vehicles

MDPI

Article

Lane Detection Based on CycleGAN and Feature Fusion in Challenging Scenes

Eric Hsueh-Chan Lu * and Wei-Chih Chiu

Department of Geomatics, National Cheng Kung University, No.1, University Rd., Tainan 701, Taiwan; cyc256489@gmail.com
* Correspondence: luhc@mail.ncku.edu.tw

Abstract: Lane detection is a pivotal technology of the intelligent driving system. By identifying the position and shape of the lane, the vehicle can stay in the correct lane and avoid accidents. Image-based deep learning is currently the most advanced method for lane detection. Models using this method already have a very good recognition ability in general daytime scenes, and can almost achieve real-time detection. However, these models often fail to accurately identify lanes in challenging scenarios such as night, dazzle, or shadows. Furthermore, the lack of diversity in the training data restricts the capacity of the models to handle different environments. This paper proposes a novel method to train CycleGAN with existing daytime and nighttime datasets. This method can extract features of different styles and multi-scales, thereby increasing the richness of model input. We use CycleGAN as a domain adaptation model combined with an image segmentation model to boost the model's performance in different styles of scenes. The proposed consistent loss function is employed to mitigate performance disparities of the model in different scenarios. Experimental results indicate that our method enhances the detection performance of original lane detection models in challenging scenarios. This research helps improve the dependability and robustness of intelligent driving systems, ultimately making roads safer and enhancing the driving experience.

Keywords: lane detection; domain adaptation; GAN; image segmentation

check for updates

Academic Editors: Hocine Imine and Claudio Lantieri

Received: 4 December 2024
Revised: 25 December 2024
Accepted: 30 December 2024
Published: 1 January 2025

Citation: Lu, E.H.-C.; Chiu, W.-C. Lane Detection Based on CycleGAN and Feature Fusion in Challenging Scenes. *Vehicles* **2025**, *7*, 2. https://doi.org/10.3390/vehicles7010002

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

According to the 2022 report of Markets & Markets [1], an international research and survey organization, the market value of self-driving artificial intelligence may reach USD 7 billion in 2027. With the advancement of artificial intelligence in computer vision and other advances in vehicle automatic control technologies, such as in the research of [2], the challenges of automatic lane changing and collision avoidance of autonomous vehicles can be effectively solved. Autonomous driving is no longer technology that is out of reach. The research in [3] also shows a high acceptance of autonomous driving in recent years, and the possibility that autonomous vehicles will be gradually introduced into our markets. Improving driving safety and efficiency, reducing accidents, and enhancing the driving experience have been long-term goals in the development of intelligent driving systems such as self-driving vehicles and advanced driver assistance systems (ADASs). Autonomous driving technology includes many key assistance systems, including the lane centering assist system (LKAS) and lane departure warning system (LDWS). The objective of LKAS is to assist the vehicle in staying in the center of the lane. The system uses lane detection technology to identify lane lines and monitor the vehicle's position. When the

vehicle deviates from the center of the lane, it achieves stability and comfort through slight automatic adjustments, helping to reduce fatigue driving and the risk of accidents. LDWS also monitors the vehicle's position and lane lines through lane detection technology. When the vehicle deviates from the lane, the system will sound, vibrate, or visually alert the driver to remind the driver to take action. Such systems can help drivers avoid accidents, especially lane departures, due to distraction or fatigue.

Lane detection is the core technology of these systems. Lane detection uses sensors and cameras to capture lane markings and features on the road and uses image processing technology to identify the position and shape of the lane. This provides critical information for the intelligent driving system to ensure that the vehicle stays in the correct lane and avoids drifting or colliding with other vehicles. In recent years, significant advancements have been made in lane detection methods due to research in image-based deep learning. The deep learning model can learn how to extract the features of the lane based on the dataset to achieve high-accuracy lane detection. This approach addresses the challenge of complex and dynamic road environments that traditional manual feature extraction methods struggle to handle, thereby enhancing system accuracy and robustness. The image-based deep learning lane detection technology enables self-driving technologies such as LKAS and LDWS to be realized, thereby allowing the driving experience to become safer and more comfortable.

In recent years, lane detection methods based on deep learning models have achieved outstanding performance, enabling nearly real-time detection. Although these models perform well in general daytime scenarios, their performance degrades significantly at night or in other challenging scenarios. Low-light conditions and insufficient lighting at night, as well as other factors such as strong glare, shadows, rain, or snow, can impact the performance of the lane detection system. At night, images suffer from insufficient external lighting, noise, motion blur, and optical blur, and lane markings also have reduced contrast, making them harder to identify. In other challenging scenarios, such as shadowed or wet roads, the clarity and visibility of lane lines can also suffer, making detection difficult.

Another important challenge is insufficient diversity within the training dataset. Deep learning models used for lane detection rely on extensive training data to update model parameters and enhance the model's ability to recognize lane features. However, when the training data lack enough samples of challenging scenes, the performance of lane detection models will be constrained in these scenarios. Collecting and labeling data for scenes that include various weather conditions or otherwise is costly in terms of labor and time. In recent years, domain adaptation models based on generative adversarial networks (GANs) have seen significant advancements in addressing data distribution disparities between different domains. Image style transfer is one important application which involves the technique of transferring the style of one image to another. SIM-CycleGAN [4] uses a style transfer method to convert daytime images into nighttime images to augment the training dataset for lanes. However, such a data augmentation method has limited impact on enhancing model performance, and cannot reduce the difference in the detection ability of the model in different scenarios. We believe that while doing style conversion, we can simultaneously consider extracting multi-scale features and constraining the output differences of different scenes. This inspired us to use the domain adaptation model to conduct image style conversion and combine feature extraction methods to address the issue of limited training data diversity.

To address these challenges, CycleGAN [5] is a promising solution that can transform images from one domain to another without relying on labeled data. CycleGAN is a generative adversarial network (GAN) that has numerous applications in domain adaptation tasks, such as converting ordinary photos into artwork or game footage into

real-world environments. Unlike other style transfer methods, CycleGAN provides a more flexible solution for real-world applications since it does not require strictly paired images. This is an indispensable advantage in our method, since it is almost impossible to collect paired daytime and nighttime images in road scenes. Our proposed method will use the existing daytime and nighttime images of the lane dataset to train CycleGAN, then use the trained CycleGAN to extract different styles and multi-scale features and combine image segmentation models to enhance the model's detection capability across different styles of scenes. The experiments demonstrate that our method enhances the detection performance of the original lane detection model across various scenarios.

The contributions of this paper are listed below:

1. The proposed method combines a domain adaptation model and image segmentation model to improve the model's detection performance in challenging scenes.
2. The proposed method uses the domain adaptation model to extract different styles and multi-scale feature training models to improve the robustness of the model.
3. The proposed consistent loss function constrains the final output of feature maps of different styles in the model, which will reduce the performance difference of the model in different scenarios.
4. The model is assessed through experimental research to understand the impact of various methods and training strategies on the lane detection model.
5. In the public lane test dataset, our method can enhance the F1 score of the original model by 1% to 3.3%.

The following is the structure of this paper. First, we review the research on domain adaptation, lane detection, and domain adaptation combined with image recognition in Section 2. Section 3 explains our proposed method and model architecture. Then, Section 4 presents the experimental results and analysis of our proposed method. Following that, we present the conclusion and future work of the paper in Section 5.

2. Related Work

In this section, the literature review explores two main areas: domain adaptation and lane detection. In the domain adaptation section, we discuss the use of GAN to transfer information between different image domains. The lane detection section focuses on deep learning methods for detecting lane lines in road scenes. Lastly, the combination of domain adaptation and image recognition is discussed for enhancing performance in challenging road scene recognition tasks.

2.1. Domain Adaption

In recent years, GAN has become a popular and effective method for learning and generating realistic data distributions. In computer vision, image domain adaptation is a challenging task that seeks to transfer information learned from an original domain to a target domain where the labeled data are scarce or unavailable. Traditional approaches often rely on handcrafted features or explicit alignment methods, which may not generalize well to diverse and complex image distributions. However, with the emergence of GAN, novel techniques have been introduced to tackle image domain adaptation by learning mappings between different image domains. CoGAN [6] is a notable generative model that extends the concept of traditional GAN to learn a joint distribution of multiple domains simultaneously. It uses the same noisy input to generate two styles of images. It ensures that the overall appearance of the two outputs is similar by sharing the shallow weights of the generator. pix2pix [7] proposes a method based on conditional GAN to realize image-to-image conversion and uses UNet [8] as the network architecture of the generator. This method is limited by the need for pairs of images to train the model.

BicycleGAN [3] mainly consists of two different GANs. The encoder maps real images to their corresponding latent vectors, while the decoder reconstructs images from the latent vectors. Unlike most one-to-one transformation models, their method can produce diverse results. UNIT [9] presents a novel approach that combines GAN and variational autoencoders to achieve high-quality image translation without the need for paired examples. To achieve this, UNIT employs two main components: a shared-latent space and domain-based encoders and decoders. The shared-latent space serves as a bottleneck layer, where both the content and style representations are encoded. Images from one domain are mapped to the shared-latent space and vice versa by domain-based encoders and decoders. CycleGAN [5] has significantly advanced the field of image domain adaptation by enabling unpaired image-to-image translation. A cycle consistency loss is introduced, requiring that translated images from the target domain be mapped back to their originating domain while maintaining their content. A novel viewpoint for image domain adaptation is provided by DualGAN [10], which expands upon the idea of unpaired image-to-image translation. It introduces a reconstruction loss, the concept of which is the same as the CycleGAN cycle consistency loss. This loss enforces the translated images to remain close to the original images in both the original and target domains, enhancing the preservation of content and style during the adaptation process. DiscoGAN [11] uses the concept of learning bidirectional mappings between domains to provide a novel approach for unpaired image-to-image translation. The DiscoGAN method shares similarities with CycleGAN. It focuses primarily on small-sized images in its experimental analysis. Additionally, DiscoGAN introduces a distinct model architecture that sets it apart from CycleGAN.

MUNIT [12] proposes a polymorphic unsupervised image translation framework, assuming the image representations can be divided into content feature vectors and style feature vectors. By separating these two components, MUNIT enables independent manipulation of content and style, allowing for the generation of diverse translations without the need for paired training data. DRIT [13] uses two encoders to respectively output the context space that retains the image content and a domain-based attribute space. As long as the attribute space of different images is used as input, images of different domains can be generated. INIT [14] is based on the architecture of MUNIT and divides the domain transformation into global style and instance style to guide the model to generate more complete objects. Although paired images are not required, images with bounding boxes are required for training. CIGAN [15] focuses on unsupervised low-light enhancement methods, and it also uses low-light images and normal-light images to train the model. It adds noise in the generator to simulate the noise of low-light images and synthesizes more realistic low-light images.

2.2. Lane Detection

Lane detection is essential to the visual navigation system of intelligent vehicles, and it has always been one of the popular areas of research in autonomous driving. In recent years, lane detection methods based on deep learning models have achieved remarkable performance and can almost achieve real-time detection. Wang et al. propose a deep learning lane line detection method called LaneNet [16], which demonstrates high stability in both highway and urban roads without relying on the number of lanes. Liang et al. propose a convolutional neural network (CNN) architecture called LineNet [17] with lane prediction layers and scaling modules. SCNN [18] improves the traditional CNN layers into sequential convolutional layers to facilitate the information transfer between pixels in rows and columns, thereby enhancing lane feature extraction. EL-GAN [19] is the first method to use a GAN for lane detection. This network simultaneously uses the predicted

lane results and label-trained discriminator to propose an embedding loss to stabilize the entire network's training and achieve better lane prediction results.

ENet-SAD [20] introduces a new self-attention distillation method that enables the model to learn from feature output maps at different scales, capturing rich contextual information. This lightweight model achieves advanced lane detection performance and decent detection speed. ConvLSTM [21] proposes a model that combines CNN and recurrent neural networks and introduces a new method for lane detection using consecutive images. Compared to single-image-based methods, using multiple images can effectively detect lanes in complex scenes with shadows or occlusions. LaneATT [22] proposes an anchor-based lane detection model. Lane distribution often follows specific patterns, and anchor detection based on global information can generate more accurate lane positions even when lane markings are occluded or absent. Ripple-GAN [23] utilizes a Wasserstein generative adversarial network and semantic segmentation to learn lane line features. By classifying targets into the background, lane surface, and lane lines, it exhibits better detection capability for complex or ambiguous lane lines. LaneAF [24] predicts binary segmentation and generates corresponding lane instances by processing horizontal and vertical clustering row by row. This method demonstrates efficient detection capability without requiring a predefined maximum number of lanes.

PINet [25] uses a multi-loss function trained hourglass model. The clustering problem of predicting critical points is regarded as an instance segmentation problem, and the model size can be chosen based on the target's computational capacity. The model can be trained regardless of the number of lane lines. BezierLaneNet [26] introduces a novel method for defining lane lines based on curve parameters. Utilizing parameterized Bezier curves solves the optimization difficulties encountered in traditional polynomial curve representation of lane lines during model training. Additionally, it possesses advantages such as ease of computation and high stability. The model can maintain high accuracy while operating with low latency. CLRNet [27] focuses on exploiting high-level semantic features and low-level texture features in lane detection. By utilizing more contextual information, it can obtain local lane features to improve localization accuracy. SCNN_Unet_LSTM [28] combines the lane feature extraction method of SCNN and the method of ConvLSTM that uses multiple images for lane prediction to enhance detection capability in challenging driving scenarios.

2.3. The Combination of Domain Adaption and Image

For image recognition of road scenes, including vehicles, street signs, lane lines, etc., collecting training data is hard work that costs a lot of time and money. Most of the existing road datasets lack challenging scenes, making deep learning models unable to perform stably when faced with these challenging scenes. Using domain adaptation to improve the issue of insufficient training data has also become a line of potential research. The CyCADA model [29] converts images to the target domain, inputs the translated image into a semantic segmentation model to calculate the loss of semantic segmentation, and also calculates the semantic consistency loss with the original image. This model successfully achieves good performance in both road environment scenes and digit classification.

AugGAN [30] also uses the domain adaptation method to generate nighttime images to expand the dataset of detected vehicles. It uses an encoder paired with two decoders as a model architecture. One of the decoders outputs the synthesized image, and the other outputs the image segmentation map. This constrains the model to maintain the integrity of the object when generating synthetic images. SIM-CycleGAN [4] proposes a data enhancement method based on style conversion, which converts daytime images into nighttime images through CycleGAN, thereby increasing the proportion of nighttime

images. M-AugGAN [31] is based on the method of multimodal GAN. In a model architecture similar to AugGAN, random vectors are input to achieve various output results, to generate diverse and realistic nighttime images to assist model training. DUNIT [32] first uses the detection subnetwork to detect instance objects and then processes the image and object in parallel. After fusing the two, it uses the decoder to generate the full image. Generated images can strike a balance between keeping objects and translating images.

FTEGanOd [33] uses the CycleGAN feature fusion enhancement module to cascade the object detection module to improve vehicle detection performance in complex lighting environments. It builds an end-to-end network combining an image domain adaptation network and object detection network, and uses a feature fusion enhancement module to combine nighttime and daytime features to enhance nighttime vehicle detection. This combination is used to preserve detected objects during image translation. AugGAN-Det [34] adds the object detection model to the GAN model for joint training. The generated image is guided to maintain the integrity of the object as much as possible so that the generated image is more conducive to expanding the dataset of object detection. Most approaches treat the image as a whole and directly input it into the model, which makes it difficult to accurately preserve the objects in the image or to keep them "real".

3. Methodology

In order to better extract the lane features of challenging scenes, the domain adaptation model was employed to extract various styles of image features, which were then utilized as input for the image segmentation model in lane detection. The framework of our proposed method, which is primarily separated into two stages, is depicted in Figure 1. The first stage involved the training of the domain adaptation model, and the second stage involved the training of the lane detection model. Using daytime and nighttime images, the CycleGAN model was trained in the first stage of domain adaption model training, and the trained CycleGAN Generator parameters were saved. The domain adaption module, the feature fusion module, and the image segmentation module were combined in the second stage of the lane detection model for lane detection training. In Section 3.1, we introduce CycleGAN and describe how it is used to train a domain adaptation model that transforms nighttime and daytime images. In Section 3.2, we will introduce the domain adaptation module, feature fusion module, and image segmentation module in the lane detection model respectively. We also detail our proposed consistent loss function and the entire model training process.

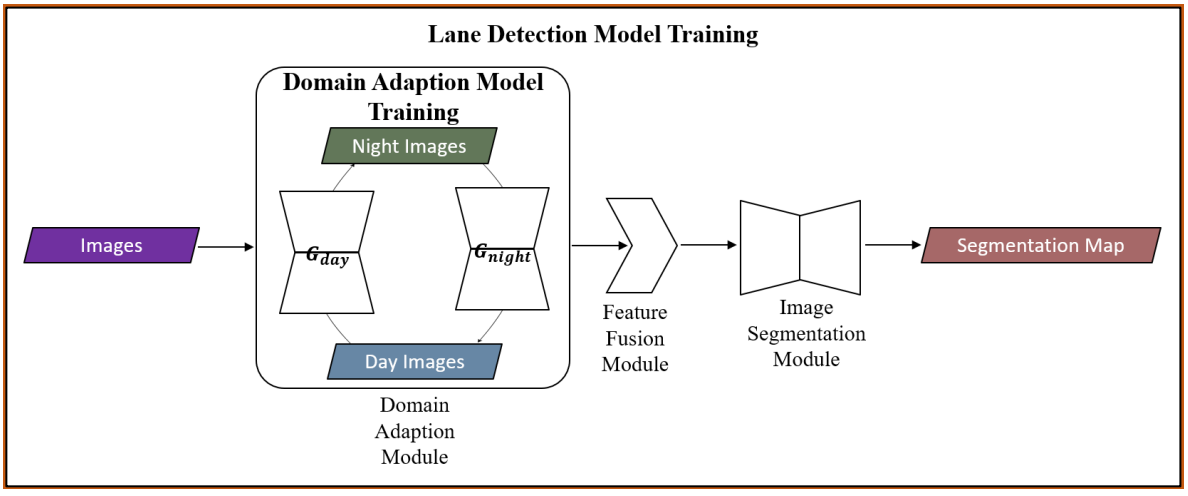


Figure 1. Illustration of the two-stage proposed method.

3.1. Domain Adaption Model Training

For training lane detection, the majority of current lane detection models simply input images into deep learning models, so they are limited by the richness and diversity of the training dataset. Most of the image data in existing lane datasets are daytime scenes. In general daytime and sunny scenes, deep learning models often have excellent recognition performance. However, in the face of relatively rare scenes (such as night, rain, fog, and other scenes), due to the limitation of the dataset, the model cannot have the same recognition performance in these scenarios. The lane detection model we propose includes a domain adaption module, a feature fusion module, and an image segmentation module. The image will be input into the domain adaptation module first, and the feature fusion module will output the feature maps representing the encoder and decoder of the domain adaption module, which will then enter the image segmentation module together for training and testing. Since our domain adaptation model is a model that converts daytime and nighttime images into each other, the feature maps output by the encoder and decoder can just represent the characteristics of daytime or nighttime. That is to say, our lane detection model can use one image to train feature maps representing daytime and nighttime at the same time. Therefore, our method can enhance the lane detection ability of the model at night and in other challenging scenes with different lighting conditions.

Based on the above motivation, we needed to first train a domain adaptation model that could convert nighttime and daytime images to each other. We used CycleGAN as our domain adaptation model. The CycleGAN model is an unsupervised image translation method. Its objective is to learn to map between two domains without paired training data. Not requiring paired data is an important advantage of using CycleGAN, since it is almost impossible to collect paired daytime and nighttime road scenes in our approach. However, paired labels are not required to use CycleGAN; only nighttime and daytime images are required. CycleGAN differs from other image translation models by its ability to perform image translation in both directions, called cycle consistency. The model is capable of converting images from domain X to domain Y while maintaining the properties of the original image. The model can preserve the consistency of image content in different domains by using two GANs and a cycle consistency loss to achieve this characteristic.

3.1.1. CycleGAN

During the training of CycleGAN, we used images from two different image domains for training, such as nighttime and daytime. In our study, we considered nighttime images as domain X and daytime images as domain Y. Our goal was to train two generator models that would convert domain X images to domain Y images and domain Y images to domain X images, respectively. The generator of CycleGAN mainly used auto-encoder architecture. Generator A converted domain X images into domain Y images, and generator B converted domain Y images into domain X images. Furthermore, CycleGAN would combine two discriminator models to evaluate the authenticity of generated images. Discriminator A was used to judge whether the input image was a real image from domain X or a fake image generated by generator B, while discriminator B was used to judge whether the input image was a real image from domain Y or a fake image generated by generator A. The ground truth label of the real image was 1, and the output of the discriminator for a real image needed to be as close to 1 as possible. The ground truth label of the fake image was 0, and the output of the discriminator for the fake image needed to be as close to 0 as possible. The generator needed to generate fake images that made the output of the discriminator as close to 1 as possible.

The generator and the discriminator update the parameters of the model for each other through adversarial training. The generator is driven by the adversarial loss function

to produce realistic images, preventing the discriminator from being able to differentiate generated images from actual ones. CycleGAN uses a cycle consistency loss to maintain the structure of the image. The cycle consistency loss makes sure consistency in the bidirectional transformation process. After the image enters a generator, it will generate a fake image of the other domain, and then the fake image will enter another generator to output a reconstructed image. The cycle consistency loss is used to determine the similarity of the reconstructed image to the original image. To drive the training process, CycleGAN trains the model using the two loss functions mentioned above. The generator and discriminator models will advance one another through ongoing iterative training, making the generated image closer to the target domain while maintaining the characteristics and structure of the original image.

3.1.2. CycleGAN for Nighttime and Daytime

The lighting conditions and ambiance vary greatly between daytime and nighttime. In our strategy, we wanted to train a CycleGAN model that could convert daytime to nighttime and vice versa. About 10,000 daytime and nighttime images from the CULane lane dataset were used to train the CycleGAN model. The model to be trained contained two generators (G_{night} and G_{day}) and two discriminators (D_{night} and D_{day}). The G_{night} generator transformed nighttime images into daytime and the G_{day} generator transformed daytime images into the nighttime. D_{night} and D_{day} discriminators were used to identify real and fake images in daytime and nighttime domains, respectively. Equation (1) represents the adversarial loss from the nighttime domain (X) to the daytime domain (Y):

$$L_{GAN}\left(G_{night}, D_{day}, X, Y\right) = E_{y\sim p_{data}(y)}\left[\log D_{day}(y)\right] + E_{x\sim p_{data}(x)}\left[\log\left(1 - D_{day}\left(G_{night}(x)\right)\right)\right] \quad (1)$$

where D_{day} tries to distinguish if the image is a real daytime image or a synthetic daytime image, and G_{night} attempts to generate a synthetic image of daytime that closely resembles the original image. Similarly, another adversarial loss is needed to focus on the translation from the daytime domain (Y) to the nighttime domain (X), namely $L_{GAN}\left(G_{day}, D_{night}, Y, X\right)$. In CycleGAN, an adversarial loss is used to help the generated image better match the distribution of the target domain. Additionally, a cycle consistency loss was introduced to ensure that the image transformation was reversible; this allowed for the use of the generated image to reconstruct the original image. The idea was to guarantee the integrity and consistency of the image during the conversion process. We define the cycle consistency loss as Equation (2):

$$L_{cyc}\left(G_{night}, G_{day}\right) = E_{x\sim p_{data}(x)}\left[\left\|G_{day}\left(G_{night}(x)\right) - x\right\|_1\right] + E_{y\sim p_{data}(y)}\left[\left\|G_{night}\left(G_{day}(y)\right) - y\right\|_1\right] \quad (2)$$

where G_{night} tries to convert nighttime images into day images, and G_{day} tries to convert daytime images into nighttime images. Our overall objective function is shown in Equation (3).

$$L_{GAN}\left(G_{night}, G_{day}, D_{night}, D_{day}\right) = L_{GAN}\left(G_{night}, D_{day}, X, Y\right) + L_{GAN}\left(G_{day}, D_{night}, Y, X\right) + L_{cyc}\left(G_{night}, G_{day}\right) \quad (3)$$

This function contains two adversarial losses and one cycle consistency loss, and was used to train two generators and two discriminators.

3.2. Lane Detection Model Training

Based on the method of the first stage, we trained a domain adaptation model based on CycleGAN to convert nighttime and daytime images. The overall architecture of our

proposed lane detection model is shown in Figure 2, which consists of a domain adaptation module, feature fusion module, and image segmentation module. Among them, the domain adaption module was used to input nighttime or daytime images and output the features representing nighttime and daytime. The image segmentation module received the fused upper and lower features as input after the feature fusion module extracted multi-scale features. The image segmentation module was based on the existing advanced image segmentation model and outputted the final lane prediction.

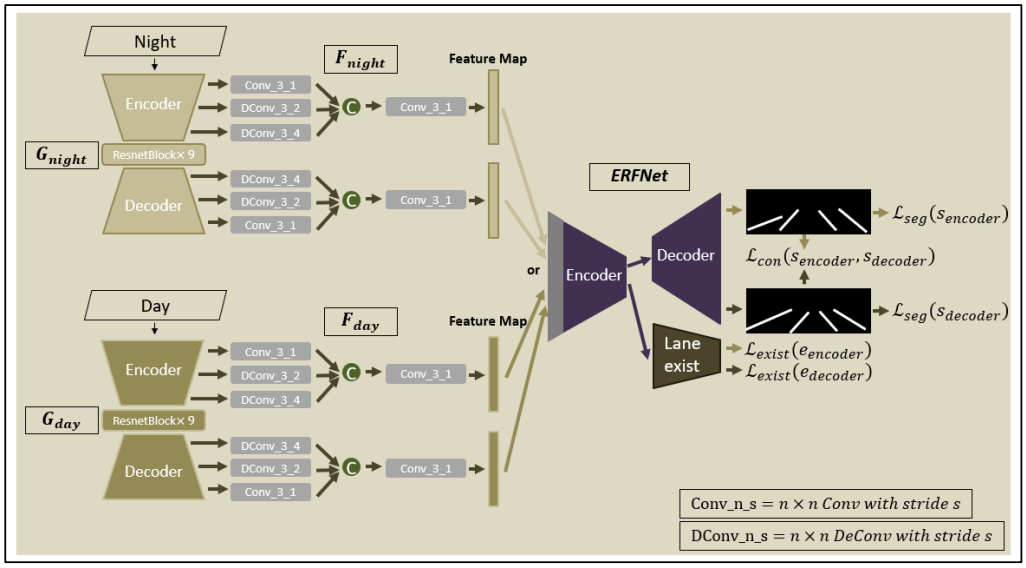


Figure 2. The architecture of the proposed lane detection model training.

3.2.1. Domain Adaption Module

In our proposed method, the domain adaptation module could be used to generate multi-scale feature maps representing nighttime and daytime so that our model could be trained to feature maps of various scales and multiple scenes, which would help enhance the accuracy of the model and robustness. We used the generators of already trained domain adaptation models as our domain adaptation modules, namely G_{night} and G_{day} in Figure 3. We would then extract feature maps of three scales each from the encoder and decoder of the domain adaptation module. In the domain adaption module, whether it is an encoder or decoder, we regarded a convolutional layer, an instance normalization layer, and a ReLU activation function as a generator block. Let B^i be a general function representing a generator block, and the generator block will output the feature map of the next layer. Suppose the dimension of the i-th layer feature map is $[w_i \times h_i \times c_i]$, w is the width of the feature map, h is the height of the feature map, and c is the number of channels output by the generator block. As Equation (4) shows, $B^i_{encoder}$ is the generator block representing the encoder in the generator. The feature map f_i generated by the encoder is shown in Equation (4):

$$\begin{gathered} f_i = B^i_{encoder}(f_{i-1}) \\ \text{Where, } f_i = \mathbb{R}^{w_i \times h_i \times c_i} \\ \text{and } w_i = W/2^{i-1};\ h_i = H/2^{i-1};\ c_i = 2^{5+i};\ i = 1,2,3 \end{gathered} \tag{4}$$

where f_0 is the image input into the model, and the size is W × H. W and H are the width and height of the input image, respectively. As shown in Figure 3, in the CycleGAN generator, there is a module called Resnetblock between the Encoder and the Decoder. There are 9 residual blocks in a Resnetblock, and the input and output dimensions of each block are the same. In our definition, the input of Resnetblock is f_3. We define the

entire Resnetblock as B_{resnet}, and the feature map f_i generated by Resnetblock is shown in Equation (5):

$$\begin{gathered} f_i = B_{resnet}(f_{i-1}) \\ \text{Where, } f_i = \mathbb{R}^{w_i \times h_i \times c_i} \\ \text{and } w_i = w_{i-1};\ h_i = h_{i-1};\ c_i = c_{i-1};\ i = 4 \end{gathered} \tag{5}$$

the feature map output by Resnetblock is f_4, which has the same dimension size as f_3, and will be used as the input of the CycleGAN decoder. $B^i_{decoder}$ is the generator block representing the decoder in the generator. The feature map f_i generated by the decoder is shown in Equation (6).

$$\begin{gathered} f_i = B^i_{decoder}(f_{i-1}) \\ \text{Where, } f_i = \mathbb{R}^{w_i \times h_i \times c_i} \\ \text{and } w_i = W/2^{6-i};\ h_i = H/2^{6-i};\ c_i = 2^{12-i};\ i = 5, 6 \end{gathered} \tag{6}$$

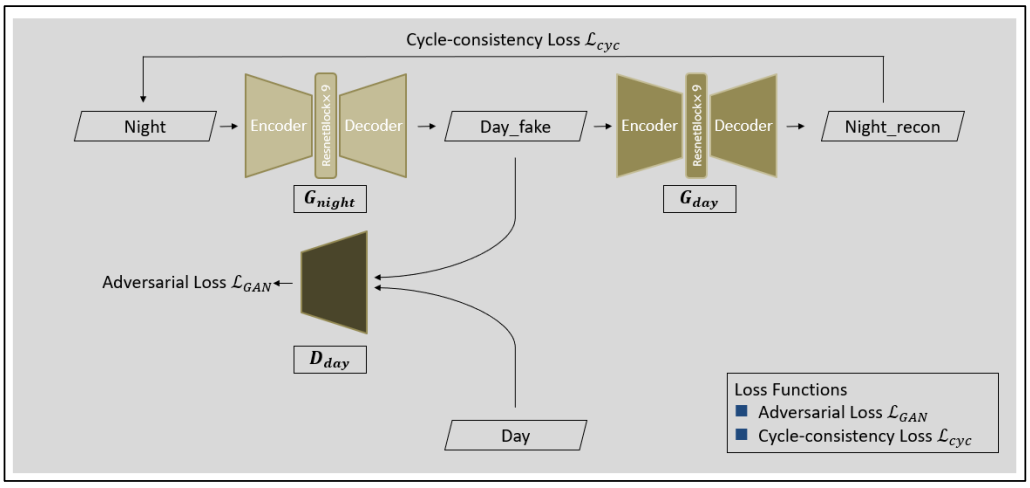


Figure 3. The architecture of the proposed domain adaption model training.

3.2.2. Data Augmentation

We extracted feature maps of various scales from the encoder and decoder of the domain adaptation module. As shown in F_{night} and F_{day} in Figure 3, the primary purpose of the feature fusion module is to connect the domain adaptation module and the image segmentation module in series. This module fuses multi-scale feature maps and outputs feature maps representing nighttime and daytime. In order to integrate multi-scale features, feature maps of different scales will enter different convolutional blocks, and the feature maps of different scales will be adjusted to the same size. In addition, the number of channels will also be converted to the same size to average the influence of each feature map. After validation in our internal experiments, we chose to output feature maps with the same dimensions as the input image. After the convolution kernel, the three feature maps were connected, and a convolution kernel was used to change the number of channels and output the final feature map to the image segmentation module. Since we used feature maps that included an encoder and decoder, our feature fusion module would eventually output two feature maps to represent different scenes. The following is the operation process of the feature fusion module. The feature map g_i is shown in Equation (7):

$$\begin{aligned} g_i &= B_{CBL_n_s}(f_i) && i = 1,\ 6;\ n = 3;\ s = 1 \\ g_i &= B_{DCBL_n_s}(f_i) && i = 2,\ 5;\ n = 3;\ s = 2 \\ g_i &= B_{DCBL_n_s}(f_i) && i = 3,\ 4;\ n = 3;\ s = 4 \end{aligned} \tag{7}$$

where f_i is the feature map extracted by the domain adaptation model. $B_{CBL_n_s}$ is represented as a convolutional block containing a convolutional layer, batch normalization layer, and a leakyReLU activation function. $B_{DCBL_n_s}$ is represented as a convolutional block containing a convolutional layer, batch normalization layer, and a leakyReLU activation function. The convolutional kernel size is represented by n, and s is the stride. g_i is the feature map after the first convolution layer. The feature map f_{concat_en} and f_{concat_de} is shown in Equation (8):

$$\begin{aligned} f_{concat_en} &= F_{concat}(g_1, g_2, g_3) \\ f_{concat_de} &= F_{concat}(g_4, g_5, g_6) \end{aligned} \tag{8}$$

where F_{concat} denotes the concatenate operation. f_{concat_en} and f_{concat_de} are the feature maps connected by the three feature maps output by the encoder and decoder, respectively. The feature map $f_{encoder}$ and $f_{decoder}$ is shown in Equation (9):

$$\begin{aligned} f_{encoder} &= B_{CBL_n_s}(f_{concat_en}) \\ f_{decoder} &= B_{CBL_n_s}(f_{concat_de}) \end{aligned} \qquad n = 3;\ s = 1 \tag{9}$$

where f_{concat_en} and f_{concat_de} need to go through a convolution block, respectively. The $f_{encoder}$ and $f_{decoder}$ are the final output of the feature fusion module and are also the input of the image segmentation module.

3.2.3. Image Segmentation Module

In the previous chapter, we obtained the feature map by combining the encoder and decoder. Next, we used the image segmentation module to input feature maps representing nighttime and daytime to detect the position and shape of the lane. The image segmentation module could choose advanced image segmentation network models such as ERFNet [35], DLA [36], and DDRNet [37]. An image segmentation model is a deep learning technique that aims to segment an input image into different regions or groups of pixels and assign a corresponding label to each region. This technique enables the model to identify and distinguish between different objects, structures, or features in the imagery, enabling precise area localization and classification. Image segmentation models usually use deep learning architectures such as CNN to learn to extract features from a huge amount of labeled data, and then achieve accurate segmentation and classification tasks for new images. In lane detection, the image segmentation model can help us effectively identify different lane lines and backgrounds, thus providing key information support for lane detection methods. We used ERFNet as an example to illustrate the changes in our method and model. ERFNet is a deep learning model for semantic segmentation tasks. The core idea of ERFNet is to use factorized convolutions and residual connections to achieve efficient and accurate models. Factorized convolution splits a large convolution operation into multiple smaller convolution operations to reduce the computational load of the model. The non-bottleneck-1D that was proposed in the encoder section is a module that combines factorized convolution and residual connection and expands the field of view of the convolution layer by adding dilated convolution.

The traditional lane detection method directly uses RGB images as the input of the image segmentation model. In our method, we inputted feature maps with 16 channels into the model. To do this, we changed the number of input channels from 3 to 16 in the first layer of the ERFNet model. In our approach, since we only needed to segment out different lanes and backgrounds, ERFNet was by definition an instance segmentation model. The CULane dataset we used would only have up to 4 lane lines at the same time, so we set the number of output channels by the last layer of the model to 5, representing 4 lane lines and the background, respectively. According to the method of ENet-SAD [20], in

order to accelerate the convergence speed of the network during training and to output the lanes with confidence values higher than the threshold during testing, we added a lane branch network parallel to the decoder in the ERFNet model. The branch network included convolutional layers and fully connected layers, and finally outputted four floating-point numbers from 0 to 1, corresponding to the confidence value of each lane.

The original ERFNet model uses two loss functions as feedback for network training during lane detection network training. The first is the instance segmentation map output by the decoder, which represents the probability of whether each pixel is a lane or background. The confidence of the existence of the corresponding lane is output by the second branch lane existence branch, and the loss function is shown in Equation (10):

$$L_{lane} = \lambda_1 L_{seg}(s, \hat{s}) + \lambda_2 L_{exist}(e, \hat{e}) \tag{10}$$

where L_{seg} is the negative log-likelihood loss function for instance segmentation, and L_{exist} is the lane existence binary cross-entropy loss function. s is the instance segmentation map output by the decoder, $\hat{s}$ is the ground truth of the instance segmentation map. e is the confidence of the existence of the lane (the value is between 0 and 1), and $\hat{e}$ is the ground truth of the lane existence; if the lane exists, it is represented by 1, and if the lane does not exist, it will be represented by 0. We set weights $\lambda_1 = 1$, $\lambda_2 = 0.1$.

3.2.4. Consistent Loss

Our model will eventually output model predictions from the feature maps of the CycleGAN encoder and decoder. Although these two output predictions are the results of the same image input, there will still be differences in the output of the predicted lane pixel probability map. Therefore, our proposed consistent loss function constrains the variance of model outputs. The two domain adaptation modules we trained can convert nighttime images into daytime and daytime images into nighttime, respectively. This means that for the nighttime-to-daytime domain adaption module, the feature map extracted from the encoder is closer to the features of the nighttime, and the feature map extracted from the decoder is closer to the features of the daytime. For the domain adaption module that changes from daytime to nighttime, it will extract feature maps close to daytime and nighttime, respectively. We expect that by restricting the input of the two feature maps to the lane pixel probability map output by the image segmentation module, the image segmentation module can output nearly consistent results under different light conditions, which can improve the performance and robustness of the neural network model. This therefore provides the model with the ability to detect lane markings that are difficult to recognize under different lighting conditions.

To this end, we proposed an algorithm for the consistent loss function, for which we used the common L1 loss function. L1 loss is the mean absolute error (MAE) between each element in the measured input. In our approach, it was used to compute the per-pixel difference in the image. The loss function is shown in Equation (11):

$$L_{con} = L_1(s_{encoder}, s_{decoder}) \tag{11}$$

where $s_{encoder}$ and $s_{decoder}$ represent the instance segmentation map predicted by the feature map extracted by the encoder and decoder, respectively.

3.2.5. Training Process

This section explains how to combine domain adaptation, feature fusion, and image segmentation modules for lane detection training. As shown in Figure 3, we divided the training images into nighttime and daytime images. Nighttime images were input into

the CycleGAN generator that turned nighttime into daytime, namely G_{night}, and daytime images were input into the CycleGAN generator that turned daytime into nighttime, namely G_{day}. The multi-scale feature maps output by G_{night} and G_{day} were then input to the respective feature fusion modules, namely F_{night} and F_{day}. The $f_{encoder}$ and $f_{decoder}$ outputs via F_{night} or F_{day} were then input to the lane detection module at the same time. The lane detection module would eventually output the instance segmentation maps $s_{encoder}$ and $s_{decoder}$ output by the decoder at the same time, as well as the confidence $e_{encoder}$ and $e_{decoder}$ of the lane existence. Equation (12) is the total loss of the lane detection model.

$$L_{total} = \lambda_1 \left(L_{seg}(s_{encoder}, \hat{s}) + L_{seg}(s_{decoder}, \hat{s}) \right) + \lambda_2 (L_{exist}(e_{encoder}, \hat{e}) + L_{exist}(e_{decoder}, \hat{e})) + \lambda_3 L_{con}(s_{encoder}, s_{decoder}) \quad (12)$$

We set weights $\lambda_1 = 1$, $\lambda_2 = 0.1$, $\lambda_3 = 1$.

Next, we will introduce our training process. Since our training data were divided into nighttime and daytime images, during the same epoch, we first trained the nighttime images and then trained the daytime images. The nighttime images and daytime images had their data loaders shuffle their respective datasets. Through our internal experiments, updating the model parameters of G_{night} and G_{day} did not help to improve the performance of the overall model, so when training nighttime images, the optimizer only updated the model parameters of F_{night} and ERFNet. When training daytime images, the optimizer only updated the model parameters of F_{day} and ERFNet.

4. Experimental Evaluations

In this chapter, detailed experimental data and settings are provided, as well as the results and analysis of the conducted experiments. In Section 4.1, we describe the dataset utilized and provide evaluation metrics for analyzing model performance. We also detail the hardware and software setup we used to train and test the models. In Section 4.2, we discuss preliminary experiments performed to explore potential ways to improve our study. We analyzed the results of these experiments and their testing value. In Section 4.3, we present the in-house experiments performed to identify the best of our proposed methods. Furthermore, in Section 4.4, we conduct external experiments on the CULane dataset, focusing on different and challenging scenarios.

4.1. Experimental Data and Settings

The proposed approach using a feature fusion architecture combined with a domain adaption model and an image segmentation model applies to any semantic segmentation-based lane detection model. We conducted experiments on the CULane [15] dataset to evaluate the performance of the proposed method. This dataset is among the most comprehensive lane detection datasets available, comprising 88,880 images for training and 34,680 images for testing. The test images contained several challenging driving scenarios, including nighttime and other challenging lighting conditions. The original size of the images in the CULane dataset was 1640×590. We referred to the experimental settings of the SIM-CycleGAN [4] study, cropping the image by 240 pixels starting at the top and resizing the image to 976×208. When using the DLA model as the image segmentation module, based on the downsampling requirements of the DLA model, the input image size was set to 992×224.

In order to evaluate our method objectively, we followed the literature of SCNN [18] to evaluate the evaluation metrics of the CULane dataset. After our model predicted the instance segmentation map of the lane, we used the procedure of SCNN [18] to convert the instance segmentation map to point coordinates in the same format as the ground truth label. We treated both the predicted and ground truth point coordinates as lane lines with a

width of 30 pixels, and calculated the intersection union (IoU) between the predicted and ground truth labels. True positive (TP) was defined as an IOU greater than 0.5. Misjudged lanes were regarded as false positive (FP), and missed lanes were regarded as false negative (FN). We calculated Precision, Recall, and the F1 score based on the above indicators.

IoU represents the overlap ratio of the predicted lane to the ground truth of the lane. According to SCNN [18], we considered the lane as a plane with a width of 30 pixels to compute the overlap ratio. If the IoU was greater than 0.5, it was regarded as true positive (TP). The higher the IoU, the more accurate the predicted position of the lane. The formula is shown in Equation (13):

$$\text{IoU} = \frac{Area\ of\ Overlap}{Area\ of\ Union} \tag{13}$$

Precision refers to the proportion of the model that correctly predicts positive samples in the predicted positive samples. In other words, Precision quantifies how many of the samples predicted as positive are actually true positives. The formula is shown in Equation (14).

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{14}$$

Recall refers to the proportion of correctly predicted positive samples in the total true positive samples. The formula is shown in Equation (15).

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{15}$$

The F1 score is the harmonic mean of Precision and Recall, considering both Precision and Recall combined metrics. The F1 score balances the trade-off between Precision and Recall. The formula is shown in Equation (16).

$$\text{F1} = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{16}$$

The training and testing process was executed on two NVIDIA 3090 graphic processing units. The CUDA version was 11.2, the Python version was 3.8.13, and the Pytorch version was 1.12.0. For the training of the CycleGAN model, we used approximately 10,000 daytime and nighttime images from the CULane dataset. The Adam optimizer was implemented with an initial learning rate of 0.0002, and training was performed for 200 epochs. It took approximately 3–4 days to complete. To train the lane detection model, we utilized the SGD optimizer with an initial learning rate of 0.01 and performed training for 24 epochs. The training time was about 1–2 days.

4.2. Preliminary Experiments

In our research, we tried various intuitive ways to improve our research, and we found that the results of some of the experiments might not meet the expected performance level. Nevertheless, we believe that these experiments still have important testing value and deserve to be described and discussed in this study.

4.2.1. Data Augmentation Method

Our main method was to train the domain adaptation model first, and then use the trained model to join the lane detection model for training. We intuitively used the trained domain adaptation model to generate synthetic daytime and synthetic nighttime images and used the generated images to join the training of the lane detection model. The experimental results in Table 1 show that the data augmentation method does not improve the recognition ability of the model in our method. In the experiment, we also tried to use

different weights to determine the extent to which the augmented data affected the model, but the performance of the model still could not be improved. We think that the possible reason is that after training with augmented data, the model may not be able to focus on the feature extraction of the original image. Therefore, we decided to only use the initial training dataset for training in subsequent experiments.

Table 1. Comparison of data augmentation methods.

Models	Data Augmentation	Augmentation Data Loss Weight	F1 Score (%)	
			Total	Night
ERFNet			**73.6**	**68.5**
	✔	1	68.4	59.7
	✔	0.5	68.6	57.7
	✔	0.1	69.4	59.0

4.2.2. Domain Adaption Model Variants

In theory, training a better domain adaptation model can lead to better feature extraction in the second stage of lane detection model training, which is helpful for the training of the overall model. Therefore, we tried several methods to improve our domain adaptation model. For the first approach, we referred to CIGAN [15] for adding noise to the domain adaptation model. We added random noise based on the normal distribution to the output of each layer of the decoder in the daytime-to-nighttime generator, and then multiplied it by a learnable weight to make the feature distribution more diverse and synthesize a more realistic nighttime image. In the second method, we tried to replace the CycleGAN generator originally based on the ResNet [27] network architecture with other backbone networks. We believed that using the new backbone network might make the model training of CycleGAN more efficient and achieve better results. Therefore, we tried to transform the backbone network of MobileNetv2 [38] and EfficientNet [39] into the network architecture of the auto-encoder to become the network architecture of the CycleGAN generator.

The experimental findings shown in Table 2 indicate that these methods have not achieved the expected results of the model. We think that the method of adding noise may be reasonable in the method of training the domain adaptation model, but will not help the extraction of features when training the lane detection model. In addition, since the noise is added to the entire image, it may make the lane lines less obvious. For the method of replacing the backbone network, we think that the possible reason is that this efficient backbone network may not be suitable for training the domain adaptation model. However, since the architecture of the network was created by us, we conducted quite limited tests, and maybe this approach still has potential. We still used the original CycleGAN to implement our method and conduct follow-up experiments based on the experimental results.

Table 2. Comparison of domain adaption model variants.

Models	Noise	Backbone	F1 Score (%)	
			Total	Night
CycleGAN	✔	ResNet	**73.6**	**68.5**
		ResNet	72.8	66.1
		MobileNetV2	71.2	60.0
		EfficientNet	72.6	65.2

4.2.3. Image Pre-Processing

We also explored the use of image pre-processing technology instead of GAN for image style conversion in order to solve the difficult problem of night scene detection.

Specifically, we tried two methods—histogram equalization (HE) and contrast limited adaptive histogram equalization (CLAHE)—to brighten the nighttime image to a state close to a daytime image. We applied the above image preprocessing method to the nighttime image in the test dataset. We transferred the image to the hue, saturation, value (HSV) coordinate system, and adjusted the value representing the brightness through the above two methods. Next, we tested on the original ERFNet using the pre-processed images rather than the original nighttime images as part of the test dataset. This step is to evaluate whether simply changing the image brightness can improve the lane detection ability of the nighttime image. However, our experimental results in Table 3 show that only adjusting the brightness of the image will not significantly increase the nighttime image's capacity for lane detection. Although the brightness of the image may thus be enhanced, there are still difficulties in lane detection due to problems such as uneven light brightness and image noise. This result illustrates the complexity of style transfer and the limitation of a single brightness adjustment for solving the lane detection problem in nighttime images.

Table 3. Comparison of image pre-processing.

Models	Version	F1 Score (%)	
		Total	Night
ERFNet		**73.1**	**67.1**
	HE	71.3	57.7
	CLAHE	71.0	55.8

4.3. Internal Experiments

To ascertain whether our model could achieve optimal performance, we performed several internal experiments to assess which method performed better. Our internal experiments included whether to train the domain adaptation model in conjunction with the lane detection model, whether to update only some parameters when training the lane detection model, and the dimensions of the feature map the feature fusion module produced.

4.3.1. Domain Adaption Model Training Strategy

In order to train the best domain adaptation model, we not only used the original CycleGAN to train the domain conversion model that converts between daytime and nighttime, but also connected the CycleGAN model to the lane detection model. We expected to use the lane detection loss to constrain CycleGAN to generate synthetic images while maintaining the features and shapes of lanes as much as possible. Regarding the way of concatenating the lane detection model, we also adopted the ERFNet model as the lane detection model, using two methods. The first method was to directly enter the synthetic image generated by CycleGAN into the lane detection model, and the second was to combine CycleGAN with the feature fusion module to output the feature map into the lane detection model. We also tested the lane detection models of the above two methods as frozen weights. We inputted the weights of the trained model into the domain adaptation module of our proposed method and tested which method has the best lane detection result.

The test results in Table 4 demonstrate that the model trained directly using the original CycleGAN had the best results. We speculate that this may mean that even if the lane detection model is not connected in series to restrict the training of the model, the original CycleGAN can also adequately preserve the characteristics of the lane line while transferring the domain. On the contrary, the lane detection loss may not allow the model to focus on domain translation. Therefore, in the following experiments, we still directly used the original CycleGAN as the domain adaptation model.

Table 4. Comparison of domain adaption model training strategies.

Models	Image Segmentation Module	Feature Fusion Module	Freeze (Image Segmentation Module)	F1 Score (%)	
				Total	**Night**
CycleGAN				**73.6 (+0.5)**	**68.5 (+1.4)**
	✔			73.2 (+0.1)	67.8 (+0.7)
	✔		✔	73.2 (+0.1)	67.8 (+0.7)
	✔	✔		72.8 (−0.3)	66.6 (−0.5)
	✔	✔	✔	72.7 (−0.4)	66.8 (−0.3)

4.3.2. Optimizer Strategy

In the lane detection model training phase, since our model included a domain adaptation module, feature fusion module, and image segmentation module, whether to update all the modules during the training phase or to freeze the parameters of some modules was a question worth discussing. We tried three ways to update the modules. The first method was to update all modules, the second method was to update only the image segmentation module, and the third method was to update only the feature fusion module and image segmentation module. Table 5 demonstrates that the superior method was to freeze the parameters of the domain adaptation module and only update the feature fusion module and the image segmentation module. Freezing the parameters of the domain adaptation module allows the module to focus on extracting the nighttime and daytime features of the image without being affected by the loss of lane detection. Freezing part of the parameters can reduce GPU memory usage and help the model converge faster. This could be a potential reason for the slightly superior performance of this method compared to others.

Table 5. Comparison of optimizer strategies.

Models	Optimizer			Consistent Loss	F1 Score (%)	
	Domain Adaption Module	Feature Fusion Module	Image Segmentation Module		**Total**	**Night**
ERFNet					72.1 (−1.0)	67.1 (+0.0)
	✔	✔	✔	✔	73.3 (+0.2)	68.6 (+1.5)
		✔	✔		73.6 (+0.5)	68.5 (+1.4)
				✔	**74.3 (+1.2)**	**69.1 (+2.0)**
			✔		70.5 (−2.6)	62.3 (−4.8)
				✔	71.1 (−2.0)	62.4 (−4.7)

4.3.3. Output Size of the Feature Map

During the training phase of the lane detection model, the feature fusion module extracts feature maps of various sizes generated by the domain adaptation module, and finally outputs the feature maps to the image segmentation module. As a method proposed by us, the feature fusion module required more tests to adjust the model architecture. In this experiment, we adjusted the architecture of the module to output feature maps of various sizes. The first feature map size was identical to that of the original input image, and the second feature map size was half the size of the input image. According to the experimental findings in Table 6, feature maps of the same size as the input image can perform better. Although the feature map of the same input image size requires more computation, it can also retain more features to the image segmentation module, making the overall model perform better.

Table 6. Comparison of output sizes of the feature map.

Models	Feature Map Size	Consistent Loss	F1 Score (%)	
			Total	Night
ERFNet	Half Size (488 × 104)		73.5 (+0.4)	67.9 (+0.8)
		✔	73.8 (+0.7)	67.2 (+0.1)
	Same Size (976 × 208)		73.6 (+0.5)	68.5 (+1.4)
		✔	**74.3 (+1.2)**	**69.1 (+2.0)**

4.3.4. Qualitative Analysis with Grad-CAM

As shown in Figure 4, we used gradient-weighted class activation mapping (Grad-CAM) to qualitative analyze whether the lane detection model we trained pays attention to lane features at various stages of different modules. First, we analyzed the outputs of the four lanes at two different locations of the feature fusion module. After analysis, in the two layers of the feature fusion module, the distribution of the heat map was not concentrated in the lane area. This means that the feature fusion module seemed to pay little attention to lane features at these levels. We also outputted heat maps for analysis in the encoder and decoder of the image segmentation module. On the part of the encoder and decoder, the distribution of heat maps was indeed concentrated in the lane area. This shows that in these layers, the model did pay attention to and learn lane-related features. As for why only the encoder and decoder parts had an obvious focus on the lane, one possible reason is that in the encoder part, the size of the feature map was compressed to a very small size, making it impossible for the shallower network layer to focus on extracting the features of the lane. In general, from the results of the Grad-CAM qualitative analysis, our lane detection model may not have paid attention to lane features in the feature fusion module, but it featured attention on the lanes in the encoder and decoder parts of the image segmentation module.

4.4. External Experiments

In order to confirm that our method could effectively enhance the performance of the original lane detection model, our external experiments consisted of using our method with three different models, testing its performance against CULane challenging scenes, and testing against additional challenging scenes of our own choice.

4.4.1. CULane Total & Night

Table 7 shows the F1 score, Precision, and Recall of the proposed method on the CULane dataset in different models. The table lists the performance difference between our method and the original model on total and night scenes. We choose three models to use our method, which was to connect the domain adaptation module and feature fusion module to these models. The final version of our method also added a consistent loss to assist the training of the model. The experiments demonstrated that the performance of the model can be effectively improved by using our proposed method. The F1 score of the original ERFNet model in the CULane test dataset was 73.1, and the F1 score in the night scene was 67.1. In our method, the overall F1 score of the ERFNet-based model was 73.6, and the F1 score of the night scene was 68.5. After adding consistent loss, the F1 score increased to 74.3, and the night scene increased to 69.5. We also tested our method with DLA and DDRNet models. Their overall F1 score could be improved by 0.5 to 2.4, and the F1 score in night scenes could be improved by 2.3 to 3.7. After further analysis of Precision and Recall, it can be found that although the Recall was sometimes slightly lower than the original model, the Precision could be greatly improved. Overall, this proves that our method is applicable to different image segmentation models and can significantly enhance the performance of the model.

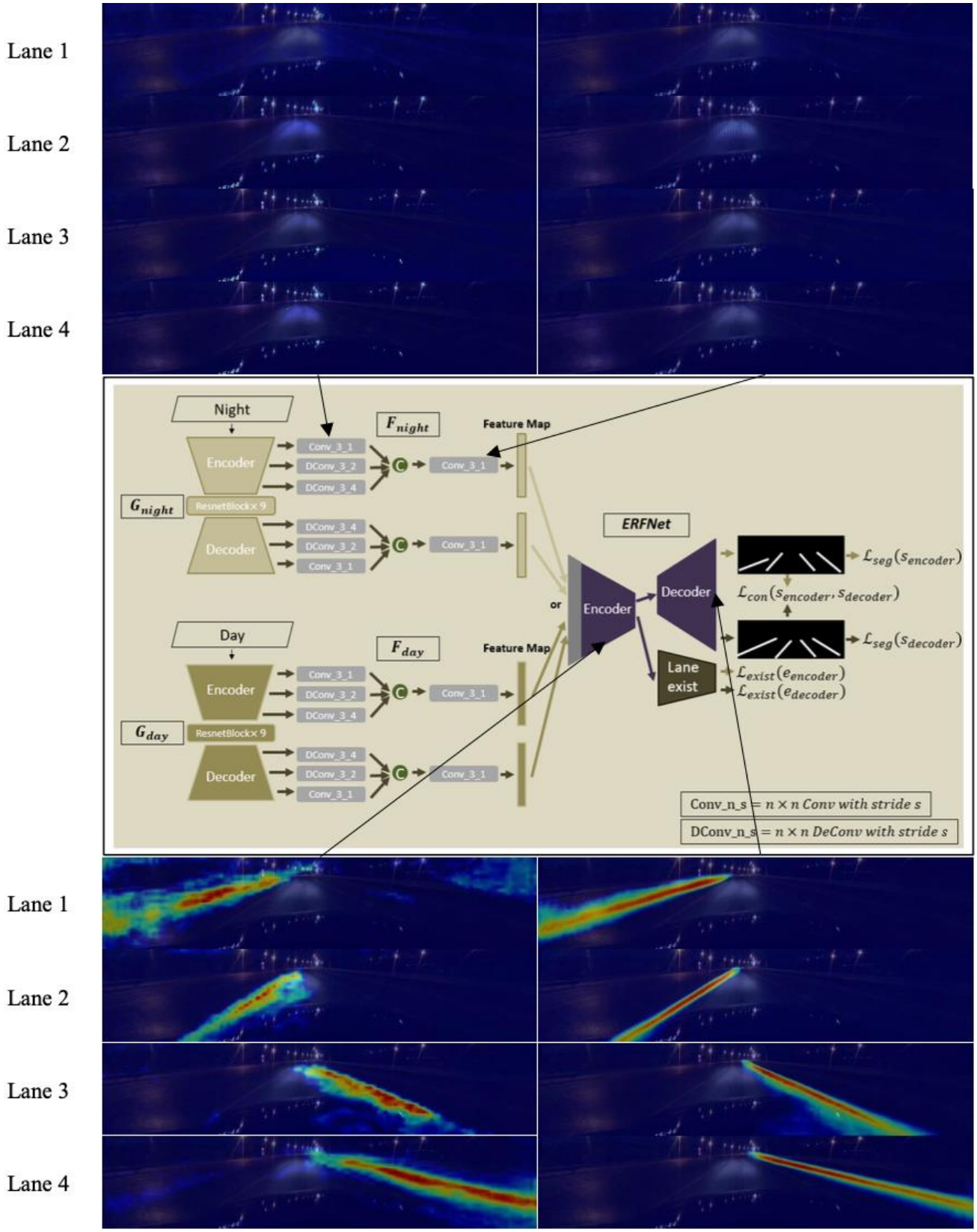


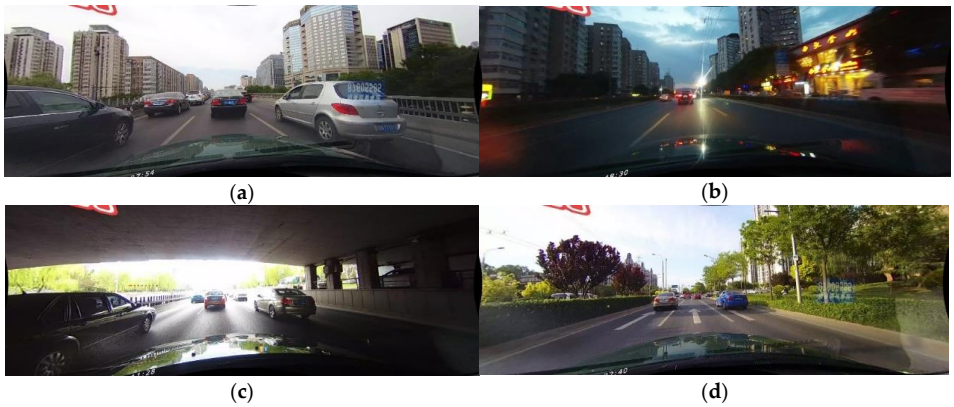
Figure 4. Qualitative analysis result of Grad-CAM.

Table 7. Comparison against CULane (total and night).

Models	Version	F1 Score (%) (Precision/Recall) (%)	
		Total	**Night**
ERFNet	Original	73.1 (73.4/72.9)	67.1 (67.5/66.8)
	Ours	73.6 (75.4/72.0)	68.5 (74.2/63.6)
	Ours (Con:L1)	74.3 (76.1/72.6)	69.1 (73.3/65.4)
DLA	Original	71.5 (71.7/71.2)	64.4 (65.4/63.4)
	Ours	73.1 (73.7/72.5)	66.6 (68.2/65.1)
	Ours (Con:L1)	73.9 (76.1/71.9)	68.1 (72.7/64.0)
DDRNet	Original	68.9 (71.8/66.3)	59.9 (67.0/54.2)
	Ours	69.4 (76.3/63.7)	61.7 (73.1/53.3)
	Ours (Con:L1)	69.6 (76.1/64.2)	62.2 (74.1/53.6)

4.4.2. CULane Challenge Scene

In the CULane test dataset, in addition to the night scene, there were other challenging scenes, termed Crowded, Dazzle, Shadow, and Arrow. Figure 5 shows examples of these scenes. Crowded represents a scene with many vehicles. Because too many vehicles will increase the probability of lane lines being obscured, it will increase the difficulty of detecting lanes. Dazzle refers to scenes with glare, which usually appear in the early morning or evening. Excessive strong light will not only reduce the quality of the entire image but also blur some lane lines, which will affect lane detection performance. Shadow refers to lane lines under bridges or other shadows. Shadows will reduce the contrast of the image, and the edges of lane lines may become less clear. Arrow refers to objects similar to lane lines such as pointing lines in the lane, which may cause the model to misjudge. We also used the ERFNet model to validate the efficacy of our approach in enhancing the model's recognition capability in these challenging scenarios. The experiments in Table 8 show that our method effectively improves the recognition ability of the model in these scenes, and the F1 score increased by about 0.1 to 1.2%. After adding consistent loss, the F1 score increased by 0.5 to 3.5% in total.



(**a**) (**b**) (**c**) (**d**)

Figure 5. The example of challenge scenarios from the CULane test dataset. (**a**) Crowded; (**b**) Dazzle; (**c**) Shadow; (**d**) Arrow.

Table 8. Comparison of CULane challenge scenes.

Models	Version	F1 Score (%) (Precision/Recall) (%)			
		Crowded	**Dazzle**	**Shadow**	**Arrow**
ERFNet	Original	71.6 (72.4/70.8	65.6 (66.3/64.9)	71.4 (71.7/71.3	87.1 (88.7/85.5)
	Ours	71.7 (72.9/70.6	66.0 (68.9/63.4)	72.6 (73.6/71.6	87.6 (89.6/85.6)
	Ours (Con:L1)	72.1 (73.7/70.6	66.3 (69.6/63.2)	74.9 (76.5/73.4	88.2 (90.0/86.5)

4.4.3. Other CULane Challenge Scene

In order to verify that our method could have a more stable recognition ability in other challenging scenes, we self-selected two special scenarios from the CULane test set to test our method. Figure 6 shows examples of these scenes. The first scene is an indoor scene, including underground parking lots and tunnels, with a total of 102 images. Indoor scenes usually lack natural light sources, which is different from daytime scenes, which comprise a large proportion of the training data. The visibility of lane markings may also be impacted by intense light from artificial lighting. The second scene is the wet ground scene, and there are 240 images in total. The wet ground will change the reflection characteristics of the road surface, which may cause the edges and color features of lane markings to become unclear, thus affecting the accuracy of lane detection. We also use the ERFNet model to verify that our method can effectively enhance the relatively stable identification ability of the model in these special scenarios. The experiments presented in Table 9 indicate that our method significantly enhances the model's recognition capability within these scenarios. The F1 score of indoor scenes was increased by about 1.5 to 2.1%, and the F1 score of the wet ground scene was increased by about 5.2 to 8.2%.



(**a**) (**b**)

Figure 6. The example of other challenge scenarios from the CULane test dataset. (**a**) Indoor; (**b**) wet ground.

Table 9. Comparison of other CULane challenge scenes.

Models	Version	F1 Score (%) (Precision/Recall) (%)	
		Indoor	**Wet Ground**
ERFNet	Original	48.5 (56.3/42.6)	54.4 (50.2/59.3)
	Ours	50.0 (67.5/39.7)	59.6 (63.4/56.2)
	Ours (Con:L1)	50.6 (70.7/39.4)	62.6 (64.7/60.7)

4.4.4. Examples of Lane Detection Results

In this section, we visualize the results of lane detection in challenging scenarios and compare the difference between our proposed method and the original ERFNet model. Our

visualization results in Figure 7 clearly illustrate that our method outperforms the original model significantly in complex scenarios such as night, shadowed areas, and indoors. For example, our method could detect the leftmost and rightmost lanes that were difficult to identify with the original ERFNet model. In some lanes, the lanes detected by our method were also relatively complete. Overall, our method could recognize lane lines more accurately in these challenging scenarios, ensuring drivers could get reliable lane guidance in any environment. It is worth noting that our method performed effectively with and without consistent loss. Compared to the original ERFNet model, our method already achieved significant performance gains without loss of consistent. When we introduced consistent loss, it further strengthened the stability and accuracy of the model, enabling it to maintain excellent performance in different scenarios. Regardless of applying the consistent loss or not, our method achieved superior performance in various scenarios, which will have a profound impact on improving the safety and reliability of self-driving and assisted-driving systems.

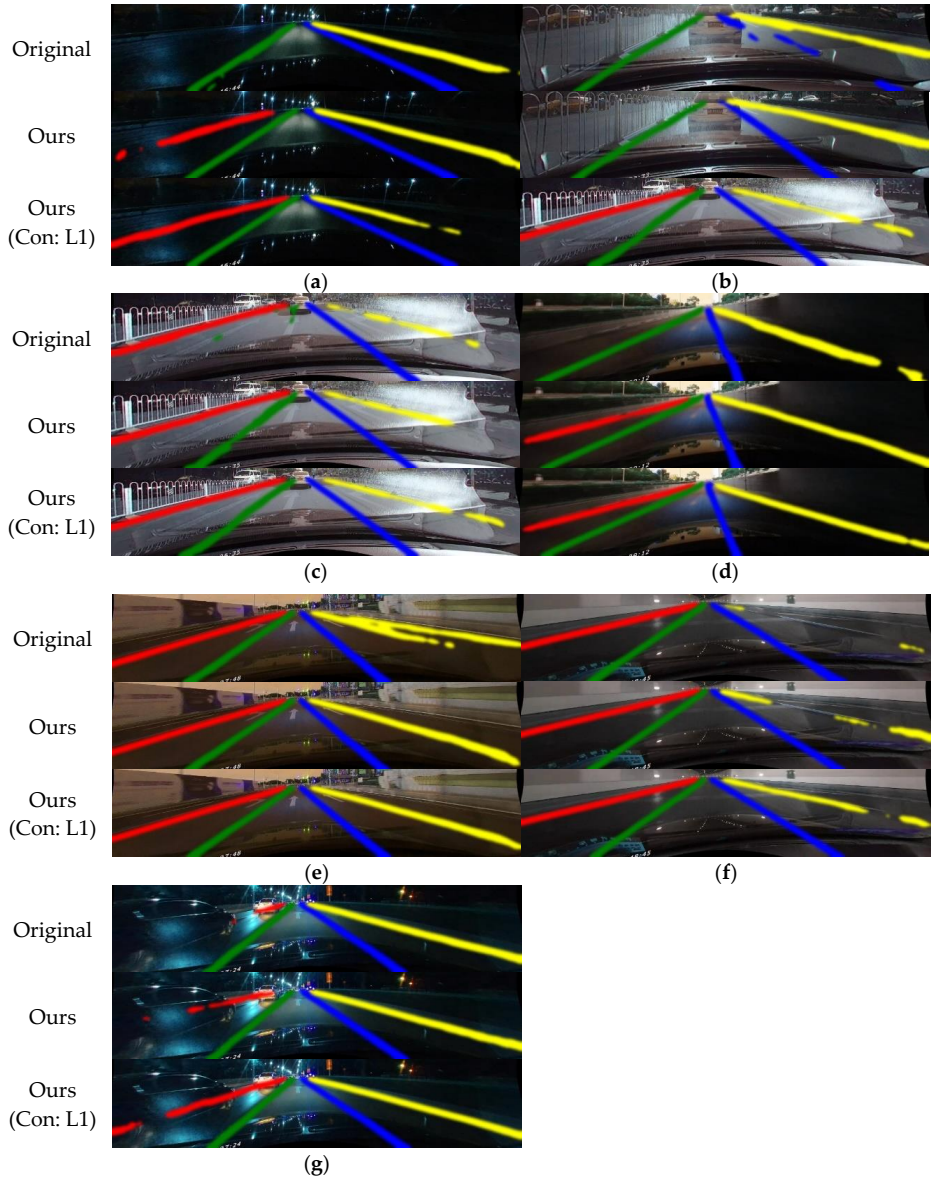


Figure 7. Visualization results of challenging scenes in the CULane test dataset. (**a**) Night; (**b**) crowded; (**c**) dazzle; (**d**) shadow; (**e**) arrow; (**f**) indoor; (**g**) wet ground.

5. Conclusions

We proposed a feature extraction enhancement strategy based on style translation to solve the poor detection performance of general lane detection models in challenging scenes due to the lack of diversity in the datasets. We proposed the use of CycleGAN for image style transfer to enhance the robustness of lane detection models at night or in other challenging scenarios. We took advantage of CycleGAN not requiring paired images to train a domain adaptation model using daytime and nighttime images from existing lane datasets. The proposed two-stage approach improved lane detection performance using a feature fusion architecture combined with a domain adaptation model and an image segmentation model. The proposed consistent loss function was utilized to mitigate variations in model detection capabilities under different driving scenarios so that the model could perform better. In the test results of the CULane test dataset, the total F1 score of the original ERFNet model was 73.1, and the F1 score of nighttime scenes was 67.1. After using our method, the performance of the model improved to an F1 score of 74.3 and an F1 score of 69.1 at night. In other challenging scenarios, our method also improved the original model by 0.7% to 15.1%. The experimental results substantiate the potential of our approach in improving lane detection performance in challenging driving scenarios, thereby enhancing driving safety.

Our approach can be applied to enhance advanced driver assistance systems (ADASs), where robust lane detection is crucial to ensure safer and more reliable autonomous driving. Another reliable application is the road inspection system, which can effectively maintain road infrastructure and ensure safe navigation of vehicles under various conditions by identifying damaged or missing lane markings. Although we have proven that our method can effectively improve the performance of the model in challenging scenarios, it is undeniable that our model also takes longer to output results during training and inference than the original model. Since we are currently focusing on the performance of the model at night, it may not have the same good performance in other, more diverse challenging scenes. Therefore, in future work, we have several directions to expand our research. In terms of our methods, first, we hope to try more different strategies to train our domain adaptation model. In our preliminary experiments and internal experiments, we mentioned that we used different strategies to train the domain adaptation model, but we could not achieve the desired experimental results. We believe that this is still a potential research direction, and therefore would be interested in attempting different strategies to train better domain adaptation models to improve the performance of lane detection models. Furthermore, we aspire to train domain adaptation models based on images of different styles or on different brightnesses. Our current approach mainly trains domain adaptation models based on daytime and nighttime images, achieving good performance in challenging scenarios under different lighting conditions. However, due to the absence of relevant datasets, we were unable to train or test our models for scenes such as rain and snow, nor could we assess whether the model's performance exhibits significant disparities under varying levels of brightness. In future work, we hope to obtain related datasets to train more domain adaptation models with different styles or different brightnesses.

Author Contributions: Conceptualization, E.H.-C.L.; Methodology, E.H.-C.L. and W.-C.C.; Software, W.-C.C.; Validation, W.-C.C.; Formal analysis, E.H.-C.L.; Investigation, W.-C.C.; Resources, E.H.-C.L.; Data curation, W.-C.C.; Writing—original draft, E.H.-C.L. and W.-C.C.; Writing—review & editing, E.H.-C.L.; Visualization, W.-C.C.; Supervision, E.H.-C.L.; Project administration, E.H.-C.L.; Funding acquisition, E.H.-C.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Science and Technology Council grant number NSTC 112-2628-M-006-008-MY2.

Data Availability Statement: The data presented in this study are available in article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Markets and Markets. Available online: https://www.marketsandmarkets.com (accessed on 1 January 2009).
2. Zhang, C.; Chen, Z.; Li, X.; Zhao, T. Anti-Collision Path Planning and Tracking of Autonomous Vehicle Based on Optimized Artificial Potential Field and Discrete LQR Algorithm. *World Electr. Veh. J.* **2024**, *15*, 522. [CrossRef]
3. Dudziak, A.; Stoma, M.; Kuranc, A.; Caban, J. Assessment of Social Acceptance for Autonomous Vehicles in Southeastern Poland. *Energies* **2021**, *14*, 5778. [CrossRef]
4. Liu, T.; Chen, Z.; Yang, Y.; Wu, Z.; Li, H. Lane Detection in Low-light Conditions Using an Efficient Data Enhancement: Light Conditions Style Transfer. In Proceedings of the 2020 IEEE Intelligent Vehicles Symposium, Las Vegas, Nevada, USA, 19 October–13 November 2020; pp. 1394–1399.
5. Zhu, J.-Y.; Park, T.; Isola, P.; Efros, A.A. Unpaired Image-to-image Translation Using Cycle-consistent Adversarial Networks. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2223–2232.
6. Liu, M.-Y.; Tuzel, O. Coupled Generative Adversarial Networks. In *Advances in Neural Information Processing Systems, Proceedings of the 38th Annual Conference on Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016*; Curran Associates Inc: Red Hook, NY, USA, 2016; Volume 29.
7. Isola, P.; Zhu, J.-Y.; Zhou, T.; Efros, A.A. Image-to-image Translation with Conditional Adversarial Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1125–1134.
8. Ronneberger, O.; Fisher, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015, Proceedings of the 18th International Conference, Munich, Germany, 5–9 October 2015*; Springer: Cham, Switzerland, 2015; pp. 234–241.
9. Liu, M.-Y.; Breuel, T.; Kautz, J. Unsupervised Image-to-image Translation Networks. In *Advances in Neural Information Processing Systems, Proceedings of the Thirty-First Annual Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017*; Curran Associates Inc.: Red Hook, NY, USA, 2017; Volume 30.
10. Yi, Z.; Zhang, H.; Tan, P.; Gong, M. DualGAN: Unsupervised Dual Learning for Image-to-Image Translation. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2849–2857.
11. Kim, T.; Cha, M.; Kim, H.; Lee, J.-K.; Kim, J. Learning to Discover Cross-Domain Relations with Generative Adversarial Networks. In Proceedings of the International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 1857–1865.
12. Huang, X.; Liu, M.-Y.; Belongie, S.; Kautz, J. Multimodal Unsupervised Image-to-image Translation. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; pp. 172–189.
13. Lee, H.-Y.; Tseng, H.-Y.; Huang, J.-B.; Singh, M.; Yang, M.-H. Diverse Image-to-Image Translation via Disentangled Representations. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; pp. 35–51.
14. Shen, Z.; Huang, M.; Shi, J.; Xue, X.; Huang, T. Towards Instance-Level Image-To-Image Translation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 3683–3692.
15. Ni, Z.; Yang, W.; Wang, H.; Wang, S.; Ma, L.; Kwong, S. Cycle-Interactive Generative Adversarial Network for Robust Unsupervised Low-Light Enhancement. In Proceedings of the 30th ACM International Conference on Multimedia, Lisbon, Portugal, 10–14 October; pp. 1484–1492.
16. Wang, Z.; Ren, W.; Qiu, Q. LaneNet: Real-Time Lane Detection Networks for Autonomous Driving. *arXiv* **2018**, arXiv:1807.01726.
17. Liang, D.; Guo, Y.; Zhang, S.; Zhang, S.-H.; Hall, P.; Zhang, M.; Hu, S. LineNet: A Zoomable CNN for Crowdsourced High Definition Maps Modeling in Urban Environments. *arXiv* **2018**, arXiv:1807.05696.
18. Pan, X.; Shi, J.; Luo, P.; Wang, X.; Tang, X. Spatial as Deep: Spatial CNN for Traffic Scene Understanding. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; Volume 32.
19. Ghafoorian, M.; Nugteren, C.; Baka, N.; Booij, O.; Hofmann, M. EL-GAN: Embedding Loss Driven Generative Adversarial Networks for Lane Detection. In Proceedings of the European Conference on Computer Vision Workshops, Munich, Germany, 8–14 September 2018.
20. Hou, Y.; Ma, Z.; Liu, C.; Loy, C.C. Learning Lightweight Lane Detection CNNs by Self Attention Distillation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1013–1021.
21. Zou, Q.; Jiang, H.; Dai, Q.; Yue, Y.; Chen, L.; Wang, Q. Robust Lane Detection from Continuous Driving Scenes Using Deep Neural Networks. *IEEE Trans. Veh. Technol.* **2019**, *69*, 41–54. [CrossRef]
22. Tabelini, L.; Berriel, R.; Paixao, T.M.; Badue, C.; De Souza, A.F.; Oliveira-Santos, T. 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, Nashville, TN, USA, 20–25 June 2021; pp. 294–302.

23. Zhang, Y.; Lu, Z.; Ma, D.; Xue, J.-H.; Liao, Q. Ripple-GAN: Lane Line Detection with Ripple Lane Line Detection Network and Wasserstein GAN. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 1532–1542. [CrossRef]
24. Abualsaud, H.; Liu, S.; David, B.L.; Situ, K.; Rangesh, A.; Trivedi, M.M. LaneAF: Robust Multi-Lane Detection with Affinity Fields. *IEEE Robot. Autom. Lett.* **2021**, *6*, 7477–7484. [CrossRef]
25. Ko, Y.; Lee, Y.; Azam, S.; Munir, F.; Jeon, M.; Pedrycz, W. Key Points Estimation and Point Instance Segmentation Approach for Lane Detection. *IEEE Trans. Intell. Transp. Syst.* **2021**, *23*, 8949–8958. [CrossRef]
26. Feng, Z.; Guo, S.; Tan, X.; Xu, K.; Wang, M.; Ma, L. Rethinking Efficient Lane Detection via Curve Modeling. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 17062–17070.
27. Zheng, T.; Huang, Y.; Liu, Y.; Tang, W.; Yang, Z.; Cai, D.; He, X. CLRNet: Cross Layer Refinement Network for Lane Detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 28–24 June 2022; pp. 898–907.
28. Dong, Y.; Patil, S.; van Arem, B.; Farah, H. A Hybrid Spatial–temporal Deep Learning Architecture for Lane Detection. *Comput. Aided Civ. Infrastruct. Eng.* **2023**, *38*, 67–86. [CrossRef]
29. Hoffman, J.; Tzeng, E.; Park, T.; Zhu, J.-Y.; Isola, P.; Saenko, K.; Efros, A.A.; Darrell, T. CyCADA: Cycle-Consistent Adversarial Domain Adaptation. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; 2018; pp. 1989–1998.
30. Huang, S.-W.; Lin, C.-T.; Chen, S.-P.; Wu, Y.-Y.; Hsu, P.-H.; Lai, S.-H. AugGAN: Cross Domain Adaptation with GAN-based Data Augmentation. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; pp. 718–731.
31. Lin, C.-T.; Wu, Y.-Y.; Hsu, P.-H.; Lai, S.-H. Multimodal Structure-Consistent Image-to-Image Translation. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 11490–11498.
32. Bhattacharjee, D.; Kim, S.; Vizer, G.; Salzmann, M. DUNIT: Detection-based Unsupervised Image-to-Image Translation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 6–12 December 2020; pp. 4787–4796.
33. Shao, X.; Wei, C.; Shen, Y.; Wang, Z. Feature Enhancement Based on CycleGAN for Nighttime Vehicle Detection. *IEEE Access* **2020**, *9*, 849–859. [CrossRef]
34. Lin, C.-T.; Kew, J.-L.; Chan, C.S.; Lai, S.-H.; Zach, C. Cycle-object Consistency for Image-to-image Domain Adaptation. *Pattern Recognit.* **2023**, *138*, 109416. [CrossRef]
35. Romera, E.; Alvarez, J.M.; Bergasa, L.M.; Arroyo, R. ERFNet: Efficient Residual Factorized ConvNet for Real-Time Semantic Segmentation. *IEEE Trans. Intell. Transp. Syst.* **2017**, *19*, 263–272. [CrossRef]
36. Yu, F.; Wang, D.; Shelhamer, E.; Darrell, T. Deep Layer Aggregation. In Proceedings of the IEEE Conference on Computer Vision. and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 2403–2412.
37. Hong, Y.; Pan, H.; Sun, W.; Jia, Y. Deep Dual-resolution Networks for Real-time and Accurate Semantic Segmentation of Road Scenes. *arXiv* **2021**, arXiv:2101.06085.
38. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.-C. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In Proceedings of the IEEE Conference on Computer Vision. and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520.
39. Tan, M.; Quoc, V. Le, EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 10–15 June 2019; pp. 6105–6114.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.