Head Localization Using YOLOv3

Simone Sandler
Data Science and Engineering
University of Applied Sciences Upper Austria
Hagenberg, Austria
simone.sandler@students.fh-hagenberg.at

Sophie Kaltenleithner
Data Science and Engineering
University of Applied Sciences Upper Austria
Hagenberg, Austria
sophie.kaltenleithner@students.fh-hagenberg.at

***Abstract*—This paper deals with the localization of human heads in images using YOLOv3. The model was trained on RGB images from the *OpenImages* data set. Different data augmentation and preprocessing strategies were applied and their influence on the final model quality were analyzed. The best results with a recall of 0.74 were obtained when data augmentation and a median filter were applied. The final model is generally able to handle larger groups of people, heads from different angles and different skin colors.**

***Index Terms*—localization, object detection, neural network, YOLO, data augmentation, preprocessing**

I. Introduction

A. Problem

The localization of heads in 2D images can be useful for a number of real world problems: people tracking for social behavior measurement, bottleneck detection, queue measurement, interesting route detection [1] as well as people counting for e.g. video surveillance [2]. There is a variety of challenges that arise when tackling this problem: Crowded environments, occlusion, image resolution and point of view. [2].

B. Goal

The goal is to use a neural in network in order to recognize human heads in color images. The localizations as seen in Figure 1 will be indicated by a bounding box and the confidence of the model in the classification.

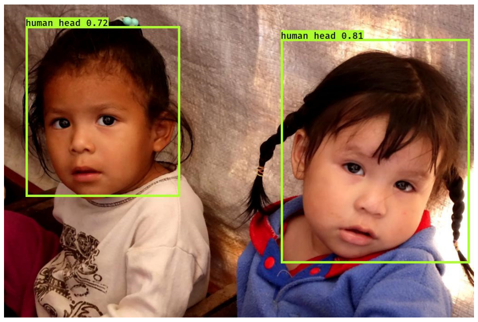

Fig. 1: Bounding boxes with confidence scores.

C. Data

The data set used is a subset of the *Open Images Dataset V6*. This data set contains approximately 1.9 million images with 16 million bounding boxes for 600 object classes. The boxes have been drawn to a large extent by professional annotators, ensuring accuracy and consistency. This, combined with the high diversity of the images and inclusion of complex scenarios containing multiple objects, makes for a high quality database. [3]

For the task of head detection, 5000 random images with bounding boxes for the class ”human head” [4] were downloaded and used for training and testing of the final model.

II. Methodology

This section discusses the methods used for generating and augmenting the training and test data, the metrics used for quality measurement as well as the actual process of training the model.

A. YOLO

The problem was solved by using a *YOLOv3* network. *YOLO* is an acronym for ”You Only Look Once”. This network combines bounding-box predictions of objects with their classifications and confidence scores [5]. It aims to make fast predictions, to learn the general representation of objects and to predict less false positives on background [5]. What sets *YOLO* apart from other detection methods like *R-CNN* is that far fewer bounding boxes are proposed. Furthermore *YOLO* combines all the parts necessary for localization into one while a *R-CNN* uses multiple stages, namely *selective search* for bounding-boxes, a *CNN* for features extraction and a *SVM* to score the boxes [5].

The basis of the work build the *GitHub* repositories *TrainYourOwnYOLO* [6] and *keras-yolo3* [7] which enable training of a *YOLO* net using *Keras*. They have been altered in terms of preprocessing, data augmentation, and evaluation of the prediction results.

A transfer-learning approach is used for training the models. For this, the models are initialized with pre-trained weights from the *Darknet YOLO* website [8]. As these weights do already contain valuable information, they are frozen during the first training stage in order to avoid destroying them, which is a common method in transfer learning [9]. Some new trainable layers are added to the pre-trained ones. The purpose of these layers is to use the output of the pre-trained layers as input to adapt their weights for the current problem [9]. In the final stage all layers are unfreezed, which enables fine tuning of the model.

B. Preprocessing

The data annotations are downloaded in *Pascal VOC XML* format [10] using the *Python* module *openimages* [11]. However, for training the data is needed in *YOLO Darknet TXT* [12] format. Thus, the filenames and bounding boxes are extracted from the original files and accumulated into one single training file, with each line being structured the following way: *filepath bb1_xmin, bb_1ymin, bb1_xmax, bb1_ymax, class ... bbn_xmin, bbn_ymin, bbn_xmax, bbn_ymax*. The "class" value hereby being always 0, as it is a single-class problem.

The data is then split 90:10 into train/validation and test data. Finally, while training and testing the input images can be altered by either using a median filter with kernel size = 3, histogram equalization or both.

C. Data Augmentation

Data augmentation refers to methods that generate new data as an addition to the original training set [13].

In this work the data augmentation is done in a real-time fashion, where the data is transformed during the training process. It was necessary to use methods, where one is able to preserve the bounding-boxes. Either the boxes are not affected at all, as it is the case with color transformations, or the boxes can be transformed in the same way as the data. Therefore, the following methods were considered:

- Zoom
- Translation
- Flipping
- Conversion to gray-scale
- Image distortion by assigning a new random aspect ratio
- Random color distortion by adjusting the HSV-values

D. Quality Metrics

The *YOLO* model is trained based on a loss function, which is the sum of the following three loss metrics [14]:

- Classification loss: squared error of the class conditional probabilities for each class (basically irrelevant here as it is a single-class problem)
- Localization loss: Errors between the localization and sizes of the predicted boundary box and the ground truth
- Confidence loss: Measures the loss of "objectness" in a box, meaning how confident the model is in it's prediction.

For evaluating the predictions on the test set, the following metrics are used:

- Mean intersection over union (IOU): The IOU is a metric for the similarity of the predicted bounding-boxes and the ground truth boxes [15].

$$IOU = \frac{Area\ of\ Overlap}{Area\ of\ Union}$$

- Non-Zero intersection ratio (nZIR): The ratio of predicted bounding-boxes that have a non-zero IOU.

$$nZIR = \frac{count(IOU > 0)}{count(Boxes)}$$

- Recall: Fraction of true positives over actual positives, where true positives (TP) are defined as having an IOU of $>=$ 0.4 and false positives (FP) are defined as having an IOU of $<$ 0.4 [16].

$$Precision = \frac{TP}{TP + FN}$$

III. Results

In this section the results obtained using different training configurations are described and examined. The influence of data augmentation and pre-processing using histogram equalization and median filter is analyzed.

A. Training With Data Augmentation

For this run, the number of epochs for the first and second stage were set to 70 resulting in a total 140. After 60 epochs in the second stage early stopping was reached. The mean IOU lies at 0.57, recall at 0.73 and the nZIR is 0.76. The validation loss after stage one is 22.08 and the final validation loss is 14.06 as can be seen in Figure 2. In the final stage, one epoch had a huge validation loss resulting in a spike in the graph. As this spike compresses the rest of the data, a median filter was applied to the loss of the final stage, resulting in the third graph of Figure 2. Because of the long training time, the epochs of the next runs where reduced to 20 in each stage, as the strongest loss decrease occurs in the first epochs.

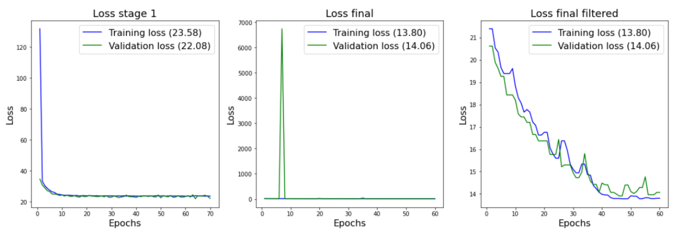

Fig. 2: Losses with data augmentation.

B. *Training Without Data Augmentation*

Without data augmentation the IOU is 0.44, recall 0.57 and nZIR is 0.61. The training loss is decreasing while the validation loss increases, as can be seen in figure 3. This behavior means that the model is overfitting. This proves that data augmentation has a great influence on the generalization ability and therefore the quality of the model.

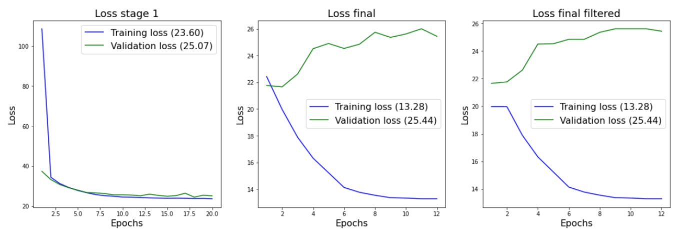

Fig. 3: Losses without data augmentation

C. *Training With Data Augmentation and Median Filter*

Applying a median filter to the images lead to the best results. In this case, the IOU is 0.57, the recall 0.74, the nZIR is 0.79 and the final loss is 14.80. This means that by using the median filter the results after just 40 epochs are better than without one and 140 epochs. Figure 4 shows that the model was still learning in the last epoch, which means that by increasing the number epochs the obtained results could might be improved even further.

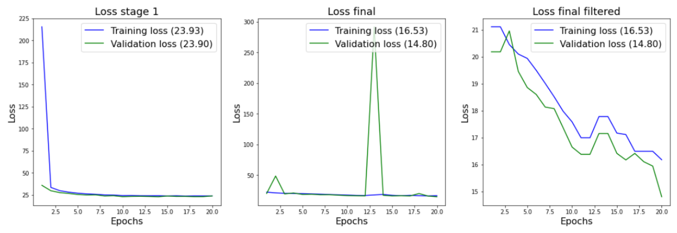

Fig. 4: Losses with data augmentation and median filter

D. *Training With Data Augmentation and HE*

When applying histogram equalization (HE) the IOU is 0.50, recall 0.64 and the nZIR is 0.70. The loss can be seen in image 6. It decreased the quality from when no preprossessing method was applied, meaning it negatively impacted the training process.

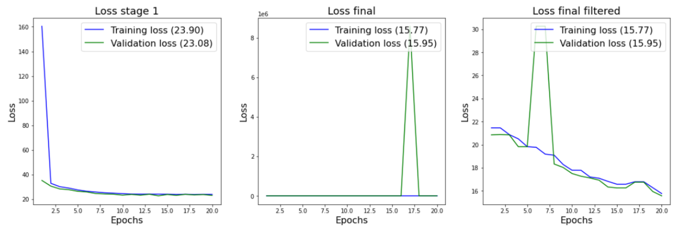

Fig. 5: Losses with data augmentation and histogram equalization.

E. *Training With Data Augmentation, Median Filter and HE*

This model was trained by applying a median filter and histogram equalization. The results are IOU 0.52, recall 0.67 and nZIR 0.72. The final loss is 16.28 as can be seen in Figure 6. The model performs better then when only histogram equalization is used, but worse then when only a median filter is applied. This further reinforces the theory that histogram equalization worsens the quality while a median filter improves it.

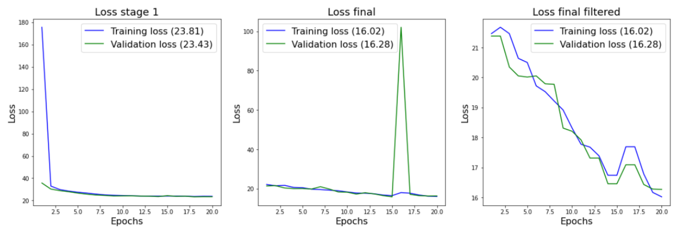

Fig. 6: Losses with data augmentation, median filter and histogram equalization.

IV. Discussion

This section discusses common problems in detecting human faces and how the final model with the best quality is able to handle them.

As can be seen in Figure 7, the model is able to detect heads of people with various skin colors. There seems to be no influence of the skin color on the confidence score. The different scores in the example images are more probably caused by occlusion.

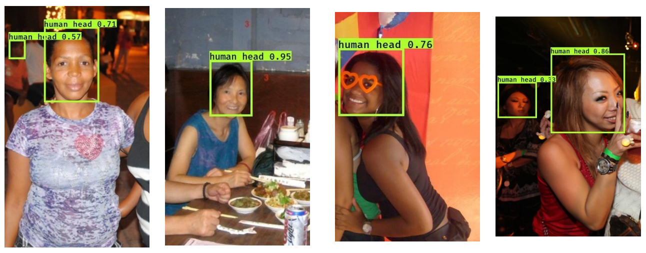

Fig. 7: People with various skin colors.

Figure 8 depicts how the models deals with larger groups of people and with heads that are photographed from uncommon angles. In these images one can see that the model mostly has no problem recognizing heads photographed from the side, the back or further away. The confidence in the predictions is rather low though. It is able to detect most heads in a bigger group. However, it still misses some.

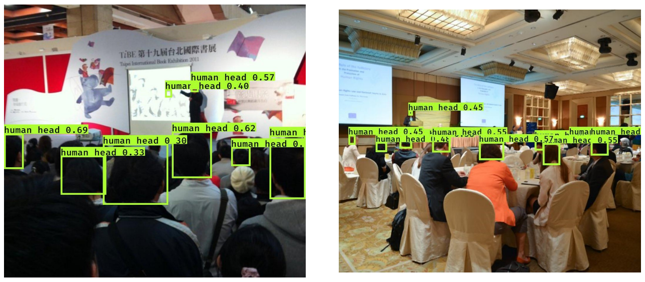

Fig. 8: Heads in uncommon settings.

In Figure 9 the problem with occlusion is illustrated. Heads that are fully uncovered are located, while faces that are partially covered through glasses, masks or other objects are not recognized at all. This phenomenon becomes especially obvious in the second image, as none of the heads are detected.

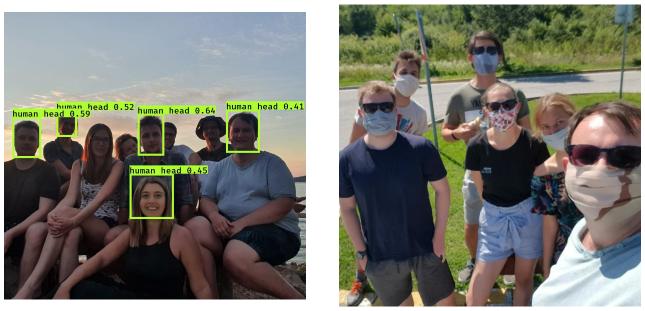

Fig. 9: Problem with occlusion.

Figure 10 shows the performance of the model on animals. Animals with a face similar to humans, e.g. monkeys, are still localized with a meaningful confidence score. Other animals like cats and dogs, which resemble humans less, are not detected.

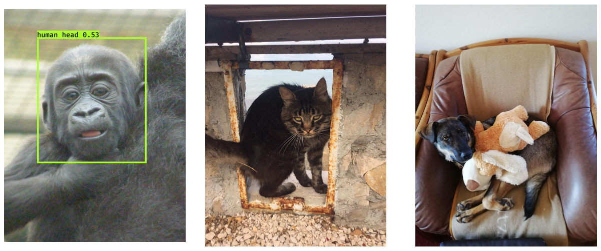

Fig. 10: Problem with human-like animals.

Horizontal flipping was not used for data augmentation. Therefore the performance suffers when flipping the image upside down, as can be seen in Figure 11. Without flipping the head of the woman is detected with a confidence of 0.69. After flipping the confidence of the actual head sinks to 0.35. The model expects the head in the upper half of the image, therefore the skirt of the woman appears more likely to be a head in this case.

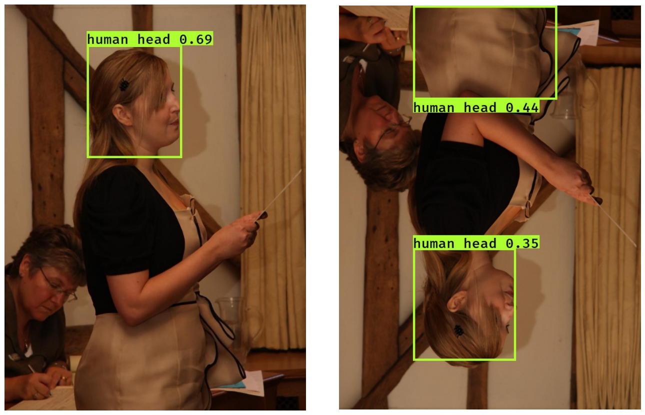

Fig. 11: Horizontal image flipping.

V. Conclusion

Table I shows the results of all five runs. The worst result was achieved was when no data augmentation was applied, supporting the theory of data augmentation being beneficial to the final quality of the model. Histogram equalization lead to worse results in all runs, while the median filter had a positive influence. The best result was achieved when both data augmentation and median filter were applied, but no histogram equalization.

TABLE I: Results

Epo.	Aug.	Med.	Hist.	IOU	Rec.	nZIR
70*2	Yes	No	No	0.57	0.73	0.76
20*2	No	No	No	0.44	0.57	0.61
20*2	Yes	Yes	No	0.57	0.74	0.79
20*2	Yes	No	Yes	0.50	0.64	0.70
20*2	Yes	Yes	Yes	0.52	0.67	0.72

A. *Model Flaws*

- If heads are covered partially the model has problems recognizing them.
- On animals that resemble humans, their head is detected as well.
- On flipped images, the actual heads are detected with less confidence, while sometimes other objects are detected as heads as well.

B. *Model Strengths*

- The model can detect several people in group images.
- The heads can be recognized from rather uncommon angles such as from the side or the back.
- There is no problem recognizing people with various skin colors.

To further optimize the model it would be sensible to retrain on an image set with more people who are wearing accessories, animals that resemble humans and horizontally flipped images.

References

[1] Van Oosterhout, T., Bakkes, S. and Kröse, B. J. (2011). ”Head Detection in Stereo Data for People Counting and Segmentation”. *VISAPP*, 620-625.

[2] Gao, C., Li, P., Zhang, Y., Liu, J. and Wang, L. (2016). ”People counting based on head detection combining Adaboost and CNN in crowded surveillance environment”. *Neurocomputing*, 208, 108-116.

[3] Krasin I., Duerig T., Alldrin N., Ferrari V., Abu-El-Haija S. et al. (2017) ”OpenImages: A public dataset for large-scale multi-label and multi-class image classification”. Retrieved Jan. 31, 2021, from https://storage.googleapis.com/openimages/web/index.html.

[4] Krasin I., Duerig T., Alldrin N., Ferrari V., Abu-El-Haija S. et al. (2017) ”OpenImages Dataset V6 - Human Head”. Retrieved Jan. 31, 2021, from https://storage.googleapis.com/openimages/web/visualizer/index.html?set =train&type=detection&c=%2Fm%2F04hgtk.

[5] Redmon, J., Divvala, S., Girshick, R. and Farhadi, A. (2016). ”You only look once: Unified, real-time object detection.” *Proceedings of the IEEE conference on computer vision and pattern recognition*, 779-788.

[6] Muehlemann, A. (2019) ”TrainYourOwnYOLO: Building a Custom Object Detector from Scratch”. Retrieved Jan. 31, 2021, from https://github.com/AntonMu/TrainYourOwnYOLO

[7] qqwweee. ”A Keras implementation of YOLOv3 (Tensorflow backend)”. Retrieved Jan. 31, 2021, from https://github.com/qqwweee/keras-yolo3

[8] Redmon, J. ”YOLO: Real-Time Object Detection”. Retrieved Jan. 31, 2021, from https://pjreddie.com/darknet/yolo/

[9] TensorFlow. ”Transfer learning and fine-tuning”. Retrieved Jan. 31, 2021, from https://www.tensorflow.org/guide/keras/ transfer_learning?hl=en

[10] Roboflow, Inc. ”Pascal VOC XML”. Retrieved Jan. 31, 2021, from: https://roboflow.com/formats/ pascal-voc-xml

[11] Adams, J. ”openimages”. Retrieved Jan. 31, 2021, from: https://github.com/mono congo/openimages

[12] Roboflow, Inc. ”YOLO Darknet TXT” Retrieved Jan. 31, 2021, from https://roboflow.com/formats/yolo-darknet-txt

[13] L. Taylor, G. Nitschke. (2017) ”Improving Deep Learning using Generic Data Augmentation”. Retrieved from http://arxiv.org/abs/1708.06020.

[14] Hui, J. ”Real-time Object Detection with YOLO, YOLOv2 and now YOLOv3” (2018). Retrieved Jan. 31, 2021 from https://jonathan-hui.medium.com/real-time-object-detection-with-yolo-yolov2-28b1b93e2088

[15] Rosenbrock A. (2017) ”Intersection over Union (IoU) for object detection”. Retrieved Jan. 31, 2021 from: https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/

[16] Tarang S. (2018) ”Measuring Object Detection models — mAP — What is Mean Average Precision?”. Retrieved Jan. 31, 2021, from https://towardsdatascience.com/what-is-map-understanding-the-statistic-of-choice-for-comparing-object-detection-models-1ea4f67a9dbd