**Abstract**

This project is to implement an objection detection function which is deployed on dash cameras of vehicles. We apply Yolov3 as our object detection algorithm, since it is capable to achieve real-time image processing and outperforms other object detection algorithms (e.g., SSD, Fast-RCNN) in processing speed and accuracy.

**Algorithms Description**

Yolov3

There are 252 layers in Yolov3 in total, including 75 Conv2D layers, 72 Batch Normalization layers, 72 Leaky ReLU activation layers, 23 Add layers, 5 2D Zero Padding layers, 2 Concatenate layers, 2 2D Up-sampling layers, and a very last Input layer. Yolov3 usually combine some layers as a unit. A Cov2D usually is followed by a BatchNormalization layer with a LeakyReLU layer behind. These three layers form a Darknet\_Conv2D (DBL) unit. A Residual (Res) unit consist of 2 DBL units and an Add layer which adds the DBL output and the input of Res unit together. In addition, Yolov3 designs Resn units, which stack a Zero Padding layer and a DBL unit with n numbers of Res units together.

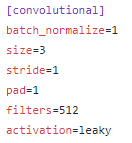
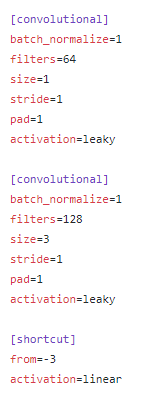


Figure 1. DBL unit

Figure 2. Res Unit

**Training process**

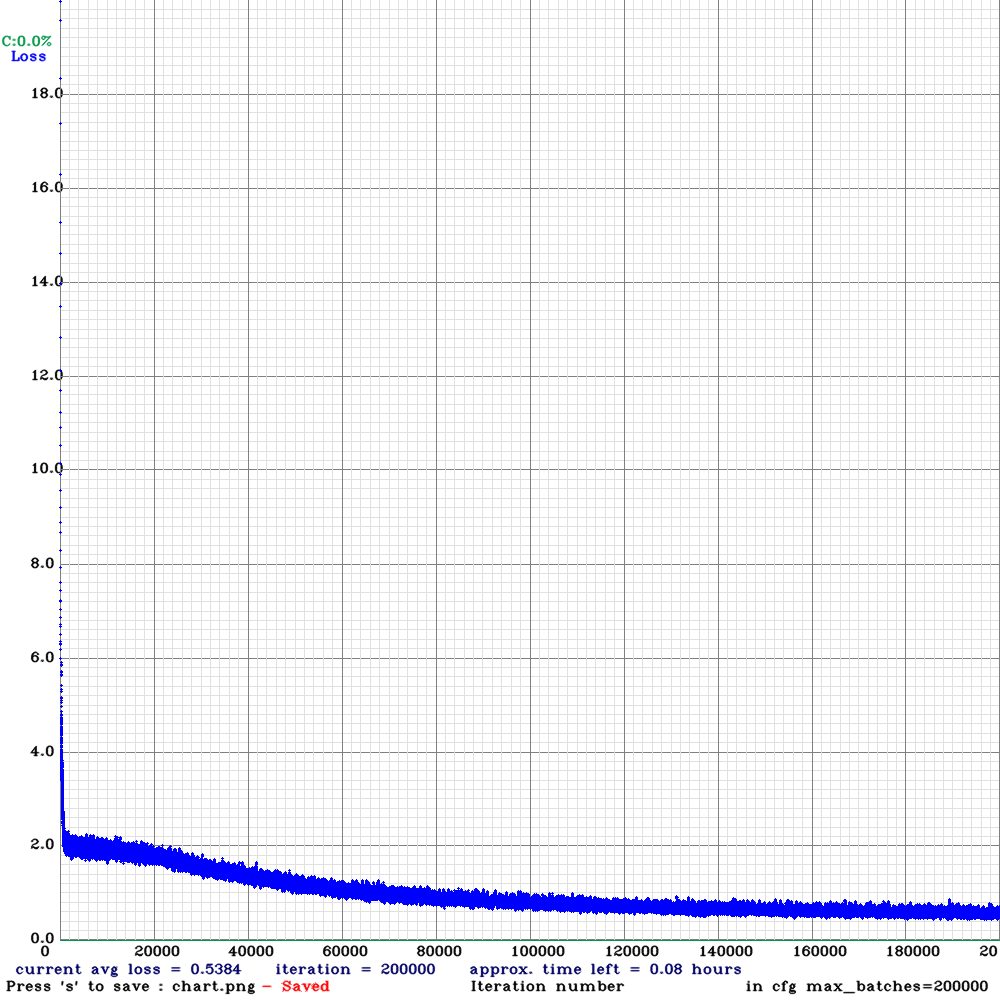
Training dataset preparation

For the training data, we used some part of the VOC dataset which contains similar on road scene. For most part of our training data, we get some urban road monitoring videos and some dashcam video of from the Internet. First, we use ffmpeg to convert these videos into images, and then we label them using the LabelImg to generate the .xml file just like the VOC dataset. One of the image that we labeled is shown below.



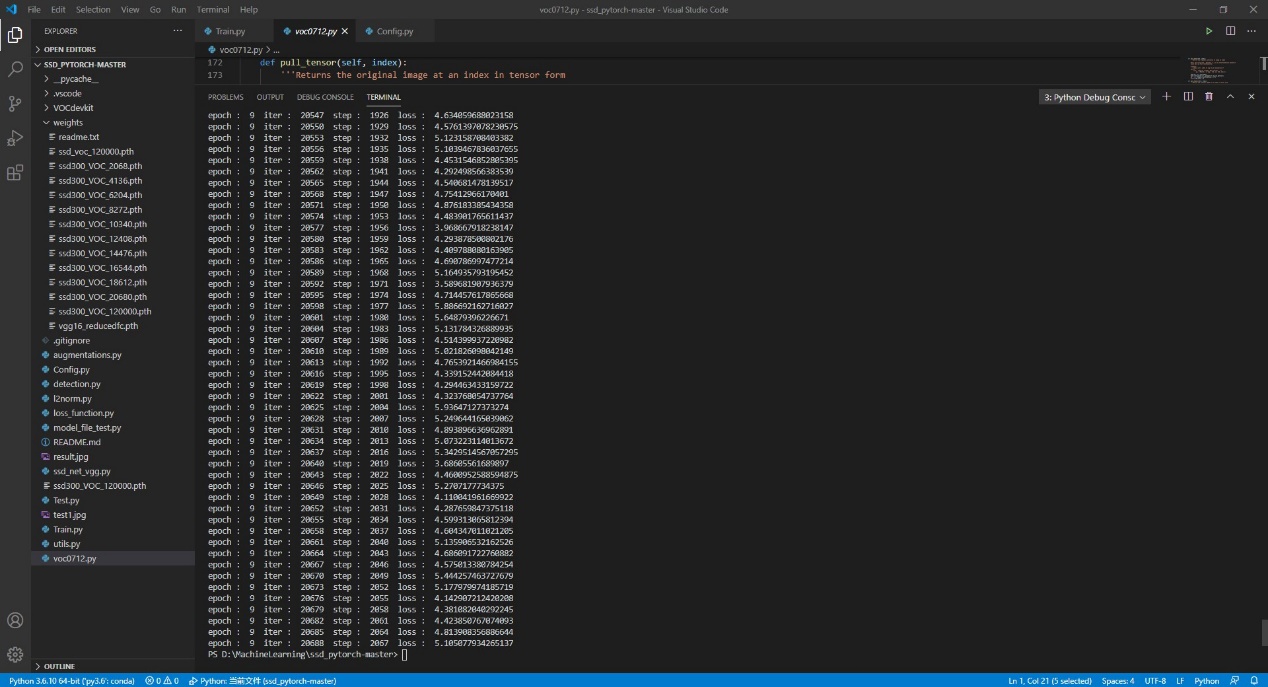
Training YOLO v3

First, we need to compile the YOLO v3 from [darknet](https://github.com/AlexeyAB/darknet) . After modify the configuration file, which is .data and yolo.cfg to fit our training category. Then, we generate download the pretrain model and start training. We trained 200000 times and the final loss is below 0.5. The training process is shown below.



Training SSD

For SSD, we use the pytorch version of SSD ([ssd.pytorch](https://github.com/amdegroot/ssd.pytorch)). After modify the config.py; VOC0721.py; train.py; eval.py and ssd.py, we started trainig.



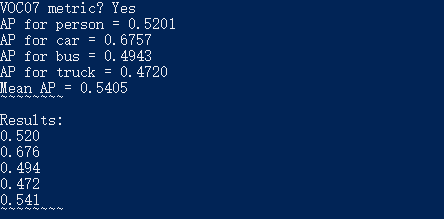
**Evaluation**

We used mAP to evaluate the training result.

In yolo v3, we get into some trouble that our rename file will mismatch the pictures and the annotation file. So we need to retrain the model.

In SSD, we successfully get the result. But, SSD does not perform well when there are multiple objects overlapping. One of the test result and the mAP are shown below.





**Expectation**

In order to run YOLO v3 on a dashcam, we need to quantize it. After finish the training, we believe that it will fit all the working scenarios.