*Face Detection, Classification, and Recognition from UAV Footage*

Anthony Vasquez   
Johns Hopkins University  
Whiting School of EngineeringBaltimor, MD  
[avasque1@jh.edu](mailto:avasque1@jh.edu)

*Abstract*—

Unmanned aerial vehicles (UAVs) are usually controlled by a ground pilot but can also be flown autonomously. UAVs are usually equipped with a streaming device such as a camera, from which humans and events can be captured from a bird’s eye-view. One application of UAV footage includes detection of humans of interest (HOI). This might include missing or lost persons, criminals, or UAV owners. In this paper, current literature and methods are explored, and a face detection and recognition pipeline is created and compared to the work done by DroneFace.

Keywords—UAV, detection, recognition, HOI

# Introduction

UAVs are increasingly less expensive to purchase and have many off-the-shelf models with optional high-grade camera upgrades. Furthermore, flight hardware, batteries, and long-range controller signals, allow for the use of UAVs at long range distances. UAVs are agile, directable, and able to get into hard-to-reach spaces. In short, UAVs are effective tools for face recognition tasks on human populations.

After doing preliminary research, it was discovered that there is no shortage of interest in this topic. The face can be thought of as similar to a visual fingerprint that humans and computer vision algorithms alike use to recognize one another. The use-cases for face detection and recognition are countless, but one example is finding a missing elderly person (Hsu, 2015). Attempts of face recognition using UAVs have been around for more than a decade, and many open-source and licensed implementations have been created. However, there are many challenges that have yet to be solved without expensive hardware or labor-intensive data curation. Furthermore, many failure modes exist, such as large angle and long distance of camera location relative to target. Camera quality, framerate, weather, and other factors potential lead to failure modes. This work attempts to take an open-source dataset, label images, and use open-source architectures to create a cheap off-the-shelf multi-stage face recognition model.

# Related work

## Face Detection

The authors of the FaceDrone experiment investigated Face++ (Andreas Rossler, 2019), ReKognition (Amazon, 2024), Haar Cascades face detection implementation (OpenCV, 2024), and Local Binary Patterns (LBPs) (Rosebrock, 2021). For the task of face detection, their findings showed that Face++ and ReKognition performed poorly in comparison to the open-source Haar methods. These are important findings, because it shows that for high resolution low altitude drone footage open-source methods exist that significantly outperforms pretrained models that require a license to use.

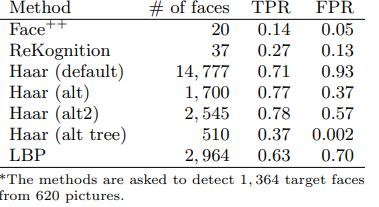


Figure 1. Face Detection Performance Results of DroneFace (Hsu, 2015)

## Face Recognition

FaceDrone also used Face++ and ReKognition to train face recognition models. These models performed much better TPR and FPR than was the case for the face detection task. It shows that there is a market for these licensed APIs, and they perform well for face recognition. However, the performance is highly dependent on angle of depression, or the angle created between the UAV and HOI­. These models are stable for most depression angles when the UAV is 4 meters from target. Unfortunately, these scores suffer as the angle of depression increases, especially for Face++.

Three models each for Face++ and ReKognition named J, P, and B are trained and tested with TPR as the performance metric. With 95% confidence, as a default match level, the trained models are tested to acquire the recognition score (TPR). A match is where the model matches the detected face with the correct profile. Results shows that ReKognition far exceeds the recognition score above the default match level then does Face++. The results from that study are shown below (Figure 2).

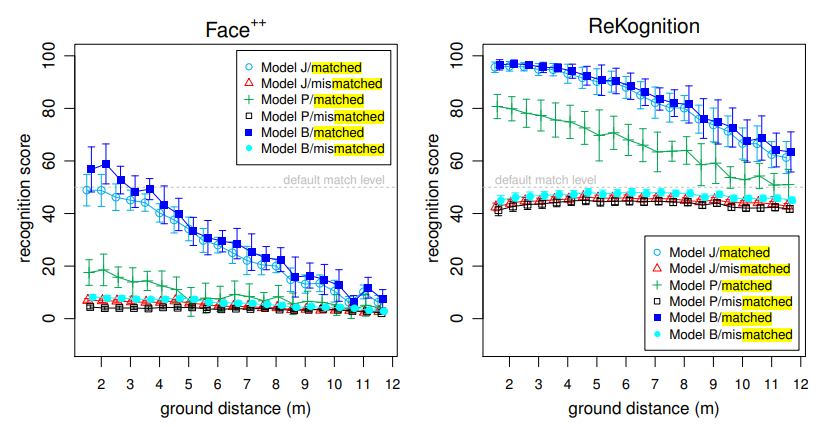


Figure 2. Matched and Mismatched Performance of Face++ and ReKognition models. (Hsu, 2015)

# Experiment

As previously mentioned, the dataset and models used are all open-source. In the case of YOLOLv8, this can be used for free, as long as monetary gain from the model is not needed. The raw data involves label generation and preprocessing. The preprocessing used include augmentations that are mentioned in more detail in the results section. The architectures are tuned using a basic policy that can be explored in more depth for future work.

## Dataset

The dataset is from DroneFace (Chen, 2015). It contains 620 GoPro (GoPro, n.d.) large-scale 24-bit RGB images of resolution 3680x2760 with a 170-degree field-of-view (FOV). Each of the large-scale frames include two to three HOIs on the center of the horizontal axis. However, there are times when several more HOIs are within the frame at different locations, because the data was collected on a university campus, so students and professors external to the experiment are seen walking through the campus. The intended HOI population include 11 subjects altogether. The footage is from 2-17 meters away from the HOIs with footage recorded at distances of 0.5m intervals. The UAV hovers either at 1.5 (eye-level), 3.0, 4.0, or 5.0 meters high.



Figure 3. Large-Scale UAV Frame

Additionally, there are 1364 24-bit RGB chipped images of 7 males and 4 females that are all HOIs within the large-scale raw images. Chipped images are images extracted from the bounding box coordinates of larger images. These chips range in resolution between 23x31 to 384x384 pixels depending on camera to HOI distance and angle. These images are all extracted from the large-scale images.



Figure 4. Example of Chipped Profile from Large-scale Images

There are an additional seven frontal images of the HOIs with white backgrounds with resolution 1174x1430.



Figure 5. Example of Ideal Profile Image

There are also 30 frontals, right, and left profile pictures of each HOI of resolution 696x696. However, since these images were not taken in a controlled environment, the background colors are not uniform.



Figure 6. Example of Left Profile Image

## Preprocessing

The dataset included chipped images of all HOIs, however, it does not provide bounding box information of the location of the detection within the large-scale image. The dataset is small enough to manually label, since it only has 620 large-scale images. For that reason, these images were manually labelled and will be used for match/unmatched data. Yolo\_Label (Kwon, 2023) was used to generate the labels. Each large-scale image had an associated text file with the same name as the image, and each object was listed (one object per line) in the format: class, upper left x, upper left y, lower right x, lower right y.

## Face Detection

As mentioned by other researchers including FaceDrone, a face can only be recognized if it can first be detected. For that reason, a well-known and generally high performing object detector was used for face recognition, namely You-Only-Look-Once Version 8 (Ultralytics YOLOv8, 2024). This version of YOLOv8 is different than that of Joseph Redmon’s implementation (Redmon J. , 2016), which uses darknet53 as a CNN backbone to the object detection algorithm.

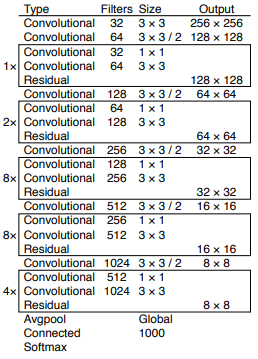


Figure 7. Darknet53 Backbone which YOLOv8 is derived from (Redmon J. F., 2018)

Unfortunately, YOLOv8 is not open-source and the full implementation, including the backbone is unknown by its users, but it is based on the YOLOv3 implementation. It is also more of a framework than YOLOv3, as it can be used for classification, segmentation, or object detection. Though it boasts of higher benchmark performance, in terms of speed and mean-average-precision (mAP) on common computer vision datasets, Ultralytics has yet to publish a paper on the subject. What makes YOLOv8 attractive is the ease of use and speed of development. Also, as long as monetary gain is not a goal of the use-case, YOLOv8 is a viable option.

## Face Recognition

The Face recognition, will likely be a Siamese network where both images are forward passed through a trained model and contrastive loss is used to compare each of the profile images to the chipped detection output. To recycle prior work, the VGG16 architecture can be used to implement the network.

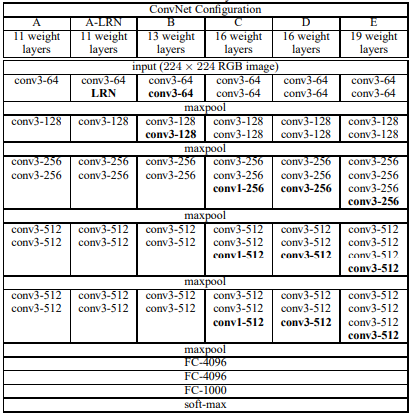


Figure 8. VGG-16 Architecture (Zisserrman, 2015)

The idea behind using VGG16, is that a pretrained checkpoint trained on ImageNet can be leveraged for model fine-tuning (Pytorch, 2017), which will likely lead to better overall performance. At least two of the fully connected layers removed from the network prior to contrastive loss calculation. If the difference between the profile image and chip is large beyond a predetermined threshold, then in theory, the people are not considered a match. The subject of Siamese networks and contrastive loss will be a greater focus of the final project. This will include a discussion of Euclidean distance, genuine and imposter pair, the threshold margin, contrastive loss, center loss, image preprocessing, and general training and testing of the Siamese network.

# Evaluation of Performance

TPR (1) is the metric used to compare performance of DroneFace and this face detection method. Face detection performance is expected to be comparable with LBP, because YOLOv8 is a reliable architecture capable of performing most object detection tasks well compared to other similar architectures. For face recognition, TPR will be used for matched and unmatched results. Unlike face detection, face recognition may not perform as well, because the model might be archaic in comparison to the Face++ and ReKognition models which were built with face recognition in mind. This will likely be updated as more research on the topic, and how those models were trained come to light.

(1)

# Results

## Face Detector Training and Testing Results

The training and validation sets are made of 620 large-scale manually labelled images, with a 10% split for validation. Because the dataset is so small, the dataset was only split two ways for train and validation, as opposed to training, validation, and testing. This methodology should be revisited pending procuration of new data.

Most of the truth labels are located along the center of the horizontal axis, but some faces also occur around the image boundaries. This due to bystander campus traversal. Altogether, there are about 1400 face instances that vary in width and height about 0.02 to 0.10 normalized units in comparison to the large-scale images. Normalization here refers to the spatial dimensions of the bounding box with respect to the size of the input image. The top right quadrant of Figure 9 shows the relative sizes of the bounding box to the image size. The lower quadrants show meta-data about the detections.

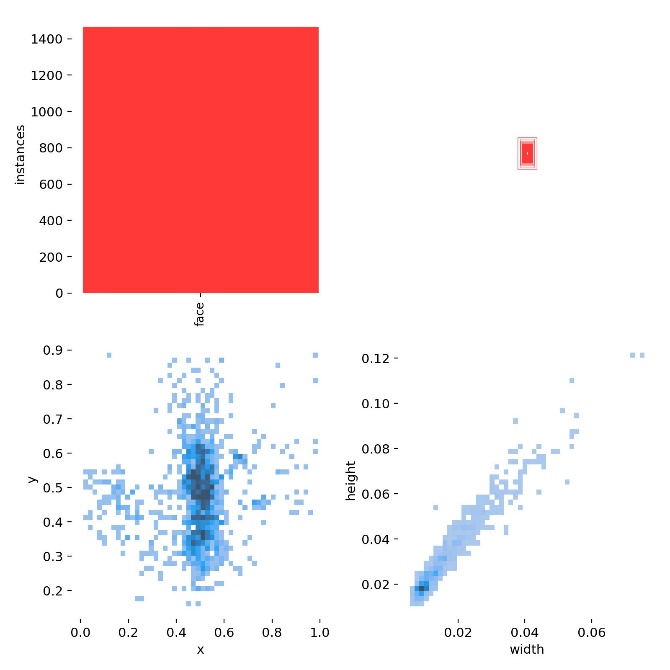


Figure 9. Basic Train Set Statistics

Moderate image augmentations were used to keep the model from overfitting the data. The augmentations used were horizontal flip, 30 percent rotations, RGB to HSV transfer, saturation, contrast, scaling, translation, copy-paste, and mosaic. An example of what that looks like can be seen in Figure 10.

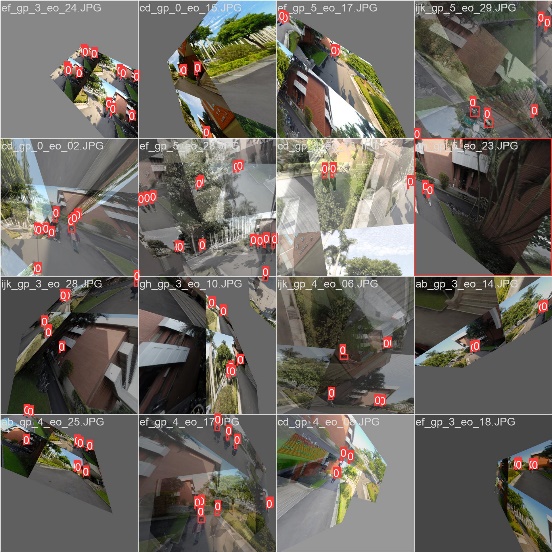


Figure 10. Augmentations Used During Training

The model trained for 65 epochs before converging to a F1 score of about 0.91 at a confidence of 0.50.

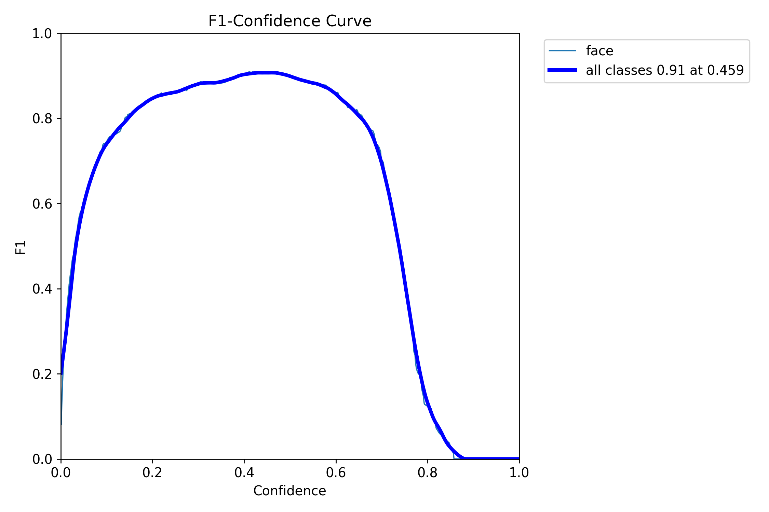


Figure 11. F1 Confidence Curve Obtained from the Validation Set

Other performance metrics were also gathered from training and generally show performance convergence, including train box loss, train class loss, train and test precision, train and test recall, and train and test mAP scores.

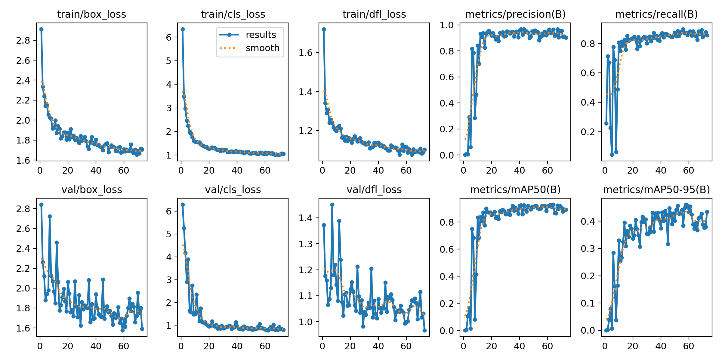


Figure 12. Common Performance Metrics

TPR was captured for the validation set and scored 0.96 at a 0% confidence and 0.92 on 0.5% confidence as shown by the Recall curve.

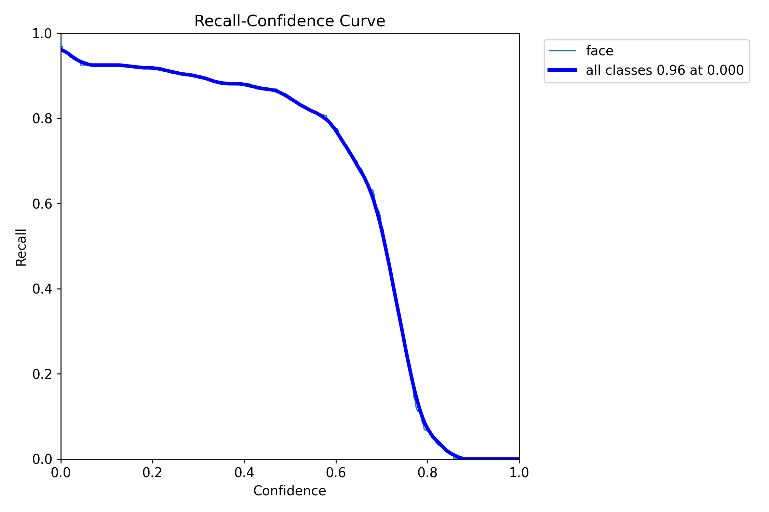


Figure 13. Recall vs. Confidence Curve on Validation Set

These results show that a face detector trained on a small amount of data can be implemented with YOLOv8. It also shows that this method performed better than the best LBP method used by the DroneFace authors. It should be noted, that this network was trained on the same dataset, but their actual train/test split is unknown. For that reason, direct score comparisons cannot be done. However, 14% is a big improvement and a solid start for face recognition of UAV footage.

## Face Classification

The cropped images included with the DnHFaces dataset are used to train a classifier on the eleven participants. These snapshots are like image chips output by the object detector. However, the object detector was only trained to detect a face within an image. However, the cropped images were labelled, making an identification classifier possible. In this case, each of the eleven people were labelled as a letter (a-k). Each class had 125 images, with a total dataset of about 1375.

Train, validation, and test sets were split randomly, with 70, 15, 15, percent stratified split respectively.

##### References

[1] Adam Pazke, E. a. (2017). Automatic Differentiation in Pytorch.

[2] Amazon. (2024, April 10). *What is Amazon Rekignition*. Retrieved from aws: https://docs.aws.amazon.com/rekognition/latest/dg/what-is.html

[3] Andreas Rossler, E. a. (2019, August 2019). FaceForensics++: Learning to Detect Manipulated Facial Images. Munich, Germany.

[4] Chen, H.-J. H.-T. (2015). *DroneFace*. Retrieved from DroneFace: https://hjhsu.github.io/DroneFace/

[5] GoPro. (n.d.). *GoPro*. Retrieved from https://gopro.com/en/us/

[6] Hsu, H.-J. a.-T. (2015, May). Face Recoognition on Drones: Issues and Limitations. Taipei, Taiwan.

[7] Kwon, Y. (2023). *Yolo\_Label*. Retrieved from github.com: https://github.com/developer0hye/Yolo\_Label

[8] OpenCV. (2024, April 9). *Face Detection using Haar Cascades*. Retrieved from OpenCV.org: https://docs.opencv.org/4.x/d2/d99/tutorial\_js\_face\_detection.html

[9] Pytorch. (2017). *VGG16*. Retrieved from pytorch.org: https://pytorch.org/vision/main/models/generated/torchvision.models.vgg16.html

[10] Redmon, J. (2016). *darknet*. Retrieved from pjreddie.com: https://pjreddie.com/darknet/

[11] Redmon, J. F. (2018, April 8). YOLOv3: An Incremental Improvement. Washington.

[12] Rosebrock, A. (2021, May 3). *Face Recognition with Local Binary Patterns (LBPs) and OpenCV*. Retrieved from pyimagesearch: https://pyimagesearch.com/2021/05/03/face-recognition-with-local-binary-patterns-lbps-and-opencv/

[13] *Ultralytics YOLOv8*. (2024). Retrieved from ultralytics.com: https://github.com/ultralytics/ultralytics

[14] Zisserrman, K. S. (2015, April 10). Very Deep Convolutional Networks For Large-Scale Image Recognition. Oxford, United Kingdom.