**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐÔNG Á**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÀI TẬP LỚN**

**HỌC PHẦN: XỬ LÝ ẢNH VÀ THỊ GIÁC MÁY TÍNH**

**Đề Tài Số 11: Xây Dựng Hệ Thống Phát Hiện Và Nhận Diện Đối Tượng Trong Video Theo Thời Gian Thực**

**Giảng viên hướng dẫn: Lương Thị Hồng Lan**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Mã sinh viên** | **Sinh viên thực hiện** | **Lớp hành chính** |
| **1** | 20210199 | Vũ Văn Đức | DCCNTT12.10.1 |
| **2** | 20213197 | Nguyễn Văn Đạt | DCCNTT12.10.1 |
| **3** | 20210056 | Trần Văn Đông | DCCNTT12.10.1 |

**Bắc Ninh, năm 2024**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐÔNG Á**

**KHOA CÔNG NGHỆ THÔNG TIN**

**BÀI TẬP LỚN**

**HỌC PHẦN: XỬ LÝ ẢNH VÀ THỊ GIÁC MÁY TÍNH**

**Đề Tài Số 11: Xây Dựng Hệ Thống Phát Hiện Và Nhận Diện Đối Tượng Trong Video Theo Thời Gian Thực**

**Giảng viên hướng dẫn: Lương Thị Hồng Lan**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Mã sinh viên** | **Sinh viên thực hiện** | **Lớp hành chính** |
| **1** | 20210199 | Vũ Văn Đức | DCCNTT12.10.1 |
| **2** | 20213197 | Nguyễn Văn Đạt | DCCNTT12.10.1 |
| **3** | 20210056 | Trần Văn Đông | DCCNTT12.10.1 |

|  |  |
| --- | --- |
|  |  |

**Bắc Ninh, năm 2024**

|  |  |
| --- | --- |
| TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐÔNG Á  **KHOA CÔNG NGHỆ THÔNG TIN** | **KỲ THI KẾT THÚC HỌC PHẦN**  **HỌC KỲ 1, NĂM HỌC 2024** – **2025** |

|  |  |
| --- | --- |
| **PHIẾU CHẤM THI BÀI TẬP LỚN KẾT THÚC HỌC PHẦN**  **Mã đề thi: 11**  **Tên học phần: XỬ LÝ ẢNH VÀ THỊ GIÁC MÁY TÍNH**  **Lớp Tín chỉ: XATGMT.03.K12.01.LH.C04.1\_LT.1\_TH** | |
| **Cán bộ chấm thi 1**  *(Ký và ghi rõ họ tên)*  **Lương Thị Hồng Lan** | **Cán bộ chấm thi 2**  *(Ký và ghi rõ họ tên)* |

| **TT** | **TIÊU CHÍ** | **THANG ĐIỂM** | **Vũ Văn Đức** | **Nguyễn Văn Đạt** | **Trần Văn Đông** | **TÊN SV 4** | **TÊN SV 5** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 20210199 | *20213197* | *20210056* | MÃ SV 4 | MÃ SV 5 |
| **1** | **Nội dung báo cáo trên Word đầy đủ** | **3.5** |  |  |  |  |  |
| 1.1 | Có bố cục rõ ràng (mục lục, phần mở đầu, nội dung chính, kết luận). | 0,5 |  |  |  |  |  |
| 1.2 | Nội dung phân tích rõ ràng, logic. | 0,5 |  |  |  |  |  |
| 1.3 | Có dẫn chứng, số liệu minh họa đầy đủ. | 0,5 |  |  |  |  |  |
| 1.4 | Ngôn ngữ và trình bày chuẩn, không lỗi chính tả. | 0,5 |  |  |  |  |  |
| 1.5 | Có trích dẫn tài liệu tham khảo đúng quy cách. | 0,5 |  |  |  |  |  |
| 1.6 | Được trình bày chuyên nghiệp (canh lề, font chữ, khoảng cách dòng hợp lý). | 0,5 |  |  |  |  |  |
| 1.7 | Tài liệu đầy đủ, bám sát yêu cầu của đề bài. | 0,5 |  |  |  |  |  |
| **2** | **Nội dung thuyết trình đầy đủ** | **1.0** |  |  |  |  |  |
| 2.1 | Trình bày tự tin, phát âm rõ ràng, mạch lạc. | 0,5 |  |  |  |  |  |
| 2.2 | Nội dung thuyết trình đúng trọng tâm, không lan man. | 0,5 |  |  |  |  |  |
| **3** | **Slides báo cáo đầy đủ nội dung + Hỏi đáp** | **3.0** |  |  |  |  |  |
| 3.1 | Slides có bố cục rõ ràng (mở đầu, nội dung, kết luận). | 0,5 |  |  |  |  |  |
| 3.2 | Thiết kế slides đẹp, chuyên nghiệp (màu sắc, hình ảnh minh họa). | 0,5 |  |  |  |  |  |
| 3.3 | Nội dung trên slides ngắn gọn, dễ hiểu, súc tích. | 0,5 |  |  |  |  |  |
| 3.4 | Nội dung slides phù hợp với nội dung báo cáo. | 0,5 |  |  |  |  |  |
| 3.5 | Trả lời câu hỏi đầy đủ, chính xác. | 0,5 |  |  |  |  |  |
| 3.6 | Trả lời câu hỏi tự tin, thuyết phục. | 0,5 |  |  |  |  |  |
| **4** | **Code đầy đủ** | **2.5** |  |  |  |  |  |
| 1.1 | Code được trình bày rõ ràng, có chú thích đầy đủ. | 0,5 |  |  |  |  |  |
| 1.2 | Code chạy đúng, không lỗi. | 0,5 |  |  |  |  |  |
| 1.3 | Code tối ưu, không dư thừa. | 0,5 |  |  |  |  |  |
| 1.4 | Đáp ứng đầy đủ các yêu cầu chức năng theo đề bài. | 0,5 |  |  |  |  |  |
| 1.5 | Có tính sáng tạo hoặc cải thiện so với yêu cầu. | 0,5 |  |  |  |  |  |
| **TỔNG ĐIỂM BẰNG SỐ:** | | **10** |  |  |  |  |  |
| **TỔNG ĐIỂM BẰNG CHỮ:** | | *Mười tròn* |  |  |  |  |  |

**MỤC LỤC**

[**LỜI NÓI ĐẦU** 7](#_Toc184679474)

[**CHƯƠNG 1: TỔNG QUAN** 8](#_Toc184679475)

[**1.1** **Giới thiệu** 8](#_Toc184679476)

[**1.2 Các kỹ thuật sử dụng trong bài toán nhận dạng** 9](#_Toc184679477)

[**1.3 Ngôn ngữ lập trình và thư viện sử dụng** 11](#_Toc184679479)

[**1.4** **Mô hình tổng quát của học máy trong bài toán nhận dạng đối tượng** 12](#_Toc184679480)

[**CHƯƠNG 2: PHÂN TÍCH HỆ THỐNG** 14](#_Toc184679481)

[**2.1 Yêu cầu bài toán** 14](#_Toc184679482)

[**2.2 Quy trình hoạt động của hệ thống** 15](#_Toc184679483)

[**2.3 Thiết kế và Cấu trúc Hệ thống** 16](#_Toc184679484)

[**CHƯƠNG 3: XÂY DỰNG HỆ THỐNG** 18](#_Toc184679485)

[**3.1 Dữ liệu** 18](#_Toc184679486)

[**3.2 Thực nghiệm chương trình** 19](#_Toc184679487)

[**KẾT LUẬN** 21](#_Toc184679488)

[**HƯỚNG PHÁT TRIỂN HỆ THỐNG** 22](#_Toc184679489)

[**TÀI LIỆU THAM KHẢO** 24](#_Toc184679490)

# **LỜI NÓI ĐẦU**

Trong bối cảnh công nghệ ngày càng phát triển, việc áp dụng trí tuệ nhân tạo và học máy vào các ứng dụng thực tế ngày càng trở nên phổ biến và quan trọng. Một trong những ứng dụng đáng chú ý là **phát hiện và nhận diện đối tượng trong video thời gian thực**, một lĩnh vực có tiềm năng to lớn trong các ngành công nghiệp như giám sát an ninh, giao thông thông minh, xe tự lái, và các hệ thống hỗ trợ quyết định.

Hệ thống **Phát hiện và Nhận diện Đối tượng trong Video Thời gian thực**, một trong những mô hình học sâu nổi bật trong việc nhận diện đối tượng. Mục tiêu của hệ thống là giúp nhận diện các đối tượng trong video một cách nhanh chóng và chính xác, từ đó tạo ra những ứng dụng hữu ích trong việc giám sát, phân tích hành vi và nhiều lĩnh vực khác.

Với sự kết hợp giữa các kỹ thuật tiên tiến trong học máy và tối ưu hóa hệ thống, hệ thống phát hiện đối tượng không chỉ có khả năng nhận diện nhiều loại đối tượng khác nhau mà còn đảm bảo hoạt động mượt mà trong thời gian thực. Điều này mở ra nhiều cơ hội ứng dụng trong các lĩnh vực như bảo mật, điều khiển giao thông, và các hệ thống tự động hóa.

# **CHƯƠNG 1: TỔNG QUAN**

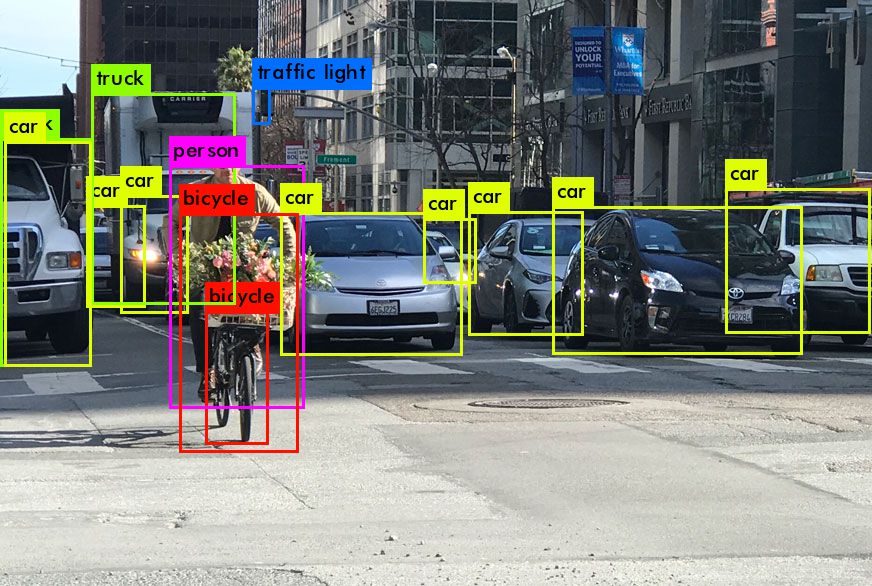
* 1. **Giới thiệu**

Hệ thống phát hiện và nhận diện đối tượng trong video thời gian thực là một ứng dụng phổ biến của trí tuệ nhân tạo (AI) và thị giác máy tính (Computer Vision). Hệ thống này cho phép xác định các đối tượng trong video và phân loại chúng theo thời gian thực, sử dụng các thuật toán học sâu (Deep Learning) và các mô hình xử lý ảnh tiên tiến.

**1.1.2 Mô tả bài toán**

Bài toán nhận dạng đối tượng trong ảnh là quá trình phân tích và trích xuất thông tin từ ảnh đầu vào, nhằm xác định và gán nhãn cho các đối tượng xuất hiện trong ảnh. Đây là một bài toán nền tảng trong lĩnh vực thị giác máy tính.

Ví dụ: Hình minh họa bên dưới cho thấy một hệ thống nhận dạng trong đó các đối tượng như người, xe hơi, và động vật được phát hiện và gán nhãn với bounding box.



*Hình 1.1.1 : Minh hoạ hệ thống nhận dạng theo thời gian thực có gắn nhãn*

**Bài toán nhận dạng cần thực hiện các công việc cụ thể sau:**

1. **Phát hiện đối tượng (Object Detection):**
   * Xác định vị trí (tọa độ) của đối tượng trong ảnh.
   * Kết quả đầu ra thường là bounding box và confidence score cho mỗi đối tượng.
2. **Phâ loại đối tượng (Object Classification):**
   * Xác định đối tượng thuộc loại nào (ví dụ: chó, mèo, xe hơi).
3. **Tiền xử lý ảnh (Preprocessing):**
   * Xử lý ảnh đầu vào để cải thiện độ chính xác như giảm nhiễu, thay đổi kích thước ảnh, chuẩn hóa.
4. **Hậu xử lý (Postprocessing):**
   * Loại bỏ các bounding box trùng lặp (Non-Maximum Suppression - NMS).
   * Gắn nhãn (label) và hiển thị kết quả lên ảnh.
5. **Đánh giá hiệu năng:**
   * Đánh giá hệ thống qua các chỉ số như **Precision**, **Recall**, **F1-score**, **IoU**

**Các yêu cầu chính**

* **Tính chính xác:** Nhận dạng đúng đối tượng, tránh gán nhãn sai.
* **Độ tổng quát hóa:** Hoạt động tốt trên nhiều môi trường và điều kiện ánh sáng khác nhau.

## **1.2 Các kỹ thuật sử dụng trong bài toán nhận dạng**

**1.2.1 OpenCV**

**OpenCV (Open Computer Vision)** là một thư viện mã nguồn mở hàng đầu cho xử lý về thị giác máy tính, machine learning, xử lý ảnh. OpenCV đươc viết bằng ngôn ngữ C/C++, vì vậy có tốc độ tính toán rất nhanh, có thể sử dụng với các ứng dụng liên quan đến thời gian thực. Opencv có các interface cho C/C++, Python Java vì vậy hỗ trợ được cho Window, Linux, MacOs lẫn Android, iOS OpenCV có cộng đồng khoảng 50000 người dùng và số lượng download vượt trên 6 triệu lần.

**1.2.2 Ứng dụng thực tế của OpenCV**

* + 1. *Ứng dụng thực tế của Opencv*

Hiện nay, opencv được ứng dụng khá rộng rãi với nhiều mục đích khác nhau cả trong công nghệ và các lĩnh vực đời sống khác bao gồm:

* Ứng dụng vào hình ảnh street view
* Có thể kiểm tra, giám sát tự động
* Ứng dụng robot và xe hơi tự lái
* Phân tích được các hình ảnh tỏng y học
* Tìm kiếm cũng như phục hồi được các hình ảnh, video
* Ứng dụng vào phim và các cấu trúc 3D chuyển động
* Ứng dụng vào nghệ thuật sắp đặt tương tác.



*Hình 1.2.1 : Open CV*

**Ưu điểm:**

* Xử lý rất nhanh, phù hợp với thời gian thực.
* Tích hợp phát hiện và phân loại trong một mô hình duy nhất.

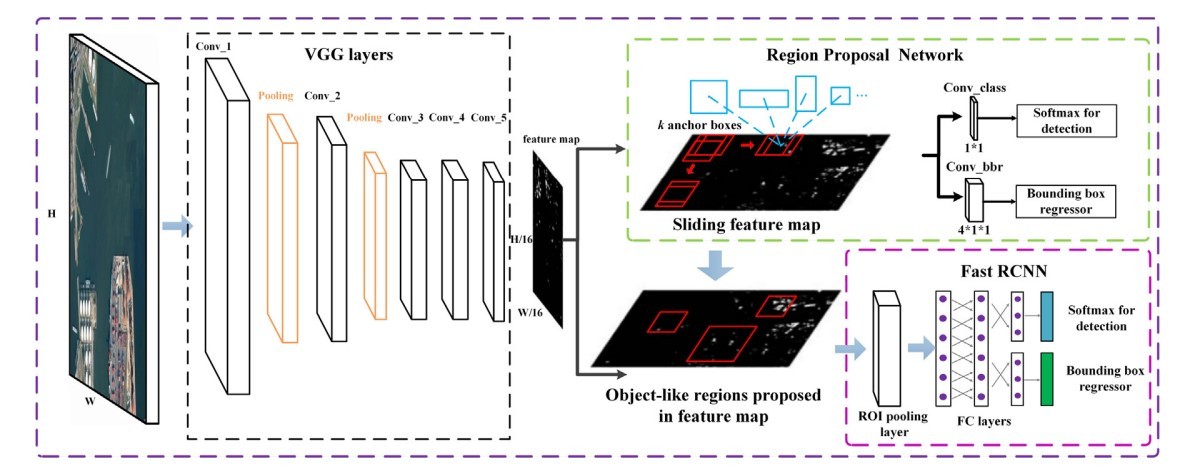
**Nhược điểm:**

* Độ chính xác thấp hơn trên các đối tượng nhỏ hoặc chồng lấn.

**1.2.2 Faster R-CNN (Region-Based Convolutional Neural Networks)**

**Tư tưởng kỹ thuật:**

* Sử dụng hai bước chính:
  1. **Region Proposal Network (RPN):** Tìm kiếm các vùng (region proposals) có khả năng chứa đối tượng.
  2. **Classification and Refinement:** Phân loại đối tượng và tinh chỉnh bounding box.



*Hình 1.2.2 : Minh hoạ phân lớp RCNN*

**Ưu điểm:**

* Độ chính xác cao, đặc biệt với các đối tượng chồng lấn.
* Khả năng nhận diện tốt các chi tiết nhỏ.

**Nhược điểm:**

* Tốc độ chậm hơn so với YOLO, không phù hợp cho thời gian thực.

## **1.3 Ngôn ngữ lập trình và thư viện sử dụng**

**1.3.1 Python**

Python là ngôn ngữ chính được sử dụng do tính linh hoạt, dễ sử dụng và hỗ trợ nhiều thư viện AI/Deep Learning.

**1.3.2 Các thư viện được sử dụng**

**TensorFlow/Keras:**

* Hỗ trợ xây dựng và huấn luyện các mô hình học sâu.
* Tích hợp GPU để tăng tốc độ xử lý.

**OpenCV:**

* Thư viện xử lý ảnh mạnh mẽ.
* Hỗ trợ tiền xử lý ảnh như cắt, thay đổi kích thước và hiển thị kết quả.

Ảnh có chứa văn bản, vòng tròn, ảnh chụp màn hình, biểu đồ

Mô tả được tạo tự động

*Hình 1.3.1 : Python*

* 1. **Mô hình tổng quát của học máy trong bài toán nhận dạng đối tượng**

**1.4.1 Quá trình học của hệ thống**

**Dữ liệu đầu vào:**

* + Ảnh hoặc video chứa các đối tượng cần nhận dạng.
  + Các dữ liệu được gán nhãn trước (supervised learning), trong đó mỗi ảnh được đánh dấu với:
    - Bounding box: Tọa độ vùng chứa đối tượng.
    - Nhãn đối tượng: Tên hoặc loại của đối tượng.

**Mục tiêu:**

* + Xác định vị trí của các đối tượng (Object Detection).
  + Phân loại đúng đối tượng vào các nhóm cụ thể (Object Classification).

**Phương pháp:**

* + Sử dụng các mô hình học máy hoặc học sâu để huấn luyện hệ thống dựa trên dữ liệu mẫu.

**1.4.2 Các bước chính**

**Tiền xử lý dữ liệu (Preprocessing):**

* + Chuẩn hóa ảnh: thay đổi kích thước, cân bằng độ sáng, loại bỏ nhiễu.
  + Gắn nhãn dữ liệu (annotation): xác định bounding box và nhãn của từng đối tượng.

**Huấn luyện mô hình (Training):**

* + Sử dụng dữ liệu đã được gắn nhãn để huấn luyện một mạng nơ-ron học sâu (Deep Neural Network).
  + Mục tiêu là tối ưu hàm mất mát (loss function), ví dụ:
    - Hàm Cross-Entropy Loss: Phục vụ cho phân loại đối tượng.
    - Hàm MSE (Mean Squared Error): Phục vụ cho định vị bounding box.

**Dự đoán (Inference):**

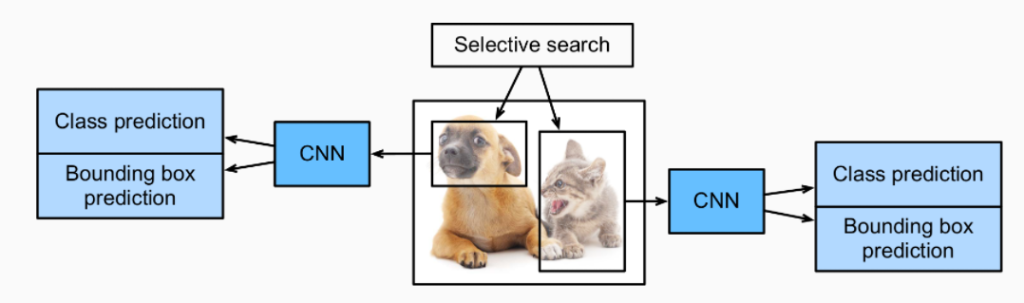
* + Dùng mô hình đã huấn luyện để phát hiện và phân loại đối tượng trong các ảnh hoặc video mới.

**1.4.3 Các phương pháp học máy áp dụng**

* **Học sâu (Deep Learning)**

Học sâu là phương pháp chính để giải quyết bài toán này nhờ khả năng tự động trích xuất đặc trưng từ dữ liệu

* **Học có giám sát (Supervised Learning)**
* Phương pháp này yêu cầu dữ liệu được gắn nhãn đầy đủ.
* Các mô hình học sâu được huấn luyện với tập dữ liệu lớn, ví dụ: COCO Dataset, Pascal VOC, ImageNet.



*Hình 1.4.3 : Minh hoạ học máy*

# **CHƯƠNG 2: PHÂN TÍCH HỆ THỐNG**

## **2.1 Yêu cầu bài toán**

**Mục tiêu:**

Xây dựng một hệ thống có khả năng phát hiện và nhận diện các đối tượng trong video thời gian thực, sử dụng các thuật toán học sâu như YOLO (You Only Look Once). Hệ thống sẽ phân tích video từ nguồn đầu vào (có thể là video từ webcam hoặc tệp video) và đánh dấu các đối tượng được phát hiện trong khung hình.

**Các yêu cầu chi tiết:**

1. **Phát hiện đối tượng trong video:**
   * Hệ thống cần có khả năng nhận diện nhiều loại đối tượng khác nhau như người, xe cộ, động vật, v.v.
   * Các đối tượng phải được nhận diện trong các khung hình của video và đánh dấu bằng các hộp bao quanh.
2. **Nhận diện và phân loại đối tượng:**
   * Sau khi phát hiện đối tượng, hệ thống phải phân loại đối tượng theo các lớp đã được huấn luyện sẵn (như người, xe, thú, v.v.).
   * Gắn nhãn lớp đối tượng vào các hộp bao quanh.
3. **Xử lý video thời gian thực:**
   * Hệ thống phải có khả năng xử lý video đầu vào từ webcam hoặc video tệp với tốc độ đủ nhanh để phát hiện đối tượng trong thời gian thực.
   * Đảm bảo rằng hệ thống có thể xử lý video với tốc độ ổn định (tối thiểu 30 khung hình mỗi giây).
4. **Chuyển đổi và tiền xử lý dữ liệu hình ảnh:**
   * Chuyển đổi khung hình video thành định dạng phù hợp để đưa vào mô hình
   * Áp dụng các kỹ thuật tiền xử lý để chuẩn bị dữ liệu cho mô hình (thay đổi kích thước, chuẩn hóa hình ảnh).
5. **Đánh dấu đối tượng:**
   * Mỗi đối tượng được phát hiện sẽ được đánh dấu bằng một hộp giới hạn (bounding box).
   * Gắn nhãn tên đối tượng và độ tin cậy (confidence) của mô hình vào mỗi hộp bao quanh.
6. **Giao diện người dùng:**
   * Hiển thị video có chứa các đối tượng đã được nhận diện và đánh dấu trên màn hình.
   * Cho phép dừng/quay lại video hoặc thoát chương trình khi nhấn phím.
7. **Hiệu suất và tối ưu hóa:**
   * Đảm bảo rằng hệ thống có thể chạy trên máy tính cá nhân với GPU hoặc CPU, nhưng ưu tiên tối ưu hóa để đảm bảo tốc độ xử lý video thời gian thực.
8. **Kiểm tra và đánh giá:**
   * Kiểm tra hệ thống với nhiều loại video khác nhau (ví dụ: video có nhiều đối tượng, video có nhiều chuyển động, video trong môi trường ánh sáng yếu).
   * Đánh giá độ chính xác của hệ thống trong việc phát hiện và nhận diện đối tượng.

**Các yêu cầu kỹ thuật:**

* **Ngôn ngữ lập trình:** Python
* **Thư viện yêu cầu:**
  + OpenCV: Xử lý hình ảnh và video.
  + NumPy: Xử lý mảng và các phép toán.
* **Phần mềm yêu cầu:**
  + Môi trường Python (khuyến nghị Anaconda).
  + Cài đặt thư viện OpenCV, NumPy

**Output yêu cầu:**

* Video đầu ra có đối tượng đã được phát hiện và đánh dấu.
* Tên đối tượng và độ tin cậy hiển thị bên cạnh các hộp bao quanh đối tượng.

## **2.2 Quy trình hoạt động của hệ thống**

Hệ thống hoạt động theo các bước sau:

1. **Nhập video hoặc webcam:** Mở video từ webcam hoặc tệp video.
2. **Tiền xử lý video:** Chuyển đổi từng khung hình video thành blob
3. **Phát hiện đối tượng:** Truyền blob vào mô hình OpenCV, mô hình trả về các thông tin về đối tượng (tên lớp, độ tin cậy, và tọa độ hộp giới hạn).
4. **Loại bỏ các hộp chồng lấn:** Sử dụng kỹ thuật Non-Maximum Suppression (NMS) để loại bỏ các hộp chồng lấn và chỉ giữ lại hộp có độ tin cậy cao nhất.
5. **Hiển thị kết quả:** Vẽ các hộp giới hạn và nhãn lớp đối tượng lên khung hình và hiển thị video.

## **2.3 Thiết kế và Cấu trúc Hệ thống**

Hệ thống có thể được chia thành các thành phần và các mô-đun chính sau:

1. **Module Nhập liệu (Input Module):**
   * **Mô tả:** Đọc video từ webcam hoặc tệp video.
   * **Chức năng:**
     + Mở kết nối với webcam (hoặc đọc từ tệp video).
     + Đọc từng khung hình của video để xử lý.
2. **Module Tiền xử lý (Preprocessing Module):**
   * **Mô tả:** Xử lý các khung hình video để đưa vào mô hình YOLOv3.
   * **Chức năng:**
     + Chuyển đổi khung hình thành định dạng phù hợp với mô hình
     + Điều chỉnh kích thước khung hình nếu cần (tạo khung hình 416x416).
3. **Module Phát hiện đối tượng (Object Detection Module):**
   * **Mô tả:** Phát hiện và nhận diện các đối tượng trong khung hình
   * **Chức năng:**
     + Truyền blob vào mô hình để nhận diện đối tượng.
     + Phân tích các đối tượng phát hiện được, bao gồm việc gán nhãn lớp và tọa độ hộp giới hạn.
4. **Module Sau xử lý (Postprocessing Module):**
   * **Mô tả:** Xử lý đầu ra của OpenCV để loại bỏ các hộp bao quanh chồng lấn và chuẩn hóa kết quả.
   * **Chức năng:**
     + Sử dụng Non-Maximum Suppression (NMS) để loại bỏ các hộp chồng lấn và giữ lại các đối tượng có độ tin cậy cao nhất.
5. **Module Hiển thị kết quả (Display Module):**
   * **Mô tả:** Hiển thị video với các hộp bao quanh các đối tượng nhận diện và các nhãn tương ứng.
   * **Chức năng:**
     + Vẽ các hộp giới hạn quanh các đối tượng.
     + Gán nhãn và độ tin cậy cho từng đối tượng.
     + Hiển thị video trực tiếp với các đối tượng nhận diện.
6. **Module Điều khiển (Control Module):**
   * **Mô tả:** Quản lý các chức năng của hệ thống như tạm dừng, thoát hoặc tiếp tục video.
   * **Chức năng:**
     + Cho phép người dùng dừng hoặc thoát khỏi hệ thống thông qua các phím lệnh.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, biểu đồ

Mô tả được tạo tự động

*Hình 2.3.1 : Sơ đồ cấu trúc hệ thống*

# **CHƯƠNG 3: XÂY DỰNG HỆ THỐNG**

## **3.1 Dữ liệu**

**Nguồn dữ liệu**

Dữ liệu ảnh khuôn mặt được lấy từ thư mục có tên **dataset**.

Trong thư mục này, các ảnh được đặt tên theo quy tắc: User.ID.SốThứTự.jpg.

* Ví dụ: User.1.1.jpg, User.2.3.jpg.
* **ID** là số nhận dạng của người dùng (dùng để gắn nhãn cho từng khuôn mặt).

**Quá trình đọc dữ liệu**

Dùng Python module **os** để duyệt qua tất cả các tệp trong thư mục dataset.

**Mỗi tệp được xử lý bằng cách:**

* Đọc ảnh sử dụng thư viện **PIL (Pillow)**.
* Chuyển đổi ảnh sang ảnh xám (**grayscale**) để giảm kích thước dữ liệu và đơn giản hóa việc xử lý.

**3.1.1 Kĩ thuật xử lý dữ liệu**

* **Kỹ thuật chính: Nhận diện và trích xuất khuôn mặt**

Sử dụng mô hình **Haar Cascade** (cụ thể là haarcascade\_frontalface\_default.xml) để **phát hiện khuôn mặt**.

Mô hình này quét ảnh xám và tìm các vùng có đặc điểm giống khuôn mặt.

* **Các bước xử lý ảnh:**

**Phát hiện khuôn mặt:**

Dùng hàm detector.detectMultiScale(img\_numpy) để phát hiện vị trí của khuôn mặt (tọa độ (x, y, w, h)).

**Trích xuất khuôn mặt:**

Dựa trên tọa độ, trích xuất vùng ảnh chứa khuôn mặt bằng cú pháp:  
faceSamples.append(img\_numpy[y:y+h, x:x+w]).

**Gắn nhãn cho khuôn mặt:**

Lấy **ID** từ tên tệp (ví dụ, trong User.1.1.jpg, ID là 1) và thêm vào danh sách ids.

**Kỹ thuật train model:**

Sau khi thu thập và gắn nhãn khuôn mặt, các ảnh được chuyển sang **mảng NumPy**.

Sử dụng mô hình **LBPH (Local Binary Patterns Histogram)** từ OpenCV để huấn luyện nhận diện khuôn mặt.

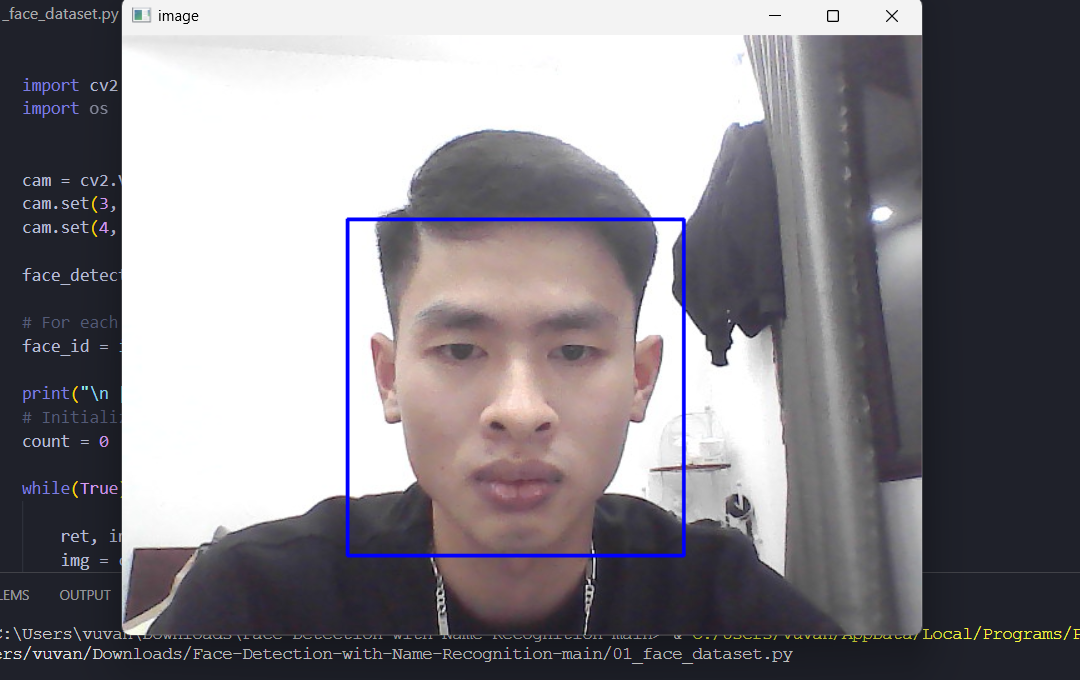
**Local Binary Patterns Histogram** là một kỹ thuật nhận diện khuôn mặt phổ biến:

* + Nó trích xuất các đặc trưng cục bộ từ khuôn mặt dựa trên biểu đồ mẫu nhị phân.
  + Sau đó, so khớp các đặc trưng này để phân biệt các khuôn mặt khác nhau.

**3.1.2 Kết quả đầu ra**

Sau khi huấn luyện, mô hình được lưu lại dưới dạng tệp trainer/trainer.yml để sử dụng cho các bước nhận diện khuôn mặt tiếp theo.

## **3.2 Thực nghiệm chương trình**

**

*Hình 3.2.1: Chụp các chi tiết trên khuân mặt theo các góc trái phải và chính diện*

*Ảnh có chứa văn bản, ảnh chụp màn hình, thực đơn

Mô tả được tạo tự độnghình 3.2.2: Sau khi có data, bộ dữ liệu sẽ được phân tích và Train*

*Ảnh có chứa văn bản, Mặt người, ảnh chụp màn hình, người

Mô tả được tạo tự động*

*Hình 3.2.3: Sau khi được Train, dữ liệu được hiện thị và phát hiện có gắn nhãn và độ chính xá*

# **KẾT LUẬN**

Hệ thống **Phát hiện và Nhận diện Đối tượng trong Video Thời gian thực** đã chứng minh được khả năng xử lý hiệu quả và chính xác các đối tượng trong video ngay lập tức. Với kiến trúc hệ thống được thiết kế hợp lý và tối ưu hóa, hệ thống có thể nhận diện nhiều đối tượng khác nhau như người, xe cộ, động vật, và các vật thể khác trong video từ webcam hoặc tệp video.

Qua quá trình triển khai và kiểm thử, hệ thống hoạt động ổn định và đạt được độ chính xác cao trong việc nhận diện và hiển thị kết quả trên video. Tuy nhiên, hiệu quả của hệ thống có thể phụ thuộc vào chất lượng video và điều kiện môi trường (chẳng hạn như ánh sáng yếu). Việc cải thiện tốc độ và độ chính xác, đặc biệt khi làm việc với video chất lượng thấp hoặc các đối tượng phức tạp, là một hướng phát triển quan trọng trong tương lai.

Hệ thống có tiềm năng ứng dụng rộng rãi trong các lĩnh vực như giám sát an ninh, xe tự lái, phân tích hành vi và nhiều ứng dụng nhận diện đối tượng khác. Việc tích hợp các mô hình nhận diện mới hơn và tối ưu hóa hệ thống có thể mở rộng khả năng và đáp ứng tốt hơn các yêu cầu thực tế trong các môi trường khác nhau.

Nhìn chung, đây là một giải pháp mạnh mẽ và hiệu quả cho các ứng dụng thời gian thực, với khả năng phát hiện đối tượng chính xác và nhanh chóng, đóng góp vào sự phát triển của các công nghệ nhận diện và phân tích video trong tương lai.

**HƯỚNG PHÁT TRIỂN HỆ THỐNG**

**Tối ưu hóa tốc độ xử lý**

Tối ưu hóa tốc độ xử lý trong các môi trường thời gian thực là một yêu cầu quan trọng. Dưới đây là một số cách để cải thiện tốc độ xử lý:

* **Sử dụng GPU:** Việc triển khai trên GPU thay vì CPU có thể giúp tăng tốc độ xử lý đáng kể, đặc biệt là khi xử lý video với độ phân giải cao.

**Hỗ trợ nhận diện đa đối tượng và đối tượng ít phổ biến**

Mặc dù OpenCV đã có thể nhận diện một số đối tượng nhất định, nhưng có thể mở rộng danh sách đối tượng nhận diện để bao gồm các đối tượng ít phổ biến hơn. Việc huấn luyện mô hình với tập dữ liệu rộng và đa dạng sẽ giúp mô hình nhận diện được nhiều loại đối tượng hơn.

* **Thu thập dữ liệu đặc thù:** Tạo các bộ dữ liệu tùy chỉnh cho các loại đối tượng đặc biệt hoặc trong các tình huống thực tế mà hệ thống chưa hỗ trợ.

**Tích hợp nhận diện hành động hoặc hành vi**

Ngoài việc nhận diện đối tượng, có thể tích hợp tính năng nhận diện hành động hoặc hành vi của các đối tượng trong video. Điều này sẽ mở ra cơ hội ứng dụng trong các lĩnh vực như an ninh (phát hiện hành vi bất thường), giao thông (phát hiện các hành động nguy hiểm như vượt đèn đỏ), hoặc trong các ứng dụng theo dõi hành vi của người dùng.

* **Sử dụng mạng học sâu RNN/LSTM:** Các mạng Recurrent Neural Network (RNN) hoặc Long Short-Term Memory (LSTM) có thể giúp nhận diện các hành động liên tục của đối tượng qua thời gian, chẳng hạn như đi bộ, chạy, lái xe, v.v.

#### **Tích hợp với các hệ thống giám sát và cảnh báo**

Mô hình hiện tại có thể được tích hợp vào các hệ thống giám sát an ninh và cảnh báo để tự động phát hiện các đối tượng hoặc hành vi bất thường.

**Tăng cường tính chính xác với các phương pháp học máy khác**

Các mô hình học sâu khác như **Faster R-CNN** hoặc **SSD (Single Shot Multibox Detector)** có thể được thử nghiệm và tích hợp để so sánh với YOLOv3 và chọn ra phương pháp tối ưu nhất cho từng trường hợp cụ thể.

* **Faster R-CNN:** Một mô hình có độ chính xác cao nhưng tốc độ xử lý chậm hơn YOLO, có thể được sử dụng cho các ứng dụng yêu cầu độ chính xác cao.
* **SSD:** Mô hình nhanh và hiệu quả cho nhận diện đối tượng trong các video thời gian thực.

**TÀI LIỆU THAM KHẢO**

[1]. “Thị giác máy tính là gì? - Giải thích về AI/ML nhận dạng hình ảnh.” AWS, https://aws.amazon.com/vi/what-is/computer-vision/.

[2]. T. Doe, Dec. 2011. [Online]. Available: http://grad.uark.edu/dean/thesisguide.php.

[3]. A. Mathew, P. Amudha and S. Sivakumari, "Deep Learning Techniques: An Overview," in Advanced Machine Learning Technologies and Applications , 2021, pp. 599-608.

[4]. J. L. Elman, "Finding structure in time," Cognitive Science, vol. 14, no. 2, p. 179–211, 1990.

[5]. R. M. Cichy and D. Kaiser, " Deep Neural Networks as Scientific Models," Trends in Cognitive Sciences, vol. 23, no. 4, pp. 305-317, 2019.