Assignment #2: Basic Detection

Catarina Fernandes
IBB 22/23, FRI, UL
cj92629@student.uni-lj.si

I. Introduction

Object detection is a field that has been growing increasingly, especially when applied to biometry. It combines two tasks, image classification, which involves predicting the class of one object in an image, and object localization, which involves identifying the location of one or more objects in an image.

The task at hand was to evaluate two popular object detection methods: Haar Cascades [1] and YOLO (v5) [2] [3]. For this, we used the provided trained model and cascades and the provided data set made up of 500 images including ears and their respective annotations.

II. Related Work

There have been a number of object detectors developed throughout time. Currently, some of the most relevant are based on YOLO, "You only look once" and FCOS, "Fully Convolutional One-Stage Object Detection". YOLO, one of the algorithms this study focuses on, is an object detection algorithm that divides images into a grid system. Each cell in the grid is responsible for detecting objects within itself. [4] The other algorithm to be focused on, Viola-Jones or Haar Cascades, is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images. [5]

III. Methodology

The goal was to analyze both the Viola-Jones and YOLO (v5), so, to start, the first thing to be done was to establish a baseline. The Viola-Jones algorithm without fine-tuning was chosen. All the chosen algorithms performed the same detection task over the same data set.

For the next step, it was important to compare the baseline with improved algorithms. The first comparison aimed for was Viola-Jones with optimized parameters for the ear detection task. In order to establish what those parameters were, an experiment was done to compare various combinations of the parameters *Scale Factor* and *Minimum Number of Neighbours*. The results were then classified by being compared to the ground truth annotations and using *mean accuracy* as the metric.

For the second comparison, YOLO (v5) was used, and the weights chosen were the ones provided.

IV. Experiments

As a basis for the experiments, all the images were loaded by their original size and gray-scale values.

As mentioned in the previous section, the first experiment consisted in fine-tuning the Viola-Jones algorithm. For that, we provided two lists of possible values for the parameters and ran all possible combinations. The provided values were:

```
Scale factor = [1.1, 1.2, 1.3, 1.4,
    1.5, 1.6, 1.7, 1.8, 1.9, 2]
Minimum number of neighbors =
    [0, 1, 2, 3, 4, 5, 6]
```

Then, the Mean Accuracy and Intersection over Union (IoU) were calculated and the Mean Accuracy was used to rate the performance of each combination. With this in mind, the top five best and worst-performing combinations were selected for comparison. A Precision-Recall curve over all thresholds (step of 0.01) for the baseline was plotted (Fig. 1), as well as for the selected combinations (Best performing - Fig. 2, 3, 4, 5, 6) (Worst performing - Fig. 7, 8, 9, 10, 11). The x-axis represents the value of the threshold and the y-axis is the percentage of images whose IoU value is bigger than the threshold.

Regarding YOLO (v5), the five best predictions and the worst ten predictions were extracted and the Precision-Recall curve was computed (Fig.12.

V. Results and Discussion

After running both Viola-Jones and YOLO (v5), there is enough data to analyze and discuss the results.

A. Results

In table I below, we can find the Mean Accuracy and Mean Intersection over Union of the baseline, selected Viola-Jones parameters and YOLO (v5).

It is also possible to see the image representations of 10 failed Viola-Jones predictions, the five best Viola-Jones representations and the five best YOLO predictions, as measured by accuracy, in the appendix.

B. Discussion

The best performing algorithm was YOLO (v5). Despite not having the highest accuracy, it matches its high accuracy with

TABLE I
ALGORITHM COMPARISON - FIVE BEST AND WORST PERFORMING VIOLA-JONES ALGORITHMS, VIOLA-JONES BASELINE AND YOLO (V5)

Algorithm	Accuracy	IoU
(baseline) VJ S=1.1 N=3	0.392663	0.314424
VJ S=1.1 N=0	0.783817	0.493783
VJ S=1.2 N=0	0.726743	0.465292
VJ S=1.3 N=0	0.674724	0.439796
VJ S=1.4 N=0	0.635800	0.417217
VJ S=1.5 N=0	0.625254	0.410071
VJ S=2 N=6	0.019200	0.016354
VJ S=1.8 N=6	0.023048	0.019480
VJ S=1.9 N=6	0.023132	0.019473
VJ S=1.4 N=6	0.026978	0.023258
VJ S=1.7 N=6	0.028924	0.024869
YOLO v5	0.749345	0.728178

an equally high IoU, contrary to the other algorithms. Regarding Viola-Jones, we can observe that the best performing parameters are the lower ones, namely a scale factor of 1.1 and a minimum number of neighbors of 0.

However, despite some fine-tuned Viola-Jones algorithms getting high accuracy, their predictions lack quality. The bounding boxes are often out of place or with a different dimension than the object they should be detecting. This is reflected in the IoU scores, which aren't as high as the accuracy and indicate that much of the detected bounding box doesn't match the ground truth.

VI. CONCLUSION

By the end of the experiment, it was safe to assume that YOLO (v5) had a better performance than Viola-Jones, even when fine-tuned. Another advantage was how fast it was when compared to Viola-Jones, but the trade-off is that it is more computationally expensive.

For future work, it would be interesting to make the same comparison using other classes, instead of just ”ear”, and compare other similar algorithms.

REFERENCES

[1] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, vol. 1. Ieee, 2001, pp. I–I.

[2] “ultralytics/yolov5,” original-date: 2020-05-18T03:45:11Z. [Online]. Available: https://github.com/ultralytics/yolov5

[3] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “Yolov4: Optimal speed and accuracy of object detection,” *arXiv preprint arXiv:2004.10934*, 2020.

[4] YOLOv5 documentation. [Online]. Available: https://docs.ultralytics.com/

[5] OpenCV: Face detection using haar cascades. [Online]. Available: https://docs.opencv.org/3.1.0/d7/d8b/tutorial$_{py_face_detection.htmlgsc.tab} = 0$

APPENDIX

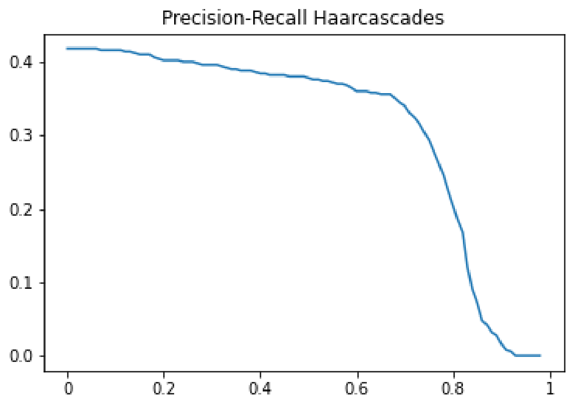

Fig. 1. Precision-Recall curve for the baseline Viola-Jones algorithm.

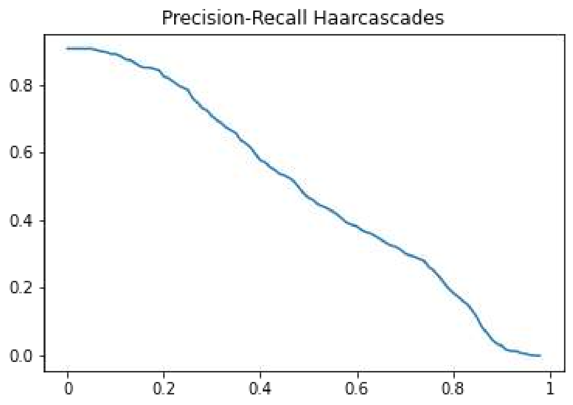

Fig. 2. Precision-Recall curve for the best performing fine-tuned Viola-Jones algorithm.

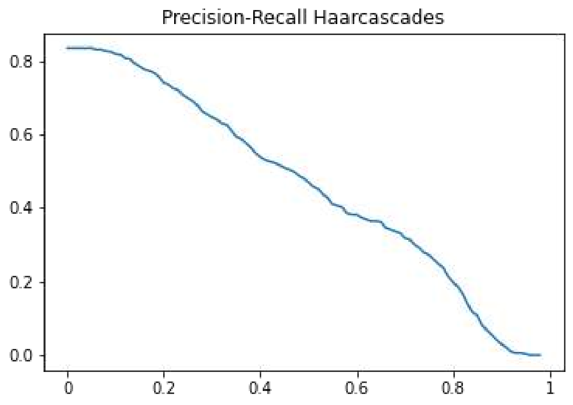

Fig. 3. Precision-Recall curve for the second best performing fine-tuned Viola-Jones algorithm.

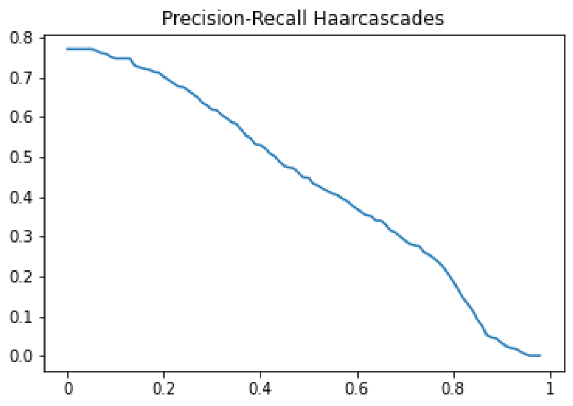

Fig. 4. Precision-Recall curve for the third best performing fine-tuned Viola-Jones algorithm.

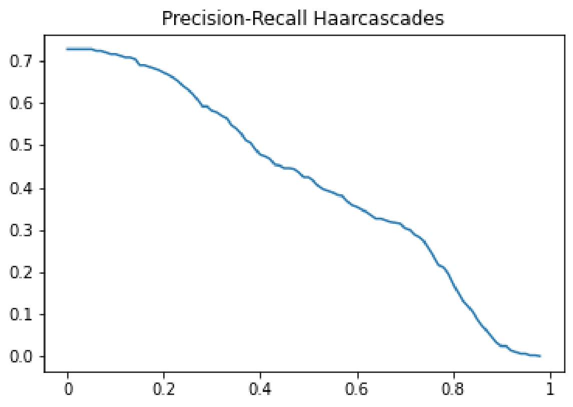

Fig. 5. Precision-Recall curve for the fourth best performing fine-tuned Viola-Jones algorithm.

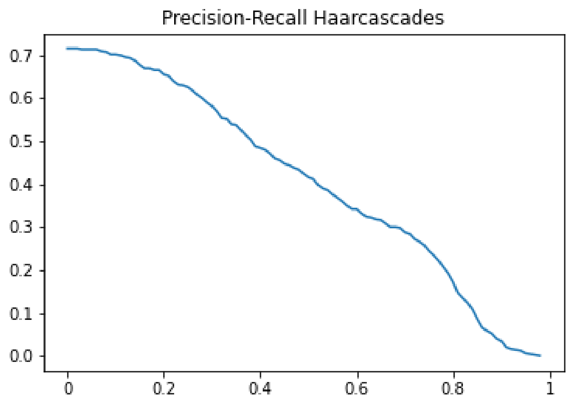

Fig. 6. Precision-Recall curve for the fifth best performing fine-tuned Viola-Jones algorithm.

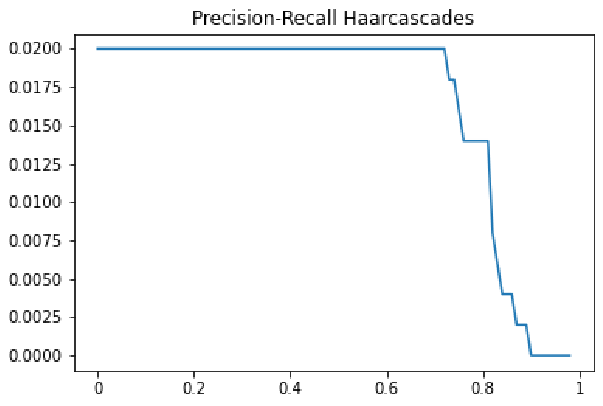

Fig. 7. Precision-Recall curve for the worst performing fine-tuned Viola-Jones algorithm.

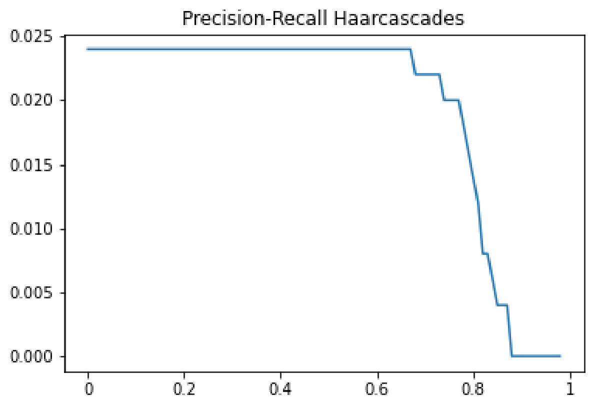

Fig. 8. Precision-Recall curve for the second worst performing fine-tuned Viola-Jones algorithm.

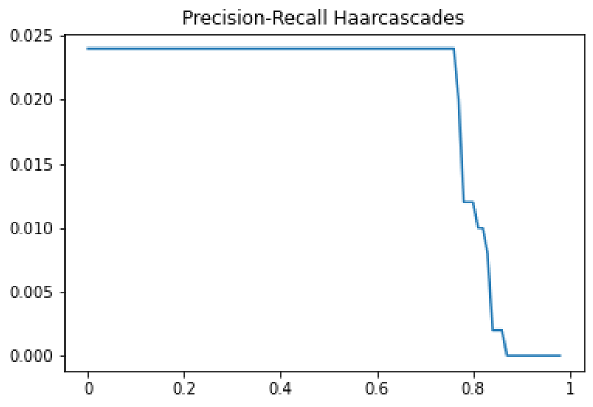

Fig. 9. Precision-Recall curve for the third worst performing fine-tuned Viola-Jones algorithm.

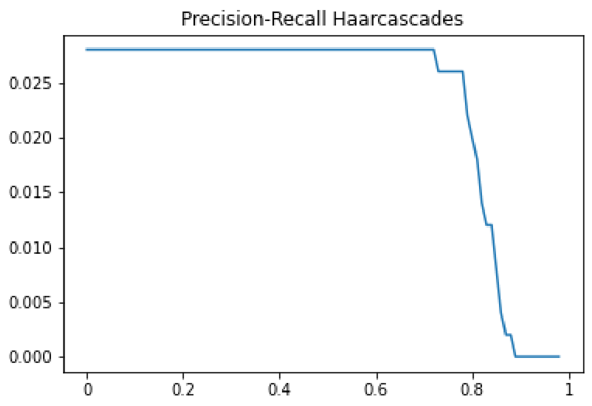

Fig. 10. Precision-Recall curve for the fourth worst performing fine-tuned Viola-Jones algorithm.

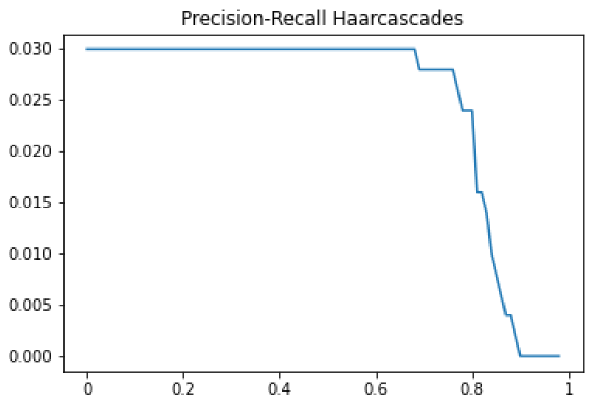

Fig. 11. Precision-Recall curve for the fifth worst performing fine-tuned Viola-Jones algorithm.

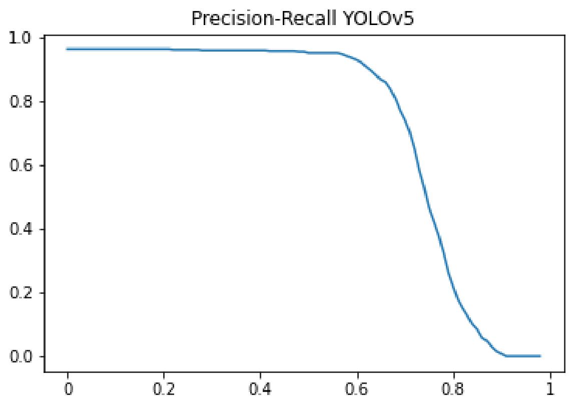

Fig. 12. Precision-Recall curve for the YOLO (v5) algorithm.

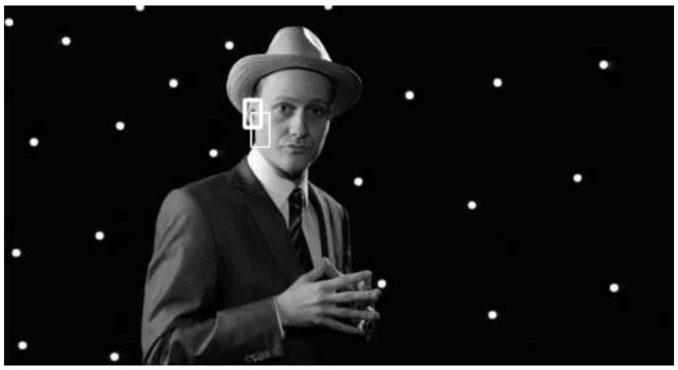

Fig. 13. Image with the best accuracy in the Viola-Jones algorithm fine-tuned (0590.png).

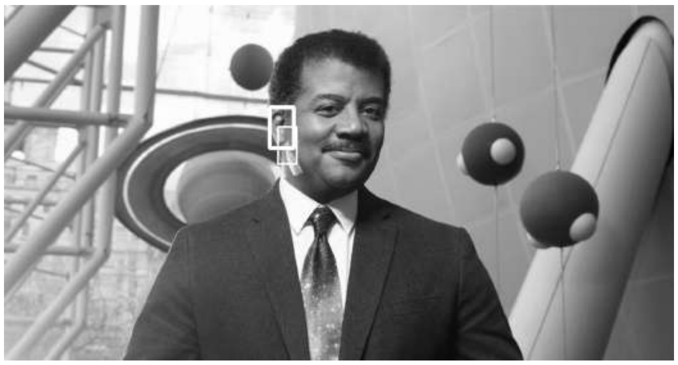

Fig. 14. Image with the second best accuracy in the Viola-Jones algorithm fine-tuned (0664.png).

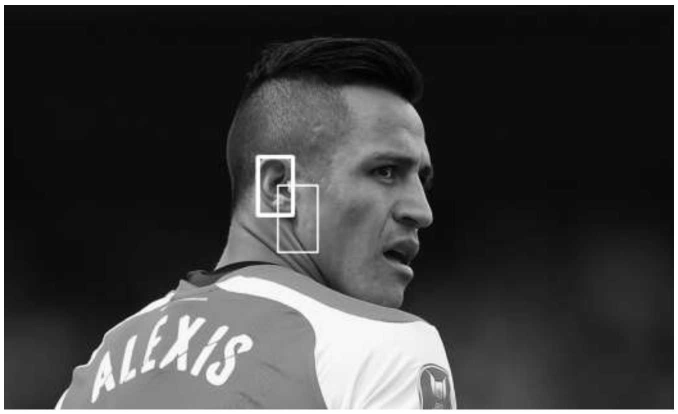

Fig. 15. Image with the third best accuracy in the Viola-Jones algorithm fine-tuned (1921.png).

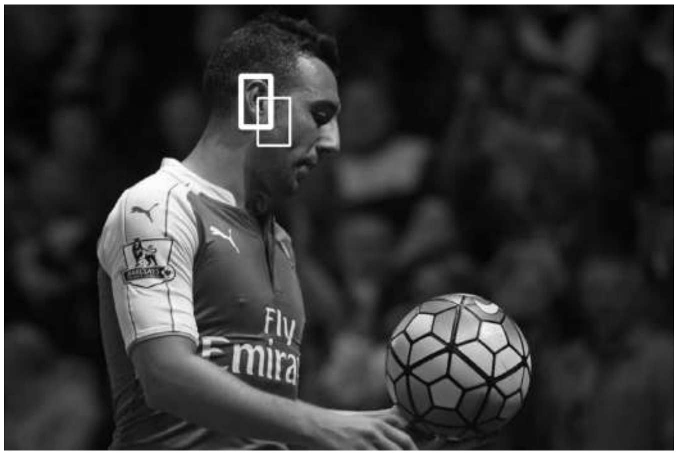

Fig. 16. Image with the fourth best accuracy in the Viola-Jones algorithm fine-tuned (1939.png).

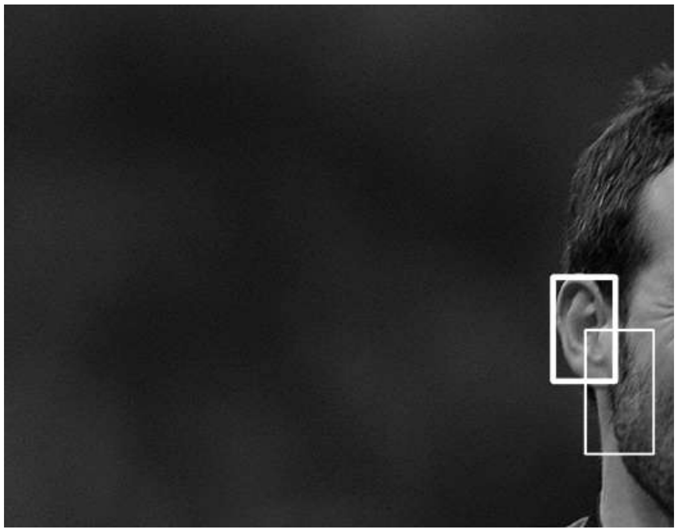

Fig. 17. Image with the fifth best accuracy in the Viola-Jones algorithm fine-tuned (1927.png).

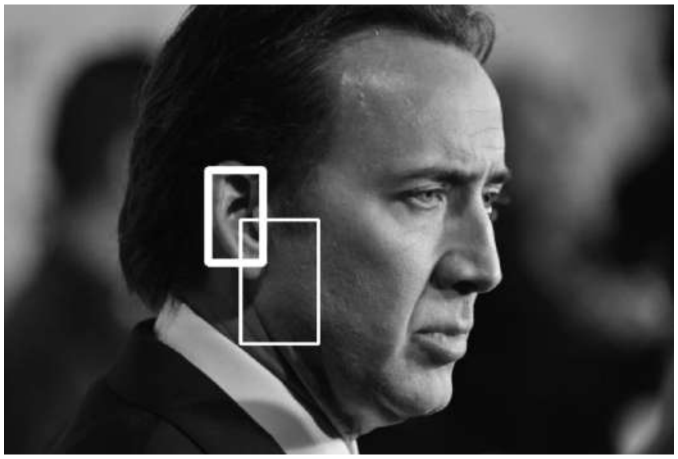

Fig. 18. Image with the worst accuracy in the Viola-Jones algorithm fine-tuned (0509.png).

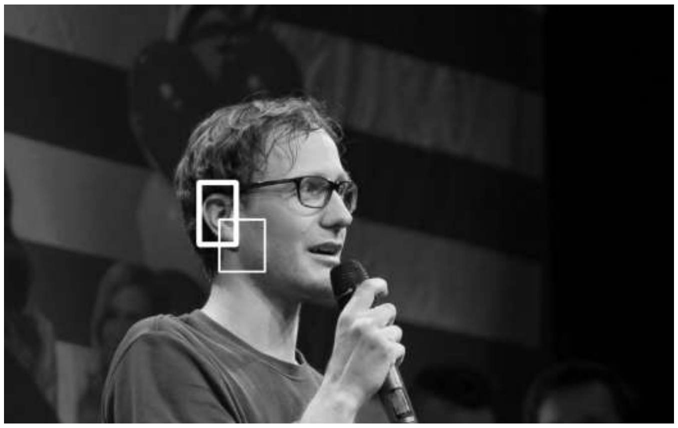

Fig. 19. Image with the second worst accuracy in the Viola-Jones algorithm fine-tuned (0536.png).

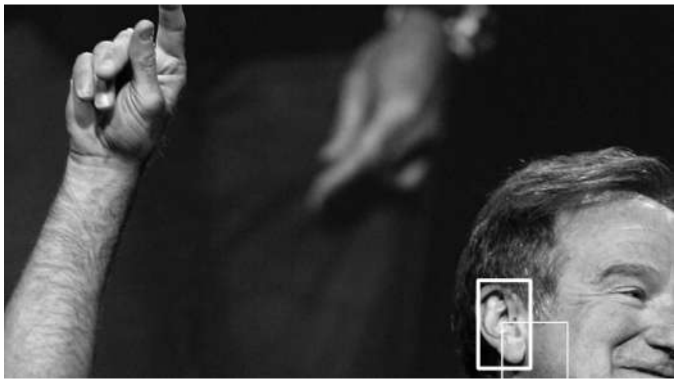

Fig. 20. Image with the third worst accuracy in the Viola-Jones algorithm fine-tuned (0579.png).

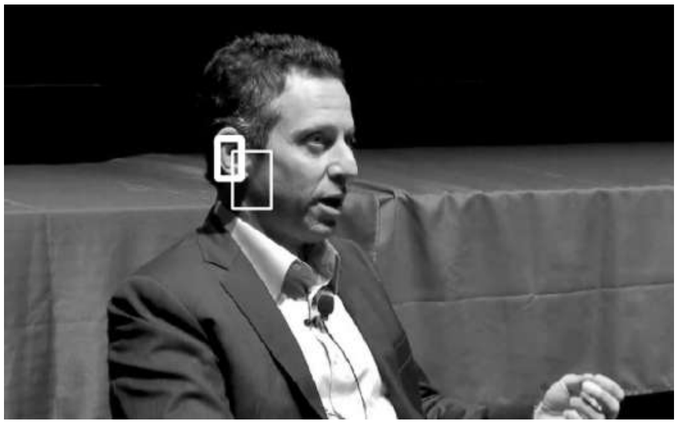

Fig. 21. Image with the fourth worst accuracy in the Viola-Jones algorithm fine-tuned (0602.png).

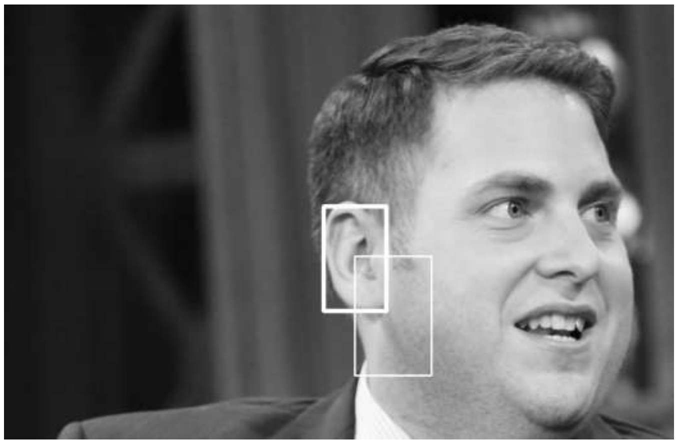

Fig. 22. Image with the fifth worst accuracy in the Viola-Jones algorithm fine-tuned (0639.png).

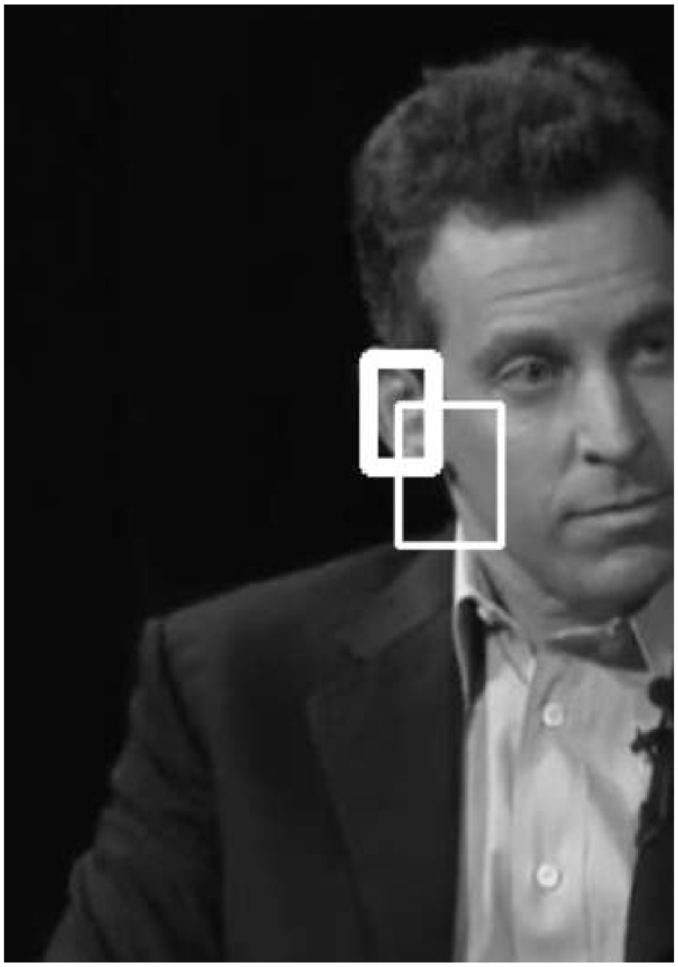

Fig. 23. Image with the sixth worst accuracy in the Viola-Jones algorithm fine-tuned (0653.png).

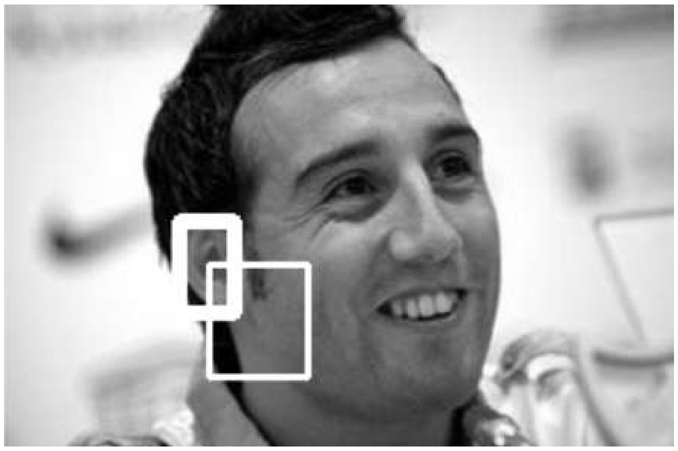

Fig. 24. Image with the seventh worst accuracy in the Viola-Jones algorithm fine-tuned (1977.png).

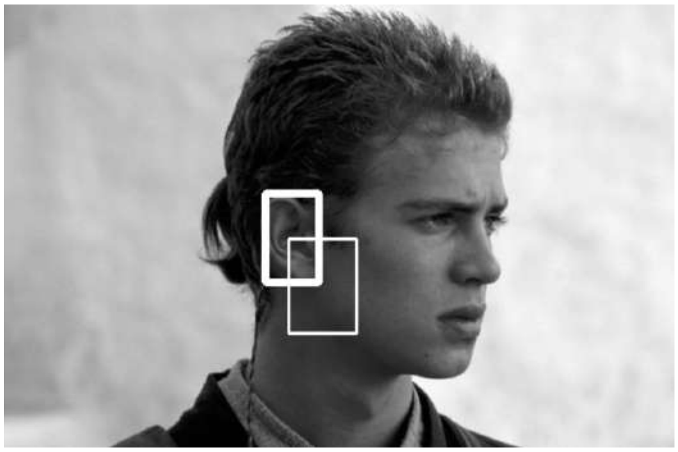

Fig. 25. Image with the eighth worst accuracy in the Viola-Jones algorithm fine-tuned (2035.png).

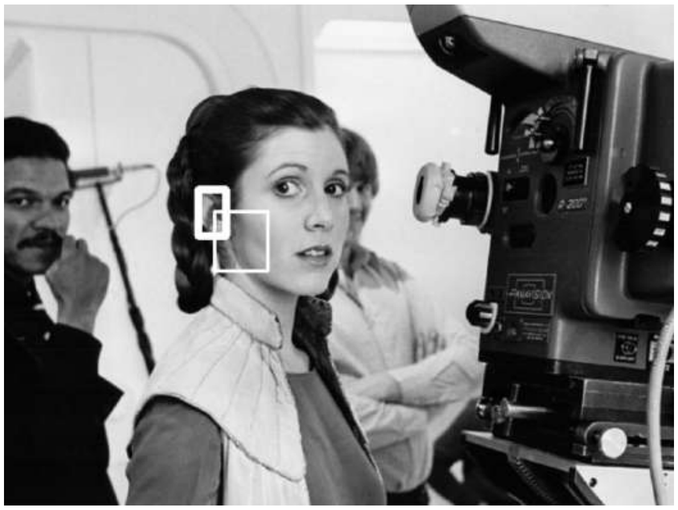

Fig. 26. Image with the ninth worst accuracy in the Viola-Jones algorithm fine-tuned (2058.png).

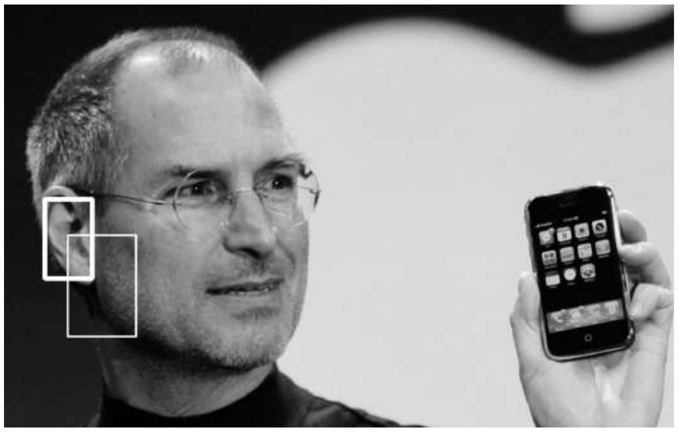

Fig. 27. Image with the tenth worst accuracy in the Viola-Jones algorithm fine-tuned (0519.png).

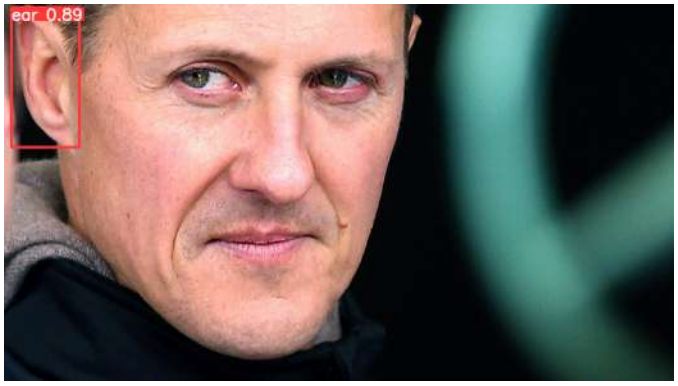

Fig. 28. Image with the best accuracy in the YOLO (v5) algorithm (0533.png).

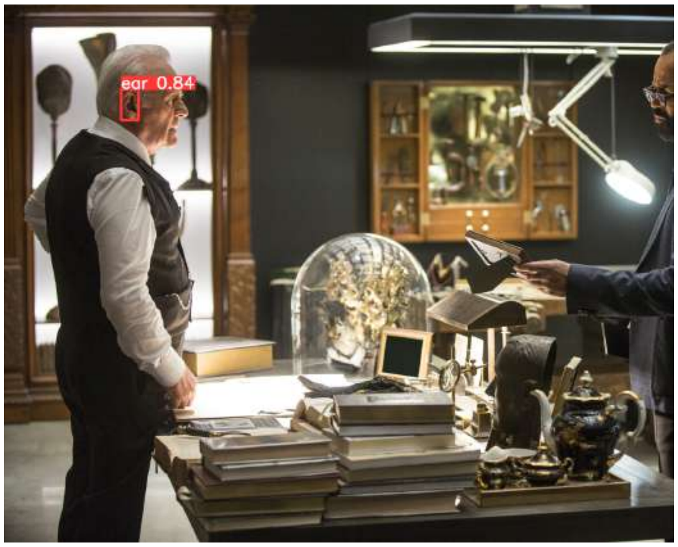

Fig. 29. Image with the second best accuracy in the YOLO (v5) algorithm (2131.png).

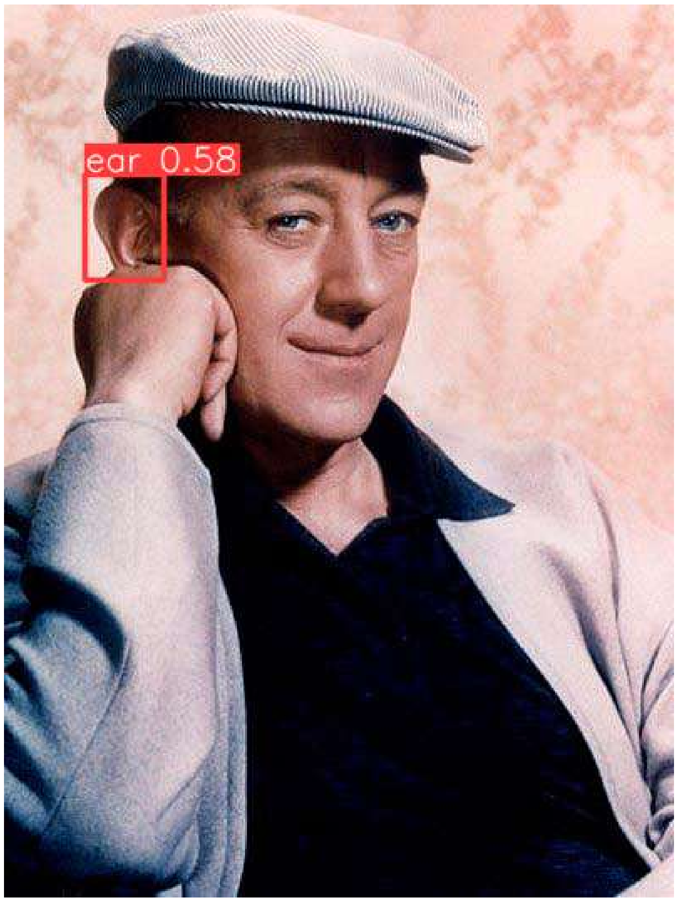

Fig. 30. Image with the best accuracy in the YOLO (v5) algorithm (2063.png).

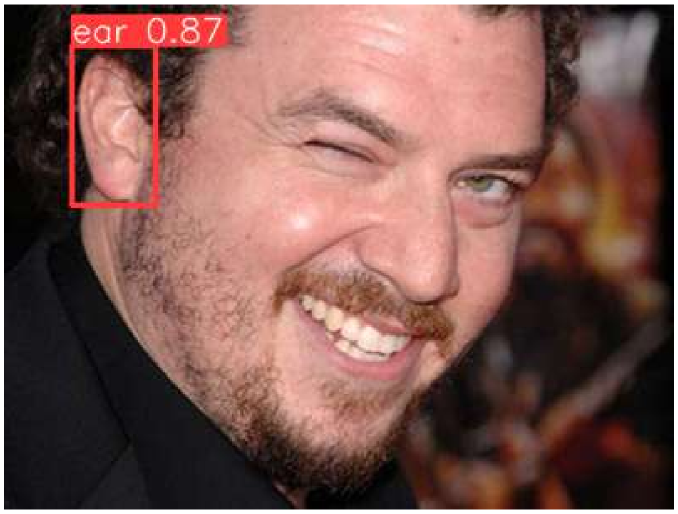

Fig. 31. Image with the best accuracy in the YOLO (v5) algorithm (0649.png).

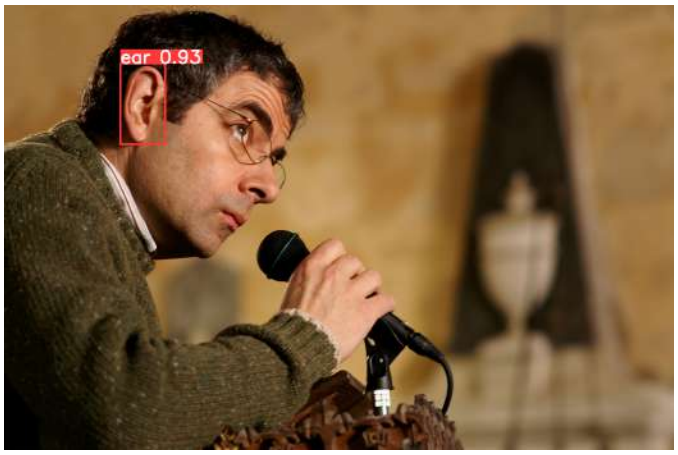

Fig. 32. Image with the best accuracy in the YOLO (v5) algorithm (0504.png).