Computer Vision: Realtime Face Blurring System

Project Overview

This manual outlines the implementation of a real-time face blurring system designed to enhance privacy in live video streams. The system anonymizes faces while preserving the video's visual context, making it a valuable tool for privacy protection in various applications.

Group Members

- BSEF21M009 (Abdullah Malik)
- BSEF21M016 (Faiqa Nasir)

Technical Approach

1. Face Detection

The system detects faces in each frame of the live video feed using a fine-tuned YOLO (You Only Look Once) model. The model was trained on the Face Detection Dataset available at: https://www.kaggle.com/datasets/fareselmenshawii/face-detection-dataset/data

2. Face Extraction and Blurring

Once faces are detected, the system applies blurring techniques, such as Gaussian blur, to anonymize the face regions effectively. Convex hull algorithms are utilized to ensure accurate face masking.

3. Real-Time Processing

The face detection and blurring system is integrated into a real-time video processing pipeline. The blurred face regions are seamlessly merged back into the video frames to produce a continuous and privacy-protected video stream.

User Interface

A professional, simple interface has been developed using Flask for the web application. Users can:

1. Upload a video file for face blurring
2. Use a live camera feed for real-time face blurring

In both cases, the system processes the input and provides an option to save the output video with blurred faces.

Applications

The real-time face blurring system has numerous applications in privacy protection and beyond:

1. **Public Space Surveillance:** Protect individuals' privacy in CCTV footage of public areas.
2. **News Media:** Anonymize bystanders or sensitive subjects in news broadcasts.
3. **Social Media Platforms:** Allow users to automatically blur faces in shared videos.
4. **Medical Imaging:** Protect patient identity in educational or research videos.
5. **Corporate Security:** Maintain confidentiality in footage from secure areas or meetings.

Technical Implementation

- Face Detection Model: Fine-tuned YOLO model
- Training Dataset: Face Detection Dataset
- Web Application Framework: Flask
- Video Processing: OpenCV

Usage Instructions

1. Access the web interface through the provided URL.
2. Choose between uploading a video file or using live camera feed.
3. If uploading a video, select the file from your device.
4. For live camera feed, ensure your device's camera is accessible.
5. The system will process the input in real-time, displaying the output with blurred faces.
6. Once processing is complete, you can save the output video with anonymized faces.

Installation and Running the Application

1. Extract the contents of the attached .zip file.
2. Ensure all required libraries are installed (see below).
3. Run the application by executing `python app.py` in the terminal or command prompt.

Required Libraries

Make sure to have the following libraries installed:

- Flask
- OpenCV (cv2)
- NumPy
- PyTorch (for YOLO)

You can install these libraries using pip:

```
pip install flask opencv-python numpy torch torchvision
```