Object Detection on Bully and Victim

CPSC8810 Deep Learning Final Report

Chong Meng, chongm@g.clemson.edu Ziheng He, zihengh@g.clemson.edu

***Abstract*—In this report we present our procedure of applying object detection on 9 bullying class of images to identify bully and victims. First, we explain our strategy of labeling images. After that, we introduce the model we used for object detection. We then present our procedure of training and evaluating. A brief instruction about how to run our code will be given at the end.**

***Keywords*—Object Detection, VGG-based SSD, Cyber bullying**

I. Introduction

After achieving an approximately 70% accuracy on classifying different bullying in the mid-term task, we now focus on detect the predictor and victim in bullying images.

II. Label images

A. About LabelImg

LabelImg[1] is a tool widely used to draw box of classes on images. It saves the label information in VOC format.

B. Label bully and victim

Object names are set as 'bully' and 'victim'. All annotations are uploaded to folder. For images do not contain a victim, like Figure 1, we only box the bully object. For images about people bullying each other, like Figure 2, we identified both as bully. Meanwhile, we deleted dozens of images that have nothing to do with the title.

Figure 1

Figure 2

III. VGG-based SSD model

We learned from online source code of a SSD object based on VGG[2]. The repository contains a TensorFlow re-implementation of the original Caffe code. At present, it only implements VGG-based SSD networks (with 300 and 512 inputs), but the architecture of the project is modular, and should make easy the implementation and training of other SSD variants (ResNet or Inception based for instance). Present TF checkpoints have been directly converted from SSD Caffe models.

IV. Training process

A. Spliting data

We randomly split images from 9 classes into training set and test set by the ratio of 4 to 1. We mixed up images into one group instead of leaving them in 9 different groups. We deleted all images that are not jpeg type and do not have an xml file.

B. Convert Data

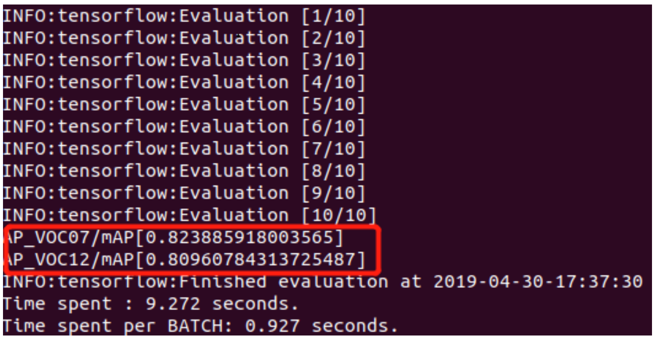
Before training, we convert all images and xml files into tfrecord format data. Tfrecord is a binary data combined images and labelling information and can be read much more efficiently from disk.

C. Training from pretrained model

Our training process started from an existing checkpoint 'SSD-300 VGG-based'[3]. It is a vgg-based SSD model trained for 120000 iterations, detecting for 21 classes of objects. Period work will be saved as a checkpoint after a certain number of steps and we are able to evaluate a checkpoint with a certain index.

V. Evaluating

The evaluating script can give us a result of mAP on all images of dataset. As we can see in Figure 3, the mAP is about 0.82.



```
INFO:tensorflow:Evaluation [1/10]
INFO:tensorflow:Evaluation [2/10]
INFO:tensorflow:Evaluation [3/10]
INFO:tensorflow:Evaluation [4/10]
INFO:tensorflow:Evaluation [5/10]
INFO:tensorflow:Evaluation [6/10]
INFO:tensorflow:Evaluation [7/10]
INFO:tensorflow:Evaluation [8/10]
INFO:tensorflow:Evaluation [9/10]
INFO:tensorflow:Evaluation [10/10]
\P_VOC07/mAP[0.823885918003565]
\P_VOC12/mAP[0.80960784313725487]
INFO:tensorflow:Finished evaluation at 2019-04-30-17:37:30
Time spent : 9.272 seconds.
Time spent per BATCH: 0.927 seconds.
```

Figure 3

We can also view the prediction on images. We draw prediction on all images in the test set. Here we show four images (Figure 4 to Figure 7) to present predictions from a checkpoint. Class 1 stands for victim and 2 stands for bully.

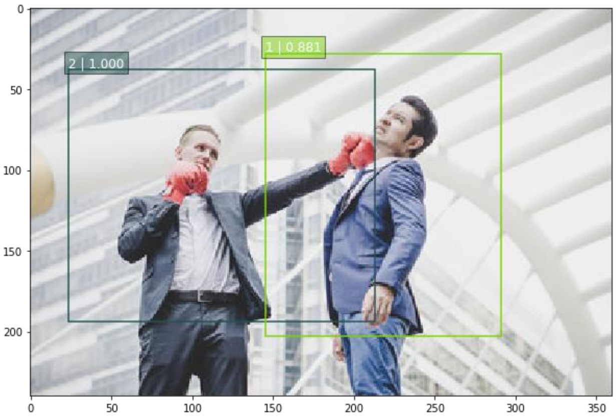


Figure 4

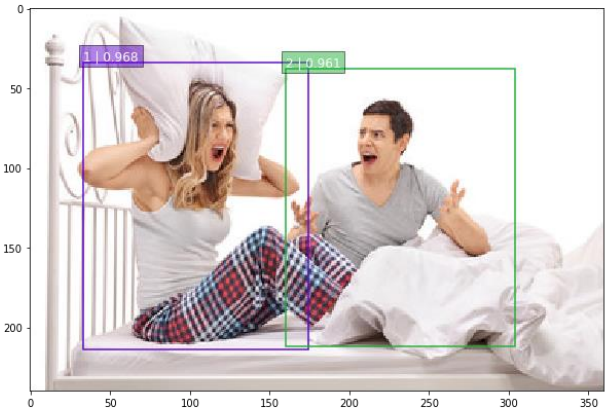


Figure 5

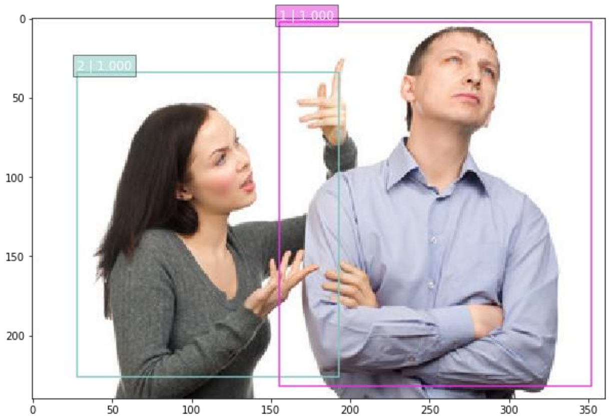


Figure 6

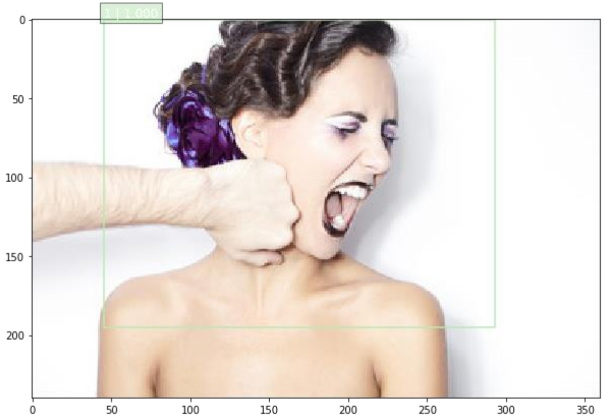


Figure 7

VI. Conclusion and Future Work

Since we don’t have enough time, we trained the data with an already existing model. Most of the predictions are quite good. However, this work can not be applied for further research as the model isn’t built by ourself.

We are trying to build our own SSD model and are looking forward to trying other deep learning techniques, like attention, to analyze the body language and facial movement of different characters. It may improve the mAP value.

VII. Instruction about Running Our Code

We have provided a python script “test.py” stored in the folder “results”, it accepts one command line argument which is the path of the test set folder. After execution of the “test.py”, it will store all the labeled images in the ‘labeledImages’ folder at the same level, i.e. ‘results/labeledImages’. You can check our results from these labeled images.

References

[1] https://github.com/tzutalin/labelImg

[2] https://github.com/balancap/SSD-Tensorflow

[3] https://drive.google.com/file/d/0B0qPCUZ-3YwWUXh4UHJrd1RDM3c/view