# Final Project - README

## Title - QuadraFuse: Local–Global Feature Fusion for Robust Face Recognition

QuadraFuse investigates a fundamental limitation of current face recognition systems — their over-reliance on global feature representations, which leads to degraded performance under real-world variations such as occlusion, pose shifts, and illumination changes.

This project addresses the research question:

"Can a change in model training — by focusing on local features and fusing them with global features — significantly improve robustness, accuracy, and reliability compared to existing methods?"

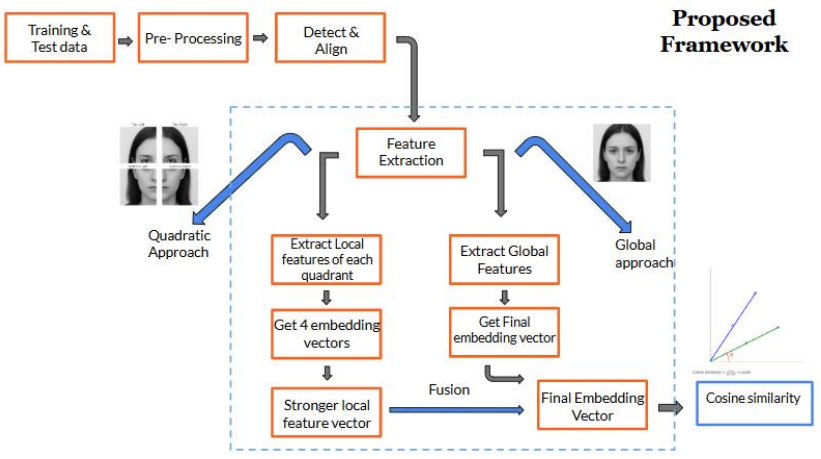
To answer this, QuadraFuse proposes a two-branch deep learning architecture that extracts:

- Global facial features preserving structural and contextual information.
- Local quadrant-level features capturing fine-grained details that are less sensitive to occlusion.

The fused embedding is designed to be more discriminative, occlusion-resistant, and generalizable, enabling stronger performance across masked faces, glasses, pose variations, and lighting shifts.

This repository contains the code, model design, and experiments supporting this research.

## Architecture Overview:




The QuadraFuse framework follows a dual-branch architecture that extracts both local quadrant features and global facial features.

- The face is first detected and aligned, then split into four quadrants for local feature extraction.
- In parallel, a global embedding is generated from the full face.
- Both embeddings are fused to form a stronger, more discriminative representation.

## Trustworthyness : robustness and reliability

In this project, our main focus is to see how reliable and robust our face-recognition system really is when it faces real-world challenges. Instead of testing only on clean, perfect images, we exposed both models to many difficult scenarios—masked faces, random occlusions, lighting changes, rotations, blur, and other common distortions. The traditional CNN struggled in these situations, showing noticeable dips in accuracy and large swings in confidence.

QuadrantFusionNet, however, handled these shifts much better because it learns both global facial patterns and local quadrant-based details. The improved, occlusion-aware version performs even more reliably by simply ignoring the blocked regions and relying on the visible parts of the face. When we analyzed confidence trends, the QuadrantFusion model produced much smoother, more stable confidence levels, showing that it is less sensitive to noise. Even under strong perturbations, its accuracy dropped far less than the baseline. Grad-CAM results also confirmed that it consistently attends to meaningful facial areas, even when parts of the face are hidden.

Altogether, the results show that QuadrantFusionNet behaves in a more steady, predictable, and reliable way when compared to a traditional CNN.

## Datasets:

QuadraFuse/

|

— ……..Dataset1/ # Training dataset (5 classes)

| — tom/

| — dwayne/

| — camila/

| — andy/

| — alexandra/

|

— ……Dataset_test/ # Testing dataset (occulsed faces)

| — tom/

| — dwayne/

| — camila/

| — andy/

| — alexandra/

|

— baseline_face_recognition.pth # Global-only face recognition model (baseline)

— mask_detector_resnet18.pth # Mask detection model

— quadrant_fusion_faces_v3.pth # Proposed QuadraFuse model

**For this phase of the project, we have used a different dataset from the previous dataset. So, the dataset we used have 5 different classes from the previous one, and the name of the dataset called Dataset1, and for testing Dataset_test.**

The training and evaluation of QuadraFuse are performed using **custom curated datasets** and **purpose-built models**:

**Dataset Structure**

- **Dataset1/** – Base dataset with **5 identity classes**:
  tom, dwayne, camila, andy, alexandra
  → Used for training both traditional model and quadrafuse model.

- **Dataset_test/** – Masked dataset with the same 5 identities.
  → Used for testing and evaluating robustness under occlusion (mask scenarios).

**Pretrained & Custom Models**

- **baseline_face_recognition.pth** — Standard CNN-based face recognition model (global-only).

- **mask_detector_resnet18.pth** — Mask detection model trained to identify whether a face is masked or not.

- **quadrant_fusion_faces_v3.pth** — Proposed **QuadraFuse model**, combining local quadrant embeddings with global features for enhanced robustness.

All datasets were preprocessed (face alignment & resizing), and augmentations (lighting variation, rotation, occlusion simulation) were applied to increase generalization.

**Note on Occlusion Handling:**
In this phase of the project, we focused on lower-face occlusion using masks as the primary scenario to evaluate the model's robustness.
The choice of masks was intentional — it allows us to analyze the system's response to structured and realistic occlusion that commonly affects face recognition performance.
Additional occlusions such as spectacles and other accessories will be incorporated under the trustworthiness and robustness evaluation in the final stage of the project to further validate the model's generalization capability.

**FaceQuadrantNet/**

|

— **data(dataset for mask detection model)**

__ **FaceQuadranet**

— **Traditional_facereco_model_tr.ipynb # Training - Traditional (Global Only)**

— **quadra_model_tr.ipynb # Training - QuadraFuse (Local + Global Fusion)**

— **compare_traditional_with_masketstimages.ipynb # Testing - Traditional (Masked dataset)**

— **Test_with_mask.ipynb # Testing - QuadraFuse (Masked dataset)**

— **line_graph.ipynb # Accuracy / Confidence visualization**

— **Mask_tr.ipynb # Mask detection model training**

## Implementation:

To run this repository, first download or clone. Next, upload the files to drive and link the drive to colab. Open the files and change the runtime to the best one. Also, take care of the relative path after you clone the repository, because after you clone, the relative path may differs due to the environment you use to execute. For more details and understanding, look for the structure if the file structure.

To run the API, go inside the trustworthy folder and run the following command("$ python manage.py runserver"). open the browser and hit localhost:8000.

## Model Training:

QuadraFuse includes two main training approaches — a traditional global-only method and our proposed local–global fusion method.

We did 3 trainings:

1. Training traditional global only approach model which generated **baseline_face_recognition.pth** file
2. Training QuadraFuse model, combining local quadrant embeddings with global which generated **quadrant_fusion_faces_v3.pth** file.
3. Training model to perform mask detection which generated **mask_detector_resnet18.pth** file.

### baseline_face_recognition.pth file:

We trained a **baseline face recognition model** using **ResNet-18** as the backbone.
The model was trained on the unmasked dataset with:

- **Batch size:** 32
- **Epochs:** 15
- **Learning rate:** 1e-4 (Adam optimizer with StepLR scheduler)
- **Transforms:** resize, horizontal flip, random rotation, color jitter, normalization
- **Validation split:** 15%

During training, we logged train/validation accuracy and loss for each epoch.
The model with the best validation accuracy was saved as baseline_face_recognition.pth.
Finally, evaluation was done using accuracy, classification report, and a confusion matrix.

**quadrant_fusion_faces_v3.pth file**:

This module implements the **proposed QuadraFuse model**, which enhances face recognition robustness under occlusions by fusing **global** and **quadrant-level local features** through an **attention-based fusion** mechanism.

**Key Steps & Components:**

- **Dataset Handling**
  - Organized into class-wise folders (5 identities in this project).
  - Optional INFER_FROM_FILENAME mode to auto-infer labels from filenames if no folders exist.
  - Uses MTCNN for face alignment and preprocessing.
  - On-the-fly augmentation (resize, flip, color jitter) for better generalization.
- **Quadrant Strategy**
  - Each aligned face is split into 4 quadrants (TL, TR, BL, BR).
  - Global features are extracted from the entire image.
  - Local features are extracted from each quadrant separately.
- **Model Pipeline**
  - ResNet-18 backbone (pretrained on ImageNet) for both global and local embeddings.
  - A lightweight attention module assigns **dynamic weights** to each quadrant.
  - Final fused embedding = Global + Weighted Local.
  - Classification head trained using Cross-Entropy Loss.
- **Training Configuration**
  - **Batch size:** 32
  - **Epochs:** 15
  - **Learning rate:** 1e-4 (Adam optimizer)
  - **Validation split:** 15%
  - **Alignment:** MTCNN face crop enabled
  - **Device:** GPU (if available)
- **Inference Utilities**
  - predict_class(img_path) → predicts class and quadrant attention weights.
  - face_embedding(img_path) → extracts normalized embeddings for downstream tasks.

- Dynamic attention visualization makes the model interpretable.

*This model forms the backbone of our work — enabling face recognition that can intelligently **focus on unoccluded regions** (e.g., upper face during mask scenarios) and maintain stable performance.*

### mask_detector_resnet18.pth file:

We trained a **binary classification model** using **ResNet-18** to detect whether a face is **masked or unmasked**.

- **Backbone:** ResNet-18 (ImageNet pretrained) with the final FC layer modified for 2 output classes.
- **Dataset:** ImageFolder structured with two classes — *masked* and *unmasked*.
- **Transforms:** resize, flip, rotation, color jitter, normalization.
- **Training:**
  - Batch size = 64
  - Epochs = 8
  - Learning rate = 3e-4 (Adam optimizer + StepLR scheduler)
  - 15% validation split.
- **Checkpointing:** The best model (highest validation accuracy) is saved as mask_detector_resnet18.pth.
- **Evaluation:** Final validation accuracy, classification report, and confusion matrix are generated to measure performance.

This model is later used in the pipeline to **identify occlusion (masks)** and evaluate **QuadraFuse robustness** under masked scenarios.

## Model Testing:

### Testing with traditional method on occulsed face: (compare_traditional_with_masketstimages.ipynb)

In this stage, we evaluate the baseline global-only ResNet-18 face recognition model on the masked dataset (lower-face occlusion).
The goal is to measure how traditional approaches perform under real-world occlusion without any adaptive mechanisms.

Key Steps:

- Load the trained baseline_face_recognition.pth model.
- Test on the masked face dataset with five identity classes.

- Record predictions for each image:
    - Extract global embeddings from the full image.
    - Predict the class and compare with ground truth.
- Track progressive accuracy trend across test samples to visualize model degradation over time.

Metrics & Visualizations:

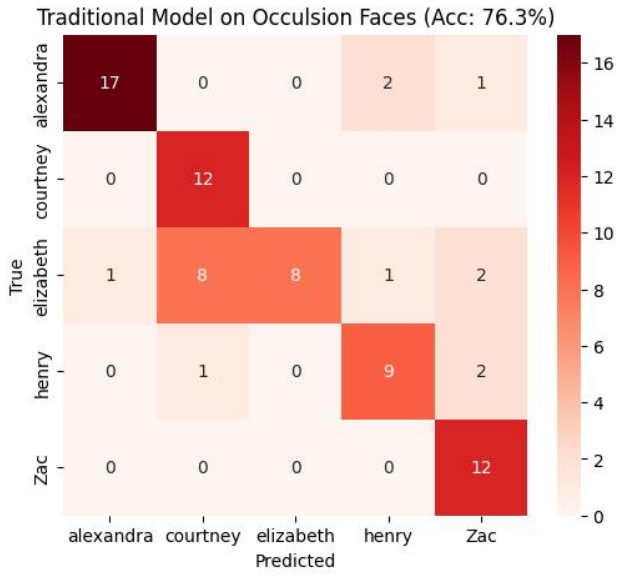
- Final accuracy, confusion matrix, and classification report.
- Accuracy trend curve — progressive accuracy over test samples.
- Prediction correctness graph — shows correct (1) vs. incorrect (0) classification per image.
- Confidence trend graph — shows model confidence and rolling average trend line.

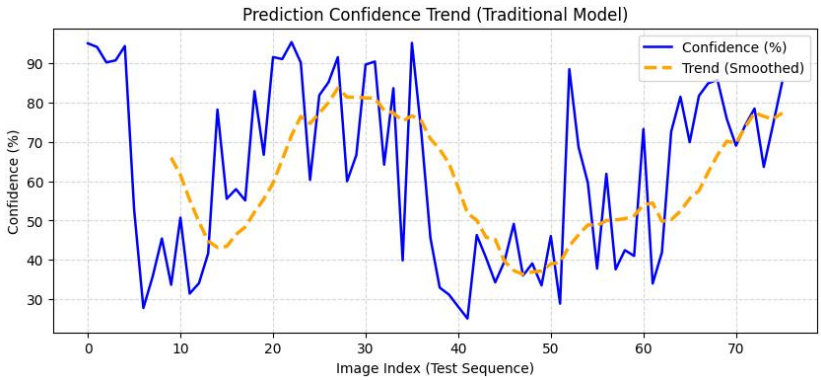
*This experiment establishes the baseline performance of the traditional global-only model under lower-face occlusion, which is then compared against the QuadraFuse approach.*

## Outputs:

Final Accuracy: 76.32%

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| alexandra | 0.944 | 0.850 | 0.895 | 20 |
| courtney | 0.571 | 1.000 | 0.727 | 12 |
| elizabeth | 1.000 | 0.400 | 0.571 | 20 |
| henry | 0.750 | 0.750 | 0.750 | 12 |
| Zac | 0.706 | 1.000 | 0.828 | 12 |
| accuracy | | | 0.763 | 76 |
| macro avg | 0.794 | 0.800 | 0.754 | 76 |
| weighted avg | 0.832 | 0.763 | 0.750 | 76 |






you can see that in the top plot (the traditional CNN), the average prediction confidence is around 61–62%. More importantly, the confidence curve has sharp rises and sudden drops, meaning the model is unstable whenever the face changes or is partially occluded.

## Testing with QuadraFuse method on occulsed face:

(Test_with_mask.ipynb)

This testing pipeline evaluates the proposed local–global fusion model (QuadraFuse) on the masked face dataset, integrating a mask detector to dynamically adjust feature extraction.
Unlike the traditional model, this approach intelligently focuses on visible quadrants when masks are present. For this phase of the project, we trained a model called “” (advanced qquadrafuse with mask aware mechanism)

Key Steps:

- Load mask_detector_resnet18.pth to detect mask presence in each test image.
- Load quadrant_fusion_faces_v3 — the trained QuadraFuse model.
- For each image:
    - Detect whether the face is masked.
    - If masked → use only the top quadrants (TL, TR), ignoring occluded lower half.
    - Fuse local + global embeddings to predict the identity.
- Collect predictions and compute evaluation metrics.

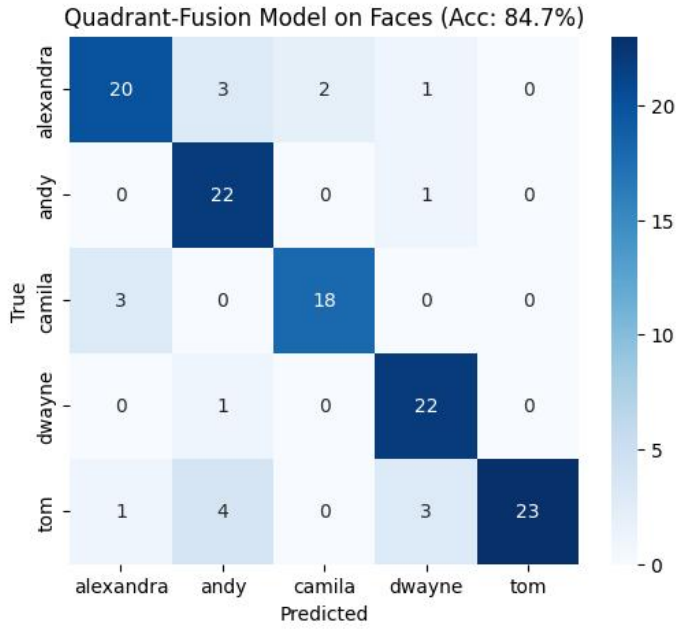
Metrics & Visualizations:

- Final accuracy, classification report, and confusion matrix.
- Prediction correctness trend (per image).
- Prediction confidence trend with rolling average smoothing.
- Dynamic attention weight visualization per quadrant when masks are detected.

This mask-aware testing demonstrates how QuadraFuse maintains stable recognition accuracy even under lower-face occlusion, by leveraging local quadrant attention.

## Outputs:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| alexandra | 0.833 | 0.769 | 0.800 | 26 |
| andy | 0.733 | 0.957 | 0.830 | 23 |
| camila | 0.900 | 0.857 | 0.878 | 21 |
| dwayne | 0.815 | 0.957 | 0.880 | 23 |
| tom | 1.000 | 0.742 | 0.852 | 31 |
| accuracy | | | 0.847 | 124 |
| macro avg | 0.856 | 0.856 | 0.848 | 124 |
| weighted avg | 0.864 | 0.847 | 0.847 | 124 |




### Advanced Quadrafuse model with mask aware mechanism:

This architecture follows the same flow of the above mechanism but, during the testing, instead of system considering the covered parts, this mechanism identifies the covered part and avoid it during the prediction process. For this to make it happen, we have trained a mask detector and used it in testing for detecting mask in an image to ignore the specific covered part. This mechanism help model to remove misleading parts like mask for prediction of face. Along with this,by taking out this misleading and unnecessary information, model only need to focus on specific parts of the face instead of entire face.

Timing information : Traditional CNN: ~0.0082 seconds per prediction

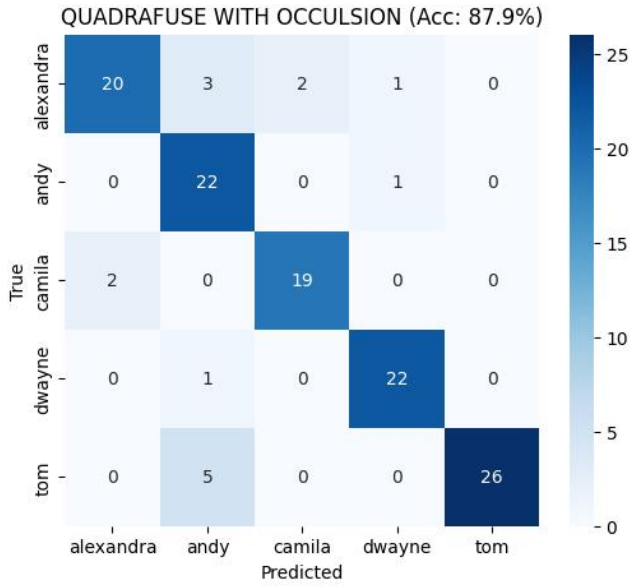
Quadrant-Fusion: ~0.0214 seconds per prediction

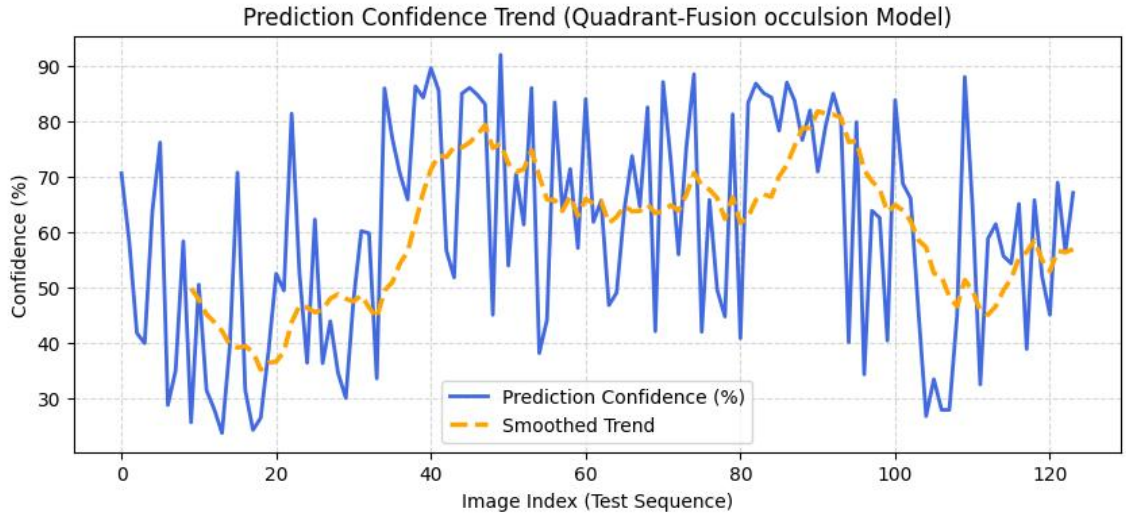
Improved Occlusion-Aware Quadrant-Fusion: ~0.0259 seconds per prediction

Results :

Accuracy: 87.90%

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| alexandra | 0.909 | 0.769 | 0.833 | 26 |
| andy | 0.710 | 0.957 | 0.815 | 23 |
| camila | 0.905 | 0.905 | 0.905 | 21 |
| dwayne | 0.917 | 0.957 | 0.936 | 23 |
| tom | 1.000 | 0.839 | 0.912 | 31 |
| accuracy | | | 0.879 | 124 |
| macro avg | 0.888 | 0.885 | 0.880 | 124 |
| weighted avg | 0.896 | 0.879 | 0.881 | 124 |




Even though the confidence values still fluctuate—but they does not took larger interval as like baseline cnn. The predictions stay closer to the smoothed orange trend line, without large intervals or sudden dips. This tells us that the Quadrant-Fusion model maintains much more stable and consistent confidence compared to the baseline CNN.

## Final Results and Conclusion:

The core objective of this project was to address the accuracy drop in face recognition systems when exposed to real-world occlusions, such as masks, by enhancing the way models learn and use local + global facial features.

| Model | Accuracy on Masked Faces | Avg.Confidence | Observations |
|---|---|---|---|
| Traditional (Global Only) | 76.3% | 61.89% | Even though its not trained on masked faces, the predictions are good because of CNN excellence. |
| QuadraFuse (Mask-Aware) | 87.9% | 64.94% | This model gets way more good acuracy compared to the baseline CNN. Along with this, the confidence graph shows smooth predictions, rather than fluctuating like baseline CNN |

- The traditional model performed well on unmasked faces but collapsed to 76.3 % accuracy on masked faces.
- In contrast, the QuadraFuse model, equipped with a mask detector and quadrant-level fusion, achieved 87.9 % accuracy, maintaining strong confidence and stable trends throughout testing.
- Confidence trends showed that traditional models remained overconfident in wrong predictions, while QuadraFuse displayed more consistent and interpretable attention behavior.
- Attention visualization confirmed the model's ability to ignore occluded quadrants and rely on the upper face region.

### Why This Matters:

- Real-world face recognition (security, authentication, access control, etc.) often faces challenges due to partial occlusion (e.g., masks, sunglasses, scarves).
- Our Quadrant-based fusion provides a lightweight and effective solution without the need for additional complex architectures.
- The approach is modular — it can be combined with existing CNN/Transformer backbones to improve robustness.

### Final Verdict:

- This project successfully demonstrates that occlusion-aware quadrant fusion can dramatically improve masked face recognition performance.
  The experiment shows the improvement of the accuracy compared to baseline CNN(76%), Advanced

quadrafuse(87%). Along with it,for advanced quadrafuse - the confidence is stable for the predictions with the ground truths but, whereas te baseline fluctuates a lot with large intervals which shown the non-stability of the algorithm.

## Contributions:

**Vishnu Swarup Pujari** : Developed the new proposed framework model for face recognition, and compared the models for analysis. Worked on GitHub repository. Worked on implementing the robustness of the model.

**Tejaswini Varampati :** Developed the traditional model for architecture comparison with the proposed model, and written the README file. Gathered the dataset and other source from internet. Worked on implementing the robustness of the model.

**Supradeep Chitumalla** : Developed the Django server for api layer to showcase the project demonstration, generated the masked images for testing. Along with this, mask detection model for testing is done. Worked on GitHub repository. Worked on implementing the robustness of the model.

**Disclaimer:**

*** This is a novel idea which includes architectural change, and does not include codes or other references from any source of internet except datasets (masked dataset).***