MOBIP: A LIGHTWEIGHT MODEL FOR DRIVING PERCEPTION USING MOBILENET

Minghui Ye
School of Mechanical and Electrical Engineering
Guangzhou University
minghuiye@e.gzhu.edu.cn

Jinhua Zhang
School of Mechanical and Electrical Engineering
Guangzhou University
zjhjd@gzhu.edu.cn

ABSTRACT

Visual perception is essential for autonomous driving systems. Lightweight design of the visual perception model can enhance the vehicle's ability to provide timely responses. In this paper, we propose a lightweight multi-task model to simultaneously perform traffic object detection, drivable area segmentation, and lane line detection. The model consists of a shared encoder for feature extraction and two decoders to handle three visual perception tasks. By using MobileNetV2 as the backbone, our model significantly reduces the amount of computation, thereby facilitating rapid inference speed. Specifically, the model achieves 56 FPS on NVIDIA Tesla V100 and 4.8 FPS on Raspberry Pi 4B. The model not only excels in terms of inference speed but also achieves competitive performance compared to other multi-task models across all three visual perception tasks. Furthermore, the effectiveness of our lightweight design choices are verified via ablative studies.

Keywords Self-driving · Multi-task learning · Semantic segmentation · Traffic object detection

1 Introduction

Visual perception model is the core module of autonomous driving, providing the basis for trajectory planning and vehicle control. Equipment such as cameras and radars are used to build the perception hardware system of self-driving cars. Since cameras are cheaper than radars, they are widely used in autonomous vehicles, bring the requirement of visual perception algorithm to deal with the images. Visual perception models can process the captured images and provide vehicles with information about the surrounding traffic environment. Specifically, the images captured by the camera can be used to perform traffic object detection, which provides information such as the exact position and size of surrounding obstacles. In addition, semantic analysis of drivable areas and lane lines from images is the basis for path planning and they can ensure vehicles to comply with traffic rules. Due to the limitation of the computation capability of intelligent vehicles, the inference speed and performance of the visual perception model usually need to be balanced. Therefore, the goal of the visual perception model is to achieve the performance required for safe driving and minimize the inference latency.

Many works have been done on single visual tasks of driving perception. YOLO family [Bochkovskiy et al., 2020, Redmon and Farhadi, 2017, 2018], Faster R-CNN [Ren et al., 2015] can be used for traffic target detection tasks. PSPNet [Zhao et al., 2017] and UNet [Ronneberger et al., 2015] can be used to perform drivable area segmentation and lane line segmentation tasks. These works have achieved excellent performance on single tasks. In the context of autonomous driving, the driving perception system needs to perform three tasks simultaneously to provide sufficient information while maintaining real-time inference speed. The multi-task approach can be used for driving perception. These methods build a model to perform multiple tasks at the same time, which can avoid the need of running different models to complete corresponding tasks, achieving higher computational efficiency. Many multi-task models have been proposed, YOLOP [Wu et al., 2022], HybridNets [Vu et al., 2022], and YOLOPV2 [Han et al., 2022] perform the three visual perception tasks simultaneously using a single encoder and multiple decoder heads. However, these models are computationally intensive and rely on dedicated GPU devices to achieve real-time inference. The main objective of this research is to investigate multi-task model that can perform driving perception with greatly re-

duced computation, so that the model can be deployed to much more edge devices with lower computation capability, including CPU-based embeded computing device.

We propose a novel multi-task model that can efficiently handle the tasks of traffic object detection, drivable area segmentation, and lane line segmentation simultaneously. This model contains an encoder and two decoders. The encoder contains the lightweight CNN, MobilnetV2 [Sandler et al., 2018], and a feature pyramid network [Lin et al., 2017a] as the neck. Different from the previous design of using corresponding decoders for three tasks, our model involves two decoders for object detection and semantic segmentation respectively in order to save computation. Specifically, the traffic object detection is completed by the object detection head. Drivable area segmentation, lane line segmentation are combined as a multi-class segmentation task which is completed by the semantic segmentation head. The Multi-Adds of our model is only 355.6M under the input image size of 640x640. Our model achieves an inference speed of 56FPS on the Nvidia Tesla V100, and the inference speed of 4.8FPS on an edge CPU device (Raspberry Pi 4b).

In summary, the main contributions of this work are: (1) We propose a novel lightweight multi-task architectures that can simultaneously handle three visual tasks for autonomous driving perception with significant low computation and high inference speed. (2) Compatible loss functions and data augmentation methods are designed for the proposed model, and their effectiveness is verified by ablation experiments. (3) We validate the performance of the driving perception model on the BDD100K dataset and deploy the model in edge CPU computing devices.

2 Related Work

The object detection models can be divided into two categories, one is two-stage object detection, and the other is one-stage object detection. The two-stage object detection models include Faster R-CNN [Ren et al., 2015], Fast R-CNN [Girshick, 2015] and R-CNN [Girshick et al., 2014]. They usually have a region proposal mechanism, and the features in the proposed region are used predict the location and the category of the object. By comparison, two-stage object detection methods are slower than one-stage detectors. The one-stage detector, YOLO [Redmon et al., 2016], is widely used. In the region proposal, it divides the picture into multiple regions with grids. Instead of generating anchor on each pixel, it generates multiple proposals in each region and predicts the corresponding category, which significantly decrease the amount of computation. YOLO4 [Bochkovskiy et al., 2020] and yolov7 [Wang et al., 2023] improved the YOLO architecture, they mainly combine the latest model designing tricks, activation functions, and loss functions to improve the detection accuracy.

Many models based on convolutional neural networks have achieved good results in semantic segmentation tasks. These methods can be used for drivable area segmentation and lane line detection tasks. FCN [Long et al., 2015] introduced the full convolutional network into the semantic segmentation task. They discarded the fully connected layer at the end of VGG and replaced it with a CNN module that can perform dense category prediction, but it is still limited to low-resolution segmentation. U-Net [Ronneberger et al., 2015] uses a U-shaped encoder-decoder structure. The encoder involves convolution and downsampling, and the decoder is composed of upsampling and convolution blocks. It combines features from different layers by concat rather than add. PSPNet [Zhao et al., 2017] propose to extract features in multiple scales through the pyramid pooling module, and then use the merged features for semantic segmentation. In the Enet [Paszke et al., 2016], the inference speed is improved by reducing the size of the feature map.

Multi-task learning is to train vision models that perform multiple perception tasks simultaneously. The MultiNet [Teichmann et al., 2018] architecture uses an encoder to extract features, and the extracted features are used for scene classification, object detection and segmentation of the driving area with three decoders. YOLOP [Wu et al., 2022] can simultaneously performs three perception tasks (object detection, lane line detection, drivable area segmentation). This model is based on YOLOv5s and add two segmentation branches to perform two segmentation tasks respectively, achieving high inference speed and enabling real-time inference in Tesla V100 GPU. HybridNets [Vu et al., 2022] chooses EfficientDet as the backbone and uses Bifpn as the neck, which improves the model performance, but reduces the inference speed.

3 Methodology

We present a lightweight multi-task learning framework designed for driving perception. As shown in Figure 1, our driving perception model, termed as Mobip, comprises a shared encoder and two decoders. The shared encoder efficiently extracts features, thereby conserving computational resources and facilitating multi-task learning. The decoders consist of two heads: the detect head for traffic object detection, and the segment head for drivable area segmentation and lane detection.

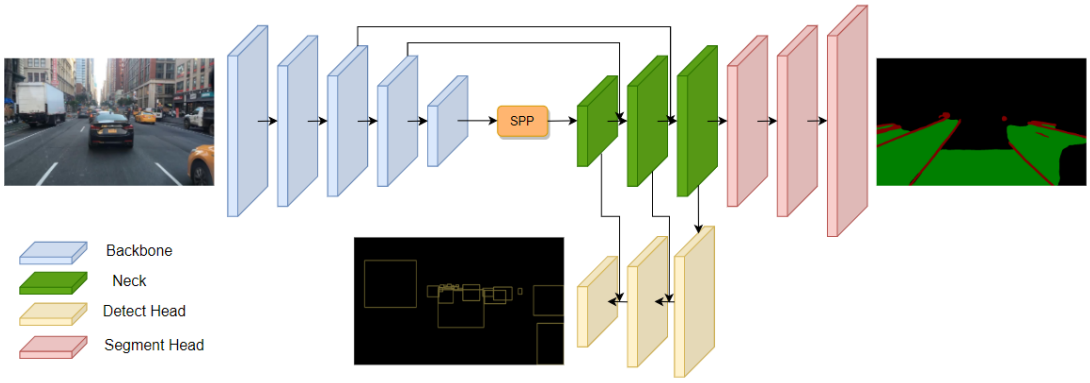


Figure 1: The architecture of Mobip. Mobip contains one shared encoder for feature extraction and two decoders for object detection and semantic segmentation.

3.1 Encoder

The shared encoder comprises two components, namely a backbone network and a neck network.

3.1.1 Backbone

The backbone network serves a critical role in the driving perception model as a feature extractor. Contemporary network architectures often leverage networks that have exhibited high accuracy on the ImageNet dataset for feature extraction purposes. Given the efficient nature of the MobileNetV2 [Sandler et al., 2018] network and its exceptional performance in object detection and semantic segmentation, we specifically select it as the backbone network for our model implementation. Moreover, the utilization of inverted residual as the basic building block enhances our model's efficiency.

3.1.2 Neck

The features extracted by the backbone are fused through the Neck network. The Neck network consists of two modules, including Spatial Pyramid Pooling(SPP) [He et al., 2015] for fusing features of different scales and Feature Pyramid Network(FPN) [Lin et al., 2017a] for fusing features at different semantic layers. Within the FPN module, high-level semantic features are concatenated with low-level features after bilinear interpolation to facilitate aggregation.

3.2 Decoders

The task of driving perception is divided into either a detection task or a segmentation task, subsequently performed by the corresponding task head.

3.2.1 Detect Head

Similar to YOLOv4, our approach employs an anchor-based multi-scale detection scheme. Multiple feature layers within the Feature Pyramid Network (FPN) are passed to the detect head for further feature aggregation. First, the Path Aggregation Network (PAN) [Liu et al., 2018] is utilized to perform bottom-up feature fusion, enabling better localization feature extraction. The aggregated multi-scale feature from PAN is then used for traffic object detection. Within the multi-scale feature map, every grid is assigned three anchors with varying aspect ratios. The detect head then generates predictions for the offset of position, scaled height and width, as well as the corresponding probabilities and confidences for each class prediction.

3.2.2 Segment Head

Different from previous methods that handle drivable area segmentation and lane detection separately with two segmentation heads, we adopt a single segmentation branch for multi-class segmentation, which effectively reduces computational redundancy and achieves faster inference speed. The segmentation head is connected to the bottom layer

of FPN and up-samples the feature maps for three times with bilinear interpolation method. This process restores the output feature map to the size of original image, generating the semantic segmentation result.

3.3 Loss Function

In our multi-task learning approach, the two loss specifically for detect head and segment head are calculated with weights and added together as a final loss as shown in Equation (1).

$$\mathcal{L}_{all} = \gamma_1 \mathcal{L}_{det} + \gamma_2 \mathcal{L}_{seg}, \tag{1}$$

where $\mathcal{L}_{det}$ is the detection loss and $\mathcal{L}_{seg}$ is the segment loss. The detection loss $\mathcal{L}_{det}$ is a weighted sum of classification loss, object loss and bounding box loss, as shown in Equation (2).

$$\mathcal{L}_{det} = \alpha_1 \mathcal{L}_{class} + \alpha_2 \mathcal{L}_{obj} + \alpha_3 \mathcal{L}_{box}, \tag{2}$$

where $\mathcal{L}_{class}$ is classification loss while $\mathcal{L}_{obj}$ is the loss for the confidence of one prediction. $\mathcal{L}_{class}$ and $\mathcal{L}_{obj}$ are calculated with focal loss [Lin et al., 2017b] to force the model to learn the hard example. $\mathcal{L}_{box}$ is $CIoU$ loss [Zheng et al., 2020] which measures the distance of overlap rate, aspect ratio and scale similarity between predicted results and ground truth. For the segment loss $\mathcal{L}_{seg}$, we deploy a hybrid loss including focal loss and dice loss, as shown in Equation (3).

$$\mathcal{L} = \mathcal{L}_{Dice} + \beta \mathcal{L}_{Focal}, \tag{3}$$

The dice loss in the segment head can mitigate the data-imbalance problem since there is a multi-class segmentation where the data of lane line is way less than other segment class. The focal loss can help the model to learn the pixels with poor classification. The values of $\gamma_1, \gamma_2, \gamma_3, \alpha_1, \alpha_2, \beta$ are tuned for the balance of the loss.

4 Experiments

4.1 Dataset and Experimental Setting

The BDD100K [Yu et al., 2020] dataset contains a variety of driving scenes, in which a large number of data are collected from driving recorders, capturing more "long tail" driving scenes in different environments. The dataset includes various annotations, including drivable areas, object detection, attributes, road types, and lane lanes etc, which can support scientific research on a range of driving perception tasks. Therefore, we validated the effectiveness of the proposed driving perception model by comparing it with state-of-the-art methods on the BDD100K dataset. The BDD100K dataset consists of images at 1280×720 resolution, with a total of 100K images, divided into three splits: 70K for training set, 10K for validation set, and 20K for test set.

At the training stage, we use the Adam optimizer, with learning rates, β_1 and β_2 set to 1×10^{-2}, 0.937 and 0.999, respectively. Cosine annealing and warm-up are applied to adjust the learning rate [Loshchilov and Hutter, 2016]. Data augmentation techniques, including mirror, translation, shearing, rotation, photometric distortion, Mosaic and Mixup, is used to boost the performance. Following [Hou et al., 2019], We resize the input image from $1280 \times 720 \times 3$ to $640 \times 384 \times 3$. All modules are implemented using the PyTorch framework [Paszke et al., 2019], and all experiments were run on NVIDIA Tesla V100.

4.2 Result

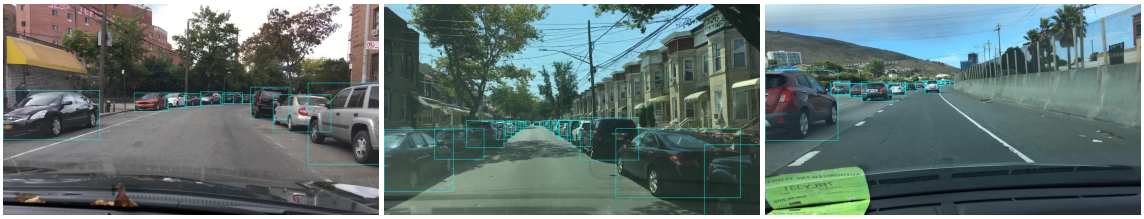
In this section, We compare Mobip to other representative models on all three tasks. First, we compare Mobip with other multi-tasking methods in terms of parameters, number of computations and inference speed in GPU (NVIDIA Tesla V100). We then evaluate the performance of Mobip on three driving perception task, comparing it with multi-task methods and networks that focus on the single task. Finally, the model is deployed to embedded devices(Raspberry Pi 4B) to test inference performance.

4.2.1 Model Parameter and Inference Speed

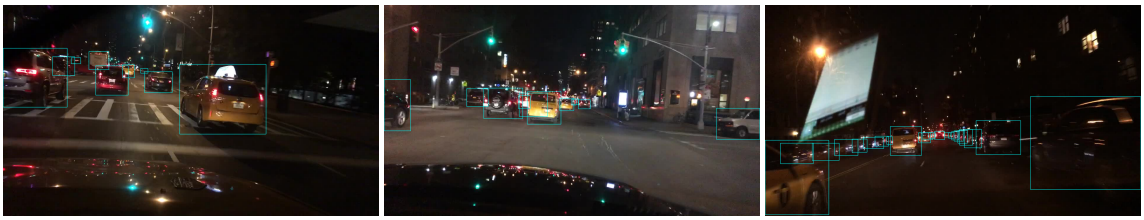
Table 1 presents the comparison of Mobip, YOLOP and HybridNets in term of parameters, computations and inference speed. By adopting MobileNet as the backbone network and the inverted residual as the building block for the driving perception model, Mobip have a considerable advantage in the amount of calculation (355.6M), achieving the fastest inference speed (56 FPS) in the TESLA V100 GPU. Notably, the performance comparison between Mobip and HybridNets in each driving perception task is not provided in the following passage, as the primary focus of HybridNets lies in enhancing perception performance, rather than emphasizing lightweight model design and inference speed improvement.

Table 1: Computational cost for various multi-task models and inference speed on Nvidia Tesla V100

Network	Input Shape	Params	Multi-adds	Speed(fps)
YOLOP	640x640	7.9M	9.3G	40
HybridNets	640x640	12.8M	7.8G	27
Mobip(Ours)	640x640	10.5M	355.6M	56



(a) Day-time result



(b) Night-time result

Figure 2: Visualization of traffic object detection results of Mobip. (a) Results in day conditions and (b) Results in night conditions

4.2.2 Traffic Object Detection Result

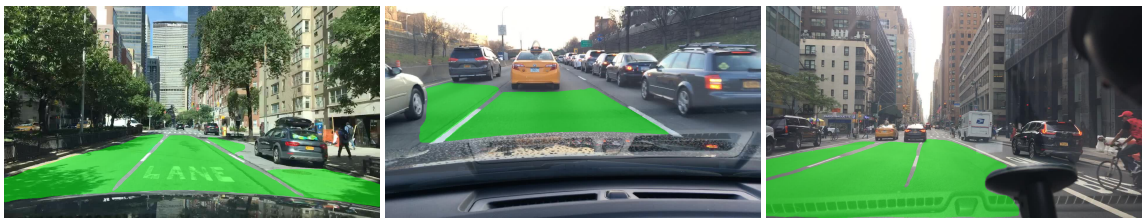
The quantitative comparison in the traffic object detection task are shown in Table 2. We compared three multi-task models, including MultiNet, DLT-Net, and YOLOP, and two single-task object detection models, including Faster R-CNN and YOLOv5s. Mobip achieved the best detection result (89.6) on the Recall metric, which is higher than the best multi-task model YOLOP (89.2) and the detection model YOLOv5s (86.8). However, due to the limitations of the MobileNet backbone model, Mobip (75.4) is slightly lower than YOLOP (76.5) and YOLOv5s (77.2) on the mAP50 indicator. The advantage of Mobip is mainly reflected in the inference speed, which is 40% faster than YOLOP. The reason why YOLOv5s achieves higher speed is that it does not need to perform drivable area segmentation and lane line detection tasks. We visualize the traffic object detection results of Mobip, as shown in Figure 2. Mobip works well in both day and night conditions, and it can even detect vehicles that are difficult to recognize by human in the night examples. It can also be seen from the figure that the model perform well in the images with dense and small objects. Obvious detection failure was not found in the examples.

4.2.3 Drivable Area Segmentation Result

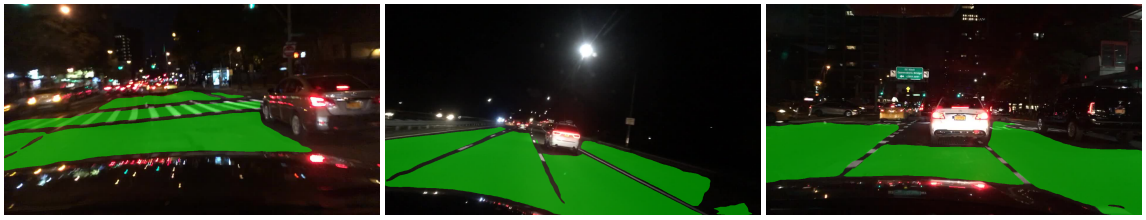
As shown in Table 3, we compare our model with one classic single-task segmentation model, PSPNet, and three multi-task models including MultiNet, PSPNet, and DLT-Net. Mobip achieved the result of 90.0% on the mIoU

Table 2: Results on traffic object detection.

Network	Recall(%)	mAP50(%)	Speed(fps)
Faster R-CNN	81.2	64.9	8.8
YOLOv5s	86.8	77.2	82
MultiNet	81.3	60.2	8.6
DLT-Net	89.4	68.4	9.3
YOLOP	89.2	76.5	40
Mobip(Ours)	89.6	75.4	56



(a) Day-time result



(b) Night-time result

Figure 3: Visualization of drivable area segmentation results of Mobip. (a) Results in day conditions and (b) Results in night conditions

Table 3: Results on drivable area segmentation.

Network	**mIoU(%)**	**Speed(fps)**
PSPNet	89.6	11.1
MultiNet	71.6	8.6
DLT-Net	71.3	9.3
YOLOP	91.5	40
Mobip(Ours)	90.0	56

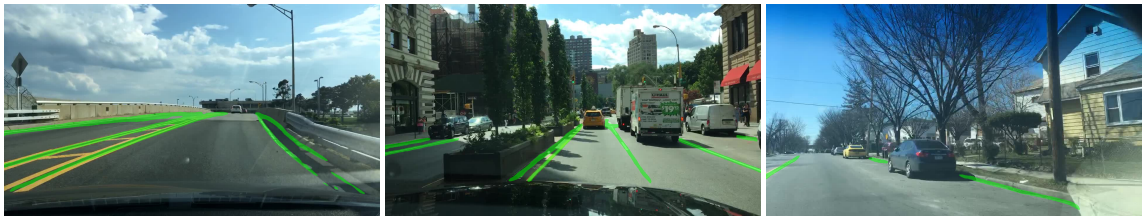
metric, outperforming PSPNet in both inference speed and performance. Mobip is better than MultiNet, PSPNet and DLT-Net, but 1.5% lower than YOLOP. The performance of this task may be limited by the feature extraction capability of MobileNet or the multi-class segmentation branch, which will be verified in the following Ablation Studies. The visualization result of the drivable area segmentation is shown in the Figure 3. Mobip performs excellent in this task, demonstrating strong capability in semantic reasoning. The examples show that it can judge whether it is a drivable area based on the cars and lane lines in the surrounding.

4.2.4 Lane Detection Result

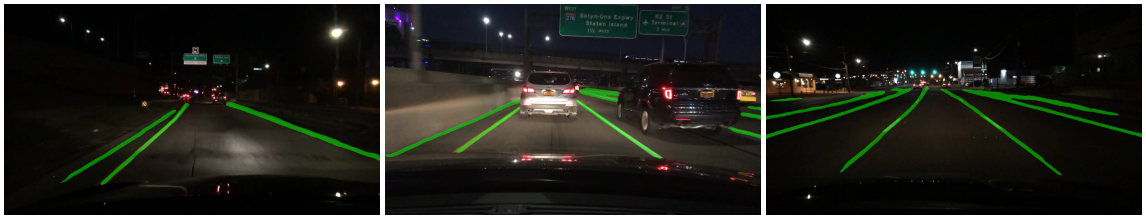
We used IoU and accuracy metrics to measure the effectiveness of the model in lane line detection. The result is shown in Table 4. Mobip shows the advantages in performance and speed over previous models. In the visualization example as shown in Figure 4, it can be seen that the predicted lane line area is wider than the real lane line, which is consistent with (28.5%) IoU metric. The advantage of Mobip is that the accuracy of lane line detection is high. Even if the IoU metric is still low, this does not cause much negative impact in the deployment.

Table 4: Results on lane line segmentation.

Network	**Accuracy(%)**	**IoU(%)**	**Speed(fps)**
ENet	34.12	14.64	100
SCNN	35.79	15.84	19.8
ENet-SAD	36.56	16.02	50.6
YOLOP	70.5	26.20	40
Mobip(Ours)	88.9	28.5	56



(a) Day-time result



(b) Night-time result

Figure 4: Visualization of lane line segmentation results of Mobip. (a) Results in day conditions and (b) Results in night conditions

Table 5: Ablation studies that evaluate the effect of decoder, backbone and data augmentation.

Training Method	Recall(%)	AP	mIoU	Accuracy	IoU	Speed(fps)	Params	Mult-adds
YOLOP	89.2	**76.5**	**91.5**	70.5	26.2	40	7.9M	9.28G
Multi-Class Seg	89.3	76.1	88.9	88.2	27.3	49	7.6M	6.28G
Backbone	87.7	73.2	89.3	88.6	26.4	56	10.5M	355.6M
Mosaic + Mixup	**89.6**	75.4	90.0	**88.9**	**28.5**	**56**	10.5M	**355.6 M**

4.2.5 Ablation Study

In this section, we verify the effectiveness of our model design choice. Compared with the previous multi-task model YOLOP, our improvements mainly lie in the backbone network, decoder, and data augmentation methods. Therefore, we quantificat the effects of the three designs separately through ablation experiments. The quantitative results are shown in Table 5. After combining the two segmentation tasks, including drivable area segmentation and lane line segmentation, to multi-class segmentation, the inference speed increased by 22.5%. The performance of object detection is basically unchanged, and the effect of lane line detection is also improved, while the performance of drivable area segmentation drops by 2.6% (mIoU metric). We achieve a substantial increase in the inference speed with little performance cost. After replacing the backbone with MobileNet and the building blocks with inverted residuals, the inference speed of the model increases by 14%. The performance on object detection greatly drops: the recall metric drops by 1.6%, and the AP drops by 2.9%. But the performance in two segmentation task is basically unchanged. This shows that the backbone only affects the performance of traffic object detection, but exerts no effect on the semantic segmentation tasks. This shows that the reason why the drivable area segmentation result in section 4.2.3 is worse than that of YOLOP comes from the design of the segmentation head. The data augmentation methods, Mosaic and Mixup, shows great effect on improving the performance of the model in both object detection and semantic segmentation tasks, which can compensate the performance loss caused by the lightweight designs.

4.2.6 Inference on Embeded Device

The trained model is further deployed in edge computing devices. In this study, the Raspberry Pi 4B was selected for testing because it is a widely used for embedded application development, especially in mobile robots. The trained model uses the quantization method (i.e. Static Quantization) for model compression, which can convert the model from Float32 to Int8 precision, which greatly reduces the required memory bandwidth and improves inference speed, making it more suitable for deployment on edge CPU computing devices. Finally, the model achieved the inference speed of 4.8 FPS (208 ms/frame) with the input image size of 320x192.

5 Conclusion

In this paper, we propose a novel multi-task model architecture, Mobip, which can simultaneously perform three driving perception tasks including traffic object detection, drivable area segmentation and lane detection. Our model has great advantages in the inference speed and the amount of computation, while delivering exceptional performance on all three driving perception tasks on the BDD100K dataset. The efficiency of our lightweight design has also been demonstrated by the fast inference speed achieved on edge CPU devices. Furthermore, we quantify the effect of the lightweight design in both the backbone and decoder components by ablation experiments. However, it is noteworthy that our current model deployment is restricted to embedded devices with limited computational capacity, which imposes constraints on the inference speed. Therefore, we will evaluate our model on devices with higher computing capacity, and incorporate model scaling methods to improve the performance on the perception tasks in the future work.

References

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

Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.

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

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

Dong Wu, Man-Wen Liao, Wei-Tian Zhang, Xing-Gang Wang, Xiang Bai, Wen-Qing Cheng, and Wen-Yu Liu. Yolop: You only look once for panoptic driving perception. *Machine Intelligence Research*, pages 1–13, 2022.

Dat Vu, Bao Ngo, and Hung Phan. Hybridnets: End-to-end perception network. *arXiv preprint arXiv:2203.09035*, 2022.

Cheng Han, Qichao Zhao, Shuyi Zhang, Yinzi Chen, Zhenlin Zhang, and Jinwei Yuan. Yolopv2: Better, faster, stronger for panoptic driving perception. *arXiv preprint arXiv:2208.11434*, 2022.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7464–7475, 2023.

Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

Adam Paszke, Abhishek Chaurasia, Sangpil Kim, and Eugenio Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation. *arXiv preprint arXiv:1606.02147*, 2016.

Marvin Teichmann, Michael Weber, Marius Zoellner, Roberto Cipolla, and Raquel Urtasun. Multinet: Real-time joint semantic reasoning for autonomous driving. In *2018 IEEE intelligent vehicles symposium (IV)*, pages 1013–1020. IEEE, 2018.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1904–1916, 2015.

Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017a.

Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8759–8768, 2018.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017b.

Zhaohui Zheng, Ping Wang, Wei Liu, Jinze Li, Rongguang Ye, and Dongwei Ren. Distance-iou loss: Faster and better learning for bounding box regression. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 12993–13000, 2020.

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.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

Yuenan Hou, Zheng Ma, Chunxiao Liu, and Chen Change Loy. Learning lightweight lane detection cnns by self attention distillation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1013–1021, 2019.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.