Semantic Segmentation using Resource Efficient Deep Learning

Naresh Kumar Gurulingan
naresh.gurulingan@smail.inf.h-brs.de

Hochschule Bonn-Rhein-Sieg

April 30, 2018

Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

Table of Contents

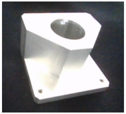
Overview of the dataset

Objects in the dataset:

(a) axis

(b) bearing

(c) bearing box AX01

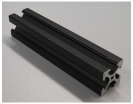


(d) bearing box AX16

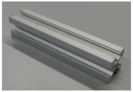
(e) container blue

(f) container red

(g) distance tube



(h) F20_20_B



(i) F20_20_G

(j) M20 [1]

(k) M20_100

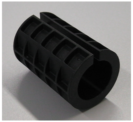
(l) M30

Objects in the dataset [1]

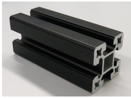
Overview of the dataset

Objects in the dataset:

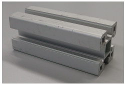
(a) motor



(b) R20



(c) S40_40_B



(d) S40_40_G

(e) em_01

(f) em_02

Objects in the dataset [1]

- 18 objects associated to RoboCup @Work are used for the dataset.
- 540 images (30 of each object) were taken to create the training dataset.
- These images were labeled using the MATLAB ImageLabeler app.

Motivation behind the augmentation algorithm

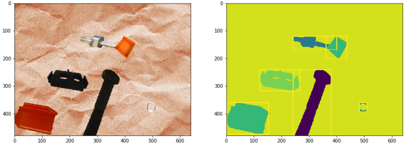
- Manually labeling 540 images using MATLAB ImageLabeler takes roughly 2160 minutes (roughly 4 minutes per image). This is equivalent to around 4 working days. Hence, creating a large dataset with manual labeling is not feasible.
- Taking images in a variety of real world backgrounds is also time consuming.
- Labeling images with multiple objects would take an even longer time.

These drawbacks could be overcome by randomly placing objects on a variety of different background images automatically using an algorithm.

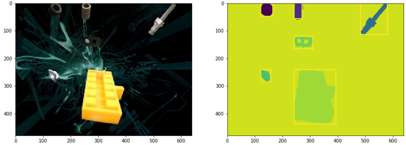
About the augmentation algorithm

- 150 background images were downloaded from the internet using "Google Images Download" [2] with the keywords: 1. 640x480 background images, 2. 640x480 textured images and 3. 640x480 wallpapers.
- Using the object locations obtained from manually labeled images, objects were extracted and placed randomly in different locations and scales on these backgrounds.
- The algorithm provides several tunable parameters which control the occlusions between objects and other aspects as detailed in the dataset report.
- The algorithm can additionally also provide object detection labels.

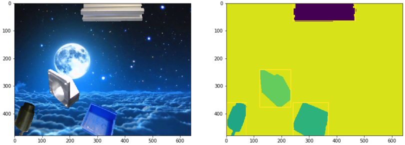
Sample results



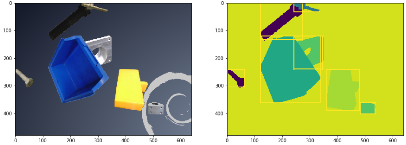
(a) Sample result 1



(b) Sample result 2



(c) Sample result 3



(d) Sample result 4

Sample results

Meta-data of the dataset

Details regarding the dataset is provided in table 1.

	Training	Validation	Test
Real Images	30 per object. Total: $30\times18=540$	5 per object. Total: $5\times18=90$	5 per object. Total: $5\times18=90$
Augmented Images	5000	810	810
Total Images	5540	900	900

Meta-data of the dataset

Selected segmentation architectures for training

Focus on accuracy:

1. Deeplab v3+ [3]:
   - Currently holds the second position in cityscapes leaderboard [4].
   - Implementation is available in the tensorflow/models repository [5].
   - Uses Xception backbone to reduce computation time and memory requirements.
2. PSPNet [6]:
   - Currently among the top 10 in the cityscapes leaderboard [4].
   - Different sources of implementations are available.
   - The authors attempt to reduce global context information loss between sub-regions of the input image which is essential for semantic segmentation.

Selected segmentation architectures for training

Focus on resource consumption:

3 ICNet [7]:
  - Focus on attaining real time inference by using a cascade of networks without significant loss of accuracy.
  - Input image is taken in 3 different resolutions. Most of the computations is performed on the low resolution image and the computations performed on higher resolution image is only intended to improve prediction of small objects.

4 MobileNet v2 [8]:
  - Focus on memory efficient inference by using depthwise separable convolutions and other techniques.
  - The authors also evaluate the use of mobileNet v2 on semantic segmentation (in addition to image classification).
  - Implementation is available in the tensorflow/models repository [9].

Pending tasks

- Training the 4 selected segmentation models on the created dataset and reporting results.
- Finalizing pruning and quantization methods.
- Finding and using implementations of the finalized pruning and quantization methods.
- Reporting the effects of pruning and quantization on the 4 selected segmentation models.

References

[1] github: robocup-at-work/rulebook/images. URL: https://github.com/robocup-at-work/rulebook/tree/master/images.

[2] github: hardikvasa/google-images-download. URL: https://github.com/hardikvasa/google-images-download.

[3] L.-C. Chen et al. "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation". In: arXiv:1802.02611 (2018).

[4] Cityscapes leaderboad. URL: https://www.cityscapes-dataset.com/benchmarks/.

[5] Tensorflow models: deeplab. URL: https://github.com/tensorflow/models/tree/master/research/deeplab.

[6] H. Zhao et al. "Pyramid Scene Parsing Network". In: CoRR abs/1612.01105 (2016). arXiv: 1612.01105. URL: http://arxiv.org/abs/1612.01105.

[7] H. Zhao et al. "ICNet for Real-Time Semantic Segmentation on High-Resolution Images". In: CoRR abs/1704.08545 (2017). arXiv: 1704.08545. URL: http://arxiv.org/abs/1704.08545.

[8] M. Sandler et al. "Inverted Residuals and Linear Bottlenecks: Mobile Networks for Classification, Detection and Segmentation". In: CVPR. 2018.

[9] Tensorflow models: MobileNetV2. URL: https://github.com/tensorflow/models/tree/master/research/slim/nets/mobilenet.