MULTI TASK LEARNING OF DEEP NEURAL NETWORKS IN VEHICLE PERCEPTION

Koundinya Durbha
Department of Automotive Engineering
Clemson University
Greenville, SC 29607
koundid@clemson.edu

Priyanshu Rawat
Department of Automotive Engineering
Clemson University
Greenville, SC 29607
prawat@clemson.edu

Vishal Jadhav
Department of Automotive Engineering
Clemson University
Greenville, SC 29607
vjadhav@clemson.edu

May 7, 2022

ABSTRACT

An autonomous driving system often contains multiple neural networks working together to predict bounding boxes, segmentation maps, depth maps, lane lines, etc. A separate neural network for each task significantly impacts the system's processing speed. In this project, we tried implementing a multi-task learning-based neural network for autonomous vehicles and mobile robot applications. The attempt is to implement an architecture that performs multiple tasks simultaneously and uses asymmetric datasets with uneven numbers of annotations per modality. We showcase how our model can perform two tasks on two datasets, all at once, performing depth estimation and segmentation with a single model. This sort of network is essential in autonomous vehicles.

1 Introduction

As the deep learning model shows impressive results in performing a number of tasks in an autonomous vehicle, the number of models that achieve such results keeps increasing. It makes it hard to deploy such a large number of deep learning models to perform multiple tasks due to limited computation resources. Some tasks besides structural similarity tend to share the same dataset. For example, a deep learning model to learn different tasks like image classification, object detection, and semantic segmentation shares the same dataset. But tasks like semantic segmentation and depth estimation rarely share the same dataset. We used the NYUD and KITTI datasets, asymmetric datasets with semantic segmentation and depth estimation annotation. We labeled missing data in those datasets using the 'Teacher network.' We implemented a multi-task learning-based neural network presented in [1] on our dataset. The attempt is to implement an architecture that has an encoder-decoder structure. It takes an RGB image as an input and predicts a segmentation mask and a depth map in a single forward pass. The idea is to have a common backbone for extracting feature maps. Then according to the required task, the decoder structure is plugged onto this encoder to generate predictions. This sort of network is essential for Autonomous Driving. We performed two tasks, depth estimation and semantic segmentation, with few architectural changes and no complicated pipelines.

2 Network Architecture

A. Backbone Network Our model is an encoder-decoder network which has two outputs- one for semantic segmentation and the other for depth estimation. We used Light-Weight Refinenet architecture built on top of the

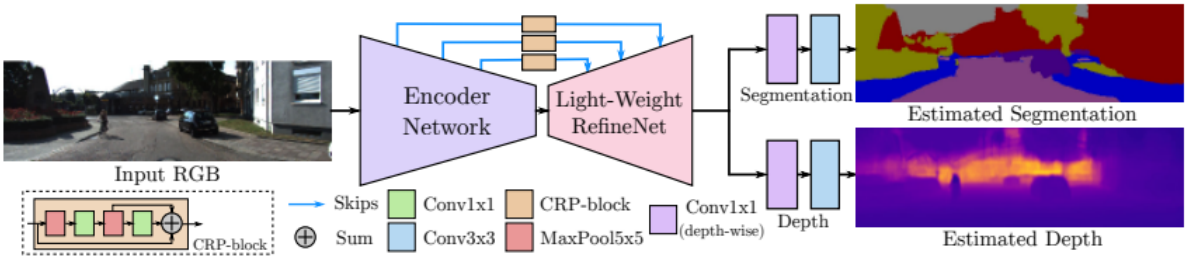

Figure 1: Model inference on custom indoor image.

MobileNet-v2 classification network. This architecture extends the classifier by appending several simple contextual blocks, called Chained Residual Pooling (CRP), consisting of a series of 5 × 5 max-pooling and 1 × 1 convolutions. We take reference from [1] to replace 1 X 1 convolution in the last CRP block with its depth-wise equivalent.

Used recently proposed Light-Weight Refinenet [2] on top of MobileNet-v2 [4] as our baseline architecture. Light-Weight refinenet is a more compact, powerful semantic segmentation architecture based on the RefineNet [3]. It is suitable for tasks requiring real-time performance on high-resolution inputs. MobileNet-v2 is an encoder in our model, and Multi-task light weight refinenet is a decoder.

B. Joint Semantic Segmentation and Depth Estimation It is important to branch out the backbone network into separate task-specific path in order to achieve optimal performance on both task simultaneously. Branch out right after last CRP block, and append two additional convolution layers (one depth-wise 1 X 1 and one plain 3 X 3) for each task.

We take loss function from the base paper [1]. Denoting the output of the network before the branching as $\tilde{y} = f_{\theta b}(I)$, where $f_{\theta b}$ is the backbone network with a set of parameters θ_b, and I is the input RGB image, then depth and segmentation prediction can be given as $\tilde{y}s = g\theta s(\tilde{y})$ and $\tilde{y}d = g\theta_d(\tilde{y})$, where $g_{\theta s}$ and $g_{\theta d}$ are segmentation and depth estimation parameters. Used the standard softmax cross-entropy loss for segmentation and and inverse Huber loss for depth estimation. Total loss contains an additional scaling parameter, , which, we set to 0.5:

$$L_{total}(I, G_S, G_d; \theta_b, \theta_s, \theta_d) = (\lambda.L_{segm}(I, G_S; \theta_b, \theta_S) + (1-)L_{depth}(I, G_d; \theta_b, \theta_d)) \tag{1}$$

$$L_{segm}(I, G) = \frac{-1}{|1|} \sum_{i \epsilon I} \log(softmax(\tilde{y}_s) i G_i) \tag{2}$$

$$L_{depth}(I, G) = \{ \ (|(\tilde{y}_d - G|, \mathrm{if}|(\tilde{y}_d - G| \leq c((\tilde{y}_d - G)^2 + c^2)/(2c), otherwise \tag{3}$$

$$c = 0.2.max|(\tilde{y}_d) - G|, \tag{4}$$

Where; Gs and Gd denote ground truth segmentation mask and depth map, correspondingly; (.)ij in the segmentation loss in the probability value of class j at pixel i.
For evaluating the multi-task learning model, we used mean IoU for semantic segmentation and RMSE for depth estimation.

C. Expert Labeling for Asymmetric Annotations It is impossible to have all the ground truth sensory information available for each image in the dataset. So first, we run the baseline model on the NYUD dataset. We had a highly asymmetric dataset if we wanted to run that model on the KITTI dataset. KITTI dataset with 400 segmentation annotated images but without depth estimation annotation. Then, in that case, we took the teacher network, which is the pre-trained neural network on the KITTI dataset, for depth estimation. Using this model on KITTI dataset images with only semantic segmentation annotation, we generated ground truth for depth estimation. And these depth-estimated images are used for multi-task learning.

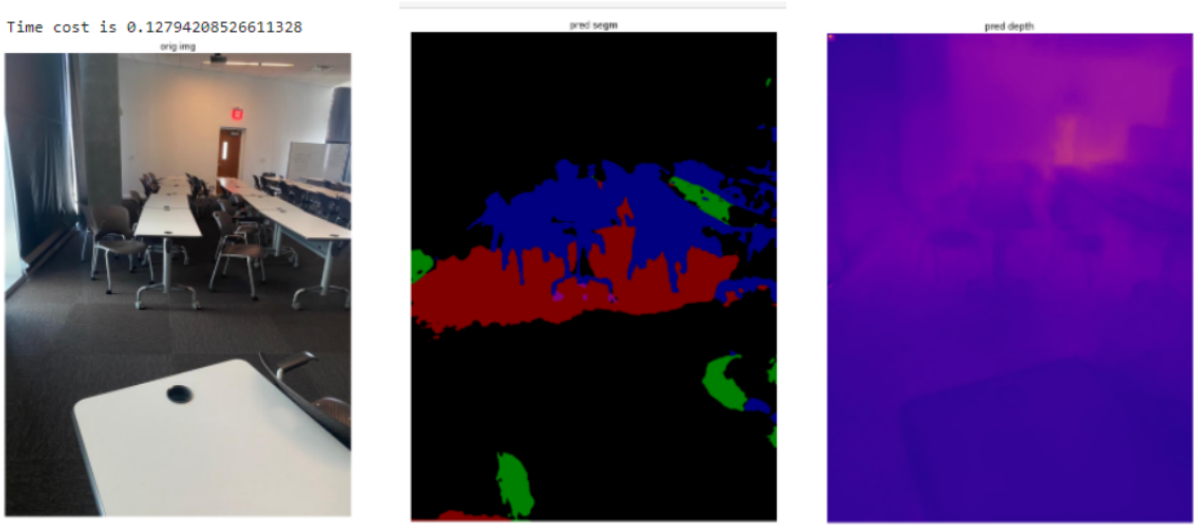

Figure 2: Model inference on custom indoor image.

3 Experiments

We trained this model on two different datasets, NYUD for indoor scenarios and KITTI dataset for vehicle perception tasks. We used an Nvidia Tesla V100 (16 GB memory), PyTorch 1.11, CUDA 11.1 and cuDNN 8.1 for training the model. The encoder models are all initialized on weights trained on ImageNet dataset. Input images are normalized, data augmentation techniques like random crop and random mirror are performed.

3.1 Training

The model was trained on NYUD dataset for indoor scenarios and KITTI dataset for vehicle perception and outdoor scenarios. We used library aims to ease typical workflows involving dense per-pixel tasks in PyTorch. The progress in such tasks as semantic image segmentation and depth estimation have been significant over the last years, and in this library we provide an easy-to-setup environment for experimenting with given (or your own) models that reliably solve these tasks.

3.1.1 NYUD Dataset

NYUD is an indoor dataset for semantic segmentation and depth estimation. It has 40 classes for semantic segmentation and 1449 images with aligned semantic and depth estimation ground truths. 795 of these were taken for training and 695 for validation. Since the NYUD dataset already has both semantic and depth estimation ground truths for the same RGB images, the data could be directly fed into the model with little pre-processing required.

Random crop and random mirror operations were performed on the dataset to improve generalization and the resulting image was given as input to the model. We chose mean IoU as an evaluation metric for semantic segmentation and RMSE loss for depth estimation. We experimented on batch size, encoder and decoder learning rates, and weight decays for encoder and decoder to fine tune our model. After multiple iterations, we achieved a mean IoU of 0.2927 for batch size of 3, encoder learning rate of 1e-3, encoder decay of 1e-4,decoder learning rate of 1e-3 and decoder decay of 1e-5 when trained for 138 epochs. Stochastic Gradient Descent (SGD) was used as optimizer for the model.

After tuning, we used the final model to perform inferencing on our own images and note the time computation for inference. The semantic segmentation was not satisfactory given the low IoU values of 0.29, but the depth estimation was accurate.

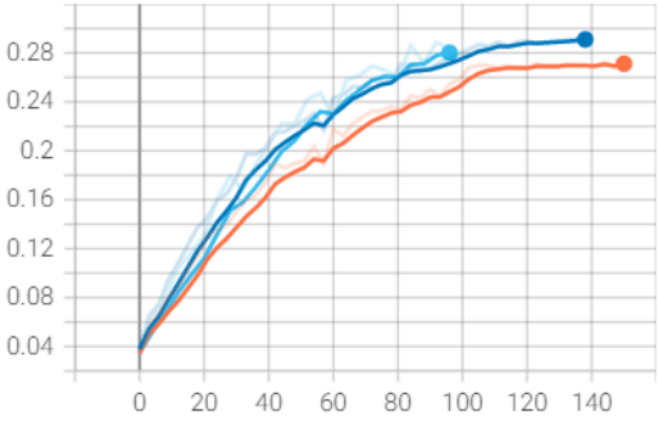

Figure 3: Mean IoU of model on NYU Dataset.

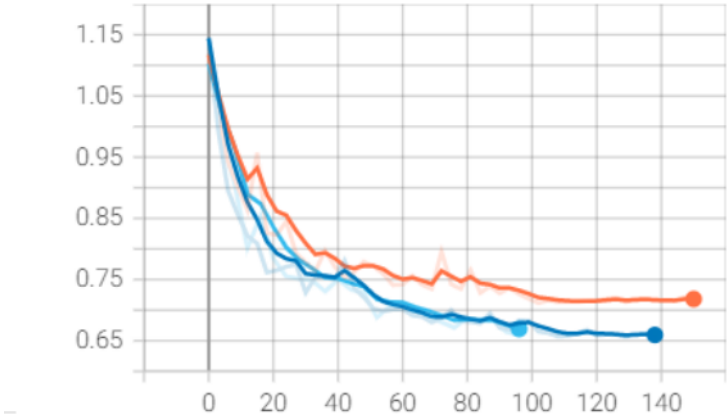

Figure 4: RMSE Loss of model on NYU Dataset.

3.1.2 KITTI Dataset

The KITTI semantic dataset has 200 training and 200 test images. However, the corresponding depth annotations were not available for this dataset. In order to obtain the depth annotations for these images, we used another neural network trained for depth estimation on the KITTI depth estimation dataset, and used it to predict depth images for the RGB images provided in our semantic dataset. The obtained inference results were used as ground truth for our multi task model. While it does not guarantee perfect annotations, this would be the best alternative to deal with asymmetric annotations, as in this case.

Random crop and random mirror operations were performed on the input images and the resulting images were sent into the encoder. The evaluation metrics were mean IoU for semantic segmentation and RMSE for depth estimation.

After making multiple changes to batch sizes, encoder and decoder learning rates, and decay rates, we achieved an IoU of 0.161 and an RMSE loss of 0.07. Small dataset size could be a reason for the low IoU obtained. We performed inferencing on KITTI test data as well as our own inputs and observed that segmentation and depth both were poor on our input image while depth estimation was satisfactory on the KITTI test image.

4 Conclusion

Idea was to achieve real-time performance for the joint task of depth estimation and semantic segmentation but need to improve further for IoU and RMSE. The model must be further fine-tuned for both KITTI and NYUD datasets since the resulting IoU values are not satisfactory. Also, comparing the inference on KITTI dataset with custom image inference,

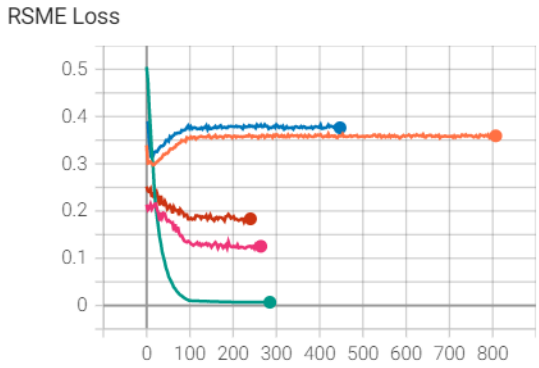

Figure 5: RMSE Loss vs Epochs for different iterations on KITTI dataset.

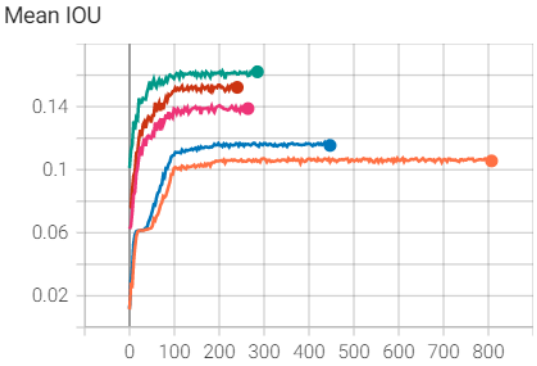

Figure 6: Mean IoU vs Epochs for different iterations on KITTI dataset.

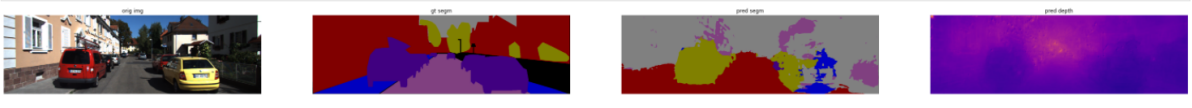

Figure 7: Model inference on KITTI test image.

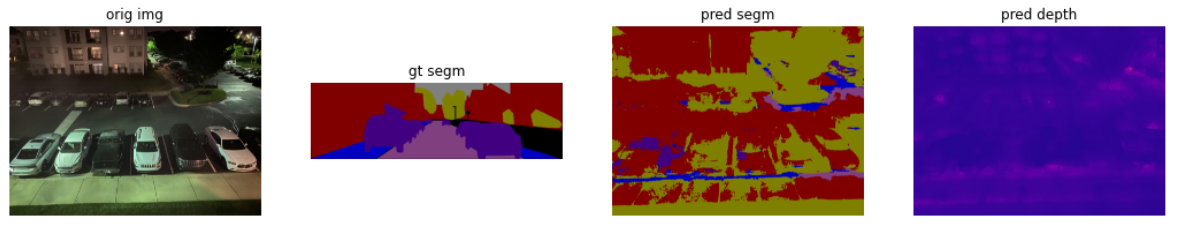

Figure 8: Model inference on custom test image.

it can be said that the model is not generalized enough and is overfitted to images similar to KITTI dataset. Also, the time taken for inference is 0.127 seconds which needs to be further reduced for a real-time use-case.

5 Future Scope

For future improvement, the size of the KITTI dataset can be increased with labels for both semantic segmentation and depth estimation. Secondly, the model complexity could be improved to increase the accuracy and reduce the time complexity of the model. Lastly, additional tasks could be added alongside semantic segmentation and depth estimation which will enable the model to perform three or more tasks in one forward pass. The segmentation mask and the depth map can be fused using libraries like Open3D to create a Point Cloud representation of 3D objects in each scene. We can not only get classification and pixel coordinates of each object in the image but we can also go a step ahead and compute their depth from the vehicle in real world.

References

[1] Vladimir Nekraso, Thanuja Dharmasiri, Andrew Spek, Tom Drummond, Chunhua Shen and Ian Reid, "Real-time segmentation of on-line handwritten arabic script", In *ICRA 2019*

[2] Vladimir Nekraso, Chunhua Shen and Ian Reid, "Light-Weight RefineNet for Real-Time Semantic Segmentation", In BMVC 2018.

[3] Guosheng Lin, Anton Milan, Chunhua Shen, Ian Reid, "RefineNet: Multi-Path Refinement Networks for High-Resolution Semantic Segmentation", In *2016.*

[4] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, Liang-Chieh Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks", In 2016.