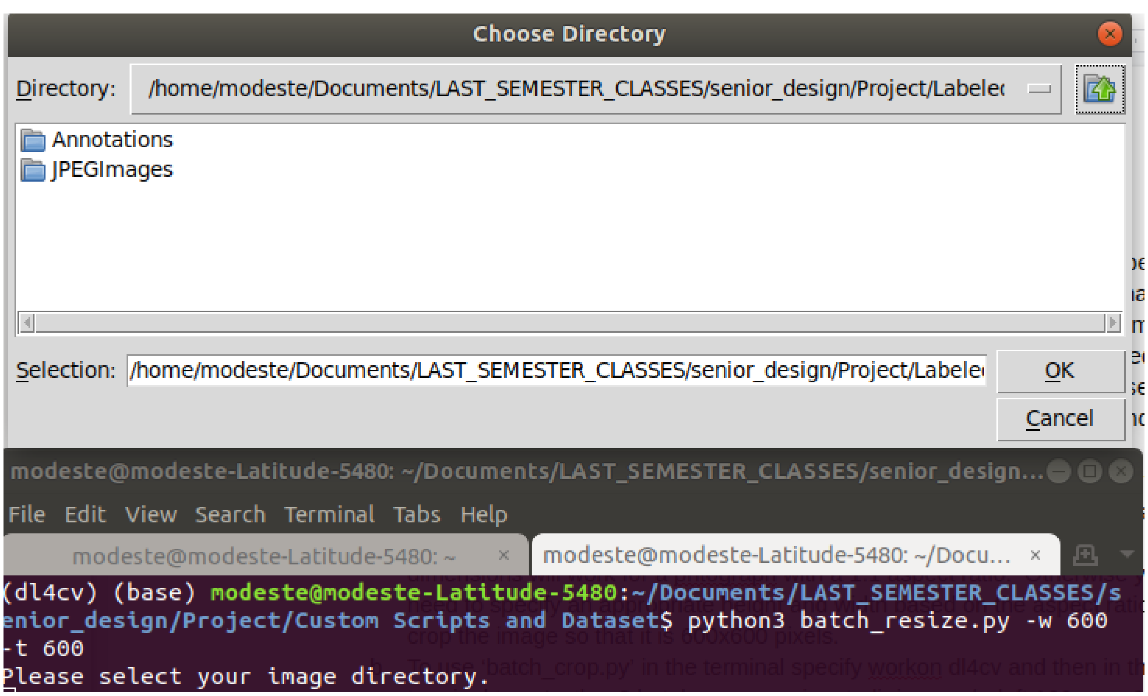
Preface

This documentation is largely based off of the tutorial series '[Training Custom Object Detector - TensorFlow Object Detection API Tutorial]' with some additional tools and tricks. When attempting to do this if someone gets stuck they can reference the tutorial series. Note that you will first need to have Cuda for GPU acceleration installed as well as Tensorflow and OpenCV installed. I have created documentation for installing Cuda 9, Tensorflow r1.12, and supplementary instructions for installing OpenCV 4.0.0 in supporting documentation. Note that these can also be found in the online repository

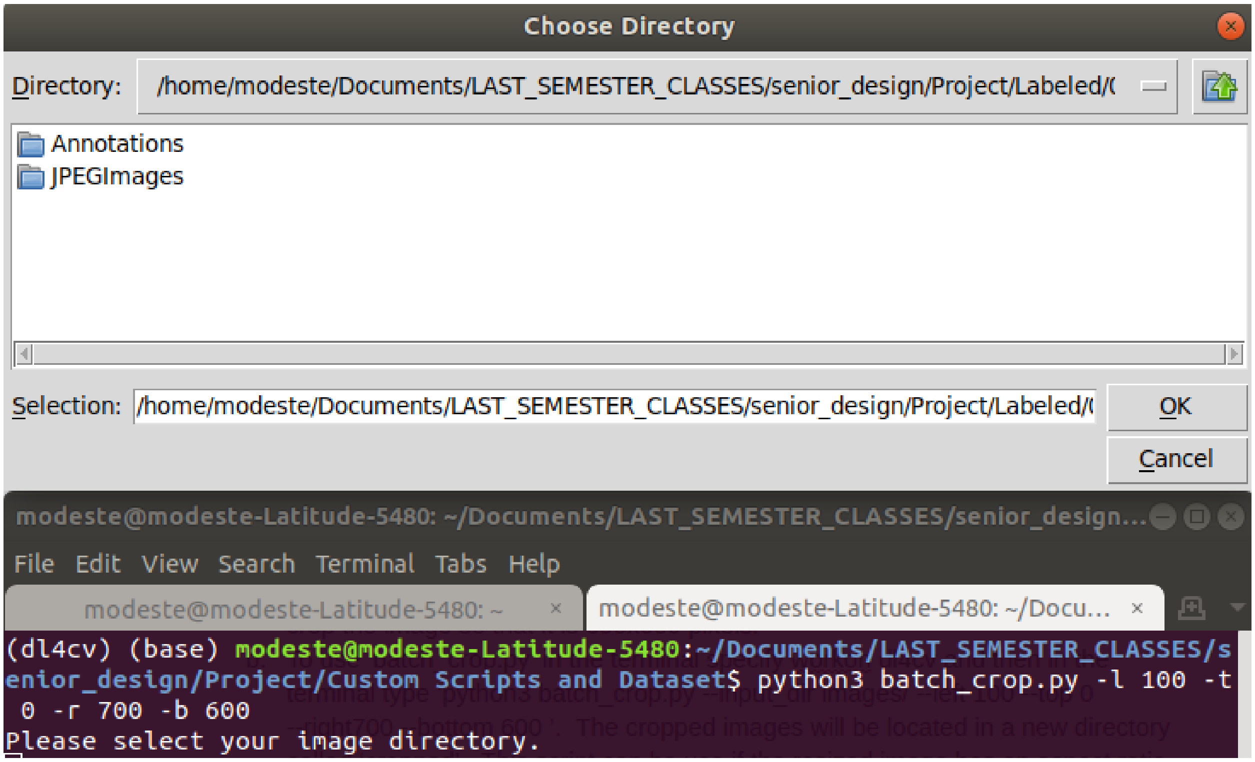
Dataset Preprocessing

1. It's important to train at a resolution similar to that that the actual detection will be performed at. Also note that images at lower resolutions will result in training that is significantly faster. For the Final SnapCrack dataset we collected images from multiple datasets as well as a few images of our own. We resized them and then cropped them so that each image is 600x600 pixels. If adding images to the preexisting dataset it is recommended to use the 'batch_resize.py' and 'batch_crop.py' scripts resize and crop images respectively.

    a. To use 'batch_resize.py' in the terminal specify workon dl4cv and then in the terminal type 'python3 batch_resize.py --width 600 --height 600', alternatively you can enter 'python3 batch_resize.py -w 600 -t 600'. The resized images will be located in a new directory called 'resized'. The sample dimensions will work for a phtograph with a 1:1 aspect ratio. Otherwise you will need to specify an appropriate height and width based on the aspect ratio then crop the image so that it is 600x600 pixels.



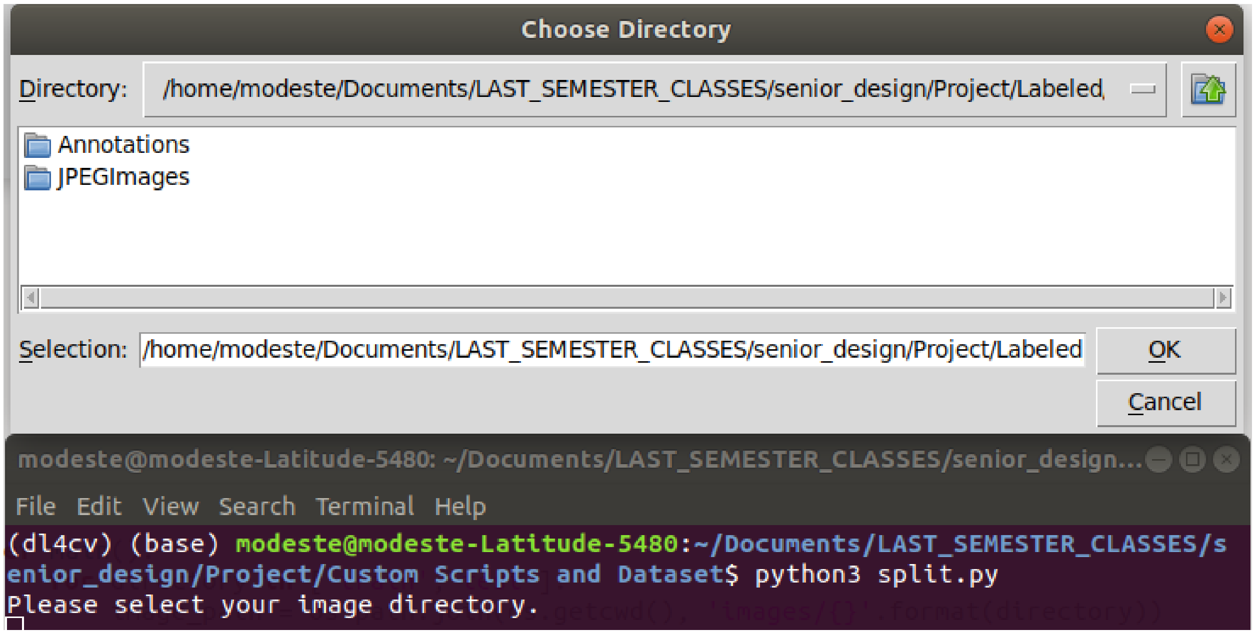
b. To use ‘batch_crop.py’ in the terminal specify workon dl4cv and then in the terminal type ‘python3 batch_crop.py --left 100 --top 0 --right 700 --bottom 600 ’. Alternatively you may enter ‘python3 batch_crop.py -l 100 -t 0 -r 700 -b 600‘. The cropped images will be located in a new directory called ‘cropped’. This script can be use if the resized image has an aspect ratio different than 1:1



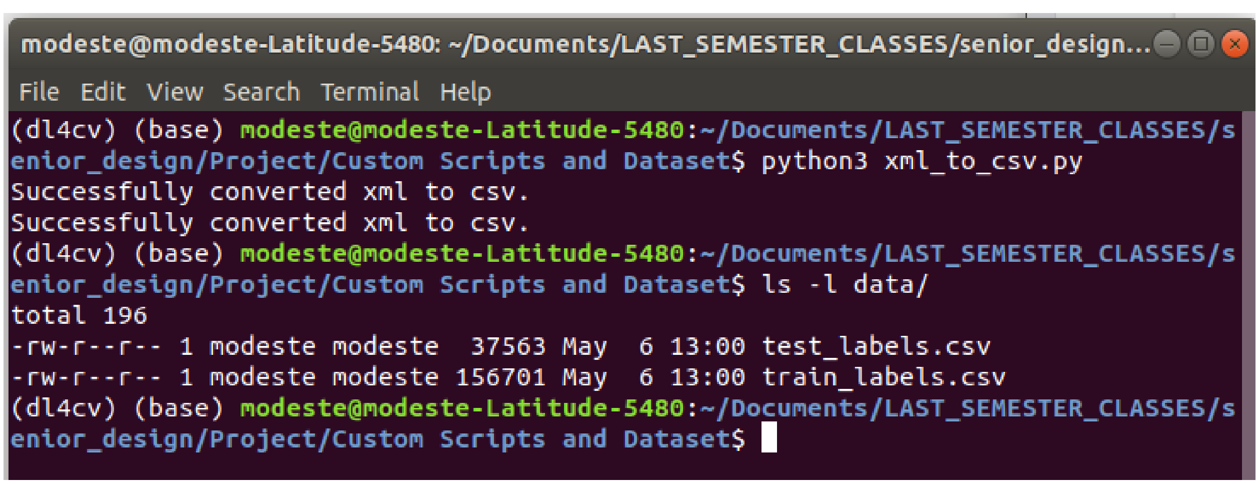
c. After this you can label the images and add them to your final dataset

Splitting the Dataset and Generating the tfrecord

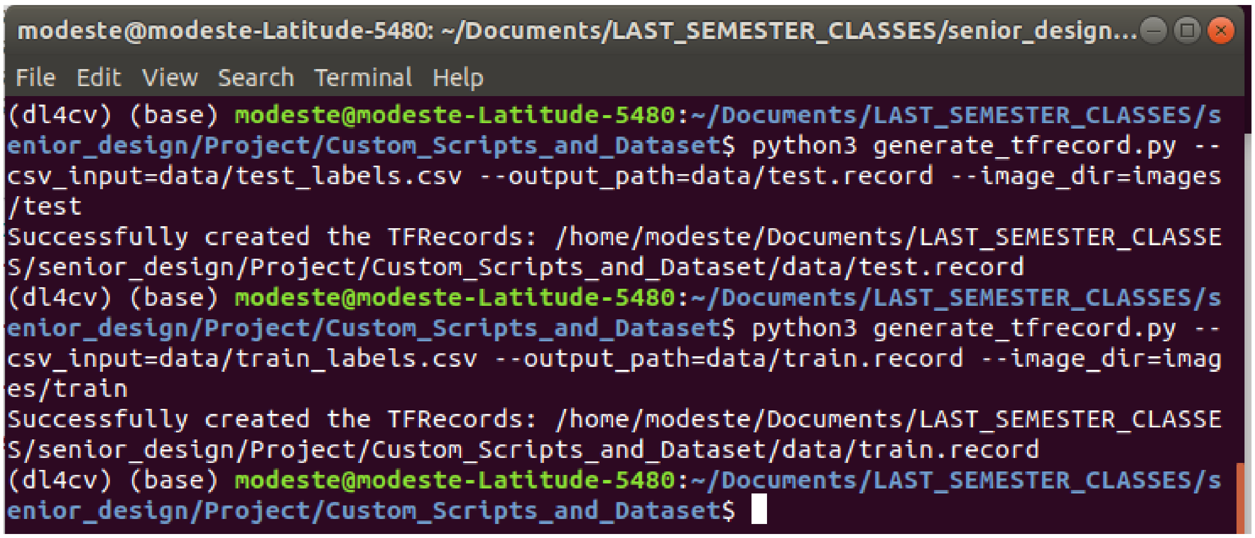
1. The split.py script randomly splits the dataset into test and train datasets. When you run this script these datasets appear in a new image directory. Additionally there will be a version of the dataset with the data it was created in an image_archive directory.



2. Next we will convert the xml files to csv files. Make sure the images directory and a data directory is in your current working directory then run ‘python3 xml_to_csv.py’ and the corresponding csv files for the testing and training labels will be generated in the data directory.



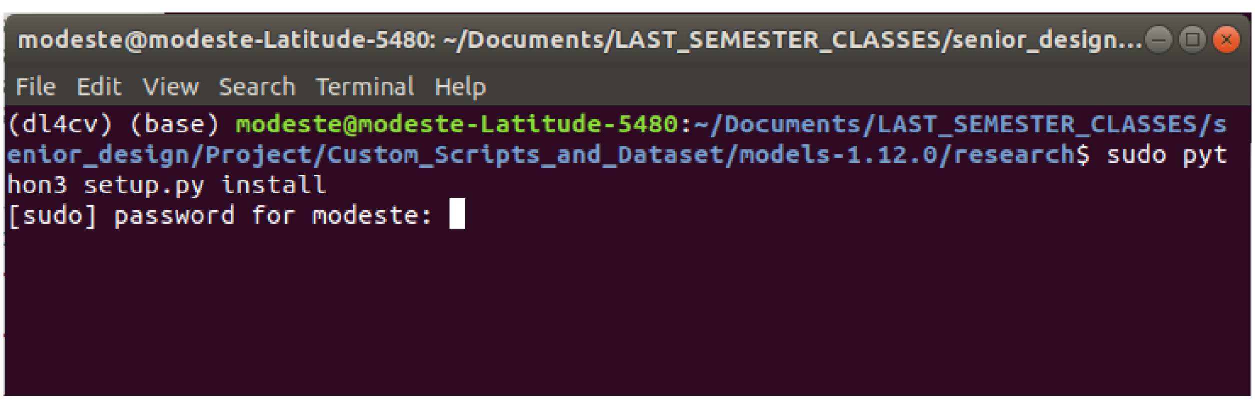
3. To change or add labels, this is done in the generate_tfrecord.py script. Here we have labeled our L0, T0, and P0 for longitudinal cracks, transverse cracks, and potholes respectively. These correspond with the labeling of the final custom dataset. We need 2 tf records, one for the testing data and one for the training data. We use the respective command line arguments to generate these. ‘python3 generate_tfrecord.py --csv_input=data/test_labels.csv --output_path=data/test.record --image_dir=images/test’ and ‘python3 generate_tfrecord.py --csv_input=data/train_labels.csv --output_path=data/train.record --image_dir=images/train’

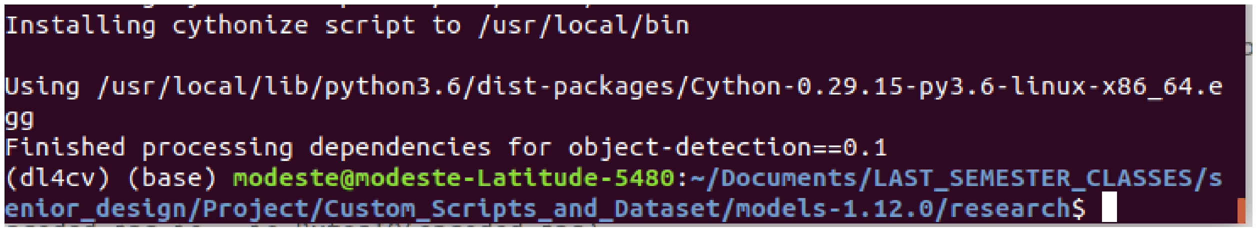


Installing the object detection API

1. Next follow installation instructions for installing the Tensorflow object_detection API here -> https://github.com/tensorflow/models/blob/r1.12.0/research/object_detection/g3doc/installation.md note that I used protoc version 3.4.0.

2. Next in the models-1.12.0/research directory we use command ‘sudo python3 setup.py install’





Setting up the CNN model config file

1. Can use any of the mobilenet models. All of the configs are located here. There are a number of parameters that need to be changed in the config. These include num_classes, batch_size… from there every "PATH_TO_BE_CONFIGURED" will need to be changed. From the config for ssd_mobilenet_v1_pets these config parameters include:
   - Fine_tune_checkpoint
   - input_path and label_map_path for train_input_reader
   - input_path and label_map_path for eval_input_reader

2. Note that we need to create a .pbtxt file in our training directory that corresponds to the labels in the generate_tfrecord.py script.

3. The config file we used to train has already been downloaded and is already configured. Models can be downloaded here, to download the model that we used for fine tuning in our project you can enter the following in the terminal. Once entered extract the model.
   - wget http://download.tensorflow.org/models/object_detection/ssd_mobilenet_v1_coco_11_06_2017.tar.gz

Training the Object Detection Model

4. Now copy the following folders into your models-1.12.0/research/object_detection directory: data/, images/, training/, ssd_mobilenet_v1_coco_11_06_2017/, and ssd_mobilenet_v1_pets.config. You can see more about running training locally here.

5. In the models-1.12.0/research/object_detection directory run the following command
   - ```
     python3 model_main.py \
       --pipeline_config_path=ssd_mobilenet_v1_pets.config \
       --model_dir=training/ \
       --num_train_steps=150000 \
       --sample_1_of_n_eval_examples=1 \
       --alsologtostderr
     ```
     ```

6. To view in tensorboard you can type the following
    - tensorboard --logtostderr --logdir=eval/

```
(dl4cv) (base) modeste@modeste-Latitude-5480:~/Documents/LAST_SEMESTER_CLASSES/senior_design/Projec
t/Custom_Scripts_and_Dataset/models-1.12.0/research/object_detection$ python3 model_main.py \
>     --pipeline_config_path=ssd_mobilenet_v1_pets.config \
>     --model_dir=training/ \
>     --num_train_steps=150000 \
>     --sample_1_of_n_eval_examples=1 \
>     --alsologtostderr
```

```
Use `tf.data.Dataset.batch(..., drop_remainder=True)`.
2020-05-06 17:26:16.281581: E tensorflow/stream_executor/cuda/cuda_driver.cc:300] failed call to cu
Init: CUDA_ERROR_UNKNOWN: unknown error
2020-05-06 17:26:16.281644: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:163] retrieving C
UDA diagnostic information for host: modeste-Latitude-5480
2020-05-06 17:26:16.281659: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:170] hostname: mo
deste-Latitude-5480
2020-05-06 17:26:16.281702: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:194] libcuda repo
rted version is: 440.59.0
2020-05-06 17:26:16.281738: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:198] kernel repor
ted version is: 440.59.0
2020-05-06 17:26:16.281749: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:305] kernel versi
on seems to match DSO: 440.59.0
creating index...
index created!
creating index...
index created!
Running per image evaluation...
Evaluate annotation type *bbox*
DONE (t=1.61s).
Accumulating evaluation results...
DONE (t=0.27s).
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.000
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.002
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.000
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.001
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.000
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.001
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.001
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.009
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.043
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.001
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.019
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.112
```

7. To evaluate a model use the following
    - python3 model_main.py \
      --run_once \
      --checkpoint_dir=training/ \
      --model_dir=eval/ \
      --pipeline_config_path=training/ssd_mobilenet_v1_pets.config