Winter 2020 CS231A Final Project Report

Depth estimation from a single view augmented by semantic segmentation

Changqing Lu (*eljulu*)
eljulu@stanford.edu

March 19, 2021

1 Introduction

Autonomous vehicles are an important and fast-developing industry nowadays, where lots of artificial intelligence techniques are being developed and researched. For a successful autonomous vehicle system to work, the vechile must be able to sense and map its environment in a fast and accurate manner so that the control algorithm can navigate through complicated traffic situations. Computer vision techniques such as 3D recognition, 3D reconstruction, motion estimation and tracking can help ensure the safety of the passengers on the vehicle as well as other vehicles or pedestrians on the road.

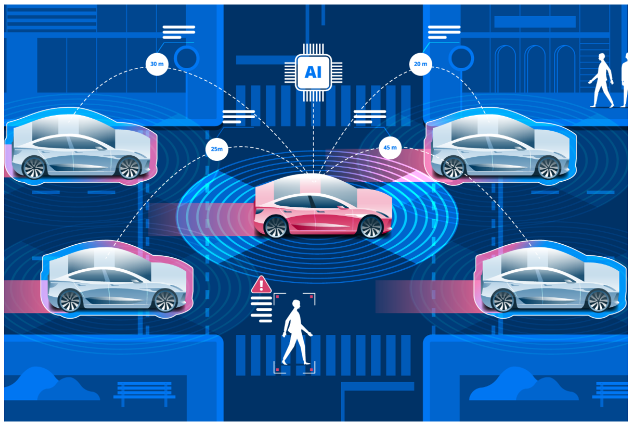


Figure 1: Autonomous Driving System[1]

Apart from that, depth estimation of the entire street scene is also an important topic. Even though nowadays vehicles highly rely on radar or LiDAR for proximity detection, it would be helpful for the system to have additional visional verification of the distance estimation. For depth estimation, for example, Markov Random Field (MRF) can be used to estimate both the relative depth and absolute depth using certain feature vectors along with local depth relationships as well as global image correlations [3].

Semantic segmentation is also helpful because using semantic segmentation, the control system of the vehicle can better understand the scene so that it can make "better decisions". Features such as texture variation, gradients, haze or defocus can be used to separate the image patches [3].

We can see that from the comparison, depth estimation and semantic segmentation share image features such as gradients and texture variation, which suggest some correlation between depth estimation and semantic segmentation.

Thereofre, for this project, we intend to investigate whether depth estimation of a single view can be improved by semantic segmentation.

2 Technical Approach

2.1 Data Set

For the dataset, we are using the Make3D dataset[2], which contains 534 320×240 RGB images as well as the pixel-by-pixel depth value and semantic classification data for each image. There

[1]Source:https://www.smartcitiesworld.net/opinions/opinions/driving-autonomous-vehicles-forward-with-intelligent-infrastructure

[2]Make3D dataset: http://dags.stanford.edu/projects/scenedataset.html

are altogether 9 classes: unknown, sky, tree/bush, road/path, grass, water, building, mountain, foreground. The dataset is separated into 400 training images and 134 test/evaluation images.

2.2 Models

Since we are trying to investigate whether semantic segmentation data can improve depth estimation results, we need to setup two models, one of which is a benchmark.

For both models, the output is $320 \times 240 \times 1$ pixel-by-pixel depth value.

For the input, the first model will not see any data regarding the semantic segmentation of the images since it is constructed as a benchmark. So the input is simply the $320 \times 240 \times 3$ RGB images.

For the second model, the semantic segmentation data will be added to the input as another layer or channel of the image. Therefore, the input will be $320 \times 240 \times 4$ RGB images plus semantic segmentation labels.

After training both models, we will see whether the results on the test sets suggest that the semantic segmentation data is helpful to more improved depth estimation results.

3 Result and Discussion

Benchmark Model

For the first model, we first tried to implement a simple CNN network (Figure 2). As we can see from the structure, since all the layers are convoluted layers, this model doesn't deal with global image relations.

```
self.model = nn.Sequential(
    nn.Conv2d(3, 16, 5, stride = 1, padding = 2),
    nn.ReLU(),
    nn.Conv2d(16, 32, 5, stride = 1, padding = 2),
    nn.ReLU(),
    nn.Conv2d(32, 1, 5, stride = 1, padding = 2)
)
```

Figure 2: Initial benchmark model structure

Using this model, the training results were not satisfactory (Figure 3). As we can see from the training loss versus epoch plot, the training loss doesn't decrease monotonically. Note that in our traning process, we are using the standard mean-squared error loss. Since the pixel-by-pixel depth estimation value is continuous, the mean-squared error loss might be large because we are using a simple architecture, but the loss should still decrease monotonically. This unstable loss over epoch might suggest insufficient parameters.

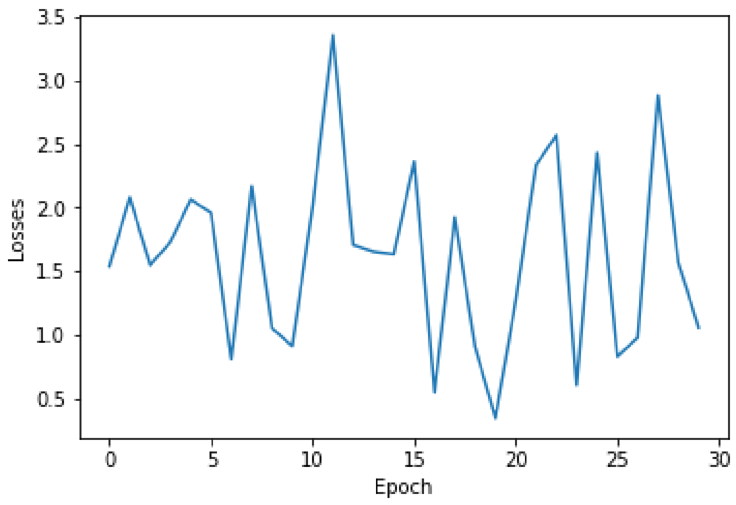
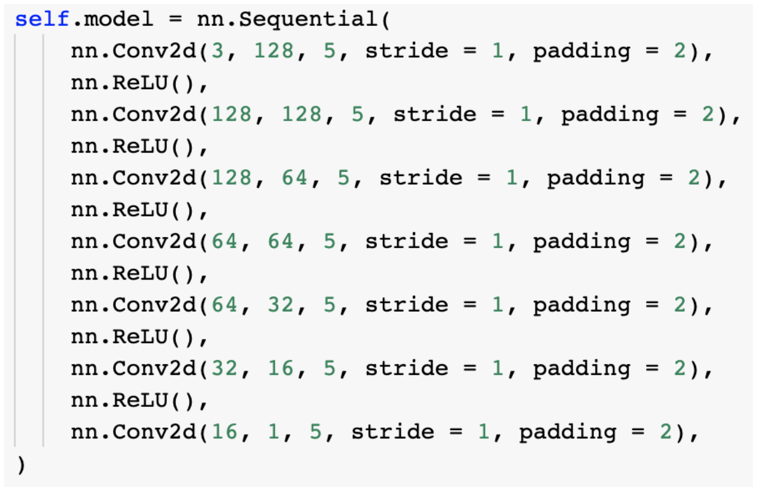


Figure 3: Initial model: Training loss vs. Epochs

Therefore, we tried to increase the complexity of the model (Figure 4). As we can see from the architecture, now it has more convoluted layers, which try to extract more local features from the images.



```
self.model = nn.Sequential(
    nn.Conv2d(3, 128, 5, stride = 1, padding = 2),
    nn.ReLU(),
    nn.Conv2d(128, 128, 5, stride = 1, padding = 2),
    nn.ReLU(),
    nn.Conv2d(128, 64, 5, stride = 1, padding = 2),
    nn.ReLU(),
    nn.Conv2d(64, 64, 5, stride = 1, padding = 2),
    nn.ReLU(),
    nn.Conv2d(64, 32, 5, stride = 1, padding = 2),
    nn.ReLU(),
    nn.Conv2d(32, 16, 5, stride = 1, padding = 2),
    nn.ReLU(),
    nn.Conv2d(16, 1, 5, stride = 1, padding = 2),
)
```

Figure 4: Model with increased complexity

Using the above architecture, the training result is the following (Figure 5).

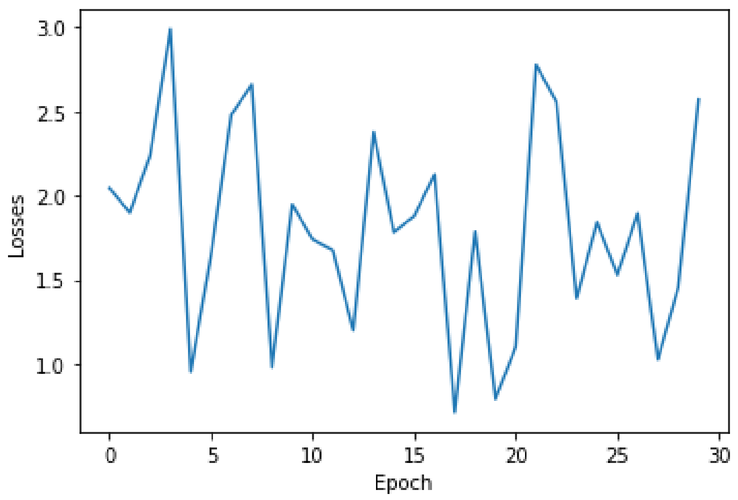


Figure 5: Training loss over epochs for model with increased complexity

We can see from the figure that the traning loss is still somehow "unstable". But compared to the initial model, we can see that the overall training loss for each epoch has decreased. Since for the initial model, we can see data points where the training loss is over 3.0 but for the second model, the maximum training loss is under 3.0. Therefore, we can conclude that for the initial model, there are not sufficient parameters.

However, since for the second model, the traning loss is still not decreasing monotonically, more parameters are still needed. And now we tried to include a global connection between pixels from the image (Figure 6).

```
linear_input_size = 153600
linear_hidden_size = 76800
linear_output_size = 76800

self.model = nn.Sequential(
  nn.Conv2d(3, 16, 5, stride = 1, padding = 2),
  nn.ReLU(),
  nn.MaxPool2d(2, stride = 1),
  nn.Conv2d(16, 32, 5, stride = 1, padding = 2),
  nn.ReLU(),
  nn.MaxPool2d(2, stride = 1),
  nn.Flatten(),
  nn.Linear(linear_input_size, linear_hidden_size),
  nn.Linear(linear_hidden_size, linear_hidden_size),
  nn.Linear(linear_hidden_size, linear_output_size)
)
```

Figure 6: Model with both convoluted layers and fully-connected layers

This model has both convoluted layers and fully-connected layers, where convoluted layers try to extract local features of the image while the fully-connected layers try to build global connections so that all the features can contribute to final depth estimation results. However, due to the computation capacity, we were unable to train this model since due to the image size of the data set ($320 \times 240 \times 3$), although we tried to reduce the size before the fully-connected layer, the number of paramters are still too large.

Augmented Input

However, we still wanted to see whether the input with semantic segmentation might produce better training loss results. Now our input size is $320 \times 240 \times 4$. And for the initial model, we have the following training loss (Figure 7).

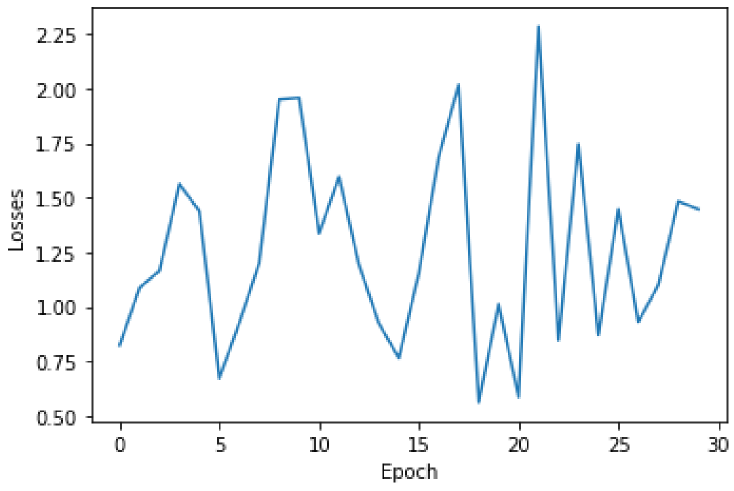


Figure 7: Initial model with semantic segmentation data

Note that the overall training loss is **less than** the overall training loss compared to the model without the semantic segmentation data.

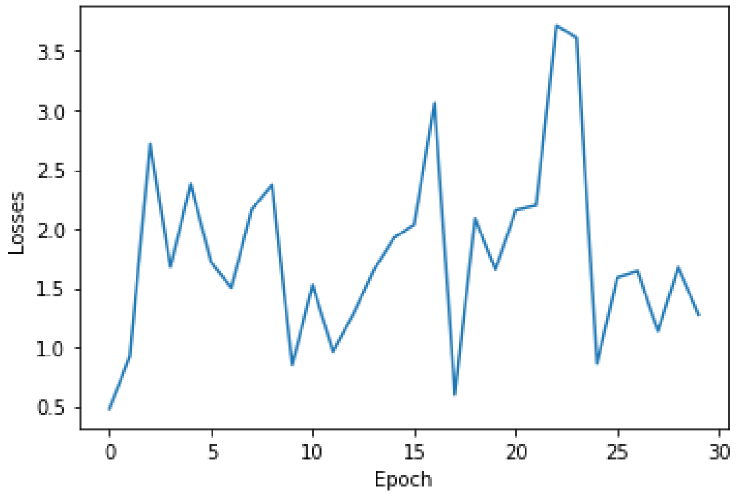


Figure 8: Increased complexity model with semantic segmentation data

Using the second model where the number of convolution layers are increased, we can see the same trend where the overall training loss at each epoch is **less than** the one without the semantic segmentation data.

Therefore, even without sufficient parameters, we can see a trend that the semantic segmentation data **can improve the depth estimation**.

Limitation

The major problem of our models are that they are with insufficient parameters, which result in the situation where the training loss is not monotonically decreasing.

Note that from the dataset, the depth estimation values are all continuous, which might be a reason that a more complicated model is required to have better results. Since the depth values are continous, after extraction of local features, the fully-connected layers are responsible for linear regression instead of logistic regression (classification), which requires far more parameters. That is to say, for the depth estimation value of each pixel, we should first extract local image features, then use the fully-connected layers to conduct linear regression algorithms.

There are researches on using depth estimation values to improve semantic segmentation results [6], which can suggest a strong correlation between depth estimation and semantic segmentation. And since the output of a semantic segmentation process is labels instead of continous depth values, it might require less complicated model or less parameters.

However, with insufficient parameters, the training process still suggest that with semantic segmentation data, the model can have a better prediction on depth estimation of an image.

References

[1] A.Torralba, A. Oliva. **Depth Estimation from Image Structure**. *IEEE Trans. On Pattern Analysis and Machine Intelligence (PAMI)*, 2002.

[2] A. Saxena, S. H. Chung, A. Y. Ng. **Learning Depth from Single Monocular Images**. *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, 2005.

[3] A. Saxena, M. Sun, A. Y. Ng. **Make3D: Learning 3D Scene Structure from a Single Still Image**. *IEEE Trans. of Pattern Analysis and Machine Intelligence (PAMI)*, 2009.

[4] B. Liu, S. Gould, D. Koller. **Single Image Depth Estimation from Predicted Semantic Labels**. *Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[5] S. Gould, R. Fulton, D. Koller. **Decomposing a Scene into Geometric and Semantically Consistent Regions**. *Proceedings of International Conference on Computer Vision (ICCV)*, 2009.

[6] L. Hoyer, D. Dai, Y. Chen, A. Koring, S. Saha, L. Gool. **Three ways to improve semantic segmentation with self-supervised depth estimation**. *arXiv preprint arXiv:2012.10782*, 2020.

[7] M. Liu, S. Lin, S. Ramalingam, O. Tuzel. **Layered Interpretation of Street View Images**. *Robotics Science and System (RSS)*, 2015.