ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA TOÁN - TIN HỌC

MÔN HỌC: PHÂN TÍCH VÀ XỬ LÝ ẢNH

PHÂN LOẠI ĐỘNG VẬT DỰA TRÊN MÔ HÌNH HỌC SÂU
ANIMAL CLASSIFICATION USING DEEP LEARNING

Giảng viên: **Huỳnh Thanh Sơn**
Sinh viên thực hiện: Lê Đặng Gia Khánh 22110081
Trần Anh Quốc 22110178
Nguyễn Bách Sơn 22110187

Tp. Hồ Chí Minh, Tháng 01/2025

Mục lục

Tóm tắt

Phân loại động vật đóng vai trò quan trọng trong nhiều lĩnh vực của cuộc sống hiện đại, đặc biệt là trong việc quản lý tài nguyên thiên nhiên, nghiên cứu sinh thái, và bảo tồn động vật hoang dã. Đây là một phương pháp được ứng dụng rộng rãi trong các hệ thống nhận dạng tự động, hỗ trợ nghiên cứu khoa học và giáo dục. Với sự phát triển mạnh mẽ của công nghệ, nhu cầu về các hệ thống phân loại động vật chính xác và hiệu quả ngày càng gia tăng, đặc biệt khi xử lý các bộ dữ liệu lớn và phức tạp. Giai đoạn đầu tiên của quá trình nghiên cứu thường tập trung vào các phương pháp truyền thống, sử dụng các giải thuật xử lý ảnh cơ bản để trích xuất đặc trưng. Tuy nhiên, các phương pháp này, dù đạt được một số kết quả khả quan, vẫn còn hạn chế về độ chính xác và khả năng xử lý dữ liệu phức tạp, như ảnh có nền nhiễu hoặc các loài động vật có đặc điểm tương đồng.

Trong thời gian gần đây, việc áp dụng các mô hình học sâu đã mở ra những hướng tiếp cận mới trong bài toán phân loại động vật. Các mô hình học sâu, đặc biệt là các mạng nơ-ron tích chập (CNN), với khả năng tự động trích xuất các đặc trưng từ dữ liệu và xử lý tốt các vấn đề phi tuyến tính, đã đạt được nhiều thành tựu trong việc tăng độ chính xác và khả năng tổng quát hóa của hệ thống. Tuy nhiên, vẫn còn những thách thức trong việc lựa chọn kiến trúc mô hình phù hợp và tối ưu hóa tham số để đạt hiệu suất tốt nhất. Các nghiên cứu so sánh giữa các mô hình học sâu hiện đại đã chỉ ra rằng, quá trình trích xuất đặc trưng từ ảnh động vật và việc kết hợp chúng với các kỹ thuật phân loại tiên tiến đóng vai trò then chốt trong việc cải thiện hiệu quả hệ thống.

Bài báo cáo này tập trung vào việc thử nghiệm và đánh giá hiệu suất của các mô hình học sâu trong bài toán phân loại động vật. Cụ thể, các đặc trưng của ảnh động vật được tự động trích xuất thông qua mạng nơ-ron tích chập, sau đó được sử dụng làm đầu vào cho các thuật toán phân loại để dự đoán loài động vật. Kết quả nghiên cứu cho thấy, cách tiếp cận này không chỉ cải thiện độ chính xác mà còn tăng khả năng mở rộng, hứa hẹn ứng dụng rộng rãi trong nhiều lĩnh vực liên quan đến nhận diện và phân loại động vật.

1 Giới thiệu bài toán phân loại động vật

Phân loại động vật là một trong những bài toán phổ biến và quan trọng, được áp dụng rộng rãi trong nhiều lĩnh vực như nghiên cứu khoa học, bảo tồn thiên nhiên, và giáo dục. Việc phân loại các loài động vật dựa trên hình ảnh đóng vai trò quan trọng trong việc quản lý tài nguyên sinh học, phân tích môi trường, cũng như hỗ trợ các hệ thống nhận dạng tự động. Tuy nhiên, bài toán phân loại động vật không hề đơn giản, đặc biệt khi xử lý dữ liệu lớn và đa dạng, với những loài động vật có hình thái hoặc đặc điểm tương đồng. Từ đó, việc xây dựng một hệ thống tự động phân loại động vật một cách nhanh chóng và chính xác trở thành một thách thức quan trọng.

Đặc điểm hình thái của động vật là một trong những yếu tố chính khiến bài toán phân loại trở nên khó khăn. Các yếu tố như kích thước, hình dáng, màu sắc, hoặc cấu trúc cơ thể của động vật có thể thay đổi đáng kể tùy thuộc vào điều kiện ánh sáng, góc chụp, hay môi trường sống. Đôi khi, các loài động vật khác nhau có thể sở hữu những đặc điểm tương tự, làm tăng mức độ phức tạp trong việc phân loại. Một hệ thống phân loại hiệu quả cần phải có khả năng trích xuất những đặc trưng riêng biệt của từng loài, đồng thời phân biệt được những khác biệt nhỏ giữa các loài động vật.

Nhìn chung, có nhiều cách tiếp cận để giải quyết bài toán phân loại động vật, trong đó có thể chia thành hai hướng chính: phương pháp truyền thống và phương pháp học sâu dựa trên mạng nơ-ron tích chập (CNN). Trong phương pháp truyền thống, ảnh động vật được xử lý và trích xuất các đặc trưng bằng cách sử dụng các công cụ như biến đổi Wavelet, Fourier, hoặc phân tích histogram. Những đặc trưng này có thể bao gồm hình dạng tổng thể, màu sắc, hoặc cấu trúc vùng cụ thể của ảnh. Sau đó, các thuật toán so sánh như khoảng cách Euclidean hoặc cosine similarity được sử dụng để đánh giá mức độ tương đồng giữa đặc trưng của ảnh cần phân loại và các mẫu ảnh đã biết. Tuy nhiên, các phương pháp này thường bị giới hạn về độ chính xác và khó mở rộng cho các tập dữ liệu lớn hoặc phức tạp.

Trong khi đó, các phương pháp học sâu, đặc biệt là mạng nơ-ron tích chập (CNN), đã cho thấy tiềm năng vượt trội trong việc giải quyết bài toán phân loại động vật. Với khả năng tự động trích xuất các đặc trưng phi tuyến tính và phức tạp từ ảnh, CNN không chỉ cải thiện độ chính xác mà còn giảm thiểu sự phụ thuộc vào các bước tiền xử lý. Một số nghiên cứu gần đây đã kết hợp việc trích xuất đặc trưng từ mạng CNN với các thuật toán phân loại tiên tiến để đạt được hiệu quả cao hơn. Bài báo cáo này sẽ tập trung vào việc phân tích và đánh giá các phương pháp học sâu trong bài toán phân loại động vật, cụ thể là sử dụng các mô hình CNN hiện đại và quy trình tiền xử lý phù hợp để nâng cao độ chính xác và tính ổn định của hệ thống.

2 Xây dựng mô hình phân loại động vật

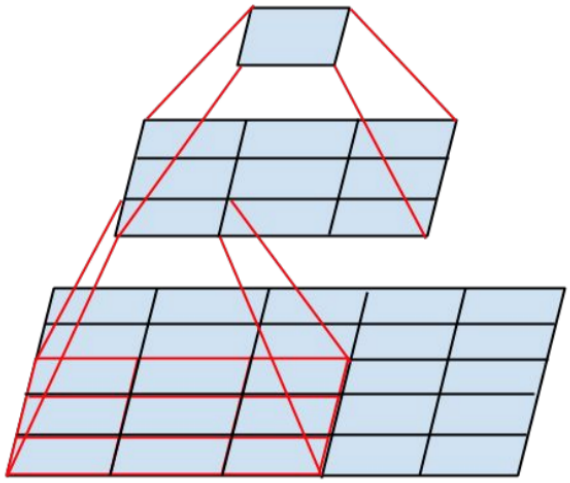
Phần này sẽ trình bày cụ thể mô hình học sâu được sử dụng để thực hiện bài toán phân loại động vật trong bài báo cáo này. Mô hình được xây dựng dựa trên những phương pháp nổi bật gần đây trong lĩnh vực học sâu, đặc biệt là các mạng nơ-ron tích chập (CNN). Một số phương pháp được áp dụng từ các bài toán phân loại hình ảnh lớn như ImageNet, đồng thời tích hợp các cải tiến phù hợp với bài toán phân loại động vật để đảm bảo hiệu quả và độ chính xác cao.

Dữ liệu huấn luyện và kiểm tra được tổ chức dưới dạng thư mục, trong đó mỗi thư mục con đại diện cho một loài động vật, chứa các ảnh thuộc về loài đó. Việc nạp và xử lý dữ liệu được thực hiện thông qua ImageDataGenerator, một công cụ tạo dữ liệu động, cho phép chuẩn hóa ảnh, tăng cường dữ liệu (Data Augmentation), và nạp dữ liệu theo từng batch trong quá trình huấn luyện. Generator tự động cung cấp dữ liệu dưới dạng các batch (x_batch, y_batch), trong đó:

- x_batch: Batch ảnh đã được resize và chuẩn hóa (kích thước $224 \times 224 \times 3$).
- y_batch: Batch nhãn được mã hóa dạng one-hot, tương ứng với các loài động vật.

Mô hình học sâu sử dụng kiến trúc **Inception v3** làm mạng cơ sở (base model). Đây là một mạng nơ-ron tích chập tiên tiến, được thiết kế để trích xuất các đặc trưng phức tạp từ hình ảnh. Phần đầu ra cuối cùng của Inception v3 được thay thế bằng một lớp fully connected (Dense layer) với số lượng nút bằng số loài động vật cần phân loại, sử dụng hàm kích hoạt **softmax** để dự đoán xác suất cho từng lớp.

Để có cái nhìn tổng quát hơn về mô hình **Inception v3**, ta sẽ tìm hiểu các khái niệm sau.
Trước tiên, ta sẽ tìm hiểu về **Phân tách tích chập với kích thước lớn** (Factorized Convolutions). Phân tách các tích chập với bộ lọc lớn nhằm tăng hiệu quả tính toán của mạng. Việc sử dụng các kỹ thuật phân tách giúp giảm số lượng tham số và chi phí tính toán trong khi vẫn duy trì hoặc cải thiện hiệu suất của mạng. Ở đây, ta sẽ chỉ tìm hiểu về Phân tách thành các tích chập nhỏ hơn (Smaller convolutions).



Hình 1: **Mạng con thay thế tích chập 5×5.**

Hình ảnh này minh họa một mạng con (mini-network) được sử dụng để thay thế một phép tích chập 5×5. Thay vì thực hiện một phép tích chập lớn 5×5 vốn tốn nhiều chi phí tính toán, hình minh họa cho thấy cách thay thế nó bằng hai tầng tích chập 3×3 nhỏ hơn.

Phân tích chi phí tính toán
Để phân tích sự tiết kiệm chi phí tính toán kỳ vọng, chúng ta thực hiện một số giả định đơn giản hóa áp dụng cho các tình huống thông thường: Giả sử $n = \alpha m$, nghĩa là số lượng kích hoạt trên mỗi đơn vị thay đổi theo một hệ số không đổi α. Vì tích chập 5×5 là tích lũy (aggregating), α thường lớn hơn một chút so với 1 (khoảng 1.5 trong trường hợp GoogLeNet). Khi thay thế tầng 5×5 bằng hai tầng, chúng ta có thể đạt được sự mở rộng này trong hai bước: tăng số lượng bộ lọc theo căn bậc hai của α trong mỗi bước. Để đơn giản hóa, chọn $\alpha = 1$ (không mở rộng). Nếu ta trượt một mạng mà không tận dụng việc tính toán lại giữa các ô lân cận, chi phí tính toán sẽ tăng lên. Tuy nhiên, với hai tầng tích chập 3×3, ta tái sử dụng các kích hoạt giữa các ô lân cận, dẫn đến:

$$\text{Chi phí tính toán} = \frac{9+9}{25}, \text{tức là tiết kiệm } 28\%.$$

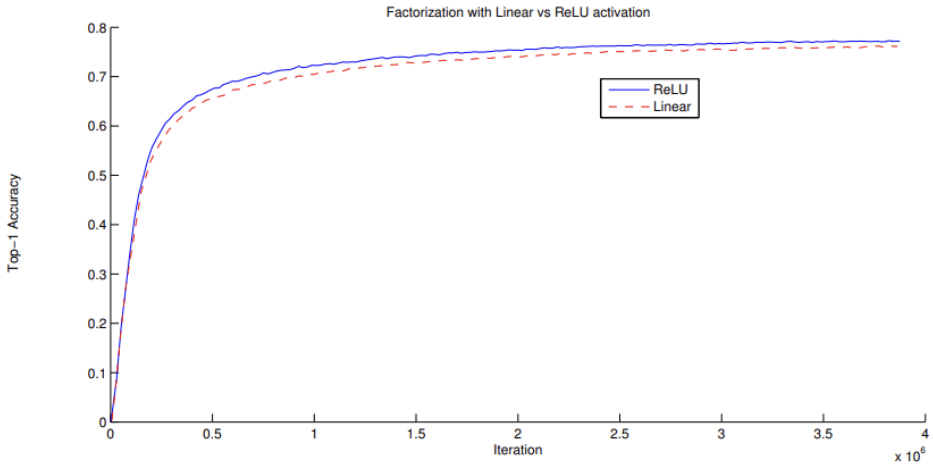
Ở đây, **chúng ta có thể đặt câu hỏi liệu một tích chập 5 x 5 có thể được thay thế bằng một mạng nhiều tầng với ít tham số hơn nhưng vẫn giữ nguyên kích thước đầu vào và đầu ra không ?**

Nếu phân tích biểu đồ tính toán của tích chập 5 x 5, ta sẽ thấy rằng mỗi đầu ra giống như một mạng fully-connected nhỏ đang trượt qua các ô 5×5 trên đầu vào (xem Hình 1).
Khi thiết kế một mạng thị giác, việc tận dụng tính chất bất biến dịch chuyển là điều tự nhiên, vì vậy ta có thể thay thế thành phần fully-connected bằng một kiến trúc tích chập hai tầng:

- Tầng đầu tiên là tích chập 3×3,
- Tầng thứ hai là một lớp fully-connected trên lưới đầu ra của tầng 3×3 đầu tiên (xem hình 1).

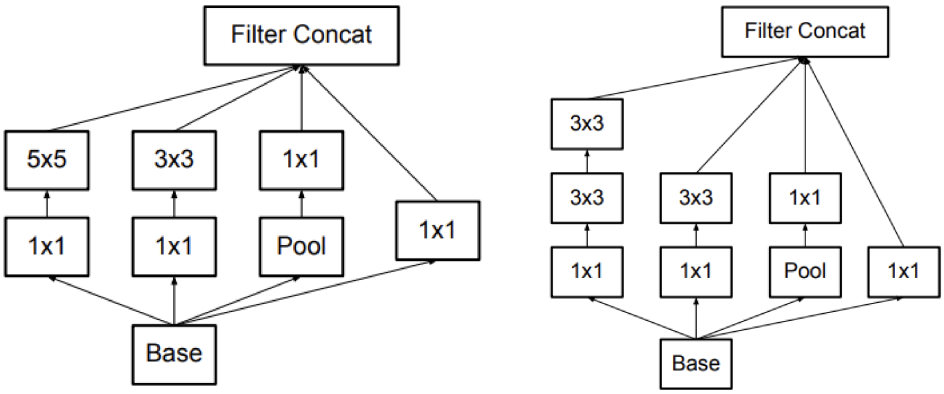
Để trả lời cho câu hỏi ở trên, ta tiến hành xem xét về kết quả thí nghiệm ở hình 2. Kết quả thí nghiệm cho thấy qua các thí nghiệm kiểm soát, kết quả cho thấy không có sự mất mát đáng kể. Các nhà thí nghiệm cho rằng, sự cải thiện này xuất phát từ không gian biến đổi rộng hơn mà mạng có thể học được, đặc biệt khi đầu ra được chuẩn hóa theo batch. Hiệu ứng tương tự cũng được quan sát thấy khi sử dụng

hàm kích hoạt tuyến tính cho các thành phần giảm chiều.



Hình 2: **Một trong số các thí nghiệm kiểm soát được thực hiện giữa hai mô hình Inception, trong đó một mô hình sử dụng phân tách thành các tầng kết hợp giữa hàm tuyến tính và ReLU, còn mô hình kia sử dụng hai tầng ReLU.**

Kết luận: việc thay thế 5×5 bằng hai tầng 3×3 là cách hiệu quả để giảm chi phí tính toán và tham số, trong khi vẫn duy trì khả năng biểu diễn của mạng (xem hình 3).



Hình 3: **Mô hình inception ban đầu và Mô hình Inception với mỗi 5×5 thay bằng hai 3×3.**

Tiếp theo, ta sẽ tìm hiểu đến **Tích chập bất đối xứng** (Asymmetric convolutions). Tích chập bất đối xứng là một kỹ thuật được sử dụng trong các mạng nơ-ron tích chập (Convolutional Neural Networks - CNN) để thay thế các tích chập đối xứng ($n \times n$) bằng các tổ hợp tích chập có kích thước khác nhau, chẳng hạn thay thế một bộ lọc 3×3 bằng hai bộ lọc bất đối xứng:

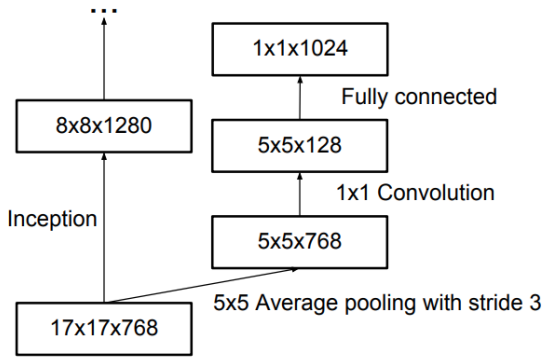
- Một bộ lọc 1×3 (tích chập theo chiều ngang).
- Một bộ lọc 3×1 (tích chập theo chiều dọc).

Kỹ thuật này được thiết kế nhằm giảm chi phí tính toán và số lượng tham số trong mô hình, đồng thời vẫn giữ được khả năng biểu diễn đặc trưng của mạng.

Vậy, tích chập 3×3 có thể được thay thế bằng tích chập 1×3 theo sau là tích chập 3×1. Nếu tích chập 3×3 được thay thế bằng tích chập 2×2, số lượng tham số sẽ cao hơn một chút so với tích chập bất đối xứng được đề xuất. (xem hình 4)

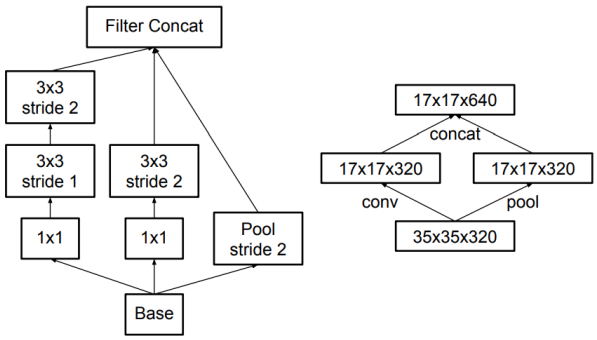
Hình 4: **Mô hình inception ban đầu và Mô hình Inception với với bộ lọc đầu ra được mở rộng.**

Tiếp đến, ta tìm hiểu về **Bộ phân loại phụ trợ** (Auxiliary classifier) là một khái niệm được giới thiệu trong các mạng nơ-ron sâu để cải thiện quá trình huấn luyện và hội tụ của mạng. Bộ phân loại phụ trợ là một CNN nhỏ được chèn giữa các lớp trong quá trình đào tạo và tổn thất phát sinh được thêm vào tổn thất mạng chính. Trong GoogLeNet, bộ phân loại phụ trợ được sử dụng cho mạng sâu hơn, trong khi trong Inception v3, bộ phân loại phụ trợ hoạt động như một bộ điều chỉnh.



Hình 5: **Bộ phân loại phụ (Auxiliary classifier) trên đỉnh của tầng 17×17 cuối cùng.**

Cuối cùng, ta tìm hiểu **Giảm kích thước lưới** (Grid Size Reduction) là một kỹ thuật quan trọng trong các mạng nơ-ron tích chập (Convolutional Neural Networks - CNN) được sử dụng để giảm kích thước không gian (spatial dimensions) của các đặc trưng (feature maps) khi mạng tiến sâu hơn.



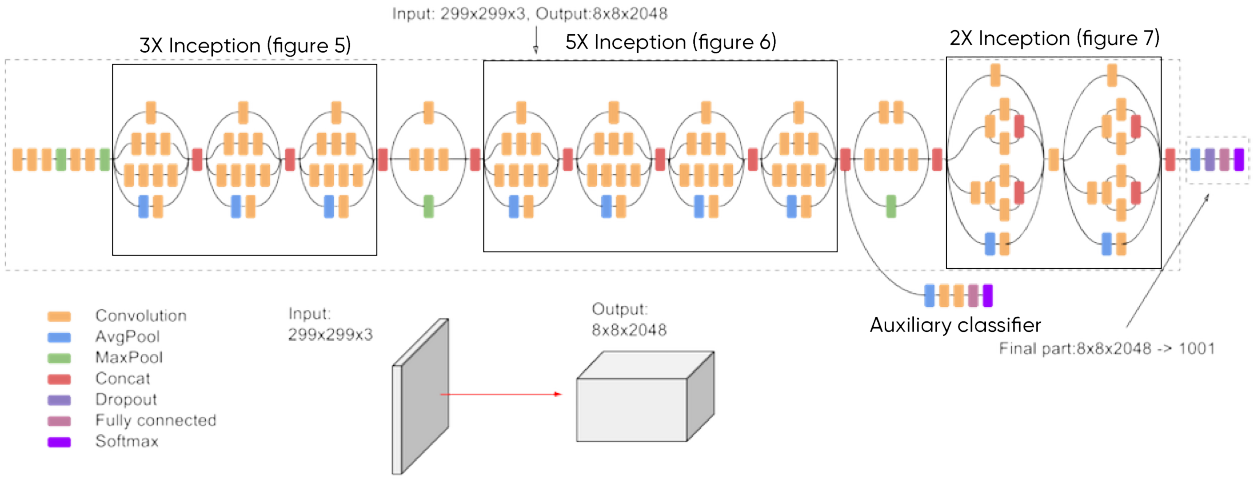
Hình 6: **Mô-đun Inception giảm kích thước lưới đồng thời mở rộng bộ lọc.**

Mô hình Inception-v3

Từ tất cả các khái niệm trên, ta rút ra được sơ đồ tổng quát của mô hình Inception-v3 như sau:

type	patch size/stride or remarks	input size
conv	3×3/2	299×299×3
conv	3×3/1	149×149×32
conv padded	3×3/1	147×147×32
pool	3×3/2	147×147×64
conv	3×3/1	73×73×64
conv	3×3/2	71×71×80
conv	3×3/1	35×35×192
3×Inception	As in figure 5	35×35×288
5×Inception	As in figure 6	17×17×768
2×Inception	As in figure 7	8×8×1280
pool	8 × 8	8 × 8 × 2048
linear	logits	1 × 1 × 2048
softmax	classifier	1 × 1 × 1000

Hình 7: **The outline of the proposed network architecture.**



Hình 8: **Mô hình Inception-v3.**

Dựa vào hình trên, ta có thể thấy rằng đầu vào của mô hình là ảnh có kích thước là 299x299 và đầu ra là 1001 nhãn, vì đây là mô hình đã train và output không phù hợp với đầu ra của bài toán của em nên em sẽ xóa **Fully connected** và **Softmax** của mô hình, sau đó thay thế bằng yêu cầu bài toán của em là 10 nhãn loài vật đang xét. Vậy đầu vào sẽ là ảnh có kích thước 299x299 và đầu ra sẽ là 10 nhãn loài vật

3 Thực nghiệm và kết quả

3.1 Dữ liệu

Tập dữ liệu Animals-10 (`https://www.kaggle.com/datasets/alessiocorrado99/animals10/data`), bộ dữ liệu này chứa khoảng 28.000 hình ảnh động vật có chất lượng trung bình thuộc 10 loại: chó, mèo, ngựa, nhện, bướm, gà, cừu, bò, sóc, voi. Tất cả các hình ảnh đều được thu thập từ "google images" và đã được kiểm tra bởi con người. Có một số dữ liệu sai để mô phỏng các điều kiện thực tế (ví dụ, hình ảnh được người dùng của ứng dụng chụp). Thư mục chính được chia thành các thư mục con, mỗi thư mục chứa các hình ảnh cho từng loại. Số lượng hình ảnh cho mỗi loại dao động từ 2.000 đến 5.000 hình. Bộ dữ liệu này được chia thành ba tập con: training-validation-testing lần lượt ứng với 80%-10%-10%.

Trong quá trình kiểm tra, ta luôn sử dụng tập testing (10%) để đảm bảo năm mô hình được đánh giá một cách khách quan.

3.2 Tăng Cường Dữ Liệu (Data Augmentation)

Để cải thiện độ chính xác của mô hình và giảm thiểu tình trạng *overfitting*, em đã sử dụng kỹ thuật *tăng cường dữ liệu* để mở rộng bộ dữ liệu huấn luyện. Kỹ thuật này giúp tạo ra nhiều biến thể khác nhau của các hình ảnh trong bộ dữ liệu huấn luyện, nhằm nâng cao khả năng tổng quát của mô hình.

Cụ thể, em đã sử dụng lớp `ImageDataGenerator` từ thư viện Keras để thực hiện tăng cường dữ liệu. Các tham số được sử dụng trong quá trình tăng cường dữ liệu bao gồm:

- **Rescale**: Để chuẩn hóa giá trị pixel của hình ảnh, em đã chia tất cả các giá trị pixel cho 255, giúp đưa các giá trị về khoảng [0, 1]. Điều này giúp mô hình học dễ dàng hơn và đạt hiệu quả cao hơn.
- **Zoom Range**: Để tạo ra sự biến đổi hình ảnh, tham số `zoom_range=0.2` cho phép phóng to hoặc thu nhỏ hình ảnh ngẫu nhiên trong khoảng từ 80% đến 120% so với kích thước gốc. Điều này giúp mô hình học được các đặc trưng của đối tượng ở các tỷ lệ khác nhau.
- **Horizontal Flip**: Tham số `horizontal_flip=True` giúp tạo ra các biến thể đối xứng theo chiều ngang của hình ảnh, giúp mô hình trở nên mạnh mẽ hơn trong việc nhận dạng các đối tượng không phụ thuộc vào hướng của chúng.

Cụ thể, các bộ dữ liệu huấn luyện, kiểm tra và xác thực được xử lý như sau:

- **Huấn luyện**: Đối với bộ dữ liệu huấn luyện, em đã sử dụng `ImageDataGenerator` với các kỹ thuật tăng cường dữ liệu để tạo ra các biến thể ngẫu nhiên của hình ảnh trong mỗi lần huấn luyện.

```
train_generator = ImageDataGenerator(rescale=1/255.,
                                     zoom_range=0.2,
                                     horizontal_flip=True)
```

- **Xác thực và Kiểm tra**: Đối với bộ dữ liệu xác thực và kiểm tra, em chỉ áp dụng việc *chuẩn hóa* ảnh mà không sử dụng các phép biến đổi khác, vì các bộ dữ liệu này sẽ được sử dụng để kiểm tra hiệu quả mô hình với dữ liệu chưa được thay đổi.

```
valid_generator = ImageDataGenerator(rescale=1/255.)
test_generator = ImageDataGenerator(rescale=1/255.)
```

Việc sử dụng các kỹ thuật tăng cường dữ liệu này giúp mô hình có khả năng tổng quát tốt hơn và giảm thiểu khả năng *overfitting*, đồng thời giúp mô hình nhận diện được các đối tượng trong nhiều điều kiện khác nhau.

3.3 Huấn Luyện Mô Hình (Model Training)

Để huấn luyện mô hình phân loại động vật, em sử dụng mô hình `InceptionV3` làm cơ sở, với trọng số đã được huấn luyện sẵn từ `imagenet`. Điều này giúp giảm thiểu thời gian huấn luyện và cải thiện độ chính xác nhờ vào việc tái sử dụng các đặc trưng học được từ một mô hình đã được huấn luyện trên một lượng dữ liệu lớn.

3.3.1 Cấu Trúc Mô Hình

Em không sử dụng lớp đầu ra của `InceptionV3` mà thay vào đó thêm các lớp phân loại vào mô hình. Cấu trúc của mô hình được xây dựng như sau:

- **Global Average Pooling**: Sau lớp InceptionV3, em thêm một lớp `GlobalAveragePooling2D` để giảm kích thước của tensor đầu ra và giúp giảm số lượng tham số, từ đó làm giảm overfitting.
- **Dense Layer**: Tiếp theo, em thêm một lớp `Dense` với 512 đơn vị và sử dụng `ReLU` làm hàm kích hoạt. Lớp này giúp mô hình học các đặc trưng phi tuyến tính.
- **Softmax Layer**: Lớp `Dense` cuối cùng có 10 đơn vị (tương ứng với 10 lớp động vật) và sử dụng hàm kích hoạt `Softmax` để chuyển đổi đầu ra thành xác suất cho từng lớp.

Mô hình hoàn chỉnh có đầu vào từ `InceptionV3`, nhưng với lớp đầu ra thay thế bằng các lớp phân loại mà em đã tạo.

3.3.2 Tối Ưu Hóa Mô Hình

Để tối ưu hóa mô hình, em sử dụng **Adam optimizer**, một thuật toán tối ưu phổ biến và hiệu quả trong các mô hình học sâu. Adam giúp điều chỉnh tốc độ học cho từng tham số trong quá trình huấn luyện. Em sử dụng hàm mất mát `categorical_crossentropy` vì đây là bài toán phân loại đa lớp, và theo dõi độ chính xác (`accuracy`) trong quá trình huấn luyện để đánh giá hiệu quả mô hình.

3.3.3 Sử Dụng Các Callback

Trong quá trình huấn luyện, em sử dụng ba **callback** quan trọng để cải thiện quá trình huấn luyện và tránh overfitting:

- **EarlyStopping**: Callback này giúp dừng quá trình huấn luyện khi độ chính xác trên bộ kiểm tra không cải thiện sau một số epoch nhất định. Điều này giúp ngừng huấn luyện sớm, tránh lãng phí tài nguyên và giảm overfitting.
- **ModelCheckpoint**: Callback này lưu lại mô hình tốt nhất trong quá trình huấn luyện. Mô hình sẽ được lưu mỗi khi độ chính xác trên bộ kiểm tra được cải thiện.
- **ReduceLROnPlateau**: Callback này giảm tốc độ học (`learning rate`) khi độ chính xác trên bộ kiểm tra không cải thiện sau một số epoch nhất định. Điều này giúp cải thiện độ hội tụ của mô hình.

3.4 Cấu Hình Huấn Luyện

Dưới đây là mã Python thể hiện quá trình huấn luyện mô hình:

```
# Dinh nghia mo hinh voi cac lop phan loai
x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(512, activation='relu')(x)
predictions = Dense(10, activation='softmax')(x)
model = Model(inputs=base_model.input, outputs=predictions)

# Bien dich mo hinh
model.compile(
    optimizer='adam',
    loss='categorical_crossentropy',
    metrics=['accuracy']
)
# Ham callback
early_stopping=EarlyStopping(monitor='val_loss',patience=8,verbose=1,
                             restore_best_weights=True)
checkpoints=ModelCheckpoint('inception_v3.keras', monitor='val_loss',
                             save_best_only=True,verbose=1)
reduceonplateau=ReduceLROnPlateau(monitor='val_loss', factor=0.1, patience=5,
                             min_lr=0.001)
callback=[early_stopping,reduceonplateau,checkpoints]
```

```
22
23 # Huan luyen mo hinh
24 history = model.fit(train_data,
25                     epochs=10,
26                     validation_data=valid_data,
27                     callbacks=callback,
28                     batch_size=64)
```

Sau khi huấn luyện xong, em sử dụng các phương pháp trên để đánh giá mô hình và điều chỉnh các tham số học tập cho phù hợp.

3.5 Kết quả

Sau quá trình huấn luyện mô hình và tối ưu hóa, các kết quả đạt được như sau:

- Độ chính xác trên tập kiểm tra (test set): Mô hình đạt được độ chính xác là **0.97** (tương đương 97%), cho thấy khả năng dự đoán chính xác trên dữ liệu chưa từng được mô hình nhìn thấy.

Nhận xét:

- Mô hình có hiệu suất ổn định và không bị overfitting đáng kể.Kết quả này phản ánh rằng mô hình đã học được đặc trưng quan trọng từ dữ liệu và có thể áp dụng cho các bài toán thực tế tương tự.

	precision	recall	f1-score	support
butterfly	0.98	0.97	0.97	187
cat	0.98	0.95	0.97	174
chicken	0.99	0.97	0.98	299
cow	0.94	0.91	0.92	223
dog	0.96	0.99	0.98	493
elephant	0.99	0.98	0.98	147
horse	0.99	0.96	0.97	250
sheep	0.89	0.95	0.92	168
spider	0.99	0.99	0.99	495
squirrel	0.99	0.99	0.99	182
accuracy			0.97	2618
macro avg	0.97	0.97	0.97	2618
weighted avg	0.97	0.97	0.97	2618

Hình 9: **Kết quả sau quá trình huấn luyện**

Thử nghiệm với chữ ký trên tập thử nghiệm (Test set)

Hình 10: **Thử nghiệm mô hình**

4 Kết luận

Trong bài báo cáo này, chúng mình đã xây dựng và triển khai mô hình mạng nơ-ron tích chập (CNN) để giải quyết **bài toán phân loại loài động vật**. Với khả năng tự động học và trích xuất các đặc trưng phức tạp từ hình ảnh, CNN đã được sử dụng làm công cụ chính để phân tích và phân loại các loài động vật. Mô hình đã được huấn luyện trên tập dữ liệu ảnh động vật, bao gồm nhiều loài như chó, mèo, ngựa, bướm, gà, cừu, bò, nhện, sóc và voi, đồng thời kiểm tra hiệu suất trên tập kiểm tra độc lập. Kết quả cho thấy mô hình hoạt động tốt, đạt độ chính xác cao và khả năng tổng quát hóa ổn định khi xử lý các loài động vật khác nhau.

Mô hình CNN đã chứng minh khả năng trích xuất hiệu quả các đặc trưng hình học quan trọng từ các loài động vật, giúp phân biệt giữa các loài động vật một cách chính xác. Kết quả từ các chỉ số đánh giá, bao gồm độ chính xác, độ nhạy và độ đặc hiệu, cho thấy rằng mô hình có thể được áp dụng trong các hệ thống phân loại động vật tự động thực tế với hiệu suất đáng tin cậy. Đặc biệt, khả năng học không giám sát các đặc trưng đã giúp mô hình giảm thiểu sự phụ thuộc vào các quy trình thiết kế thủ công, từ đó tăng cường tính linh hoạt khi áp dụng cho các tập dữ liệu mới.

Tuy nhiên, vẫn còn một số hạn chế cần được cải thiện. Đầu tiên, việc mở rộng tập dữ liệu với nhiều mẫu ảnh động vật từ các nguồn đa dạng, bao gồm ảnh từ nhiều nhóm đối tượng và các điều kiện ánh sáng khác nhau, có thể giúp cải thiện tính tổng quát của mô hình. Thứ hai, việc thử nghiệm các kiến trúc mạng sâu hơn, chẳng hạn như mạng song song (Siamese Network) hoặc tích hợp với các cơ chế chú ý (Attention Mechanism), có thể giúp tăng cường khả năng phân biệt đối với các loài động vật có sự tương đồng lớn về đặc điểm hình thái. Thứ ba, tích hợp thêm các kỹ thuật tiền xử lý hình ảnh trước khi tăng cường và điều chỉnh siêu tham số có thể cải thiện hiệu suất tổng thể của hệ thống.

Ngoài ra, việc triển khai mô hình vào thực tế đòi hỏi tích hợp với các hệ thống bảo mật nhằm tăng cường tính chính xác và độ tin cậy. Điều này đặc biệt quan trọng trong các ứng dụng đòi hỏi sự chính xác cao như bảo vệ động vật hoang dã hoặc các hệ thống nhận diện động vật trong bảo tồn thiên nhiên. Kết hợp CNN với các phương pháp nhận diện khác, chẳng hạn như nhận diện khuôn mặt động vật hoặc dấu vết sinh học, cũng có thể là một hướng đi tiềm năng để tăng cường hiệu quả nhận dạng.

Nhìn chung, nghiên cứu này đã chứng minh rằng CNN là một công cụ mạnh mẽ và hiệu quả để giải quyết bài toán phân loại động vật. Với những cải tiến trong tương lai, hệ thống có thể trở thành một giải pháp toàn diện cho các ứng dụng thực tiễn, góp phần tự động hóa các quy trình phân loại và nâng cao hiệu quả trong việc bảo vệ động vật cũng như hỗ trợ nghiên cứu sinh học.

Tài liệu

Digital Ocean: A Review of Popular Deep Learning Architectures: ResNet, InceptionV3, and SqueezeNet | 10/10/2024
Kaggle: Animals-10 | 2020
Bài báo: Rethinking the Inception Architecture for Computer Vision | 2016