Face Detection and Emotion Recognition

Developed by Ballarin Tommaso - Spicoli Piersilvio - Onorati Jacopo

Project overview

The aim of this project is to design and implement a face detection and emotion recognition system, capable of analyzing a set of test images with corresponding ground-truth annotations. For each image, the system identifies the bounding boxes of all faces, predicts the emotional state of each detected face among 7 categories and computes a range of evaluation metrics to assess the performance of both the detection and recognition components.

The system addresses real-world challenges by detecting frontal, profile and rotated faces, and it applies Non-Maximum Suppression to remove duplicate and overlapping detections. Face detection is performed using Haar cascade classifiers, which are applied at multiple orientations and mirrored perspectives to improve robustness. Detected faces are extracted as Regions of Interest (ROIs), preprocessed and passed through a TensorFlow model for emotion recognition.

To evaluate performance the system computes precision, recall, Intersection over Union (IoU) and emotion recognition accuracy, both on individual images and as global statistics across the dataset. The processed images are displayed with annotated bounding boxes and predicted emotions, providing visual feedback alongside quantitative metrics.

Finally, this project demonstrates a complete computer vision pipeline that integrates detection, recognition, preprocessing and evaluation, highlighting the system's ability to accurately detect faces and classify emotions in diverse and challenging scenarios.

Implementation

The project development follows these steps:

1. **Acquisition and pre-processing:** the workflow begins with image acquisition, where all the available images are loaded from a designated dataset directory and a selection interface allows the user to specify which images to analyze. For each selected image, ground-truth annotations are imported from YOLO-format label files, converted into pixel coordinates and used as a reference for evaluation. The images are subsequently preprocessed by conversion to grayscale and histogram equalization to enhance contrast and improve robustness under varying illumination conditions.

2. **Face detection and post-processing:** Haar cascade classifiers are used for both frontal and profile views, including detection on flipped images, to capture left-facing profiles and, on rotated versions (±10°), to handle tilted faces. The resulting candidate detections are then aggregated into a single set. To improve detection reliability, a sequence of post-processing filters is applied: removal of bounding boxes

with unrealistic sizes, exclusion of boxes with implausible aspect ratios, suppression of nested detections where a smaller box lies within a larger one and non-maximum suppression (NMS) to eliminate duplicate detections.

3. **Model training:** he data was sourced from the FER2013 Facial Expression Recognition dataset, which was initially presented as a Kaggle competition: https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition-challenge/data. This dataset consists of 48×48 grayscale images of faces, each annotated with one of seven facial expressions (0=angry, 1=disgust, 2=fear, 3=happy, 4=sad, 5=surprise, 6=neutral). The training set contained 28,709 images while both the public and private test sets contained 3,589 images each. In the notebook, images are normalized by dividing pixel values by 255 and further standardized using ImageDataGenerator with feature-wise centering and normalization. Data augmentation is applied during training with random rotations up to 20°, zoom up to 0.15, horizontal and vertical shifts up to 0.2, shear transformations up to 0.15 and horizontal flipping. Images are reshaped to include a single-channel dimension, producing tensors of size 48×48×1. The convolutional neural network consists of an input layer of 48×48×1 followed by a Conv2D block with 32 filters of size 3×3 using ReLU activation, batch normalization, max pooling with a 2×2 window and dropout 0.25. This is followed by two additional Conv2D layers with 32 filters and ReLU, batch normalization, max pooling and dropout 0.25. The feature maps are flattened and connected to a fully connected layer with 512 units, ReLU activation, batch normalization and dropout 0.5. The final output layer is a softmax classifier over the seven target classes. Training is performed with the Adam optimizer at a learning rate of 3e-4, using sparse categorical crossentropy as the loss function and a batch size of 256. The model is trained for up to 1000 epochs with EarlyStopping monitoring validation accuracy with patience 50. Before the final training, a learning rate range test is conducted sweeping values from 1e-8 to 1e0 across about 20 epochs to identify a suitable initial learning rate. At the end of training the model is exported in .pb format to enable inference with our system, with a separate saving script written in py

4. **Emotion recognition:** Each detected face is extracted as a Region of Interest (ROI) and undergoes preprocessing steps such as resizing and normalization to match the input requirements of the trained emotion recognition model. The system then performs emotion classification using a custom TensorFlow convolutional neural network, which outputs a probability distribution over the possible emotion classes. The most likely class is selected as the predicted emotion for each face.

5. **Visualization of the results and image saving:** the system visualizes the results by displaying the input image with ground-truth bounding boxes in red and predicted bounding boxes in distinct colors, also overlaying the predicted emotion labels. Then, all the selected images are saved in /output.

6. **Evaluation metrics for face detection:** to evaluate system performance, a set of standard metrics is computed for both face detection and emotion recognition. For face detection, True Positives (TP), False Positives (FP) and False Negatives (FN)

are calculated based on the Intersection over Union (IoU) between predicted and ground-truth bounding boxes. A detection is considered correct if its IoU exceeds a predefined threshold (IoU_threshold = 0.45). Using these counts, precision (TP / (TP + FP)) and recall (TP / (TP + FN)) are computed for each image to assess the system's ability to correctly detect faces while minimizing false alarms.

7. **Evaluation metrics for emotion recognition:** per-image classification accuracy is computed by comparing the predicted emotion label of each correctly detected face with its corresponding ground-truth label.

8. **System-level evaluation:** finally, all the metrics are aggregated to obtain overall results across the entire dataset, enabling a comprehensive evaluation of both detection and classification performance.

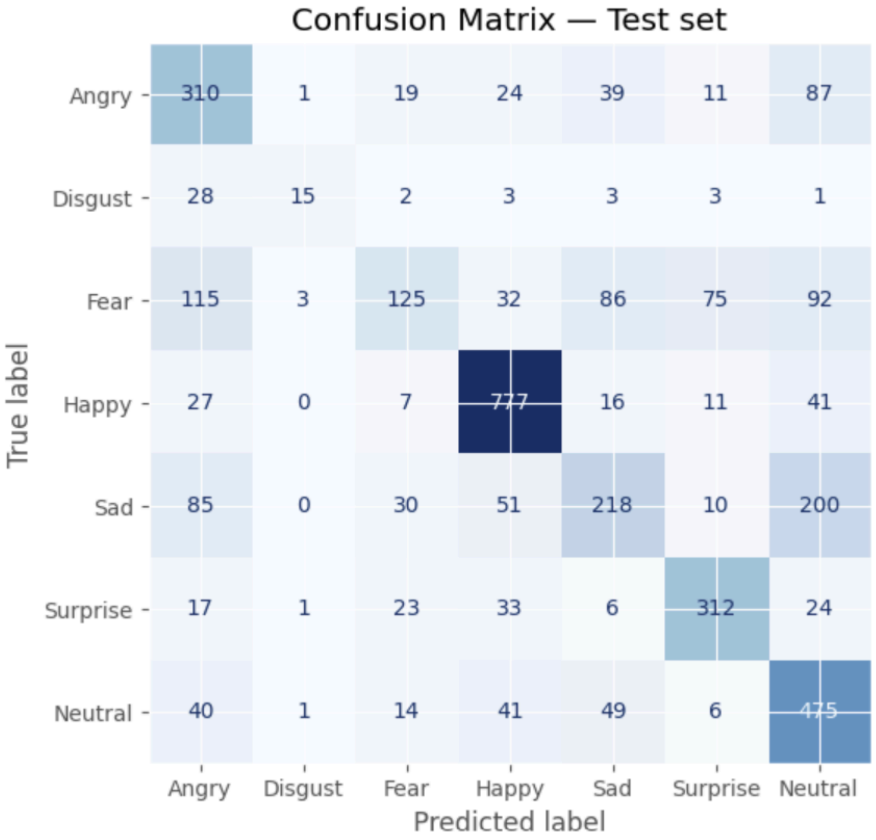
9. **Model-level evaluation:** The trained model was evaluated on the PrivateTest set to provide an unbiased assessment of its performance. A classification report was generated to compute precision, recall and F1-score for each class, allowing the identification of classes that the model recognized reliably and those where it struggled due to factors such as data imbalance in icml_face_data.csv. A confusion matrix was also calculated using a confusion matrix and visualized with sklearn to highlight misclassification patterns across emotions. These evaluation metrics were chosen to provide a comprehensive overview that goes beyond overall accuracy, capturing both the strengths and limitations of the trained convolutional neural network

Performance

The system is tested on a dataset of 46 labeled images, each with its respective ground-truth bounding boxes shown in red. Performance metrics are computed for each image and the overall results are as follows:

- **Overall precision** = 91.53%
- **Overall recall** = 76.06%
- **Overall emotion accuracy** = 51.85%

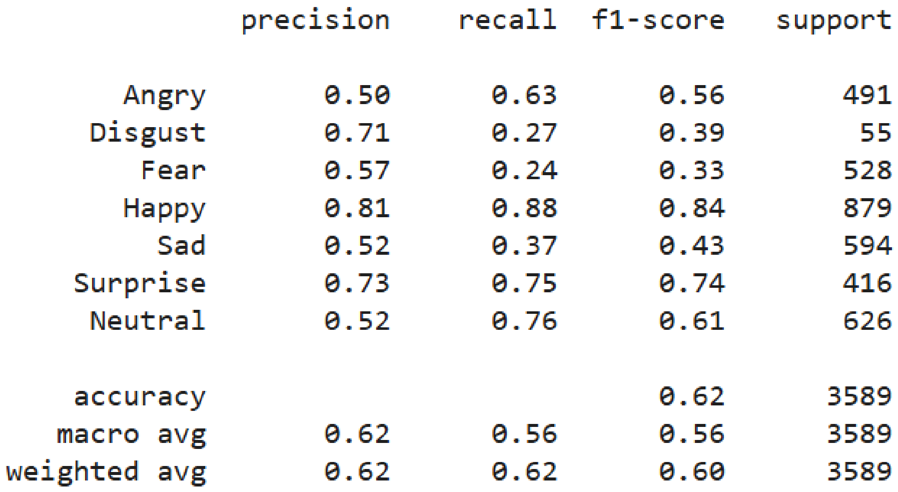
Regarding the **evaluation of the CNN model** developed in the "models/CNN_model.ipynb" notebook, both the confusion matrix and the accuracy were calculated using the test split of the icml_face_data.csv dataset. A complete description of the training pipeline, validation strategy and detailed evaluation results are available directly in the aforementioned notebook within the repository.



Confusion Matrix — Test set

	Angry	Disgust	Fear	Happy	Sad	Surprise	Neutral
Angry	310	1	19	24	39	11	87
Disgust	28	15	2	3	3	3	1
Fear	115	3	125	32	86	75	92
Happy	27	0	7	777	16	11	41
Sad	85	0	30	51	218	10	200
Surprise	17	1	23	33	6	312	24
Neutral	40	1	14	41	49	6	475

The result above reported highlights heterogeneous performance across the different emotion classes. The model shows strong recognition for **Happy** (777 correctly classified instances) and **Neutral** (475) as well as a solid performance on **Angry** (310). On the other hand classes such as **Fear** and **Sad** present a higher number of misclassifications often confused with **Neutral** and **Angry** indicating overlapping features among these emotions. **Disgust** also shows limited recognition with frequent misclassification into other categories. Overall the results suggest that while the model is effective in detecting more distinctive emotions it struggles with those that share visual similarities leaving room for improvement in class separation.

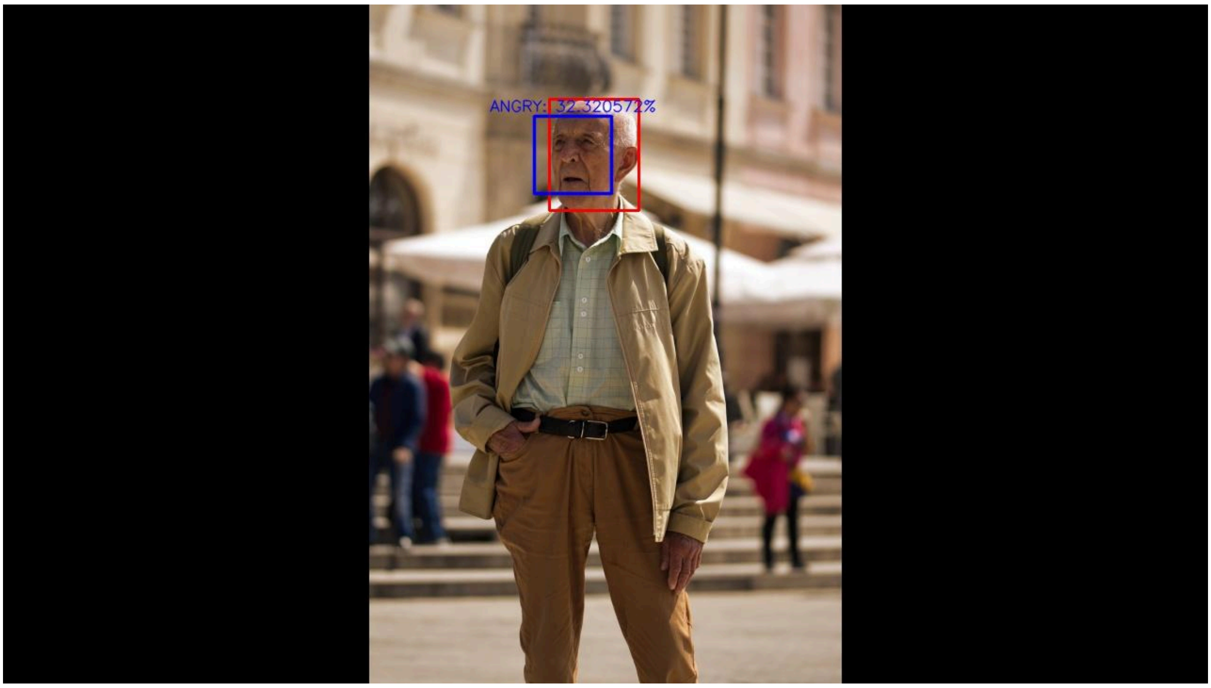


```
              precision    recall  f1-score   support

       Angry       0.50      0.63      0.56       491
     Disgust       0.71      0.27      0.39        55
        Fear       0.57      0.24      0.33       528
       Happy       0.81      0.88      0.84       879
         Sad       0.52      0.37      0.43       594
    Surprise       0.73      0.75      0.74       416
     Neutral       0.52      0.76      0.61       626

    accuracy                           0.62      3589
   macro avg       0.62      0.56      0.56      3589
weighted avg       0.62      0.62      0.60      3589
```

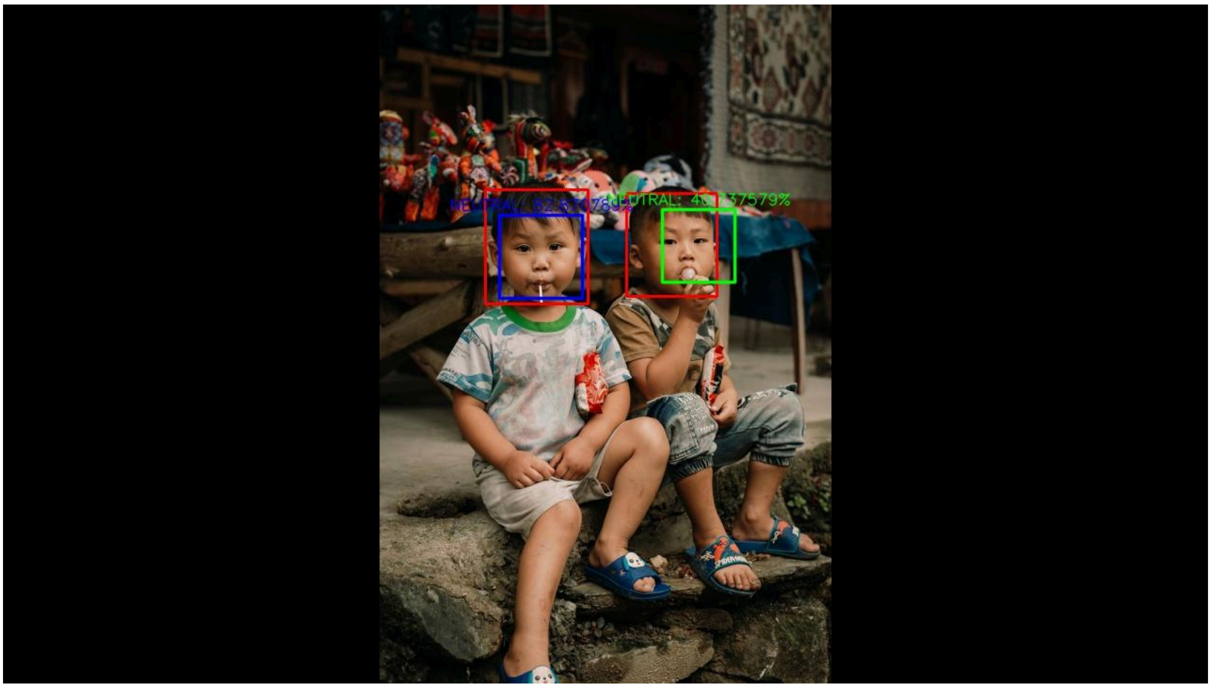
The classification report on the test set shows an overall accuracy of **0.62** with similar values for the macro and weighted averages. The best performance is obtained for **Happy** with an f1-score of **0.84** followed by **Surprise** (**0.74**) and **Neutral** (**0.61**). On the other hand the model struggles with **Fear** (**0.33**), **Sad** (**0.43**) and especially **Disgust** (**0.39**) where recall is particularly low (**0.27**). These results indicate that while the network is effective in recognizing distinctive emotions such as happiness it has difficulties in distinguishing more subtle or visually overlapping categories which lowers the balance across classes.

All the 46 test images are shown below:

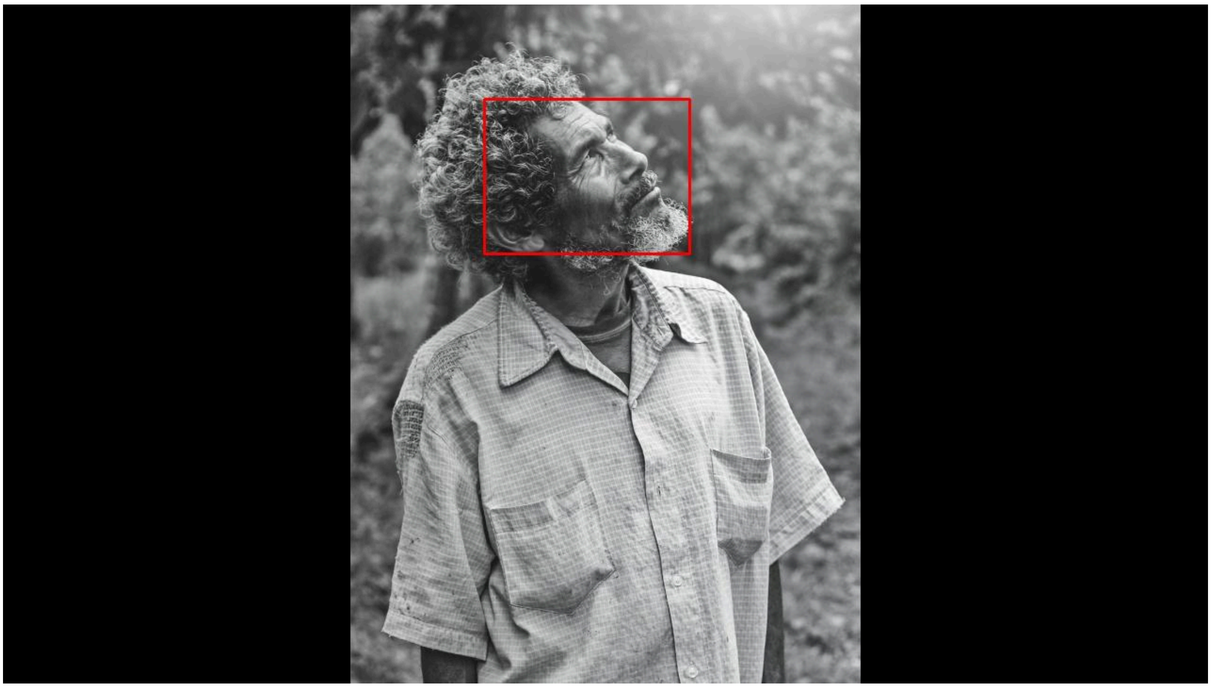
- **"surprise (1).jpg" Precision** = 0 **Recall** = 0 **Mean IoU** = 0.4356 **Accuracy** = 0



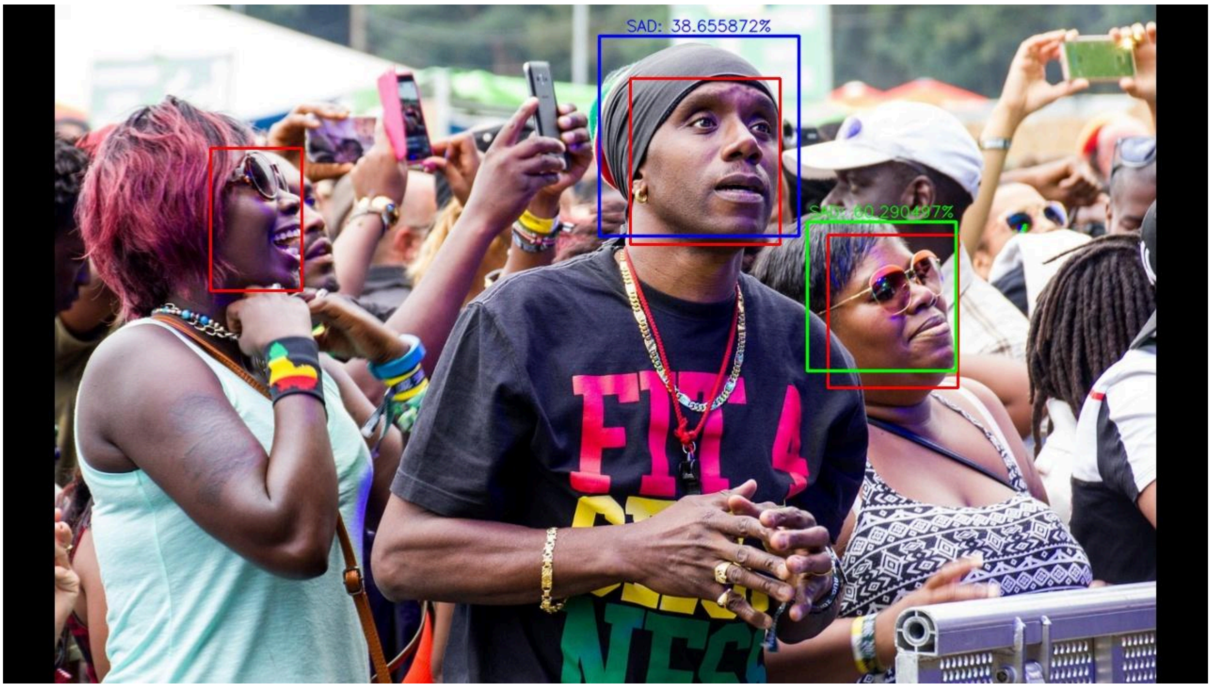
- **“surprise (2).jpg” Precision** = 0.5 **Recall** = 0.5 **Mean IoU** = 0.4841 **Accuracy** = 0



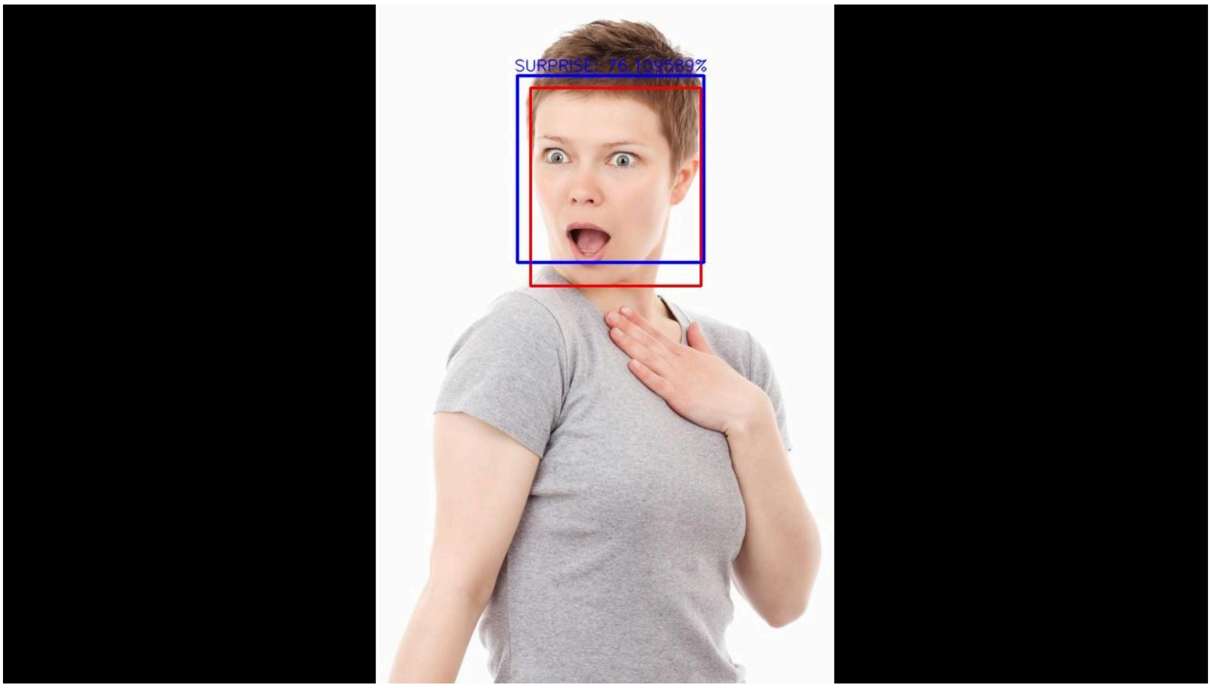
- **“surprise (3).jpg”** Face not detected



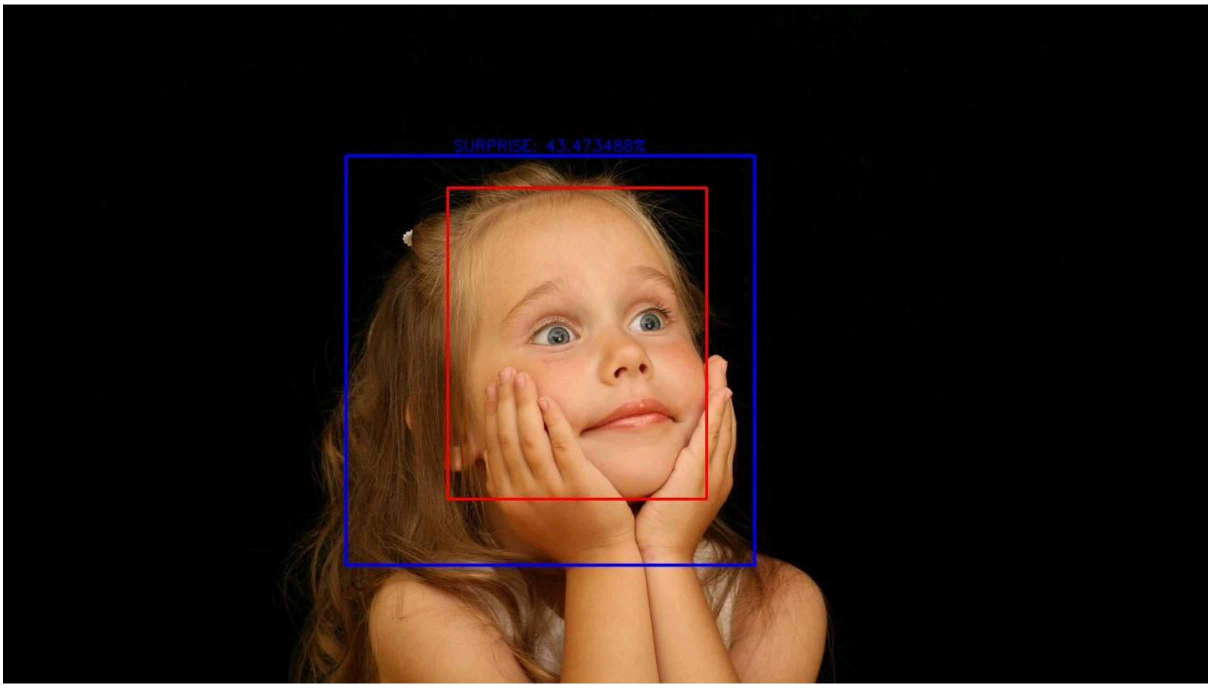
- **“surprise (4).jpg” Precision** = 1 **Recall** = 0.67 **Mean IoU** = 0.6246 **Accuracy** = 0



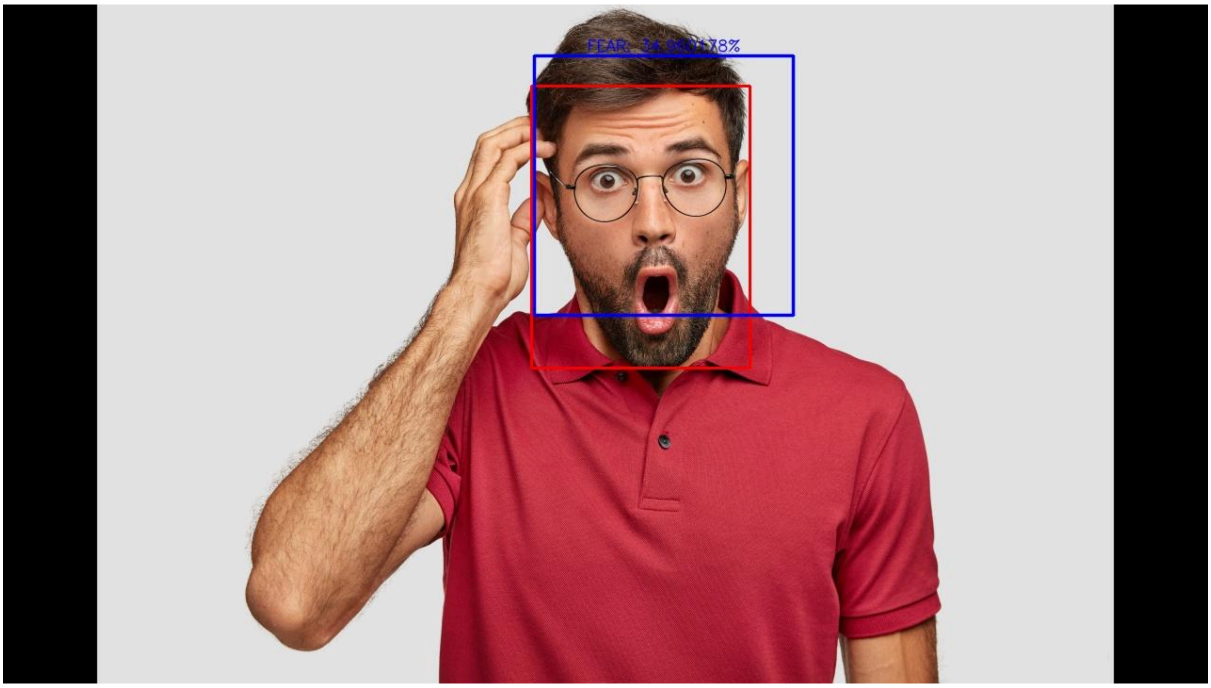
- **“surprise (5).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** = 0.7662 **Accuracy** = 1



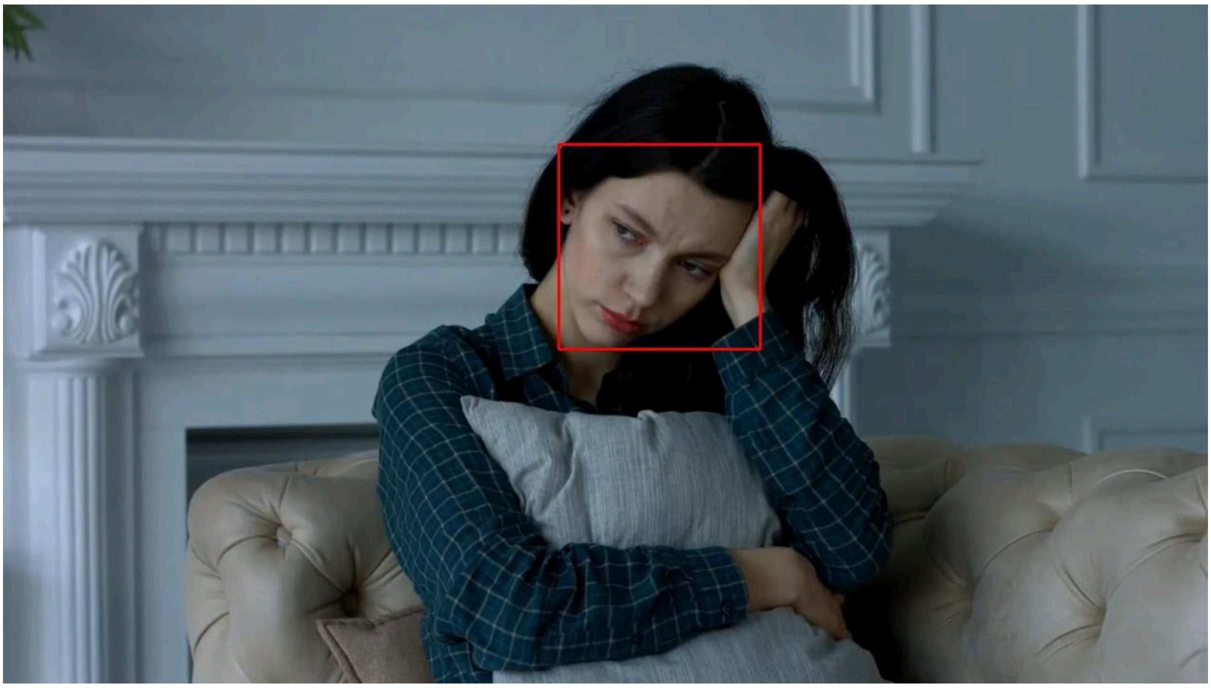
- **“surprise (6).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** = 0.4833 **Accuracy** = 1



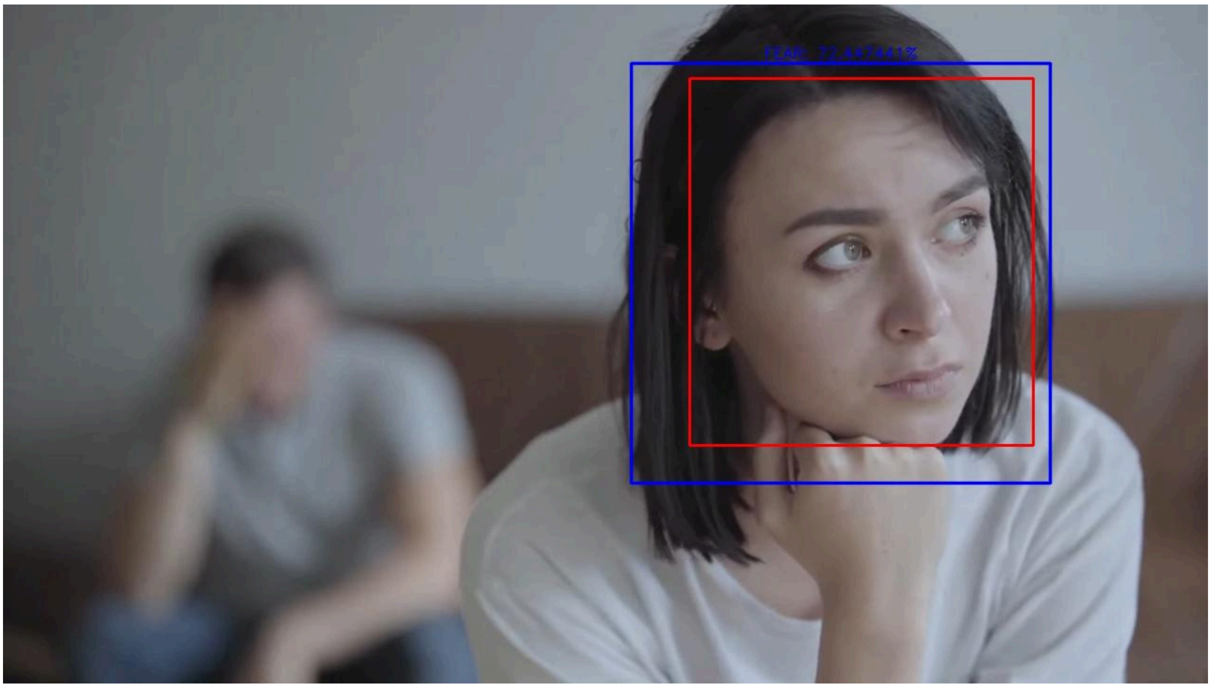
- **“surprise (7).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** = 0.6227 **Accuracy** = 0



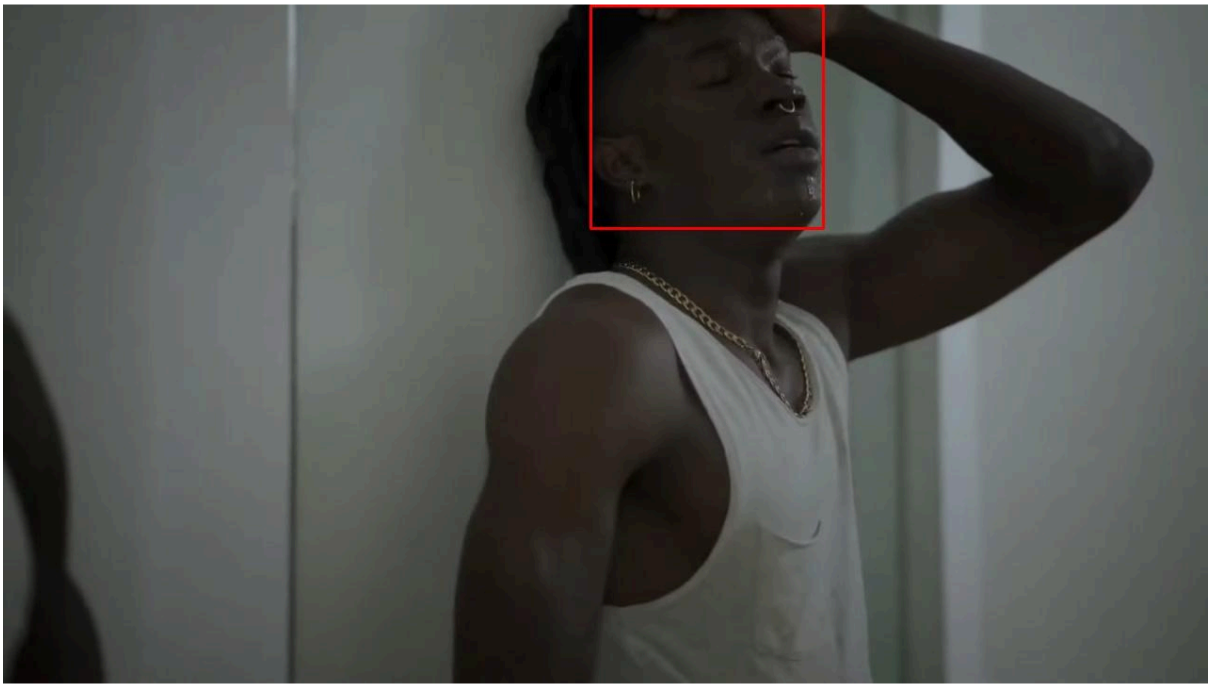
- **“sad (1).jpg”** Face not detected



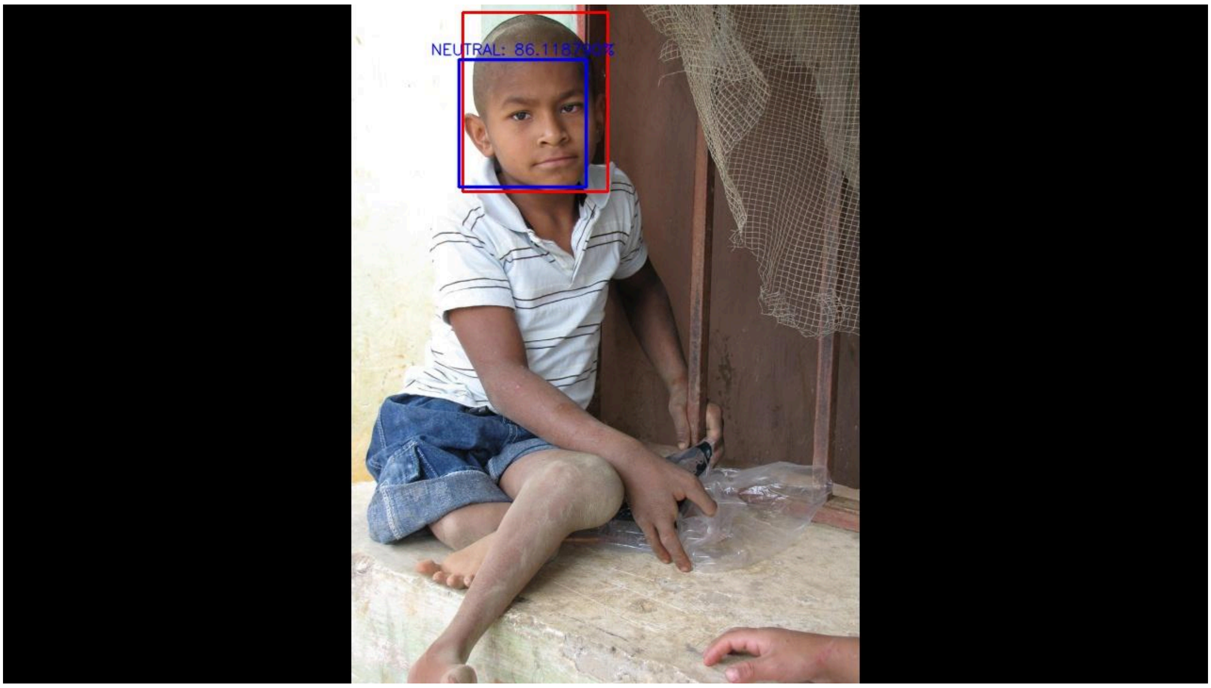
- **“sad (2).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** = 0.7175 **Accuracy** = 0



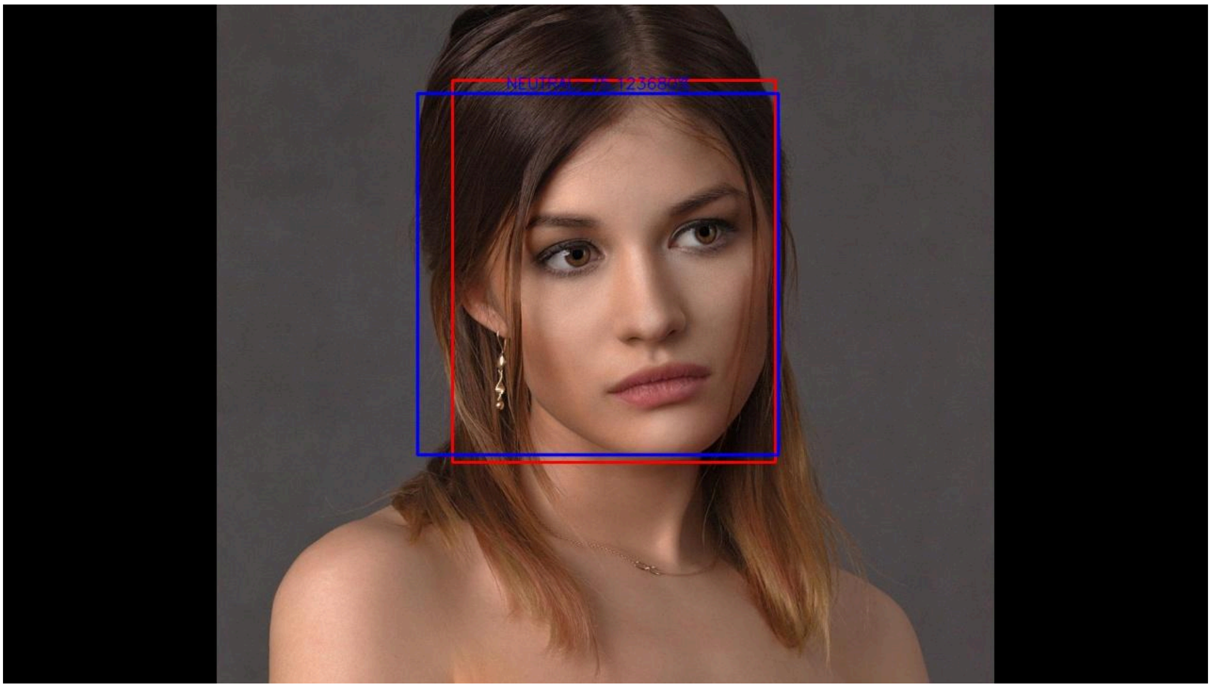
- **“sad (3).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** = 0.5929 **Accuracy** = 0



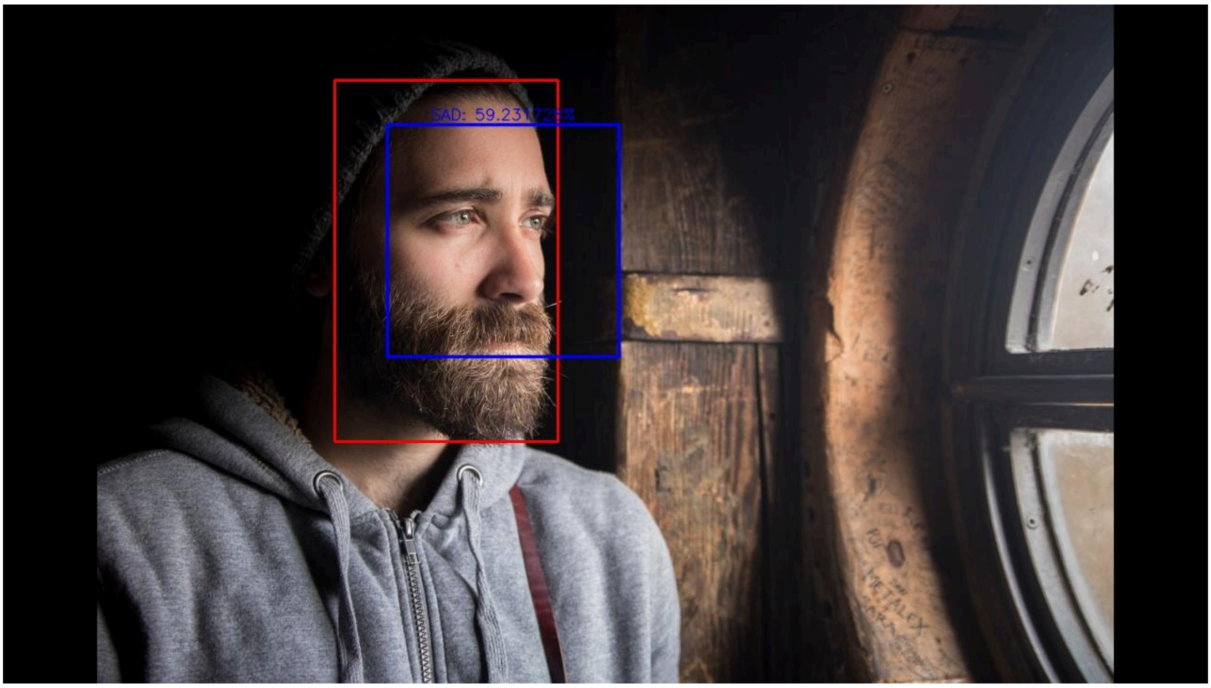
- **“sad (4).jpg”** Face not detected



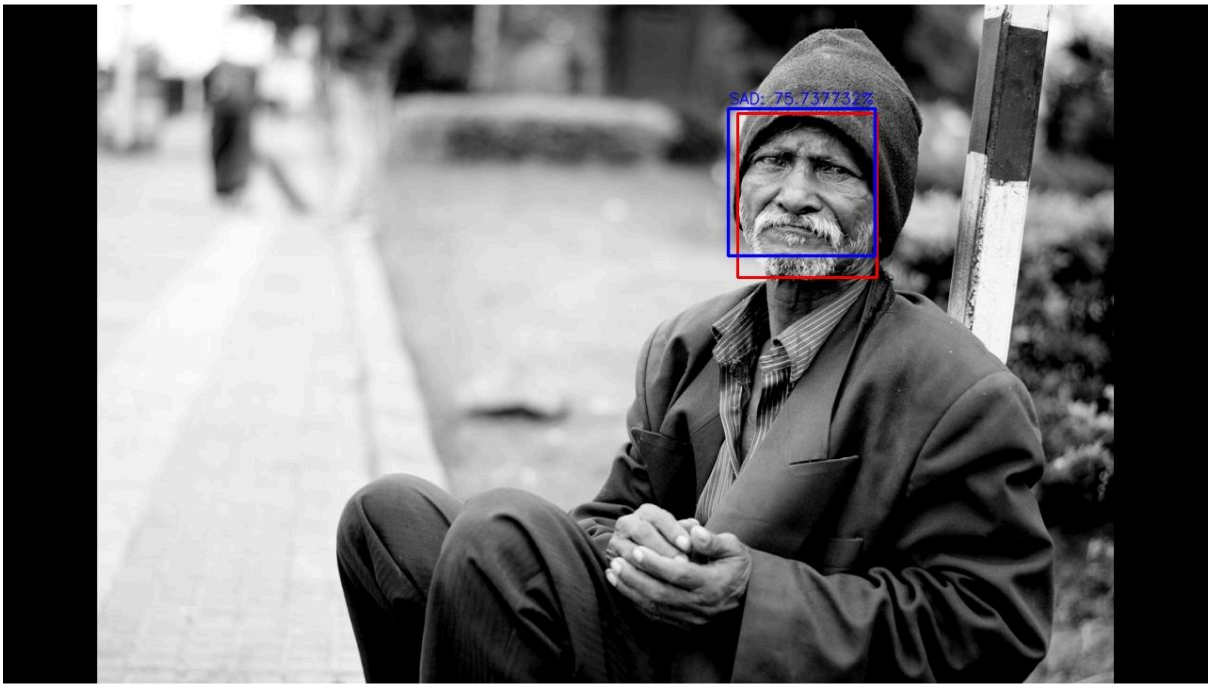
- **“sad (5).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** = 0.8521 **Accuracy** = 0



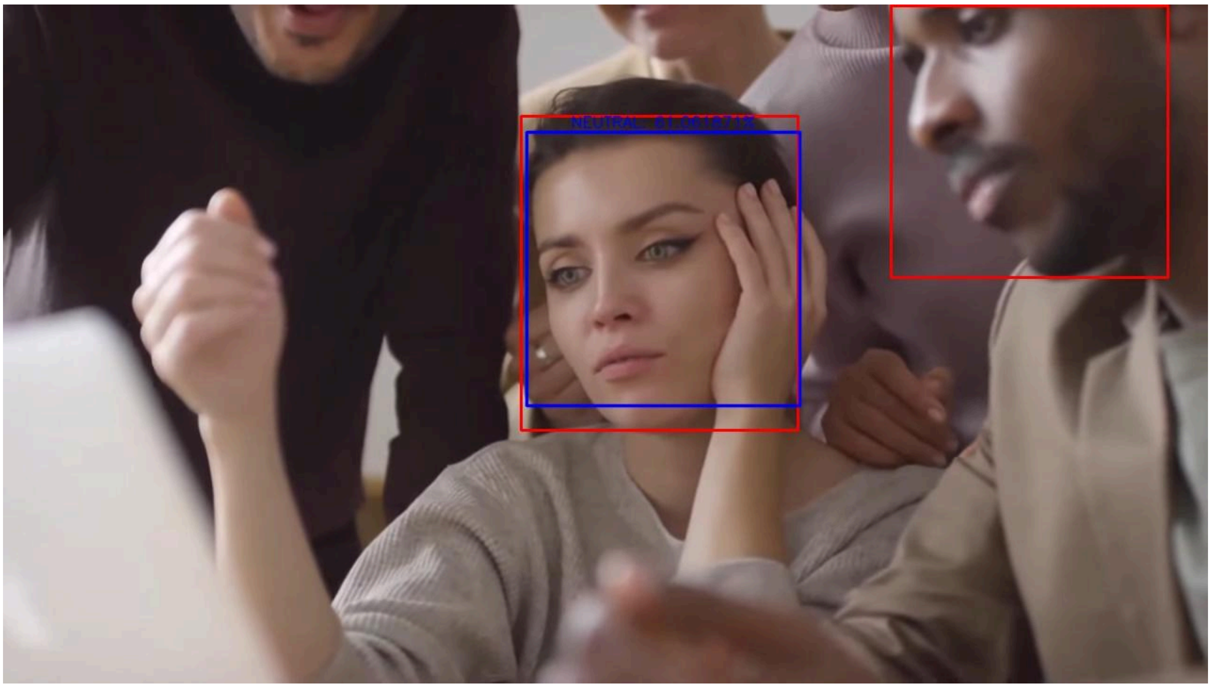
- **“sad (6).jpg” Mean IoU** = 0.4167 < IoU_threshold, face not detected



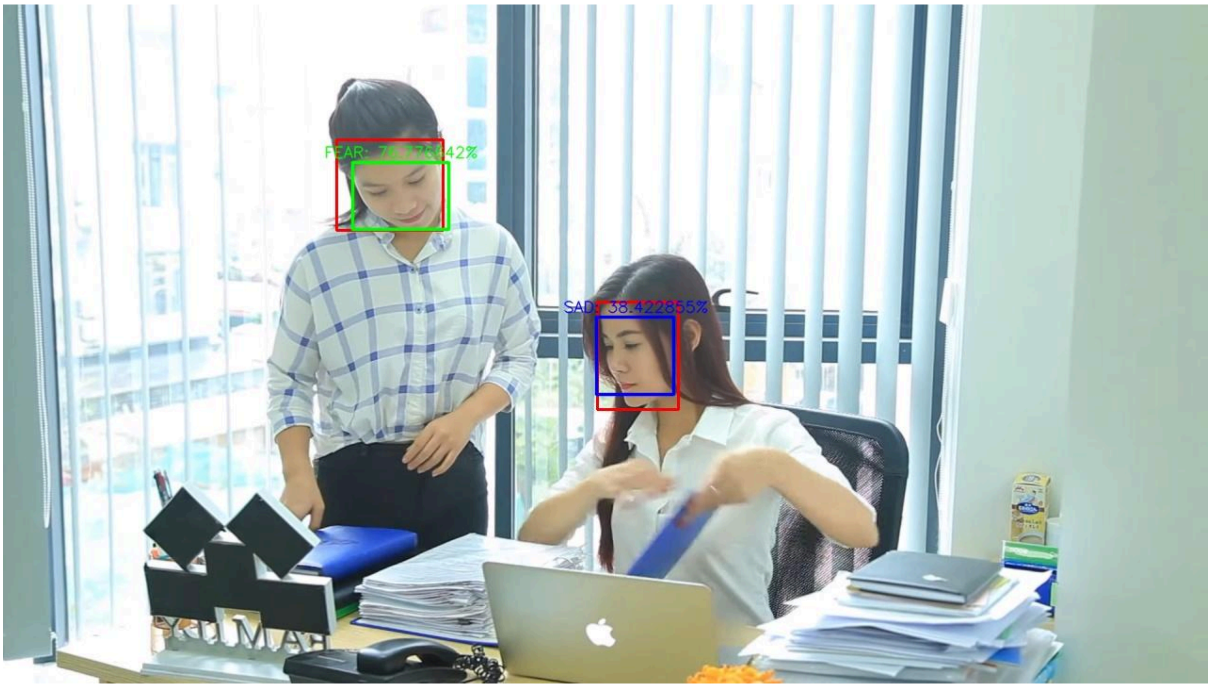
- **“sad (7).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** = 0.7846 **Accuracy** = 1



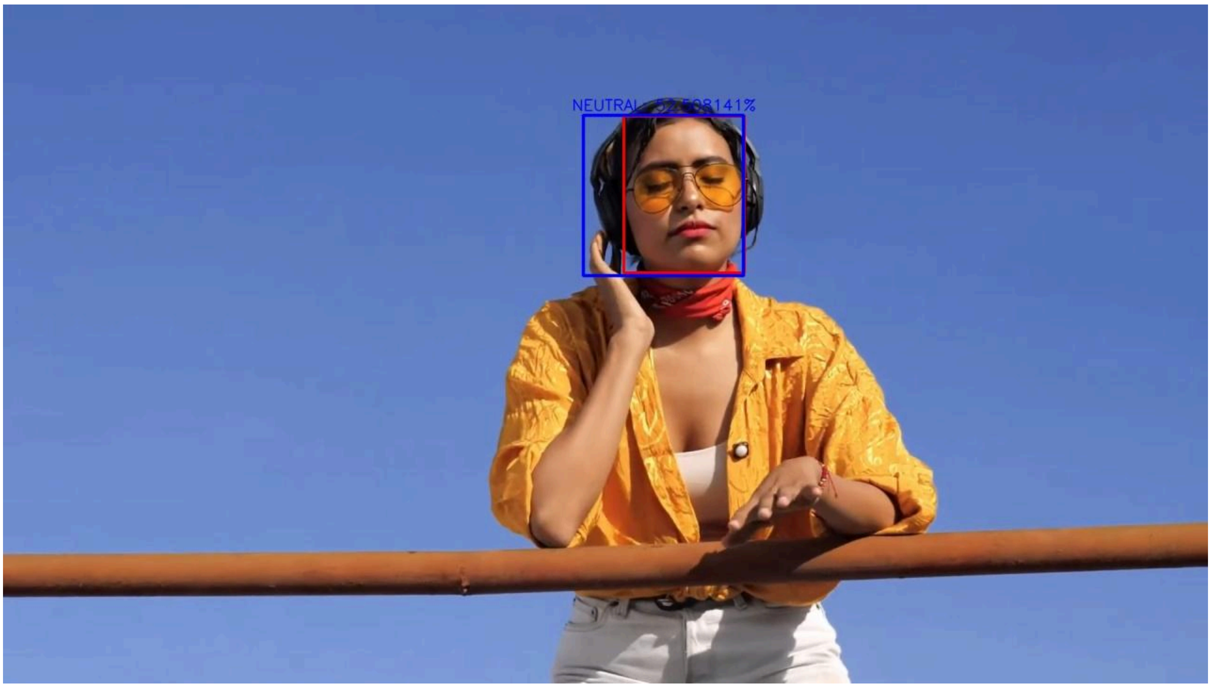
- **“neutral (1).jpg” Precision** = 1 **Recall** = 0.5 **Mean IoU** = 0.8478 **Accuracy** = 1



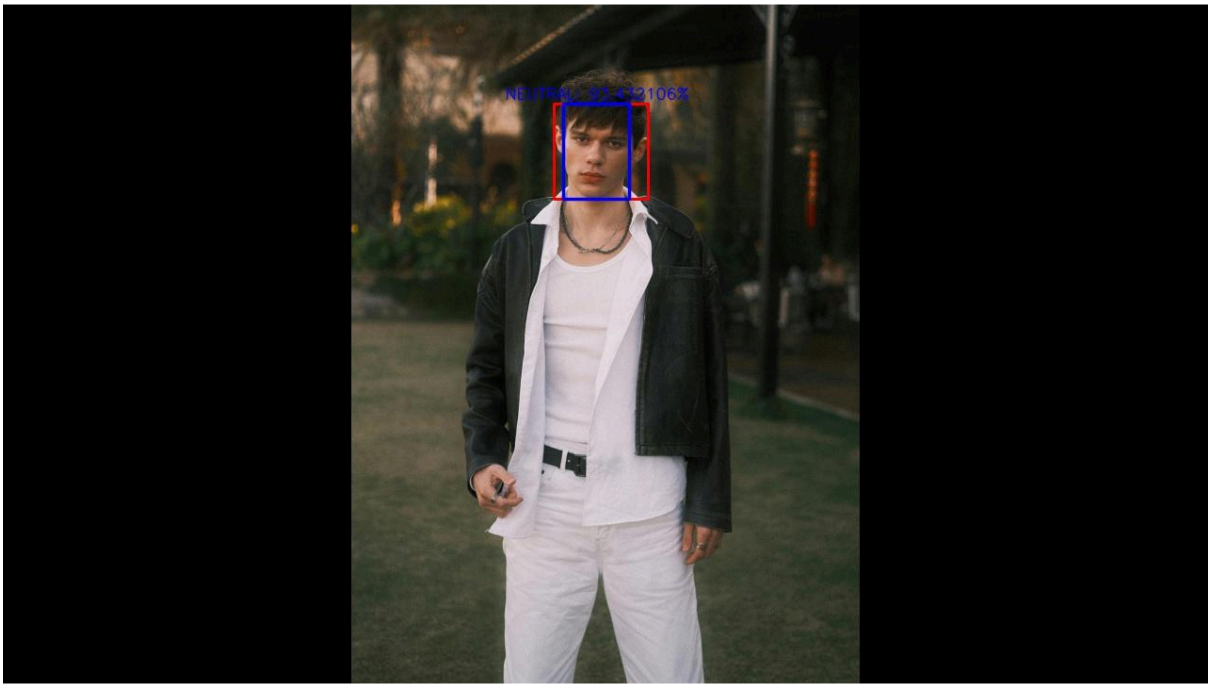
- **“neutral (2).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** = 0.6453 **Accuracy** = 0



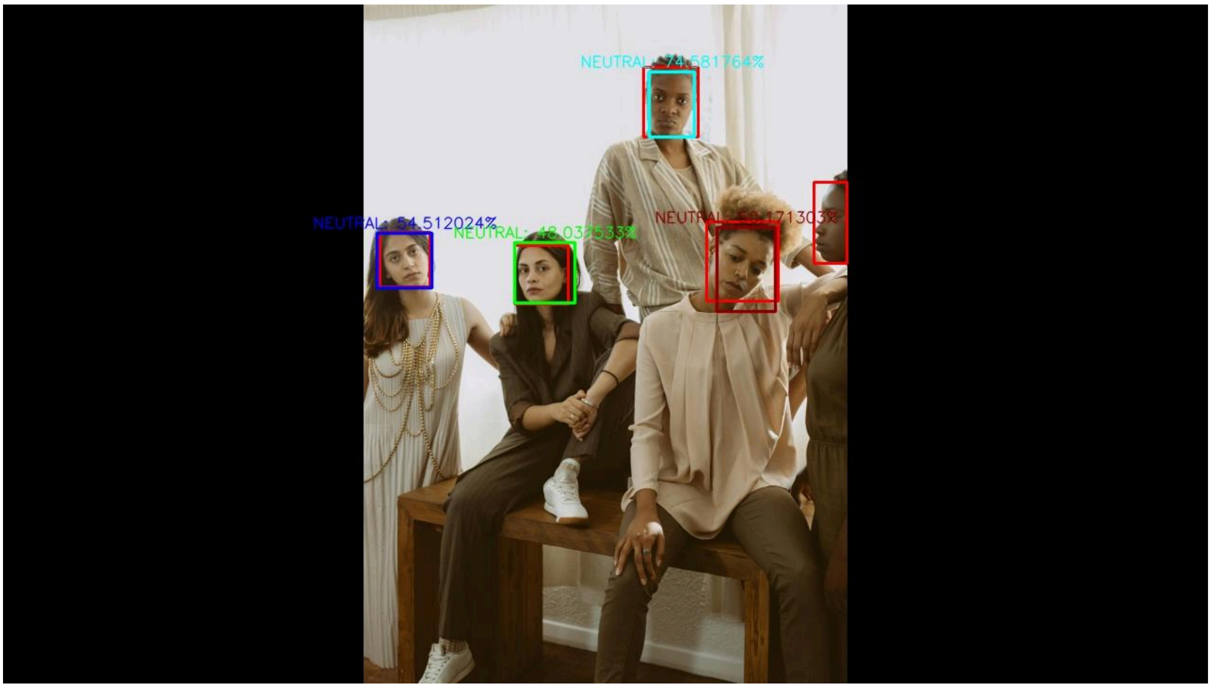
- **“neutral (3).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** = 0.7297 **Accuracy** = 1



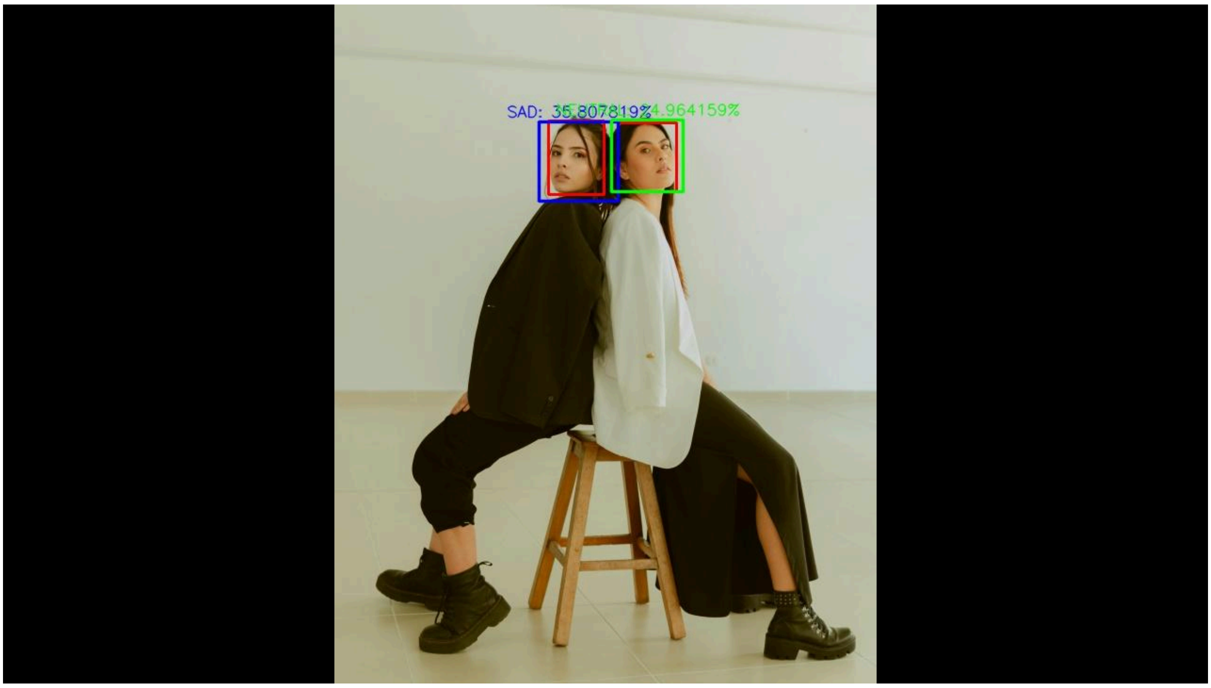
- **“neutral (4).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** = 0.7077 **Accuracy** = 1



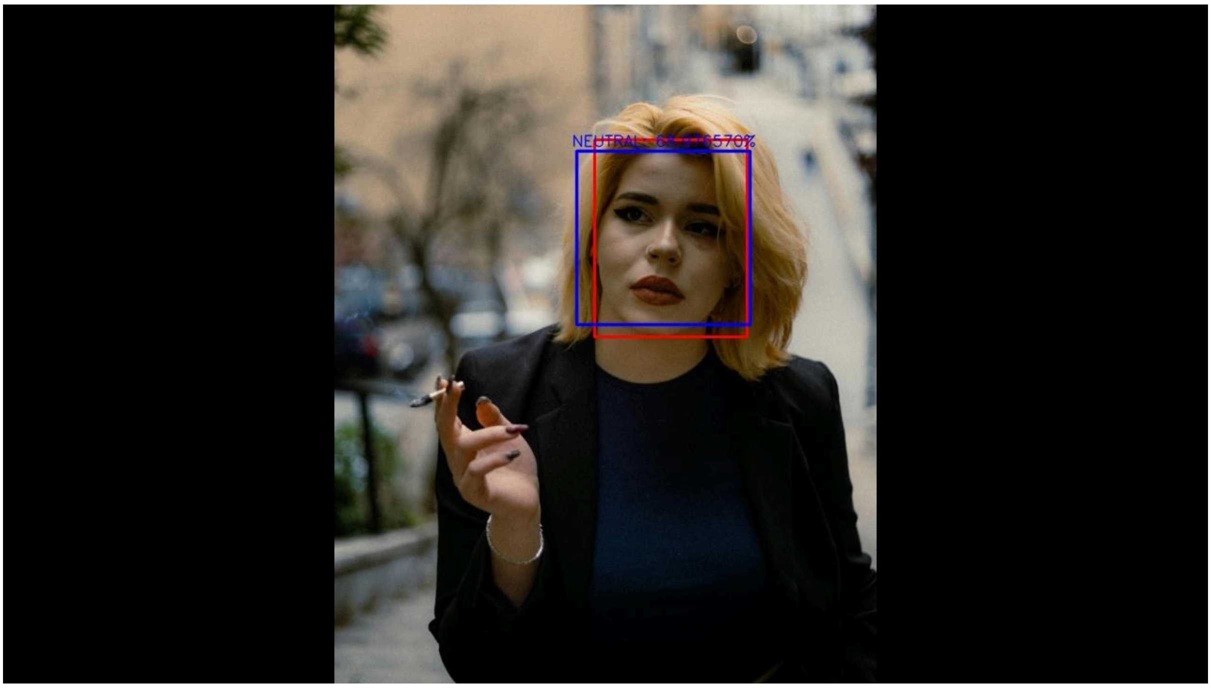
- **“neutral (5).jpg” Precision** = 1 **Recall** = 0.8 **Mean IoU** = 0.7993 **Accuracy** = 1



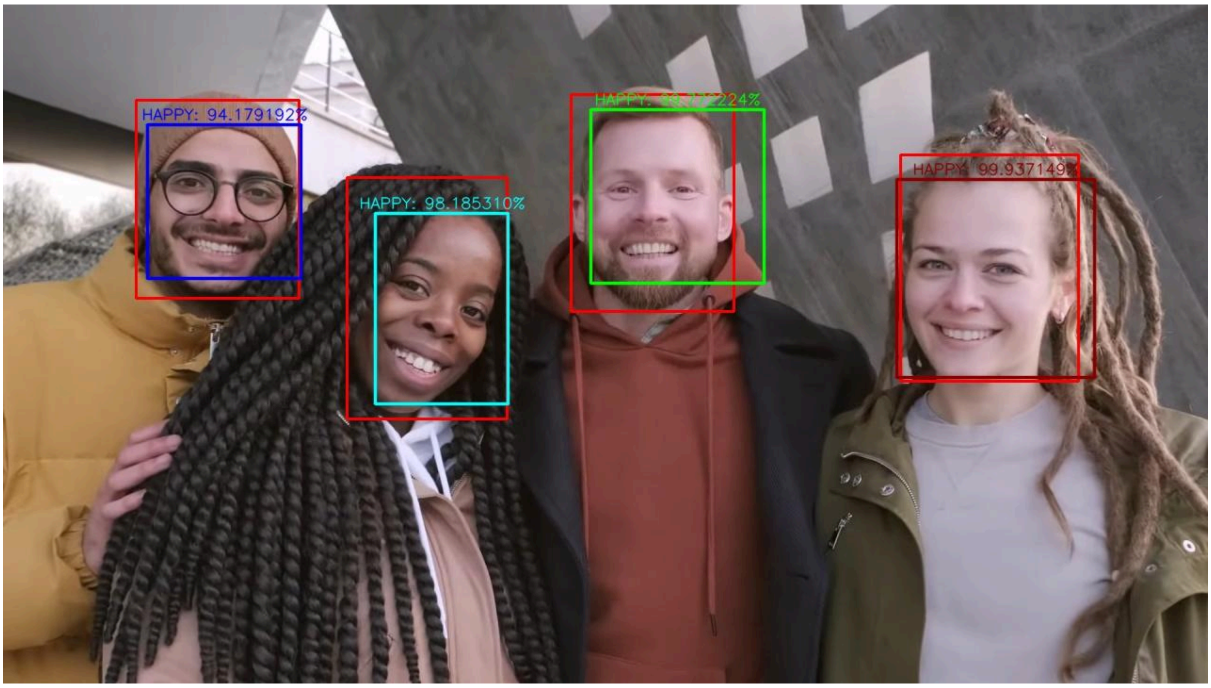
- **“neutral (6).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** = 0.4740 **Accuracy** = 0.5



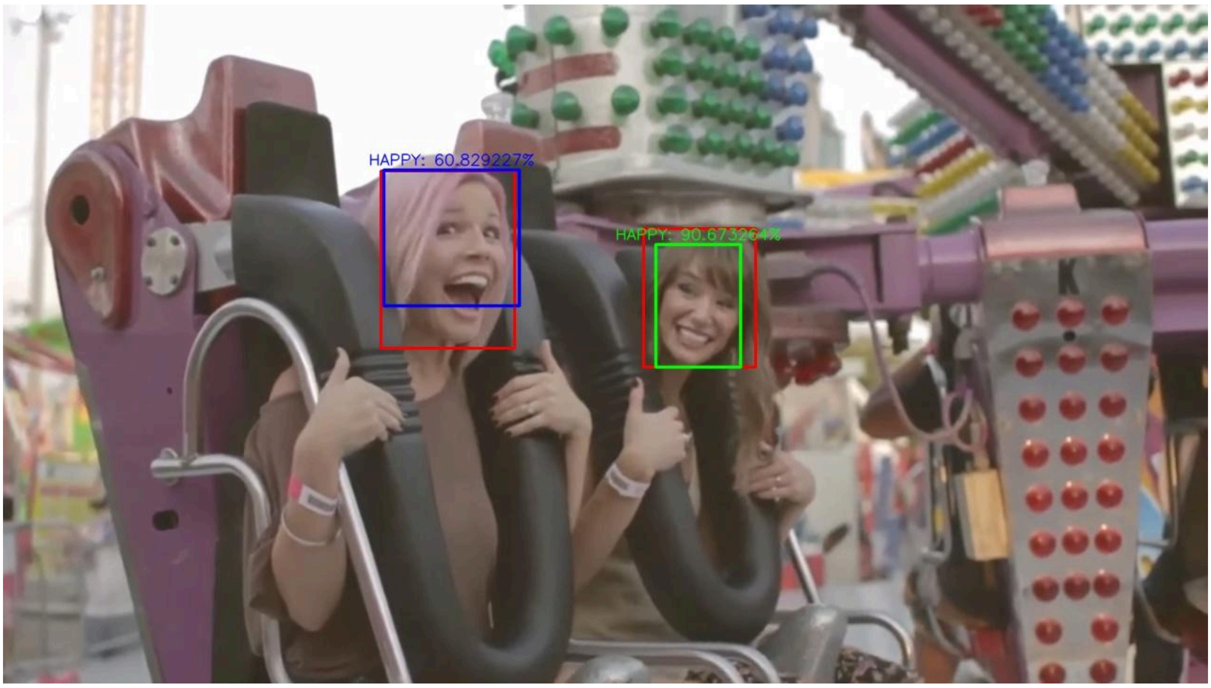
- **“neutral (7).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** = 0.7876 **Accuracy** = 1



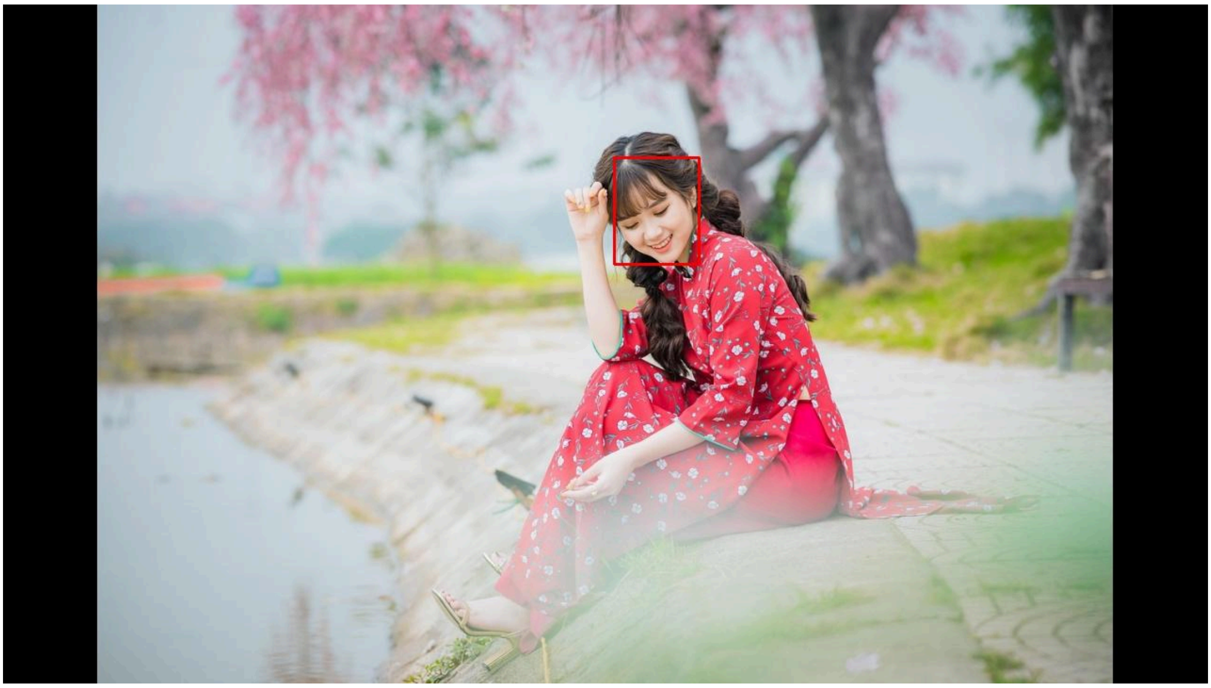
- **“happy (1).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** = 0.6953 **Accuracy** = 1



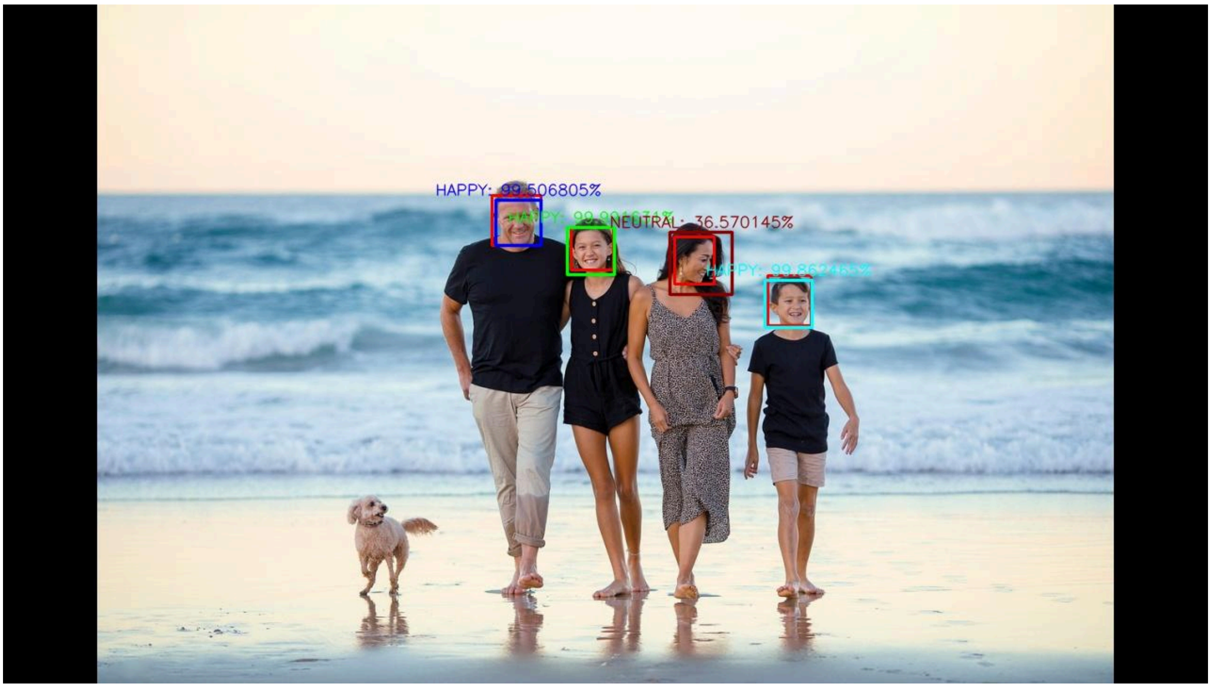
- **“happy (2).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** = 0.6974 **Accuracy** = 1



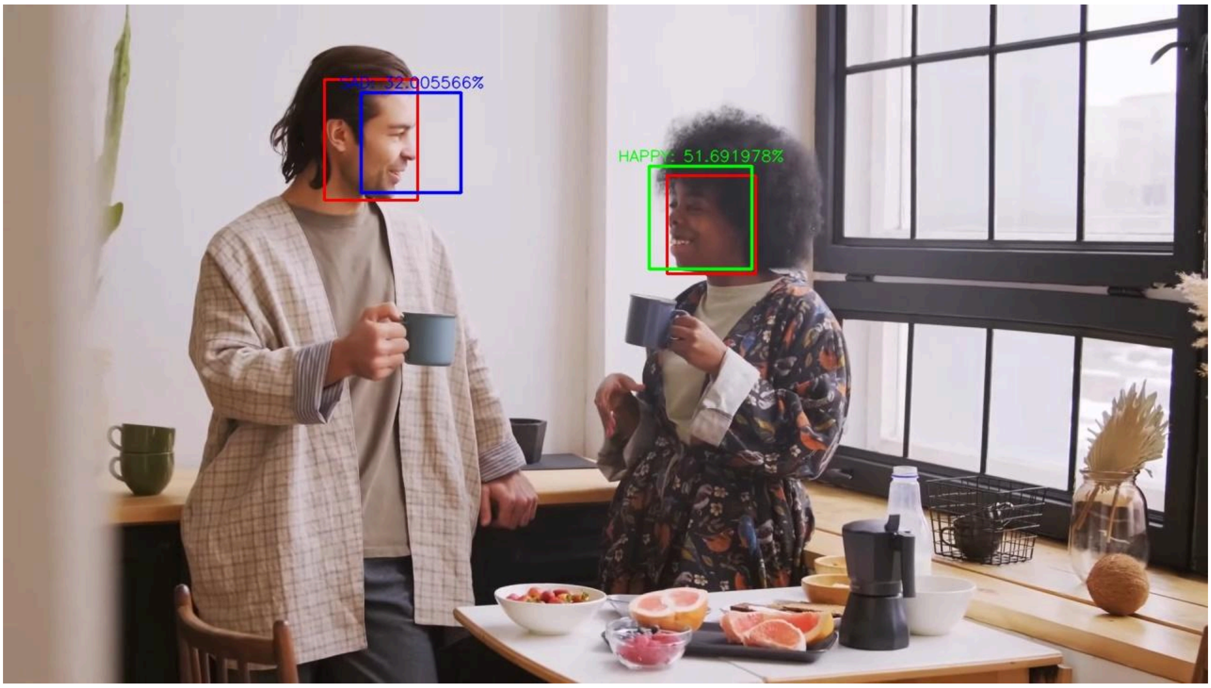
- **“happy (3).jpg”** Face not detected



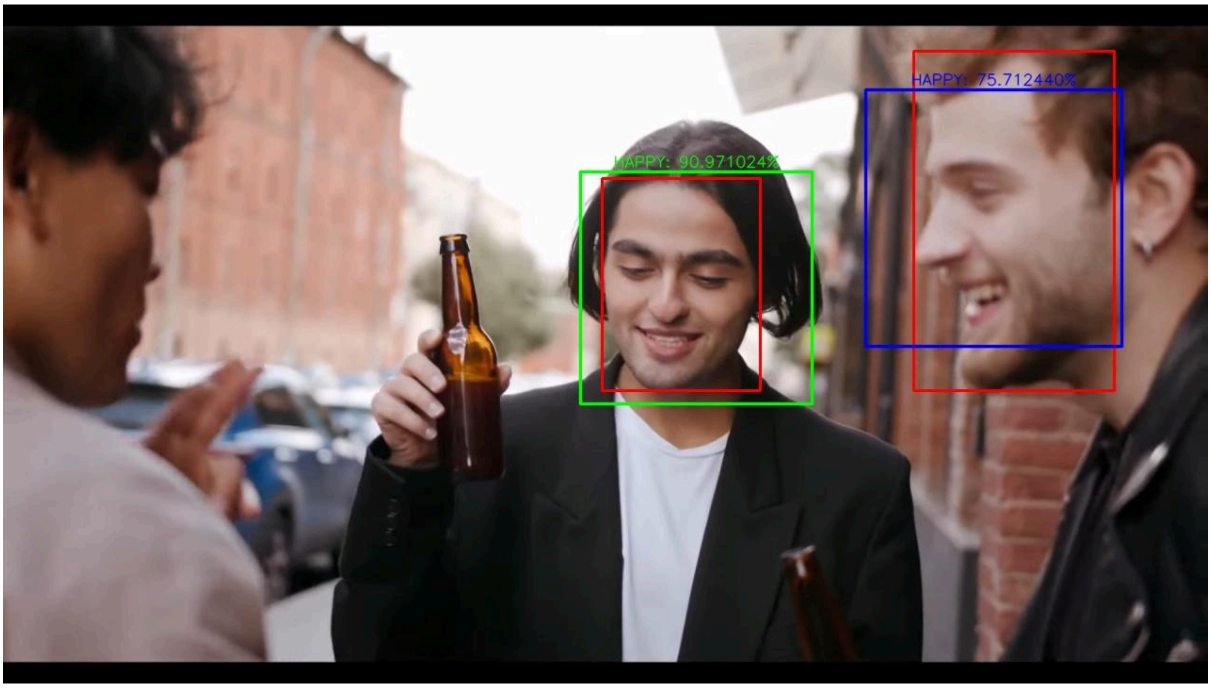
- **“happy (4).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** = 0.7413 **Accuracy** = 0.75



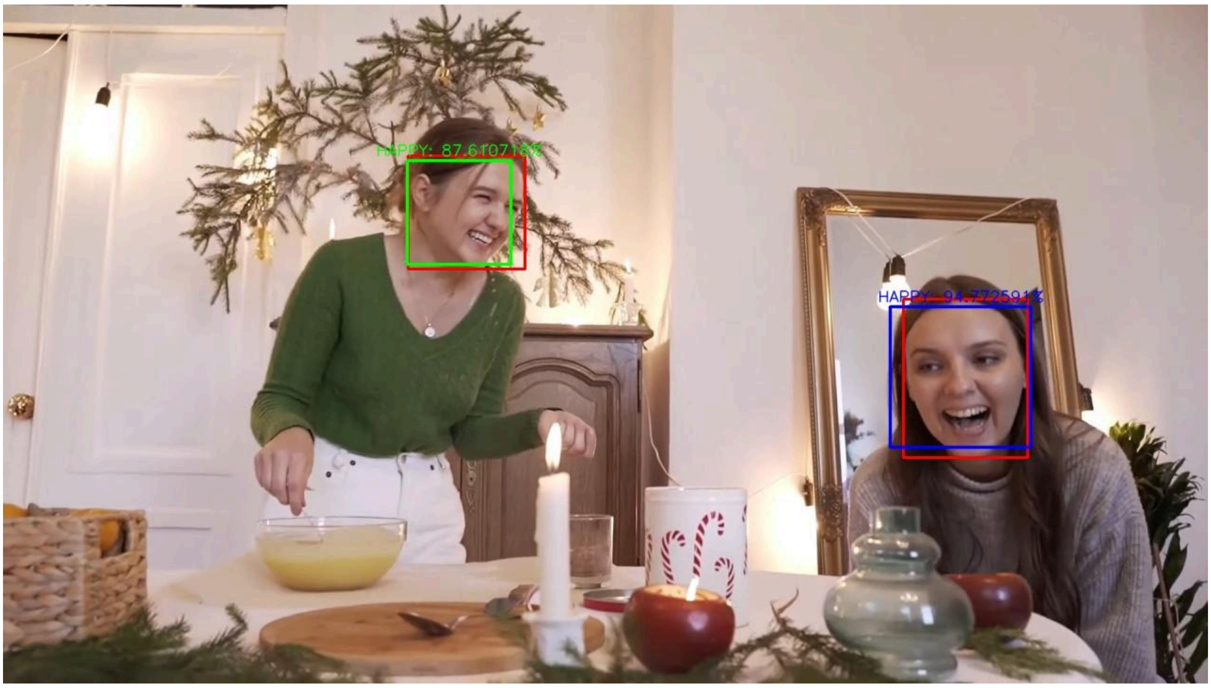
- **“happy (5).jpg” Precision** = 0.5 **Recall** = 0.5 **Mean IoU** = 0.5331 **Accuracy** = 1, left box has IoU < IoU_threshold



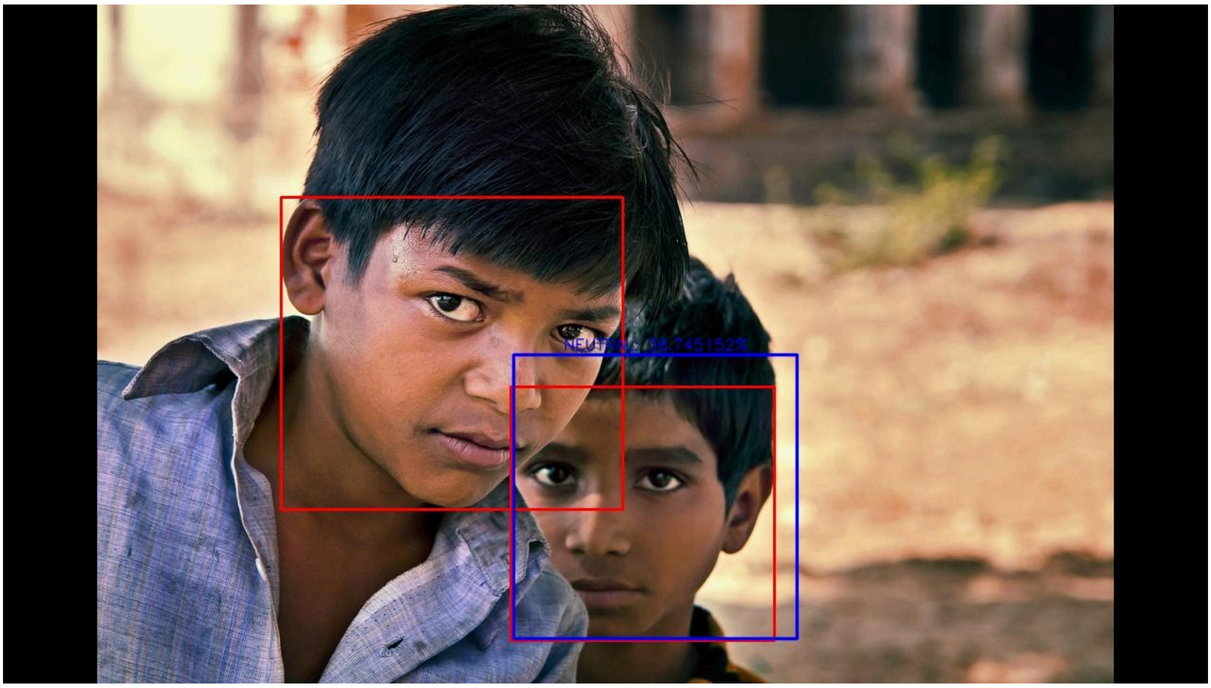
- **“happy (6).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** = 0.6246 **Accuracy** = 1



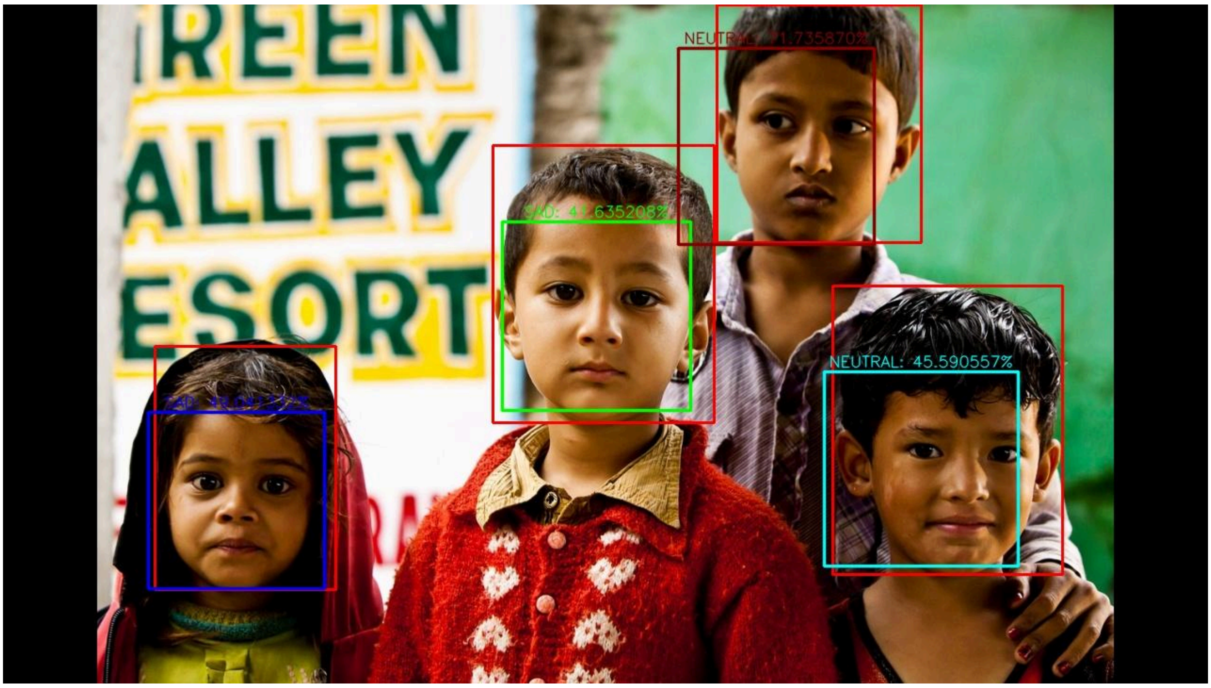
- **"happy (7).jpg" Precision** = 1 **Recall** = 1 **Mean IoU** = 0.7989 **Accuracy** = 1



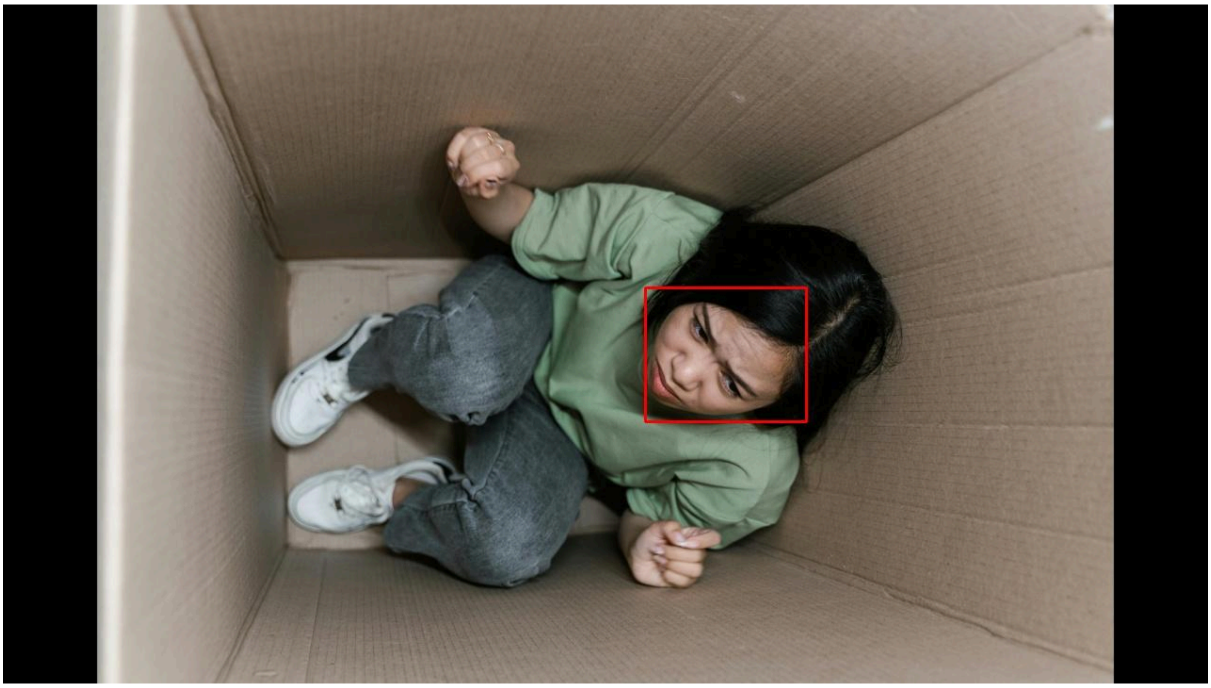
- **"fear (1).jpg" Precision** = 1 **Recall** = 0.5 **Mean IoU** = 0.4517 **Accuracy** = 0



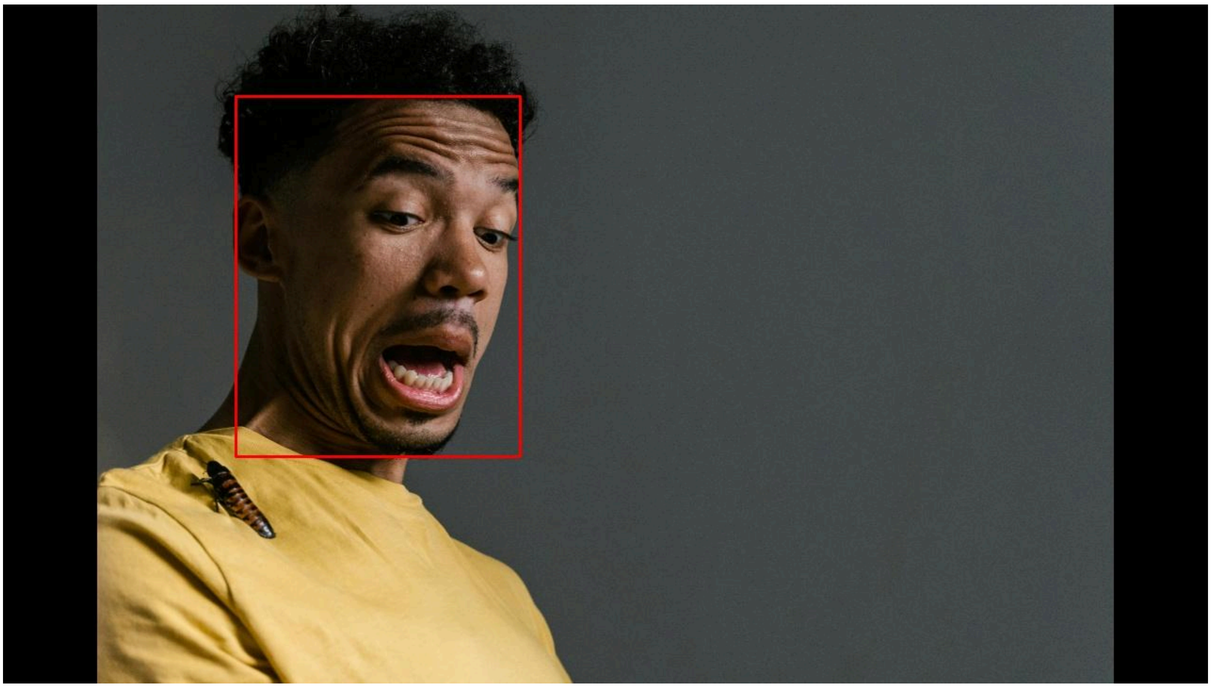
- **“fear (2).jpg” Precision** = 1 **Recall** = 0.5 **Mean IoU** = 0.4720 **Accuracy** = 0



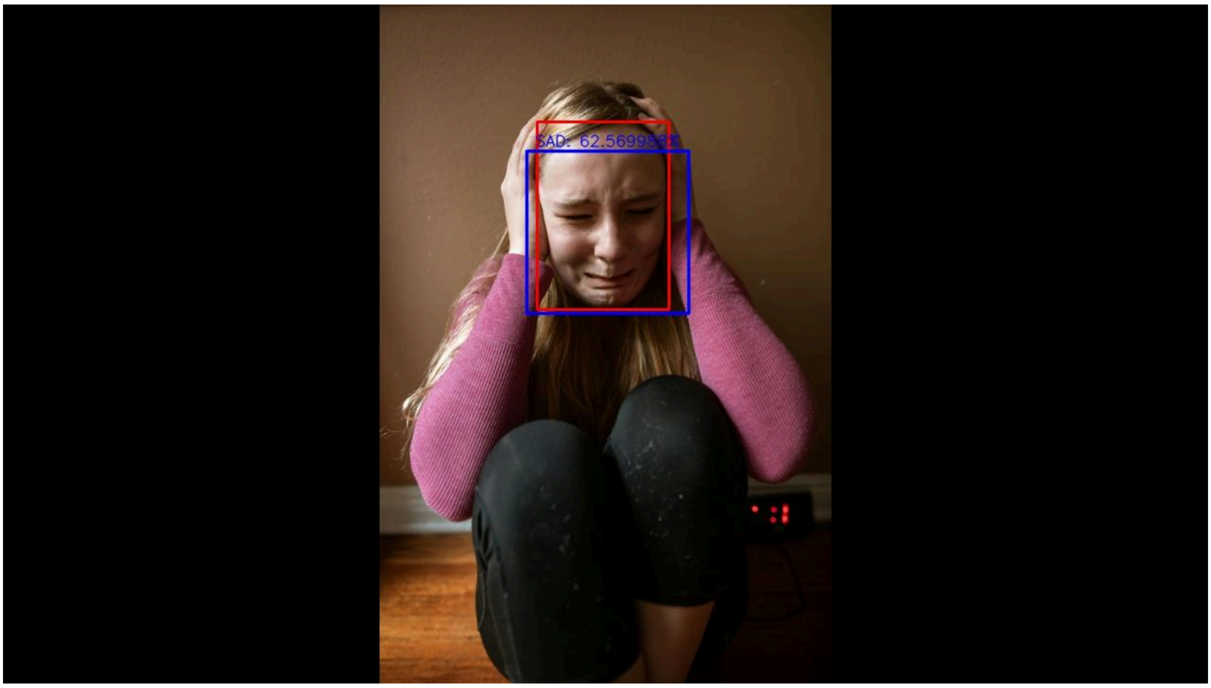
- **“fear (3).jpg”** Face not detected



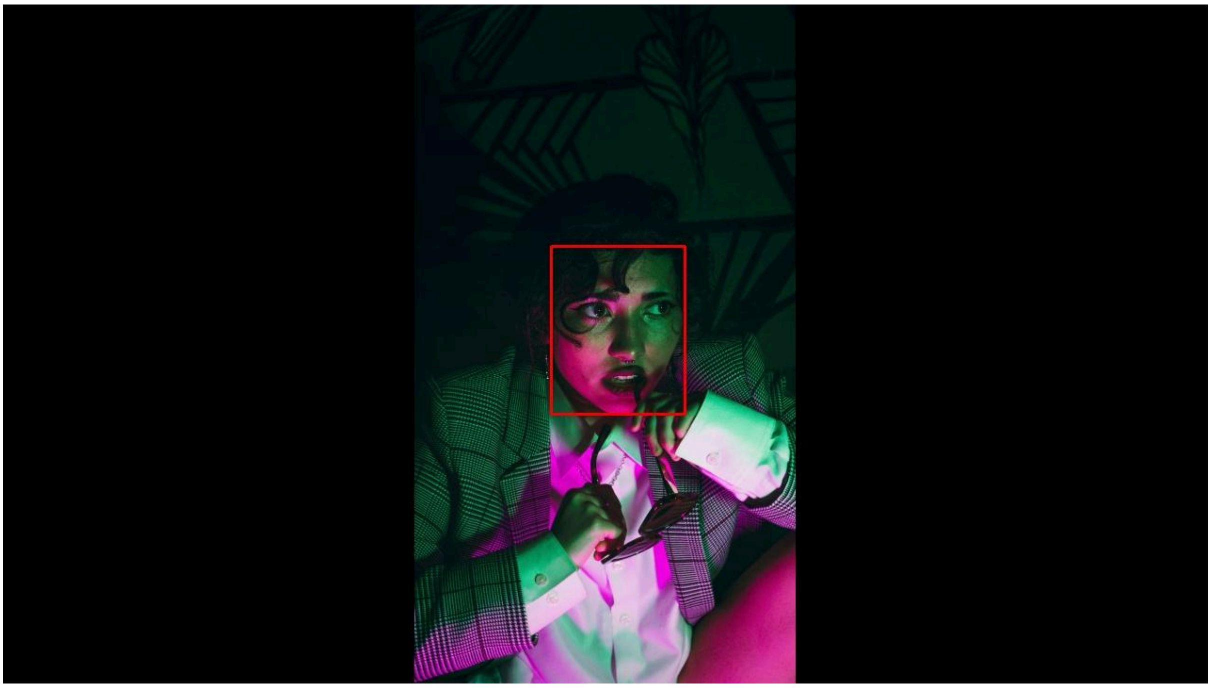
- **“fear (4).jpg”** Face not detected



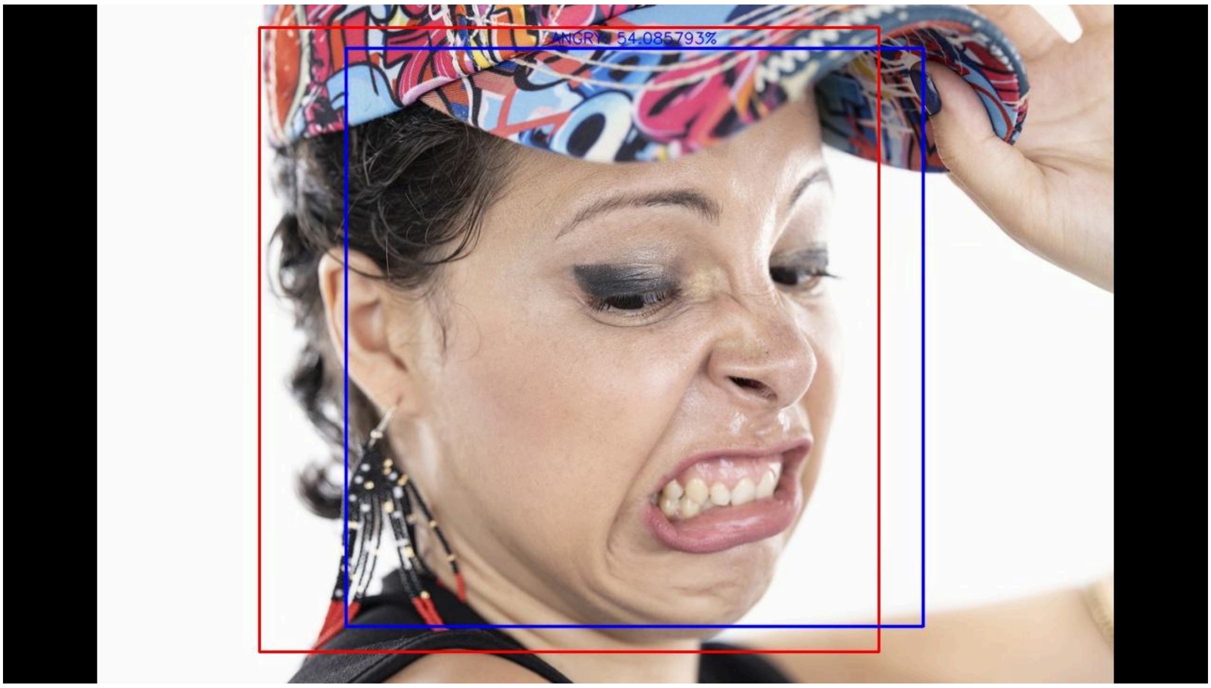
- **“fear (5).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** = 0.6958 **Accuracy** = 0



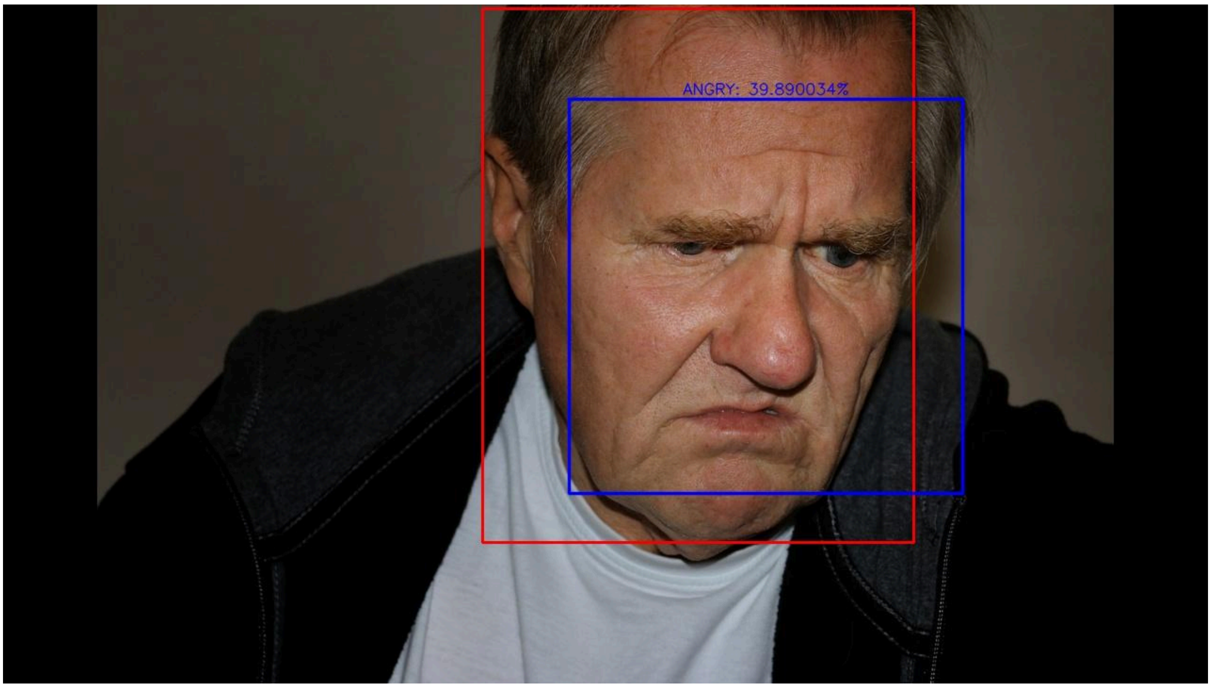
- **“fear (6).jpg”** Face not detected



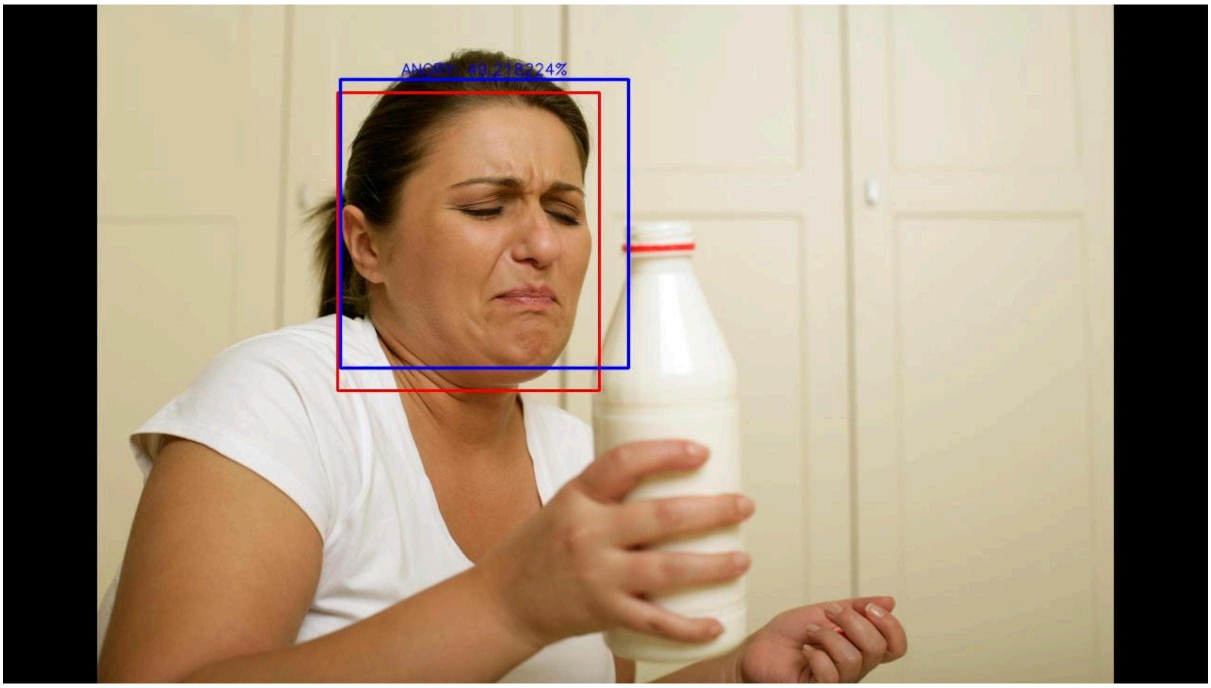
- **“disgust (1).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** = 0.7471 **Accuracy** = 0



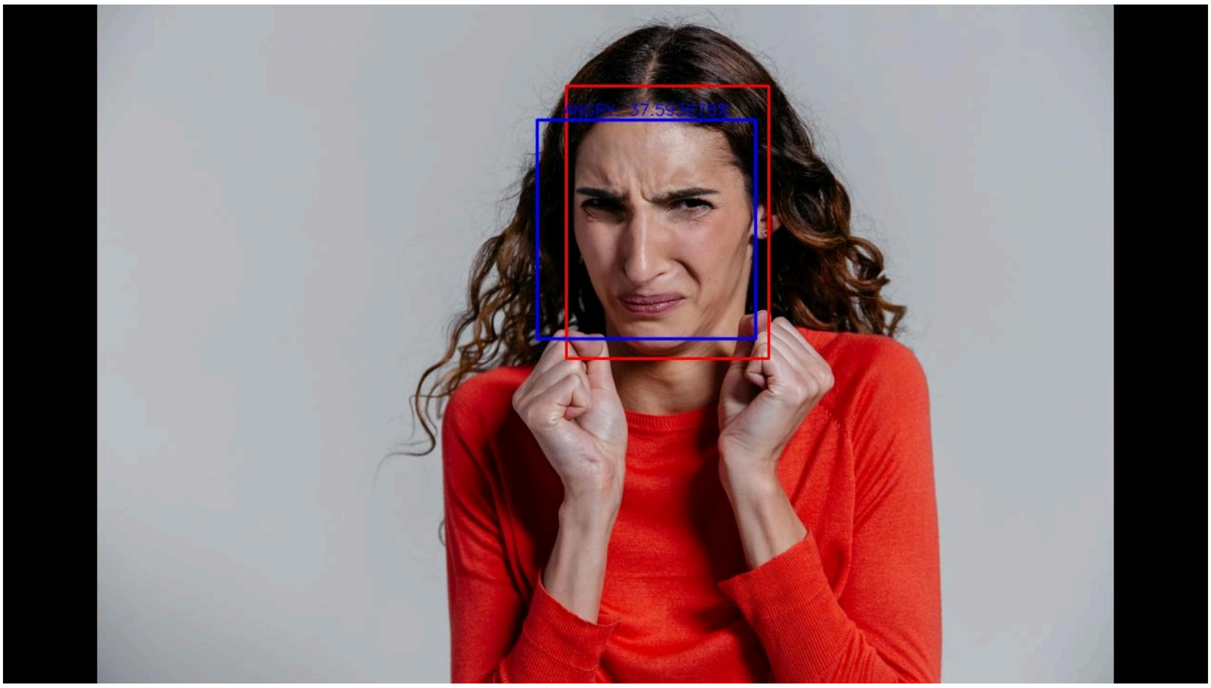
- **“disgust (2).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** = 0.5450 **Accuracy** = 0



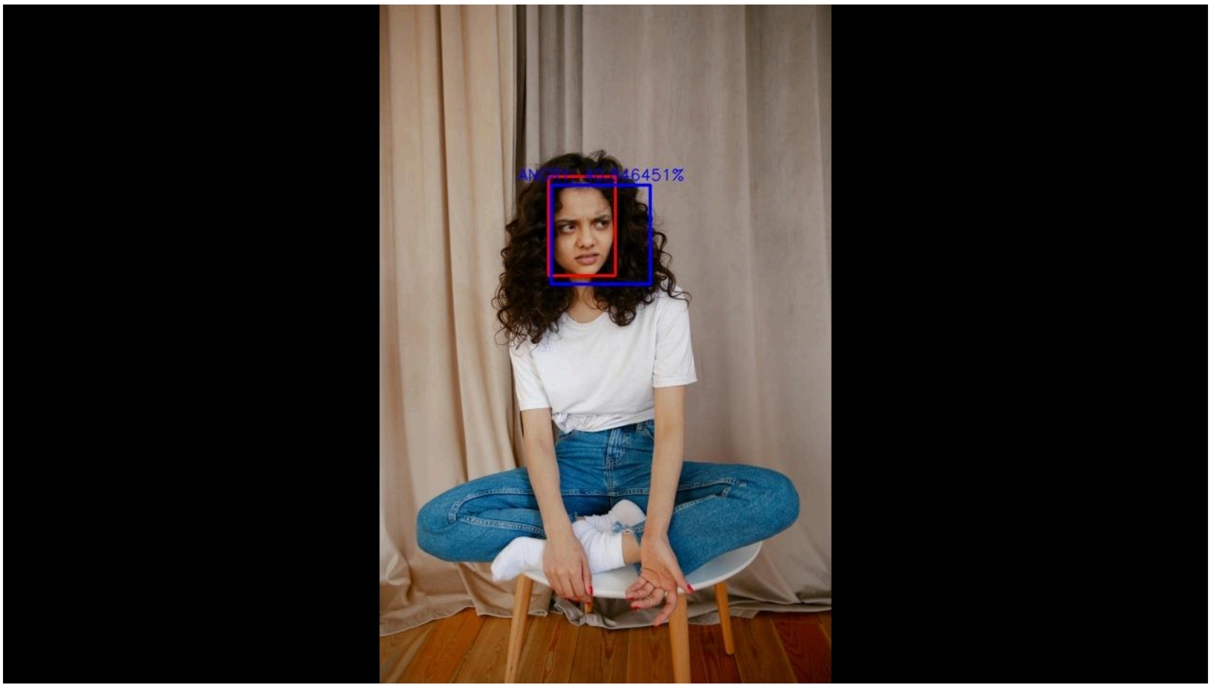
- **“disgust (3).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** = 0.7471 **Accuracy** = 0



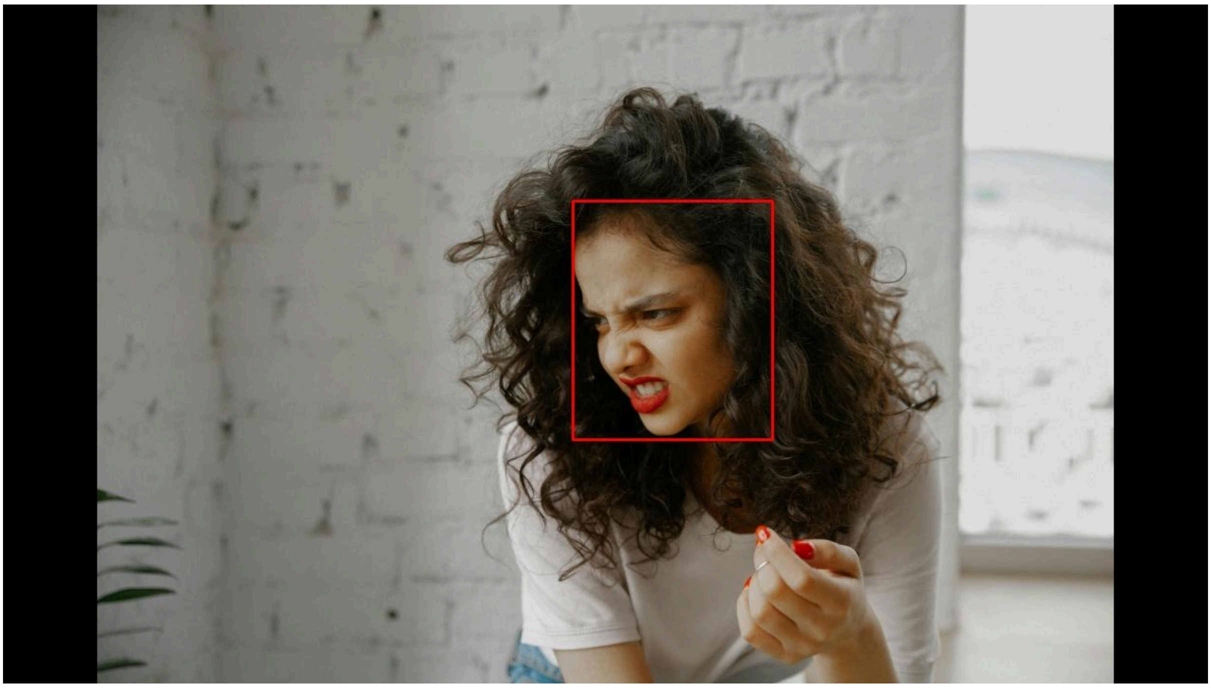
- **"disgust (4).jpg" Precision** = 1 **Recall** = 1 **Mean IoU** = 0.6807 **Accuracy** = 0



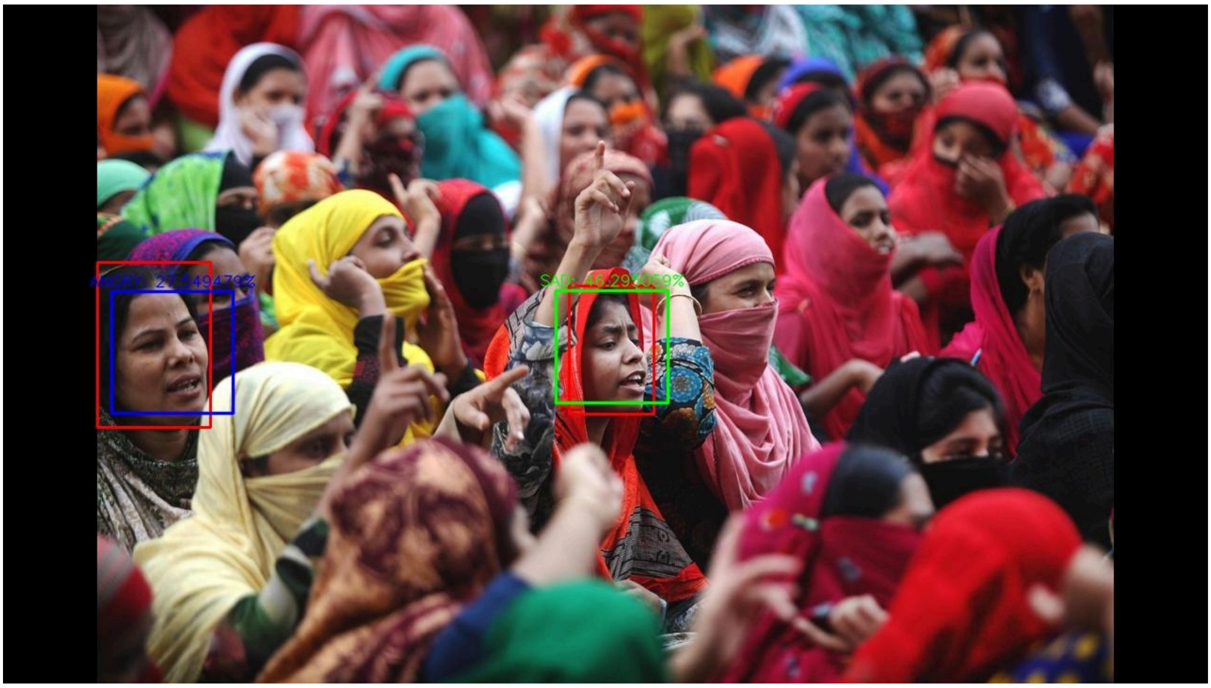
- **"disgust (5).jpg" Precision** = 1 **Recall** = 1 **Mean IoU** = 0.5570 **Accuracy** = 0



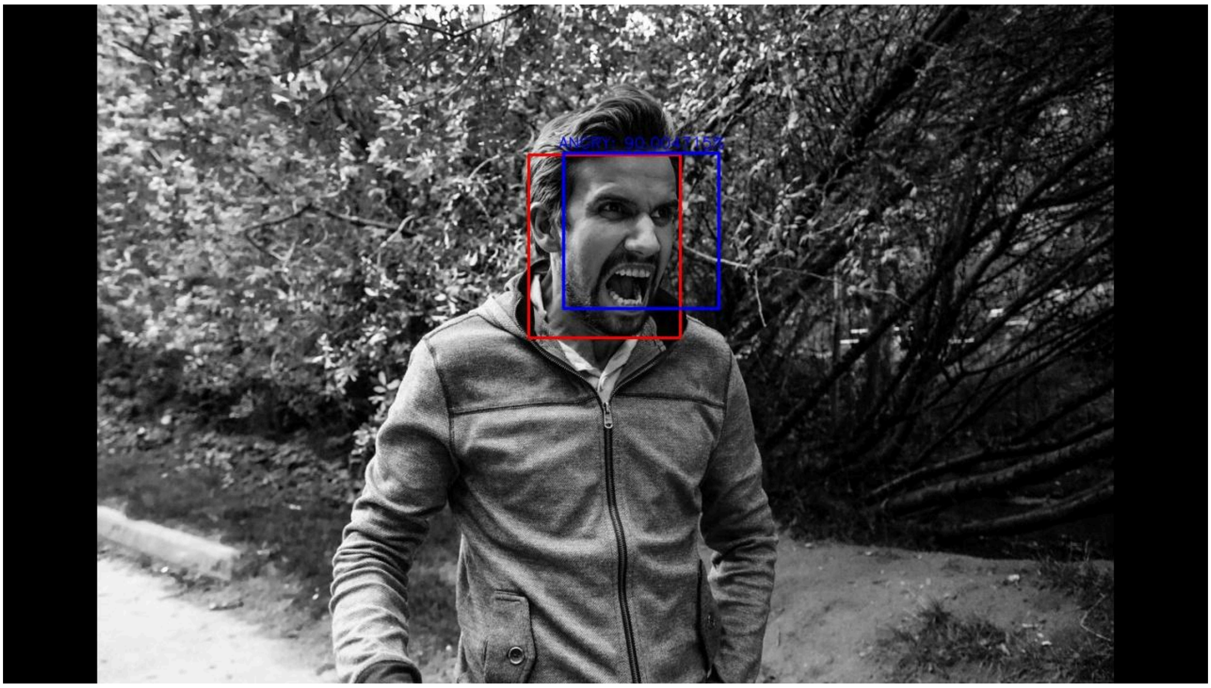
- **“disgust (6).jpg”** Face not detected



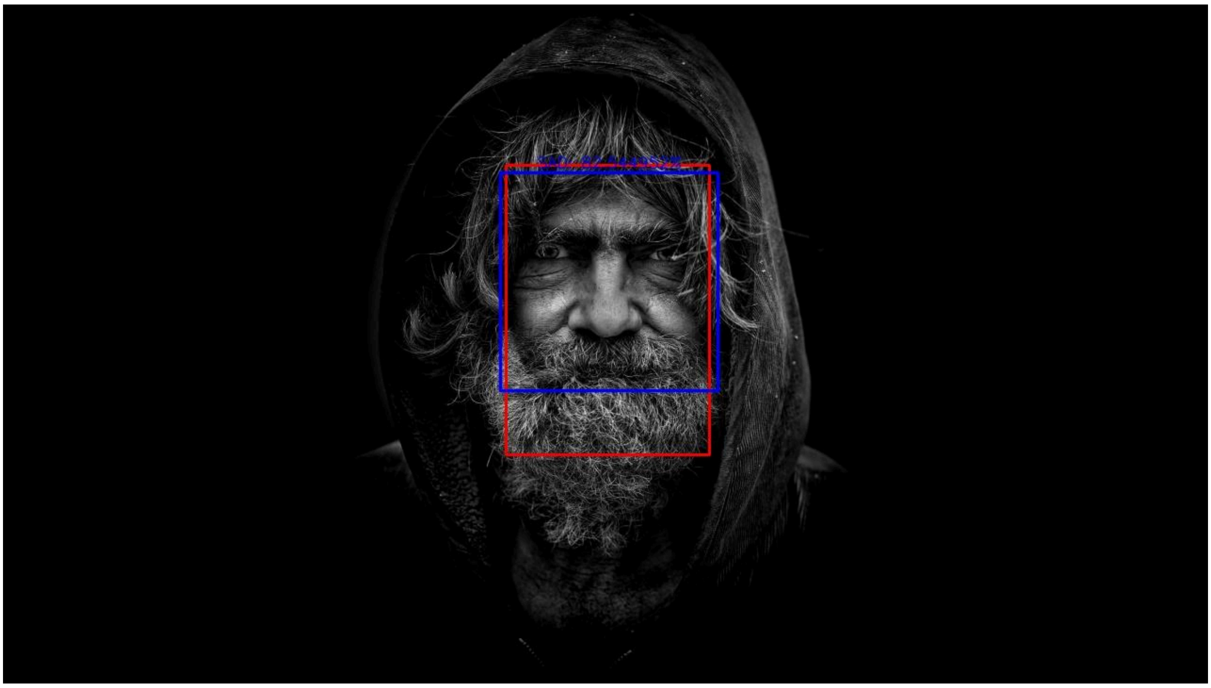
- **“angry (1).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** =0.6236 **Accuracy** = 0.5



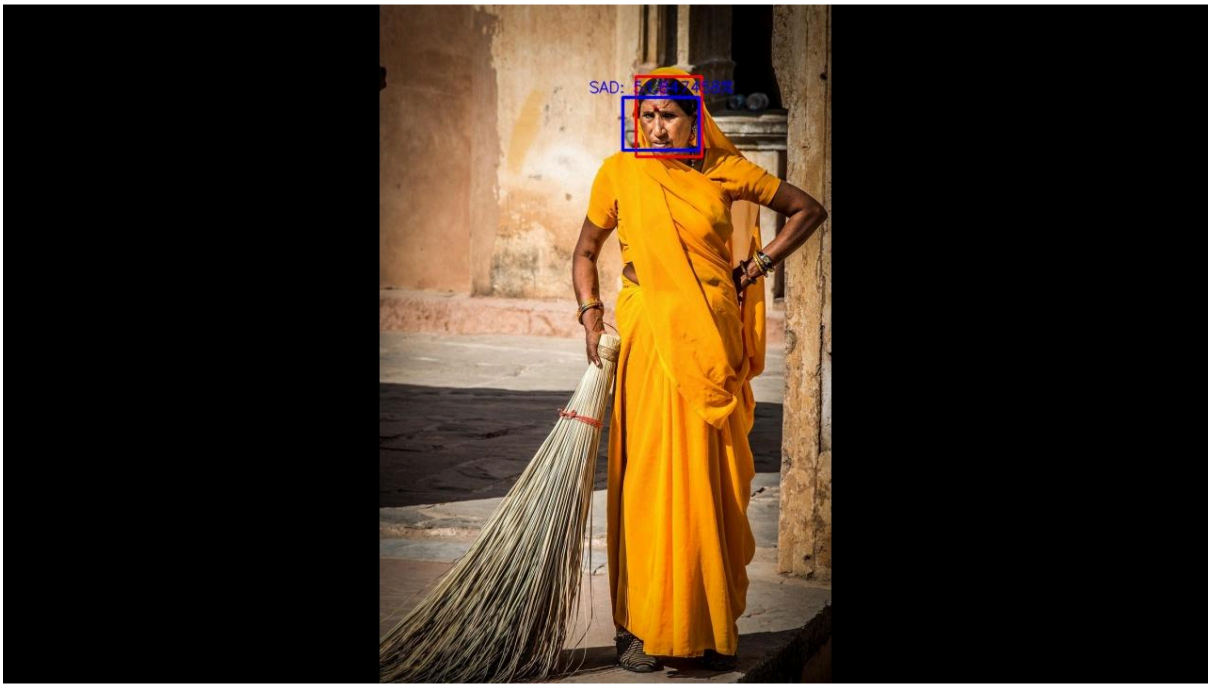
- **“angry (2).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** =0.5370 **Accuracy** = 1



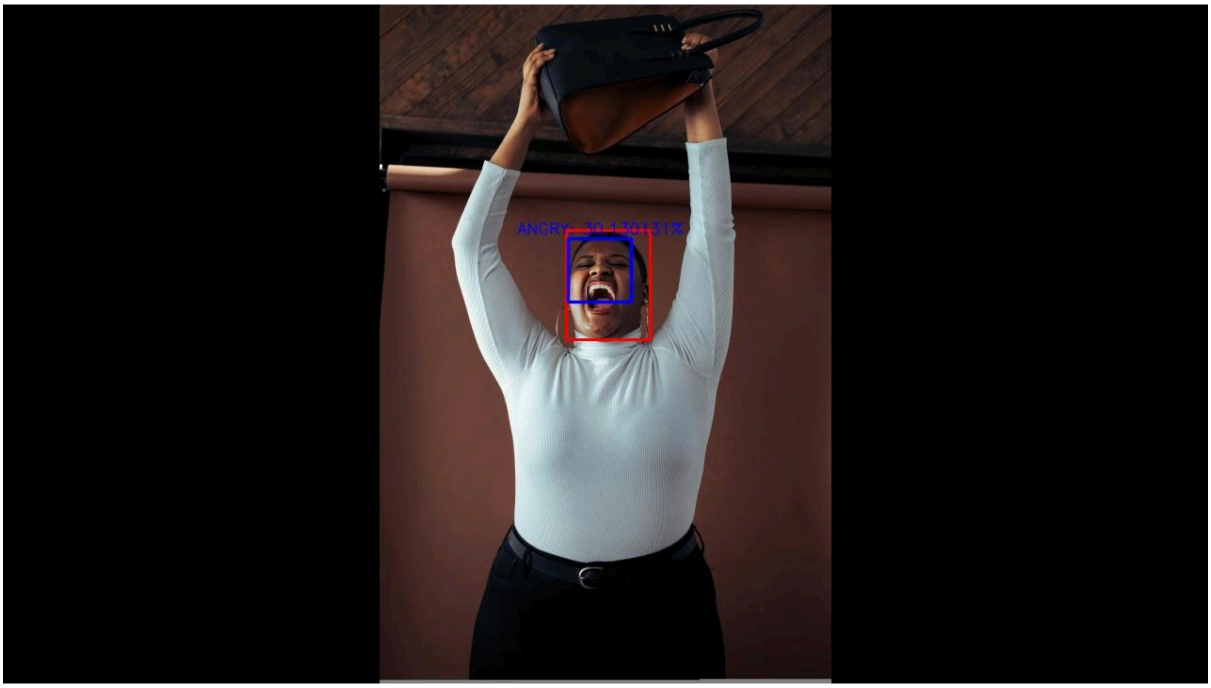
- **“angry (3).jpg” Precision** = 1 **Recall** = 1 **Mean IoU** =0.7144 **Accuracy** = 0



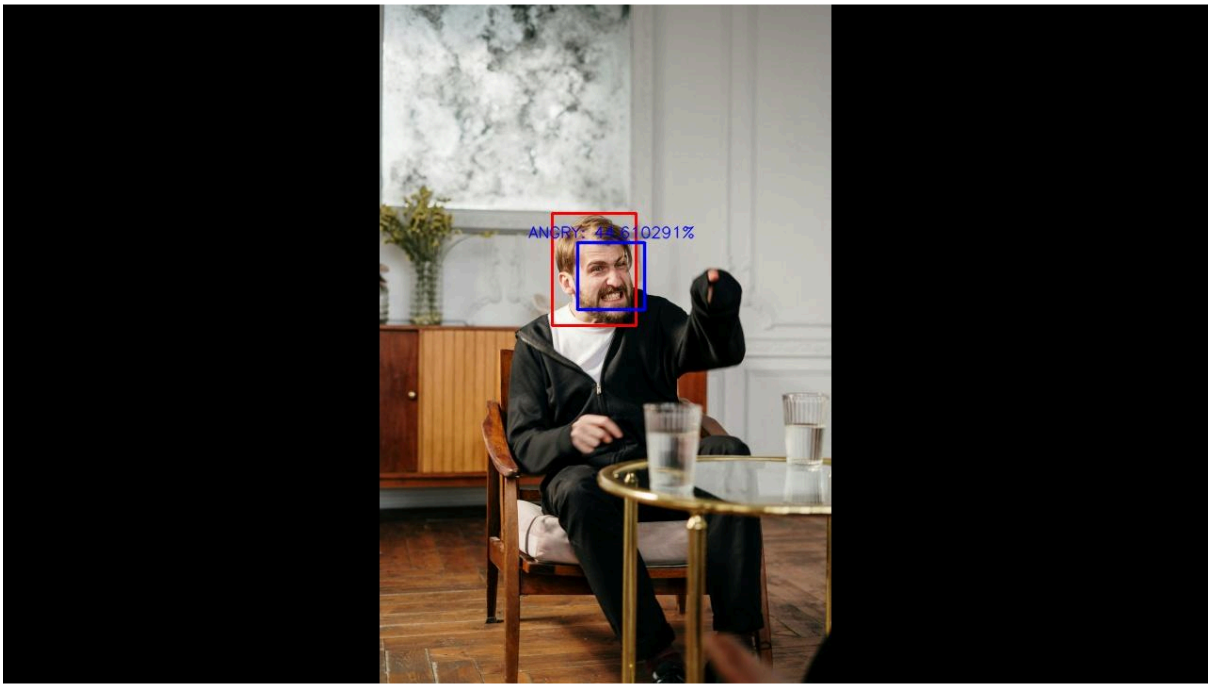
- **"angry (4).jpg" Precision** = 1 **Recall** = 1 **Mean IoU** =0.5568 **Accuracy** = 0



- **"angry (5).jpg"** IoU = 0.4375 < IoU_threshold, face not detected



- **“angry (6).jpg”** IoU = 0.4079 < IoU_threshold, face not detected



Contributes of the group components

❖ **Ballarin Tommaso:** Image selection menu (selection.cpp, selection.h), visualization of bounding boxes for both ground truth and predicted faces, including color coding and predicted labels. Evaluation metrics for Viola-Jones detection, including precision, recall and IoU (metrics.cpp, metrics.h). Accuracy value for emotion recognition both for single image and whole test dataset (evaluation.cpp, evaluation.h). Implementation of the face detection (pipeline.cpp, pipeline.h) with various post-processing filters to reduce false positives and increase true positives such as: removal of bounding boxes with unrealistic sizes, exclusion of boxes with implausible aspect ratios, suppression of nested detections where a smaller box lies within a larger one and NMS to eliminate duplicate detections (detection.cpp, detection.h). **Hours of work ~ 47 hours.**

❖ **Onorati Jacopo:** Image class that allows the instantiation of an object containing the ROIs, the preprocessed ROIs to be used as input to the CNN pre-trained model for emotion recognition, and the final output image (Image.cpp, Image.h). Preprocessing of the image containing the ROIs detected by the Viola-Jones algorithm, necessary to ensure proper functioning of the CNN model (pre-processing.cpp, emotion_recognition.h). Loading of the CNN model, performing inference, and mapping the predicted class ID to the corresponding emotion label (emotion_recognition.cpp, emotion_recognition.h). Furthermore, I contributed to the

general organization of the C++ code and to the evaluation, testing, and execution of different emotion recognition models for the final version of the project. I also converted the model suggested in the proposal to ONNX format.
The script used for conversion and the converted .onnx file can be found at:
ComputerVisionProject - additional resources
Its output results showed an excessive bias toward the "surprise" emotion, so we chose not to use it. **Hours of work** ~ **45 hours**

- **Spicoli Piersilvio:** Creation of the ../models/CNN_model.ipynb notebook for training and evaluating the CNN model. Dataset preparation and normalization, data augmentation, training with saving of the best model, metrics analysis, and final model selection. Development of the main program for the complete management of the execution flow: image loading, selection, pipeline invocation and global accuracy calculation (Main.cpp). Implementation of the configuration file (config.h) containing the paths of the Haar models and classifiers, the name of the viewing window and the scaling settings, so as to centralize the parameters. Creating a function to draw the GT bbox from .txt annotations (draw.cpp/draw.h). Creation of the repo GitHub and brainstorming of the workflow of the project. **Hours of work** ~ **40 hours**