VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
UNIVERSITY OF INFORMATION TECHNOLOGY
COMPUTER SCIENCE

Báo Cáo Đồ Án:
Phân Loại Loài Động Vật

Team members:

Nguyễn Thiên Bảo - 23520127

Phạm Huỳnh Long Vũ - 23521813

Class: CS231.P21.KHTN

Teacher: Mai Tiến Dũng

May 20, 2025

Mục lục

Báo Cáo Đồ Án: Phân Loại Loài Động Vật

1 Giới thiệu chung

Đồ án "Phân loại loài động vật" sử dụng hai kiến trúc CNN nổi bật là VGG16 và ResNet50 nhằm nâng cao độ chính xác và khả năng khái quát của mô hình trên tập dữ liệu chứa bốn loài động vật (`cat`, `deer`, `dog`, `horse`) kết hợp với lớp "other" lấy từ dataset COCO 2017 để phân biệt những hình ảnh không thuộc bốn lớp loài chính đang xét đến.[1]

1.1 Lý do chọn đề tài

Bài toán phân loại loài động vật qua ảnh được lựa chọn vì các lý do chính sau:

Trong bối cảnh ứng dụng trí tuệ nhân tạo vào lĩnh vực bảo tồn sinh thái và du lịch sinh thái, việc xây dựng hệ thống tự động phân loại các loài động vật qua hình ảnh có ý nghĩa quan trọng. Hệ thống này không chỉ hỗ trợ trong giáo dục và du lịch sinh thái như giúp người ta nhận biết thông tin con vật thông qua ảnh,... mà còn hỗ trợ giám sát hệ sinh thái tự động, phát hiện cá thể mới, cá thể quý hiếm hoặc ngoại lai xâm lấn. Cũng như góp phần vào công tác nghiên cứu bảo tồn, cung cấp dữ liệu định lượng về phân bố và mật độ các loài.

2 Phát biểu bài toán (Input, Output)

1. **Input:**
   - Giai đoạn huấn luyện (Train): Tập dữ liệu hình ảnh có chứa động vật đã được gán nhãn (images + labels) và tập dữ liệu dành riêng cho nhãn mới: "other".
   - Giai đoạn kiểm thử (Validation): Bộ ảnh chưa gán nhãn (chưa biết loài) cần dự đoán.
2. **Output:**
   - Hệ thống trả về nhãn (tên loài) động vật được nhận diện xuất hiện trong ảnh hoặc nhãn other nếu không có trong các đối tượng xem xét.
3. **Mục tiêu:**
   (a) Thiết kế pipeline tiền xử lý và augmentation để tăng độ phong phú cho tập dữ liệu, giảm overfitting.
   (b) Tận dụng kiến trúc mạng đã được pre–train trên ImageNet (VGG16, ResNet50) để thực hiện transfer learning.
   (c) So sánh hiệu năng (độ chính xác) giữa hai mô hình trên tập huấn luyện và tập validation.
   (d) Đánh giá ưu nhược điểm, đưa ra nhận xét về khả năng mở rộng và ứng dụng thực tế của mô hình.

[1]4 Animal Classification Dataset, Kaggle. https://www.kaggle.com/competitions/4-animal-classification/data. COCO 2017 Dataset, Kaggle. https://www.kaggle.com/datasets/awsaf49/coco-2017-dataset.

3 Mô tả dữ liệu

1. **4 Animal Classification Dataset**
   - Bao gồm ảnh chứa duy nhất một trong bốn loài: `cat` (mèo), `deer` (hươu), `dog` (chó), `horse` (ngựa).
   - Nguồn: Kaggle.
   - Chia thành hai phần:
     - **Train**: Ảnh đã gán nhãn tương ứng.
     - **Validation/Test**: Ảnh chưa gán nhãn (chưa biết loài).
2. **COCO 2017 Dataset**
   - Tập ảnh đa dạng (con người, đồ vật, thú, ...), được sử dụng để tạo nhãn "other" (các ảnh không thuộc 4 loài chính).
   - Mục đích: giúp mô hình học tốt khả năng phân biệt ảnh chứa động vật mục tiêu với ảnh "không liên quan".
   - Nguồn: Kaggle.
3. **Thống kê sơ bộ**
   - Số lượng ảnh trong mỗi lớp (`cat`, `deer`, `dog`, `horse`) gần cân bằng.
   - Số lượng ảnh "other" được lấy ngẫu nhiên từ COCO 2017, đảm bảo không làm mất cân bằng dữ liệu.
   - Kích thước ảnh gốc đa dạng, được đưa về chuẩn 224×224×3 trước khi đưa vào mô hình.

4 Phương pháp giải quyết

4.1 Tiền xử lý dữ liệu (Data Preprocessing)

1. **Resize ảnh:** Tất cả ảnh được thay đổi kích thước về 224×224×3 để tương thích với kiến trúc VGG16 và ResNet50 vốn được huấn luyện trên ImageNet.
2. **Chuẩn hóa số kênh:**
   - Nếu ảnh đầu vào là ảnh xám (grayscale), nhân kênh thành 3 kênh (RGB).
   - Nếu ảnh có 4 kênh (RGBA), loại bỏ kênh alpha, giữ lại kênh RGB.
3. **Ép kiểu dữ liệu:** Chuyển đổi kiểu pixel thành `float32` và chuẩn hóa giá trị pixel về khoảng [0, 1].
4. **One–hot encoding nhãn:** Nhãn dạng chuỗi (ví dụ "cat") được chuyển thành vector one–hot có 5 chiều (`cat`, `deer`, `dog`, `horse`, `other`).

4.2 Tăng cường dữ liệu (Data Augmentation)

- **Mục đích:** Làm phong phú thêm dữ liệu huấn luyện, giúp mô hình học khả năng khái quát tốt hơn, giảm thiểu overfitting khi số ảnh động vật còn hạn chế.
- **Các phép augmentation áp dụng:**
  1. **Random Crop:** Cắt ngẫu nhiên vùng ảnh con có kích thước 224×224 từ ảnh gốc.

2. **Random Horizontal Flip:** Với xác suất nhất định, ảnh được lật ngang để mô hình học được tính bất đối xứng của các loài.
3. **Random Rotation:** Xoay ảnh trong khoảng $\pm 15°$.
4. **Random Color Jitter:** Điều chỉnh độ sáng, tương phản, hue một cách ngẫu nhiên nhằm làm phong phú về mặt màu sắc.
5. **Gaussian Noise:** Thêm nhiễu để tăng tính robust cho mô hình khi gặp ảnh bị nhiễu.

4.3 Kiến trúc mô hình

4.3.1 VGG16

- **Giới thiệu:** VGG16 là kiến trúc CNN gồm 16 lớp có thể học được đặc trưng sâu nhờ sử dụng liên tiếp các convolutional layers với kernel 3×3 và các lớp max–pooling 2×2 giữa các block. Được giới thiệu trong bài báo "Very Deep Convolutional Networks for Large–Scale Image Recognition" (2014).[2]
- **Cấu trúc cụ thể:**
  - 13 lớp Conv (chia thành 5 block):
    1. **Block 1:** 2 × (Conv 3×3, 64 filters) → MaxPool 2×2
    2. **Block 2:** 2 × (Conv 3×3, 128 filters) → MaxPool 2×2
    3. **Block 3:** 3 × (Conv 3×3, 256 filters) → MaxPool 2×2
    4. **Block 4:** 3 × (Conv 3×3, 512 filters) → MaxPool 2×2
    5. **Block 5:** 3 × (Conv 3×3, 512 filters) → MaxPool 2×2
  - 3 lớp Fully Connected (FC):
    1. FC1: 4096 neuron + ReLU
    2. FC2: 4096 neuron + ReLU
    3. FC3: 1000 neuron (đầu ra ImageNet) nhưng được thay bằng FC 5 neuron (tương ứng 5 lớp) khi fine–tuning.
- **Ưu điểm & hạn chế:**
  - **Ưu điểm:**
    * Cấu trúc đơn giản, dễ triển khai.
    * Sau khi pre–train trên ImageNet, có thể tái sử dụng các bộ lọc (filter) để học nhanh hơn trên tập động vật.
  - **Hạn chế:**
    * Số lượng tham số rất lớn ($\sim$ 138 triệu), dẫn đến tốn bộ nhớ GPU VRAM và tốc độ huấn luyện/inference chậm.
    * Không có cơ chế skip connection, dễ gặp vấn đề vanishing gradient khi mạng còn sâu hơn.

4.3.2 ResNet50

- **Giới thiệu:** ResNet50 là mô hình CNN sâu 50 lớp, nổi bật với khái niệm "residual block" (khối tắt skip connection) để khắc phục vấn đề vanishing gradient. Được giới thiệu trong bài báo "Deep Residual Learning for Image Recognition" (2015).[3]

[2] Karen Simonyan, Andrew Zisserman, "Very Deep Convolutional Networks for Large–Scale Image Recognition" (2014).
[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, "Deep Residual Learning for Image Recognition" (2015).

- **Cấu trúc cơ bản:**
  - Bao gồm một lớp Conv ban đầu, theo sau là 16 residual block, mỗi block chứa 3 lớp conv (1×1, 3×3, 1×1) và một skip connection.
  - Cuối cùng là Global Average Pooling và layer FC (được điều chỉnh thành 5 neuron phù hợp bài toán).
- **Ưu điểm & hạn chế:**
  - **Ưu điểm:**
    - Skip connection giúp khắc phục gradient vanishing, cho phép huấn luyện mạng rất sâu.
    - Số tham số (∼ 25 triệu) thấp hơn VGG16, giúp tiết kiệm bộ nhớ và tăng tốc độ inference.
    - Thường đạt độ chính xác cao hơn so với nhiều kiến trúc cũ.
  - **Hạn chế:**
    - Cấu trúc phức tạp hơn, khó hiểu cho người mới.
    - Vẫn đòi hỏi tài nguyên tính toán (dù ít hơn VGG16) và cần tập dữ liệu đủ đa dạng để phát huy hết tiềm năng.

5 Chi tiết huấn luyện chung cho cả hai mô hình

1. **Initialization:** Sử dụng hai mô hình đã được pre–train trên ImageNet: VGG16 và ResNet50 (gọi từ `tensorflow.keras.applications`). Loại bỏ lớp FC cuối (đầu ra 1000 lớp) và thay bằng các tầng tùy chỉnh.
2. **Kiến trúc bổ sung phía trên:**
   (a) `GlobalAveragePooling2D()`
   (b) `Dense(4096, activation='relu')`, kèm `BatchNormalization()`, `Dropout(rate=0.2)`
   (c) `Dense(4096, activation='relu')`, kèm `BatchNormalization()`, `Dropout(rate=0.2)`
   (d) `Dense(5, activation='softmax')` (đầu ra 5 lớp: `cat`, `deer`, `dog`, `horse`, `other`)
3. **Hàm mất mát (Loss Function):** Sử dụng `Categorical Crossentropy`.
4. **Optimizer:** Sử dụng `Adam` với learning rate (lr) = 1e-4.
5. **Thông số huấn luyện:**
   - Số epochs: 100 (có thể dừng sớm nếu validation loss không cải thiện sau 16 epochs liền – callback `EarlyStopping`).
   - Batch size: 64.
   - `EarlyStopping(monitor='val_loss', patience=16, restore_best_weights=True)`
   - `ModelCheckpoint` để lưu mô hình tốt nhất (theo `val_accuracy`).
6. **Quy trình huấn luyện:**
   - Chia tập dữ liệu thành hai phần:
     - **Train** (80% dữ liệu gán nhãn), áp dụng augmentation và huấn luyện.
     - **Validation** (20% còn lại), chỉ thực hiện normalize/rescale, không augmentation.
   - Lặp qua epochs, theo dõi giá trị `train_loss`, `train_accuracy`, `val_loss`, `val_accuracy`.

6 Kết quả thực nghiệm

Sau khi đã train trên 2 tập dataset

6.1 Trên tập huấn luyện (Training set)

- **VGG16:**
  - Sau 100 epochs, mô hình đạt độ chính xác (accuracy) ≈ 98% trên tập train.
  - Đường cong loss giảm ổn định, hội tụ trước epoch 80.
- **ResNet50:**
  - Sau 100 epochs, đạt ≈ 99% accuracy trên tập train.
  - Hội tụ nhanh hơn VGG16 (thường đạt >95% accuracy trước epoch 60).

6.2 Trên tập validation

- **VGG16:**
  - Độ chính xác tối đa (best `val_accuracy`) ≈ 90% ở khoảng epoch 70.
  - Kết thúc training, `val_accuracy` ≈ 89%.
  - Nhạy cảm với overfitting khi số epoch > 70 (`val_loss` bắt đầu tăng mặc dù `train_loss` giảm).
- **ResNet50:**
  - Best `val_accuracy` ≈ 93% (đạt sớm, khoảng epoch 50–60).
  - `val_accuracy` dao động ở mức 90–93% trong giai đoạn trung bình.
  - Ít gặp overfitting hơn so với VGG16 do residual block hỗ trợ học tốt ở các tầng sâu.

6.3 Thống kê trên nhiều độ đo

- **Độ chính xác (Accuracy):** ResNet50 > VGG16 (trên cả train và validation).
- **Precision, Recall, F1-Score (theo từng lớp):**
  - Các lớp dễ nhầm lẫn: `deer` và `horse`.
  - ResNet50 cho precision và recall cao hơn ở các lớp này so với VGG16 (tăng khoảng 2–3%).
- **Ma trận nhầm lẫn (Confusion Matrix):**
  - VGG16: Nhiều ảnh `deer` bị dự đoán nhầm thành `horse` (5%).
  - ResNet50: Giảm tỉ lệ nhầm đến 2–3%.

7 Đánh giá và nhận xét

7.1 Ưu điểm chung

- Sử dụng transfer learning giúp rút ngắn thời gian huấn luyện và giảm yêu cầu dữ liệu gán nhãn.
- Kết hợp augmentation làm mô hình khái quát tốt hơn, ít nhạy cảm với thay đổi về góc độ hoặc ánh sáng.
- Sử dụng dataset COCO để tạo lớp "other" giúp tăng tính thực tiễn khi gặp ảnh không thuộc nhóm 4 loài.

7.2 Nhược điểm và hạn chế

- **VGG16:**
  - Quá nhiều tham số ($\sim$ 138 triệu) gây tốn bộ nhớ và thời gian huấn luyện/inference lâu, khó triển khai trên môi trường hạn chế tài nguyên.
  - Khả năng overfitting cao khi số epoch tăng quá lớn.
- **ResNet50:**
  - Cấu trúc phức tạp hơn, khó hiểu, khó tinh chỉnh so với VGG16.
  - Mặc dù có ít tham số hơn VGG16, residual block vẫn yêu cầu tài nguyên GPU tương đối cao.
- **Dữ liệu:**
  - Chưa đa dạng về bối cảnh thực tế (chỉ có ảnh cá nhân từng con trong môi trường tương đối "sạch"); khi áp dụng với ảnh thực địa (rừng, môi trường hỗn hợp), độ chính xác có thể giảm mạnh.
  - Nhãn "other" được lấy ngẫu nhiên từ COCO, có thể chưa bao phủ hết các trường hợp "không phải động vật" thực tế.

7.3 So sánh VGG16 và ResNet50

- **Tốc độ hội tụ:** ResNet50 hội tụ nhanh hơn VGG16 do skip connection giảm thiểu vấn đề vanishing gradient.
- **Độ chính xác:** ResNet50 nhỉnh hơn VGG16 khoảng 2–4% trên tập validation, đặc biệt ở các lớp dễ nhầm lẫn (`deer` vs `horse`).
- **Yêu cầu tài nguyên:** VGG16 có số lượng tham số gấp khoảng 5 lần ResNet50, do đó tốn bộ nhớ và thời gian huấn luyện/inference hơn.
- **Khả năng mở rộng:** ResNet50 linh hoạt khi thêm tầng sâu hơn nhờ residual block; VGG16 muốn thêm tầng phải cân nhắc kỹ lưỡng vấn đề gradient.

8 Kết luận và hướng nghiên cứu

8.1 Kết luận

- Đồ án đã minh chứng hiệu quả của transfer learning khi áp dụng VGG16 và ResNet50 cho bài toán phân loại 4 loài động vật kết hợp lớp "other".
- ResNet50 đạt chất lượng tốt hơn về độ chính xác và khả năng khái quát so với VGG16, đồng thời ít gặp overfitting hơn.
- Hệ thống hiện tại phù hợp với bộ dữ liệu có chất lượng ảnh khá, tuy nhiên khi triển khai với ảnh thực địa cần bổ sung thêm các phép augmentation đặc trưng (thay đổi độ sáng, mờ nhòe, che khuất một phần, ...).

8.2 Hướng mở rộng và nghiên cứu tiếp theo

1. **Thử nghiệm các kiến trúc nhẹ hơn:** MobileNetV2, EfficientNet để giảm tải tài nguyên, tăng tốc inference, vẫn giữ độ chính xác cao.
2. **Kết hợp với Machine Learning truyền thống:** Sử dụng đặc trưng đã trích xuất từ VGG16/ResNet50 kết hợp SVM hoặc Random Forest để cải thiện khả năng phân biệt.
3. **Ensemble Learning:** Kết hợp đầu ra từ nhiều mô hình (VGG16, ResNet50, EfficientNet) để giảm thiểu sai sót cá nhân, nâng cao độ chính xác tổng thể.
4. **Tối ưu siêu tham số (Hyperparameter Tuning):** Dùng Grid Search hoặc Bayesian Optimization để tìm learning rate, batch size, dropout rate tối ưu cho từng mô hình.
5. **Phân tích ma trận nhầm lẫn sâu hơn:** Xem xét các trường hợp nhầm lẫn giữa các lớp (`deer` vs `horse`, `cat` vs `dog`), thu thập thêm dữ liệu hoặc augmentation đặc thù để khắc phục.
6. **Ứng dụng Transfer Learning chéo lĩnh vực:** Thử fine–tune mô hình đã huấn luyện trên các tập dữ liệu y sinh (Medical Imaging) để phát hiện động vật y tế (ví dụ: phát hiện vết thương, bệnh lý bên ngoài động vật).
7. **Triển khai thực tế:** Xây dựng ứng dụng di động hoặc thiết bị giám sát tự động (edge device) gắn camera để phân loại động vật trong rừng, vườn bảo tồn.
8. **Tích hợp với hệ thống giám sát:** Kết hợp với hệ thống IoT, lưu trữ ảnh lên cloud, xử lý real–time, cung cấp cảnh báo khi phát hiện loài quý hiếm hoặc ngoại lai.

9 Đánh giá thành Viên

- Nguyễn Thiên Bảo - 23520127: Hoàn thành 100%
- Phạm Huỳnh Long Vũ -23521815: Hoàn thành 100%

10 Source Code

Source Code tụi em đã up lên github tại: link source code

11 Tài liệu tham khảo

1. 4 Animal Classification Dataset, Kaggle.
2. COCO 2017 Dataset, Kaggle.
3. Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. “Deep Residual Learning for Image Recognition” (2015).
4. Karen Simonyan, Andrew Zisserman. “Very Deep Convolutional Networks for Large–Scale Image Recognition” (2014).
5. TensorFlow Documentation.
6. “Comparison of Models For Classification”, Journal of Informatics and Mechanics.
7. Combining CNN with traditional machine learning