FAQ

Q1: What if I want to use other network backbones, such as ResNet [1], instead of only those provided ones (e.g., Xception)?

A: The users could modify the provided core/feature_extractor.py to support more network backbones. _____ Q2: What if I want to train the model on other datasets?

A: The users could modify the provided dataset/build_{cityscapes,voc2012}*data.py and dataset/segmentation_dataset.py to build their own dataset.* ___ Q3: Where can I download the PASCAL VOC augmented training set?

A: The PASCAL VOC augmented training set is provided by Bharath Hariharan et al. [2] Please refer to their website for details and consider citing their paper if using the dataset. _____ Q4: Why the implementation does not include DenseCRF [3]?

A: We have not tried this. The interested users could take a look at Philipp Krähenbühl's website and paper for details. _____ Q5: What if I want to train the model and fine-tune the batch normalization parameters?

A: If given the limited resource at hand, we would suggest you simply fine-tune from our provided checkpoint whose batch-norm parameters have been trained (i.e., train with a smaller learning rate, set `fine_tune_batch_norm = false`, and employ longer training iterations since the learning rate is small). If you really would like to train by yourself, we would suggest

1. Set `output_stride = 16` or maybe even `32` (remember to change the flag `atrous_rates` accordingly, e.g., `atrous_rates = [3, 6, 9]` for `output_stride = 32`).
2. Use as many GPUs as possible (change the flag `num_clones` in train.py) and set `train_batch_size` as large as possible.
3. Adjust the `train_crop_size` in train.py. Maybe set it to be smaller, e.g., 513x513 (or even 321x321), so that you could use a larger batch size.
4. Use a smaller network backbone, such as MobileNet-v2.

Q6: How can I train the model asynchronously?

A: In the train.py, the users could set `num_replicas` (number of machines for training) and `num_ps_tasks` (we usually set `num_ps_tasks` = `num_replicas / 2`). See slim.deployment.model_deploy for more details. _____ Q7: I could not reproduce the performance even with the provided checkpoints.

A: Please try running

```
# Run the simple test with Xception_65 as network backbone.
sh local_test.sh
```

or

```
# Run the simple test with MobileNet-v2 as network backbone.
sh local_test_mobilenetv2.sh
```

First, make sure you could reproduce the results with our provided setting. After that, you could start to make a new change one at a time to help debug. _____ Q8: What value of `eval_crop_size` should I use?

A: Our model uses whole-image inference, meaning that we need to set `eval_crop_size` equal to `output_stride` * k + 1, where k is an integer and set k so that the resulting `eval_crop_size` is slightly larger the largest image dimension in the dataset. For example, we have `eval_crop_size` = 513x513 for PASCAL dataset whose largest image dimension is 512. Similarly, we set `eval_crop_size` = 1025x2049 for Cityscapes images whose image dimension is all equal to 1024x2048. _____ Q9: Why multi-gpu training is slow?

A: Please try to use more threads to pre-process the inputs. For, example change num_readers = 4. _____

References

1. **Deep Residual Learning for Image Recognition** Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun [link], In CVPR, 2016.

2. **Semantic Contours from Inverse Detectors** Bharath Hariharan, Pablo Arbelaez, Lubomir Bourdev, Subhransu Maji, Jitendra Malik [link], In ICCV, 2011.

3. **Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials** Philipp Krähenbühl, Vladlen Koltun [link], In NIPS, 2011.