# Pupil Detection and Tracking

Praneeth Nekkalapudi
Computer Vision and Image Processing
University at Buffalo

## Overview of the project

### Application Summary

- The focus of the project is to develop a user focus estimation program. It is achieved by detecting facial features to locate the pupils in the eyes of the user and tracking the pupils across frames to make an assessment on the mental state of the user. In this project, **focused, distracted, away, tired** and **moving** are the different states that are being observed. While away and moving don't particularly signify anything of importance about the mental state of the user, they are included to provide insights into user activity.

### Rationale

- Focus and eye-movement are used in psychology to gain an understanding of the subject's state of mind. Analytics on eye-tracking data can help detect early signs of neurodivergent conditions like ADHD or ADD in children and learning disabilities like dyslexia.
- Certain disabilities hamper an individual's ability to communicate verbally or with gestures and they rely on communicating with eye movements. Eye tracking software would alleviate the difficulties in communication for these subjects and improve their quality of life.
- Furthermore, with Extended Reality applications growing every day, tracking eye movements can be essential in extending the capabilities of VR headsets.
- It can also be used to detect tired drivers and warn them to avert disaster.

### Inputs/Outputs

- The program will take the live video feed from a PC camera as the input and record user activity, and upon termination, the program outputs a pie-chart describing the user activity during the period the camera is actively recording. Additionally, the program can also be made to display the pupils detected across frames to validate the functioning of the application.

### State-of-the-art

- Today's state of the art systems use convolutional neural networks to accurately detect the eye locations, however the pipeline includes several image processing techniques to achieve greater speeds, improved localization and better accuracy. Many of these image processing approaches require considerable calibration.
- A popular approach involves specular corneal reflection to localize the eyes[1]. Another interesting approach treats the movement of the pupils as a Hidden Markov Model and the location of the pupil is predicted using particle filters[1]. In terms of neural models for detection, the new YOLO NAS architecture has recently achieved state-of-the-art results on object detection benchmarks [2]. Alternatively, the objective of

pupil tracking can also be represented as a segmentation task and the Segment Anything Model that has recently been released by Meta research has proven to be a robust generalized segmentation model[3].

- Other image processing methods involve fast object detection approaches using HAAR and HOG classifiers to localize the region around the eye and then performing image morphology, texture analysis through template matching to locate the pupils.
- Transformer based architectures have found their way into the fold for computer vision tasks and have been known to perform well in tasks that involve tracking long term dependencies through attention mechanisms[8, 10].

### Contribution

- For this project, I've followed two major approaches to implement the gaze tracking application. One approach relies on object(eyes) detection using two classifiers, pretrained HOG from the dlib library and a fine-tuned CNN based Yolov8 model followed by image processing to locate the pupils, and the other approach involves annotating image data following the first approach to build a neural object detector that locates the pupil directly.

## Approach

### Algorithm 1

- Histogram of Oriented Gradients is a feature descriptor commonly used in computer vision for object detection. Much like SIFT and SURF, it computes the local features and patterns in images which can be used for matching interest points robustly. The local features make up the distinguishing inputs to the object detection task. A linear SVM classifier is trained on the features obtained from the HOG descriptor. This part is implemented using the face detection module in dlib library. Facial landmarks are extracted and the landmarks corresponding to eyes are used to crop the image to extract the eyes.

Fig 1: cropped eyes

- Additionally, to supplement the HOG + linear SVM based eye region detection, a YOLOv8 model is trained on an image dataset consisting of faces and annotated regions for eyes. The two models work as an ensemble to improve the accuracy for detecting eyes.

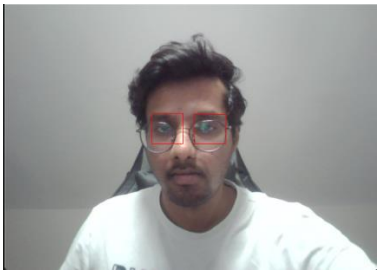

Fig 2: YOLOv8 model eye detection.

- Next, the image is binarized by finding an adaptive threshold. The adaptive threshold ensure that the algorithm generalizes to detecting all kinds of eyes. Then morphological operations are performed to enhance the distinctness of the pupils. The Image is dilated before thresholding to make the pupils larger and eroded after binarization the return to the original size and remove any specular artefacts.

Fig 3: Binarized pupil

- Pupil displacements are computed between the frames to determine the user's focus levels.

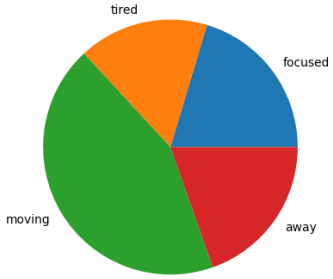


Fig 4: User focus analysis

- While this approach is fast and can be used for real-time inference, the image processing component suffers from extreme sensitivity to environmental factors like illumination, glare and reflections.

## Algorithm 2

- This approach uses algorithm1 to locate pupils in the data that was used to train the eye region detector and annotate the images with bounding boxes of the pupils. Use the annotated data to train a YOLOv8 model to detect pupils directly.
- The YOLOv8 (You Only Look Once) model has a convolutional backbone and a linear head that minimizes the triple objective of box loss, class loss and detector loss. Box loss corresponds to the fitness of the box coordinates and dimensions, class loss corresponds to the general classification loss and is dependent on the type of detection task, multi-label or multiclass and finally the detector loss corresponds to the loss associated with the confidence scores of the predictions. Furthermore, the YOLOv8 model deploys adaptive box dimensions to accurately detect objects of all shapes. Another key improvement in the model is online mosaic image augmentation where the images are augmented and stitch together to increase the search space for the objects and eliminate any spatial correlations in the data.
- While this approach comes with the promise to generalize well while being robust to illumination and environmental influences, it suffers greatly from the lack of required image resolution from the video stream input and scale. Pupils are extremely small in the span of the whole image; as small as 3*3 pixels and this makes training the model a challenging task. The results obtained from this approach are unsatisfactory, but they highlight a key challenge in object detection tasks, that of scale and resolution.

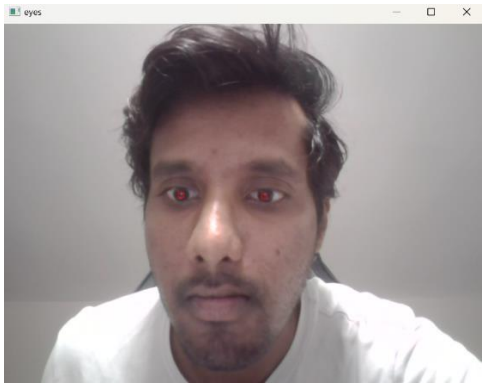

Fig 5: YOLOv8 pupil detection.

### Implementation details

- Implemented by self
  - Data preparation, augmentation, annotation, and training is implemented wholly and exclusively by myself.
  - Data preparation involves sampling 70000 images from 5 million images, processing the corresponding raw label data and organizing it in a suitable format required for training. This is performed for the task of detecting eyes from algorithm1 and the task of detecting pupils from algorithm2.
  - The logic required to annotate the data, since the pupil annotations are not available in the dataset, to make it conducive to train a YOLOv8 model in algorithm2 for pupil detection.
  - Adaptive thresholding required for effective binarization.
  - The complete code to perform analysis and describe user behavior from detected pupils.
- References online
  - Dlib usage is referenced from online sources[4].
  - The image processing component is adapted from several online sources[5,6].
- Coding effort
  - Approximately 800 lines of code.
  - Approximately 100 man-hours spent on researching and implementing the project

## Experimental Protocol

Datasets

- The dataset used for this project is a publicly available dataset that is used for the same task, capturing gaze, described in their paper **Eye Tracking for Everyone**[7] and made available at https://gazecapture.csail.mit.edu/index.php.
- The dataset contains images of people recorded from their iphone front facing cameras for face identification. The dataset contains upwards of 5 million images with annotated text files that define whether an image contains a face and eyes along with the bounding boxes.
- 70000 images are sampled from this dataset and are used to train the two YOLOv8 object detectors.
- 1000 images are used for validation.

Success criteria

- Multiple criteria for success depending on the subtask.
- Evaluation consists of a qualitative and a quantitative component.

- Quantitative measures are used to evaluate the performance of the trained models and the gaze tracking application's success in detecting eyes/pupils.
- Qualitative measures are interpretation based and study the performance of the trained models and the gaze tracking application as a whole under various influences (illumination, glare, reflection, scale).
- Criteria for success would be to use the application for inference under stated assumptions.

Compute resources

- Training is performed on the UB Center for Computational Resources servers whereas the rest of the inference and implementation is performed on a PC.

## Results

Eye Detection

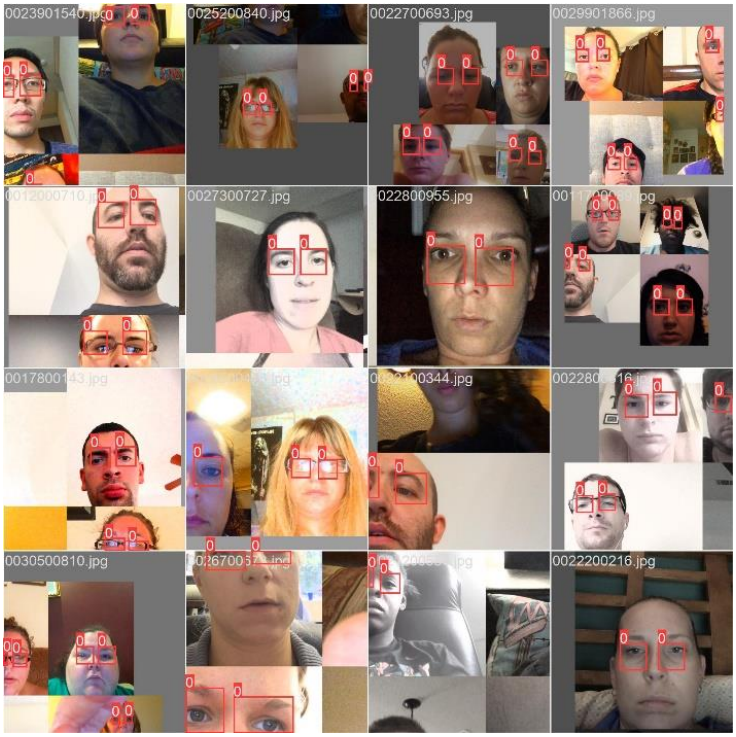


Fig 1 – Mosaiced images from train batch

The above figure illustrates the image augmentation performed during training. Images are scaled within a specified range (0.5x – 1.5x), rotated 45 degrees and a training batch is created by creating a mosaic of the images. The bounding boxes correspond to the eyes.


Fig 2: Images used in validation. Images on the left represent the ground truth and images on the right represent the predictions.

As can be seen from the pair of images above, the YOLOv8 model has learned to detect eyes accurately.

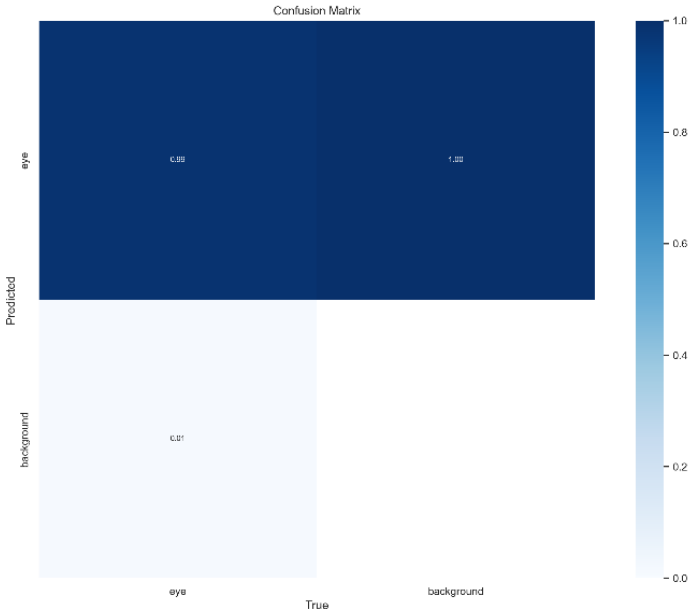


Fig 4: Confusion Matrix for eye detection

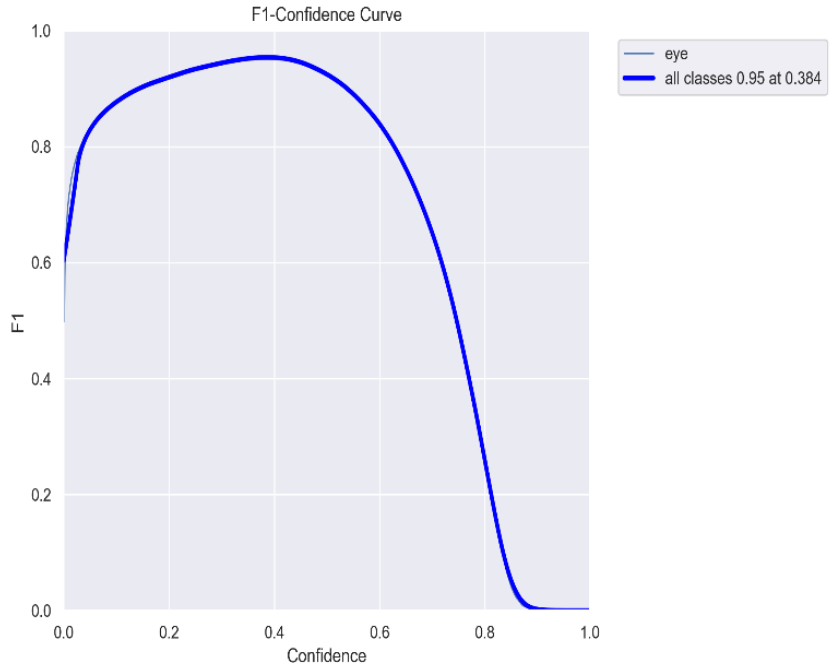


Fig 5: F1-confidence curve

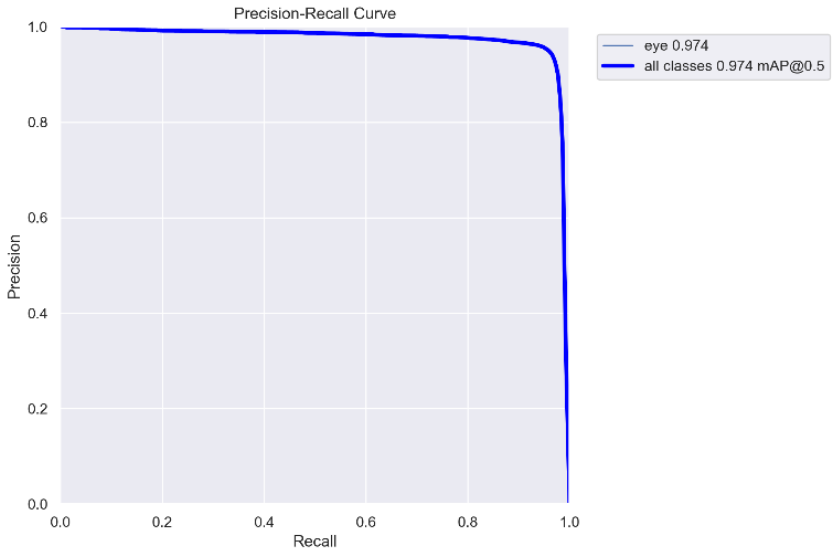


Fig 6: Precision-Recall curve

The above set of figures illustrate the performance of the trained model. Misclassification rate is low and the precision-recall curve tells us that the model detects eyes pretty reliably without missing them. The F1-confidence curve provides insight into the balance between precision and recall. The curve approaches a zero F1 score at the far extremity because recall approaches zero for high confidence levels.

## Pupil Detection

Fig 7: Mosaic train batch with annotated pupil boundind boxes.

To annotate the data with pupil bounding boxes, faces are detected using dlib HOG+linear SVM face detection module, eyes are localized using the trained eye detector. Then, image processing detailed in Algorithm 1 of the previous section localizes the pupils. The dataset is annotated for pupils in this manner.

Fig 8: Images used for validating pupil detection. Images on the left represent the ground truth and images on the right represent the predictions.

As can be observed, the pupils are not detected reliably. While the trained model has good precision, the recall is low rendering it ineffective for inference and the wider application.

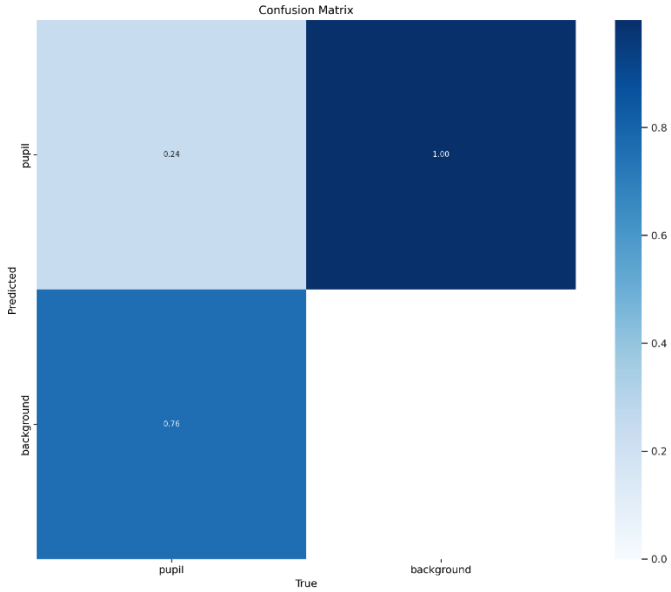


Fig 9: Confusion matrix for pupil detection

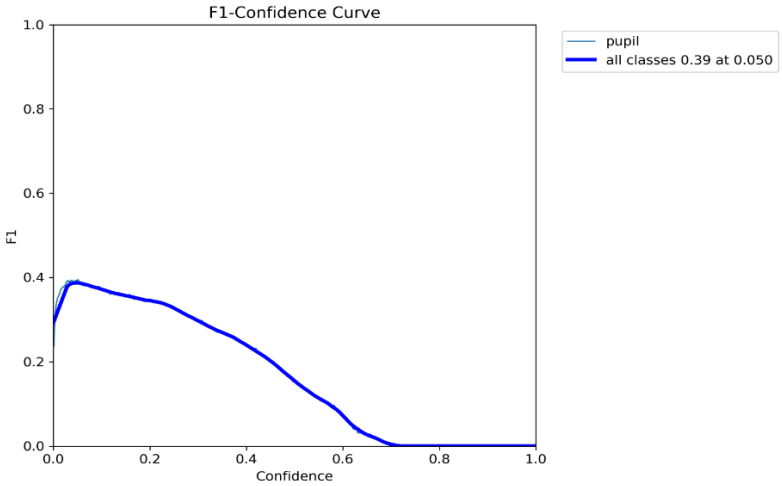


Fig 10: F1-Confidence curve

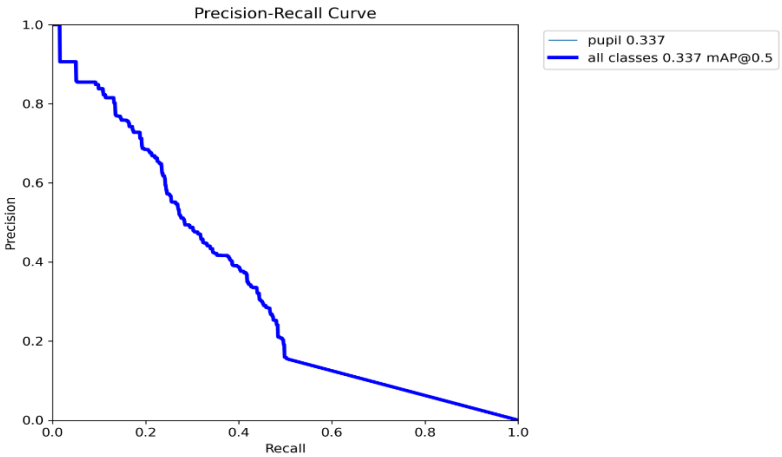


Fig 11: Precision-Recall curve

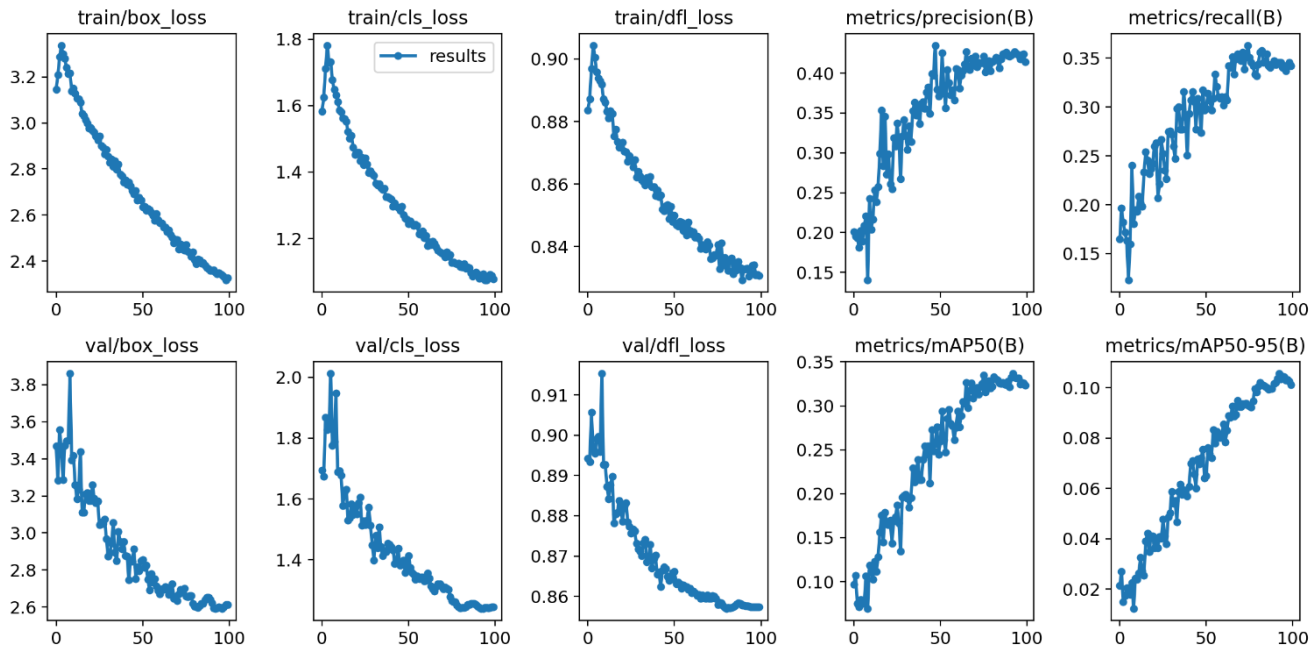


Fig 11: Training Progression

The poor performance of the model is reflected in the performance plots. The model has extremely low precision and recall, due to which the F1 score suffers. Figure 11 illustrates how little the metrics have improved over 100 epochs of training.

## Analysis

Algorithm 1

- Advantages
  - Suitable for fast, real-time applications
  - Offers greater flexibility and scope for fine-tuning. Room for improvement in the algorithm in both the phases; detecting pupils and the analysis of user emotional/mental state.
  - Easily explainable, more conducive to ablation study.
- Drawbacks
  - Although adaptive, doesn't generalize well.
  - Not robust to illumination and occlusions.
  - Considerable manual tweaking of thresholds and kernel sizes.
  - While it is relatively better at detecting pupils at different distances, the performance degenerates if the distance grows large.
  - Unreliable in the presence of multiple faces/eyes.

Algorithm 2

- Advantages
  - Better fine-tuning and comprehensive image augmentation can possibly improve the performance leading to a generalizable solution. In its present state, there are no benefits to taking this approach.
- Drawbacks
  - Training overhead.
  - Models are large unlike the light-weight solutions from algorithm 1.
  - Limited scope for explainability.
  - Poor performance.

## Lessons Learned

- Application of various image processing techniques like performing morphology, texture analysis, feature descriptors, segmentation, and object detection.
- Survey and exploring state of the art approaches in object detection, eye detection and tracking.
- Understanding the challenges in implementing computer vision solutions, especially the challenges involved in training and fine-tuning models for specific tasks.
- Exploring various libraries and technologies. Got to learn using YOLO models, processing the data for training and analyzing the results.
- Critical study of research papers.
- The importance and influence of scale of objects and background in object detection tasks will be a key factor in the future applications that I will be working on.
- Proactively reason and predict plausible challenges ahead of experimenting and evaluating the hypothesis before reaching conclusions.

## Bibliography


1. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4839304/
2. https://deci.ai/blog/yolo-nas-object-detection-foundation-model/
3. https://hackernoon.com/metas-new-segment-anything-model-sam-is-a-game-changer
4. http://dlib.net/python/index.html#dlib_pybind11.rectangle
5. https://github.com/antoinelame/GazeTracking/tree/master/gaze_tracking
6. https://towardsdatascience.com/real-time-eye-tracking-using-opencv-and-dlib-b504ca724ac6
7. Eye Tracking for Everyone
   K.Krafka*, A. Khosla*, P. Kellnhofer, H. Kannan, S. Bhandarkar, W. Matusik and A. Torralba
   IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016
8. Visual Transforer for Object Detection, Michael Yang, June 2022 , https://arxiv.org/abs/2206.06323
9. https://blog.roboflow.com/whats-new-in-yolov8/
10. https://huggingface.co/docs/transformers/model_doc/vit
11.