Data Requirements for Semantic Segmentation

Hannes Liik

May 2019

Abstract

This is a technical report on the data requirements for the semantic segmentation part of the perception module. We tested different dataset sizes as subsets from our simulation training set. We observe that the performance gain with respect to the size of the simulation training set quickly plateaus. Furthermore, the amount of data needed to train semantic segmentation networks seems to be less than expected for deep learning, and benefits a lot more from variety over quantity. The results support manual labelling over simulation data generation. Furthermore, we show that by augmenting the styles of the images, we are able to squeeze out significantly more performance using the dataset.

1 Introduction

Semantics segmentation (i.e. pixel-wise image classification, examples in Figure 1) is an essential part of perception for autonomous vehicles. It enables path-planning by predicting traversability cost maps. Manual labelling for semantic segmentation is expensive, however. Simulation environments can instead be used to create datasets, where labels are automatically generated.

This introduces a trade-off: variety comes for free in the real world, but labels are expensive. Variety in the simulation world is manually created by a designer, while labels come free. In this report we attempt to estimate the amount of data needed to train a semantic segmentation network using our own simulation dataset.

2 Dataset

We have created a simple simulation environment that replicates a real world location. From it, we extracted 1853 training and 200 validation images. The images are sequential and captured by traversing through the environment. To test the data requirements of the model, train with \( n \) first instances the training set. We try different \( n \in \{1500, 1000, 500, 100\} \).

3 Architecture and Training

We run all experiments with the Deeplabv3+ [1] using Xception [2] backbone. For all experiments we trained the network from scratch for 40,000 iterations with batch size 9. The images are scaled to 513 pixels on the shorter side. The data is augmented with:

- random 513 \( \times \) 513 crops,
- random horizontal flips
- random scale \([0.5, 2.0]\).

4 Results

4.1 Simulation Validation Set

The results of the networks on the simulation validation set are tabulated in Table 1. It turns out that in the simulation environment, extracting more data does not help much. We observed no loss of performance when using less than a third of the whole training set.

Perhaps this is because the network learns the textures, which are exactly duplicated throughout the environment, even when the scenes are completely different. It may also be that segmentation labels are so dense, that we need much less data than we would assume for deep learning.

We finally see a loss of performance when training on only the first 100 images. In this case, the network has
only seen one sub-environment of the simulation, which is quite different from the environment of the validation set, and might explain poorer performance.

We also experimented with training on 100 random samples from the training set. This is essentially a sparse version of the whole training set. The drop of performance compared to training on the full training set was insignificant.

4.2 Real World Validation Set

With smaller datasets, we expect to see more overfitting. To test the effect of this, we evaluated our models on a real world validation set. The results are tabulated in Table 2. The simulation is modelled after the same real world place, so we expect our models to somewhat well, if simulation training is useful at all. Unfortunately, none of the models produce compelling results. An example by the best model is shown in Figure 2.

Figure 2: Example prediction on real world validation set. Red color marks humans, dark green is vegetation, light green is grass, blue is vehicles, and beige is road. Model was trained on the first 100 dataset images.

5 Texture Augmentation

5.1 Motivation

The simulation set uses a limited set of assets, which are duplicated in different positions to achieve variety. Because the local features (colors, textures) are identical, the network will overfit to these features. In order to limit the dependence on local features, we apply Style Randomization (SR) [3]. This will hopefully allow the network to learn from more images from different scenes, as it is forces to learn more global features such as shapes and context.

5.2 Method

SR mixes the contents of one image and visual style of another random image (from a set of given style images). Examples are shown in Figure 3. Because it was complicated to fit SR into the training pipeline, we pre-generated augmented images into the training set. The ration of stylized to original images is 1:1.

Figure 3: Examples of style randomization.
Table 3: Style randomization results on real world validation set. First column is the number of images sampled from the training set (first \(N\)). Second column is the approximate number of additional stylized images generated from the samples.

<table>
<thead>
<tr>
<th>Samples</th>
<th>N generated</th>
<th>Pixel Accuracy</th>
<th>Mean Accuracy</th>
<th>Mean IU</th>
<th>Freq. Weighted IU</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>750:</td>
<td>0.75</td>
<td>0.51</td>
<td>0.34</td>
<td>0.65</td>
</tr>
<tr>
<td>100</td>
<td>10000:</td>
<td>0.75</td>
<td>0.55</td>
<td>0.37</td>
<td>0.65</td>
</tr>
<tr>
<td>1853</td>
<td>10000:</td>
<td>0.65</td>
<td>0.60</td>
<td>0.39</td>
<td>0.54</td>
</tr>
</tbody>
</table>

from the simulation with more augmented versions of a reduced version of the training set, we will converge to similar accuracy. It should be noted, that the pixel accuracy is higher for the reduced datasets, while mean accuracy is lower. We suspect this is because the reduced version actually resembles the real world validation set more closely and leads to better priors, which increases the global accuracy by intuition.

By training on the whole dataset, the network cannot rely as much on global statistics, which are not as informative for the larger dataset. This would explain the poorer global metrics (pixel accuracy and fwIU) for this validation set, and better detailed/local metrics (mean accuracy and IU). The latter should generalize better.

Our conclusion is that by adding Style Randomization, we enable the network to benefit from a bigger simulation set. Without it, the network can simply memorize the textures and stop learning. Because it cannot rely on texture information, the network will be able to learn from the shape information of additional dataset instances. With that being said, differences are not big enough to be conclusive in our opinion.

6 Conclusions

In this work, we experimented with different simulation dataset sizes. Our observations were the following:

- **The amount of data required for semantic segmentation is surprisingly low.** We believe this is because the labels are very information dense. Our models generalized with a training set of only 100 images.

- **Neural networks do not benefit from more samples of similar data.** In our experiments, networks trained with a fraction of the whole dataset (100 vs 1500) had similar performance. In our limited real world validation set, smaller datasets actually outperformed bigger datasets. We suspect the networks overfit on textures and stop learning.

- **Style randomization** [3] greatly improves network generalization. By augmenting our datasets with style randomization, we achieved greater performance on real data with all networks. By increasing the amount of generated images, the network improved in all metrics. Furthermore, there was benefit in using the whole dataset when applying style randomization. We suspect that because the network was not allowed to overfit on textures, it was able to learn more from additional images (new perspectives etc.). The difference between using the smaller and bigger datasets were still little.

From our observations we made the following conclusions:

- **Labelling real world images is better than simulation generated data.** The networks require variety, which is expensive to create in simulation. Even with good variety, there still exists a domain shift. For semantic segmentation, the amount of images needed to train is low, but the variety requirements are big. It is likely more reasonable to pay for manual labelling than for simulation engineering.

- **Simulation environments should be small, but varied and numerous.** Big environments tend to be more of the same. We did not observe much benefit from sampling more images from different parts of a big environment.

References

