Human Recognition by Biometric Methods

Face detection and classification

Mikołaj Małkiński

7 June 2018

1 Introduction

The goal of this task was to design, implement and train YOLO object detection system (YOLOv3 was chosen) for detecting faces of Polish rappers. Then, the extracted faces should be piped to previously trained model in order to classify them.

The original dataset defined around 700 bounding boxes for faces. Several boxes could refer to the same image. The dataset was split automatically into two parts by Keras, using 80% of the data for training and remaining 20% for validation.

The models are resizing the images before processing, so data of arbitrary size can be used. The final mean average precision at 0.5, evaluated on the whole dataset (containing images used for training), is **97.26%**.

The project consists of following python scripts:

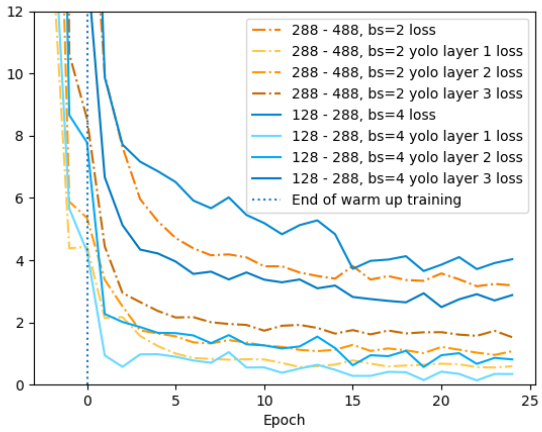
- *yolo_detect_classify* - detects faces in image and predicts their classes,
- *yolo_evaluate* - evaluates model accuracy (Mean Average Precision at 0.5) using specified data directory with the same structure as provided dataset,
- *yolo_reproduce_results* - creates plots shown in the report,
- *yolo_train* - trains detection system using specified configuration file.

2 Analysis of results

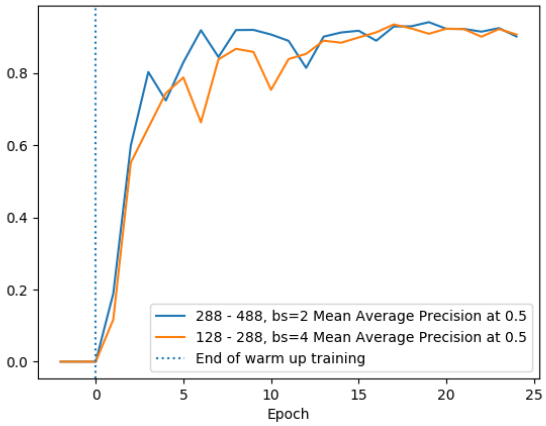
Figure 1 describes configurations used for analysis. Min and max input size define the range to vary the size of the image during training (should be a multiple of 32). Figure 2 presents comparison of results using first and second configurations. It can be observed that although bigger input sizes resulted in faster mAP improvement, the final mAP@0.5 after 24 epochs rounded to 2 digits after the comma, was the same - 91%. Obviously, mentioned mean average precisions are evaluated purely on the validation set, which contains only data not used during training. However, Figure 3 shows that the input size can't be too small. Unfortunately, the chosen model requires a substantial amount of graphics card memory (nearly 4GB were available), so larger batch sizes couldn't be tested. Finally, in the Figure 4 the whole training process of the best performing configuration is presented.

Configuration	Min input size	Max input size	Batch size
1	128	288	4
2	288	448	2
3	32	96	4

Figure 1: Details of configurations



(a) Loss



(b) Mean Average Precision at 0.5

Figure 2: Comparison of two configurations

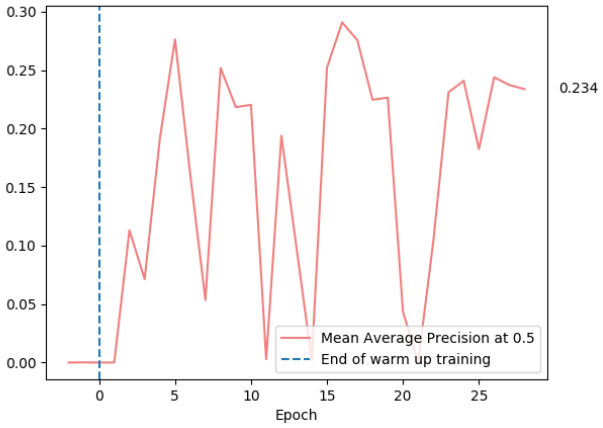
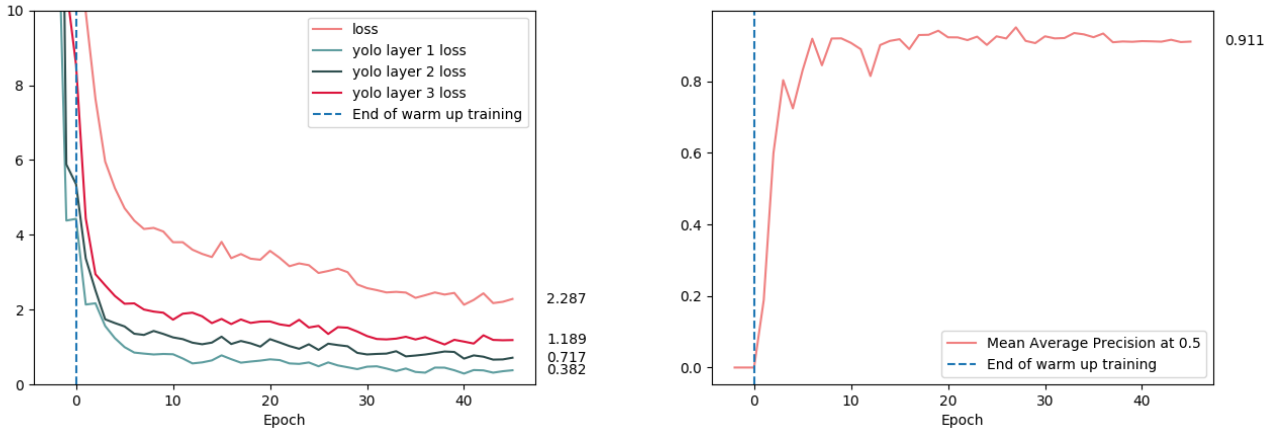


Figure 3: Mean Average Precision at 0.5 when input sizes are too small



(a) Loss

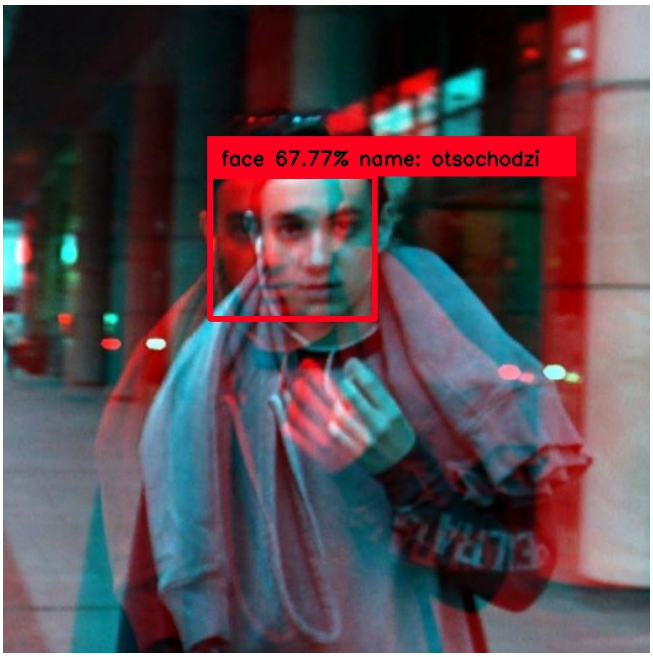
(b) Mean Average Precision at 0.5

Figure 4: Results of the best performing configuration

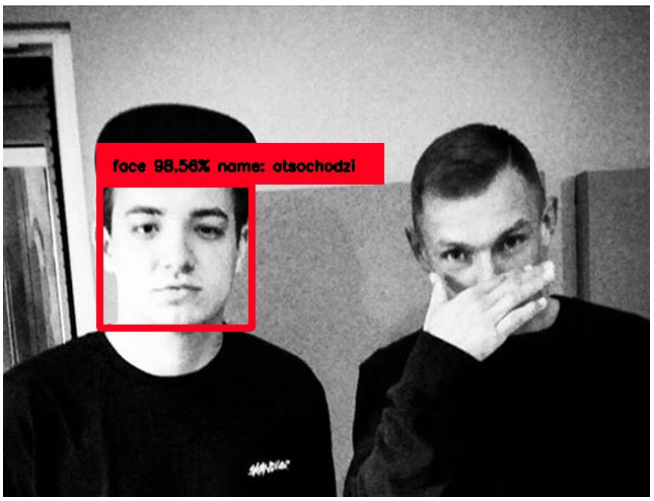
3 Examples of detection and classification

The results of trained models are certainly satisfactory - 91% mAP@0.5 for detection and close to 97% for classification. Figures 5 and 6 shows how the whole pipeline performs on test images. It is worth observing, that this model works well with images of different resolution as well as objects varying in size. Obviously, since the classificator was trained to recognize only 4 people, some of the faces have wrong labels, simply because they weren't present in the dataset. Example of executed command to process the images, is presented below:

```
python yolo_detect_classify.py -c yolo/configs/config_faces_01_06_15-25.json
-i ~/datasets/raw_dataset/ostr/505.jpg
```



(a) Blurry face



(b) Partly hidden face

(c) Woman face with longer hair

(d) Small faces

(e) Faces of different size

(f) Similar faces

Figure 5: Examples of detection and classification

(a) Partly covered faces

(b) Clearly visible face

(c) Face with glasses

(d) Normal faces

Figure 6: Examples of detection and classification