lane, dataset development platform, evaluation metric, and the detection head of the proposed LMN, in Appendix A, B, C, and D, respectively. Lastly, subsection E shows ablation study for MixSegNet hyper-parameters that are key parameters of the proposed Projector-MixSegNet and Pillars-MixSegNet.

Appendix A. Introduction to K-Lane

Section A provides the three details of the K-Lane: sensor suite used for K-Lane dataset collection, four sequences of the K-Lane, and the dataset composition of the K-Lane.

Details of Sensor Suite K-Lane is the world’s first Lidar lane dataset for various urban roads and conditions, acquired using OS2-64 Lidar sensor (Ouster 2020) whose details are shown in Table 3 and BFLY-PGE-23S6C-C camera that produces RGB images of 1920 × 1200 size in 40Hz. As shown in Fig. 8, the camera is located at the center of the windshield to capture the front view of the vehicle, while the Lidar sensor is positioned 0.92m behind the camera and 1.69 m above the ground (0.74m above the camera) for collecting a wide surrounding view. High-accuracy camera-Lidar calibration parameters (i.e. intrinsic and extrinsic parameters), measured by the calibration program shown in Appendix B, are used, and about four front camera images are captured for each point cloud. The front camera images are useful when we provide annotation on the Lidar data and we can evaluate a trained Lidar lane detection network when we project the detected lane into the viewpoint of the front camera.

| Details                  | Value   |
|--------------------------|---------|
| Vertical resolution      | 64 channels |
| Maximum range            | 240 m   |
| Vertical field of view   | 22.5°   |
| Precision                | ±2.5 ~ 8 cm |
| Rotation rate            | 10 Hz   |

Table 3: Details of Lidar

![Figure 8. Sensor suite for K-Lane and the locations of the Lidar and camera](image)

Details of Each Sequence K-Lane dataset consists of four sequences that have different set of road conditions. The details of the sequences are shown in Table 4. For the test data, we provide additional description on each frame (i.e., curve, occlusion, zig-zag, merging, and number of lanes), annotated with description tool shown in Appendix B.

Details of Dataset Composition The K-lane has four directories for each of sequence where each directory contains:

- directory for each of sequence where each directory contains:
- curve, occlusion, zig-zag, merging, and number of lanes)
- details of the sequences are shown in Table 4. For the test data, we provide additional description on each frame (i.e., curve, occlusion, zig-zag, merging, and number of lanes), annotated with description tool shown in Appendix B.

Details of Dataset Composition The K-lane has four directories for each of sequence where each directory contains:

- curve, occlusion, zig-zag, merging, and number of lanes)
- details of the sequences are shown in Table 4. For the test data, we provide additional description on each frame (i.e., curve, occlusion, zig-zag, merging, and number of lanes), annotated with description tool shown in Appendix B.
files for the collected point cloud data, BEV point cloud tensor (i.e., stacked pillars shown in Fig. 2), BEV label, front (camera) images, calibration parameters, and description of the sequence, as shown in Table 5. Explanation about the interface for preprocessing the files is provided in Appendix B.

### Appendix B. Total Development Platform

In addition to the proposed LMN, a total development platform (TDP) to use K-Lane, to add more dataset, and to develop lane detection neural networks is opened to the public in the form of three programs: (1) TDC - Total Development (for training, evaluation, and test) Code, (2) EGT - Easy-to-use GUI-based annotation, evaluation, visualization Tools, and (3) RLT – ROS (Robot Operating System)-based Labeling Tools. We open this TDP to the public to support developers to mainly focus on the neural network development but not on other miscellaneous tasks. Total development code TDC is a complete neural network development platform that supports pre-processing of input data and label, total evaluation metric, handling input & output as dictionary type, modularization of LMN (BEV encoder, GFE, detection head), and dealing every experiments with every single configuration file. Therefore, TDC provides diverse convenience to developers. Easy-to-use GUI-based Tools EGT is a GUI program used with TDC. EGT provides the visualization of inference results for each scene as point cloud or camera image with projected lanes as shown in Fig. 9 (b), high-accuracy calibration of camera and Lidar sensors with specific points of the lanes as shown in Fig. 9 (c), and annotation of each frame easily with set of buttons as shown in Fig. 9 (d).

**ROS-based Labeling Tools** RLT provides an easy way to develop a labeled dataset for a Lidar and a front view camera, regardless of the Lidar and camera models. As shown in Fig. 10, RLT has three subplots (left, middle, and right). The left subplot provides an easy way for labeling by showing the intensity of point cloud in a BEV image, the middle one shows a synchronized front camera image for easy labeling of point cloud, and the right one shows saved labeled point cloud.

### Appendix C. Evaluation Metric

In this paper, we use the F1 score as the metric of network evaluation, since it represents the harmonic mean between precision and recall, where the F1 score can be expressed as

\[
F1 = \frac{1}{\frac{1}{\text{precision}} + \frac{1}{\text{Recall}}} = \frac{TP}{TP + 0.5(FP + FN)} \tag{1}
\]

where TP, FP, and FN are the numbers of true positives, false positives, and false negatives, respectively. Since the width of a lane mark is about only one grid width, 16cm, a slight deviation of up to one grid width in the prediction is possible. Therefore, we consider the one-grid offset in the accuracy evaluation is acceptable. This is comparable to the evaluation of the camera-based lane detection on the widely-used CULane (Pan et al. 2018) dataset, where lane marks are 30-pixels wide, and a true positive is counted when the prediction and the ground truth have at least an IoU of 0.5.

For the performance evaluation, we first define the confidence map label and the confidence map prediction

\[
x_{\text{conf}} = \{x_{\text{conf}i,j} | x_{\text{conf}i,j} \in \{0, 1\}, i \in I, j \in J\},
\]

\[
x_{\text{conf}}^{\hat{}} = \{\hat{x}_{\text{conf}i,j} | \hat{x}_{\text{conf}i,j} \in \{0, 1\}, i \in I, j \in J\},
\]

respectively, where \(I = \{1, ..., N\}, J = \{1, ..., M\}\), \(N\) is the number of rows, \(M\) is the number of columns. A thresholding operation is then applied to each \(\hat{x}_{\text{conf}i,j}\) such that

\[
\hat{x}_{\text{conf}i,j} = \begin{cases} 
1 & \text{if } \hat{x}_{\text{conf}i,j} > \sigma_{\text{conf}} \\
0 & \text{otherwise} \end{cases}
\]  \tag{2}

where the \(\sigma_{\text{conf}}\) is the confidence threshold for a prediction to be considered as a lane. And let \(\{x_{\text{conf}i+p,j+q}\} | p \in \{-1, 0, 1\}, q \in \{-1, 0, 1\}\) be a set containing the elements of \(x_{\text{conf}}\) at grid \((i,j)\) and its eight neighboring grids.

For a positive prediction at grid \((i,j)\), the true positives (TPs) and false positives (FPs) are computed as follows:

- TP occurs when there exists at least one positive label at grid \((i,j)\) or its eight neighboring grids, defined as

\[
TPs = \sum_{i}^{N} \sum_{j}^{M} \max \{\hat{x}_{\text{conf}i,j}, x_{\text{conf}i+p,j+q}\} 
\]

\[p \in \{-1, 0, 1\}, q \in \{-1, 0, 1\}\]  \tag{3}

| Sequence | Number of Frames | Location       | Time  | Traffic density | Road conditions                                                                 |
|----------|-----------------|----------------|-------|----------------|--------------------------------------------------------------------------------|
| Sequence 1 | 1495            | City (Se-jong) | Night | Uncrowded       | Curve, Diverging, Converging, zig-zag, many lanes (>4 lanes)                   |
| Sequence 2 | 703             | City (Dae-jeon)| Day   | Crowded         | Curve, Occlusion, Diverging, Converging, Many lanes                           |
| Sequence 3 | 227             | City (Dae-jeon)| Day   | Crowded         | Occlusion, Many lanes                                                         |
| Sequence 4 | 479             | University (KAIST) | Day   | Uncrowded       | Curve, Many lanes                                                             |

| Sequence 1 | 1495 | City (Se-jong) | Night | Uncrowded | Curve, Diverging, Converging, zig-zag, many lanes (>4 lanes) |
| Sequence 2 | 703  | City (Dae-jeon)| Day   | Crowded   | Curve, Occlusion, Diverging, Converging, Many lanes             |
| Sequence 3 | 227  | City (Dae-jeon)| Day   | Crowded   | Occlusion, Many lanes                                         |
| Sequence 4 | 479  | University (KAIST)| Day   | Uncrowded | Curve, Many lanes                                              |

Table 4: Details of the sequences in K-Lane
| Datum Type            | Extension | Format                        | Comment                                                                 |
|----------------------|-----------|-------------------------------|-------------------------------------------------------------------------|
| Point cloud          | .pcd      | Point cloud with 131072 points| Input to point projector and Heuristic Lidar Lane Detection            |
| BEV point cloud tensor | .pickle   | \( N_g \times N_c \times N_p \) size numpy array | Input to pillar encoder                                                 |
| BEV label            | .pickle   | \( H_b ev \times (W_b ev + 6) \) size numpy array | Lane label including unlabeled lane per row (6 Columns are for the possible row-wise detection-based approaches) |
| Front image          | .png      | 1920 \( \times \) 1200 RGB image | For annotation and visualization                                         |
| Calibration parameters | .txt      | Intrinsic and extrinsic parameters | For Lidar-camera projection                                             |
| Description          | .txt      | Descriptions (e.g., night and day) |                                                                          |

Table 5: Details of the Dataset Files

\[
FPs = \sum_{i}^{N} \sum_{j}^{M} \left(1 - \max\{\hat{x}_{conf_{i,j}}x_{conf_{i+p,j+q}}\}\right)
\]

\[
p \in \{-1, 0, 1\}, q \in \{-1, 0, 1\} \tag{4}
\]

For a positive label at grid \((i, j)\), the false negatives (FNs) are computed as follows,

\[
FNs = \sum_{i}^{N} \sum_{j}^{M} \left(1 - \max\{\hat{x}_{conf_{i+p,j+q}}x_{conf_{i,j}}\}\right)
\]

\[
p \in \{-1, 0, 1\}, q \in \{-1, 0, 1\} \tag{5}
\]

For classification predictions, instead of thresholding, we assign lane instance (i.e., lane 1, lane 2,...) to each grid based on the predicted class probabilities. We define the classification map label \(\hat{x}_{cls} = \{x_{cls_{i,j}} | x_{cls_{i,j}} \in 1,..., C, i \in I, j \in J\}\), and the raw classification map prediction \(\hat{x}_{cls}^{raw} = \{\hat{x}_{cls_{i,j,k}} | i \in I, j \in J, k \in K\}\), where \(x_{cls_{i,j}}\) is the lane class label at grid \((i, j)\), \(\hat{x}_{cls_{i,j,k}}\) is the raw classification map prediction at grid \((i, j)\) for lane class \(k\), \(N\) is the number of rows, \(M\) is the number of columns, and \(C\) is the number of lane class. A classification map prediction \(x_{cls} = \{x_{cls_{i,j}} | i \in I, j \in J\}\), can be constructed by assigning the class with the highest probability on the raw classification map to the classification map

\[
\hat{x}_{cls_{i,j}} = \arg \max_k \{x_{cls_{i,j,k}} | k \in K\} \tag{6}
\]

where \(K = \{1,..., C\}\). For each lane class \(k \in K\), we create additional binary maps \(x_{cls}^k\) and \(\hat{x}_{cls}^k\) which indicate the grid where the lane class exists on the classification map label and the classification map prediction, respectively,

\[
x_{cls_{i,j}}^k = \begin{cases} 
1 & \text{if } x_{cls_{i,j}} = k \\
0 & \text{otherwise}
\end{cases}, \quad (7)
\]

\[
\hat{x}_{cls_{i,j}}^k = \begin{cases} 
1 & \text{if } \hat{x}_{cls_{i,j}} = k \\
0 & \text{otherwise}
\end{cases}. \quad (8)
\]

Each pair of binary maps is evaluated based on the previously described evaluation criteria for calculating TP, FP, and FN. The overall classification performance can be calculated by accumulating the TPs, FPs, and FNs from each pair of binary maps evaluation and using equation (1) to obtain the F1-score.

**Appendix D. Detection Head and Loss Functions**

Fig. x shows the detection head, first introduced in Section 3.2.3. There are two segmentation heads namely the classification head and the confidence head. Given an input of \(H_{bev} \times W_{bev} \times C\) feature map from the GFE output, we employ two consecutive shared-MLPs to create the final prediction maps output. The first shared-MLP then transforms the feature maps from \(2C\) to \(N_{cls}\) for both classification and confidence heads to increase the representation capacity. The second shared-MLP then transforms the feature maps from \(2C\) to \(N_{cls}\) and from \(2C\) to 1 for the classification head and confidence head, respectively, resulting in a classification map and confidence map predictions. We then apply a grid-wise softmax to the classification map to get the \(H_{bev} \times W_{bev} \times N_{cls}\) classification map output, and a grid-wise sigmoid to the confidence map to get the \(H_{bev} \times W_{bev} \times 1\) confidence map output. The classification map shows per-class-probabilities of each grid, while the confidence map only shows the probability of a grid being a lane or not. The implementation of both classification and confidence tasks in parallel enables the LMN to simultaneously predict the lane shape and the lane class. As stated...
in Section 3.2.3, we use the soft dice loss (Sudre et al. 2017) for supervising the confidence loss $L_{\text{conf}}$, defined as

$$L_{\text{conf}} = 1 - \frac{2 \sum_{i}^{N} \sum_{j}^{M} x_{\text{conf}}_{i,j} \hat{x}_{\text{conf}}_{i,j}}{\sum_{i}^{N} \sum_{j}^{M} x_{\text{conf}}_{i,j}^2 + \sum_{i}^{N} \sum_{j}^{M} \hat{x}_{\text{conf}}_{i,j}^2 + \epsilon}$$  \hspace{1cm} (9)$$

where $x_{\text{conf}} = \{x_{\text{conf}}_{i,j} | x_{\text{conf}}_{i,j} \in 0,1, i \in I, j \in J\}$ is the confidence map label, $\hat{x}_{\text{conf}} = \{\hat{x}_{\text{conf}}_{i,j} | x_{\text{conf}}_{i,j} \in 0,1, i \in I, j \in J\}$ is the confidence map prediction, $I = \{1, ..., N\}$, $J = \{1, ..., M\}$, $N$ is the number of rows, $M$ is the number of columns, and $\epsilon$ is set to be $10^{-12}$ to prevent division by zero. The grid-wise cross-entropy loss (Ronneberger, Fischer, and Brox 2015) is used as the classification loss $L_{\text{cls}}$, defined as

$$L_{\text{cls}} = \frac{1}{NM} \sum_{i}^{N} \sum_{j}^{M} \log(p(\hat{x}_{\text{raw}}^{\text{cls}}_{i,j}))$$  \hspace{1cm} (10)$$

where $x_{\text{cls}} = \{x_{\text{cls}}_{i,j} | x_{\text{cls}}_{i,j} \in 1, ..., C\}$ is the classification map label, $\hat{x}_{\text{raw}}^{\text{cls}} = \{\hat{x}_{\text{raw}}^{\text{cls}}_{i,j,k} | i \in I, j \in J, k \in K\}$ is the raw classification map prediction, $I = \{1, ..., N\}$, $J = \{1, ..., M\}$, $K = \{1, ..., C\}$, $N$ is the number of rows, $M$ is the number of columns, $C$ is the number of classes, and $p(\hat{x}_{\text{raw}}^{\text{cls}}_{i,j,k})$ is the softmax of the classification prediction for class $k = x_{\text{cls}}_{i,j}$ at grid $(i,j)$, defined as

$$p(\hat{x}_{\text{raw}}^{\text{cls}}_{i,j,k}) = \frac{\exp(\hat{x}_{\text{raw}}_{i,j,k})}{\sum_{k'}^{C} \exp(\hat{x}_{\text{raw}}_{i,j,k'})}$$  \hspace{1cm} (11)$$
Appendix E. Ablation Study

Ablations on network depth

We perform several ablation studies on the MixSegNet and Point Projector models. As shown in Table 6, increasing the depth of MixSegNet from 1 to 3 increases the performance by +0.52 and +5.58 for Projector41 and PointPillars projector, respectively, but decreases the performance by -0.05 for Projector28. Furthermore, performance degradations are observed when the MixSegNet depth is increased from 3 to 5 for Point Projector and from 5 to 7 for Point Pillars. From our ablation studies, we find that the model with the appropriate model capacity is Projector41-MixSegNet3. Note that for models with larger capacities, some regularizations or more sophisticated learning techniques may be applied to reduce overfitting. However, those learning techniques are out of the scope of our study, since we focus on the novel network architecture and dataset.

Ablation on hidden dimension

We perform ablation studies on different hidden dimension size for Projector41-MixSegNet3 and Pillars-MixSegNet5, both of which are the best performing model of the base Projector-MixSegNet and Pillars-MixSegNet models, respectively. As denoted in Section 3.2.2, the hidden dimension \( N_h \) is the number of channels for each patch after going through the patch-wise linear transform, indicating that higher value of hidden dimension leads to higher model capacity per each grid. As shown in Table 7, regarding regarding the Projector-MixSegNet, the hidden dimension of \( N_h=512 \) for Projector41-MixSegNet3 outperforms other variants, such as \( N_h=128 \) and \( N_h=2048 \). On the other hand, since Pillars-MixSegNet requires less model capacity per each grid than Projector-MixSegNet, Pillars-MixSegNet with \( N_h=128 \) outperforms the model with \( N_h=512 \).

Ablation on patch size

We also perform ablation studies on the patch size of the Projector41-MixSegNet3 and Pillars-MixSegNet5. We increase the patch size, \( P \), from the 8 to 16, and the results in Table 7 shows that there is a significant drop in performance. When the patch size is increased to 16 (from 8), the number of grids that are covered by the patch increases four times, which means that the GFE has to extract global features from a map with four times lower resolution.

Additional discussion of classification metric

Table 8 is a summary of F1-score considering the classification (i.e., F1-score on classification) performance for various models, based on Projector-MixSegNet, Pillars-MixSegNet, Projector-ResNetFPN, and Pillars-ResNetFPN, which have been discussed or compared with in this paper. The F1-score on classification is an evaluation of networks based on the accuracy of both the lane line localization and the lane class. Therefore, F1-score on classification is a more strict evaluation metric and performance degradation (from the F1-score on confidence) can be found for all models. However, we observe two interesting differences depending on the choice of GFE. First, there is a noticeable degradation of MixSegNet performance with F1-score on classification, when the patch-size is increased or hidden dimension size is reduced. This is because classification task requires larger model capacity than confidence tasks. For example, Table 8 shows that Projector41-MixSegNet3, Projector41-MixSegNet3/P16, and Projector41-MixSegNet3/N128 lose 0.32%, 4.04%, and 9.1%, respectively. Second, since MixSegNet considers the whole area in the input feature map in the extraction of global features but ResNetFPN considers local areas, MixSegNet achieves a much superior performance as shown with the F1-score on classification. In other words, MixSegNet is more capable of finding the lane class relative to the ego-lane. As shown in Table 8, Projector41-MixSegNet3 with \( N_h=512 \) and \( P=8 \) has 0.22%~0.47% degradation for F1-score on classification.
| Model                  | Total | Daylight | Night  | Crowded | Uncrowded | Curve  | Occlusion | Size (MB) | Speed (ms) |
|-----------------------|-------|----------|--------|---------|-----------|--------|-----------|-----------|------------|
| Projector41-MixSegNet1| 89.66 | 90.19    | 91.80  | 85.67   | 92.52     | 87.08  | 83.48     | 49.1      | 69.85      |
| Projector28-MixSegNet1| 91.13 | 91.53    | 93.32  | 87.11   | 94.01     | 89.66  | 83.86     | 70.7      | 82.60      |
| Projector41-MixSegNet1| 91.15 | 91.56    | 93.26  | 87.40   | 93.84     | 88.78  | 84.70     | 72.5      | 93.73      |
| Projector14-MixSegNet3| 91.08 | 91.51    | 93.51  | 87.04   | 93.98     | 89.68  | 83.99     | 94.2      | 83.19      |
| Projector41-MixSegNet3| 91.67 | 92.15    | 93.47  | 88.34   | 94.07     | 89.85  | 85.29     | 96.0      | 94.24      |
| Projector41-MixSegNet5| 90.79 | 91.04    | 93.24  | 86.82   | 93.63     | 89.43  | 83.52     | 96.2      | 71.41      |
| Projector28-MixSegNet5| 91.05 | 91.37    | 93.43  | 87.03   | 93.94     | 89.82  | 83.35     | 117.8     | 84.08      |
| Projector41-MixSegNet5| 90.70 | 90.97    | 93.04  | 86.86   | 93.46     | 89.39  | 82.26     | 119.6     | 96.65      |

Table 6: F1-score (on confidence) of LMNs with Various Model Capacities (i.e., Depth)

| Model                  | Total | Daylight | Night  | Crowded | Uncrowded | Curve  | Occlusion | Size (MB) | Speed (ms) |
|-----------------------|-------|----------|--------|---------|-----------|--------|-----------|-----------|------------|
| Pillars-MixSegNet1    | 84.42 | 85.18    | 86.56  | 79.22   | 88.14     | 77.80  | 78.78     | 37.8      | 69.26      |
| Pillars-MixSegNet3    | 89.70 | 90.17    | 91.32  | 85.88   | 92.24     | 86.04  | 85.40     | 61.4      | 70.07      |
| Pillars-MixSegNet5    | 89.78 | 90.41    | 91.68  | 86.07   | 92.45     | 86.70  | 85.32     | 84.9      | 71.00      |
| Pillars-MixSegNet7    | 89.71 | 90.38    | 91.34  | 86.07   | 92.32     | 86.46  | 85.27     | 108.5     | 71.76      |

Table 7: F1-score (on confidence) of LMNs with Larger Patch Size & Various Hidden Dimension

(from F1-score on confidence), whereas Projector41-ResNetFPN shows 7.73%~13.24% degradation. In fact, this difference in performance degradation becomes severe when networks uses Pillar encoders such that MixSegNet shows 0.09%~1.64% degradation and ResNetFPN shows 11.87%~33.18% degradation.
| BEV Encoder | GFE | Model | Total | Daylight | Night | Crowded | Uncrowded | Curve | Occlusion |
|-------------|-----|-------|-------|----------|-------|---------|-----------|-------|-----------|
| MixSegNet   |     | Proj.41-MixSegNet1 | 90.68 | 91.18 | 92.73 | 86.93 | 93.36 | 87.28 | 84.39 |
|             | MixSegNet | Proj.41-MixSegNet3 | **91.35** | **91.83** | **93.04** | **88.12** | **93.66** | **89.34** | **85.03** |
|             |      | Proj.41-MixSegNet5 | 90.48 | 90.85 | 92.67 | 86.75 | 93.16 | 89.03 | 82.39 |
| MixSegNet (ablations) |     | Proj.41-MixSegNet3/P16 | 83.43 | 84.58 | 86.13 | 77.35 | 87.79 | 72.64 | 76.80 |
|             | MixSegNet | Proj.41-MixSegNet3/Nh128 | 88.65 | 89.03 | 91.39 | 84.15 | 91.88 | 85.05 | 80.50 |
|             |      | Proj.41-MixSegNet3/Nh2048 | 90.87 | 91.20 | 93.18 | 87.00 | 93.64 | 88.41 | 84.37 |
| ResNetFPN |     | Proj.41-ResNet8FPN | 71.85 | 70.96 | 79.14 | 57.84 | 81.90 | 64.96 | 50.25 |
|             | MixSegNet | Proj.41-ResNet18FPN | 74.42 | 72.78 | 80.35 | 63.76 | 82.07 | 65.15 | 56.69 |
|             |      | Proj.41-ResNet23FPN | 77.67 | 77.91 | 83.17 | 67.40 | 85.03 | 69.16 | 63.50 |
| Pillar Encoder |     | Pillars-MixSegNet1 | 82.78 | 83.61 | 85.20 | 77.04 | 86.92 | 72.29 | 77.60 |
|             | MixSegNet | Pillars-MixSegNet3 | 89.61 | 90.17 | 91.32 | 85.33 | 91.60 | 84.60 | 84.97 |
|             |      | Pillars-MixSegNet5 | 89.41 | 90.02 | 91.26 | 85.79 | 92.01 | 86.13 | 84.83 |
|             | MixSegNet (ablations) | Pillars-MixSegNet7 | 89.18 | 89.89 | 90.82 | 85.51 | 91.81 | 85.42 | 84.93 |
|             |      | Pillars-MixSegNet5/P16 | 74.47 | 75.23 | 77.22 | 67.03 | 79.77 | 55.80 | 70.46 |
|             | MixSegNet | Pillars-MixSegNet5/Nh128 | 81.90 | 83.05 | 84.60 | 76.12 | 86.06 | 75.52 | 76.81 |
|             |      | Pillars-MixSegNet5/Nh2048 | 88.86 | 89.48 | 90.84 | 84.92 | 91.69 | 84.15 | 84.27 |
| ResNetFPN |     | Pillars-ResNet8FPN | 31.23 | 31.19 | 34.10 | 19.95 | 39.34 | 17.32 | 13.60 |
|             | MixSegNet | Pillars-ResNet18FPN | 60.27 | 60.18 | 66.61 | 47.22 | 69.08 | 36.65 | 48.25 |
|             |      | Pillars-ResNet23FPN | 62.35 | 62.43 | 67.25 | 50.17 | 71.09 | 44.41 | 46.83 |

Table 8: F1-score on classification of variations of LMN, Projector-ResNetFPN, Pillars-ResNetFPN. Models without the indication of hidden dimension and patch sizes use Nh=512 and P=8. 'Proj.' is the abbreviation of 'Projector'