**RETINAFACE + USER LEARNING + CREATE FACE DATABASE**

**Plan:**

1. Build base RetinaFace detector class
2. Add face landmark detection
3. Implement facial features learning with deep learning
4. Add data storage functionality

python main.py --shape-predictor shape\_predictor\_68\_face\_landmarks.dat --enable-learning

**TÊN DỰ ÁN: HỆ THỐNG PHÁT HIỆN BUỒN NGỦ TIÊN TIẾN (ADVANCED DROWSINESS DETECTION SYSTEM)**

**Mục tiêu chính:** Giám sát người dùng và cảnh báo khi có dấu hiệu buồn ngủ để nâng cao an toàn (ví dụ: cho người lái xe, người vận hành máy).

**I. Cách thức hoạt động tổng thể:**

Dự án hoạt động dựa trên nguyên lý phân tích video theo thời gian thực. Nó thu thập các khung hình từ camera, xử lý các khung hình này bằng các thuật toán thị giác máy tính để phát hiện khuôn mặt và theo dõi trạng thái mắt, từ đó suy ra mức độ buồn ngủ.

1. **Khởi tạo và Cấu hình:**
   * Chương trình bắt đầu bằng việc đọc và phân tích các đối số dòng lệnh (ví dụ: --use-retinaface, --enable-learning, --feature-db-path) để xác định cấu hình hoạt động.
   * Các thông tin cấu hình ban đầu và hướng dẫn sử dụng cơ bản được in ra console.
   * Một đối tượng DrowsinessDetector cốt lõi được khởi tạo với các cấu hình đã chọn.
2. **Vòng lặp xử lý chính:**
   * Hệ thống liên tục thu nhận các khung hình video từ camera.
   * **Phát hiện Khuôn mặt:** Áp dụng thuật toán phát hiện khuôn mặt đã chọn (Dlib HOG hoặc RetinaFace).
   * **Phát hiện Điểm mốc Khuôn mặt (Facial Landmarks):** Xác định 68 (hoặc một số lượng khác) điểm mốc quan trọng trên khuôn mặt, đặc biệt là quanh vùng mắt.
   * **Tính toán EAR (Eye Aspect Ratio):** Sử dụng công thức EAR dựa trên các điểm mốc mắt để đo độ mở của mắt. EAR=2⋅∣∣P1​−P4​∣∣∣∣P2​−P6​∣∣+∣∣P3​−P5​∣∣​ (Trong đó Pi​ là tọa độ các điểm mốc của mắt, ∣∣.∣∣ là khoảng cách Euclidean.)
   * **Phát hiện Buồn ngủ:** Theo dõi giá trị EAR. Nếu EAR giảm xuống dưới một ngưỡng nhất định (EAR Threshold) trong một số lượng khung hình liên tiếp, hệ thống xác định có dấu hiệu buồn ngủ.
   * 
   * **Nhận diện Người dùng (Tùy chọn):** Nếu được bật, hệ thống trích xuất vector đặc trưng khuôn mặt (facial embeddings) từ khuôn mặt được phát hiện và so sánh với cơ sở dữ liệu đã lưu trữ để nhận diện người dùng hoặc học khuôn mặt mới.
   * **Kích hoạt Cảnh báo:** Nếu phát hiện buồn ngủ, hệ thống sẽ kích hoạt cảnh báo âm thanh và/hoặc nháy đèn (tùy theo cấu hình alarm).



* + **Cập nhật Giao diện người dùng (GUI):** Hiển thị trực quan video với các bounding box khuôn mặt, trạng thái mắt, các chỉ số EAR, và trạng thái cảnh báo.
  + A computer screen with different screens

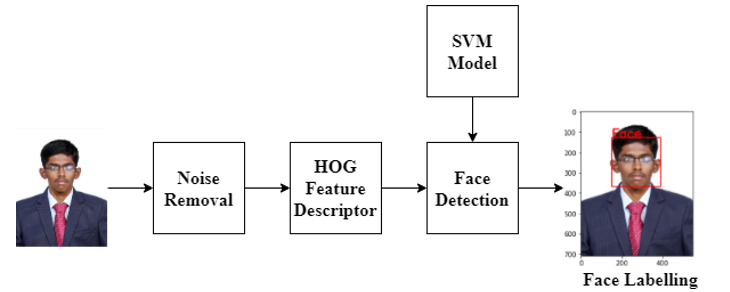
    AI-generated content may be incorrect.
  + **Xử lý Phím tắt:** Liên tục lắng nghe và phản hồi các phím tắt từ người dùng để điều khiển hệ thống trong thời gian thực (ví dụ: điều chỉnh ngưỡng EAR, bật/tắt cảnh báo, lưu cơ sở dữ liệu đặc trưng).
  + 

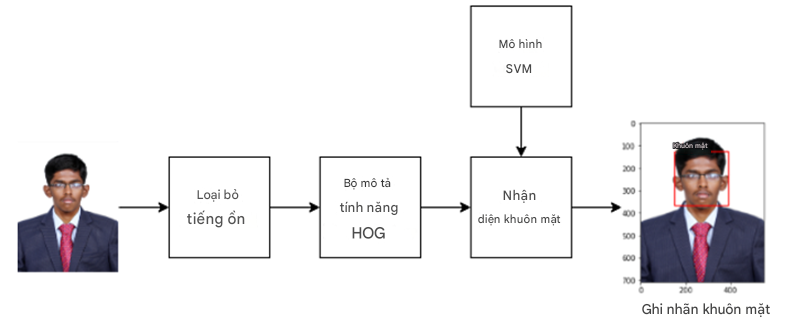
1. **Xử lý Lỗi:**
   * Chương trình có các khối try-except để bắt và xử lý các lỗi thường gặp như FileNotFoundError (ví dụ: không tìm thấy tệp mô hình dlib) hoặc các lỗi ValueError và Exception không mong muốn khác, đảm bảo độ ổn định.



**II. Các Thuật toán và Công nghệ Chính được sử dụng:**

1. **Phát hiện Khuôn mặt:**
   * **Dlib HOG (Histogram of Oriented Gradients):** Thuật toán mặc định. Khá nhanh và ổn định trên CPU, nhưng có thể kém hiệu quả hơn trong các điều kiện phức tạp (ánh sáng yếu, góc nghiêng) hoặc với nhiều khuôn mặt.

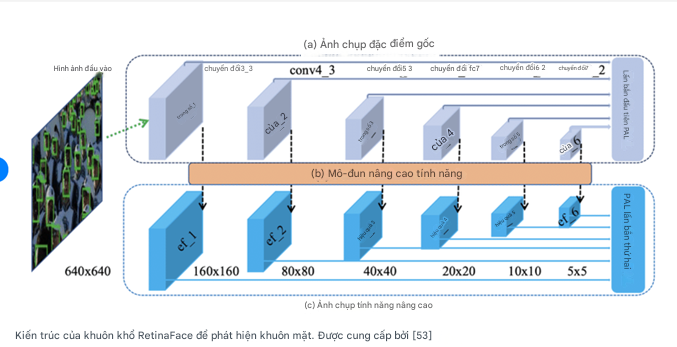




* + **RetinaFace:** Thuật toán nâng cao (tùy chọn --use-retinaface).

A diagram of a diagram of a structure

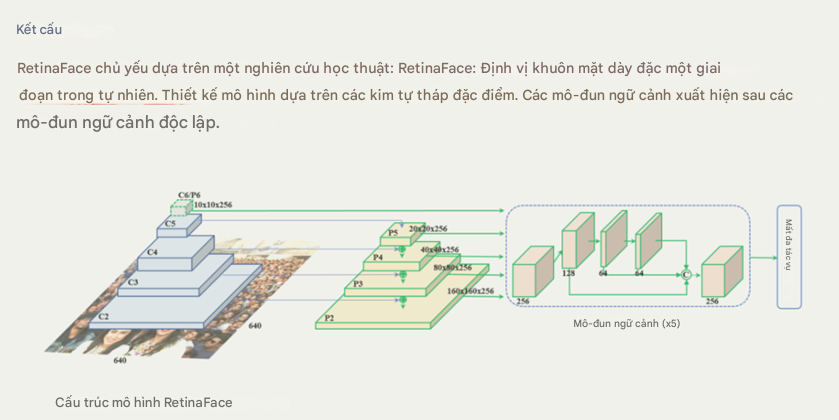
AI-generated content may be incorrect.



<https://www.researchgate.net/figure/Process-of-Dlib-HOG-face-detection_fig2_375599348>

A diagram of a structure

AI-generated content may be incorrect.



<https://sefiks.com/2021/04/27/deep-face-detection-with-retinaface-in-python/>

* + - **Ưu điểm:** Độ chính xác rất cao, đặc biệt với khuôn mặt nhỏ, bị che khuất một phần (occlusion), hoặc ở các góc độ khó. Sử dụng kiến trúc Deep Learning tiên tiến (có thể bao gồm FPN, multi-task learning).
    - **Nhược điểm:** Thường yêu cầu GPU mạnh để đạt hiệu suất thời gian thực cao.
    - **Ưu tiên lựa chọn:** Nếu hệ thống có GPU đủ mạnh, RetinaFace là lựa chọn ưu tiên cho độ chính xác vượt trội. Nếu không, hệ thống sẽ tự động chuyển sang Dlib HOG.

1. **Phát hiện Điểm mốc Khuôn mặt:**
   * Thường sử dụng mô hình được đào tạo trên Dlib (hoặc tương tự) để xác định các điểm mốc 68 điểm trên khuôn mặt, từ đó tính toán EAR.
2. **Phát hiện Buồn ngủ:**
   * **Công thức EAR (Eye Aspect Ratio):** Là công thức toán học chính được sử dụng để định lượng độ mở của mắt.
   * **Ngưỡng (Threshold):** Một giá trị EAR cố định mà người dùng có thể điều chỉnh (+/-).
   * **Bộ đếm (Counters):** Theo dõi số lượng khung hình liên tiếp mà EAR nằm dưới ngưỡng.
3. **Học và Nhận diện Khuôn mặt (Tùy chọn):**
   * Yêu cầu thư viện **PyTorch** (--enable-learning).
   * Sử dụng các mô hình học sâu (Deep Learning models) để trích xuất **đặc trưng khuôn mặt (facial embeddings)** – các vector số đại diện cho khuôn mặt của mỗi người.
   * Cơ sở dữ liệu đặc trưng (facial\_features.pkl) lưu trữ các vector này cùng với ID/tên người dùng.
   * Khi nhận diện, hệ thống so sánh vector đặc trưng hiện tại với các vector trong DB để tìm danh tính phù hợp.

**III. Khả năng xử lý các trường hợp đặc biệt:**

1. **Nhận diện riêng từng trường hợp theo data có sẵn (Nhận diện Người dùng):**
   * **Có khả năng:** Nếu --enable-learning và --user-recognition được bật, hệ thống có thể học và nhận diện các cá nhân đã được lưu trữ trong facial\_features.pkl. Dữ liệu ở đây là các "facial embeddings" (vector đặc trưng) của từng người.
   * **Phân loại Data:** Cơ sở dữ liệu chứa các cặp (ID người dùng, vector đặc trưng khuôn mặt), cho phép hệ thống "nhớ" và phân biệt các cá nhân khác nhau.

A black text on a white background

AI-generated content may be incorrect.

1. **Khả năng xử lý các trường hợp khuyết tật mặt hay bị hí mắt:**
   * **Khuyết tật mặt ảnh hưởng đến vùng mắt:**
     + **Hạn chế:** Khả năng xử lý phụ thuộc vào mức độ biến dạng. Nếu khuyết tật làm biến dạng nghiêm trọng hoặc che khuất các điểm mốc cần thiết cho việc tính toán EAR, độ chính xác sẽ giảm mạnh hoặc không thể phát hiện. RetinaFace có thể tốt hơn Dlib HOG với các trường hợp che khuất một phần, nhưng biến dạng nặng vẫn là thách thức.
     + 
   * **Người bị hí mắt (mắt nhỏ tự nhiên):**
   * A close up of a person's eyes

     AI-generated content may be incorrect.
     + **Với phát hiện buồn ngủ (EAR):** Có thể dẫn đến dương tính giả (false positive) do EAR tự nhiên thấp.
       - **Giải pháp:** Người dùng có thể **điều chỉnh ngưỡng EAR (+/-)** xuống thấp hơn để phù hợp với đặc điểm mắt của người đó.
       - **Nâng cao:** Một phiên bản phát triển hơn có thể lưu trữ ngưỡng EAR cá nhân hóa cho từng người dùng sau khi nhận diện, giúp hệ thống tự động điều chỉnh ngưỡng phù hợp.
     + **Với nhận diện người dùng:** Các thuật toán học sâu hiện đại được huấn luyện trên dữ liệu khuôn mặt đa dạng thường có khả năng nhận diện tốt ngay cả với người có mắt hí, vì đặc điểm "mắt hí" cũng là một phần của đặc trưng khuôn mặt được học.

