SafeDrive: Enhancing Road Safety with an Advanced Detection System

Duy Nguyen - duyan2 Aniketh Aangiras - aniketh3 Dahui Song - dahuis2
Ronald Roy - rroy21

1. Introduction

SafeDrive aims to address the critical need for enhanced road safety through an advanced detection system. This system is designed to identify key elements such as lanes, traffic signs, and pedestrians, which are essential for reducing traffic accidents and improving driving conditions. The motivation for this project stems from the increasing demand for sophisticated safety features in modern automobiles and the potential for applications in other areas such as road construction and maintenance machinery.

Existing literature indicates that while individual detection systems for pedestrians [9], lanes [8], and traffic signs [11] exist, their integration into a unified system presents a significant challenge. Our approach to SafeDrive involves the development of a modular, ML-based framework that allows for the individual training of each feature (lanes, traffic signs, pedestrians) and their integration into a comprehensive safety detection system. This modular design facilitates the potential inclusion of new ML-trained features to further provide a well-rounded system for ensuring road safety. In addition, we aim at improving the robustness of our framework under various extreme weather and lighting conditions, as we notice the available traffic datasets [4, 17] contain only a small number of these kinds of data, hence potentially yielding inconsistent and unreliable results for these scenarios. We tackle these issue by adopting a training strategy that can make our system capable of performing zero-shot prediction under these scenarios.

The preliminary results highlight the potential of our ML models in individual tasks. In addition, our incorporation of the weather-invariance training pipeline with the lane detection task suggests the great potential of our current strategy when dealing with extreme environmental conditions. By building on existing studies and addressing the identified gaps, SafeDrive is positioned to significantly contribute to the evolution of road safety technologies.

2. Details of the approach

Figure 1 illustrates the overall workflow of our SafeDrive system, which comprises three distinct modules: Lane Detection, Pedestrian Detection, and Traffic Sign Detection.

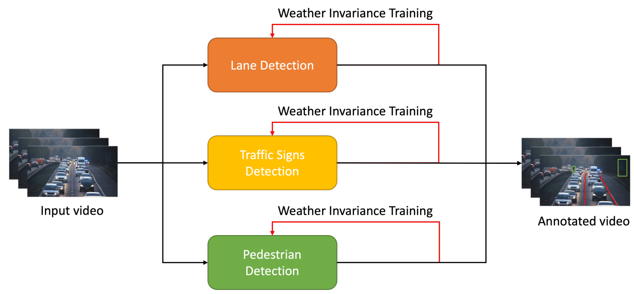

Figure 1. Overall pipeline of SafeDrive system.

Each module is tasked with specific functions that collectively enhance driving safety.

During the training phase, each module is trained and evaluated independently. In addition to the standard training strategy that focuses on task-specific objectives, we have implemented a unique training step called *Extreme Invariance Training*. This step aims to enhance the robustness of each module against variations in environmental conditions such as lighting changes or weather changes. Detailed descriptions of each module can be found in the respective subsections: Lane Detection (2.1), Traffic Sign Detection (2.2) and Pedestrian Detection (2.3). The methodology and benefits of the Extreme Invariance Training are elaborately discussed in Section 2.4. Following the independent training, the modules are combined using a late fusion approach for deployment.

In the deployment phase, the system processes input video data in parallel across all modules. The results from each module are then aggregated and presented simultaneously, ensuring a comprehensive and real-time response to driving conditions.

2.1. Lane Detection

Our primary objective in the lane detection task is to identify lane markings in 2D space (image space) by pinpointing the 2-D coordinates of various lane lines. We are committed to detecting all visible lanes located between the left and right curbsides.

Baseline. With this task, we utilize the CondLSTR [5], a transformer-based architecture that has demonstrated supe-

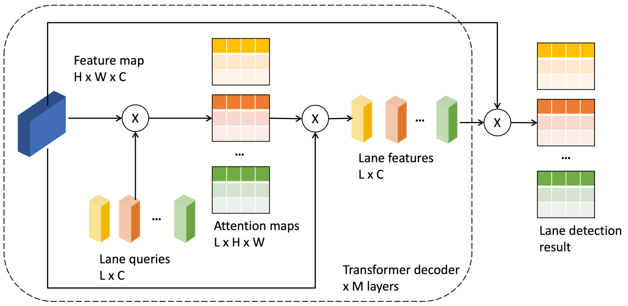

Figure 2. Simplified Architecture of CondLSTR. This figure is redrawn from original paper.

rior performance in benchmarks such as OpenLane [4] and CurveLane [17]. Figure 2 illustrates the decoder architecture of this framework, which consists of a series of Transformer blocks designed to accomplish two primary tasks: (1) generating dynamic convolutional kernels for each lane line from the extracted feature map, and (2) applying these dynamic kernels to the feature map to detect final lane lines. The process begins with a CNN backbone that extracts a feature map from the input image. Subsequently, within the Transformer block, the dynamic kernel head employs a sequence of learnable *lane queries* representing different prototypes of lane lines, together with cross-attention mechanism to generate dynamic convolutional kernels tailored for each lane line from the feature map. This architecture produces two sets of dynamic kernels, one generating a heat map and the other an offset map for each lane line. The lane detection head then detects lane lines by convolving these dynamic kernels with the feature map. Additionally, the framework predicts a vertical range and an object score for each lane line, enhancing the accuracy of the lane detection. The detected lane points are extracted from the heat maps, offset maps, vertical ranges, and object scores through a series of post-processing steps. To refine the model's accuracy, a bipartite matching loss is computed between the predicted lane lines and ground-truth data. The transformer-based generation of kernels captures the global information of lane lines across the entire feature map, enabling the model to handle occlusions and complex lane line topologies more effectively compared to traditional methods that generate kernels from specific key locations.

Modification. Although CondLSTR currently achieves state-of-the-art results on several benchmark datasets, we acknowledge that the training costs associated with operating this entire Transformer-based pipeline are substantial and highly dataset-specific. Drawing inspiration from extensive research that employs frozen image encoders for various downstream tasks [10, 14], we propose a hypothesis: by using a robust and generic frozen image encoder, we may only need to fine-tune the Decoder module of CondLSTR for different datasets. This hypothesis is based on the complexity and high representational capability of the architecture. Motivated by this hypothesis, we replaced the trainable ResNet-34 backbone [7] in CondLSTR with the frozen image encoder from CLIP [15]. To accommodate the CLIP encoder, we adjusted the dimensions in the subsequent decoding steps accordingly. This substitution significantly reduces the number of trainable parameters when adapting to different datasets. Furthermore, this modification plays a pivotal role in enhancing the efficiency of the Extreme Invariance Training process, which is detailed further in Section 2.4.

2.2. Traffic Sign Detection

Baseline. We selected YOLOv8 [1] as the backbone for our traffic detection task due to its leading-edge capabilities in real-time object detection and image segmentation. This choice was driven by several key benefits:

- **Rapid and Accurate Detection.** YOLOv8 delivers fast and precise object detection, essential for real-time applications, which perfectly suits the need of SafeDrive system.
- **Enhanced Traffic Management.** The model's efficiency in recognizing various traffic signs contributes significantly to improving road safety through better traffic management systems.

To accelerate convergence, we adopt the pretrained YOLOv8 on COCO dataset [13] for detection task. Owing to constraints on computational resource, we work with the nano variance of this model - `YOLOv8n`.

2.3. Pedestrian Detection

Baseline. Initially, we aimed to develop a Traffic Sign Detection feature using YOLOv8 trained on the Penn-Fudan dataset [16]. However, due to the limited size of this dataset, we were unable to effectively train the model, prompting us to utilize a pre-trained YOLOv8 model from Ultralytics [1], the same choice we adopt for the Traffic Detection task. We initially chose the Penn-Fudan dataset for its urban environment pedestrian images, which are scarce in larger datasets. Despite attempts to expand the dataset through data augmentation, its size ultimately required the adoption of a pre-trained model.

Modification. Our pedestrian detection framework, however, is robust enough to handle various types of pedestrian movements, such as walking or biking. It is designed to process video inputs effectively, as demonstrated in our demo, and can be updated to capture live vehicle footage with minor code adjustments. The feature is optimized to process frames significantly faster than 30 frames per second, with our tests showing it can handle 640 by 384 frames at 43 frames per second in the worst-case scenario. Therefore, it can easily manage live feeds at 30fps or even at higher resolutions under the current configuration.

2.4. Extreme Invariance Training

Baseline. To enhance performance under extreme weather and lighting conditions, we initially considered augmenting raw image frames and fine-tuning the SafeDrive modules using this data, following the methodology outlined in [12]. However, we encountered practical limitations due to constraints in storage and computational resources. As an alternative, we adopted a technique from a recent study [6] that modifies models to adapt to new domains using text prompts about the target domain, thus eliminating the need for actual target images. This innovative method utilizes CLIP encoders to align the image features from the source domain with the text embeddings of the target domain. It preserves the original content and semantics through Prompt-driven Instance Normalization (PIN). The transformations are stored in a memory-resident dictionary, which facilitates the efficient fine-tuning of the classifier without incurring significant memory or time costs.

Modification. We have incorporated the methodology described in [6] into our training pipeline, which we term *Extreme Invariance Training*, for each targeted task within the SafeDrive system. Given the limited time constraints of our group project, we implemented this strategy specifically for the Lane Detection task. In this context, the source domain involves lane detection under normal conditions, while the target task focuses on lane detection in extreme environmental scenarios. The Extreme Invariance Training procedure is structured into three distinct steps:

- **Source-only training:** Train SafeDrive's modules on the desired tasks using source datasets.
- **Zero-shot feature augmentation:** Utilize the PIN module [6] to learn the style parameters necessary for transforming normal image features into features that reflect varied weather or lighting conditions.
- **Classifier fine-tuning:** Retrain the modules' classifiers by integrating the newly styled features into the original image features of the training datasets. This results in a framework with zero-shot capabilities, enabling robust performance under extreme weather or lighting conditions.

It is important to note that Extreme Invariance Training involves the use of a set of text templates representing the target domains, which are essential for adapting source tasks to extreme weather settings. We have developed these template sets specifically tailored for the datasets of interest.

3. Experiment Settings and Results

3.1. Lane Detection

Dataset. We utilize a streamlined version of OpenLane [4] for training and evaluating our 2D Lane Detection task. The training dataset comprises a total of 240 video sequences, with 60 sequences reserved for testing and evaluation. Additionally, this dataset includes metadata annotations about each video's scene information. We leverage the metadata fields `weather` and `hours` to selectively identify a subset of sequences that exhibit extreme weather conditions, such as rain, fog, or overcast skies, as well as challenging lighting conditions, such as nighttime or dawn/dusk. This subset is then used to evaluate the weather invariance characteristics of our trained framework. Figure 3 illustrates frames captured from two videos corresponding to these two testing sets.

Normal Testing Dataset   Extreme Case Dataset

Figure 3. Our two OpenLane testing datasets.

Metrics. For quantitative analysis, we employ the same evaluation metrics as those detailed in [5], namely F1 Score, Precision, and Recall. These metrics are calculated by comparing the lanes detected by our system with the ground-truth lane annotations. The determination of whether a lane detection is classified as a true positive (TP) or a false negative (FN) relies on the intersection-over-union (IoU) metric. Specifically, the IoU for lane lines is assessed based on the overlap between the masks of the predicted and ground-truth lanes, which maintain a uniform line width of 30 pixels, the alignments of at least 20 pixels are considered TPs, as established in [5].

Settings. In this experiment, we evaluate three variants of the CondLSTR framework: (1) the original CondLSTR equipped with a ResNet-34 backbone; (2) CondLSTR augmented with a frozen CLIP encoder; and (3) CondLSTR enhanced with both a frozen CLIP encoder and Extreme Invariance Training. Variants (1) and (2) are each trained over 20 epochs without the use of a validation set. Variant (3) undergoes an additional fine-tuning phase where the detector from Variant (2) is fine-tuned using style-injected image features for 5 extra epochs. The metrics reported for the testing dataset reflect the outcomes from the models at the completion of their respective final epochs.

Result. Normal testing set. Table 1 presents the evalua-

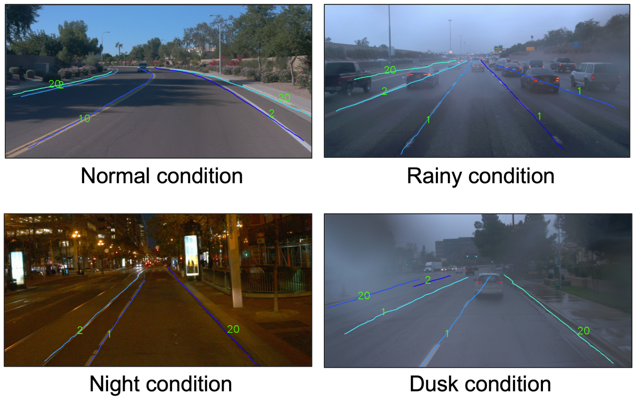

Figure 4. Prediction made in different environmental conditions.

tion results of all model variants on the standard OpenLane testing dataset, which consists of 60 segments. Replacing the ResNet-34 backbone with the CLIP image encoder not only reduces the number of trainable parameters but also enhances performance, supporting our initial hypothesis for this modification. Furthermore, the implementation of Extreme Invariance Training significantly improves performance, indicating that this strategy is effective in enhancing model robustness under extreme conditions while preserving its efficacy in normal scenarios.

Table 1. Lane Detection evaluation on normal testing set.

Model	Avg. F1	Avg. Precision	Avg. Recall
CondLSTR (1)	0.739	0.793	0.692
CondLSTR (2) (our)	0.766	0.824	0.716
CondLSTR (3) (our)	**0.799**	**0.835**	**0.767**

Extreme testing set. Similarly, we report the performance of the three variants on our specially constructed datasets, which include scenes under extreme conditions. As observed in the normal settings, the same pattern is evident here: both modified variants demonstrate improvements in average performance across all recorded metrics compared to the original model. Figure 4 illustrates some actual predictions made by the third variant of CondLSTR under various environmental conditions.

Table 2. Lane Detection evaluation on extreme testing set.

Model	Avg. F1	Avg. Precision	Avg. Recall
CondLSTR (1)	0.761	0.782	0.742
CondLSTR (2) (our)	0.799	0.835	0.767
CondLSTR (3) (our)	**0.827**	**0.863**	**0.795**

3.2. Traffic Sign Detection

Dataset. Our YOLOv8 model for traffic sign detection is trained and evaluated on the Kaggle's traffic signs detection dataset [2], which comprises 4,969 samples. These samples are distributed across training (71%), validation (16%), and testing (13%) sets. Figure 5 illustrate some images sampled from this dataset.

Figure 5. Kaggle Traffic Sign Detection dataset.

Metrics. The mean Average Precision (mAP) was calculated by averaging the Average Precision (AP) scores across various Intersection over Union (IoU) thresholds. These thresholds range from 0.5 to 0.95, increasing in increments of 0.05. This calculation provides a comprehensive measure of the model's detection precision across a spectrum of overlapping scenarios.

Settings. Training was carried out for 50 epochs, with early stopping implemented to prevent overtraining based on the validation loss. Each image was resized to 416×416 pixels. The model's hyperparameters were determined through experimentation, with a batch size of 64 and a dropout rate of 0.15. Table 3 summarize some statistics recorded during training on different epochs, with different choice of learning rate configurations.

Table 3. Traffic Sign detection training statistics.

Hyperparameters		mAP50	mAP50-95
Epoch	10	0.189	0.132
	30	0.482	0.321
	50	**0.651**	**0.548**
Learning rate (epoch10)	1e-4	0.189	0.132
	3e-4	**0.193**	**0.173**

Results. After the training process, the qualitative results on the Kaggle testing dataset indicated that mAP50 reached a score of 0.651, and mAP50-95 scored 0.549. These outcomes were achieved by the variant that recorded the highest mAP score on the validation dataset during training. Additionally, the PR-curve and F1-confidence curve are visualized in Figure 6 for this model. As illustrated in Figure 6a, the PR AUC for the classes red light, green light, and stop sign is significantly high. However, the speed limit sign class shows lower accuracy, which impacts the overall PR AUC of the model negatively. The PR curve indicates a sharp drop from the top right corner, suggesting that improvements are needed to enhance recall. Furthermore, Figure 6b reflects high confidence in classifying green light, red light, and stop sign, aligning with the PR curve findings. However, classes such as speed limit 100, speed limit 110, and speed limit 80, which show high uncertainty in predictions, have low F1 scores. This highlights the necessity for additional targeted training for these specific classes.

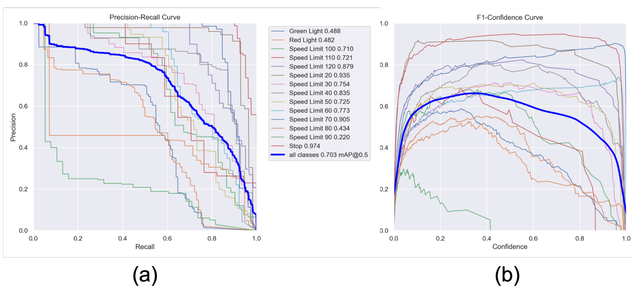

Figure 6. P-R curve and F1-condidence curve of our YOLOv8 model for Traffic Detection.

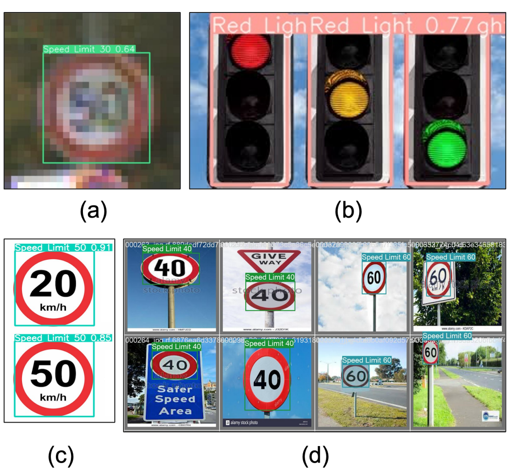

Figure 7. Traffic sign detection in challenging cases.

To understand the model's behavior, it was evaluated on challenging test cases, including images with low resolution, multiple traffic signs, and cluttered backgrounds. We conducted a qualitative evaluation under the following scenarios:

- **Low resolution.** Images with low resolution challenge the model's ability to detect and classify signs accurately under degraded image quality, which is common in real-world scenarios due to factors such as camera quality or image compression. As shown in Figure 7a, the model struggles to distinguish speed limit signs when the image resolution is very low.

- **Multiple signs in a single image.** This test case assesses the model's capability to detect and classify multiple signs within a single image, which is crucial because real-world environments often contain multiple signs in close proximity. The model generally distinguishes these signs (Figure 7b), but can sometimes misclassify similar-looking numbers (e.g., '20' and '50' on speed limit signs), as shown in Figure 7c.

- **Cluttered background.** Images with complex or cluttered backgrounds present challenges in isolating and recognizing traffic signs. This test case evaluates the model's robustness in real-world driving environments where signs may be surrounded by various distractions. Despite these challenges (Figure 7d), the model is expected to perform well in detecting signs accurately.

3.3. Pedestrian Detection

We include the result of pretrained YOLOv8 model on COCO dataset [13] for `person` class in Figure 8 and Table 4.

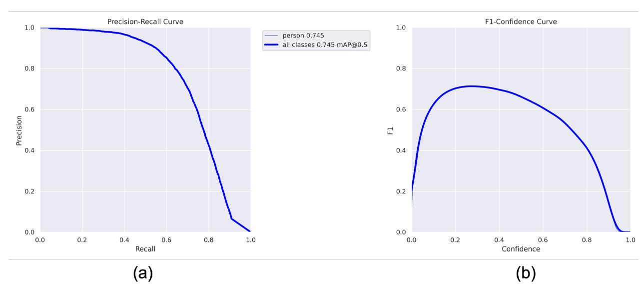

Figure 8. Person detection with YOLOv8 on COCO.

Table 4. Person detection with YOLOv8 on COCO.

Class	Box Precision	Box Recall	Box mAP50	Box mAP50-05
Pedestrian	0.755	0.671	0.745	0.514

Since a pre-trained model, YOLOv8, was used to handle the detections for this feature, we wanted to compare how accurate the detects are for this feature compared to other models. To qualitatively verify this, we first find public demos of pedestrian detection. Given the same input videos/photos, we can see how well our pedestrian detection feature can do given the same inputs. Doing that we can visually see how our feature compares to other models.

Figure 9 presents key observations from our experiment. It highlights that the detections marked in blue demonstrate a tighter region, indicating more precise detections compared to the comparison model, which marks its detections

Figure 9. Qualitative assessment of YOLOv8 compared with other models.

in green/yelow using a HOG+SVG approach [3]. This suggests that our model provides enhanced accuracy in identifying targets, compared to this existing pipelines.

3.4. Joint evaluation

We establish a joint pipeline to aggregate the results from all modules using input videos, producing the final outputs for SafeDrive. We have prepared demo videos along with their corresponding outputs, which are available on a shared Google Drive accessible via this link. The selection of demo videos includes one recorded under normal environmental conditions and another under extreme conditions, to demonstrate the system's effectiveness across different scenarios. Our presentation that briefly go through each module, can be found at this link.

4. Conclusion and Discussion

4.1. Lane Detection

We upgraded the CondLSTR framework, a leading solution for 2D Lane Detection, by replacing its ResNet backbone with CLIP's frozen image encoder, significantly reducing trainable parameters and slightly improving results. Integrating Extreme Invariance Training also enhanced the framework's resilience to extreme environmental conditions. However, due to time and computational constraints, our experiments were limited in number and scope, potentially affecting the completeness of model convergence and our findings' robustness. Future work will focus on validating our methods across more datasets for a thorough evaluation.

4.2. Traffic sign detection

The qualitative evaluation of our traffic sign detection model revealed key issues: it struggles with accurate detection and classification in low-resolution images and distinguishing similar numbers on speed limit signs in complex or cluttered backgrounds. These challenges may stem from hardware limitations and the limited training data scope. Despite using data augmentation, the dataset size and quality remain constraints. The model required over a day to train for 50 epochs. Possible improvements include acquiring more diverse data, increasing training duration, and using techniques like histogram of oriented gradients to better differentiate similar features. While promising, the model requires further optimization for real-world efficacy.

4.3. Pedestrian Detection

The development of our pedestrian detection feature revealed several challenges. Inconsistencies in detections were noted, particularly under harsh or low lighting conditions, varying weather, and differing skin tones in the training data. These factors are critical, especially in vehicular applications where accurate detection is essential to avoid potentially fatal outcomes in self-driving models. Addressing these challenges requires utilizing large datasets encompassing a broad range of training scenarios. Another issue was the significant overhead development required to integrate large, multiple datasets, which limited our available development time. Ultimately, we employed a smaller dataset that yielded poorer results compared to the pre-trained YOLOv8 model.

5. Work done by Individuals

- **Duy Nguyen - duyan2 - contribution weight: 0.28** Conducted a literature review, adapted and implemented enhancements atop two existing code repositories, tuned parameters and ran experiments, and prepared the report and presentation for the Lane Detection and Extreme Environmental Invariance tasks. Additionally, implemented the joint evaluation pipeline for all modules.

- **Dahui Song - dahuis2 - contribution weight: 0.28** Implemented traffic sign detection, trained the YOLOv8 model on a manageable dataset, and experimented with various training parameters. Conducted comprehensive quantitative and qualitative evaluations, including diverse test cases, to assess the model's performance.

- **Aniketh Aangiras - aniketh3 - contribution weight: 0.22** Focused on developing the Pedestrian Detection feature, beginning with dataset selection and literature review. Transitioned to using a pre-trained YOLOv8 model due to limitations of the Penn-Fudan dataset in training an effective model.

- **Ronald Roy - rroy21 - contribution weight: 0.22** Initially aimed to create a rear-end accident detection feature. Faced challenges due to the unavailability of suitable datasets due to privacy laws and minimal high deceleration event data. Shifted focus to enhance the Pedestrian Detection feature, achieving significant improvements in accuracy and processing speed, and integrated the feature into the common framework with robust and adaptable code.

References

[1] GitHub - ultralytics/ultralytics: NEW - YOLOv8 in PyTorch - ONNX - OpenVINO - CoreML - TFLite — github.com. https://github.com/ultralytics/ultralytics. [Accessed 05-05-2024]. 2

[2] Traffic Signs Detection — kaggle.com. https://www.kaggle.com/datasets/pkdarabi/cardetection. [Accessed 04-05-2024]. 4

[3] Muhammad Bilal and Muhammad Shehzad Hanif. Benchmark revision for hog-svm pedestrian detector through reinvigorated training and evaluation methodologies. *IEEE transactions on intelligent transportation systems*, 21(3):1277–1287, 2019. 6

[4] Li Chen, Chonghao Sima, Yang Li, Zehan Zheng, Jiajie Xu, Xiangwei Geng, Hongyang Li, Conghui He, Jianping Shi, Yu Qiao, et al. Persformer: 3d lane detection via perspective transformer and the openlane benchmark. In *European Conference on Computer Vision*, pages 550–567. Springer, 2022. 1, 2, 3

[5] Ziye Chen, Yu Liu, Mingming Gong, Bo Du, Guoqi Qian, and Kate Smith-Miles. Generating dynamic kernels via transformers for lane detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6835–6844, 2023. 1, 3

[6] Mohammad Fahes, Tuan-Hung Vu, Andrei Bursuc, Patrick Pérez, and Raoul De Charette. Poda: Prompt-driven zero-shot domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18623–18633, 2023. 3

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2

[8] Hiroto Honda and Yusuke Uchida. Clrernet: improving confidence of lane detection with laneiou. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1176–1185, 2024. 1

[9] Abdul Hannan Khan, Mohammed Shariq Nawaz, and Andreas Dengel. Localized semantic feature mixers for efficient pedestrian detection in autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5476–5485, 2023. 1

[10] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR, 2023. 2

[11] Yi Li, Jinguo Li, and Ping Meng. Attention-yolov4: a real-time and high-accurate traffic sign detection algorithm. *Multimedia tools and applications*, 82(5):7567–7582, 2023. 1

[12] Zhe Li, Yi Jin, Yidong Li, Zhiping Lin, and Shan Wang. Imbalanced adversarial learning for weather image generation and classification. In *2018 14th IEEE International Conference on Signal Processing (ICSP)*, pages 1093–1097. IEEE, 2018. 3

[13] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014. 2, 5

[14] Ziyi Lin, Shijie Geng, Renrui Zhang, Peng Gao, Gerard De Melo, Xiaogang Wang, Jifeng Dai, Yu Qiao, and Hongsheng Li. Frozen clip models are efficient video learners. In *European Conference on Computer Vision*, pages 388–404. Springer, 2022. 2

[15] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 2

[16] Liming Wang, Jianbo Shi, Gang Song, and I-fan Shen. Object detection combining recognition and segmentation. In *Asian conference on computer vision*, pages 189–199. Springer, 2007. 2

[17] Hang Xu, Shaoju Wang, Xinyue Cai, Wei Zhang, Xiaodan Liang, and Zhenguo Li. 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*, pages 689–704. Springer, 2020. 1, 2

SafeDrive: Enhancing Road Safety with an Advanced Detection System
Supplementary Material

A. Lane Detection

A.1. Experiment Settings

While most hyper-parameters are inherited from original work of [5], we list our some important change made in our codebase regarding experimental settings in Table 10. Figure 10 is the screen capture recording our training statistics with Tensorboard. Due to the mis-configuration in setting individual run names, we can only visualize the aggregated training statistics of all runs.

Table 5. Key change in hyper-parameters setting for CondLSTR.

	CondLSTR (1)	CondLSTR (2)	CondLSTR (3)
Input size	940x480	768x768	768x768
Encoder's output dim	256	2048	2048
Train epochs	20	20	5
Train module	Encoder + Detector	Detector	Detector - on weight of (2)

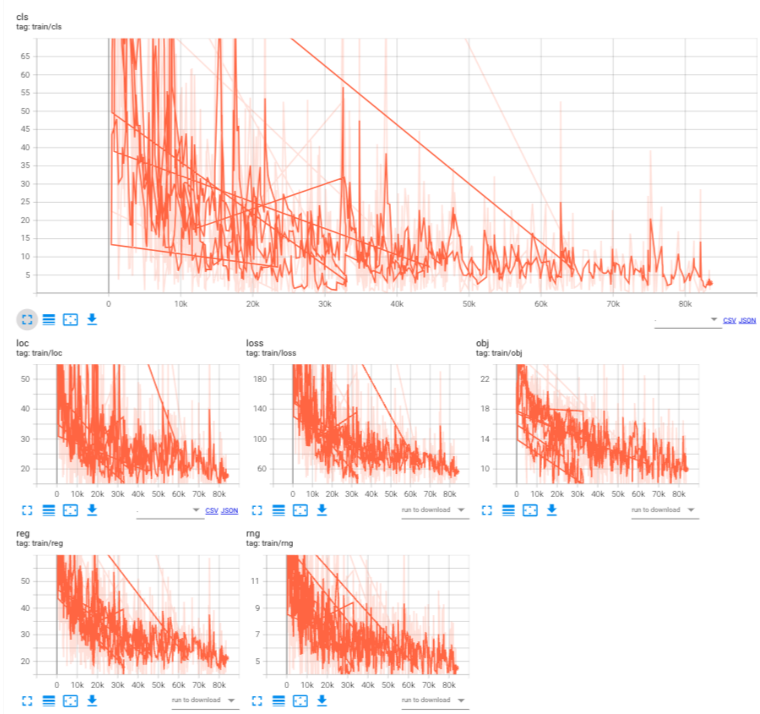

Figure 10. Training statistics of CondLSTR variances.