

MobileNetSSD on DPU

V 1.0, June 5 2019

1. First Install the Caffe with GPU on Ubuntu 16.04 Machine, for installation follow this guide: [LogicTronix Caffe Installation Tutorial](#).
2. Make sure you install the <https://github.com/weiliu89/caffe/tree/ssd> , fork of caffe which has SSD example in your HOME directory. We will refer to home directory as $\${HOME}$ from here.

Note: Replace $\${HOME}$ with absolute path of your home directory.

3. Run following command:

```
export PYTHONPATH= $\${HOME}$ /caffe/python
```

Whenever you error related to python extension files, it could because you missed this step and you have to do this in every new instance of terminal.

4. Now prepare the VOC database:

- a. Download the pre-trained: VGG_ILSVRC_16_layers_fc_reduced_deploy.prototxt (description) and VGG_ILSVRC_16_layers_fc_reduced.caffemodel (weights) files of the VGG network. These will be used as the backbone (feature extraction layers) for our SSD network and since they are pre-trained, it will help to reduce training time.
- b. Copy these files into the $\${HOME}$ /caffe/models/VGGNet/ folder. Create the folder if it doesnot exist.
- c. Download the PASCAL VOC dataset from the following three links:
 - http://host.robots.ox.ac.uk/pascal/VOC/voc2012/VOCtrainval_11-May-2012.tar
 - http://host.robots.ox.ac.uk/pascal/VOC/voc2007/VOCtrainval_06-Nov-2007.tar
 - http://host.robots.ox.ac.uk/pascal/VOC/voc2007/VOCtest_06-Nov-2007.tar
- d. Copy the 3 tar files to $\${HOME}$ /caffe/data. Extract all files from the 3 archives and merge them under a single VOCdevkit folder. At the end of this process you should have a new folder name VOCdevkit which contains the VOC2007 and VOC2012 subfolders.

Note: This dataset contains the training, validation, and testing images as well as annotations which are the bounding box locations contained in .xml format.

- e. For the next step, you will be processing the annotations (labels) and the training images into LMDB files which can then be directly used by Cffe for the training process.
- f. For this step, you need two shell scripts *create_list.sh* and *create_data.sh* which are available in $\${HOME}$ /caffe/data/VOC0712 directory along with another file named *labelmap_voc.prototxt*.

Inside *create_list.sh* file edit root_dir to

```
root_dir= $\${HOME}$ /caffe/data/VOCdevkit/
```

Similarly in *create_data.sh* file edit data_root_dir to

```
data_root_dir= $\${HOME}$ /caffe/data/VOCdevkit
```

- g. Execute the following commands and change the directory location:

```
cd ${HOME}/caffe/data/VOC0712/
source create_list.sh
source create_data.sh
```

```
logictronix@logictronix-Lenovo-ideapad-Y700-15ISK: ~/caffe
logictronix@logictronix-Lenovo-ideapad-Y700-15ISK:~/caffe/data/VOC0712$ ls
coco_voc_map.txt  create_data.sh  create_list.sh  labelmap_voc.prototxt  test_name_size.txt  test.txt  trainval.txt
logictronix@logictronix-Lenovo-ideapad-Y700-15ISK:~/caffe/data/VOC0712$ source create_list.sh
Create list for VOC2007 trainval...
Create list for VOC2012 trainval...
Create list for VOC2007 test...
[0705 13:50:52.277324 8196 get_image_size.cpp:61] A total of 4952 images.
[0705 13:50:54.082756 8196 get_image_size.cpp:100] Processed 1000 files.
[0705 13:50:55.818579 8196 get_image_size.cpp:100] Processed 2000 files.
[0705 13:50:57.580237 8196 get_image_size.cpp:100] Processed 3000 files.
[0705 13:50:59.341400 8196 get_image_size.cpp:100] Processed 4000 files.
[0705 13:51:01.015326 8196 get_image_size.cpp:105] Processed 4952 files.
logictronix@logictronix-Lenovo-ideapad-Y700-15ISK:~/caffe/data/VOC0712$ source create_data.sh
/home/logictronix/caffe/build/tools/convert_annotset --anno_type=detection --label_map_file=/home/logictronix/caffe/data/VOC0712/../../data/VOC0712/labelmap_voc.prototxt --check_label=True --min_dim=0 --max_dim=0 --resize_height=0 --resize_width=0 --backend=lmdb --shuffle=False --check_size=False --encode_type=jpg --encoded=True --gray=False /home/logictronix/caffe/data/VOCdevkit/ /home/logictronix/caffe/data/VOC0712/../../data/VOC0712/test.txt /home/logictronix/caffe/data/VOCdevkit/VOC0712/lmdb/VOC0712_test_lmdb
[0705 13:51:06.201283 8221 convert_annotset.cpp:122] A total of 4952 images.
[0705 13:51:06.201982 8221 db_lmdb.cpp:35] Opened lmdb /home/logictronix/caffe/data/VOCdevkit/VOC0712/lmdb/VOC0712_test_lmdb
[0705 13:51:11.139374 8221 convert_annotset.cpp:195] Processed 1000 files.
[0705 13:51:16.121017 8221 convert_annotset.cpp:195] Processed 2000 files.
[0705 13:51:21.202070 8221 convert_annotset.cpp:195] Processed 3000 files.
[0705 13:51:26.128345 8221 convert_annotset.cpp:195] Processed 4000 files.
[0705 13:51:30.930958 8221 convert_annotset.cpp:201] Processed 4952 files.
/home/logictronix/caffe/build/tools/convert_annotset --anno_type=detection --label_map_file=/home/logictronix/caffe/data/VOC0712/../../data/VOC0712/labelmap_voc.prototxt --check_label=True --min_dim=0 --max_dim=0 --resize_height=0 --resize_width=0 --backend=lmdb --shuffle=False --check_size=False --encode_type=jpg --encoded=True --gray=False /home/logictronix/caffe/data/VOCdevkit/ /home/logictronix/caffe/data/VOC0712/../../data/VOC0712/trainval.txt /home/logictronix/caffe/data/VOCdevkit/VOC0712/lmdb/VOC0712_trainval_lmdb
[0705 13:51:31.624477 8241 convert_annotset.cpp:122] A total of 16551 images.
[0705 13:51:31.625301 8241 db_lmdb.cpp:35] Opened lmdb /home/logictronix/caffe/data/VOCdevkit/VOC0712/lmdb/VOC0712_trainval_lmdb
[0705 13:52:12.758421 8241 convert_annotset.cpp:195] Processed 1000 files.
[0705 13:52:51.925642 8241 convert_annotset.cpp:195] Processed 2000 files.
[0705 13:53:30.639240 8241 convert_annotset.cpp:195] Processed 3000 files.
[0705 13:54:09.440342 8241 convert_annotset.cpp:195] Processed 4000 files.
[0705 13:54:48.724550 8241 convert_annotset.cpp:195] Processed 5000 files.
[0705 13:55:26.655509 8241 convert_annotset.cpp:195] Processed 6000 files.
[0705 13:56:04.842099 8241 convert_annotset.cpp:195] Processed 7000 files.
[0705 13:56:43.782294 8241 convert_annotset.cpp:195] Processed 8000 files.
[0705 13:57:22.401336 8241 convert_annotset.cpp:195] Processed 9000 files.
[0705 13:58:01.288853 8241 convert_annotset.cpp:195] Processed 10000 files.
[0705 13:58:40.360512 8241 convert_annotset.cpp:195] Processed 11000 files.
[0705 13:59:19.479089 8241 convert_annotset.cpp:195] Processed 12000 files.
[0705 13:59:57.243762 8241 convert_annotset.cpp:195] Processed 13000 files.
[0705 14:00:36.096668 8241 convert_annotset.cpp:195] Processed 14000 files.
[0705 14:01:14.433995 8241 convert_annotset.cpp:195] Processed 15000 files.
[0705 14:01:55.620456 8241 convert_annotset.cpp:195] Processed 16000 files.
[0705 14:02:16.798876 8241 convert_annotset.cpp:201] Processed 16551 files.
logictronix@logictronix-Lenovo-ideapad-Y700-15ISK:~/caffe$
```

- h. This process creates two LMDB databases stored in `${HOME}/caffe/data/VOCdevkit/VOC0712/lmdb`, respectively of size ~1.8GB the training validation database and 445MB the test database.

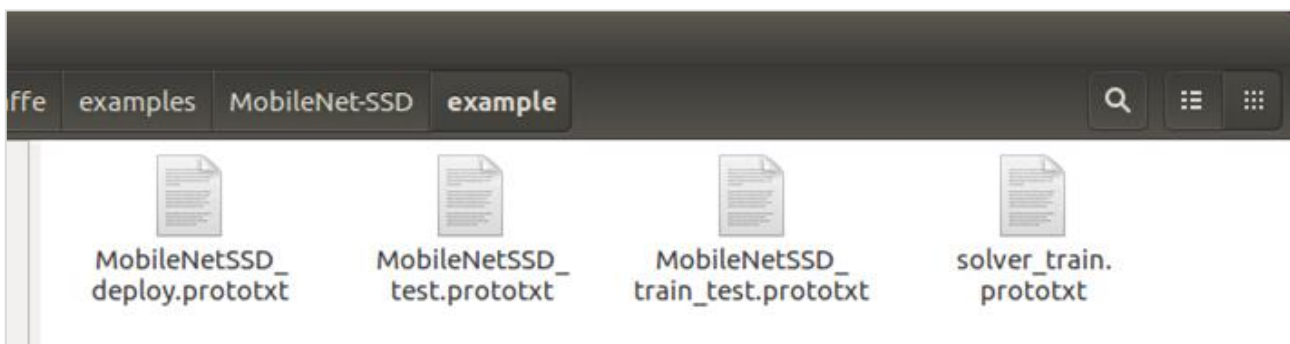
5. Training the SSD model:

- a. Download the MobileNet-SSD for caffe from this repo: http://host.robots.ox.ac.uk/pascal/VOC/voc2007/VOCtest_06-Nov-2007.tar
- b. Copy this MobileNet-SSD folder to caffe examples directory i.e. `${HOME}/caffe/examples`
- c. Copy `labelmap_voc.prototxt` from `${HOME}/caffe/data/VOC0712` to `${HOME}/caffe/examples/MobileNet-SSD` and rename it to `labelmap.prototxt`
- d. Change the directory to MobileNet-SSD inside caffe examples
`cd ${HOME}/caffe/examples/MobileNet-SSD`
- e. There is a script called ***gen_model.sh***.
`./gen_model.sh`
- f. This will create a example folder inside which we have our training prototxt. Following files are created:
 - `MobileNetSSD_deploy.prototxt`
 - `MobileNetSSD_test.prototxt`
 - `MobileNetSSD_train.prototxt`

- g. Change the directory to `${HOME}/caffe/examples/MobileNet-SSD/example`

```
cd ${HOME}/caffe/examples/MobileNet-SSD/example
```
- h. Create file `solver_train.prototxt` and add the following content


```
net: "example/MobileNetSSD_train_test.prototxt"
test_iter: 673
test_interval: 10000
base_lr: 0.0005
display: 10
max_iter: 120000
lr_policy: "multistep"
gamma: 0.5
weight_decay: 0.00005
snapshot: 10000
snapshot_prefix: "snapshot/mobilenet"
solver_mode: GPU
debug_info: false
snapshot_after_train: true
test_initialization: false
average_loss: 10
stepvalue: 20000
stepvalue: 40000
iter_size: 4
type: "RMSPProp"
eval_type: "detection"
ap_version: "11point"
```
- i. Now rename the `MobileNetSSD_train.prototxt` to `MobileNetSSD_train_test.prototxt`



6. Modifying MobileNetSSD Prototxt Files for Compatibility with the DPU/DNNDK
 - a. The modified Prototxt are attached with this document. Replace those files with the files inside


```
${HOME}/caffe/examples/MobileNet-SSD/example
```
 - b. Edit file `MobileNetSSD_train_test.prototxt`.
 - In line 49, replace `${HOME}` with absolute path of your home directory.

```

}
}
data_param {
  source: "${HOME}/caffe/data/VOCdevkit/VOC0712/lmdb/VOC0712_trainval_lmdb"
  batch_size: 8
  backend: LMDB
}

```

- In line 161, replace `${HOME}` with absolute path of your home directory.

```

}
data_param {
  source: "${HOME}/caffe/data/VOCdevkit/VOC0712/lmdb/VOC0712_test_lmdb"
  batch_size: 8
  backend: LMDB
}

```

- c. Now Run the following commands:

```
cd ${HOME}/caffe/examples/MobileNet-SSD
```

```

${HOME}/caffe/build/tools/caffe train --
solver="${HOME}/caffe/examples/MobileNet-
SSD/example/solver_train.prototxt" --gpu 0

```

This start training the model for 120K iterations. The trained caffemodel will be saved in `caffe/examples/MobileNet-SSD/snapshot`. Also this will take a snapshot every 10000 iterations and also if you stop the training process by Ctrl+C.

You can then later resume the training process from last iteration with following command:

```

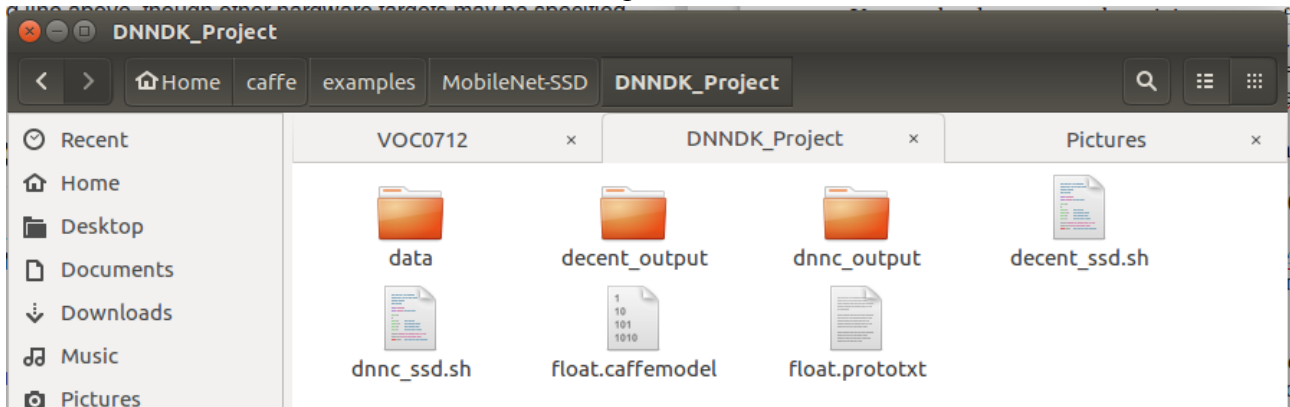
${HOME}/caffe/build/tools/caffe train --
solver="${HOME}/caffe/examples/MobileNet-
SSD/example/solver_train.prototxt" --
snapshot="${HOME}/caffe/examples/MobileNet-
SSD/snapshot/mobilenet_iter_10000.solverstate" --gpu 0

```

Here `mobilenet_iter_10000.solverstate` is the name of last snapshot, you have to modify it according to the last iteration during your training process. Since both training and test network are in same prototxt, testing will also take place every 10000 iteration, so you can see the accuracy.

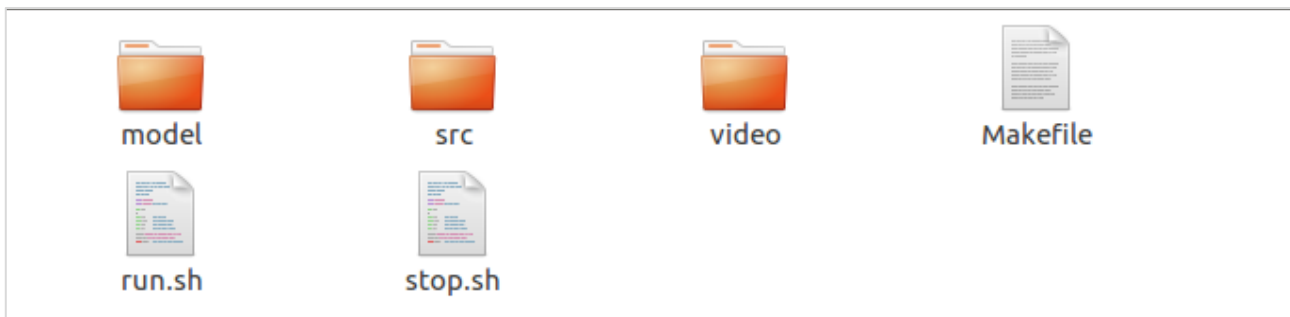
7. After the training process you will have caffemodel named ***mobilenet_iter_120000.caffemodel*** inside the snapshot directory. Our caffemodel and prototxt are now ready for Quantization and Compilation in DNNDK.
8. Configuring the Files for Quantization and Compilation:
 - a. We create a float.prototxt which is identical to train_test.prototxt with a few modifications.
The input layers have been modified to specify paths to the calibration data as well as to enable the autotest capabilities within DNNDK.

- b. Copy your latest trained model from Caffe into the DNNDK_Project directory and rename it “float.caffemodel”. Assuming you ran the full 120K training iterations, this model should have the following location and name:
`"${HOME}/caffe/examples/MobileNet-SSD/snapshot/mobilenet_iter_120000.caffemodel"`
- c. Run the quantization tools by changing the directory to the DNNDK_Project and running the following command:
`cd ${HOME}/caffe/examples/MobileNet-SSD/DNNDK_Project`
`./decent_ssd.sh`
- d. After step above, you should see a “deploy.prototxt” and “deploy.caffemodel” in the decent_output directory. At this point, you can call the following command to compile the model into an executable file that can be run on the DPU
`./dnnc_ssd.sh`
- e. At this point, an elf file should have been created in the **dnnc_output** directory which can be used in the final step which is to run the model on the ZCU102



9. Running the SSD Model on the ZCU102

- a. We have ZCU_102 Application to run the model and this Applications folder contains the following files as shown below. These files can be found in EDGE AI ML-SSD_PASCAL repo.



You can follow the EDGE AI ML-SSD-PASCAL tutorial Step number 6, to run the application on the board. [ML-SSD-PASCAL](#)

- b. The first step that needs to be done by the user is to copy you `ssd` compiled `.elf` file into the **model** directory and name it `dpu_ssd.elf`.
The ZCU102 Application directory structure contains the following:
- **Makefile** : Used to build the application on the ZCU102 and link both the software application `.elf` with the compiled SSD model `.elf`
 - **run.sh** : Used to execute the application on the target by running `source run.sh` on the ZCU102.
 - **stop.sh** : Used to execute the application on the target by running `source stop.sh` on the ZCU102.
 - **model** directory : This is the directory where you need to copy your compiled model executable and make sure it is named `dpu_ssd.elf`
 - **src** directory : This directory contains the ARM source code that will be executed to read in a video file, resize it, and call the DPU APIs to process the images and then display the output with overlays on a displayport monitor.
 - **Video** directory : User should add a video file to evaluate the network against in this directory.
- c. Download the Boot Image for ZCU102 called 2018-12-04-zcu102-desktop-stretch.img.zip from: [ai-developer-hub.html#edge](#)
- d. Before you run the application, you need to setup the board and install DNNDK tools and libraries in the ZCU102.
- e. To install the DNNDK tool in ZCU102 follow [UG1327](#).

Thank You!