Camera-based 2D Object Detection and 2D semantic segmentation for Environmental Perception

11910931 Xintong DUAN
11911633 Xiaoxuan WANG
11910838 Hanxi SUN

June 7, 2023

1 Research Problem

Automatic driving technology has become an important field of artificial intelligence technology applications, and environmental perception is one of the four core technologies of automatic driving. Environmental perception is in the front part of the automatic driving system, which focuses on analysing data of city road scenario pictures. The analysis results of environmental perception module will support the decision making of driving systems. At the same time, the development of autonomous driving is currently limited by environmental perception.The application of environmental perception systems remains a large gap between theory and realization.

In order to meet the challenges, we apply camera - based 2D object detection, lane detection and 2D semantic segmentation for environment perception. The tasks mainly aim at object recognition and tracking including identifying landmarks, pedestrians, vehicles, lane and other objects encountered on the road and tracking dynamic objects to prevent collisions.

Meanwhile, recognition models trained along with representations on one large dataset do not generalize well on other datasets because of the phenomenon of dataset bias and domain shift. Therefore, addressing domain shift by applying domain adaptation plays an important role in the task of environmental perception to meet the standards for automatic driving.

2 Research Goal

Object detection[15], lane detection and semantic segmentation are important computer vision tasks aiming at detecting instances of visual objects of a cer-

tain class (such as humans, vehicles, guardrail, lane) in digital images, which satisfies the goal of environment perception. Hence, we focus on applying object detection, lane detection and semantic segmentation to track dynamic or static targets on the road in order to enhance environmental perception for automatic drive.

In 2D object detection task, We will train YOLOV3-Darknet53, a common collocation for target detection in neural network model to manage object detection task. In addition, we will also use YOLOV5 model, which possesses very good accuracy and speed at present. For our object detection tasks on city road scenario datasets, speed obtained for a given computation mechanism is in the millisecond level. The trained models are equipped with the ability to manage object recognition, object tracking, identifying landmarks, pedestrians, vehicles and other objects encountered on the road to prevent traffic accidents.

Turning to lane detection task, we apply ERFNet(Efficient Residual Factorized ConvNet) and SCNN (Spatial CNN) to detect the lane on the road.

Meanwhile, in the 2D semantic segmentation task, we train DeepLab-v2[3] framework with ResNet-101[4],a common collocation for target semantic segmentation in neural network model to manage street scenes semantic segmentation task to support environmental perception. To improve the generalization of the model, we implied DCT domain adaptation and category-level adversaries for semantics consistent domain adaptation.

3 Research Design

In this project, we will be using different object detection models and semantic segmentation models to manage object recognition and tracking tasks in environment perception. Meanwhile, different combinations of datasets and data augmentation techniques and other domain adaptation techniques are adopted to improve the performance of the models and to address the domain shift in order to improve the generalization of the models.

3.1 Object detection

3.1.1 Dataset

- BDD100K
  BDD100K[13] is a large-scale diverse self driving dataset. It contains 120,000,000 examples collected in different cities, under different weathers and at different time of a day.

- Tsinghua-Tencent 100K
  Tsinghua-Tencent 100K[14] is a dataset with 100,000 images containing 30,000 traffic sign instances and it is mainly used for road sign detection. We use this dataset as a supplement to BDD100K as the road-sign and

pedestrian labels in BDD100K is insufficient compared with the number of car labels.

- Other Datasets
  After combining above datasets, the number of different labels is still very imbalanced. For example, the number of bicycles in the dataset is only 5% of the number of vehicles. Severe imbalance of the label counts of different categories will hinder the improvement of the map of the model. Therefore, we manually pick out pictures with bicycles and pedestrians from the Internet and take pictures of bicycles and pedestrians in the campus and label them, as shown in Figure 1 and Figure 2

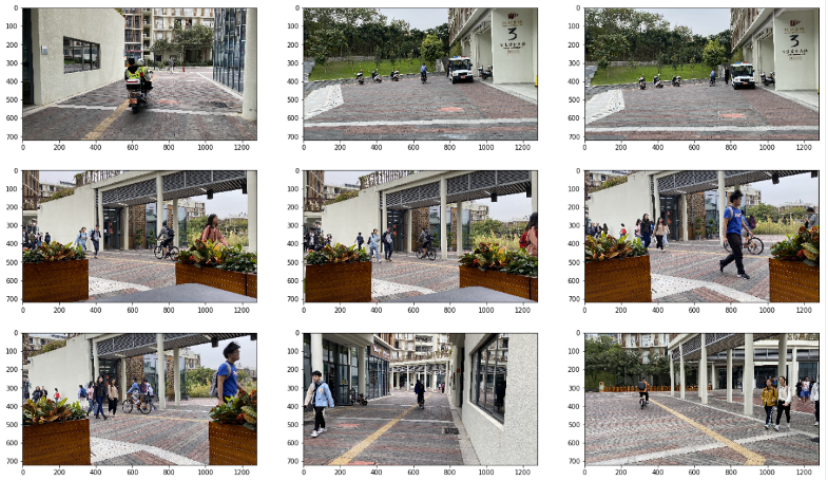

Figure 1: Pictures of bicycles and pedestrians in campus

3.1.2 Data preprocessing

Dataset cleaning is a very import step before training. In BDD100K, we found that there are many pictures that are taken at night. Also, there are many overlapping labels for cars parking alongside the road. This will significantly affect our training. Therefore, we decide to remove all the pictures at night and pick out the largest 2000 pictures for the best resolution. Sample pictures are shown in Figure 3.

Besides, to overcome the label imbalance problem in the original dataset, we manually pick out pictures with relatively large amount of bicycle labels as bicycles are rare in the dataset. Meanwhile, we also removed some obvious mistakes and very tiny labels.

We use Tsinghua-Tencent 100K dataset because of its high resolution and it is taken in China in daytime. We also manually select pictures with many

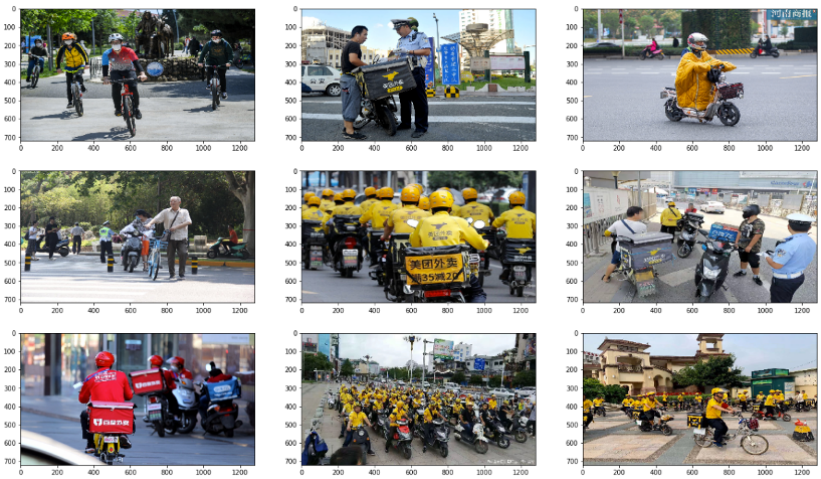

Figure 2: Pictures of bicycles on the Internet

Figure 3: Sample pictures selected from BDD100K and the visualization of labels

pedestrians and bicycles. After then, we use "Labelme" to manully label the dataset.

Dataset Augmentation is a very common trick in Computer vision to make the best use of current limited dataset. We use PLT and OpenCV module in python to manually perform Gaussian Blur, Salt and pepper noise, lightness adjustments to the pictures. Sample pictures shown in Figure 4

Figure 4: Augmentation of the bicycle pictures from Internet

3.1.3 YOLOV3 + Darknet53

First we will train YOLOv3 + Darknet53, a very classic model for object detection, on the dataset as the baseline. YOLOV3 use Darknet53 as its backbone. Darknet53 has 53 convolutional layers and there are 52 convolutional layers that form the body network and one last Fully Connected layer to process the output[6]. Looking into the main network body, the first layer is a convolution kernel with 32 filters. Then there are five residual blocks. In these residual blocks, there are one convolutional layer and a set of repeating convolutional layers. The repeating convolutioanl layers repeat for 1, 2, 8, 8, 4 times respectively. Therefore, we have $52 = 1+(1+1*2)+(1+2*2)+(1+8*2)+(1+8*2)+(1+4*2)$.

The stride of the first convolutioanl layer in every residual block is 2. Therefore, the dimension of YOLOV3 network will be reduced 5 times, which will also be $1/(2^5) = 1/32$ of the origianl size. The dimension of the final output feature figure is $416/32 = 13$ and the number of channels in the last layer is 1024[6].

Darknet53 will output three different sizes of detection figures for different objects of different sizes, as shown in Figure 5.

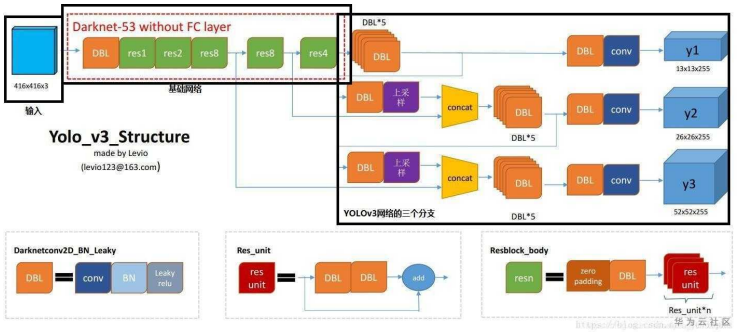

Figure 5: Structure of YOLOV3 + Darknet53

3.1.4 YOLOv5

YOLOv5 is a recent state-of-art approach for object detection provided by Ultralytics[9]. It is divided into four parts: input, backbone, neck, and prediction[8]. In the input part, Mosaic data augmentation, self adaptive scaling and anchor calculation is adopted. Mosaic basically takes the input pictures and randomly crop and resemble to from new training data. In YOLOV5, the generation of the anchor has been integrated in the main program and therefore the anchor is calculated on every training epoch and different on different dataset. CSP[10] and focus structure are added in the backbone. Different from YOLOV4, where CSP can only be found in backbone, YOLOV5 has two different CSP structures which can be found in the neck and backbone. FPN+PAN[2] structure is utilized in the neck and when prediction, GIOU_Loss[2] is used to measure the accuracy of the model. There are four models for YOLOV5, which are YOLOv5s, YOLOv5m, YOLOv5l and YOLOv5x[9]. The number of convolution kernels are different for different models, thus the third dimension of the feature map is different. The structure of YOLOV5 is shown in Figure 6

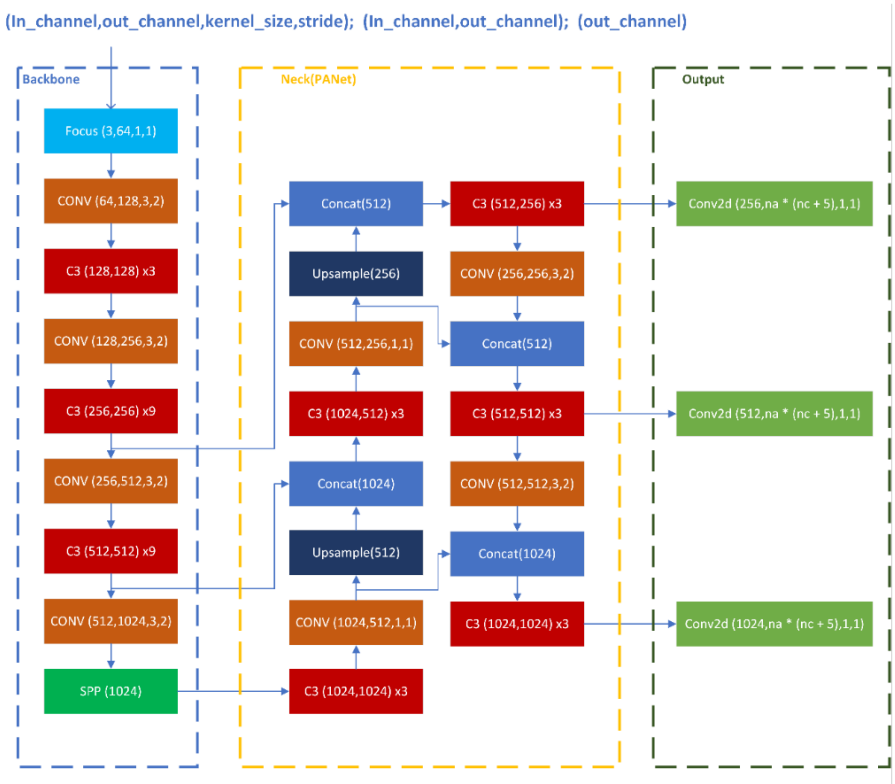

Figure 6: Structure of YOLOV5

3.2 Lane Detection

3.2.1 Dataset

- CULane
  CULane is a large scale dataset for academic research on traffic lane detection. It is collected by cameras mounted on six different vehicles driven by different drivers in Beijing. More than 55 hours of videos were collected and 133,235 frames were extracted. We have divided the dataset into 88880 for training set, 9675 for validation set, and 34680 for test set[11].

- TuSimple
  The TuSimple dataset consists of 6,408 road images on US highways. The resolution of image is 1280×720. The dataset is composed of 3,626 for training, 358 for validation, and 2,782 for testing called the TuSimple test set of which the images are under different weather conditions.

- LLAMAS
  The unsupervised llamas dataset is an automatically annotated lane marker dataset using high definition maps. It contains over 100,000 annotated images and the resolution of images is 1276 x 717 pixels[1].

- Self sampled SUSTECH dataset
  Self sampled SUSTECH dataset contains images of street scenes in Southern University of Science and Technology, which are totally collected by our group members.

3.2.2 ERFNet

ERFNet stands for Efficient Residual Factorized ConvNet for Real-time Semantic Segmentation. It improves the performance of Enet by increasing its accuracy. In ERFnet, it completely adopt 1D convolutions and the parameter number can be further reduced without losing the accuracy drastically. Besides, non-linear module is added in between 1D convolution layers to increase its ability to study more information[7]. The structure is shown in TABLE II.

3.2.3 SCNN

Spatial CNN abandon the layer-by-layer layout of traditional CNN and adopt a new connection between layers. It utilizes the slice-by-slice format in feature map and thus allow information to flow in between the rows and columns of pixels in a picture[5].

Traditional CNNs can not cope with long distance and continuous shapes, especially when there is blocking. SCNN uses wide kernels on both vertical and horizontal directions to pass the information recurrently[5]. Therefore, the spacial information is enhanced and this is especially useful for structural information. The structure is shown in Figure 3.2.3.

TABLE II

LAYER DISPOSAL OF OUR PROPOSED NETWORK (ERFNET).
"OUT-F": NUMBER OF FEATURE MAPS AT LAYER'S OUTPUT.
"OUT-RES": OUTPUT RESOLUTION FOR AN EXAMPLE
INPUT SIZE OF 1024 × 512

	Layer	Type	out-F	out-Res
ENCODER	1	**Downsampler block**	16	512x256
	2	**Downsampler block**	64	256x128
	3-7	5 x **Non-bt-1D**	64	256x128
	8	**Downsampler block**	128	128x64
	9	**Non-bt-1D** (dilated 2)	128	128x64
	10	**Non-bt-1D** (dilated 4)	128	128x64
	11	**Non-bt-1D** (dilated 8)	128	128x64
	12	**Non-bt-1D** (dilated 16)	128	128x64
	13	**Non-bt-1D** (dilated 2)	128	128x64
	14	**Non-bt-1D** (dilated 4)	128	128x64
	15	**Non-bt-1D** (dilated 8)	128	128x64
	16	**Non-bt-1D** (dilated 16)	128	128x64
DECODER	17	**Deconvolution** (upsampling)	64	256x128
	18-19	2 x **Non-bt-1D**	64	256x128
	20	**Deconvolution** (upsampling)	16	512x256
	21-22	2 x **Non-bt-1D**	16	512x256
	23	**Deconvolution** (upsampling)	C	1024x512

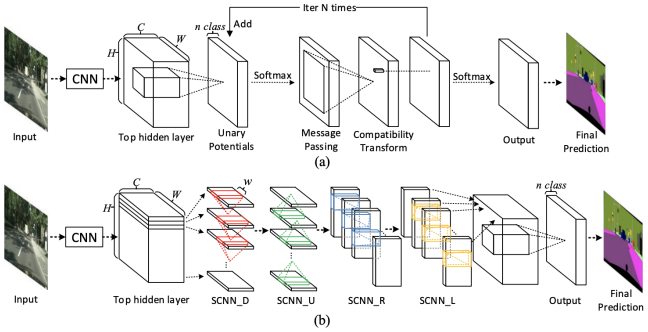

Figure 3: (a) MRF/CRF based method. (b) Our implementation of Spatial CNN. MRF/CRF are theoretically applied to unary potentials whose channel number equals to the number of classes to be classified, while SCNN could be applied to the top hidden layers with richer information.

3.3 Semantic segmentation

3.3.1 Dataset

- Cityscapes (use as target domain)
  Cityscapes is a large-scale database which focuses on semantic understanding of urban street scenes. It provides semantic, instance-wise, and dense pixel annotations for 30 classes grouped into 8 categories (flat surfaces, humans, vehicles, constructions, objects, nature, sky, and void). A sample is given in Figure 7.

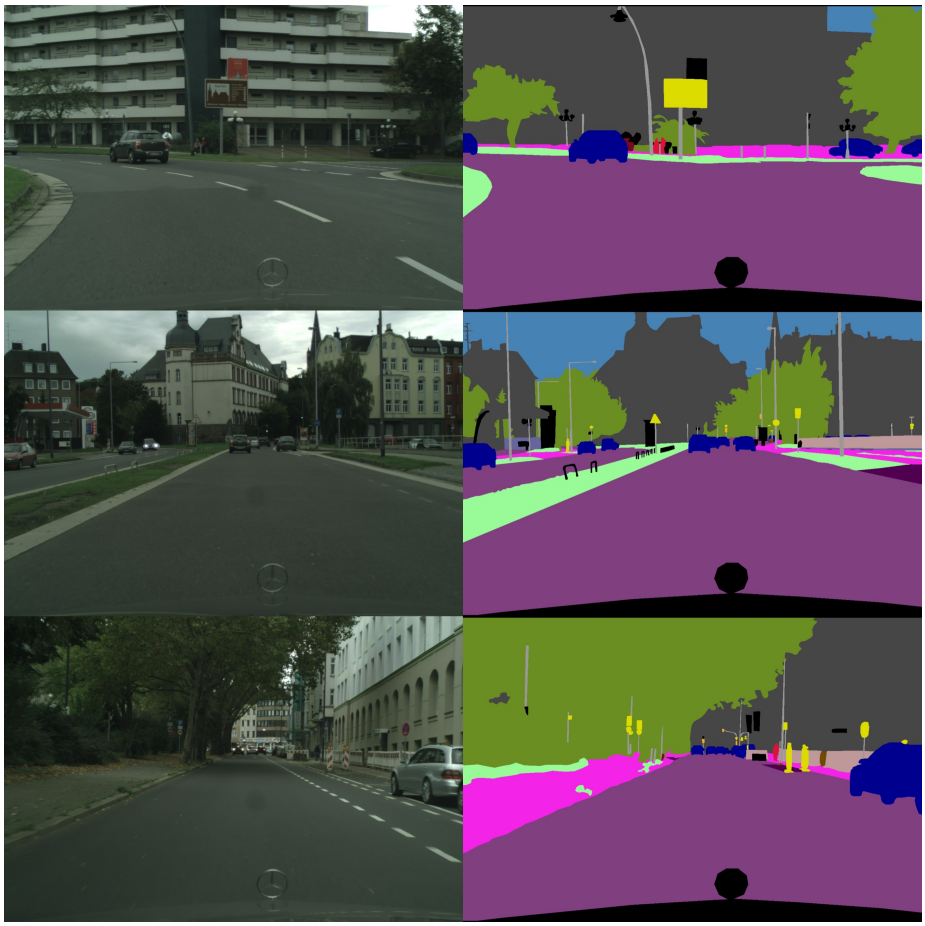

Figure 7: Sample pictures selected from Cityscapes and the visualization of labels

- GTA5 (use as source domain)
  The GTA5 dataset contains 24966 synthetic images with pixel level semantic annotation. The images have been rendered using the open-world video game Grand Theft Auto 5 and are all from the car perspective in the streets of American-style virtual cities. There are 19 semantic classes which are compatible with the ones of Cityscapes dataset. A sample is given in Figure 8.

- Self sampled SUSTECH dataset (use as target domain)
  Self sampled SUSTECH dataset contains images of street scenes in Southern University of Science and Technology.A sample is given in Figure 9

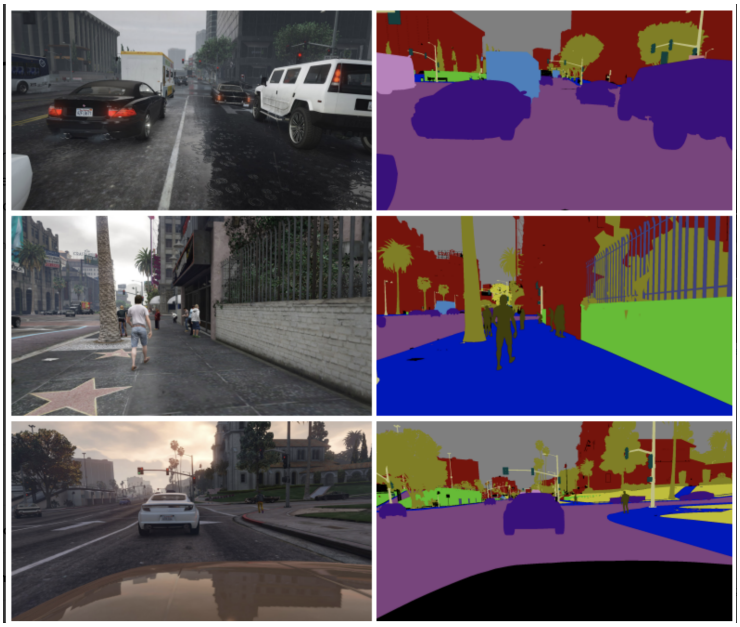

Figure 8: Sample pictures selected from GTA5 and the visualization of labels

Self sampled SUSTECH dataset

Figure 9: Sample pictures selected from Self sampled SUSTECH dataset

3.3.2 DeeplabV2+Resnet101

DeepLabv2 (shown in Figure 10)is an architecture for semantic segmentation that build on DeepLab with an atrous spatial pyramid pooling scheme.It is mainly based on DCNNs and probabilistic graph model to achieve pixel-level classification task, that is, image semantic segmentation. Due to the translation

invariance of DCNNs, DCNNs is used in many abstract image tasks. Here it has parallel dilated convolutions with different rates applied in the input feature map, which are then fused together. As objects of the same class can have different sizes in the image, ASPP helps to account for different object sizes.

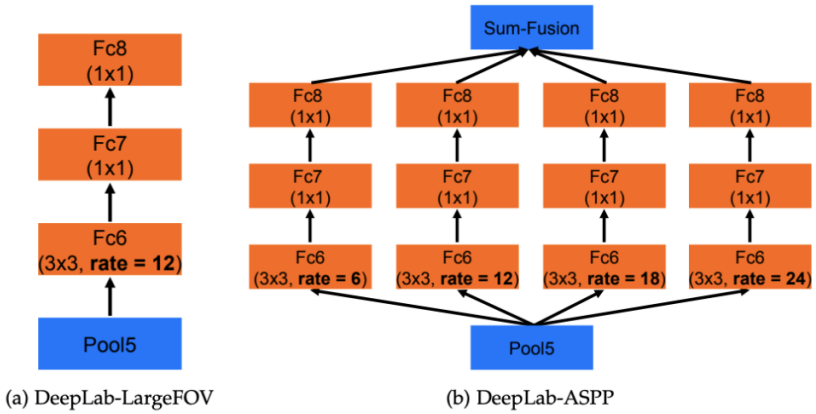

Figure 10: Structure of DeeplabV2

3.3.3 DCT adaptation

DCT (Discrete Cosine Transform) adaptation can be regarded as a simple method for unsupervised domain adaptation, whereby the discrepancy between the source and target distributions is reduced by swapping certain frequency spectrum of one with the other.
Inspired by the idea of fourier domain adaptation for semantic [12] that converts spatial images into multiple frequency components . Here we use Discrete Cosine Transform (DCT) to convert source domain images to the frequency space. In this part, frequency analysis is used to identify main components and minor components.

Source domain images after converted into frequency space by DCT are decomposed into 64 FCs by a band-pass filter. Then we identify main components and minor components in FCs by a set of control experiments. We group the 64 FCs into 6 parts and for each source image(shown in Figure 22), we first filter out FCs with indexes between certain lower or upper thresholds with a band reject filter and then train models with remaining FCS. We use classification model to identify main components and minor components by training models with certain FCs on source domain. Meanwhile, test it with target domain images.The results are shown in Table 6

After we get the main components and minor components in source domain images, we then use DCT to convert source domain images and target domain images to the frequency space. After that, we keep the main components in source-domain images while replacing the minor components in source domian images with the corresponding frequency band of the target-domain images. Finally, we use Inverse DCT to convert the source domain images back to the picture space. The whole process is given in Figure 11. So now we get some source-domain images with target style.

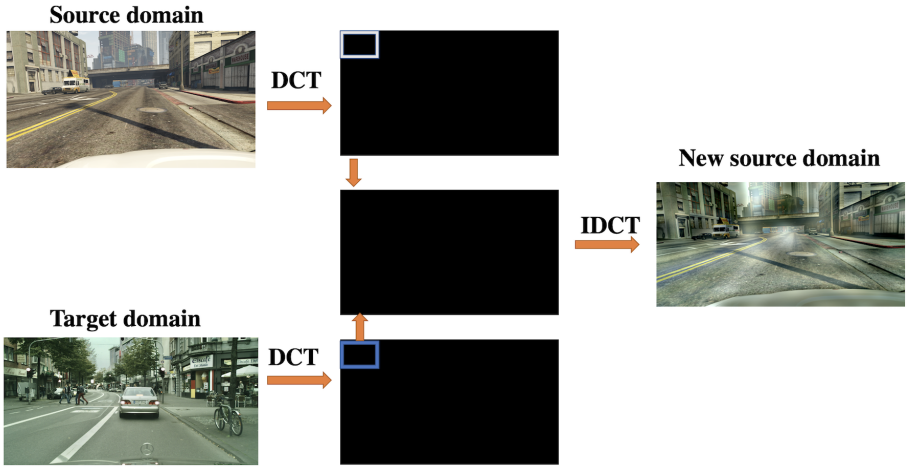

Figure 11: DCT domain adaptation workflow

3.3.4 Category-level Adversaries for Semantics Consistent Domain Adaptation

To address the limitation of the global adversarial learning, we try a category-level adversarial network (CLAN), prioritizing category-level alignment which will naturally lead to global distribution alignment. The key idea of CLAN is two-fold. First, we identify those classes whose features are already well aligned between the source and target do- mains, and protect this category-level alignment from the side effect of adversarial learning. Second, we identify the classes whose features are distributed differently between the two domains and increase the weight of the adversarial loss during training. In this process, we utilize co-training , which enables high-confidence predictions with two diverse classifiers, to predict how well each feature is semantically aligned between the source and target domains. Specifically, if the two classifiers give consistent predictions, it indicates that the feature is predictive and achieves good semantic alignment. The whole structure of CLAN is given in Figure 12.

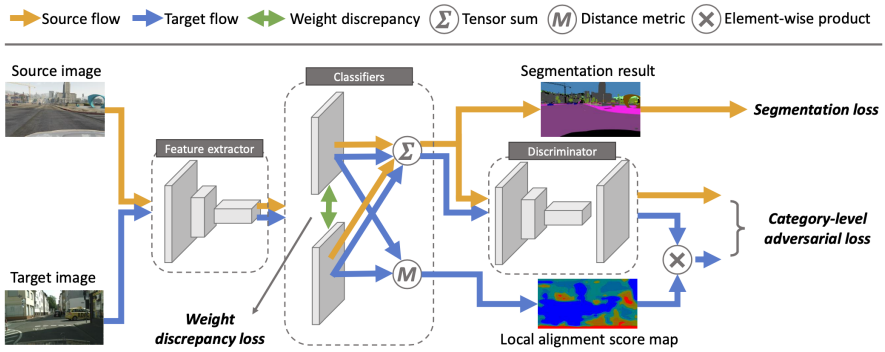

Figure 12: Overview of the category-level adversarial network.

The proposed network is featured by three loss functions (the segmentation loss, the weight discrepancy loss and the self-adaptive adversarial loss).

4 Results

4.1 Dataset

After preprocessing our database, we provide the visualization of several randomly selected pictures and their labels using python script:

The dataset we selected includes more than 2000 pictures and we manually labeled more than 1000 pictures to make the data more balanced. All pictures resized to 1280 * 720. After then, we perform data augmentation using OpenCV and the final dataset contains about 2944 pictures.

The statistical count of labels are shown in Table 1 and Figure 13

labels	pedestrians	vehicles	bicycles	signs
count	7181	17513	3274	13588

Table 1: Statistical analysis of the dataset

4.2 YOLOV3 + Darknet 53

We use the dataset to train our YOLOV3 + Darknet53 model as a bench mark. The training platform information is shown in Table 2 and Table 3

The parameters in YOLOV3 + Darknet53 model is shown in Table 4

The model mAP is 0.636 before pruning and after pruning and retraining the mAP of the model becomes 0.501. After we decrease the pruning degree

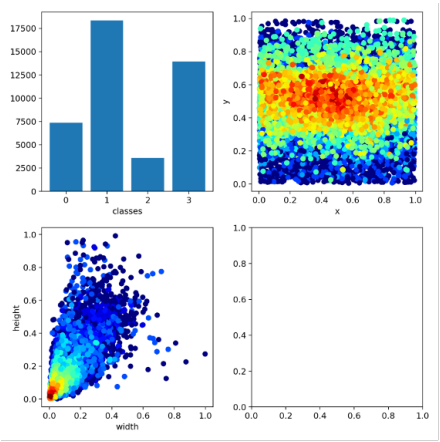

Figure 13: Distribution of labels

CPU	Intel Xeon Gold 5222 @ 3.80GHz, 4 cores, x2
GPU	NVIDIA TITAN RTX 24GB, x2
Memory	8*32 GB, DDR4 2133Mhz
Hard Disk	240GB SSD + 1.92TB NVMe

Table 2: Hardware Information

OS	Ubuntu 18.04.5 LTS
Compiler	GNU Compiler v7.5.0; CUDA Compiler v10.0
MPI	Open MPI 4.0.4
Python version	Python 3.6.9

Table 3: software Information

Parameters	Explanation	Values
batch_size_per_gpu	Train batch size(Modified according to different GPU)	32
num_epochs	Number of epochs during training	300
pth	Degree of Pruning	0.5
num_epochs	Number of epochs after Pruning(At retraining)	300
val_split	Dataset split ratio	0.2

Table 4: Parameters for YOLOV3 + Darknet53

pth, the model's output is not ideal and its size is large. One sample prediction is shown in Figure 14

The mAP for individual classes is shown in Table 5

Figure 14: Prediction of YOLOV3 + Darknet53

Model	YOLOV3+Darknet53	YOLOV3+Darknet53 Pruned
Size	236(M)	23.5(M)
epoch	300	300
Total mAP	0.636	0.501
Bicycle mAP	0.699	0.497
Pedestrian mAP	0.553	0.405
Road Sign mAP	0.505	0.398
Vehicle mAP	0.786	0.705

Table 5: Training result of YOLOV3 + Darknet53

4.3 YOLOV5

Before we balance the training data, the performance of YOLOV5s is shown in Figure 15.

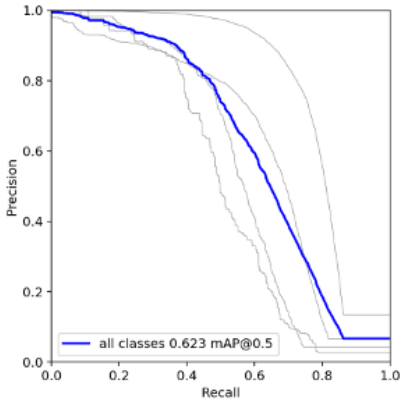

Figure 15: Prediction of YOLOV5s before dataset improvement

After we balance the training data, the performance of YOLOV5s is shown in Figure 16.

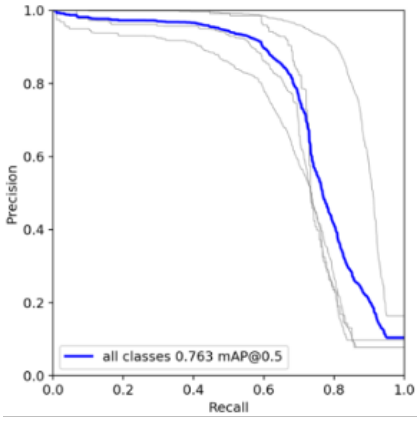

Figure 16: Prediction of YOLOV5s after dataset improvement

We want to further improve the map of the model, so we train YOVOV5l and the performance is shown in Figure 17.

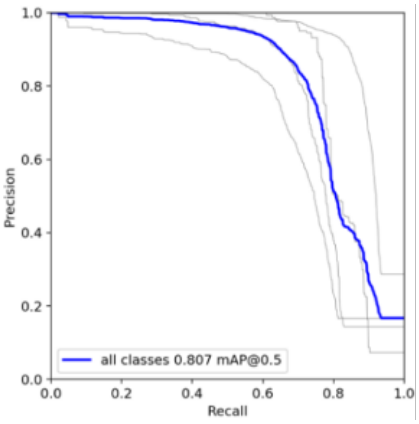

Figure 17: Prediction of YOLOV5l after dataset improvement

As shown in above figures, we have improved the map of YOLOV5 models from 0.623 to 0.763 and finally to 0.807. The gray lines in the figures are the map of four categories separately. From Figure 15, we can see that the map of vehicles is high but the map of the other three categories is very low and the total performance is not satisfactory. Therefore, we enhance the dataset, adding more bicycles and pedestrians to the dataset and we can see from Figure 16, the accuracy difference between class vehicles and other class is not that significant compared with Figure 15. After then, we use the improved dataset

to train YOLOV5l and the final map we can achieve is 0.807. It is worth noting that since our dataset is composed of three different datasets, when splitting the train, test and validation set, it is important to draw sample from all three sources according to their proportion.

Finally, we use our trained model and test it on the environment in SUSTech, the result is shown in Figure 18, Figure 19 and Figure20.

Figure 18: Prediction of YOLOV5l on Liyuan

Figure 19: Prediction of YOLOV5l on Liyuan

Figure 20: Prediction of YOLOV5l on Liyuan

4.4 Lane Detection

We use SCNN with ERFnet as its backbone and test the result on videos we take in SUSTech. Some frames of the video is shown below in Figure 21.

Figure 21: Lane detection in SUSTech

We can see from above figures that the prediction result is rather satisfying.

4.5 2D semantic segmentation

4.5.1 Dataset gained from DCT domain adaptation

First, we did frequency analysis which is used to identify main components and minor components. The sample of different frequency part of the image are given in Figure 22.

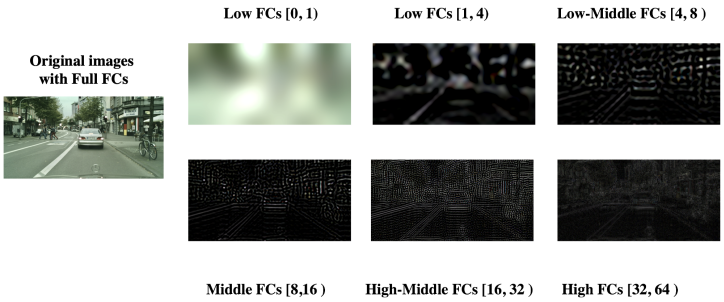

Figure 22: DCT frequency analysis with 64 FCs into 6 parts

Classification tasks are done to figure out the main components and the result are given in Table 6.

Rejected band	Source accuracy(%)	Target accuracy(%)
Null	96.2	66.7
$[0, 1)$	95.9	67.9
$[1, 4)$	95.6	66.1
$[4, 8)$	96.1	63.7
$[8, 16)$	95.8	63.2
$[16, 32)$	95.5	62.9
$[32, 64)$	96.4	66.7

Table 6: Band-reject Spectrum analysis based GTA5 as source domain and Cityscapes as target domain

Hence, we conclude that [4,32) of the frequencies are main components of the picture.

Then we remain the main components of the source domain images and add some minor components from the target domain images to gain new source

domain images. A sample of the new source domain images is given in Figure 23.

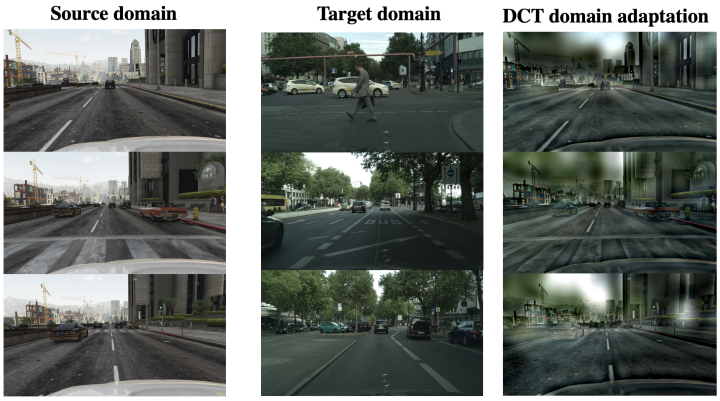

Figure 23: A sample of the new dataset

We can regard this part as a process of data augmentation. Compared with GAN model to gain new images which have target information, DCT technique is more easy and needs less GPU to manage the task which can reduce the cost of gaining images with target style.

4.5.2 Results of street scene semantic segmentation

After data processing, we applied CLAN model to manage the street scene semantic segmentation task. Here we chose GTA5 dataset to be the source domain. Meanwhile, Cityscape and Self sampled SUSTECH dataset are used as target domain to test the model.

We have done different experiments to compare the performance of different models and the results are given in Table 7.

Model	CLAN+DCT	CLAN	DeeplabV2+Resnet101
Road mIoU	0.822	0.798	0.537
Pedestrian mIoU	0.576	0.577	0.228
Sign mIoU	0.418	0.409	0.106
Bike mIoU	0.321	0.326	0.135
Car mIoU	0.733	0.731	0.401
Bus mIoU	0.366	0.338	0.099
Sky mIoU	0.739	0.717	0.466
mIoU	0.498	0.483	0.304

Table 7: Training result of CLAN+DCT, CLAN and baseline

Meanwhile, the training parameters in CLAN model are shown in Table 8.

Parameters	Explanation	Values
batch_size_per_gpu	Train batch size(Modified according to different GPU)	1
num_epochs	Number of epochs during training	100
lr	Learning rate	0.025
M	Momentum	0.9

Table 8: Parameters for DeeplabV2+ Resnet101

We can figure out that CLAN and CLAN+DCT manage the tasks well compared with the baseline. Especially DCT domain adaptation enhances the performance of the model to gain better results.

One sample prediction on Cityscapes dataset gained from different models is shown in Figure 24.

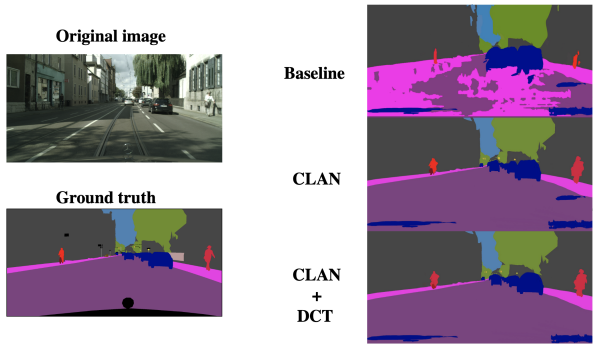

Figure 24: The result of semantic segmentation on Cityscapse dataset

Meanwhile, one sample of prediction on Self sampled SUSTECH dataset is given in Figure 26.

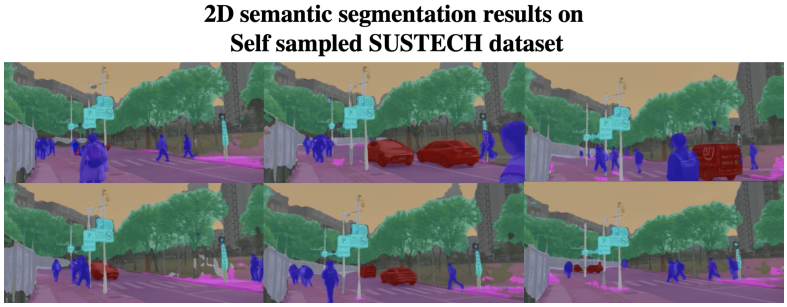

Figure 25: The result of semantic segmentation on Self sampled SUSTECH dataset

5 Staffing Plan

All the staffing information are provided in the following.

- Xintong DUAN (35%)
  - Research on Lane Detection
  - Research on YOLOV5
  - Construct Self sampled SUSTECH dataset
  - Data labeling
  - Data preprocessing for Yolo
  - Data augmentation for Lane detection
  - Reconstruct YOLOV5
  - YOLOV5 training
  - Lane detection
- Xiaoxuan WANG (35%)
  - Research on 2D semantic segmentation
  - Construct Self sampled SUSTECH dataset
  - Data labeling
  - Data preprocessing for Yolo
  - YOLOV3 training
  - Data augmentation for 2D semantic segmentation
  - 2D semantic segmentation
  - DCT Domain Adaptation
- Hanxi SUN (30%)
  - Research on DeeplabV2
  - Research on AdaptSegNet
  - Data labeling
  - Data preprocessing for Yolo
  - YOLOV3 training
  - Semantic segmentation on baseline training
  - Try AdaptSegNet training

6 Timeline

Timeline are shown in Figure 26.

Week	TASK
10	• Start training YOLOV5 • Data preprocessing for Lane detection and semantic segmentation • Research on Lane detection and semantic segmentation
11	• Gain the initial results of YOLOV5 • Collect SUSTech datasets • Train the initial results of Lane detection and semantic segmentation on datasets
12	• Improve the accuracy of YOLOV5 • Clean and label the SUSTech datasets for YOLOV5 • Collect SUSTech datasets for Lane detection and semantic segmentation
13	• Improve the time efficiency of the model YOLOV5 • Recollect SUSTech datasets for Lane detection and semantic segmentation • Get the results on SUSTech datasets
14	• Analyze the initial results of YOLOV5, Lane detection and semantic segmentation • Plot the data / visualizing the results • Improve the initial results Lane detection and semantic segmentation
15	• Analyze the final results of YOLOV5, Lane detection and semantic segmentation • Visualize the final results
16	• PPT and Report • Warp up the final results • Presentation video making

Figure 26: Timeline

References

[1] Karsten Behrendt and Ryan Soussan. Unsupervised labeled lane markers using maps. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[2] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020.

[3] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

[4] 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.

[5] Xingang Pan, Jianping Shi, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Spatial as deep: Spatial cnn for traffic scene understanding. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[6] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[7] Eduardo Romera, José M. Álvarez, Luis M. Bergasa, and Roberto Arroyo. Erfnet: Efficient residual factorized convnet for real-time semantic segmentation. *IEEE Transactions on Intelligent Transportation Systems*, 19(1):263–272, 2018.

[8] seekFire. Overview of model structure about yolov5. Accessed 10 November 2021. https://github.com/ultralytics/yolov5/issues/280, 2020.

[9] Ultralytics. Yolov5. Accessed 10 November 2021. https://github.com/ultralytics/yolov5/, 2021.

[10] Chien-Yao Wang, Hong-Yuan Mark Liao, Yueh-Hua Wu, Ping-Yang Chen, Jun-Wei Hsieh, and I-Hau Yeh. Cspnet: A new backbone that can enhance learning capability of cnn. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 390–391, 2020.

[11] Ping Luo Xiaogang Wang Xingang Pan, Jianping Shi and Xiaoou Tang. Spatial as deep: Spatial cnn for traffic scene understanding. In *AAAI Conference on Artificial Intelligence (AAAI)*, February 2018.

[12] Yanchao Yang and Stefano Soatto. Fda: Fourier domain adaptation for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4085–4095, 2020.

[13] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2636–2645, 2020.

[14] Zhe Zhu, Dun Liang, Songhai Zhang, Xiaolei Huang, Baoli Li, and Shimin Hu. Traffic-sign detection and classification in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2110–2118, 2016.

[15] Zhengxia Zou, Zhenwei Shi, Yuhong Guo, and Jieping Ye. Object detection in 20 years: A survey. *arXiv preprint arXiv:1905.05055*, 2019.