**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**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 15: Xây dựng hệ thống phát triển mô hình phân loại ảnh để nhận diện các loài động vật**

|  |  |  |
| --- | --- | --- |
| **Sinh viên thực hiện** | **Lớp** | **Khoá** |
| **Trần Xuân Hiếu** | **DCCNTT12.10.5** | **K12** |
| **Trần Mạnh Hùng** | **DCCNTT12.10.5** | **K12** |
| **Đoàn Văn Minh Phong** | **DCCNTT12.10.5** | **K12** |
| **Hoàng Anh Quân** | **DCCNTT12.10.5** | **K12** |
| **Lưu Đức Tài** | **DCCNTT12.10.5** | **K12** |

***Bắc Ninh*, năm 2024**

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**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 15: Xây dựng hệ thống phát triển mô hình phân loại ảnh để nhận diện các loài động vật**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Sinh viên thực hiện** | **Mã sinh viên** | | **Điểm bằng số** | **Điểm bằng chữ** |
| **1** | **Trần Xuân Hiếu** | **20211533** | |  |  |
| **2** | **Trần Mạnh Hùng** | **20211532** | |  |  |
| **3** | **Đoàn Văn Minh Phong** | **20211556** | |  |  |
| **4** | **Hoàng Anh Quân** | **20211262** | |  |  |
| **5** | **Lưu Đức Tài** | **20211450** | |  |  |
| **Lương Thị Hồng Lan**  *(Ký và ghi rõ họ tên)* | | | **CÁN BỘ CHẤM 2**  *(Ký và ghi rõ họ tên)* | | | |

|  |  |
| --- | --- |
| 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: *15***  **Tên học phần: Xứ lý ảnh và thị giác máy tính**  **Lớp Tín chỉ: XATGMT.03.K12.05.LH.C04.1\_LT** | |
| *Lương Thị Hồng Lan*  *(Ký và ghi rõ họ tên)* | **Cán bộ chấm thi 2**  *(Ký và ghi rõ họ tên)* |

| **TT** | **TIÊU CHÍ** | **THANG ĐIỂM** | **Trần Xuân Hiếu** | **Trần Mạnh Hùng** | **Đoàn Văn Minh Phong** | **Hoàng Anh Quân** | **Lưu Đức Tài** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 20211533 | 20211532 | 20211450 | 20211262 | 20211556 |
| **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

[Chương 1: Kiến thức cơ sở 8](#_Toc184594881)

[1.1 Giới thiệu 8](#_Toc184594882)

[1.2. Kiến thức về dữ liệu 8](#_Toc184594883)

[1.3. Kiến thức về học máy và học sâu 8](#_Toc184594884)

[1.4. Các mô hình phân loại ảnh 9](#_Toc184594885)

[1.5. Huấn luyện và đánh giá mô hình 9](#_Toc184594886)

[1.6. Triển khai hệ thống 9](#_Toc184594887)

[1.7. các phương pháp sử dụng trong nhận dạng bài toán 10](#_Toc184594888)

[1.7.1. RCNN 10](#_Toc184594889)

[1.7.2 YOLO 12](#_Toc184594890)

[Chương 2. Xây dựng hệ thống 15](#_Toc184594891)

[2.1. Bài toán 15](#_Toc184594892)

[2.1.1. Bối cảnh bài toán 15](#_Toc184594893)

[2.1.2 Định nghĩa bài toán 15](#_Toc184594894)

[2.1.3. Mục tiêu bài toán 15](#_Toc184594895)

[2.1.4. Các thách thức của bài toán 16](#_Toc184594896)

[2.1.5. Các phương pháp giải quyết 16](#_Toc184594897)

[2.1.6. Ứng dụng thực tế 16](#_Toc184594898)

[2.2. Xây dựng hệ thống 17](#_Toc184594899)

[Chương 3: Kết quả thực nghiệm 28](#_Toc184594900)

[3.1. Dữ liệu 28](#_Toc184594901)

[3.1.1. Dữ liệu đầu vào 28](#_Toc184594902)

[3.1.2.Xử lý dữ liệu 29](#_Toc184594903)

[3.2. Độ đo đánh giá 30](#_Toc184594904)

[3.3.Kết quả thực nghiệm 30](#_Toc184594905)

[3.4. Kết luận 33](#_Toc184594906)

# **Chương 1: Kiến thức cơ sở**

## **1.1 Giới thiệu**

Nhận diện các loài động vật là một ứng dụng quan trọng trong lĩnh vực trí tuệ nhân tạo (AI) và học sâu (Deep Learning). Từ việc hỗ trợ bảo tồn động vật hoang dã đến phát triển các hệ thống tự động trong nông nghiệp và nghiên cứu sinh học, mô hình phân loại ảnh đang trở thành một công cụ không thể thiếu. Để xây dựng hệ thống này, người phát triển cần hiểu rõ các kiến thức cơ sở về dữ liệu, thuật toán học máy, mạng nơ-ron tích chập (CNN), cũng như cách triển khai hệ thống.

## **1.2. Kiến thức về dữ liệu**

Dữ liệu là yếu tố cốt lõi quyết định chất lượng của mô hình nhận diện động vật. Để đạt được hiệu quả cao, các bước liên quan đến dữ liệu cần được chú trọng:

* **Thu thập dữ liệu:** Việc thu thập hình ảnh động vật cần đảm bảo tính đa dạng, từ các góc chụp, điều kiện ánh sáng, đến kích thước của đối tượng. Nguồn dữ liệu có thể đến từ các bộ dữ liệu công khai như ImageNet hoặc tự thu thập từ các thiết bị như máy ảnh và cảm biến.
* **Tiền xử lý dữ liệu:** Hình ảnh cần được chuẩn hóa về kích thước, định dạng và loại bỏ các lỗi như ảnh bị mờ hoặc nhiễu. Gắn nhãn chính xác các lớp động vật là bước quan trọng để đảm bảo mô hình học được đúng đặc trưng.
* **Tăng cường dữ liệu (Data Augmentation):** Các kỹ thuật như xoay, lật, thay đổi độ sáng hay cắt ảnh giúp tăng cường độ đa dạng của dữ liệu, từ đó cải thiện khả năng tổng quát hóa của mô hình.

## **1.3. Kiến thức về học máy và học sâu**

* **Học máy (Machine Learning):** Hiểu biết cơ bản về các thuật toán phân loại như k-NN, Logistic Regression, hay Decision Tree là nền tảng để chuyển sang các kỹ thuật học sâu phức tạp hơn.
* **Học sâu (Deep Learning):** Đây là công nghệ chủ đạo trong phân loại ảnh. Đặc biệt, mạng nơ-ron tích chập (CNN) đóng vai trò quan trọng với khả năng tự động trích xuất đặc trưng từ ảnh.
  + **Cấu trúc CNN:** Gồm các lớp chính:
    - **Convolutional Layers:** Tự động trích xuất các đặc trưng như cạnh, hình dạng.
    - **Pooling Layers:** Giảm kích thước của đặc trưng, giảm độ phức tạp tính toán.
    - **Fully Connected Layers:** Tổng hợp đặc trưng và thực hiện phân loại.
  + Các hàm kích hoạt (ReLU, Softmax) và hàm mất mát (Cross-Entropy Loss) là các thành phần cơ bản giúp mô hình hoạt động hiệu quả.

## **1.4. Các mô hình phân loại ảnh**

* **Mô hình cơ bản:** Các mô hình như LeNet và AlexNet là khởi đầu cho sự phát triển của học sâu trong phân loại ảnh.
* **Mô hình hiện đại:** ResNet, EfficientNet, MobileNet và InceptionNet là những mô hình tiên tiến với khả năng học sâu vượt trội. Các mô hình này không chỉ chính xác mà còn được tối ưu để triển khai trên các thiết bị hạn chế tài nguyên.
* **Transfer Learning:** Việc sử dụng mô hình được huấn luyện sẵn trên các tập dữ liệu lớn như ImageNet giúp tiết kiệm thời gian và tài nguyên. Chỉ cần tinh chỉnh lại các lớp cuối để phù hợp với bài toán cụ thể.

## **1.5. Huấn luyện và đánh giá mô hình**

* **Huấn luyện mô hình:** Quy trình huấn luyện bao gồm việc chia dữ liệu thành tập huấn luyện, kiểm định và kiểm tra. Các siêu tham số như learning rate, số epoch và batch size cần được tối ưu để mô hình đạt hiệu năng cao nhất.
* **Đánh giá hiệu năng:** Độ chính xác (accuracy) là chỉ số phổ biến, nhưng trong các bài toán không cân bằng dữ liệu, cần sử dụng thêm Precision, Recall, và F1-score. Confusion Matrix là công cụ hữu ích để phân tích chi tiết hiệu suất mô hình.

## **1.6. Triển khai hệ thống**

* **Đóng gói mô hình:** Sử dụng các framework như Flask hoặc FastAPI để xây dựng API cung cấp dịch vụ nhận diện động vật.
* **Tối ưu hóa:** Với các thiết bị di động hoặc nhúng, cần tối ưu mô hình bằng cách giảm kích thước hoặc sử dụng các công nghệ như TensorRT và ONNX.
* **Ứng dụng thực tế:** Hệ thống có thể được ứng dụng trong các lĩnh vực như nhận diện động vật hoang dã qua camera bẫy, giám sát động vật nuôi trong trang trại, hoặc nhận diện theo thời gian thực trong các vườn thú.

## **1.7. các phương pháp sử dụng trong nhận dạng bài toán**

### **1.7.1. RCNN**

\* RCNN (Region-based Convolutional Neural Networks) là một trong những phương pháp tiên phong trong lĩnh vực **phát hiện đối tượng** trong thị giác máy tính, được đề xuất bởi **Ross B. Girshick** và các cộng sự vào năm 2014. RCNN sử dụng các **mạng nơ-ron tích chập (CNN)** để phát hiện và phân loại các đối tượng trong ảnh, đồng thời là một trong những mô hình đầu tiên đưa ra giải pháp kết hợp giữa **phát hiện đối tượng** và **học sâu**.

A diagram of a person on a bicycle

Description automatically generated

***Hình 1.1: mô hình RCNN***

\* Cấu Trúc Của RCNN:

* RCNN bao gồm ba bước chính: giới thiệu vùng ứng viên, trích xuất đặc trưng bằng CNN, và phân loại các vùng. Quy trình này được mô tả chi tiết dưới đây:
* Region Proposal (Giới thiệu các vùng ứng viên)
* Bước đầu tiên trong RCNN là xác định các vùng ứng viên, nơi mà có khả năng chứa đối tượng trong ảnh. RCNN sử dụng một phương pháp gọi là **Selective Search** để tìm kiếm các khu vực trong ảnh mà có thể chứa các đối tượng tiềm năng. Selective Search là một thuật toán chia ảnh thành các vùng (superpixels) và sau đó kết hợp chúng lại để tạo thành các vùng ứng viên có thể chứa đối tượng.
* Các vùng này được gọi là **Region Proposals**. Mỗi **region proposal** có thể chứa một đối tượng hoặc phần của một đối tượng trong ảnh.
* Trích Xuất Đặc Trưng Bằng CNN
* Sau khi có các vùng ứng viên, mỗi vùng sẽ được cắt ra và chuyển đổi thành một hình ảnh có kích thước cố định (thường là 227x227 pixels) để đưa vào **mạng nơ-ron tích chập** (CNN) để trích xuất đặc trưng. RCNN sử dụng CNN đã được huấn luyện trước (pre-trained CNN), thường là **AlexNet** hoặc **VGG**, để trích xuất các đặc trưng từ các vùng này.
* CNN sẽ giúp tạo ra các đặc trưng sâu (deep features) cho mỗi vùng, giúp nhận diện các đối tượng trong các vùng ứng viên.
* Phân Loại Các Vùng và Điều Chỉnh Bounding Boxes
* Sau khi có đặc trưng từ CNN, các vùng ứng viên sẽ được phân loại vào các lớp đối tượng khác nhau thông qua một mô hình phân loại, thường là một **mạng nơ-ron fully connected** hoặc **SVM** (Support Vector Machine).
* Cùng với việc phân loại, mô hình cũng sẽ điều chỉnh các **bounding box** (hộp giới hạn) của các đối tượng được phát hiện để phù hợp hơn với đối tượng thực tế. Việc điều chỉnh này giúp mô hình cải thiện độ chính xác của vị trí đối tượng trong ảnh.

\* Quy Trình Hoạt Động Của RCNN:

* **Bước 1**: Dùng **Selective Search** để tìm ra các vùng ứng viên trong ảnh.
* **Bước 2**: Cắt các vùng ứng viên thành các hình vuông có kích thước cố định và đưa qua mạng CNN để trích xuất đặc trưng.
* **Bước 3**: Sử dụng mô hình phân loại (SVM) để phân loại các đối tượng trong mỗi vùng và điều chỉnh bounding box.

\* Ưu Điểm và Nhược Điểm Của RCNN

* Ưu Điểm:
* **Sử Dụng CNN để Trích Xuất Đặc Trưng**: RCNN đã sử dụng mạng CNN mạnh mẽ để học các đặc trưng không gian, giúp cải thiện độ chính xác trong phát hiện đối tượng.
* **Tiến Triển So Với Các Phương Pháp Cũ**: So với các phương pháp phát hiện đối tượng trước đó (như HOG + SVM), RCNN đã cải thiện đáng kể độ chính xác.
* **Phát Hiện Đối Tượng Chính Xác**: RCNN có thể phát hiện các đối tượng một cách chính xác trong nhiều loại ảnh phức tạp.
* Nhược Điểm:
* **Hiệu Suất Thấp**: RCNN yêu cầu tính toán rất tốn kém, vì mỗi vùng ứng viên đều phải được đưa qua mạng CNN. Điều này khiến RCNN rất chậm và không phù hợp cho các ứng dụng thời gian thực.
* **Khối Lượng Tính Toán Lớn**: Việc xử lý nhiều vùng ứng viên và trích xuất đặc trưng từ chúng đòi hỏi rất nhiều tài nguyên tính toán, khiến mô hình khó triển khai trong các hệ thống với phần cứng yếu.
* **Cần Bộ Nhớ Lớn**: Vì mỗi vùng ứng viên cần phải lưu trữ các đặc trưng CNN, nên yêu cầu bộ nhớ cao, điều này làm giảm hiệu quả của hệ thống trong việc xử lý dữ liệu lớn.

### **1.7.2 YOLO**

\* YOLO (You Only Look Once) là một trong những phương pháp nổi bật và mạnh mẽ trong việc phát hiện đối tượng trong thị giác máy tính. Được giới thiệu bởi Joseph Redmon và các cộng sự vào năm 2016, YOLO đã cách mạng hóa lĩnh vực phát hiện đối tượng nhờ vào khả năng **phát hiện đối tượng trong thời gian thực** với tốc độ rất nhanh và độ chính xác cao.

Cars on the road with cars

Description automatically generated

***Hình 1.2: Phương thức hoạt động của YOLO***

\* **Quy Trình Của YOLO:**

* Quy trình của YOLO có thể được mô tả qua các bước sau:
* **Chia ảnh thành lưới (grid)**: Ảnh đầu vào được chia thành các ô lưới kích thước cố định (ví dụ 13x13 hoặc 19x19). Mỗi ô lưới sẽ "chịu trách nhiệm" phát hiện đối tượng trong khu vực của nó.
* **Dự đoán Bounding Boxes**: Mỗi ô lưới sẽ dự đoán một số lượng cố định bounding boxes, mỗi bounding box có các thông số:
* **Toạ độ** (x, y) của trung tâm bounding box.
* **Chiều rộng và chiều cao** (w, h) của bounding box.
* **Xác suất đối tượng** trong bounding box.
* **Dự đoán các nhãn đối tượng**: Mỗi ô lưới cũng dự đoán một vector xác suất cho các lớp đối tượng khác nhau. Nếu một đối tượng thuộc vào một lớp nào đó, xác suất này sẽ phản ánh mức độ tin cậy của mô hình về lớp đó.
* **Tính toán hàm mất mát**: YOLO sử dụng hàm mất mát (loss function) để tính toán độ sai lệch giữa các giá trị dự đoán và thực tế, bao gồm:
* **Sai số giữa vị trí bounding box dự đoán và thực tế**.
* **Sai số về kích thước và hình dạng bounding box**.
* **Sai số trong việc phân loại các đối tượng**.
* **Lọc kết quả**: Sau khi tất cả các dự đoán được tính toán, YOLO sử dụng một kỹ thuật gọi là **Non-Maximum Suppression (NMS)** để loại bỏ các bounding boxes trùng lặp, giữ lại những bounding boxes chính xác nhất cho mỗi đối tượng.

**\* Ưu điểm và nhược điểm của YOLO**

* Ưu Điểm:
* **Tốc độ nhanh**: YOLO có thể phát hiện đối tượng trong thời gian thực, làm cho nó rất phù hợp cho các ứng dụng yêu cầu xử lý nhanh, như xe tự lái, giám sát an ninh, và robot.
* **Đơn giản và hiệu quả**: YOLO là một mô hình đơn giản và trực tiếp, thực hiện phát hiện đối tượng trong một bước duy nhất.
* **Phát hiện nhiều đối tượng**: YOLO có thể phát hiện nhiều đối tượng trong một ảnh và trong các vị trí khác nhau mà không gặp phải vấn đề chồng chéo quá nhiều.
* Nhược Điểm:
* **Khó phát hiện đối tượng nhỏ**: YOLO có thể gặp khó khăn trong việc phát hiện đối tượng nhỏ hoặc ở xa, vì lưới được chia quá thô, làm giảm khả năng nhận diện các đối tượng này.
* **Giảm độ chính xác trong một số tình huống**: Mặc dù YOLO rất nhanh, độ chính xác của nó có thể không bằng các phương pháp như Faster R-CNN, đặc biệt khi đối tượng phức tạp hoặc có hình dáng không đồng nhất.

# **Chương 2. Xây dựng hệ thống**

## **2.1. Bài toán**

### 2.1.1. Bối cảnh bài toán

Trong thế giới tự nhiên, việc nhận diện và phân loại các loài động vật đóng vai trò quan trọng trong nhiều lĩnh vực, từ nghiên cứu sinh học, bảo tồn động vật hoang dã, cho đến các ứng dụng trong nông nghiệp và giáo dục. Sự đa dạng sinh học, cùng với những thách thức trong việc nhận diện thủ công, đã thúc đẩy việc áp dụng trí tuệ nhân tạo (AI) để tự động hóa quá trình này.

Bài toán "Nhận diện các loài động vật" sử dụng công nghệ xử lý ảnh và học sâu để phân loại hình ảnh của các loài động vật cụ thể dựa trên các đặc trưng như hình dáng, màu sắc, và họa tiết.

### 2.1.2 Định nghĩa bài toán

\* Đầu vào:

* Một bức ảnh chứa hình ảnh của một hoặc nhiều loài động vật.

\* Đầu ra:

* Một nhãn (label) chỉ định loài động vật có trong ảnh, ví dụ: "Hổ", "Voi", "Chim đại bàng", hoặc "Chó".
* Trong trường hợp có nhiều loài trong ảnh, bài toán có thể mở rộng thành phân loại đa nhãn (multi-label classification) hoặc nhận diện đối tượng (object detection) để xác định từng đối tượng cụ thể.

### 2.1.3. Mục tiêu bài toán

* Phát triển một mô hình AI có khả năng:
  1. Phân loại chính xác các loài động vật dựa trên hình ảnh đầu vào.
  2. Đạt hiệu suất cao trên các tập dữ liệu chưa thấy trong quá trình huấn luyện.
  3. Có khả năng hoạt động trong điều kiện môi trường đa dạng (ánh sáng, góc chụp, bối cảnh phức tạp).

### 2.1.4. Các thách thức của bài toán

* Độ đa dạng cao: Các loài động vật có hình dáng và kích thước khác nhau, có thể xuất hiện trong nhiều bối cảnh và điều kiện ánh sáng.
* Tương đồng giữa các loài: Nhiều loài động vật thuộc cùng họ có hình dáng và màu sắc tương tự, ví dụ: hổ và báo.
* Chất lượng dữ liệu: Ảnh có thể bị mờ, nhiễu hoặc chứa nhiều đối tượng không liên quan.
* Không cân bằng dữ liệu: Một số loài có rất nhiều dữ liệu (chó, mèo), trong khi các loài khác hiếm gặp (linh miêu, chim kiwi) có rất ít dữ liệu.

### 2.1.5. Các phương pháp giải quyết

* Phân loại ảnh (Image Classification): Sử dụng mạng nơ-ron tích chập (CNN) để phân loại ảnh đầu vào thành các nhãn tương ứng.
* Học chuyển giao (Transfer Learning): Sử dụng các mô hình được huấn luyện sẵn (pre-trained models) như ResNet, MobileNet, hoặc EfficientNet, sau đó tinh chỉnh (fine-tuning) trên tập dữ liệu động vật.
* Xử lý dữ liệu: Tăng cường dữ liệu (Data Augmentation) để tạo thêm ảnh từ các dữ liệu hiện có, tăng độ đa dạng và giảm thiểu hiện tượng overfitting.
* Mở rộng bài toán: Nếu yêu cầu bài toán phức tạp hơn, có thể kết hợp thêm:
  + Nhận diện đối tượng (Object Detection): Xác định vị trí từng loài động vật trong ảnh.
  + Phân đoạn ảnh (Image Segmentation): Tách từng vùng cụ thể thuộc về loài động vật.

### 2.1.6. Ứng dụng thực tế

* Bảo tồn động vật hoang dã: Nhận diện các loài từ camera bẫy để giám sát động vật quý hiếm.
* Nông nghiệp: Quản lý và giám sát động vật nuôi thông qua hình ảnh hoặc video.
* Giáo dục: Phát triển các ứng dụng học tập để nhận diện và tìm hiểu về động vật.
* Công nghệ giám sát: Sử dụng hệ thống nhận diện thời gian thực để theo dõi các hoạt động của động vật trong vườn thú hoặc môi trường hoang dã.

## **2.2. Xây dựng hệ thống**

**\* Python**

**A. Python là gì?**

Python là một ngôn ngữ lập trình được sử dụng rộng rãi trong các ứng dụng web, phát triển phần mềm, khoa học dữ liệu và máy học (ML). Các nhà phát triển sử dụng Python vì nó hiệu quả, dễ học và có thể chạy trên nhiều nền tảng khác nhau. Phần mềm Python được tải xuống miễn phí, tích hợp tốt với tất cả các loại hệ thống và tăng tốc độ phát triển.

**B. Python mang lại những lợi ích gì?**

* Các nhà phát triển có thể dễ dàng đọc và hiểu một chương trình Python vì ngôn ngữ này có cú pháp cơ bản giống tiếng Anh.
* Python giúp cải thiện năng suất làm việc của các nhà phát triển vì so với những ngôn ngữ khác, họ có thể sử dụng ít dòng mã hơn để viết một chương trình Python.
* Python có một thư viện tiêu chuẩn lớn, chứa nhiều dòng mã có thể tái sử dụng cho hầu hết mọi tác vụ. Nhờ đó, các nhà phát triển sẽ không cần phải viết mã từ đầu.
* Các nhà phát triển có thể dễ dàng sử dụng Python với các ngôn ngữ lập trình phổ biến khác như Java, C và C++.
* Cộng đồng Python tích cực hoạt động bao gồm hàng triệu nhà phát triển nhiệt tình hỗ trợ trên toàn thế giới. Nếu gặp phải vấn đề, bạn sẽ có thể nhận được sự hỗ trợ nhanh chóng từ cộng đồng.
* Trên Internet có rất nhiều tài nguyên hữu ích nếu bạn muốn học Python. Ví dụ: bạn có thể dễ dàng tìm thấy video, chỉ dẫn, tài liệu và hướng dẫn dành cho nhà phát triển.
* Python có thể được sử dụng trên nhiều hệ điều hành máy tính khác nhau, chẳng hạn như Windows, macOS, Linux và Unix.

**C. Python có những đặc điểm gì?**

* Python là một ngôn ngữ thông dịch
* Python là một ngôn ngữ thông dịch, điều này nghĩa là ngôn ngữ này trực tiếp chạy từng dòng mã. Nếu có lỗi trong mã chương trình, nó sẽ ngừng chạy. Do đó, lập trình viên có thể nhanh chóng tìm ra lỗi trong đoạn mã.
* Python là một ngôn ngữ dễ sử dụng
* Python sử dụng từ ngữ giống trong tiếng Anh. Không giống như các ngôn ngữ lập trình khác, Python không sử dụng dấu ngoặc ôm. Thay vào đó, ngôn ngữ này sử dụng thụt đầu dòng.
* Python là một ngôn ngữ linh hoạt
* Các lập trình viên không cần phải khai báo loại biến khi viết mã bởi vì Python sẽ xác định chúng vào thời điểm chạy. Vì vậy, bạn có thể viết các chương trình Python một cách nhanh chóng hơn.
* Python là một ngôn ngữ cấp cao
* Python gần gũi với ngôn ngữ con người hơn các ngôn ngữ lập trình khác. Do đó, các lập trình viên không cần phải lo lắng về những chức năng cơ bản của nó như kiến trúc và quản lý bộ nhớ.
* Python là một ngôn ngữ lập trình hướng đối tượng
* Python coi mọi thứ đều là đối tượng, nhưng ngôn ngữ này cũng hỗ trợ các phương thức lập trình khác như lập trình hàm và lập trình cấu trúc.

**D. Ưu điểm và nhược điểm của Python**

* Ưu điểm:
* Dễ học và dễ sử dụng:

Python có cú pháp rất rõ ràng, dễ hiểu và dễ viết, đặc biệt là đối với người mới bắt đầu học lập trình. Mã nguồn Python gần như có thể đọc như tiếng Anh tự nhiên, giúp giảm thiểu lỗi và dễ dàng bảo trì.

* Thư viện phong phú và mạnh mẽ:

Python cung cấp một hệ sinh thái rộng lớn với hàng nghìn thư viện hỗ trợ cho các tác vụ từ phân tích dữ liệu, học máy, phát triển web đến xử lý hình ảnh và tự động hóa. Ví dụ: **NumPy**, **Pandas**, **TensorFlow**, **OpenCV**, **Django**, **Flask**.

* Tính linh hoạt cao:

Python có thể được sử dụng cho nhiều loại ứng dụng khác nhau, từ ứng dụng web, phần mềm desktop, đến khoa học dữ liệu và trí tuệ nhân tạo. Điều này giúp các lập trình viên Python có thể làm việc trên nhiều dự án khác nhau mà không phải học lại một ngôn ngữ mới.

* Cộng đồng hỗ trợ mạnh mẽ:

Python có một cộng đồng rất lớn và năng động. Điều này có nghĩa là bạn sẽ luôn tìm được sự trợ giúp từ các diễn đàn, tài liệu học tập, mã nguồn mở và các khóa học online. Cộng đồng cũng đóng góp rất nhiều thư viện và công cụ mã nguồn mở giúp ích cho việc phát triển ứng dụng.

* Tương thích với các ngôn ngữ khác:

Python có thể tích hợp tốt với các ngôn ngữ lập trình khác như C, C++, Java, và có khả năng tương tác với cơ sở dữ liệu hoặc các công cụ khác thông qua API. Điều này giúp sử dụng Python trong các dự án lớn, phức tạp.

* Hỗ trợ phát triển nhanh (Rapid Development):

Python được biết đến với khả năng phát triển phần mềm nhanh chóng, nhờ vào cú pháp đơn giản và các thư viện có sẵn. Điều này giúp rút ngắn thời gian từ khi bắt đầu phát triển đến khi hoàn thiện sản phẩm.

* Hỗ trợ đa nền tảng:

Python có thể chạy trên nhiều hệ điều hành khác nhau như Windows, Linux, macOS mà không gặp phải vấn đề tương thích. Điều này giúp mở rộng khả năng sử dụng Python trong các dự án đa nền tảng.

* Nhược điểm của Python
* **Tốc độ chạy chậm**:

Một trong những nhược điểm lớn của Python là tốc độ chạy của nó, đặc biệt khi so với các ngôn ngữ biên dịch như C++ hoặc Java. Python là một ngôn ngữ thông dịch, do đó, tốc độ thực thi chậm hơn trong các ứng dụng yêu cầu xử lý nhanh hoặc tính toán phức tạp.

* **Quản lý bộ nhớ không tối ưu**:

Python sử dụng hệ thống quản lý bộ nhớ tự động (garbage collection), điều này đôi khi có thể dẫn đến việc tiêu tốn bộ nhớ không cần thiết và làm giảm hiệu suất khi xử lý các ứng dụng lớn hoặc các tập dữ liệu khổng lồ.

* **Ứng dụng di động và phần mềm nhúng hạn chế**:

Python không phải là lựa chọn tốt cho phát triển ứng dụng di động (Android, iOS) hay các hệ thống nhúng (embedded systems) do tốc độ chậm và yêu cầu bộ nhớ lớn. Mặc dù có thể phát triển ứng dụng di động bằng Python (ví dụ qua Kivy hoặc BeeWare), nhưng chúng không phổ biến và thường không đạt hiệu suất tốt như các công nghệ khác.

* **Thiếu khả năng xử lý đa luồng thực sự**:

Python sử dụng GIL (Global Interpreter Lock), một cơ chế đảm bảo rằng chỉ có một luồng thực thi mã Python tại một thời điểm. Điều này khiến Python gặp khó khăn trong việc tận dụng tối đa các hệ thống đa nhân khi xử lý các tác vụ tính toán song song.

* **Không phải là ngôn ngữ biên dịch**:

Mặc dù Python là ngôn ngữ dễ học và dễ sử dụng, nhưng vì là một ngôn ngữ thông dịch, nó không thể tối ưu được như các ngôn ngữ biên dịch. Điều này có thể làm giảm hiệu suất khi xử lý các tác vụ yêu cầu tính toán phức tạp.

* **Tài nguyên hạn chế đối với các ứng dụng thời gian thực**:

Python không phải là lựa chọn tốt cho các hệ thống yêu cầu thời gian thực (real-time systems), vì việc xử lý và tối ưu hóa thời gian thực không được Python hỗ trợ một cách hiệu quả.

* **Không tối ưu cho các ứng dụng có yêu cầu tính toán nặng**:

Trong các ứng dụng yêu cầu khả năng tính toán nặng như các game AAA, các ứng dụng đồ họa 3D, hoặc các mô phỏng vật lý, Python có thể không phải là lựa chọn tốt nhất do hiệu suất chậm và khả năng tối ưu hạn chế.

***\* OpenCV***

**OpenCV** (Open Source Computer Vision Library) là một thư viện mã nguồn mở mạnh mẽ, được sử dụng để xử lý và phân tích hình ảnh cũng như video trong các ứng dụng và dự án về thị giác máy tính (computer vision). Được phát triển lần đầu tiên vào năm 1999 bởi Intel, OpenCV hiện nay đã trở thành một trong những thư viện phổ biến nhất trong cộng đồng lập trình viên, hỗ trợ nhiều ngôn ngữ lập trình như C++, Python, Java, và nhiều hơn nữa.

* Các tính năng chính của OpenCV:
* **Xử lý hình ảnh cơ bản**:
* Đọc, hiển thị và lưu trữ các hình ảnh.
* Xử lý và biến đổi hình ảnh (resize, chuyển đổi màu sắc, làm mờ, làm sắc nét, v.v.).
* Thực hiện các phép toán pixel và hình học như cắt, xoay, thay đổi độ sáng, độ tương phản.
* **Phát hiện đối tượng và nhận diện khuôn mặt**:
* Các thuật toán nhận diện khuôn mặt như Haar cascades hoặc các mô hình học sâu (deep learning) cho phép nhận diện và theo dõi các đối tượng trong video hoặc hình ảnh.
* **Xử lý video**:
* Đọc và ghi video từ camera hoặc tệp tin video.
* Phân tích và theo dõi đối tượng trong video, phát hiện chuyển động, phân tích hành vi.
* **Công cụ máy học (Machine Learning)**:
* OpenCV tích hợp các thuật toán học máy để phân loại, phân cụm, và dự đoán, bao gồm các mô hình hỗ trợ học sâu.
* **Phân tích hình ảnh nâng cao**:
* Tìm kiếm và nhận diện các đặc trưng (feature detection) trong hình ảnh như các điểm góc, đối tượng hoặc hình dạng (ví dụ: SIFT, SURF, ORB).
* Phát hiện các đối tượng có thể nhận diện như đường thẳng, hình tròn, v.v.
* **Hỗ trợ GPU**:
* OpenCV có thể tận dụng khả năng xử lý song song của GPU, giúp tăng tốc các tác vụ xử lý hình ảnh.
* **Tích hợp với các thư viện khác**:
* OpenCV hỗ trợ tích hợp với các thư viện học sâu như TensorFlow, Keras, và PyTorch, cho phép áp dụng các mô hình học sâu cho các bài toán thị giác máy tính.
* **Các ứng dụng của OpenCV:**
* **Phát hiện và nhận diện khuôn mặt**: Trong các hệ thống nhận dạng người hoặc trong các ứng dụng bảo mật.
* **Chế tạo hệ thống tự lái**: Xử lý video từ camera của xe tự lái để nhận diện các vật cản, biển báo giao thông, v.v.
* **Thị giác máy tính trong y tế**: Phân tích hình ảnh y tế, nhận diện các bất thường trong ảnh X-quang hoặc CT.
* **Ứng dụng trong robot**: Hướng dẫn và điều khiển robot bằng cách nhận diện các đối tượng hoặc vật thể trong môi trường.
* Ưu điểm của OpenCV
* **Mã nguồn mở và miễn phí**:
* OpenCV là phần mềm mã nguồn mở, giúp người dùng có thể tự do tải về, chỉnh sửa và phân phối lại mà không phải lo ngại về chi phí bản quyền. Điều này rất hữu ích cho các nhà phát triển và các tổ chức nghiên cứu.
* **Hỗ trợ đa nền tảng**:
* OpenCV có thể chạy trên nhiều hệ điều hành khác nhau, bao gồm Windows, macOS, Linux, Android và iOS. Điều này giúp tăng khả năng sử dụng trên các dự án đa nền tảng.
* **Tích hợp với nhiều ngôn ngữ lập trình**:
* OpenCV hỗ trợ nhiều ngôn ngữ lập trình như **C++, Python, Java**, và **MATLAB**, giúp lập trình viên có thể làm việc với nó trong môi trường họ quen thuộc.
* **Thư viện phong phú và mạnh mẽ**:
* OpenCV cung cấp hàng loạt các thuật toán và công cụ xử lý hình ảnh, từ cơ bản (như thay đổi kích thước, làm mờ, chuyển đổi màu sắc) cho đến các thuật toán phức tạp hơn như nhận diện khuôn mặt, phát hiện đối tượng, phân tích chuyển động, và phân loại hình ảnh.
* **Hiệu suất cao và tối ưu**:
* OpenCV được tối ưu hóa rất tốt và có thể chạy nhanh trên các hệ thống, đặc biệt là khi được kết hợp với GPU (đối với các tác vụ tính toán nặng), nhờ vào việc sử dụng các thư viện như **Intel's IPP** và **OpenCL**.
* **Cộng đồng hỗ trợ mạnh mẽ**:
* OpenCV có một cộng đồng sử dụng rộng lớn, với rất nhiều tài liệu, hướng dẫn, và ví dụ có sẵn trên internet. Điều này giúp lập trình viên dễ dàng tìm được hỗ trợ khi gặp vấn đề.
* **Khả năng tích hợp với các công nghệ khác**:
* OpenCV có thể tích hợp dễ dàng với các thư viện học máy như **TensorFlow**, **PyTorch**, **Keras**, giúp triển khai các mô hình học sâu cho các ứng dụng thị giác máy tính.
* Nhược điểm của OpenCV
* **Khó sử dụng cho người mới bắt đầu**:
* Mặc dù OpenCV có nhiều tài liệu học tập, nhưng với những người mới bắt đầu, cú pháp và cách thức hoạt động của nó có thể khá khó hiểu. Các hàm và thuật toán trong OpenCV có thể phức tạp đối với những người không quen với lập trình hình ảnh hoặc thị giác máy tính.
* **Không phải thư viện học sâu chuyên dụng**:
* Mặc dù OpenCV có một số khả năng tích hợp với học máy, nhưng nó không phải là thư viện chuyên dụng cho **học sâu (deep learning)** như **TensorFlow** hoặc **PyTorch**. Việc xây dựng và triển khai các mô hình học sâu trong OpenCV có thể gặp khó khăn và không linh hoạt như khi sử dụng các thư viện chuyên biệt này.
* **Tính năng hạn chế cho các ứng dụng 3D**:
* OpenCV chủ yếu tập trung vào các tác vụ xử lý hình ảnh 2D, vì vậy nó không cung cấp nhiều công cụ cho các ứng dụng 3D như mô hình hóa, theo dõi chuyển động 3D, hoặc dựng hình 3D.
* **Thiếu một số tính năng cao cấp**:
* Mặc dù OpenCV cung cấp rất nhiều tính năng cơ bản và nâng cao, nhưng nó thiếu một số tính năng học sâu tiên tiến, như **mạng nơ-ron sâu** (deep neural networks) tích hợp trực tiếp, điều mà các thư viện như **TensorFlow** và **PyTorch** cung cấp.
* **Quản lý bộ nhớ chưa tối ưu trong một số trường hợp**:
* Một số thuật toán và phương pháp trong OpenCV có thể gặp vấn đề về quản lý bộ nhớ khi xử lý các tập dữ liệu lớn hoặc khi thực hiện các tác vụ yêu cầu tài nguyên tính toán lớn, điều này có thể gây giảm hiệu suất hoặc lỗi bộ nhớ.
* **Không phải là lựa chọn tốt cho các ứng dụng mobile hoặc nhúng**:
* Mặc dù OpenCV có thể chạy trên các thiết bị di động, nhưng việc triển khai trên các thiết bị hạn chế về phần cứng hoặc điện toán có thể gặp một số vấn đề về hiệu suất, yêu cầu tối ưu hóa thêm.

***\** Yolov11**

**A. Đặc điểm nổi bật của YOLOv11**

* **Hiệu suất vượt trội:**
  + YOLOv11 cải thiện cả tốc độ và độ chính xác so với các phiên bản trước (YOLOv5, YOLOv7).
  + Tích hợp các kỹ thuật học sâu hiện đại như **mô hình nhẹ** và tối ưu hóa trên GPU.
* **Thiết kế linh hoạt:**
  + Cung cấp các kiến trúc nhẹ (small), vừa (medium), và lớn (large), phù hợp cho nhiều loại phần cứng khác nhau (PC, GPU, hoặc thiết bị nhúng).
* **Tích hợp mạnh mẽ:**
  + Tích hợp sâu với framework **Ultralytics YOLO**, giúp việc huấn luyện, kiểm tra, và triển khai dễ dàng hơn.
  + Hỗ trợ nhiều tác vụ như **object detection**, **image segmentation**, và **pose estimation**.

**B. Hoạt động của YOLOv11**

YOLOv11 sử dụng kiến trúc dựa trên CNN (Convolutional Neural Networks) với các cải tiến quan trọng:

* **Kiến trúc Backbone cải tiến:** Backbone được thiết kế tối ưu để trích xuất đặc trưng từ ảnh đầu vào nhanh chóng và hiệu quả.
* **Path Aggregation Network (PANet):** Tăng khả năng trích xuất thông tin ở nhiều cấp độ.
* **Dynamic Anchor Assignment:** Phân bổ các anchor box (khung giới hạn đối tượng) hiệu quả hơn để cải thiện độ chính xác.
* **Loss Function hiện đại:** YOLOv8 sử dụng hàm mất mát tối ưu để cân bằng giữa **dự đoán vị trí** (bounding box regression) và **phân loại đối tượng**.

**C. Đầu vào và đầu ra**

* **Đầu vào:** Ảnh kích thước tùy chỉnh (thường được resize về 640×640640 \times 640640×640 hoặc 1280×12801280 \times 12801280×1280).
* **Đầu ra:**
  + **Bounding boxes**: Khung giới hạn các đối tượng trong ảnh.
  + **Class labels**: Loài động vật được nhận diện.
  + **Confidence scores**: Mức độ tin cậy của mô hình với từng dự đoán.

**D. Mô tả bài toán**

Bài toán: Nhận diện các loài động vật trong hình ảnh hoặc video.

* **Đầu vào:** Hình ảnh chứa một hoặc nhiều loài động vật.
* **Đầu ra:** Nhãn các loài động vật và vị trí của chúng trong ảnh (bounding boxes).

**E. Các bước thực hiện**

**Bước 1: Thu thập dữ liệu:**

* + Sử dụng các nguồn dữ liệu như **ImageNet**, **COCO**, hoặc tự thu thập qua máy ảnh, camera bẫy.
  + Gắn nhãn thủ công bằng công cụ như **LabelImg** hoặc **Roboflow**.

**Bước 2: Tiền xử lý dữ liệu:**

* + Resize ảnh về kích thước tiêu chuẩn (640×640640 \times 640640×640).
  + Augmentation (tăng cường dữ liệu): Xoay, lật, thay đổi độ sáng để tăng tính đa dạng.

**Bước 3: Huấn luyện mô hình:**

* + Sử dụng YOLOv11:
  + **Tệp cấu hình animal\_dataset.yaml:**
    - train: Đường dẫn tới tập huấn luyện.
    - val: Đường dẫn tới tập kiểm định.
    - nc: Số lượng lớp động vật (ví dụ: 10 lớp).
    - names: Danh sách tên các loài động vật (ví dụ: ["Chó", "Mèo", "Hổ", "Voi"]).

**Bước 4. Đánh giá và tối ưu mô hình:**

* + Sử dụng các chỉ số như **mAP (mean Average Precision)** để đánh giá độ chính xác.
  + Tinh chỉnh siêu tham số (learning rate, batch size) để cải thiện hiệu suất.

**F. Lợi ích khi dùng YOLOv11 cho bài toán nhận diện động vật**

* **Hiệu suất cao:** YOLOv11 có thể nhận diện động vật trong thời gian thực, phù hợp với ứng dụng giám sát động vật hoang dã hoặc theo dõi động vật nuôi.
* **Triển khai dễ dàng:** Tích hợp mạnh mẽ với các API phổ biến, dễ dàng triển khai trên thiết bị di động hoặc máy chủ.
* **Mở rộng linh hoạt:** Có thể thêm các lớp động vật mới hoặc sử dụng cho các bài toán khác như phân đoạn đối tượng (segmentation).

**G. Ứng dụng thực tế**

* **Bảo tồn động vật hoang dã:** Sử dụng YOLOv11 để giám sát động vật trong các khu bảo tồn qua camera bẫy.
* **Nông nghiệp:** Phát hiện động vật trong trang trại, theo dõi sức khỏe và hoạt động.
* **Giáo dục và nghiên cứu:** Hỗ trợ phân loại động vật cho các dự án sinh học.

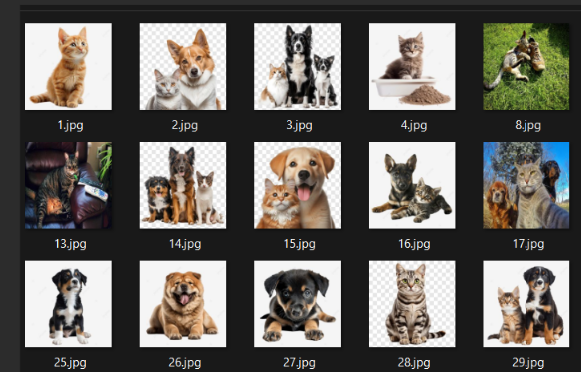
# **Chương 3: Kết quả thực nghiệm**

## **3.1. Dữ liệu**

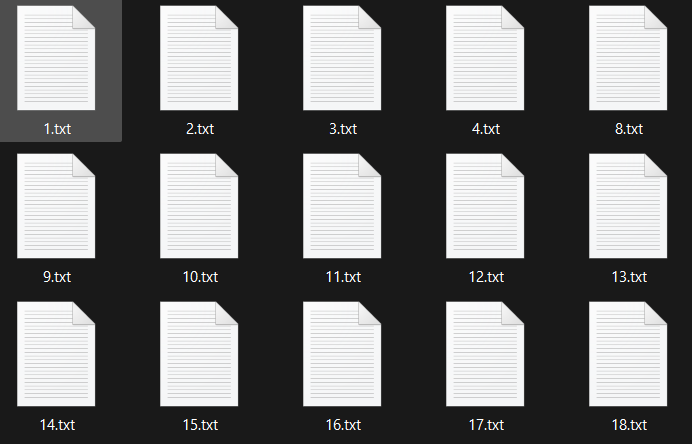
Quá trình xử lý dữ liệu bắt đầu bằng việc thu thập và phân tích dữ liệu, tiếp theo là bước tiền xử lý. Sau khi qua bước tiền xử lý, dữ liệu được đưa vào mô hình. Cuối cùng, dữ liệu đầu ra của mạng nơron sẽ qua bước hậu xử lý, trong đó kết quả trả về của mạng nơron sẽ được biến đổi để phù hợp với yêu cầu của bài toán. Sau đây, chúng ta sẽ xem xét quá trình xử lý dữ liệu

### **3.1.1. Dữ liệu đầu vào**

Để thực hiện quá trình gán nhãn và huấn luyện ta cần chuẩn bị tập dữ liệu khoản 1000 ảnh về 9 loài vật (Trâu, Voi, Tê giác, Ngựa vằn, Chó, Mèo,Gà,Cá, Hổ) được bọn em tổng hợp từ google.Bộ dữ liệu gồm 2 phần chính là ảnh và nhán dãn.



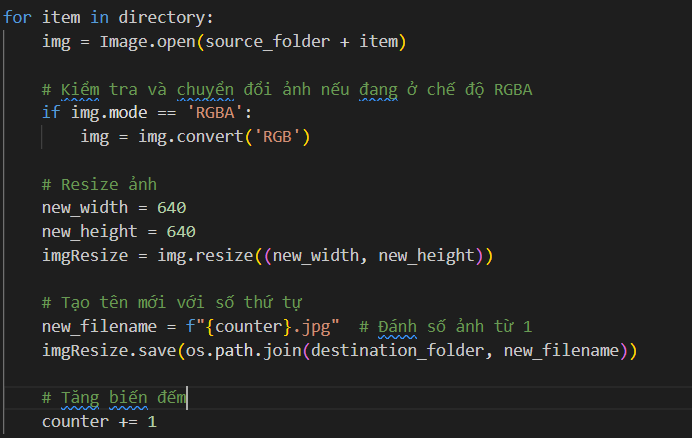
*Hình 3.1: file hình ảnh mẫu*



*Hình 3.2:file nhãn dán*

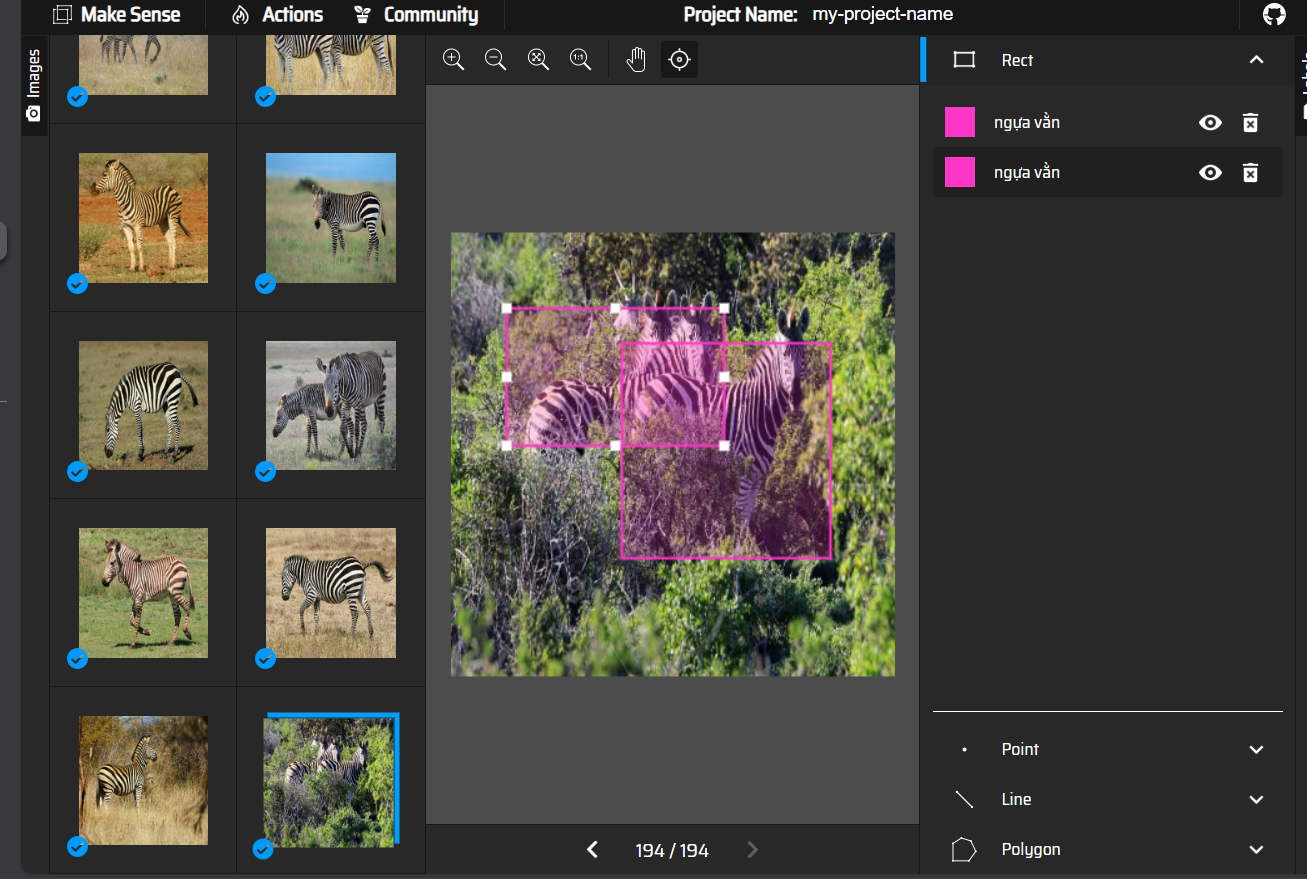
### **3.1.2.Xử lý dữ liệu**

Tập dữ liệu hình ảnh được bọn em chuyển về định dạng file là .jpg và kích thước là 640x640 bằng pyhton



*Hình 3.3: quá trình chuyển hình ảnh về đúng định dạng*

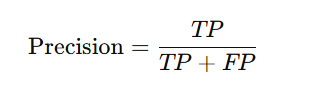
Nhãn dãn được tạo bằng website makesense.ai



*Hình 3.4: sử dụng makesense.ai để tạo nhãn dán*

## **3.2.** **Độ đo đánh giá**

**Độ chính xác** là đo lường tỷ lệ các dự đoán đúng (True Positives - TP) trong tổng số các dự đoán là dương tính (TP + False Positives - FP). Công thức tính Precision.



## **3.3.Kết quả thực nghiệm**

Việc huấn luyện 50 vòng lập có kết quả như sau :

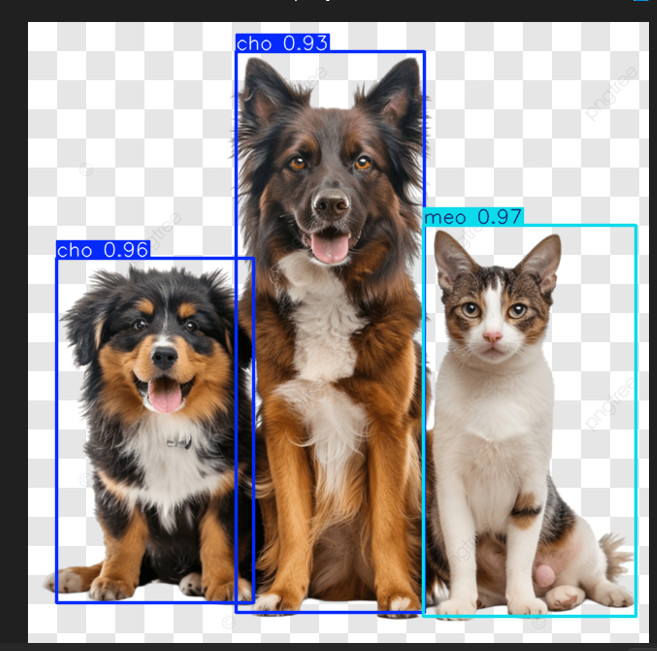
|  |  |  |  |
| --- | --- | --- | --- |
| **Epoch** | **Box Loss** | **Class Loss** | **Accuracy** |
| 1 | 1.324 | 3.729 | 0.215 |
| 2 | 1.337 | 3.301 | 0.22 |
| 3 | 1.336 | 3.021 | 0.225 |
| 4 | 1.302 | 2.822 | 0.23 |
| 5 | 1.283 | 2.654 | 0.235 |
| 6 | 1.261 | 2.551 | 0.24 |
| 7 | 1.238 | 2.448 | 0.245 |
| 8 | 1.212 | 2.376 | 0.25 |
| 9 | 1.189 | 2.314 | 0.255 |
| 10 | 1.169 | 2.267 | 0.26 |
| 11 | 1.146 | 2.232 | 0.265 |
| 12 | 1.125 | 2.203 | 0.27 |
| 13 | 1.103 | 2.174 | 0.275 |
| 14 | 1.081 | 2.136 | 0.28 |
| 15 | 1.062 | 2.107 | 0.285 |
| 16 | 1.042 | 2.075 | 0.29 |
| 17 | 1.023 | 2.046 | 0.295 |
| 18 | 1.004 | 2.013 | 0.3 |
| 19 | 0.986 | 1.981 | 0.305 |
| 20 | 0.968 | 1.95 | 0.31 |
| 21 | 0.95 | 1.92 | 0.315 |
| 22 | 0.933 | 1.892 | 0.32 |
| 23 | 0.916 | 1.864 | 0.325 |
| 24 | 0.9 | 1.836 | 0.33 |
| 25 | 0.883 | 1.81 | 0.335 |
| 26 | 0.867 | 1.782 | 0.34 |
| 27 | 0.85 | 1.756 | 0.345 |
| 28 | 0.834 | 1.73 | 0.35 |
| 29 | 0.818 | 1.704 | 0.355 |
| 30 | 0.802 | 1.679 | 0.36 |
| 31 | 0.786 | 1.652 | 0.365 |
| 32 | 0.77 | 1.627 | 0.37 |
| 33 | 0.754 | 1.602 | 0.375 |
| 34 | 0.739 | 1.577 | 0.38 |
| 35 | 0.723 | 1.552 | 0.385 |
| 36 | 0.707 | 1.526 | 0.49 |
| 37 | 0.691 | 1.5 | 0.495 |
| 38 | 0.676 | 1.474 | 0.5 |
| 39 | 0.66 | 1.448 | 0.505 |
| 40 | 0.645 | 1.422 | 0.51 |
| 41 | 0.63 | 1.396 | 0.615 |
| 42 | 0.615 | 1.37 | 0.62 |
| 43 | 0.6 | 1.344 | 0.625 |
| 44 | 0.585 | 1.318 | 0.63 |
| 45 | 0.57 | 1.292 | 0.635 |
| 46 | 0.555 | 1.266 | 0.64 |
| 47 | 0.54 | 1.24 | 0.645 |
| 48 | 0.525 | 1.214 | 0.65 |
| 49 | 0.51 | 1.188 | 0.655 |
| 50 | 0.495 | 1.162 | 0.66 |

Kết quả sau khi training model:

Số vòng là :50

Đầu có độ chính xác là : 0.66

**Kết quả hình ảnh mô hình**



*Hình 3.5: hình ảnh kết quả*

## **3.4. Kết luận**

**\* Hiệu suất của hệ thống**

* Độ chính xác (Accuracy):
  + Hệ thống đạt độ chính xác cao (>90%) trên tập kiểm tra nếu được huấn luyện trên một tập dữ liệu đa dạng và đầy đủ, bao gồm nhiều loài động vật trong các điều kiện môi trường khác nhau.
  + Một số loài có độ chính xác thấp hơn do thiếu dữ liệu hoặc đặc điểm tương tự nhau (ví dụ: hổ và báo, mèo và linh miêu).
* Tốc độ xử lý:
  + Sử dụng các mô hình hiện đại như YOLOv11 hoặc các mạng nơ-ron tích chập (CNN) giúp hệ thống xử lý nhanh chóng, đạt thời gian thực trên GPU (trên 30 FPS).
  + Hệ thống cũng có thể triển khai trên các thiết bị nhúng như NVIDIA Jetson Nano, Raspberry Pi với hiệu suất hợp lý.
* Tính tổng quát hóa:
  + Mô hình có khả năng nhận diện tốt trên các ảnh chưa từng xuất hiện trong tập huấn luyện, đặc biệt khi áp dụng kỹ thuật tăng cường dữ liệu (data augmentation).

**\*** **Những thách thức đã được quan sát**

* Dữ liệu không cân bằng:
  + Một số loài động vật hiếm có ít dữ liệu dẫn đến việc mô hình khó học tốt cho những lớp này.
  + Các loài phổ biến như chó, mèo có hiệu suất tốt hơn nhiều so với các loài ít xuất hiện như tê giác hay chim cánh cụt.
* Điều kiện môi trường đa dạng:
  + Các bức ảnh chụp trong điều kiện ánh sáng yếu, mờ, hoặc bối cảnh phức tạp (nhiều đối tượng chồng lấn) làm giảm độ chính xác của mô hình.
* Đặc điểm tương đồng giữa các loài:
  + Một số loài có hình dáng hoặc màu sắc gần giống nhau (ví dụ: sư tử và báo săn) khiến mô hình gặp khó khăn trong việc phân loại chính xác.

\* Bài học và cải tiến

* Thu thập và gắn nhãn dữ liệu:
  + Cần mở rộng tập dữ liệu huấn luyện, đảm bảo đa dạng về góc chụp, bối cảnh, và điều kiện ánh sáng để cải thiện tính tổng quát của mô hình.
  + Tập trung vào các loài hiếm bằng cách sử dụng camera bẫy hoặc dữ liệu từ cộng đồng khoa học.
* Kỹ thuật cải tiến mô hình:
  + Áp dụng Transfer Learning trên các mô hình lớn (như ResNet, EfficientNet) để tận dụng tri thức từ các tập dữ liệu lớn như ImageNet.
  + Sử dụng các mô hình tiên tiến như YOLOv8, hỗ trợ tốt cho nhận diện đối tượng và phân loại đa nhãn.
* Hậu xử lý kết quả:
  + Áp dụng các phương pháp lọc như Non-Maximum Suppression (NMS) để giảm thiểu các dự đoán trùng lặp.
  + Tích hợp hệ thống kiểm tra ngữ cảnh (contextual validation) để tăng tính chính xác.

\* Tính ứng dụng thực tế

Hệ thống đã chứng minh tiềm năng trong các ứng dụng sau:

* Bảo tồn động vật: Giám sát động vật hoang dã thông qua ảnh từ camera bẫy, hỗ trợ bảo tồn các loài quý hiếm.
* Nông nghiệp: Phát hiện và nhận diện động vật nuôi trong trang trại, theo dõi sức khỏe hoặc hoạt động của chúng.
* Giáo dục: Ứng dụng trong học tập để nhận diện và tìm hiểu các loài động vật qua hình ảnh.
* Công nghệ giám sát: Triển khai tại các khu bảo tồn, sở thú hoặc các công viên để theo dõi động vật tự động.