**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 15: XÂY DỰNG HỆ THỐNG PHÁT TRIỂN MÔ HÌNH PHÂN LOẠI ẢNH ĐỂ NHẬN DIỆN CÁC LOÀI ĐỘNG VẬT**

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

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Mã sinh viên** | **Sinh viên thực hiện** | **Lớp hành chính** |
| **1** | **20210594** | **Đỗ Hữu Phước** | **DCCNTT12.10.2** |
| **2** | **20210520** | **Đinh Như Thắng** | **DCCNTT12.10.2** |

**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 15: XÂY DỰNG HỆ THỐNG PHÁT TRIỂN MÔ HÌNH PHÂN LOẠI ẢNH ĐỂ NHẬN DIỆN CÁC LOÀI ĐỘNG VẬT**

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

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Mã sinh viên** | **Sinh viên thực hiện** | **Lớp hành chính** |
| **1** | **20210594** | **Đỗ Hữu Phước** | **DCCNTT12.10.2** |
| **2** | **20210520** | **Đinh Như Thắng** | **DCCNTT12.10.2** |

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

**Bắc Ninh, năm 2024**

**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: 15**  **Đề tài: XÂY DỰNG HỆ THỐNG PHÁT TRIỂN MÔ HÌNH PHÂN LOẠI ẢNH ĐỂ NHẬN DIỆN CÁC LOÀI ĐỘNG VẬT**  **Tên học phần: XỬ LÝ ẢNH VÀ THỊ GIÁC MÁY TÍNH**  **Lớp Tín chỉ: XATGMT.03.K12.02.LH.C04.1\_LT** | |
| **Cán bộ chấm thi 1**  **Lương Thị Hồng Lan** | **Cán bộ chấm thi 2**  *(Ký và ghi rõ họ tên)* |

| **TT** | **TIÊU CHÍ** | **THANG ĐIỂM** | **Đỗ Hữu Phước** | **Đinh Như Thắng** |
| --- | --- | --- | --- | --- |
| 20210594 | 20210520 |
| **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* |  |  |

# LỜI NÓI ĐẦU

Trong thời đại công nghệ 4.0, trí tuệ nhân tạo (AI) và học máy (Machine Learning) đang ngày càng khẳng định vai trò quan trọng trong nhiều lĩnh vực, từ y tế, giáo dục, đến môi trường và khoa học tự nhiên. Trong đó, bài toán phân loại ảnh, đặc biệt là nhận diện các loài động vật, đã và đang thu hút sự quan tâm lớn từ cộng đồng nghiên cứu và ứng dụng. Đây không chỉ là một thách thức về mặt công nghệ, mà còn mang ý nghĩa thực tiễn cao, góp phần hỗ trợ công tác bảo tồn động vật, quản lý tài nguyên thiên nhiên, và nghiên cứu sinh học.

Đề tài "Xây dựng hệ thống phát triển mô hình phân loại ảnh để nhận diện các loài động vật" nhằm mục tiêu thiết kế và triển khai một hệ thống hiệu quả, thân thiện với người dùng, giúp tự động phân loại và nhận diện các loài động vật từ hình ảnh. Đề tài không chỉ tập trung vào việc áp dụng các mô hình học sâu tiên tiến, mà còn hướng đến việc xây dựng một hệ thống có khả năng mở rộng, dễ dàng cập nhật và ứng dụng trong thực tế.

Với bối cảnh ứng dụng phong phú và tầm quan trọng trong việc bảo tồn đa dạng sinh học, việc thực hiện đề tài này sẽ góp phần nâng cao hiệu quả nghiên cứu khoa học, đồng thời mở ra cơ hội ứng dụng công nghệ hiện đại vào thực tiễn.

# LỜI CẢM ƠN

Nhóm chúng em xin gửi lời cảm ơn chân thành đến cô Lương Thị Hồng Lan, giảng viên dạy lớp em môn Xử lý ảnh và thị giác máy tính, đã tận tình chỉ dẫn và giúp đỡ chúng em trong quá trình làm bài tập lớn. Nhóm chúng em cũng xin cảm ơn các bạn sinh viên trong lớp đã góp ý và hỗ trợ chúng em.

Bài tập lớn này là kết quả của sự nỗ lực và cố gắng của nhóm chúng em. Tuy nhiên, do kiến thức và kinh nghiệm còn hạn chế, bài tập lớn không thể tránh khỏi những thiếu sót và sai sót. Nhóm chúng em rất mong nhận được những ý kiến đóng góp và góp ý của các thầy cô và các bạn để bài tập lớn được hoàn thiện hơn.

# LỜI CAM ĐOAN

Chúng em, nhóm tác giả của báo cáo này, cam đoan rằng tất cả thông tin và nghiên cứu trong báo cáo này là hoàn toàn trung thực và không gian lận hay đạo đức trong quá trình nghiên cứu và viết báo cáo. Các nguồn thông tin được sử dụng trong báo cáo này đã được dẫn chính xác theo quy định của trường và các nguồn tham khảo đã được liệt kê đầy đủ. Chúng em xác nhận rằng mọi phần công việc và bài viết trong báo cáo này đều được thực hiện bởi chúng tôi, ngoại trừ các phần được ghi chú rõ nguồn gốc. Chúng em không sao chép hoặc sử dụng công việc của người khác mà không có sự cho phép hoặc dẫn chính xác. Chúng em hiểu rằng việc vi phạm các nguyên tắc đạo đức và trí tuệ là không chấp nhận và sẽ có hậu quả pháp lý. Chúng em đảm bảo rằng tất cả các phần của báo cáo này đã được xem xét và chỉnh sửa một cách cẩn thận để đảm bảo tính trung thực và chất lượng. Chúng em chịu trách nhiệm về nội dung của báo cáo này và sẽ sẵn sàng chịu trách nhiệm trước bất kỳ cơ quan nào nếu có sự vi phạm nghiêm trọng đối với tính trung thực và đạo đức của báo cáo này.

# MỤC LỤC

[LỜI NÓI ĐẦU 1](#_Toc184666954)

[LỜI CẢM ƠN 2](#_Toc184666955)

[LỜI CAM ĐOAN 3](#_Toc184666956)

[MỤC LỤC 4](#_Toc184666957)

[DANH MỤC HÌNH ẢNH 5](#_Toc184666958)

[DANH MỤC TỪ VIẾT TẮT 6](#_Toc184666959)

[CHƯƠNG 1. TỔNG QUAN BÀI TOÁN 1](#_Toc184666960)

[1.1. Nhận dạng bài toán 1](#_Toc184666961)

[1.2. Ngôn ngữ lập trình và các thư viện 5](#_Toc184666962)

[CHƯƠNG 2. XÂY DỰNG HỆ THỐNG NHẬN DẠNG BÀI TOÁN 12](#_Toc184666963)

[2.1. Yêu cầu của bài toán 12](#_Toc184666964)

[2.2. Xây dựng hệ thống 13](#_Toc184666965)

[2.3. Các phương pháp áp dụng cho bài toán phân loại ảnh 16](#_Toc184666966)

[CHƯƠNG 3. THỰC NGHIỆM CHƯƠNG TRÌNH 30](#_Toc184666967)

[3.1. Dữ liệu 30](#_Toc184666968)

[3.2. Các độ đo so sánh 32](#_Toc184666969)

[3.3 Kết quả 33](#_Toc184666970)

[KẾT LUẬN 40](#_Toc184666971)

[TÀI LIỆU THAM KHẢO 43](#_Toc184666972)

# DANH MỤC HÌNH ẢNH

[Hình 2.2. Hình ảnh minh họa bài toán 28](#_Toc184455840)

[Hình 3.1. Biểu đồ so sánh số liệu 33](#_Toc184666947)

[Hình 3.2. Ma trận nhầm lẫn thể hiện hiệu suất phân loại của mô hình trên từng lớp dữ liệu 36](#_Toc184666948)

[Hình 3.3. Đồ thị loss và accuracy 38](#_Toc184666949)

[Hình 3.4. Test bằng hình ảnh 38](#_Toc184666950)

[Hình 3.5. Test phân loại ảnh loài mèo 39](#_Toc184666951)

# DANH MỤC TỪ VIẾT TẮT

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Từ viết tắt | Nghĩa tiếng anh | Nghĩa tiếng việt |
| 1 | SQL | Structured Query Language | Ngôn ngữ truy vấn có cấu trúc |
| 2 | URL | Uniform Resource Locator | Hệ thống định vị tài nguyên thống nhất |
| 3 | GUI | Graphical User Interface | Giao diện đồ họa người dùng |

# CHƯƠNG 1. TỔNG QUAN BÀI TOÁN

## 1.1. Nhận dạng bài toán

Đề tài "Xây dựng hệ thống phát triển mô hình phân loại ảnh để nhận diện các loài động vật" nhằm phát triển một hệ thống có khả năng phân loại các loài động vật từ hình ảnh, sử dụng các kỹ thuật học máy và thị giác máy tính. Mục tiêu chính của bài toán là xây dựng mô hình có thể nhận diện chính xác các loài động vật dựa trên ảnh đầu vào, từ đó hỗ trợ ứng dụng trong các lĩnh vực như nghiên cứu bảo tồn, giám sát động vật hoang dã và phát triển các công cụ giáo dục. Bài toán này đối mặt với các thách thức lớn như sự đa dạng và nhiễu trong dữ liệu ảnh, sự tương đồng giữa các loài động vật và yêu cầu về hiệu quả xử lý thời gian thực. Hệ thống sẽ sử dụng các mô hình học sâu tiên tiến như ResNet, EfficientNet hoặc Vision Transformer để phân loại ảnh, đồng thời áp dụng các kỹ thuật tăng cường dữ liệu để cải thiện độ chính xác. Kết quả mong đợi là một mô hình phân loại có độ chính xác cao, dễ dàng mở rộng cho nhiều loài động vật khác nhau và có thể triển khai vào các ứng dụng thực tế như giám sát tự động hoặc ứng dụng nhận diện động vật trong du lịch, bảo tồn.

#### 1.1.1. Mô tả bài toán

Đề tài "Xây dựng hệ thống phát triển mô hình phân loại ảnh để nhận diện các loài động vật" nhằm mục tiêu phát triển một hệ thống sử dụng các mô hình học sâu, đặc biệt là mô hình ResNet, để tự động nhận diện và phân loại các loài động vật từ hình ảnh. Hệ thống sẽ được huấn luyện trên một tập dữ liệu lớn chứa hình ảnh của nhiều loài động vật khác nhau, với mỗi ảnh được gán nhãn đúng loài. Quá trình xây dựng hệ thống bao gồm các bước thu thập và tiền xử lý dữ liệu, trong đó dữ liệu ảnh sẽ được chuẩn hóa kích thước, xử lý màu sắc và áp dụng các kỹ thuật tăng cường dữ liệu để giúp mô hình học được những đặc trưng quan trọng.

Mô hình ResNet, với cấu trúc mạng nơ-ron sâu và các khối residual, sẽ giúp giải quyết vấn đề vanishing gradient trong quá trình huấn luyện, từ đó cải thiện khả năng phân loại của mô hình. Hệ thống sẽ được huấn luyện bằng các phương pháp tối ưu hóa hiện đại như Adam và hàm mất mát cross-entropy, nhằm tối ưu hóa độ chính xác trong phân loại. Sau khi huấn luyện, mô hình sẽ được đánh giá với các chỉ số hiệu suất như độ chính xác, độ chính xác phân lớp và độ nhạy trên tập kiểm tra.

Cuối cùng, hệ thống sẽ được triển khai vào các ứng dụng thực tế, như nhận diện loài động vật trong các khu bảo tồn động vật hoang dã hoặc các tổ chức nghiên cứu sinh học, giúp hỗ trợ công tác bảo tồn và nghiên cứu động vật.

#### 1.1.2. Khó khăn và những vấn đề cần giải quyết

Những khó khăn:

Đa dạng và chất lượng của dữ liệu ảnh: Dữ liệu ảnh về các loài động vật có thể rất đa dạng, với sự thay đổi lớn về môi trường, ánh sáng, góc nhìn, và chất lượng ảnh. Ngoài ra, một số loài động vật có thể trông rất giống nhau, gây khó khăn cho việc phân biệt.

Khó khăn trong việc phân biệt các loài động vật tương tự nhau: Nhiều loài động vật có ngoại hình tương tự nhau, khiến mô hình dễ bị nhầm lẫn, đặc biệt là khi điều kiện hình ảnh không lý tưởng (như góc nhìn, ánh sáng kém, hoặc động vật bị khuất một phần).

Thiếu dữ liệu gán nhãn cho một số loài động vật: Việc có một bộ dữ liệu lớn và đầy đủ các loài động vật là điều khó khăn, đặc biệt đối với những loài hiếm hoặc ít được nghiên cứu.

Vấn đề về tộc độ và hiệu suất: Các mô hình học sâu yêu cầu tài nguyên tính toán lớn và thời gian huấn luyện dài, điều này có thể gây khó khăn trong việc triển khai hệ thống trong môi trường thời gian thực hoặc các thiết bị có tài nguyên hạn chế.

Tính khả dụng và bảo mật của hệ thống: Khi triển khai hệ thống nhận diện động vật trong các ứng dụng thực tế, việc đảm bảo tính khả dụng và bảo mật của hệ thống là một yếu tố quan trọng.

Xử lý các tình huống khó khăn trong môi trường thực tế: Trong môi trường thực tế, các yếu tố như ánh sáng thay đổi, các đối tượng bị khuất, hoặc sự di chuyển nhanh của động vật có thể làm giảm độ chính xác của mô hình.

Tính mở rộng của hệ thống: Khi mô hình cần nhận diện thêm nhiều loài động vật mới, việc mở rộng hệ thống trở thành một thách thức lớn.

Cách giải quyết:

Đa dạng và chất lượng của dữ liệu ảnh: Sử dụng kỹ thuật tăng cường dữ liệu (data augmentation) để tạo ra nhiều biến thể của các ảnh và giúp mô hình học tốt hơn từ các ví dụ ít ỏi.

Khó khăn trong việc phân biệt các loài động vật tương tự nhau: Sử dụng các mô hình học sâu (deep learning) phức tạp như Convolutional Neural Networks (CNN) hoặc Vision Transformers (ViTs) để tăng khả năng phân biệt chi tiết giữa các loài động vật. Áp dụng các phương pháp như fine-tuning và transfer learning từ các mô hình đã được huấn luyện trên bộ dữ liệu lớn và phong phú để cải thiện độ chính xác.

Thiếu dữ liệu gán nhãn cho một số loài động vật: Xây dựng hoặc tìm kiếm các bộ dữ liệu từ nhiều nguồn khác nhau (mạng xã hội, các dự án nghiên cứu, bảo tồn động vật). Sử dụng kỹ thuật học bán giám sát (semi-supervised learning) hoặc học không giám sát (unsupervised learning) để cải thiện khả năng học từ dữ liệu không gán nhãn hoặc ít nhãn.

vấn đề về tốc độ và hiệu suất: Tối ưu hóa mô hình sao cho nó có thể chạy hiệu quả trên các thiết bị tính toán với khả năng xử lý giới hạn (ví dụ, sử dụng các mô hình nhẹ như MobileNet hoặc EfficientNet). Sử dụng kỹ thuật giảm độ phức tạp của mô hình (model compression) hoặc truyền tải mô hình qua các công cụ như TensorFlow Lite cho các ứng dụng di động.

Tính khả dụng và bảo mật của hệ thống: Xây dựng hệ thống với giao diện dễ sử dụng và khả năng vận hành ổn định trong nhiều môi trường khác nhau (ví dụ, trong các khu bảo tồn động vật, vườn thú, v.v.). Đảm bảo rằng hệ thống bảo mật và có thể được cập nhật, bảo trì khi cần thiết để duy trì hiệu quả trong thời gian dài.

Xử lý các tình huống khó khăn trong môi trường thực tế: Sử dụng các phương pháp học sâu mạnh mẽ kết hợp với các kỹ thuật như tracking để theo dõi động vật qua nhiều khung hình và khắc phục các vấn đề do việc di chuyển hoặc bị khuất. Tăng cường bộ dữ liệu bằng các hình ảnh có tính chất khác biệt (ánh sáng yếu, góc nhìn nghiêng, động vật ở xa) để mô hình có thể học và nhận diện trong mọi điều kiện.

Tính mở rộng của hệ thống: Xây dựng hệ thống có khả năng mở rộng linh hoạt, với khả năng cập nhật và huấn luyện lại mô hình với các lớp mới mà không cần huấn luyện lại toàn bộ hệ thống từ đầu. Tích hợp khả năng tự động thêm các loài mới vào hệ thống qua các phương pháp học máy tự động (autoML).

#### 1.1.3. Ứng dụng

Bài toán "Xây dựng hệ thống phát triển mô hình phân loại ảnh để nhận diện các loài động vật" có thể có nhiều ứng dụng thực tế quan trọng trong các lĩnh vực khác nhau. Dưới đây là một số ứng dụng chính:

Bảo tồn và nghiên cứu động vật hoang dã: Hệ thống có thể được sử dụng để theo dõi và giám sát các loài động vật hoang dã trong tự nhiên, giúp các nhà nghiên cứu và tổ chức bảo tồn hiểu rõ hơn về sự phân bố, hành vi và sự phát triển của các loài động vật. Điều này hỗ trợ trong việc bảo vệ các loài động vật nguy cấp và xây dựng các chiến lược bảo tồn hiệu quả.

Giám sát động vật trong các khu bảo tồn và vườn thú: Các khu bảo tồn động vật, vườn thú và các khu vực bảo vệ động vật khác có thể sử dụng hệ thống để nhận diện và theo dõi các loài động vật, giúp kiểm soát sức khỏe của động vật, phát hiện sớm các vấn đề, và cải thiện quản lý nguồn lực.

Phát hiện động vật xâm lấn: Hệ thống có thể được áp dụng để phát hiện các loài động vật xâm lấn, những loài không phải bản địa, gây hại cho hệ sinh thái địa phương. Phát hiện và theo dõi các loài xâm lấn giúp ngăn ngừa sự lan rộng của chúng và bảo vệ sự đa dạng sinh học bản địa.

Giám sát an ninh: Trong các khu vực bảo vệ động vật hoang dã, các công viên quốc gia hoặc các khu vực có hệ sinh thái nhạy cảm, hệ thống có thể giúp giám sát động vật tự nhiên trong các môi trường không giám sát trực tiếp. Điều này cũng có thể giúp phát hiện và ngăn chặn hành vi săn bắn trái phép hoặc phá hoại môi trường sống.

Ứng dụng trong công nghệ xe tự lái: Các hệ thống nhận diện động vật từ ảnh có thể được tích hợp vào công nghệ xe tự lái để phát hiện và tránh va chạm với động vật trong quá trình di chuyển, đặc biệt trong các khu vực có động vật hoang dã.

Giáo dục và nâng cao nhận thức cộng đồng: Hệ thống có thể được sử dụng trong các ứng dụng giáo dục và trò chơi, giúp người dùng tìm hiểu về các loài động vật qua hình ảnh và nhận diện tự động. Điều này giúp nâng cao nhận thức cộng đồng về bảo tồn động vật và đa dạng sinh học.

Ứng dụng trong các chương trình nghiên cứu sinh học và di truyền học: Việc phân loại chính xác các loài động vật là cơ sở quan trọng trong nghiên cứu sinh học và di truyền học, giúp các nhà khoa học phân tích và nghiên cứu đặc điểm di truyền, hành vi và sự thích nghi của các loài động vật trong môi trường sống của chúng.

Phát triển công cụ giám sát trong các khu vực tự nhiên: Các hệ thống giám sát động vật qua hình ảnh giúp nghiên cứu về các khu vực sinh thái, đặc biệt trong các khu vực khó tiếp cận. Các camera giám sát được gắn trong các khu vực này có thể tự động phân loại và nhận diện loài động vật, giảm thiểu sự can thiệp của con người.

## 1.2. Ngôn ngữ lập trình và các thư viện

#### 1.2.1. Giới thiệu về python

##### 1.2.1.1. Khái niệm

Python là một ngôn ngữ lập trình thông dịch, dễ đọc và dễ hiểu. Nền tảng nổi tiếng với cú pháp đơn giản và được sử dụng rộng rãi trong nhiều lĩnh vực khác nhau. Điển hình như phát triển web, phân tích dữ liệu, trí tuệ nhân tạo và nhiều ứng dụng khác.

Python có cú pháp linh hoạt và cấu trúc dữ liệu mạnh mẽ, công nghệ được hỗ trợ bởi một cộng đồng lớn. Điều này đã mang đến các thư viện và framework phong phú mà người dùng có thể sử dụng để xây dựng các ứng dụng phức tạp. Python cũng là một trong những ngôn ngữ phổ biến cho người mới học lập trình nhờ vào tính linh hoạt của nó [1].

##### 1.2.1.2. Lợi ích khi sử dụng Python

Cú pháp đơn giản và dễ đọc, phù hợp cho người mới học lập trình và cũng dễ dàng cho những người có kinh nghiệm.

Python được sử dụng trong nhiều lĩnh vực như phát triển web, phân tích dữ liệu, trí tuệ nhân tạo, khoa học dữ liệu và nhiều ứng dụng khác. Điều này làm cho Python trở thành một ngôn ngữ lập trình linh hoạt và tiện lợi.

Cộng đồng Python khá lớn và luôn hoạt động tích cực nhằm cung cấp nhiều thư viện, framework hữu ích. Người dùng sẽ được nhận sự hỗ trợ từ cộng đồng thông qua tài liệu, diễn đàn và các nguồn thông tin trực tuyến.

Python cung cấp nhiều thư viện mạnh mẽ cho phân tích dữ liệu như Pandas, NumPy, và Matplotlib. Bộ ngôn ngữ giúp người dùng xử lý và thể hiện dữ liệu một cách hiệu quả.

Python có thể chạy trên nhiều hệ điều hành khác nhau và trở thành một trong những ngôn ngữ lập trình chéo phổ biến.

Ngôn ngữ Python là mã nguồn mở và miễn phí, cho phép người dùng tự do sử dụng, phân phối và thay đổi phiên bản của nó [1].

##### 1.2.1.3. Phương thức hoạt động chính của Python

###### 1.2.1.3.1. Phát triển trang web trên máy chủ

Python có nhiều framework phát triển web phổ biến, ví dụ như Django, Flask, Pyramid và Fast API. Đây là các công cụ tối ưu giúp lập trình viên phát triển ứng dụng web từ phía máy chủ một cách dễ dàng và hiệu quả [1].

Các framework này cung cấp nhiều tính năng như xử lý URL, tương tác cơ sở dữ liệu, quản lý phiên và tạo giao diện người dùng. Django được xem là một trong những framework phát triển web phổ biến nhất trong cộng đồng Python. Nền tảng cung cấp sẵn các tính năng linh hoạt và nhiều công cụ hữu ích giúp cho việc xây dựng ứng dụng web phức tạp trở nên thuận lợi hơn.

Flask là một framework nhỏ gọn và linh hoạt hơn nên nó sẽ phù hợp cho việc xây dựng ứng dụng web từ nhỏ đến trung bình. Công nghệ cho phép người dùng tùy chỉnh nhiều hơn trên một hệ thống. Pyramid cũng là tiện ích  cung cấp chế độ hoạt động mạnh mẽ, đặc biệt là việc xây dựng ứng dụng lớn và phức tạp.

Fast API là một framework mới nhưng có tốc độ làm việc khá nhanh. Nền tảng được sử dụng để xây dựng các ứng dụng web hiệu suất cao và API. Tùy thuộc vào yêu cầu cụ thể của dự án mà người phát triển nên lựa chọn framework phù hợp với ngôn ngữ Python.

###### 1.2.1.3.2. Tập lênh Python tự động hóa

Để tự động hóa các tác vụ bằng Python, bạn có thể sử dụng các tập lệnh và scripts Python để thực hiện các thao tác như quản lý tệp tin, tương tác với cơ sở dữ liệu, gửi email tự động và thậm chí là tự động hóa các tác vụ hệ thống. Dưới đây là một ví dụ về cách sử dụng Python để tự động hóa một số thao tác cơ bản:

Quản lý tệp tin và thư mục: Bạn có thể sử dụng thư viện os để thực hiện các thao tác như tạo, di chuyển, xóa tệp tin và thư mục.

Tương tác với cơ sở dữ liệu: Python có thư viện như SQLAlchemy cho việc tương tác với cơ sở dữ liệu SQL. Công nghệ có các thư viện khác cho các loại cơ sở dữ liệu khác như MongoDB, Redis và nhiều loại cơ sở dữ liệu khác.

Gửi email tự động: Thư viện smtplib có thể được sử dụng để Gửi email bằng Python.

Tự động hóa các tác vụ hệ thống: Bằng cách sử dụng thư viện như subprocess, bạn có thể gọi các lệnh hệ thống và thực hiện các tác vụ như sao chép tệp tin, nén và giải nén tệp tin và thực hiện các lệnh hệ thống khác.

###### 1.2.1.3.3. Ứng dụng trong khoa học dữ liệu

Ngôn ngữ Python được sử dụng phổ biến trong lĩnh vực khoa học dữ liệu và máy học. Cách sử dụng thư viện và framework mạnh mẽ trong Python giúp cho việc phân tích dữ liệu hoặc triển khai mô hình lập trình trở nên dễ dàng hơn [1].

Dưới đây là một số thư viện quan trọng mà người làm việc trong lĩnh vực này thường sử dụng:

NumPy: Thư viện mạnh mẽ cho các phép toán trên mảng đa chiều và ma trận, cung cấp các hàm để thao tác dữ liệu số học một cách hiệu quả.

Pandas: Pandas cung cấp cấu trúc dữ liệu và công cụ phân tích dữ liệu mạnh mẽ, đặc biệt là trong việc làm việc với dữ liệu có cấu trúc như bảng và chuỗi thời gian.

Matplotlib và Seaborn: Đây là các thư viện hỗ trợ vẽ đồ thị và trực quan hóa dữ liệu một cách dễ dàng và mạnh mẽ.

Scikit-learn: Scikit-learn là một trong những thư viện học máy phổ biến nhất trong Python, cung cấp nhiều thuật toán lập trình và công cụ cho tiền xử lý dữ liệu, đánh giá mô hình và tinh chỉnh tham số.

TensorFlow và PyTorch: Đây là hai framework phổ biến cho việc triển khai mô hình học sâu (deep learning). Cả hai đều cung cấp API mạnh mẽ để xây dựng và huấn luyện mạng nơ-ron.

Những thư viện này cung cấp hệ sinh thái để phân tích dữ liệu, xây dựng mô hình và triển khai ứng dụng trong lĩnh vực khoa học dữ liệu. Python được xem là ngôn ngữ lập trình hàng đầu trong lĩnh vực này.

###### 1.2.1.3.4. Python phát triển phần mềm

Dưới đây là một số framework và công cụ quan trọng được sử dụng cho việc phát triển phần mềm bằng Python:

Django: Đây là một framework phát triển web mạnh mẽ, cung cấp các tính năng như quản lý URL, tương tác với cơ sở dữ liệu, và quản lý phiên. Django rất thích hợp cho việc phát triển các ứng dụng web phức tạp.

Flask: Flask là một framework nhẹ linh hoạt hơn, phù hợp cho việc phát triển các ứng dụng web từ nhỏ đến trung bình và cho phép tùy chỉnh linh hoạt hơn.

PyQt và Tkinter: Đây là các thư viện được sử dụng để phát triển ứng dụng desktop bằng Python. PyQt hỗ trợ viết ứng dụng đa nền tảng bằng cách sử dụng Qt framework. Trong khi đó, Tkinter đi kèm với Python và là cách để phát triển các ứng dụng desktop GUI đơn giản.

Kivy: Kivy là một framework Python hoàn chỉnh để phát triển ứng dụng di động với mục tiêu đa nền tảng.

Ngoài ra, Python cũng được sử dụng trong các dự án DevOps như viết các scripts tự động hóa hệ thống, quản lý cấu hình và triển khai ứng dụng.

###### 1.2.1.3.5. Tự động hóa kiểm thử phần mềm

Để tự động hóa kiểm thử phần mềm bằng Python cần sử dụng nhiều thư viện và framework mạnh mẽ [1]. Dưới đây giới thiệu một số công cụ và thư viện phổ biến được tích hợp trong quá trình tự động hóa kiểm thử phần mềm bằng Python:

Selenium: Selenium là một công cụ tự động hóa trình duyệt web, cho phép làm tự động các tác vụ như điều hướng, điền thông tin và kiểm tra các yếu tố trên trang web. Selenium có thể được sử dụng để viết kịch bản kiểm thử tự động cho ứng dụng web.

PyTest: PyTest là một framework kiểm thử mạnh mẽ và dễ sử dụng cho Python. Công nghệ cung cấp các cách linh hoạt để viết các bài kiểm tra và hỗ trợ chạy tự động các bài kiểm tra, báo cáo kết quả.

Robot Framework: Robot Framework là một framework kiểm thử tự động mã nguồn mở cho việc tự động hóa kiểm thử phần mềm. Nền tảng cung cấp các thư viện mở rộng cho nhiều mục đích, bao gồm kiểm thử giao diện người dùng, dịch vụ web và ứng dụng di động.

Unittest: Unittest là một framework kiểm thử chuẩn có sẵn trong thư viện chuẩn của Python. Tiện ích cung cấp giải pháp tổ chức các bài kiểm tra và thực hiện kiểm thử tự động.

#### 1.2.2. Giới thiệu về OpenCV

##### 1.2.2.1. OpenCV là gì

OpenCV là một bộ công cụ phần mềm hiện đại được ứng dụng trong quá trình xử lý hình ảnh, video, phân tích và Machine Learning theo thời gian thực. Nó được Intel phát triển từ năm 1999 và đã phát hành phiên bản đầu tiên vào năm 2000. Bộ công cụ OpenCV có hơn 2500 thuật toán được tối ưu hoá dành cho thị giác máy tính và Machine Learning. Nó hỗ trợ cho nhiều ngôn ngữ lập trình như C, C++, Python, Java,... trong việc thiết kế đa nền tảng và hoạt động trên nhiều phần cứng khác nhau. Mặt khác, OpenCV được thiết kế để tận dụng toàn bộ khả năng có sẵn trong phần cứng. Vì thế nó đảm bảo mang lại hiệu suất tốt nhất dành cho các ứng dụng máy tính ứng dụng OpenCV [2].

##### 1.2.2.2. Mục đích khi sử dụng OpenCV

OpenCV được ứng dụng vào nhiều mục đích khác nhau, trong đó chủ yếu phục vụ cho Computer Vision và Machine Learning. Cụ thể:

Hỗ trợ xử lý hình ảnh: Trong quá trình viết thuật toán về Computer Vision, lập trình viên cần sử dụng nhiều thao tác xử lý hình ảnh cơ bản. Hầu hết các chức năng xử lý hình ảnh này đều có sẵn trong OpenCV. Do đó, bạn có thể thực hiện nhiều thao tác cơ bản như lọc hình ảnh, phân tích hình dạng, chuyển đổi màu sắc, biến đổi hình ảnh cùng với một số thao tác khác [2].

Xây dựng và xử lý toàn bộ hoạt động GUI: OpenCV được cung cấp module Highgui giúp bạn có thể giải quyết toàn bộ vấn đề liên quan tới xây dựng và hoạt động GUI một cách đơn giản. Nếu bạn muốn kiểm tra hình ảnh trước khi bắt đầu bước tiếp theo, module Highgui sẽ tạo ra một cửa sổ biểu thị video hoặc hình ảnh. Cửa sổ này sẽ giúp bạn căn chỉnh kích thước, chất lượng hình ảnh/video phù hợp trước khi bắt đầu bước tiếp theo [2].

Cung cấp tất cả tác vụ để phân tích video: OpenCV sẽ cung cấp cho bạn các tác vụ như phân tích chuyển động giữa các khung hình liên tiếp trong video, theo dõi các đối tượng khác nhau trong video hay tạo mô hình giám sát... Ngoài ra, OpenCV cũng cung cấp các chức năng xử lý sự ổn định của video. Các thiết bị hiện đại đều được áp dụng kỹ thuật này trước khi công chiếu để đạt được chất lượng hình ảnh sắc nét và ấn tượng nhất cho người dùng.

Hỗ trợ tái tạo 3D trong Computer Vision: OpenCV cung cấp các thuật toán có thể tìm ra các mối quan hệ giữa những đối tượng khác nhau trong các hình ảnh 2D để tính toán cho vị trí 3D nhờ các thuật toán cụ thể.

Khai thác tính năng của Computer Vision: Người dùng thiết kế các trình trích xuất với tính năng khác nhau và trích xuất các điểm nổi bật từ một hình ảnh nhất định. Phát hiện đối tượng thông qua vị trí của chúng trong một hình ảnh cụ thể.

Cung cấp một loại module chứa nhiều thuật toán Machine Learning như Bayes Classifier, K-Nearest Neighbors, Decision Trees, Super Vector Machines... Chúng được sử dụng rộng rãi để xây dựng các hệ thống nhận dạng đối tượng, nhận diện gương mặt, phân loại hình ảnh, tìm kiếm trực quan tốt hơn.

Ngoài ra, OpenCV cũng được ứng dụng vào việc nhận dạng khuôn mặt và đối tượng: Bạn cần dò tìm đặc điểm xác định vị trí của khuôn mặt trong trường hợp muốn xây dựng một hệ thống sinh trắc học thông minh.

Hỗ trợ phát hiện và nhận dạng văn bản nhờ OpenCV chứa nhiều thuật toán khác nhau để xử lý các phát hiện hoặc nhận dạng văn bản.

Sử dụng OpenCV trong nhiếp ảnh điện toán bằng các kỹ thuật xử lý hình ảnh hiện đại nhằm cải thiết chất lượng hình ảnh chụp từ máy ảnh.

#### 1.2.3. Thư viện hỗ trợ

Dưới đây là mô tả ngắn gọn về các thư viện được sử dụng trong Python cho đề tài

torch: Thư viện deep learning mạnh mẽ, được sử dụng để xây dựng và huấn luyện các mô hình học sâu, đặc biệt là cho các tác vụ nhận diện đối tượng.

cv2 (OpenCV): Thư viện nổi tiếng trong xử lý ảnh và video, giúp mở, đọc và chỉnh sửa video, đồng thời hỗ trợ nhận diện đối tượng và trực quan hóa kết quả.

matplotlib.pyplot: Dùng để vẽ đồ thị và trực quan hóa kết quả huấn luyện và nhận diện, giúp theo dõi hiệu suất mô hình.

pandas: Thư viện xử lý dữ liệu, thường được sử dụng để lưu trữ và phân tích dữ liệu kết quả từ mô hình nhận diện, như số lượng đối tượng phát hiện được.

os: Thư viện thao tác với hệ thống tệp, giúp quản lý các tệp dữ liệu như video và mô hình huấn luyện.

Scikit-learn: Thư viện cung cấp các công cụ và thuật toán cần thiết để xây dựng, huấn luyện, đánh giá, và triển khai các mô hình học máy một cách dễ dàng.

# CHƯƠNG 2. XÂY DỰNG HỆ THỐNG NHẬN DẠNG BÀI TOÁN

## 2.1. Yêu cầu của bài toán

Nhận dạng hình ảnh trong lĩnh vực bảo tồn động vật ngày càng trở nên quan trọng. Việc xây dựng một hệ thống tự động để nhận diện các loài động vật thông qua ảnh có thể giúp ích trong nhiều lĩnh vực như nghiên cứu động vật hoang dã, bảo vệ các loài động vật quý hiếm, cũng như phục vụ trong các dự án du lịch sinh thái.

Mục tiêu của bài toán này là xây dựng một hệ thống nhận dạng các loài động vật dựa trên các hình ảnh. Các yêu cầu cụ thể của bài toán bao gồm:

Nhận diện nhiều loài động vật:

Hệ thống phải có khả năng phân loại một bức ảnh vào một trong số nhiều loài động vật (ví dụ: chó, mèo, voi, sư tử, hổ, v.v.).

Mỗi loài động vật sẽ được gắn với một nhãn phân loại duy nhất.

Xử lý dữ liệu ảnh:

Hệ thống cần phải sử dụng một lượng lớn ảnh đã được gán nhãn (labeled data) cho quá trình huấn luyện mô hình.

Dữ liệu ảnh phải được tiền xử lý, bao gồm thay đổi kích thước, chuẩn hóa, và phân chia thành các bộ dữ liệu huấn luyện và kiểm tra.

Phát triển mô hình học sâu:

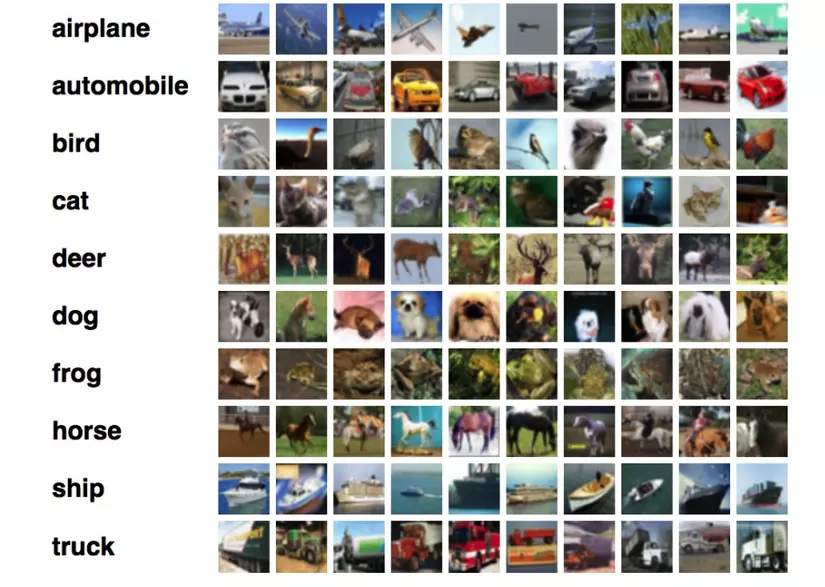
Sử dụng các mô hình học sâu như ResNet-18, VGG16, hoặc Inception để xây dựng mô hình phân loại ảnh.

Mô hình phải có khả năng học được các đặc trưng quan trọng từ ảnh để phân loại chính xác.

Đánh giá và cải thiện mô hình:

Hệ thống phải cung cấp các chỉ số đánh giá chất lượng mô hình như accuracy, precision, recall, F1-score.

Sử dụng Confusion Matrix để phân tích độ chính xác của mô hình.



Hình 2.1. Hình ảnh minh họa bài toán

## 2.2. Xây dựng hệ thống

Hệ thống nhận diện của ResNet-18

ResNet-18 là một phiên bản của mạng ResNet (Residual Networks), được thiết kế để giải quyết vấn đề "vanishing gradients" trong quá trình huấn luyện các mạng sâu (deep neural networks). ResNet-18 có 18 lớp và sử dụng kiến trúc residual (sót lại), cho phép mạng học tốt hơn và đạt hiệu quả cao khi có chiều sâu lớn mà không gặp vấn đề suy giảm gradient.

Một đặc điểm nổi bật của ResNet là sử dụng Residual Connections để bỏ qua một số lớp trong mạng, giúp giảm thiểu các vấn đề liên quan đến sự suy giảm gradient trong các mạng học sâu.

ResNet-18 có thể áp dụng cho nhiều bài toán phân loại hình ảnh khác nhau, bao gồm nhận diện các loài động vật trong bộ dữ liệu hình ảnh.

Quy trình nhận diện của ResNet-18

Chuẩn bị Dữ liệu

Trước khi huấn luyện mô hình, ta cần chuẩn bị bộ dữ liệu hình ảnh. Bộ dữ liệu này bao gồm các hình ảnh của các loài động vật khác nhau, đã được gán nhãn. Các bước chuẩn bị dữ liệu như sau:

Thu thập và Tổ chức Dữ liệu

Các ảnh động vật được phân loại theo các thư mục con, mỗi thư mục tương ứng với một loài động vật. Ví dụ, thư mục "dog" chứa ảnh của loài chó, thư mục "cat" chứa ảnh của loài mèo.

Tiền xử lý Dữ liệu

Resize ảnh: Các bức ảnh được thay đổi kích thước về một kích thước cố định, ví dụ 300x300 pixels.

Chuẩn hóa ảnh: Dữ liệu ảnh được chuẩn hóa để có giá trị pixel trong khoảng từ 0 đến 1, giúp tăng hiệu quả huấn luyện.

Chuyển đổi ảnh thành Tensor: Mô hình học sâu yêu cầu ảnh đầu vào dưới dạng tensor.

Huấn luyện Mô hình

Tải mô hình ResNet-18: Mô hình ResNet-18 được tải từ thư viện torchvision, với trọng số đã được huấn luyện trên bộ dữ liệu ImageNet.

Chỉnh sửa lớp phân loại: Vì bài toán yêu cầu phân loại 8 loài động vật, ta thay đổi lớp phân loại cuối cùng của ResNet-18 từ 1000 lớp thành 8 lớp tương ứng với 8 loài động vật.

Huấn luyện:

Optimizer: Sử dụng Adam Optimizer để cập nhật các trọng số của mô hình.

Loss Function: Sử dụng Cross-Entropy Loss để tính toán sai số giữa dự đoán của mô hình và nhãn thực tế.

Epochs: Mô hình sẽ được huấn luyện qua nhiều epoch, mỗi epoch gồm nhiều bước. Sau mỗi epoch, ta tính toán và lưu lại các chỉ số như độ chính xác và tổn thất.

Đánh giá mô hình: Sau khi huấn luyện, mô hình được đánh giá trên tập kiểm tra để đo lường hiệu quả phân loại bằng các chỉ số như Accuracy, Precision, Recall, và F1-Score.

Dự đoán và Nhận diện

Sau khi huấn luyện, mô hình có thể nhận diện các loài động vật trong các bức ảnh mới:

Tải ảnh mới: Người dùng tải lên ảnh của một loài động vật chưa được mô hình thấy trước đó.

Tiền xử lý ảnh: Ảnh được chuẩn hóa và chuyển thành tensor.

Dự đoán: Mô hình sẽ dự đoán loài động vật trong ảnh bằng cách tính toán các xác suất cho từng lớp.

Kết quả phân loại: Mô hình trả về lớp có xác suất cao.

Đánh giá và Tinh chỉnh Mô hình

Confusion Matrix: Đánh giá độ chính xác của mô hình thông qua confusion matrix.

Báo cáo Precision, Recall và F1-Score: Đánh giá các độ đo như precision, recall và F1-score cho từng loài động vật.

Fine-tuning: Nếu mô hình không đạt độ chính xác cao, có thể thực hiện fine-tuning bằng cách huấn luyện lại một số lớp nhất định trong mạng.

## 2.3. Các phương pháp áp dụng cho bài toán phân loại ảnh

#### 2.3.1. Ý tưởng của phương pháp

##### 2.3.1.1. Mô hình ResNet-18

**ResNet-18** (Residual Network) là một trong những mô hình nổi bật trong lĩnh vực học sâu (deep learning), đặc biệt trong phân loại ảnh và nhận diện đối tượng. ResNet-18 là một phần của gia đình **ResNet** (Residual Networks), được giới thiệu bởi Kaiming He và các cộng sự trong bài báo "Deep Residual Learning for Image Recognition" vào năm 2015. ResNet-18 là phiên bản nhỏ gọn nhất trong các mô hình ResNet với 18 lớp, nhưng vẫn giữ lại đặc tính mạnh mẽ của ResNet.

Việc sử dụng mô hình ResNet-18 trong bài toán "Xây dựng hệ thống phát triển mô hình phân loại ảnh để nhận diện các loài động vật" là một lựa chọn hợp lý và mang lại nhiều lợi ích đáng kể. Một trong những lý do chính là ResNet-18 giúp giải quyết vấn đề vanishing gradient (biến mất gradient), một trong những thách thức lớn khi huấn luyện các mạng nơ-ron sâu. Các kết nối dư thừa trong ResNet-18 giúp mạng học được các đặc trưng phức tạp mà không bị mất thông tin trong các lớp sâu, từ đó nâng cao độ chính xác trong việc phân loại các loài động vật.

Bên cạnh đó, ResNet-18 có khả năng học các đặc trưng hình ảnh phức tạp, điều này rất quan trọng trong việc nhận diện động vật vì mỗi loài có những đặc điểm hình dáng và màu sắc khác biệt. Mô hình này còn sử dụng trọng số đã huấn luyện trước (pretrained) từ tập dữ liệu ImageNet, giúp mô hình học được những đặc trưng cơ bản từ một lượng dữ liệu lớn, giảm thiểu thời gian huấn luyện và cải thiện độ chính xác khi phân loại các loài động vật.

Một điểm mạnh khác của ResNet-18 là khả năng tinh chỉnh (fine-tuning), cho phép người dùng điều chỉnh mô hình sao cho phù hợp với số lượng lớp phân loại động vật trong bài toán của mình. Điều này giúp mô hình linh hoạt và có thể mở rộng dễ dàng nếu cần phân loại thêm nhiều loài động vật. Hơn nữa, với chỉ 18 lớp, ResNet-18 là một mô hình nhẹ, giúp giảm thời gian huấn luyện và tiết kiệm tài nguyên tính toán mà vẫn duy trì hiệu quả cao trong phân loại ảnh.

ResNet-18 cũng đã chứng minh được hiệu suất rất tốt trong các cuộc thi phân loại ảnh như ImageNet, và việc áp dụng mô hình này vào nhận diện động vật trong môi trường thực tế là rất khả thi. Hệ thống sử dụng ResNet-18 có thể tự động phân loại các loài động vật trong các khu bảo tồn, vườn thú, hay trong các hệ thống giám sát động vật hoang dã, từ đó giúp tiết kiệm thời gian và công sức của con người.

##### 2.3.1.2. Thuật toán tối ưu Adam

**Thuật toán Adam** (viết tắt của Adaptive Moment Estimation) là một thuật toán tối ưu phổ biến được sử dụng trong huấn luyện các mô hình học sâu, đặc biệt là trong các mạng nơ-ron. Adam kết hợp ưu điểm của hai phương pháp tối ưu trước đó: **Momentum** và RMSprop. Thuật toán này đặc biệt hữu ích trong việc tối ưu các hàm mất mát phức tạp và không thể hiện rõ xu hướng gradient theo thời gian.

Việc sử dụng thuật toán Adam trong bài toán "Xây dựng hệ thống phát triển mô hình phân loại ảnh để nhận diện các loài động vật" mang lại nhiều lợi ích quan trọng. Đầu tiên, Adam có khả năng tự động điều chỉnh học suất cho mỗi trọng số trong quá trình huấn luyện. Điều này rất quan trọng trong bài toán phân loại ảnh động vật, bởi vì mỗi loài động vật có những đặc trưng hình ảnh khác nhau, và học suất cần được điều chỉnh sao cho phù hợp với từng đặc trưng này. Adam giúp tránh các vấn đề liên quan đến việc chọn học suất quá cao hoặc quá thấp, từ đó tối ưu hóa quá trình huấn luyện.

Thứ hai, thuật toán Adam giúp đảm bảo tính ổn định trong huấn luyện, đặc biệt khi huấn luyện các mô hình học sâu như ResNet18. Những đặc điểm hình ảnh của các loài động vật rất đa dạng, với sự khác biệt rõ rệt giữa các loài. Adam sử dụng các kỹ thuật như momentum và RMSprop để duy trì sự ổn định của quá trình huấn luyện, giúp mô hình học được các đặc trưng của ảnh mà không bị mất thông tin quan trọng.

Ngoài ra, Adam rất hiệu quả khi làm việc với dữ liệu thưa thớt, điều này có thể xảy ra trong bài toán phân loại ảnh động vật, khi một số loài có ít mẫu dữ liệu hơn so với các loài khác. Adam giúp cải thiện độ chính xác và giảm thiểu rủi ro do gradient không ổn định trong quá trình huấn luyện.

Một ưu điểm nổi bật khác của Adam là tính linh hoạt cao. Thuật toán này yêu cầu ít sự điều chỉnh tham số và có thể hoạt động hiệu quả với các giá trị mặc định của các tham số như β1 và β2. Điều này giúp tiết kiệm thời gian và công sức trong quá trình phát triển mô hình, đồng thời đạt được kết quả tối ưu mà không cần điều chỉnh tham số quá phức tạp.

Adam cũng giúp hội tụ nhanh chóng, điều này rất quan trọng khi làm việc với các mô hình phức tạp như ResNet18. Việc huấn luyện mô hình nhanh chóng mà vẫn duy trì độ chính xác cao là một yếu tố quan trọng trong việc phát triển hệ thống phân loại ảnh động vật, giúp tiết kiệm thời gian và tài nguyên tính toán.

Cuối cùng, Adam đã được chứng minh là hiệu quả trong các ứng dụng thực tế, bao gồm việc nhận diện ảnh động vật. Nhờ vào khả năng điều chỉnh học suất tự động và duy trì ổn định trong huấn luyện, Adam giúp mô hình đạt được độ chính xác cao và có thể ứng dụng vào các hệ thống nhận diện động vật trong môi trường thực tế.

##### 2.3.1.3. Cross - Entropy Loss

Cross - Entropy Loss là một hàm mất mát (loss function) rất phổ biến trong các bài toán phân loại, đặc biệt là trong các mô hình học sâu (deep learning). Nó được sử dụng để đánh giá sự khác biệt giữa phân phối xác suất dự đoán của mô hình và phân phối xác suất thực tế (ground truth) của dữ liệu.

Cross - Entropy Loss đo lường mức độ không giống nhau giữa hai phân phối xác suất: phân phối xác suất mà mô hình dự đoán và phân phối xác suất thực tế (thường được biểu diễn dưới dạng one-hot vector trong các bài toán phân loại). Đối với các bài toán phân loại đa lớp, hàm mất mát này sẽ tính toán độ lệch giữa các dự đoán của mô hình và nhãn thực tế cho tất cả các lớp.

Trong bài toán nhận diện các loài động vật, Cross - Entropy Loss sẽ được sử dụng để tính toán sự khác biệt giữa xác suất mà mô hình dự đoán cho mỗi loài động vật và nhãn thực tế của loài động vật trong ảnh. Ví dụ, nếu mô hình dự đoán xác suất cao nhất cho một bức ảnh là loài chó (với xác suất 0.7) trong khi nhãn thực tế là loài mèo, Cross - Entropy Loss sẽ tính toán sự sai lệch giữa xác suất dự đoán này và nhãn thực tế của loài mèo, từ đó điều chỉnh mô hình để cải thiện độ chính xác.

##### 2.3.1.4. Softmax

Softmax là một hàm kích hoạt được sử dụng phổ biến trong các bài toán phân loại, đặc biệt là trong các mô hình học sâu (deep learning) cho các bài toán phân loại đa lớp. Softmax chuyển các giá trị đầu ra của mô hình (logits) thành một phân phối xác suất, giúp mô hình đưa ra dự đoán về xác suất thuộc vào mỗi lớp.

Trong các bài toán phân loại đa lớp, mô hình sẽ đưa ra một vector đầu ra, mỗi phần tử trong vector này tương ứng với một lớp (class). Mặc dù các giá trị này có thể có bất kỳ giá trị nào (có thể âm hoặc dương), nhưng chúng không phải là xác suất. Để chuyển đổi các giá trị này thành xác suất, ta sử dụng hàm Softmax.

Trong bài toán nhận diện động vật, mỗi bức ảnh có thể thuộc về một trong nhiều lớp (ví dụ: chó, mèo, hổ, v.v.). Softmax cho phép mô hình dự đoán xác suất của bức ảnh thuộc về mỗi loài động vật, từ đó quyết định lớp có xác suất cao nhất là lớp dự đoán của mô hình.

Softmax không chỉ đưa ra lớp dự đoán mà còn cung cấp xác suất cho mỗi lớp, điều này rất hữu ích khi cần đánh giá độ chắc chắn của mô hình trong dự đoán.

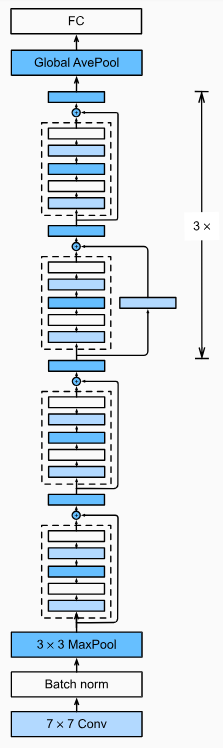
Khi sử dụng Softmax, mô hình có thể dễ dàng kết hợp với Cross Entropy Loss để tối ưu hóa quá trình huấn luyện, giúp mô hình phân loại chính xác hơn.

#### 2.3.2. Sơ đồ thuật toán và hình ảnh mô tả phương pháp

Sơ đồ thuật toán mô hình ResNet-18

* Input Layer: Nhận đầu vào là hình ảnh.
* Convolutional Layer: Áp dụng các lớp convolution để trích xuất đặc trưng.
* Residual Blocks: Các khối residual giúp giảm thiểu vấn đề gradient vanishing bằng cách thêm các kết nối tắt (skip connections).
* Pooling Layer: Giảm kích thước không gian của các đặc trưng.
* Fully Connected Layer: Kết nối toàn bộ để đưa ra dự đoán cuối cùng.
* Output Layer: Đưa ra kết quả phân loại.

Hình ảnh minh họa ResNet-18



Hình 2..2. Kiến trúc ResNet-18

Sơ đồ thuật toán tối ưu Adam

vt=β1vt−1+(1−β1)gtvt=β1vt−1+(1−β1)gt.

st=β2st−1+(1−β2)g2t.st=β2st−1+(1−β2)gt2.

Ở đây β1β1 và β2β2 là các tham số trọng số không âm thường được chọn các giá trị là β1=0.9β1=0.9 và β2=0.999β2=0.999. Điều này có nghĩa là ước lượng phương sai di chuyển chậm hơn nhiều so với số hạng động lượng. Lưu ý rằng nếu ta khởi tạo v0=s0=0v0=s0=0, thuật toán sẽ có độ chệch ban đầu đáng kể về các giá trị nhỏ hơn. Vấn đề này có thể giải quyết bằng cách sử dụng ∑ti=0βi=1−βt1−β∑i=0tβi=1−βt1−β để chuẩn hóa lại các số hạng. Tương tự, các biến trạng thái được chuẩn hóa như sau:

^v=vt1−βt1v^=vt1−β1t và ^s=st1−βt2s^=st1−β2t

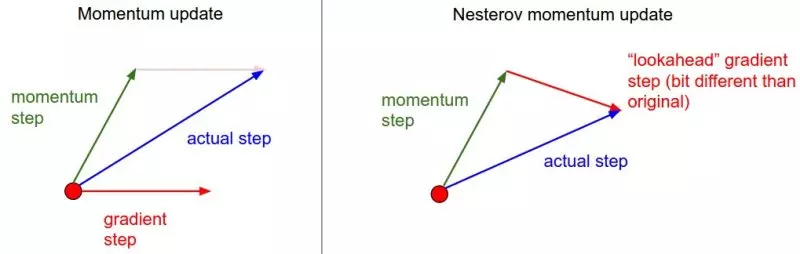
Với các ước lượng thích hợp, bây giờ chúng ta có thể viết ra các phương trình cập nhật. Đầu tiên, chúng ta điều chỉnh lại giá trị gradient, tương tự như ở RmsProp để có được:

g′t=η^vt√^st+ϵg′t=ηvt^st^+ϵ

Không giống như Rmsprop, phương trình cập nhật sử dụng động lượng ^vtvt^ thay vì gradient. Hơn nữa, có một số khác biệt nhỏ ở đây: phép chuyển đổi được thực hiện bằng cách sử dụng 1√^st+ϵ1st^+ϵ thay vì 1√^st+ϵ1st^+ϵ. Trong thực tế, cách đầu tiên hoạt động tốt hơn một chút, dẫn đến sự khác biệt này so với Rmsprop. Thông thường, ta chọn ϵ=10−6ϵ=10−6 để cân bằng giữa tính ổn định số học và độ tin cậy.  
Bây giờ ta sẽ tổng hợp lại tất cả các điều trên để tính toán bước cập nhật [3].

xt=xt−1−g′txt=xt−1−g′t.

Hình ảnh minh họa thuật toán tối ưu Adam

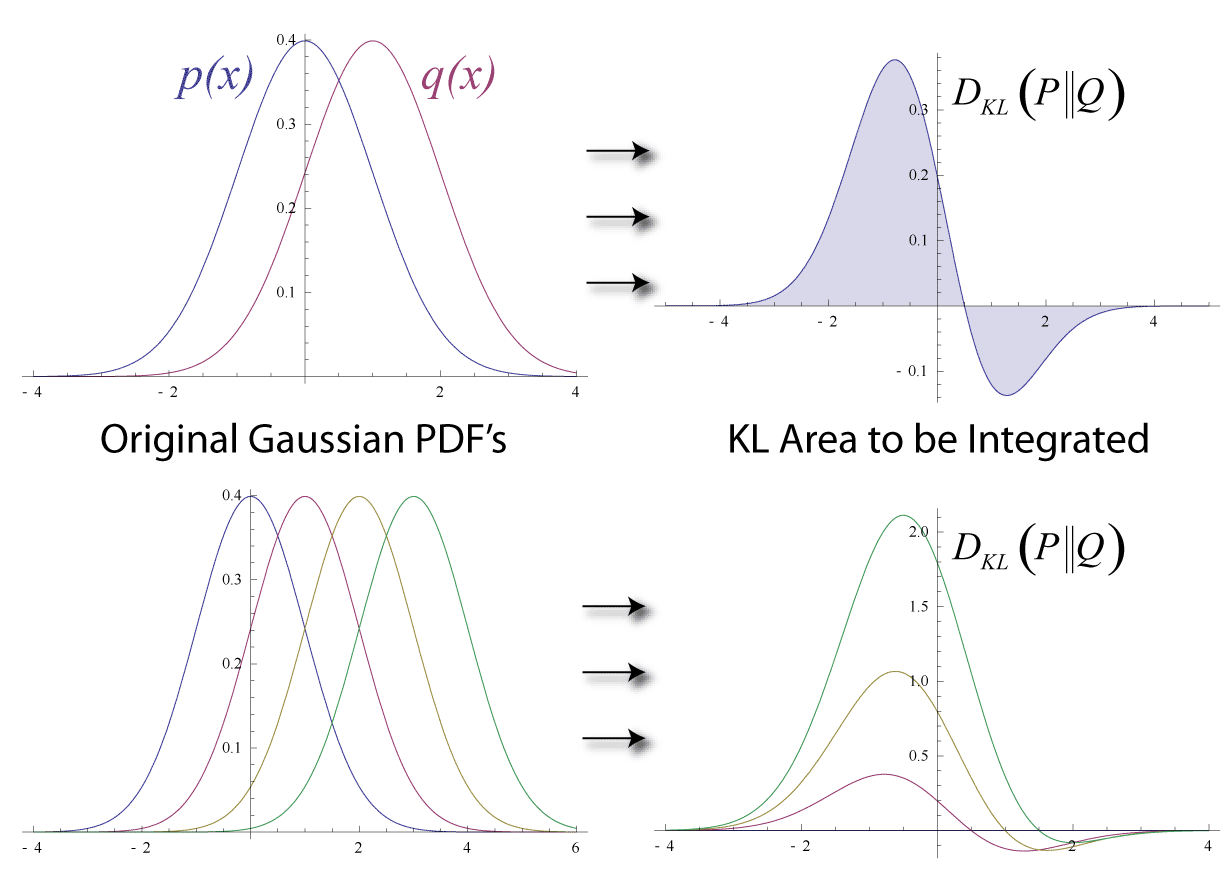


Hình 2.3. Thuật toán tối ưu Adam

Sơ đồ thuật toán Cross – Entropy Loss

* Input: Nhận đầu vào là các xác suất dự đoán của mô hình và các nhãn thực tế.
* Logarithm: Tính log của các xác suất dự đoán.
* Multiplication: Nhân các giá trị log này với các nhãn thực tế.
* Summation: Tính tổng các giá trị đã nhân.
* Negation: Lấy giá trị âm của tổng này để có được giá trị cross-entropy loss.

Hình ảnh minh họa sơ đồ thuật toán Cross – Entropy Loss

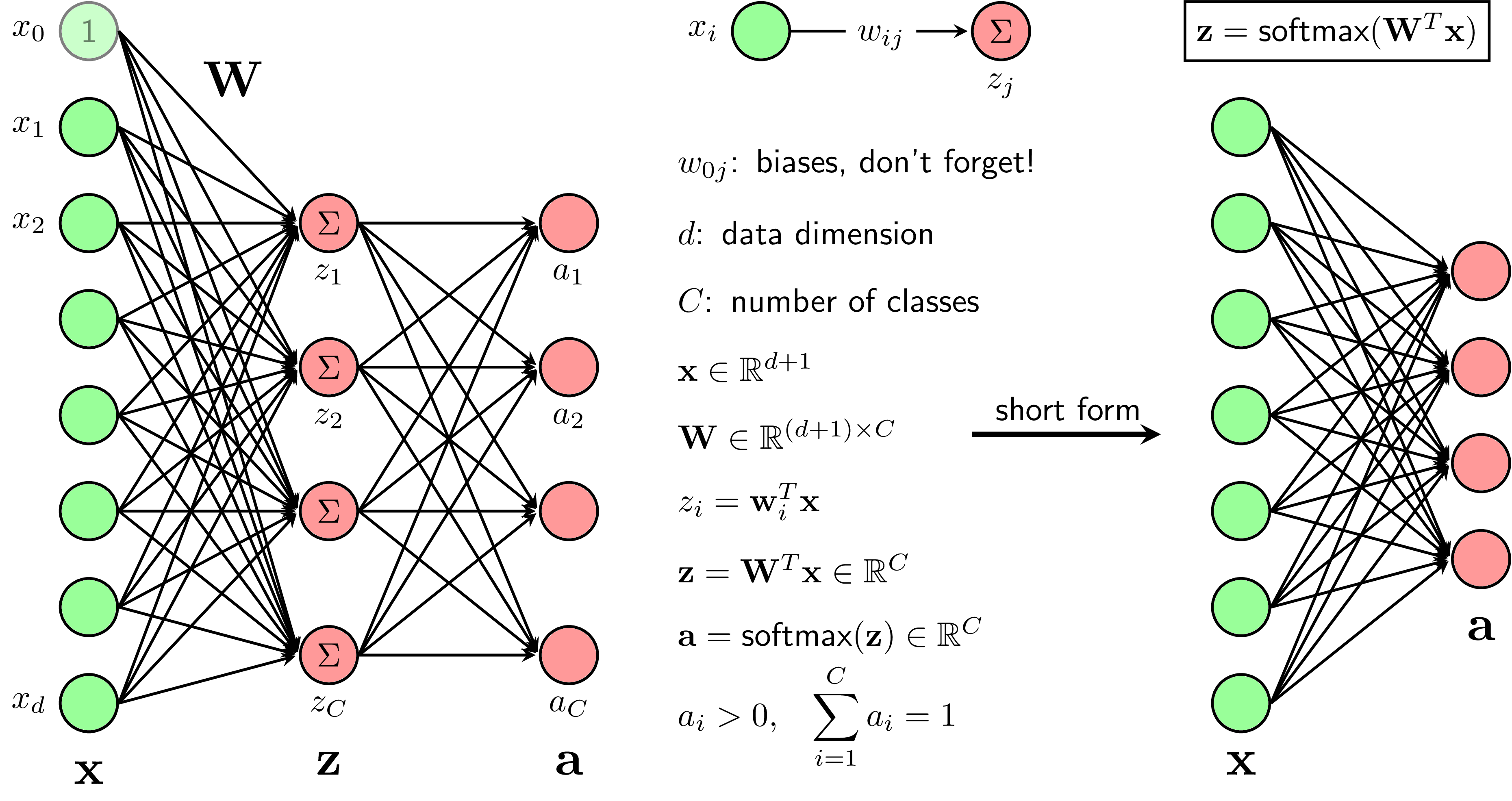


Hình 2.4. Hình ảnh minh họa sơ đồ thuật toán Cross – Entropy Loss

Sơ đồ thuật toán Softmax

* Input: Nhận đầu vào là một vector các giá trị thực (logits).
* Exponentiation: Tính giá trị mũ của từng phần tử trong vector đầu vào.
* Summation: Tính tổng các giá trị mũ vừa tính được.
* Division: Chia từng giá trị mũ cho tổng các giá trị mũ để chuẩn hóa thành xác suất.

Hình ảnh minh họa sơ đồ thuật toán Softmax



Hình 2. 5. Mô hình Softmax Regression dưới dạng Neural network

#### 2.3.3. Ưu và nhược điểm của phương pháp

##### 2.3.3.1. Ưu và nhược điểm của ResNet – 18

Ưu điểm:

Giải quyết vấn đề gradient vanishing và exploding: Một trong những vấn đề lớn trong việc huấn luyện các mạng sâu là vanishing gradient (suy giảm gradient) hoặc exploding gradient (bùng nổ gradient), đặc biệt là khi độ sâu của mạng tăng lên. ResNet giải quyết vấn đề này bằng cách sử dụng Residual Connections (kết nối dư thừa), cho phép gradient được truyền qua các lớp mà không bị mất đi quá nhiều, giúp việc huấn luyện trở nên hiệu quả hơn.

Mô hình nhẹ và nhanh: Với chỉ 18 lớp, ResNet-18 là phiên bản nhẹ hơn trong gia đình ResNet. Điều này giúp mô hình huấn luyện nhanh hơn, tiết kiệm tài nguyên tính toán và phù hợp cho các ứng dụng yêu cầu tốc độ cao hoặc có tài nguyên hạn chế, như nhận diện ảnh trong các thiết bị di động hoặc hệ thống thời gian thực.

Khả năng tổng quát tốt: ResNet-18 có khả năng học các đặc trưng phức tạp trong ảnh mà không gặp phải vấn đề overfitting (quá khớp) trong nhiều trường hợp. Điều này đặc biệt quan trọng trong các bài toán phân loại ảnh động vật, nơi có sự đa dạng lớn giữa các loài và hình ảnh.

Tính linh hoạt và dễ dàng áp dụng: ResNet-18 có thể áp dụng cho nhiều loại bài toán khác nhau, từ phân loại ảnh đến các tác vụ như phát hiện đối tượng hoặc phân đoạn ảnh. Kiến trúc này có thể dễ dàng được tích hợp vào các hệ thống khác và dễ dàng mở rộng để xử lý các bài toán phức tạp hơn.

Hiệu suất cao: ResNet-18 đạt được hiệu suất rất tốt trên các tập dữ liệu lớn như ImageNet. Các lớp residual cho phép mô hình học được từ các lớp sâu mà không gặp phải vấn đề suy giảm hiệu suất khi độ sâu của mạng tăng lên.

Nhược điểm:

Không phù hợp cho các bài toán cực kỳ phức tạp: Mặc dù ResNet-18 là mô hình mạnh mẽ, nhưng đối với các bài toán rất phức tạp hoặc yêu cầu khả năng nhận diện chi tiết với độ chính xác rất cao (ví dụ: phân loại chi tiết trong các ảnh y tế hoặc các tác vụ nhận diện ảnh phức tạp), các mô hình với nhiều lớp hơn như ResNet-50 hoặc ResNet-101 có thể mang lại kết quả tốt hơn.

Khó khăn trong việc tối ưu hóa khi mở rộng mạng: Mặc dù ResNet-18 giải quyết vấn đề gradient vanishing rất hiệu quả, nhưng khi mở rộng độ sâu của mạng (ví dụ: ResNet-152), việc tối ưu hóa mô hình vẫn có thể gặp khó khăn. Các mô hình sâu hơn yêu cầu phải điều chỉnh kỹ thuật huấn luyện và các tham số tối ưu hóa phức tạp hơn.

Chi phí tính toán và bộ nhớ lớn đối với các mô hình phức tạp: Mặc dù ResNet-18 nhẹ hơn nhiều so với các phiên bản như ResNet-50 hoặc ResNet-152, nhưng khi sử dụng trong các bài toán với dữ liệu lớn và số lượng lớp lớn, việc lưu trữ và tính toán có thể trở nên tốn kém về bộ nhớ và tài nguyên tính toán.

Khó hiểu và giải thích: Giống như các mô hình học sâu khác, ResNet-18 là một mạng lưới phức tạp với nhiều lớp và kết nối, điều này khiến việc giải thích các quyết định của mô hình (chẳng hạn như lý do mô hình phân loại ảnh thành một loài động vật cụ thể) trở nên khó khăn. Các mô hình này thường hoạt động như một "hộp đen", và khó có thể hiểu chi tiết về cách thức mô hình đưa ra dự đoán.

##### 2.3.3.2. Ưu và nhược điểm của thuật toán tối ưu Adam

Ưu điểm:

Tự điều chỉnh tốc độ học: Một trong những ưu điểm lớn nhất của Adam là khả năng tự động điều chỉnh tốc độ học (learning rate) cho từng tham số trong mô hình. Điều này giúp Adam vượt qua các vấn đề liên quan đến việc lựa chọn tốc độ học cố định, đặc biệt là khi các tham số có độ thay đổi khác nhau trong quá trình huấn luyện.

Hiệu quả với các bài toán có dữ liệu lớn và mô hình phức tạp: Adam hoạt động rất hiệu quả đối với các bài toán với dữ liệu lớn, mô hình phức tạp và nhiều tham số. Vì nó sử dụng các moment (momentum) để điều chỉnh tốc độ học, Adam giúp giảm sự dao động trong quá trình tối ưu hóa và giúp mô hình học nhanh hơn, ổn định hơn.

Khả năng hội tụ nhanh: Adam có khả năng hội tụ nhanh chóng và ổn định hơn so với các thuật toán tối ưu khác như SGD (Stochastic Gradient Descent), đặc biệt khi huấn luyện với dữ liệu có nhiều nhiễu (noisy data) hoặc các bài toán không quá mượt mà trong không gian tham số.

Ít yêu cầu về điều chỉnh siêu tham số (hyperparameters): Một ưu điểm khác của Adam là ít cần điều chỉnh siêu tham số. Mặc dù các tham số như tốc độ học (learning rate) và các hệ số beta có thể ảnh hưởng đến kết quả, Adam thường mang lại kết quả tốt ngay cả khi không tinh chỉnh quá nhiều các siêu tham số, giúp tiết kiệm thời gian và công sức.

Ổn định và không dễ bị sa lầy trong các cực trị: Adam giúp giảm thiểu việc sa lầy vào cực trị địa phương nhờ vào việc sử dụng cả moment (momentum) và động lượng (velocity) của gradient. Điều này giúp mô hình dễ dàng vượt qua những điểm thấp (local minima) và tối ưu hóa tốt hơn.

Nhược điểm:

Đôi khi hội tụ đến các điểm cực trị không tối ưu: Mặc dù Adam có khả năng hội tụ nhanh, nhưng nó có thể hội tụ đến các điểm cực trị không tối ưu trong một số bài toán. Điều này có thể xảy ra do các điều chỉnh tốc độ học riêng biệt cho mỗi tham số có thể không phù hợp trong một số trường hợp, dẫn đến sự hội tụ vào các cực trị không lý tưởng.

Có thể bị lệch trong các bài toán có số lượng mẫu nhỏ: Adam có thể không hoạt động tốt trong các bài toán với số lượng mẫu nhỏ hoặc dữ liệu không đầy đủ, vì các ước lượng moment có thể không chính xác. Điều này có thể dẫn đến việc tối ưu hóa không đạt được kết quả tốt nhất.

Yêu cầu bộ nhớ lớn hơn: Vì Adam lưu trữ các moment (momentum) cho mỗi tham số, nó yêu cầu nhiều bộ nhớ hơn so với các thuật toán như SGD. Điều này có thể gây vấn đề đối với các mô hình có quá nhiều tham số hoặc khi huấn luyện trên các thiết bị có tài nguyên hạn chế (ví dụ: GPU với bộ nhớ thấp).

Có thể bị chệch trong trường hợp độ chính xác cần thiết cao: Trong một số trường hợp, mặc dù Adam có thể hội tụ nhanh, nhưng kết quả cuối cùng không luôn đạt được độ chính xác tối ưu. Đặc biệt khi cần độ chính xác rất cao, Adam có thể gặp khó khăn trong việc tìm ra điểm cực trị toàn cục của hàm mất mát.

Không phù hợp cho tất cả các loại mô hình: Adam không phải lúc nào cũng là lựa chọn tốt nhất cho mọi bài toán. Trong một số bài toán đặc thù, như các mô hình rất đơn giản hoặc các mô hình với dữ liệu ít thay đổi, các thuật toán tối ưu đơn giản như SGD có thể hoạt động tốt hơn.

##### 2.3.3.3. Ưu và nhược điểm Cross – Entropy Less

Ưu điểm:

Được sử dụng phổ biến trong các bài toán phân loại: Cross-Entropy Loss là hàm mất mát chuẩn trong các bài toán phân loại, đặc biệt là trong các mô hình phân loại đa lớp (multi-class classification) hoặc phân loại nhị phân. Hàm này hoạt động rất tốt khi các lớp có xác suất phân bố rõ ràng và khi mô hình dự đoán xác suất cho các lớp.

Dễ dàng tối ưu hóa: Vì Cross-Entropy Loss liên quan trực tiếp đến log likelihood của phân phối xác suất, nó có các đặc tính toán học dễ dàng tối ưu hóa thông qua các phương pháp gradient descent. Hàm này thường hội tụ nhanh hơn và hiệu quả hơn so với các hàm mất mát khác trong các bài toán phân loại.

Khả năng đánh giá chính xác hiệu suất mô hình: Cross-Entropy Loss không chỉ đo lường độ sai lệch giữa dự đoán và nhãn thực tế, mà còn phản ánh mức độ tự tin của mô hình trong các dự đoán của mình. Nếu mô hình dự đoán với xác suất rất cao nhưng sai, thì hàm mất mát sẽ rất lớn, giúp mô hình điều chỉnh và học được tốt hơn.

Khả năng làm việc với nhiều lớp (multi-class): Hàm Cross-Entropy rất linh hoạt và có thể được sử dụng trong các bài toán phân loại nhiều lớp. Nó có thể tính toán loss cho các mô hình phân loại softmax, trong đó mỗi lớp có một xác suất riêng biệt và tổng xác suất là 1.

Giúp giảm thiểu overfitting trong huấn luyện: Vì Cross-Entropy Loss đo lường sự khác biệt giữa các phân phối xác suất thực tế và dự đoán, nó thúc đẩy mô hình học các đặc trưng quan trọng hơn thay vì chỉ tập trung vào các lỗi đơn lẻ, giúp giảm thiểu overfitting trong nhiều trường hợp.

Nhược điểm:

Yêu cầu dự đoán xác suất: Cross-Entropy Loss yêu cầu mô hình dự đoán xác suất cho từng lớp. Điều này có thể không phải lúc nào cũng phù hợp trong các bài toán yêu cầu dự đoán nhãn rời rạc mà không cần thông tin về xác suất, hoặc trong các bài toán mà kết quả nhãn cuối cùng là duy nhất và không cần phân phối xác suất.

Nhạy cảm với nhãn không cân bằng: Khi làm việc với các bộ dữ liệu có nhãn không cân bằng (ví dụ: một lớp xuất hiện rất nhiều lần trong khi các lớp khác xuất hiện rất ít), Cross-Entropy Loss có thể bị bị chi phối bởi lớp chiếm ưu thế. Điều này có thể khiến mô hình không học được tốt các lớp hiếm hoặc khó nhận diện, dẫn đến hiệu suất phân loại kém đối với các lớp ít dữ liệu.

Không trực tiếp tối ưu hóa độ chính xác: Mặc dù Cross-Entropy Loss giúp mô hình tối ưu hóa xác suất dự đoán, nhưng nó không trực tiếp tối ưu hóa độ chính xác (accuracy) của mô hình. Do đó, một mô hình có giá trị loss thấp có thể không đồng nghĩa với việc có độ chính xác cao trong các bài toán phân loại, nhất là khi mô hình dự đoán sai nhưng với xác suất thấp.

Không hữu ích trong các bài toán không phân loại: Trong các bài toán hồi quy hoặc các bài toán mà đầu ra không phải là nhãn phân loại, Cross-Entropy Loss không phải là sự lựa chọn phù hợp. Trong trường hợp này, các hàm mất mát khác như Mean Squared Error (MSE) sẽ phù hợp hơn.

##### 2.3.3.4. Ưu và nhược điểm Softmax

Ưu điểm:

Biến đổi thành xác suất: Softmax chuyển đổi các giá trị logits (giá trị chưa chuẩn hóa) thành xác suất, điều này rất hữu ích trong các bài toán phân loại. Xác suất cho mỗi lớp biểu thị khả năng mà mô hình dự đoán lớp đó là đúng, giúp dễ dàng đưa ra quyết định phân loại.

Tổng xác suất bằng 1: Sau khi sử dụng Softmax, các giá trị đầu ra trở thành một phân phối xác suất hợp lệ, có tổng bằng 1. Điều này rất hữu ích khi cần xác định xác suất dự đoán cho mỗi lớp trong các bài toán phân loại nhiều lớp (multi-class classification).

Dễ dàng tối ưu hóa: Softmax làm việc tốt với Hàm Mất mát Cross-Entropy (Cross-Entropy Loss), một trong những hàm mất mát phổ biến nhất trong học máy. Khi kết hợp Softmax với Cross-Entropy, mô hình có thể tối ưu hóa hiệu quả nhờ vào việc kết hợp xác suất và sai số giữa dự đoán và giá trị thực tế.

Giải quyết bài toán phân loại đa lớp: Softmax giúp giải quyết bài toán phân loại đa lớp, nơi mỗi đối tượng có thể thuộc một trong nhiều lớp. Việc chuyển đổi các logits thành xác suất giúp mô hình dễ dàng nhận diện lớp nào có xác suất cao nhất, từ đó đưa ra dự đoán.

Giải thích được kết quả: Với đầu ra là xác suất, người dùng có thể dễ dàng giải thích các kết quả dự đoán của mô hình. Ví dụ, mô hình có thể dự đoán rằng "hình ảnh này có 80% khả năng là một con mèo và 20% là một con chó", giúp người dùng hiểu rõ hơn về sự tự tin của mô hình.

Nhược điểm:

Nhạy cảm với giá trị cực đoan: Softmax có thể rất nhạy cảm với các giá trị cực đoan (outliers) trong các logits đầu vào. Nếu một trong các giá trị logits quá lớn hoặc quá nhỏ, điều này có thể dẫn đến việc một lớp có xác suất quá cao, trong khi các lớp khác có xác suất rất nhỏ, khiến mô hình trở nên quá tự tin và không chính xác.

Không làm việc tốt với phân lớp không cân bằng: Trong các bài toán phân loại mà dữ liệu không cân bằng (ví dụ: một lớp xuất hiện rất ít so với các lớp khác), Softmax có thể không hoạt động tốt vì nó có xu hướng tập trung vào các lớp phổ biến hơn. Khi kết hợp với Cross-Entropy Loss, mô hình có thể gặp khó khăn trong việc học các lớp hiếm gặp. Tuy nhiên, điều này có thể được khắc phục bằng cách sử dụng các chiến lược như weighted loss (mất mát có trọng số).

Khó khăn trong các bài toán phân loại nhị phân: Trong các bài toán phân loại nhị phân (binary classification), Softmax có thể không phải là sự lựa chọn tối ưu. Thay vào đó, trong trường hợp phân loại nhị phân, sử dụng hàm sigmoid sẽ thích hợp hơn, vì sigmoid có thể xuất ra xác suất cho hai lớp riêng biệt mà không yêu cầu phải chuẩn hóa tất cả các lớp như Softmax.

Khó khăn khi có nhiều lớp: Khi số lượng lớp trong bài toán phân loại rất lớn, Softmax có thể gặp khó khăn trong việc tính toán hiệu quả. Mặc dù vấn đề này có thể được giảm thiểu bằng cách sử dụng các phương pháp tối ưu hóa, nhưng Softmax có thể đòi hỏi tính toán nặng khi xử lý một số lượng lớp rất lớn.

Không phân biệt giữa các lớp có độ chắc chắn thấp: Mặc dù Softmax giúp tạo ra một phân phối xác suất, nhưng nó không thể phân biệt các lớp có độ chắc chắn thấp trong các trường hợp phân loại không rõ ràng. Nếu tất cả các lớp đều có xác suất thấp, Softmax vẫn sẽ phân bổ xác suất cho mỗi lớp, điều này có thể làm cho mô hình khó đưa ra dự đoán chính xác trong các trường hợp không chắc chắn.

# CHƯƠNG 3. THỰC NGHIỆM CHƯƠNG TRÌNH

## 3.1. Dữ liệu

Hệ thống dữ liệu được thu thập từ Kaggle, một nền tảng trực tuyến nổi tiếng dành cho các chuyên gia và những người đam mê dữ liệu (data science), nơi họ có thể tham gia vào các cuộc thi, học hỏi và chia sẻ kiến thức về các chủ đề liên quan đến phân tích dữ liệu, học máy (machine learning), và trí tuệ nhân tạo (AI). Nền tảng này thuộc sở hữu của Google và trở thành một trong những cộng đồng lớn nhất trong lĩnh vực khoa học dữ liệu.

Kaggle được lựa chọn vì nền tảng này cung cấp các tập dữ liệu đã được chú thích sẵn với chất lượng cao, giúp giảm đáng kể thời gian cần thiết để chuẩn bị dữ liệu cho các mô hình học máy. Ngoài ra Kernels là các tài liệu mã nguồn (code notebooks) cho phép người dùng chia sẻ các nghiên cứu, mô hình, hoặc giải pháp của họ. Thực tế, Kaggle cung cấp một môi trường làm việc trực tuyến miễn phí để người dùng có thể chạy code Python, R, và các công cụ học máy trực tiếp trên nền tảng mà không cần phải cài đặt gì trên máy tính của mình.

Mô tả dữ liệu

Tập dữ liệu sử dụng trong hệ thống bao gồm các thông tin cụ thể như sau:

Số lượng ảnh: 15,464 ảnh. Đây là số lượng đủ lớn để đảm bảo mô hình có thể học và nhận diện được các đặc trưng khác nhau trong bài toán theo dõi đối tượng.

Các lớp đối tượng:

Butterflies: Nhận diện bướm

Cat: Nhận diện mèo

Chickens: Nhận diện gà

Dog: Nhận diện chó

Elephants: Nhận diện voi

Horses: Nhận diện ngựa

Spiders: Nhận diện nhện

Squirrels: Nhận diện sóc

Các lớp này được lựa chọn dựa trên tính ứng dụng thực tiễn của hệ thống, phù hợp với các tình huống nhận diện các đối tượng trong các môi trường khác nhau

Định dạng dữ liệu:

Dữ liệu được lưu trữ dưới dạng tệp hình ảnh ở định dạng .jpg hoặc .png.

Các định dạng này được chuẩn hóa và tối ưu để phù hợp với hệ thống, đảm bảo tốc độ xử lý nhanh và độ chính xác cao.

Tỷ lệ phân chia dữ liệu:

Tập huấn luyện (train): 85% dữ liệu.

Tập kiểm thử (test): 9% dữ liệu.

Tập xác thực (validation): 5% dữ liệu.

Tỷ lệ này đảm bảo rằng mô hình có đủ dữ liệu để học, đồng thời duy trì được tính khách quan và độ chính xác khi đánh giá hiệu suất.

Tiền xử lý dữ liệu

Trước khi đưa dữ liệu vào mô hình, các bước tiền xử lý được thực hiện để tối ưu hóa khả năng học của hệ thống, bao gồm:

Chuyển đổi kích thước ảnh:

Tất cả các ảnh đầu vào được chuẩn hóa về kích thước 300x300 pixels, đây là kích thước tiêu chuẩn giúp tối ưu hóa hiệu suất.

Quá trình này đảm bảo các ảnh có cùng độ phân giải, giúp giảm tải xử lý và tăng tính nhất quán trong quá trình huấn luyện.

Chuẩn hóa màu sắc:

Các ảnh được chuẩn hóa giá trị pixel để phù hợp với các giá trị đầu vào của mạng nơ-ron tích chập (CNN).

Tăng cường dữ liệu (Data Augmentation):

Trong một số trường hợp, các kỹ thuật như xoay ảnh, thay đổi độ sáng hoặc cắt ảnh có thể được sử dụng để tạo ra nhiều biến thể khác nhau của cùng một hình ảnh. Điều này giúp cải thiện khả năng tổng quát hóa của mô hình.

Loại bỏ ảnh nhiễu:

Các ảnh bị lỗi hoặc có chất lượng kém (nhòe, mờ) được lọc ra để đảm bảo tập dữ liệu đầu vào có chất lượng cao.

## 3.2. Các độ đo so sánh

Tổng quan: Tất cả các lớp (butterflies, cat, chickens, dog, elephants, horses, spiders, squirrels) cho thấy các chỉ số đều rất cao, gần đạt 1.0, cho thấy mô hình hoạt động rất hiệu quả.

So sánh giữa các chỉ số:

Precision hầu như nằm ở mức cao nhất cho tất cả các lớp, cho thấy đây là yếu tố mà mô hình vượt trội.

Recall có giá trị thấp hơn một chút so với Precision ở một số lớp (đặc biệt là cho lớp elephants), điều này có thể chỉ ra rằng có một số đối tượng chưa được phát hiện đầy đủ.

F1-Score cao cho tất cả các lớp, cho thấy sự kết hợp giữa Precision và Recall là tốt.

Kết luận:

Mô hình cho thấy hiệu suất rất cao với tất cả các lớp. Tuy nhiên, cần chú ý đến lớp elephants, nơi Recall có vẻ chưa đạt yêu cầu bằng các lớp khác. Cần xem xét và cải thiện mô hình,

A graph of different colored bars

Description automatically generated with medium confidence

Hình 3.1. Biểu đồ so sánh số liệu

## 3.3 Kết quả

Hệ thống được huấn luyện dựa trên các thông số kỹ thuật cụ thể, nhằm đảm bảo hiệu suất cao và khả năng học tốt của mô hình trong việc phát hiện và theo dõi đối tượng. Các thông số chi tiết bao gồm:

Epochs:

Số lần huấn luyện toàn bộ tập dữ liệu: 10 epochs.

Mỗi epoch đại diện cho một lượt huấn luyện khi toàn bộ tập dữ liệu được đưa qua mô hình để tối ưu hóa các tham số. Việc huấn luyện với 10 epochs đảm bảo rằng mô hình có đủ thời gian để học và điều chỉnh các tham số nhằm đạt hiệu suất cao nhất, đồng thời tránh hiện tượng quá khớp (overfitting).

Batch size:

Kích thước lô dữ liệu: 16.

Batch size đề cập đến số lượng mẫu dữ liệu được xử lý cùng lúc trong một lần huấn luyện. Với batch size là 16, mô hình vừa đảm bảo sử dụng hiệu quả bộ nhớ GPU, vừa duy trì tốc độ huấn luyện ở mức tối ưu. Điều này phù hợp với kích thước của dữ liệu đầu vào và khả năng xử lý của phần cứng.

Kích thước ảnh đầu vào:

300x300 pixels.

Tất cả các ảnh trong tập dữ liệu đầu vào được chuẩn hóa về kích thước cố định 300x300 pixels. Kích thước này được lựa chọn dựa trên sự cân bằng giữa độ chính xác của mô hình và tốc độ xử lý, đặc biệt phù hợp với các ứng dụng thời gian thực.

Thời gian huấn luyện:

Tổng thời gian huấn luyện: 4.148 giờ.

Với tập dữ liệu lớn, thời gian huấn luyện kéo dài hơn 4 giờ, cho thấy mức độ phức tạp của bài toán. Thời gian này được tối ưu nhờ sử dụng GPU mạnh mẽ để tăng tốc quá trình tính toán.

GPU memory sử dụng:

Dung lượng bộ nhớ GPU: ~3.06GB.

Đây là dung lượng bộ nhớ được sử dụng trong quá trình huấn luyện, bao gồm việc lưu trữ các tham số mô hình, dữ liệu đầu vào và kết quả trung gian trong các bước tính toán. Dung lượng này cho thấy mô hình sử dụng tài nguyên một cách hiệu quả để xử lý các bài toán phức tạp.

Kiến trúc của mô hình ResNet-18

Mô hình ResNet-18 (Residual Network-18 layers) là một mạng nơ-ron sâu gồm 18 lớp (layers) và sử dụng Residual Blocks (Khối dư thừa) để giải quyết vấn đề vanishing gradient khi huấn luyện các mạng nơ-ron sâu. Mô hình này được thiết kế bởi Kaiming He và các cộng sự trong bài báo "Deep Residual Learning for Image Recognition" (2015), và là một trong những mô hình mạng sâu phổ biến trong các bài toán nhận diện hình ảnh.

Cấu trúc của ResNet-18 có thể được mô tả dưới đây, bao gồm các thành phần chính:

Phần đầu vào (Input Layer):

Kích thước đầu vào: 300 x 300 x 3 (Hình ảnh đầu vào có kích thước 300x300 pixels và 3 kênh màu RGB).

Convolutional Layer (Conv1):

Một lớp convolution với bộ lọc (kernel) kích thước 7x7, bước nhảy (stride) bằng 2 và padding 3.

Số bộ lọc (filters): 64.

Output size: 112 x 112 x 64.

Sau đó có một Batch Normalization và ReLU activation.

Max Pooling Layer:

Kernel size: 3x3, stride: 2.

Output size: 56 x 56 x 64.

Residual Blocks:

ResNet-18 được cấu trúc từ các Residual Blocks (Khối dư thừa), mỗi khối bao gồm các lớp convolution, batch normalization và ReLU. Những khối này có thể kết nối trực tiếp với nhau thông qua các skip connections (kết nối nhảy) để giảm thiểu vấn đề gradient vanishing.

ResNet-18 có tổng cộng 4 giai đoạn với mỗi giai đoạn có một số Residual Blocks:

Stage 1: 2 residual blocks với mỗi block có 2 lớp convolution (kích thước 3x3, bước nhảy 1, bộ lọc 64).

Stage 2: 2 residual blocks với mỗi block có 2 lớp convolution (kích thước 3x3, bước nhảy 1, bộ lọc 128).

Stage 3: 2 residual blocks với mỗi block có 2 lớp convolution (kích thước 3x3, bước nhảy 2, bộ lọc 256).

Stage 4: 2 residual blocks với mỗi block có 2 lớp convolution (kích thước 3x3, bước nhảy 2, bộ lọc 512).

Mỗi Residual Block bao gồm:

Một hoặc hai lớp convolution 3x3 (hoặc với bước nhảy 2 trong trường hợp cần giảm kích thước không gian).

Một shortcut connection (kết nối nhảy) để cộng trực tiếp đầu ra của lớp convolution vào đầu vào của block, giúp gradient được truyền qua mà không bị mất mát.

Global Average Pooling:

Sau khi đi qua các residual blocks, mô hình sử dụng Global Average Pooling thay vì Fully Connected layers truyền thống. Đây là một cách để giảm số lượng tham số và tránh overfitting, bằng cách tính giá trị trung bình của mỗi kênh trong feature map của lớp cuối cùng.

Output size: 512

Fully Connected Layer (FC):

Số lớp đầu ra: 8

Activation: Softmax hoặc Sigmoid

A graph with numbers and squares

Description automatically generated

Hình 3.2. Ma trận nhầm lẫn thể hiện hiệu suất phân loại của mô hình trên từng lớp dữ liệu

Quá trình huấn luyện chi tiết

Quá trình huấn luyện của mô hình được giám sát chặt chẽ nhằm đảm bảo hiệu suất tối ưu, đồng thời tránh các vấn đề thường gặp như overfitting. Dưới đây là các điểm nổi bật về diễn biến của quá trình huấn luyện:

Loss

Xu hướng: Biểu đồ cho thấy giá trị mất mát có xu hướng giảm dần qua các epoch. Bắt đầu từ khoảng 0.0050, giá trị mất mát giảm xuống dưới 0.0030 vào cuối quá trình, cho thấy mô hình đang học hiệu quả hơn.

Biến động: Mặc dù có một số dao động nhẹ giữa các epoch, nhưng nhìn chung, mức độ giảm đều và ổn định. Điều này cho thấy rằng mô hình đang điều chỉnh tốt để tối ưu hóa hiệu suất.

Accuracy

Xu hướng: Độ chính xác của mô hình có xu hướng tăng lên liên tục từ epoch 1 đến epoch 10, bắt đầu từ khoảng 91% và đạt gần 97% ở epoch cuối. Điều này cho thấy mô hình đang học và cải thiện khả năng phân loại.

Ổn định: Đường cong độ chính xác cho thấy sự gia tăng nhẹ nhàng mà không có sự sụt giảm đáng kể nào. Điều này là tốt, vì nó cho thấy mô hình không gặp vấn đề về quá khớp (overfitting).

Tổng quan quá trình huấn luyện

Hiệu quả: Sự giảm dần của mất mát và sự tăng lên của độ chính xác cho thấy quá trình huấn luyện diễn ra hiệu quả. Mô hình đang học tốt từ dữ liệu và cải thiện khả năng phân loại của mình.

Khả năng cải thiện: Mặc dù độ chính xác đạt gần 97%, có thể xem xét khả năng tinh chỉnh thêm siêu tham số hoặc sử dụng các kỹ thuật như tăng cường dữ liệu (data augmentation) để tối ưu hơn nữa.

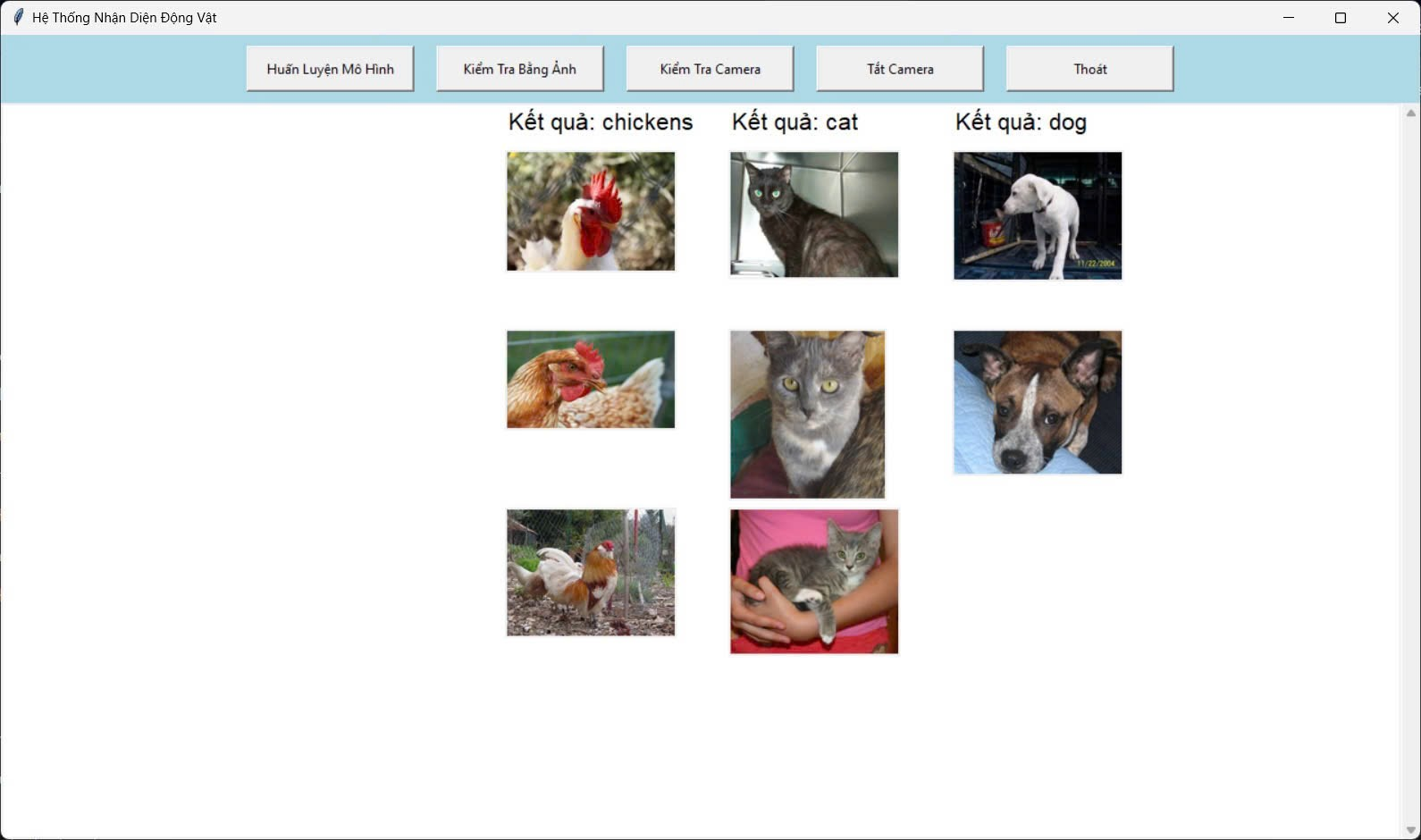
Kết luận về quá trình huấn luyện

Quá trình huấn luyện cho thấy các chỉ số tích cực về cả độ mất mát và độ chính xác. Nếu không có hiện tượng quá khớp trong các epoch tiếp theo, mô hình có thể được triển khai với hiệu suất cao trên các tập dữ liệu thực tế. Vẫn có thể theo dõi thêm các chỉ số khác như độ chính xác xác thực (validation accuracy) để đảm bảo tính tổng quát của mô hình.

A graph of a line

Description automatically generated with medium confidence

Hình 3.. Đồ thị loss và accuracy



Hình 3.. Test bằng hình ảnh

A screenshot of a computer

Description automatically generated

Hình 3.. Test phân loại ảnh loài mèo

# KẾT LUẬN

**Kết quả đạt được**

Sau khi hoàn thành bài tập lớn với đề tài "Xây dựng hệ thống phát triển mô hình phân loại ảnh để nhận diện các loài động vật", nhóm đã đạt được nhiều kết quả đáng ghi nhận. Trước hết, hệ thống phân loại ảnh đã được xây dựng và triển khai thành công với khả năng nhận diện các loài động vật từ hình ảnh một cách ổn định và chính xác, dựa trên các mô hình học sâu tiên tiến như CNN. Hệ thống được thiết kế với giao diện thân thiện, dễ sử dụng, và xử lý dữ liệu nhanh chóng. Bên cạnh đó, nhóm đã xây dựng một bộ dữ liệu hình ảnh động vật phong phú và đa dạng, được chuẩn hóa và tối ưu hóa để phục vụ hiệu quả cho việc huấn luyện mô hình. Quá trình nghiên cứu cũng bao gồm việc tối ưu hóa mô hình, từ lựa chọn kiến trúc phù hợp, tinh chỉnh siêu tham số đến áp dụng các kỹ thuật tăng cường dữ liệu, giúp đạt được hiệu suất cao với các chỉ số như độ chính xác và F1-score. Đồng thời, nhóm đã hoàn thiện báo cáo chi tiết về toàn bộ quá trình thực hiện, bao gồm phân tích kết quả và đề xuất các hướng phát triển trong tương lai như mở rộng bộ dữ liệu, tích hợp thêm tính năng nâng cao, hoặc ứng dụng thực tiễn. Những kết quả này không chỉ khẳng định tính khả thi của đề tài mà còn góp phần mở ra nhiều tiềm năng ứng dụng trong thực tế, đặc biệt trong các lĩnh vực liên quan đến bảo tồn động vật và nghiên cứu khoa học.

**Hướng phát triển**

Dựa trên những kết quả đạt được, đề tài "Xây dựng hệ thống phát triển mô hình phân loại ảnh để nhận diện các loài động vật" có thể được tiếp tục phát triển theo nhiều hướng tiềm năng. Trước hết, việc mở rộng bộ dữ liệu với nhiều hình ảnh từ các nguồn và khu vực địa lý khác nhau sẽ tăng tính đa dạng và cải thiện khả năng nhận diện các loài đặc thù. Bên cạnh đó, nâng cao hiệu quả của mô hình bằng cách áp dụng các thuật toán tiên tiến như Vision Transformers (ViT) hoặc mô hình ensemble sẽ giúp tăng độ chính xác và tối ưu hóa thời gian xử lý, đặc biệt trên các thiết bị cấu hình hạn chế. Hệ thống cũng có thể được bổ sung các tính năng nâng cao như nhận diện hành vi động vật từ video hay cung cấp thông tin chi tiết về đặc điểm sinh học, trạng thái bảo tồn của từng loài. Ngoài ra, việc tích hợp hệ thống vào các ứng dụng di động hoặc nền tảng web sẽ hỗ trợ hiệu quả cho người dùng phổ thông, nhà nghiên cứu, và các tổ chức bảo tồn. Một hướng quan trọng khác là phát triển khả năng học thêm loài mới mà không cần huấn luyện lại toàn bộ mô hình, đồng thời xây dựng cơ sở dữ liệu mở để cộng đồng cùng đóng góp và cải thiện hệ thống. Những định hướng này không chỉ giúp nâng cao chất lượng hệ thống mà còn mở rộng phạm vi ứng dụng, góp phần giải quyết các vấn đề thực tiễn trong bảo tồn và nghiên cứu động vật.

**Hạn chế**

Nhóm em đã gặp phải một số hạn chế nhất định, trước hết là bộ dữ liệu sử dụng để huấn luyện và đánh giá mô hình còn giới hạn về số lượng và tính đa dạng. Điều này dẫn đến khả năng nhận diện của hệ thống chưa thực sự tối ưu khi gặp các hình ảnh thuộc loài động vật ít phổ biến hoặc trong điều kiện ánh sáng và môi trường phức tạp. Bên cạnh đó, mô hình vẫn yêu cầu tài nguyên tính toán lớn, khiến việc triển khai trên các thiết bị có cấu hình thấp hoặc sử dụng trong thời gian thực trở nên khó khăn. Một hạn chế khác là hệ thống hiện tại mới chỉ tập trung vào việc phân loại loài động vật từ hình ảnh tĩnh, chưa hỗ trợ nhận diện hành vi động vật hoặc phân tích từ video. Cuối cùng, việc tối ưu hóa các siêu tham số và kiến trúc mạng đòi hỏi thời gian và kinh nghiệm, dẫn đến những giới hạn trong việc đạt hiệu suất tối ưu cho mô hình. Những hạn chế này là động lực để nhóm tiếp tục cải tiến và phát triển hệ thống trong tương lai.

**Lời kết thúc**

Qua quá trình nghiên cứu, thiết kế, và triển khai hệ thống, nhóm đã không chỉ đạt được các mục tiêu ban đầu mà còn tích lũy được nhiều kiến thức và kinh nghiệm quý báu trong lĩnh vực trí tuệ nhân tạo và xử lý ảnh.

Dù vẫn còn một số hạn chế cần khắc phục, hệ thống đã khẳng định tính khả thi và mở ra nhiều cơ hội phát triển, đặc biệt trong các lĩnh vực như bảo tồn động vật, quản lý tài nguyên thiên nhiên, và nghiên cứu khoa học. Nhóm hi vọng rằng đề tài này sẽ trở thành nền tảng cho các nghiên cứu tiếp theo, góp phần áp dụng công nghệ vào thực tiễn một cách hiệu quả hơn.

Chúng tôi xin gửi lời cảm ơn chân thành đến các thầy cô và bạn bè đã hỗ trợ trong suốt quá trình thực hiện. Rất mong nhận được những ý kiến đóng góp để hoàn thiện hơn nữa sản phẩm cũng như các nghiên cứu liên quan trong tương lai.

# TÀI LIỆU THAM KHẢO

1. Nhựt Liên (2024), *Python là gì*, FPT Shop
2. Admin (2022), *OpenCV là gì*, Tek4
3. Tek4 (2024), *Thuật toán tối ưu Adam*, Tek4
4. Corrado Alessio (2019), *Animals-10*, Kaggle