**Face Detection & Image Comparison Web App**

A Streamlit-based tool for identity verification using image similarity and face detection

Presenter: Navneet 2023A7PS0483G
GitHub: NAVNEET2311

**Project Overview**

This project is a Python Streamlit web app for:

- Comparing two images using:
  - SSIM (Structural Similarity Index)
  - MSE (Mean Squared Error)
- Detecting faces using:
  - Haar Cascades from OpenCV

**Image Comparison**

SSIM (Structural Similarity Index)

- What it is: A perceptual metric that compares two images based on structure, brightness, and contrast.
- How it works: Computes a score between -1 and 1; closer to 1 means the images are nearly identical.
- Why it's useful: More human-like perception of similarity than raw pixel difference.

MSE (Mean Squared Error)

- What it is: Measures the average squared difference in pixel values.
- How it works: Lower MSE = more similar; zero means identical images.
- Why it's useful: A straightforward numerical error metric.

**Face Detection**

Haar Cascades (OpenCV)

- What it is: A machine learning–based face detection technique.
- How it works: Scans the image for patterns that match faces using pre-trained classifiers.
- What it does: Draws bounding boxes around faces and crops them for further processing.

**Web UI (Streamlit)**

Upload Images

- Users can upload two images for comparison directly through a simple UI.

Detect Faces & Annotate

- The app uses Haar Cascades to detect faces in each image.
- Annotated images (with bounding boxes) are shown to the user.

View Metrics

- After face detection, SSIM and MSE are calculated between cropped face regions.
- Both numerical results and images are displayed for interpretation.

**Technology Stack**

- Language: Python
- Libraries Used:
  - Streamlit – Web UI
  - OpenCV – Image processing & face detection
  - scikit-image – SSIM, MSE image metrics

**File Structure & Script Roles**

scripts/

1. streamlit_app.py

- Main application entry point
- Sets up the Streamlit web interface:
  - Upload widgets
  - Buttons for face detection and image comparison
  - Output display (images + SSIM/MSE scores)
- Connects all other scripts together for user interaction.

2. compare_images.py

- Handles image comparison logic
- Implements SSIM and MSE using skimage.metrics
- Accepts two images and returns:
  - SSIM score
  - MSE value

3. detect_objects.py

- Handles face detection using Haar Cascades
- Takes an image and:
  - Detects face coordinates
  - Draws bounding boxes
  - Returns the cropped face image and the annotated version

4. app.py
- Minimal or alternate entry script
- May be used for testing or standalone command-line runs
- Can initialize and test components without Streamlit UI

5. check-libs.py
- Dependency checker
- Confirms that required libraries (OpenCV, Streamlit, etc.) are installed
- Prints out library status (can be used before deploying)

**Installation & Setup**

1.Clone the Repository

git clone https://github.com/NAVNEET2311/Face_detection_PS1.git

2. Install Required Libraries

pip install opencv-python streamlit scikit-image

3.Run the Application

streamlit run scripts/streamlit_app.py

**Usage Flow**

1. Launch the app in your browser via Streamlit.
2. Upload two images (e.g., a selfie and an ID photo).
3. Click on "Detect Faces" to process the images.
4. View:
    - Annotated face images
    - SSIM & MSE metrics for similarity comparison

**Potential Improvements**

- Integrate webcam capture support
- Use DNN-based or dlib face detectors for better accuracy
- Improve UI design with themes and responsive layout
- Add batch comparison and image comparison history logging