Training Caffe model for custom object detection.

So, up to now you should have done and have the following:

a) Ubuntu 16.04 system.
b) Installed SIMCAM SDK and Toolchain
   If you have not installed, you can check Toolchain installation and usage
c) Installed Opencv 3.4 or higher version

Training your own custom object detection model is very easy using SIMCAM SDK, all you need are video files contain desired object. Here is a simple guide how to do it.

Preparing data for training:

1. Open SIMCAM SDK folder and copy all your video files into train/data/Images_xmls/videos folder
2. Open terminal in train/data/Images_xmls folder and run video2img.py python script:

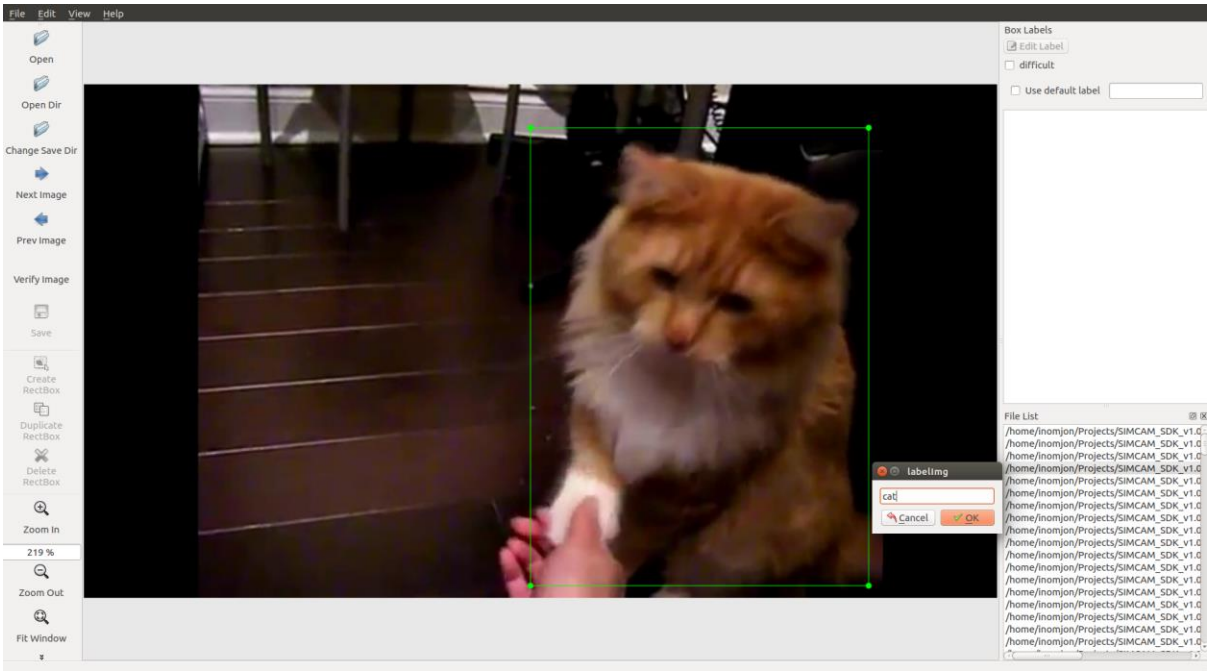
```
inomjon@inomjon-C-H310M-K-PRO:~/Projects/SIMCAM_SDK_v1.0.0/train/data/Images_xml
s$ python3 video2img.py
```

   It will save one frame as an image per second in JPEGImages folder by default. However, there are options; you can change input folder, output folder and number of frames to save.

```
inomjon@inomjon-C-H310M-K-PRO:~/Projects/SIMCAM_SDK_v1.0.0/train/data/Images_xml
s$ python3 video2img.py -h
usage: video2img.py [-h] [--input INPUT] [--output OUTPUT]
                    [--num NUMFRAMEPERSECOND]

optional arguments:
  -h, --help            show this help message and exit
  --input INPUT, -i INPUT
                        video input path
  --output OUTPUT, -o OUTPUT
                        output path
  --num NUMFRAMEPERSECOND, -n NUMFRAMEPERSECOND
                        num frame to get per second
```

3. Image annotation. You should annotate extracted images manually. We have provided an open source annotation tool named labelImg. That tool provides the object coordinate in xml format as output for further processing. Simple annotations steps are shown below:
   a) Execute labelImg file, open image dataset folder (in our case JPEGImages folder) by clicking the OpenDir icon on the left pane.
   b) Image will appear. Click “Change Save Dir” icon and choose Annotations folder as a save folder. Draw rectangle boxes around the objects by clicking the Create RectBox icon and give a label. These boxes are called bounding boxes.
   c) Repeat second step for the each image that appears. Below image shows an example of an annotated image.



4. If you finished all above steps completely, you get bunch of xml files (annotations) inside Annotation folder and images JPEGImages folder.
5. Open terminal inside the Images_xmls folder and run create_txt.py python script:

```
inomjon@inomjon-C-H310M-K-PRO:~/Projects/SIMCAM_SDK_v1.0.0/train/data/Images_xml
s$ python create_txt.py
```

   This python script will create train.txt, test.txt, trainval.txt and val.txt files in the ImageSets/Main folder
6. Go in data/lmdb_files folder and create your own labelmap.prototxt file, example has exist in the folder; you can change it according to your dataset.

```
item {
  name: "none_of_the_above"
  label: 0
  display_name: "background"
}
item {
  name: "cat"
  label: 1
  display_name: "cat"
}
```

7. In the terminal run create_list.sh script :

```
inomjon@inomjon-C-H310M-K-PRO:~/Projects/SIMCAM_SDK_v1.0.0/train/data/lmdb_files$ ./create_list.sh
/home/inomjon/Projects/SIMCAM_SDK_v1.0.0/train/data
./create_list.sh: line 17: [: missing `]'
Create list for Images_xmls trainval...
./create_list.sh: line 17: [: missing `]'
Create list for Images_xmls test...
I0325 15:51:18.579074 12349 get_image_size.cpp:61] A total of 31 images.
I0325 15:51:19.894744 12349 get_image_size.cpp:105] Processed 31 files.
```

   It will generate trainval.txt, test.txt, test_name_size.txt files in the folder

8. Last step is generating lmdb files, lmdb is caffe's data format for training.
   In the terminal run create_data.sh script:

```
inomjon@inomjon-C-H310M-K-PRO:~/Projects/SIMCAM_SDK_v1.0.0/train/data/lmdb_files$ ./create_data.sh
/opt/movidius/ssd-caffe/build/tools/convert_annoset --anno_type=detection --label_type=xml --label_map_file
=/home/inomjon/Projects/SIMCAM_SDK_v1.0.0/train/data/lmdb_files/labelmap.prototxt --check_label=True --min_
dim=0 --max_dim=0 --resize_height=0 --resize_width=0 --backend=lmdb --shuffle=False --check_size=False --en
code_type=jpg --encoded=True --gray=False /home/inomjon/Projects/SIMCAM_SDK_v1.0.0/train/data/ /home/inomjo
n/Projects/SIMCAM_SDK_v1.0.0/train/data/lmdb_files/test.txt /home/inomjon/Projects/SIMCAM_SDK_v1.0.0/train/
data/lmdb_files/lmdb/lmdb_files_test_lmdb
I0325 16:03:41.646299 12803 convert_annoset.cpp:122] A total of 31 images.
I0325 16:03:41.646529 12803 db_lmdb.cpp:35] Opened lmdb /home/inomjon/Projects/SIMCAM_SDK_v1.0.0/train/data
/lmdb_files/lmdb/lmdb_files_test_lmdb
I0325 16:03:44.710017 12803 convert_annoset.cpp:201] Processed 31 files.
/opt/movidius/ssd-caffe/build/tools/convert_annoset --anno_type=detection --label_type=xml --label_map_file
=/home/inomjon/Projects/SIMCAM_SDK_v1.0.0/train/data/lmdb_files/labelmap.prototxt --check_label=True --min_
dim=0 --max_dim=0 --resize_height=0 --resize_width=0 --backend=lmdb --shuffle=False --check_size=False --en
code_type=jpg --encoded=True --gray=False /home/inomjon/Projects/SIMCAM_SDK_v1.0.0/train/data/ /home/inomjo
n/Projects/SIMCAM_SDK_v1.0.0/train/data/lmdb_files/trainval.txt /home/inomjon/Projects/SIMCAM_SDK_v1.0.0/tr
ain/data/lmdb_files/lmdb/lmdb_files_trainval_lmdb
I0325 16:03:45.340495 12813 convert_annoset.cpp:122] A total of 171 images.
I0325 16:03:45.340721 12813 db_lmdb.cpp:35] Opened lmdb /home/inomjon/Projects/SIMCAM_SDK_v1.0.0/train/data
/lmdb_files/lmdb/lmdb_files_trainval_lmdb
I0325 16:04:02.854202 12813 convert_annoset.cpp:201] Processed 171 files.
```

   It will create trainval_lmdb and test_lmdb files in the lmdb folder.

Train model:

So now, you nearly got everything ready to train the Network with the data prepared by yourself. The last thing is, the Network! SIMCAM team provide a robust Network and all necessary scripts for you to train and deploy your own model on the SIMCAM products.

1. Run gen_model.sh script to generate Network:
   ./gen_model.sh <num>

```
inomjon@inomjon-C-H310M-K-PRO:~/Projects/SIMCAM_SDK_v1.0.0/train$ ./gen_model.sh 2
```

   "num" is number of classes in your dataset including the background class.
   It will create prototxts folder and .prototxt files inside the folder for training, testing and deploying the modelh
2. If you do not have at least Get Force GTX 1060 or higher version of GPU hardware on your Ubuntu machine, you can skip this step. Because while you are installing SIMCAM SDK and Toolchain it installs caffe-ssd CPU version on your machine automatically, in /opt/movidius/ssd-caffe path. Let's install GPU version of caffe-ssd to speed up your training process.
   To make process simpler, SIMCAM team has provided docker image in docker hub, and Dockerfile for installation GPU version of caffe-ssd. All you should to do is having docker and nvidia-docker on your Ubuntu system. Here is some information about docker and installation process of docker and nvidia-docker. Let's see simple steps to pull and run simcam/caffe-ssd:gpu docker image into your machine:
   a) sudo docker run –runtime=nvidia -ti --name=simcam -v /home/:/home/ simcam/caffe-ssd:gpu bash

```
inomjon@inomjon-C-H310M-K-PRO:~/Projects$ sudo docker run --runtime=nvidia -ti --name=simcam -v /home/inomjon/:/home/inomjon simcam/caffe-ssd:gpu bash
Unable to find image 'simcam/caffe-ssd:gpu' locally
gpu: Pulling from simcam/caffe-ssd
7b722c1070cd: Already exists
5fbf74db61f1: Already exists
ed41cb72e5c9: Already exists
7ea47a67709e: Already exists
4c2bcc2f5dbc: Already exists
6911a9137d28: Already exists
e13220175f48: Already exists
688fc9764294: Already exists
41acdcd248b2: Already exists
68ac674d8156: Already exists
970a46caeefb: Downloading [==========>                                        ]  270.6MB/1.25GB
464ef57afe4f: Downloading [===========>                                       ]  102.7MB/440.6MB
```

b) and inside the container locate simcam_sdk_root/train folder

```
root@7495d684b263:/# cd /home/inomjon/Projects/SIMCAM_SDK_v1.0.0/train/
root@7495d684b263:/home/inomjon/Projects/SIMCAM_SDK_v1.0.0/train#
```

c) Comment out “gpu 0” line in train.sh script.

```
#!/bin/sh
if ! test -f prototxts/train.prototxt ;then
        echo "error: train.prototxt does not exist."
        echo "please use the gen_model.sh to generate your own model."
        exit 1
fi
mkdir -p snapshot
/opt/movidius/ssd-caffe/build/tools/caffe train -solver="solver_train.prototxt" \
-gpu 0
```

3. To start training run train.sh script:

   ./train.sh

```
inomjon@inomjon-C-H310M-K-PRO: ~/Projects/SIMCAM_SDK_v1.0.0/train
I0325 16:18:33.705937 13035 net.cpp:228] conv1_2 does not need backward computation.
I0325 16:18:33.705941 13035 net.cpp:228] conv1_1/relu does not need backward computation.
I0325 16:18:33.705945 13035 net.cpp:228] conv1_1/scale does not need backward computation.
I0325 16:18:33.705948 13035 net.cpp:228] conv1_1/bn does not need backward computation.
I0325 16:18:33.705952 13035 net.cpp:228] conv1_1 does not need backward computation.
I0325 16:18:33.705956 13035 net.cpp:228] conv1_conv1/relu_0_split does not need backward computation.
I0325 16:18:33.705961 13035 net.cpp:228] conv1/relu does not need backward computation.
I0325 16:18:33.705965 13035 net.cpp:228] conv1/scale does not need backward computation.
I0325 16:18:33.705968 13035 net.cpp:228] conv1/bn does not need backward computation.
I0325 16:18:33.705971 13035 net.cpp:228] conv1 does not need backward computation.
I0325 16:18:33.705976 13035 net.cpp:228] data_data_0_split does not need backward computation.
I0325 16:18:33.705979 13035 net.cpp:228] data does not need backward computation.
I0325 16:18:33.705982 13035 net.cpp:270] This network produces output detection_eval
I0325 16:18:33.706091 13035 net.cpp:283] Network initialization done.
I0325 16:18:33.706586 13035 solver.cpp:75] Solver scaffolding done.
I0325 16:18:33.706887 13035 caffe.cpp:251] Starting Optimization
I0325 16:18:33.706892 13035 solver.cpp:294] Solving tiny-ssd
I0325 16:18:33.706895 13035 solver.cpp:295] Learning Rate Policy: multistep
I0325 16:18:33.709753 13035 blocking_queue.cpp:50] Data layer prefetch queue empty
I0325 16:19:08.639659 13035 solver.cpp:243] Iteration 0, loss = 42.7464
I0325 16:19:08.639771 13035 solver.cpp:259]     Train net output #0: mbox_loss = 42.7464 (* 1 = 42.7464 los
s)
I0325 16:19:08.639781 13035 sgd_solver.cpp:138] Iteration 0, lr = 0.001
```

That is all your object detection model is started training. We suggest that you should train the model until loss value become below two. You can get a snapshot in 1000 steps. Total training lasts 120000 steps.

After all, you will get **simcam_iter_xxxxx.caffemodel** inside snapshot folder. And **deploy.prototxt** file inside prototxts folder.

To deploy your trained model on SIMCAM products you can follow “Guide of toolchain installation and usage” document.