

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

CS231 - THỊ GIÁC MÁY TÍNH



BÁO CÁO ĐỒ ÁN CUỐI KÌ

ĐỀ TÀI: Animals Classification

- Giảng viên hướng dẫn: **TS. Mai Tiến Dũng**
- Sinh viên thực hiện: **23520945 - Nguyễn Văn Minh**
23521405 - Phan Nhật Tân
- Lớp: **CS231.P21.KHTN**

Nội dung chính

1. Lý do chọn đề tài
2. Về đồ án
3. Phương pháp giải quyết
4. Kết quả thực nghiệm
5. Tài liệu tham khảo

1. Lí do chọn đề tài:

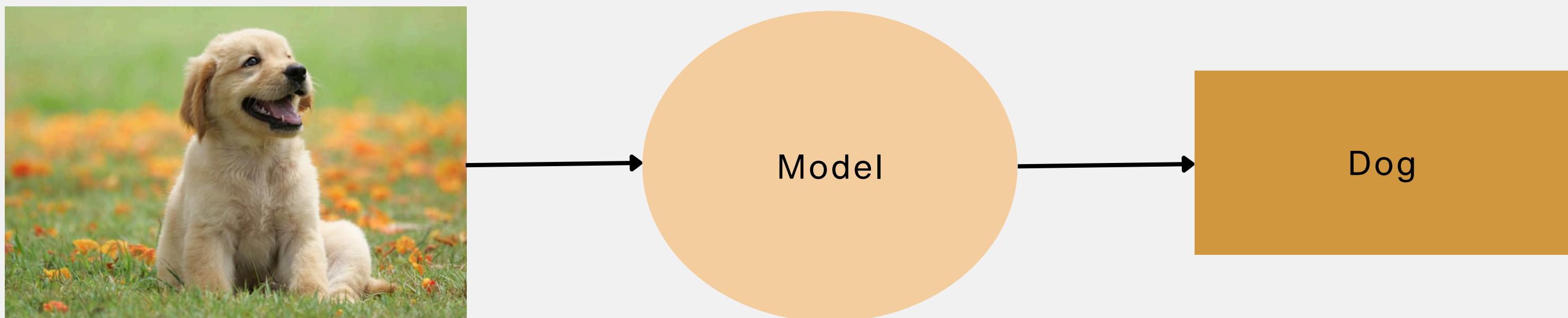
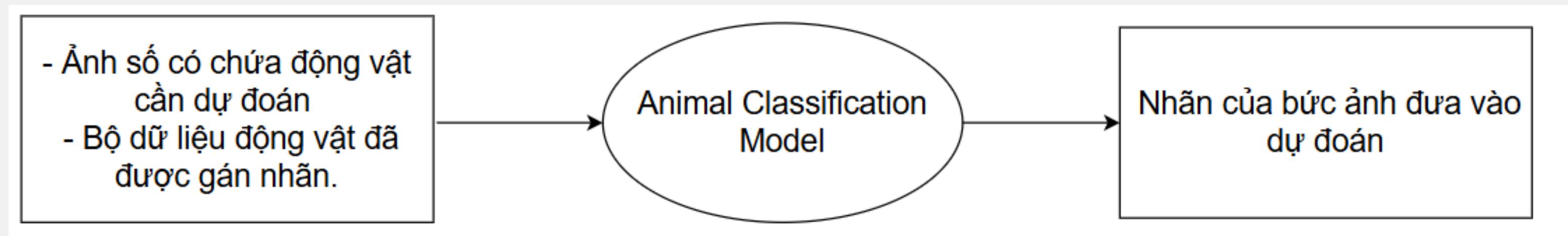
- Hệ thống phân loại động vật có thể được sử dụng trong các ứng dụng thực tế như:
 - Hệ thống hỗ trợ du lịch/sở thú (hiển thị thông tin về loài qua ảnh).
 - Hệ thống giám sát động vật bằng camera (ví dụ: bảo vệ động vật quý hiếm).
- Với bài toán phân loại nhiều loại động vật: có sự tương đồng giữa một số loài → đòi hỏi mô hình tốt để phân biệt.



Dog like cat

2. Về đồ án:

- Tên đồ án: Animals Classification
- Input: Ảnh số có chứa động vật cần dự đoán và bộ dữ liệu động vật đã được gán nhãn.
- Output: Nhãn của bức ảnh đưa vào dự đoán.



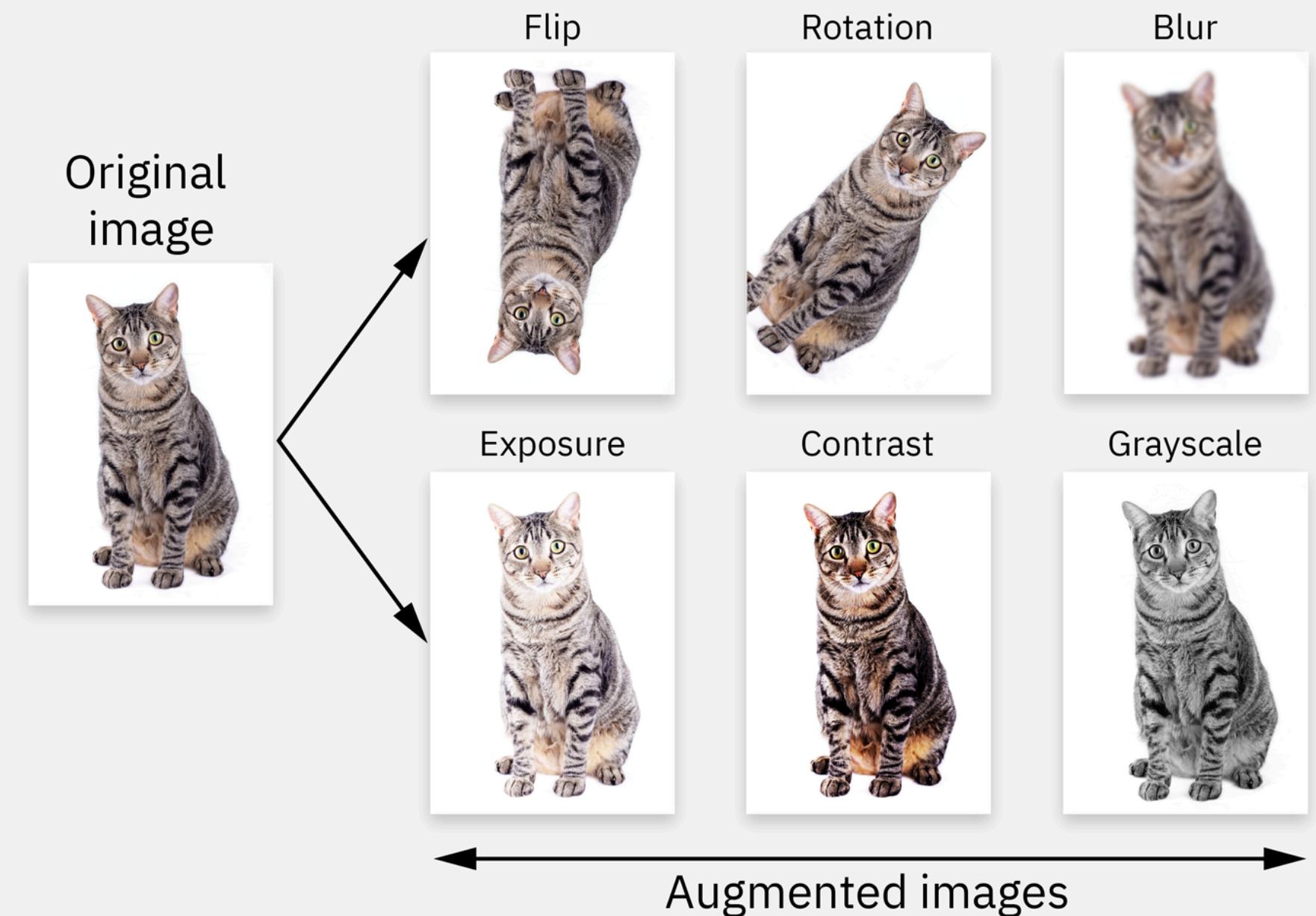
Phương pháp giải quyết



3.1 Data Augmentation:

- **Mục đích:** tăng cường bộ dữ liệu, giúp cải thiện khả năng tổng quát của mô hình và giảm thiểu hiện tượng overfitting.

```
datagen = ImageDataGenerator(  
    rotation_range=20,  
    width_shift_range=0.2,  
    height_shift_range=0.2,  
    shear_range=0.2,  
    zoom_range=0.2,  
    horizontal_flip=True,  
    fill_mode='nearest'  
)
```

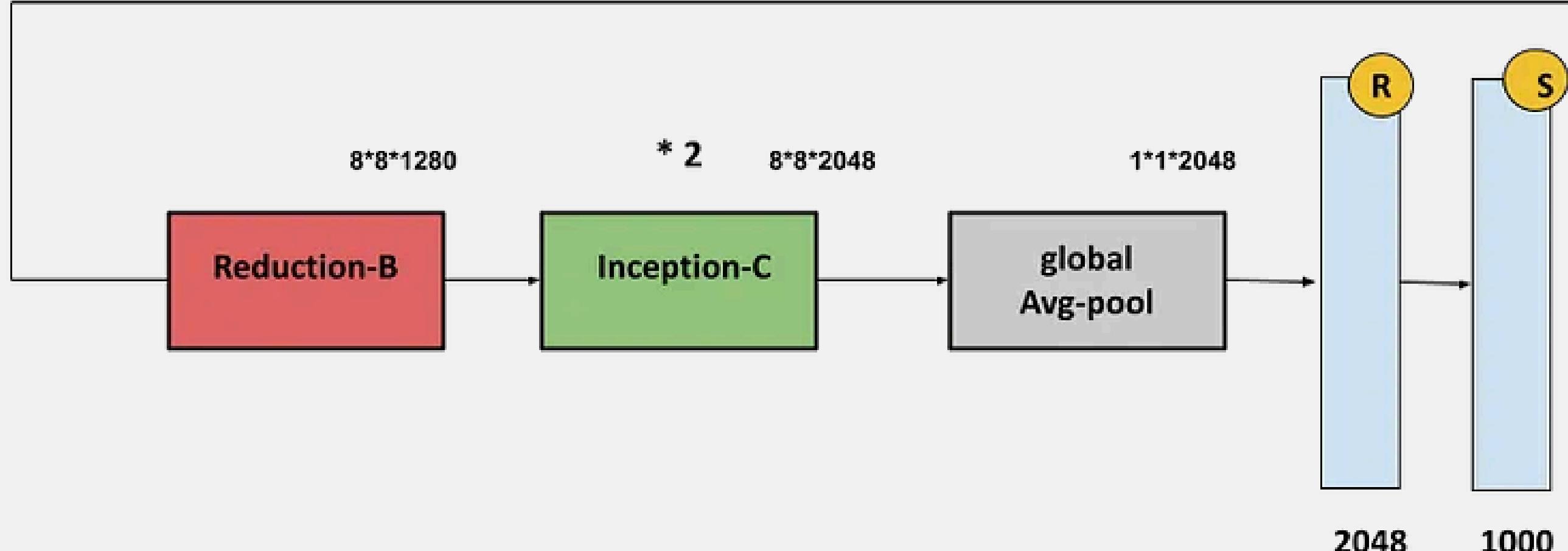
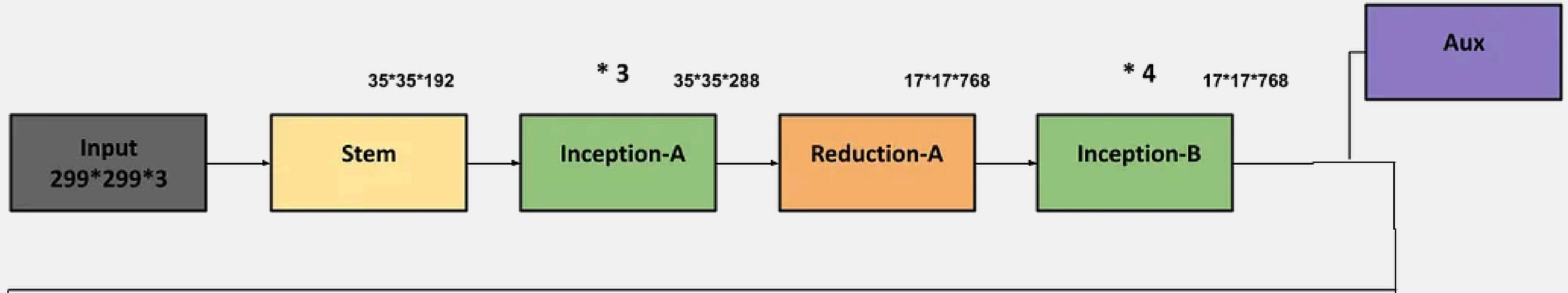


3.2 Xây dựng mô hình

3.21 Mô hình InceptionV3

3.22 Mô hình EfficientNetB3

InceptionV3:



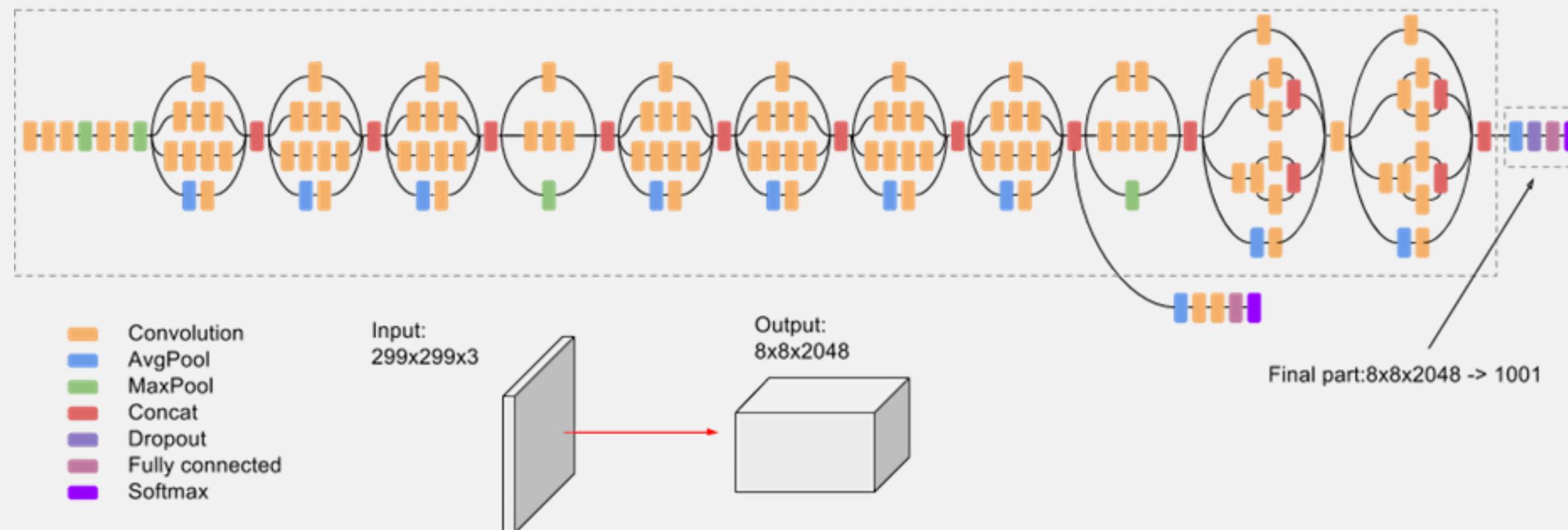
InceptionV3:

- **Sử dụng mô hình InceptionV3 pretrained:**

- Đã được huấn luyện trên **ImageNet**
- Đầu vào ảnh: $299 \times 299 \times 3$

- **Thêm các tầng tùy chỉnh:**

- **GlobalAveragePooling2D**: giảm số lượng tham số và tránh overfitting.
- **Dense 1000 units** với **selu**: hỗ trợ học sâu và ổn định.
- **Dropout 0.5**: tránh overfitting.
- **Output Dense (softmax)** với số node = số lớp → phân loại đa lớp.



Huấn luyện mô hình 2 giai đoạn:

Giai đoạn 1:

- Đóng băng toàn bộ trọng số của mô hình pretrained
- Optimizer:** Adam, learning rate = **1e-3**
- Số epoch:** 10

```
base_model = InceptionV3(weights='imagenet', include_top=False, input_shape=(180, 180, 3))
base_model.trainable = False # Giai đoạn 1: Freeze toàn bộ

model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=1e-3),
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

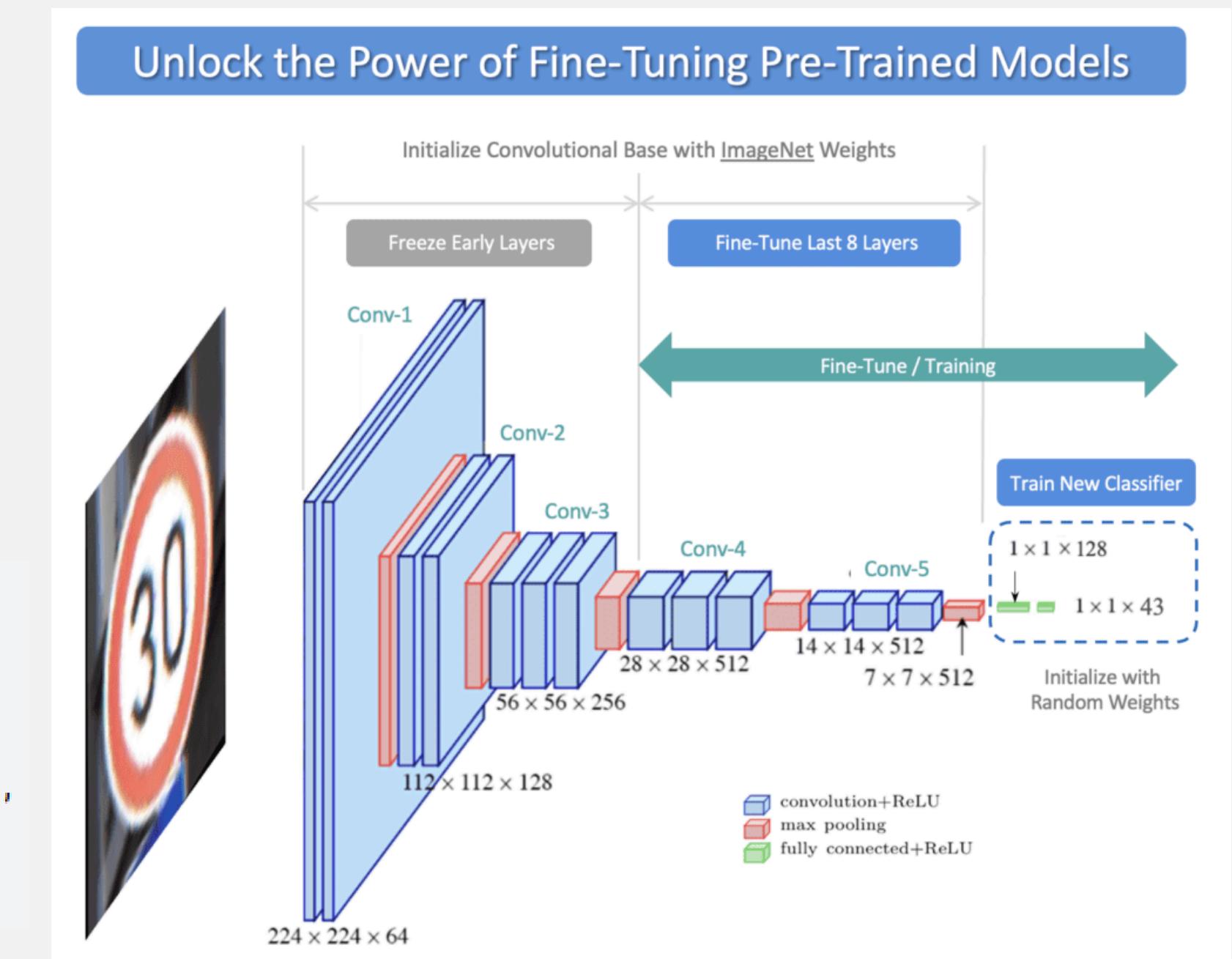
history1 = model.fit(datagen.flow(X_train, y_train, batch_size=32),
                     epochs=10,
                     validation_data=(X_test, y_test))
```

Huấn luyện mô hình 2 giai đoạn:

Giai đoạn 2:

- Bỏ đóng băng 50 lớp cuối cùng của mô hình InceptionV3
- Tiếp tục huấn luyện toàn bộ mô hình (bao gồm phần trước + backbone).

```
base_model.trainable = True  
for layer in base_model.layers[:-50]:  
    layer.trainable = False  
  
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=1e-4),  
              loss='sparse_categorical_crossentropy',  
              metrics=['accuracy'])
```



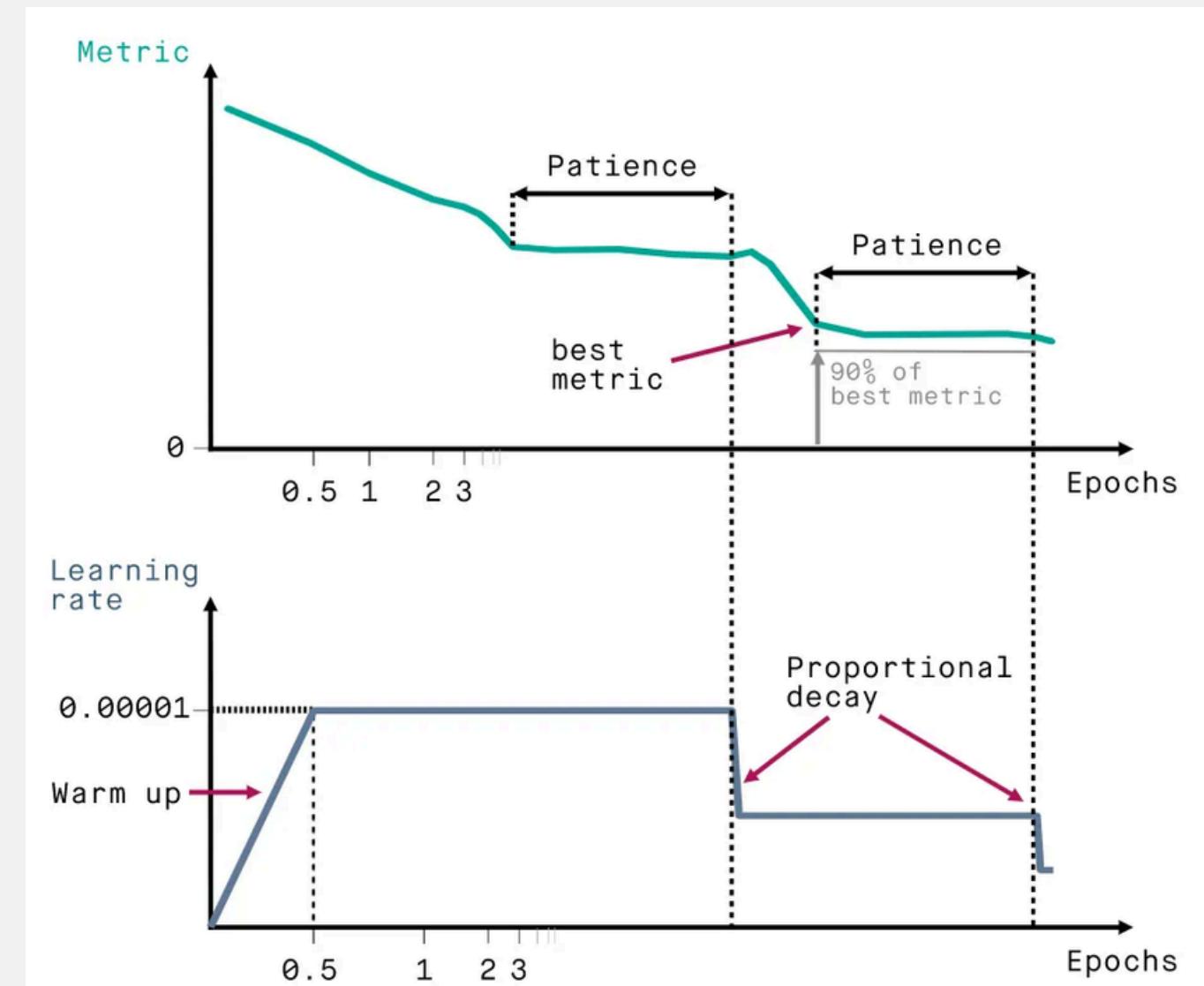
Huấn luyện mô hình 2 giai đoạn:

Giai đoạn 2:

```
lr_scheduler = ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=5, min_lr=1e-6)
```

- Theo dõi **val_loss**. Nếu không giảm sau 5 epochs, giảm learning rate xuống 0.2 lần.
- Sử dụng Adam optimizer với learning rate nhỏ để tránh làm “hỏng” các trọng số của mô hình pretrained.
- Train trên 40 epochs, batch_size = 32

```
history2 = model.fit(datagen.flow(X_train, y_train, batch_size=32),
                      epochs=40,
                      validation_data=(X_test, y_test),
                      callbacks=[lr_scheduler])
```

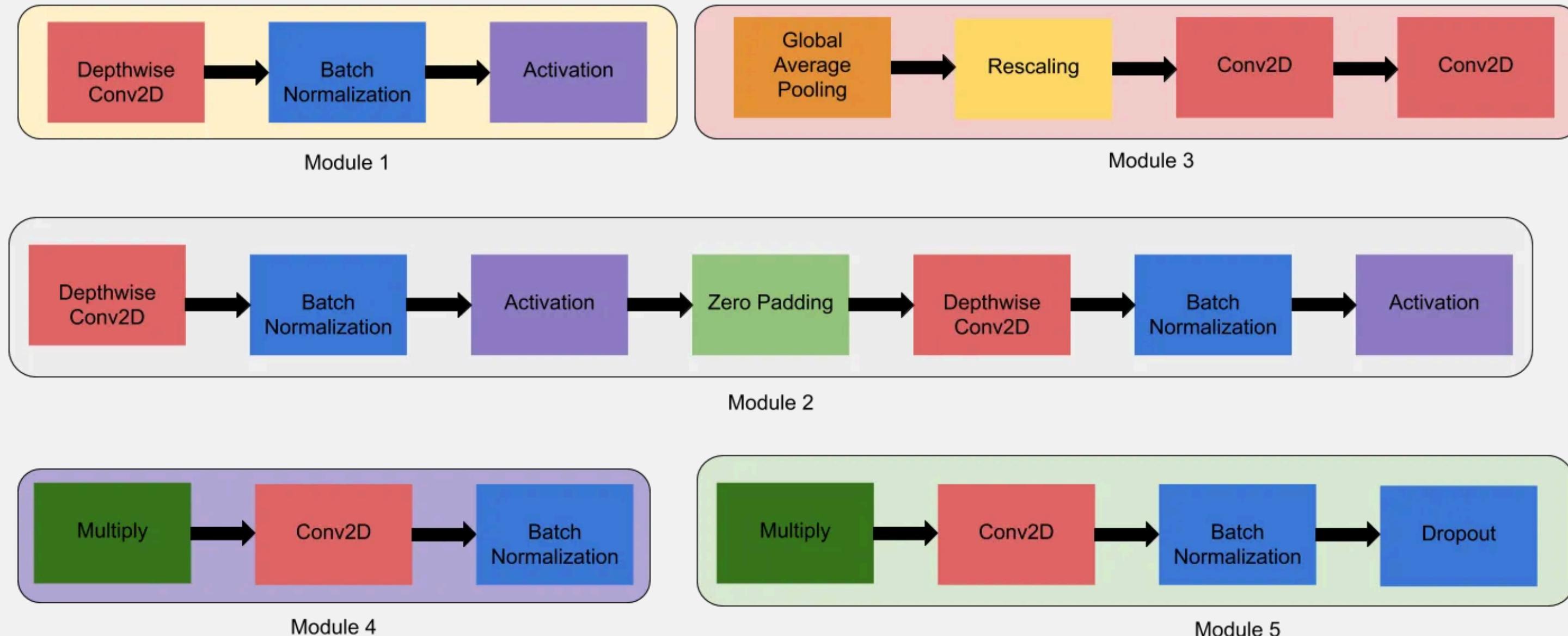


3.2 Xây dựng mô hình

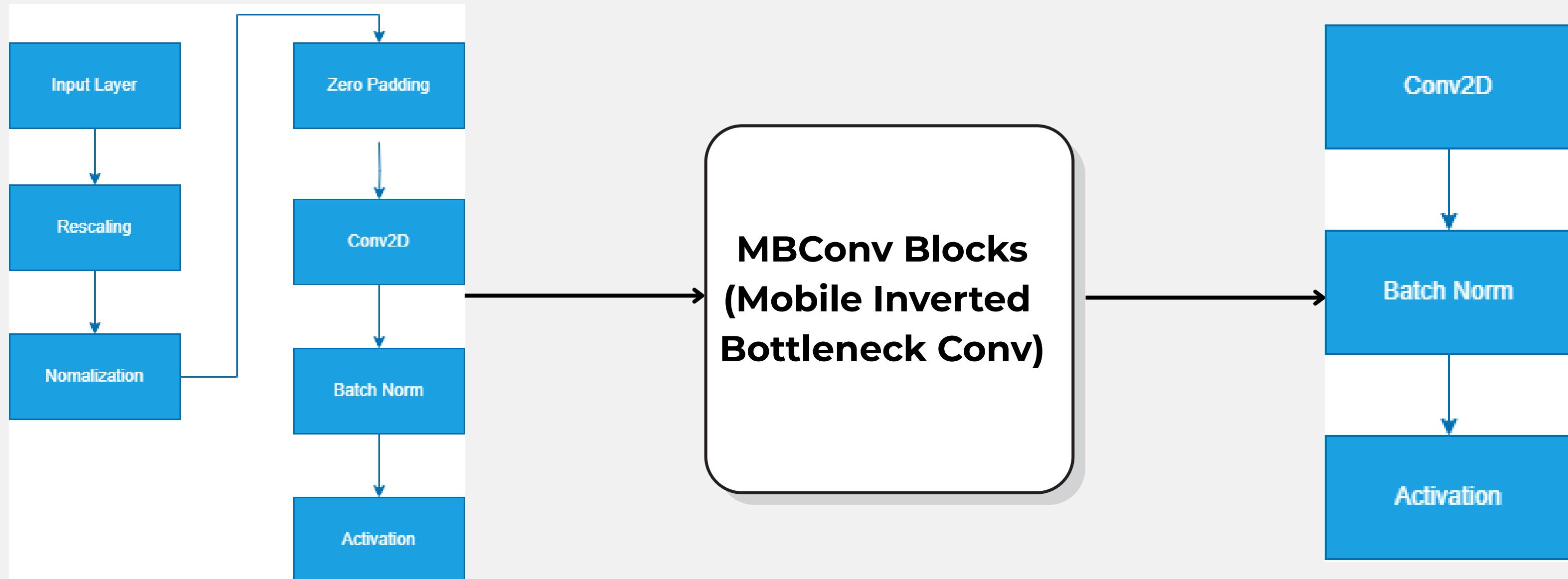
3.21 Mô hình InceptionV3

3.22 Mô hình EfficientNetB3

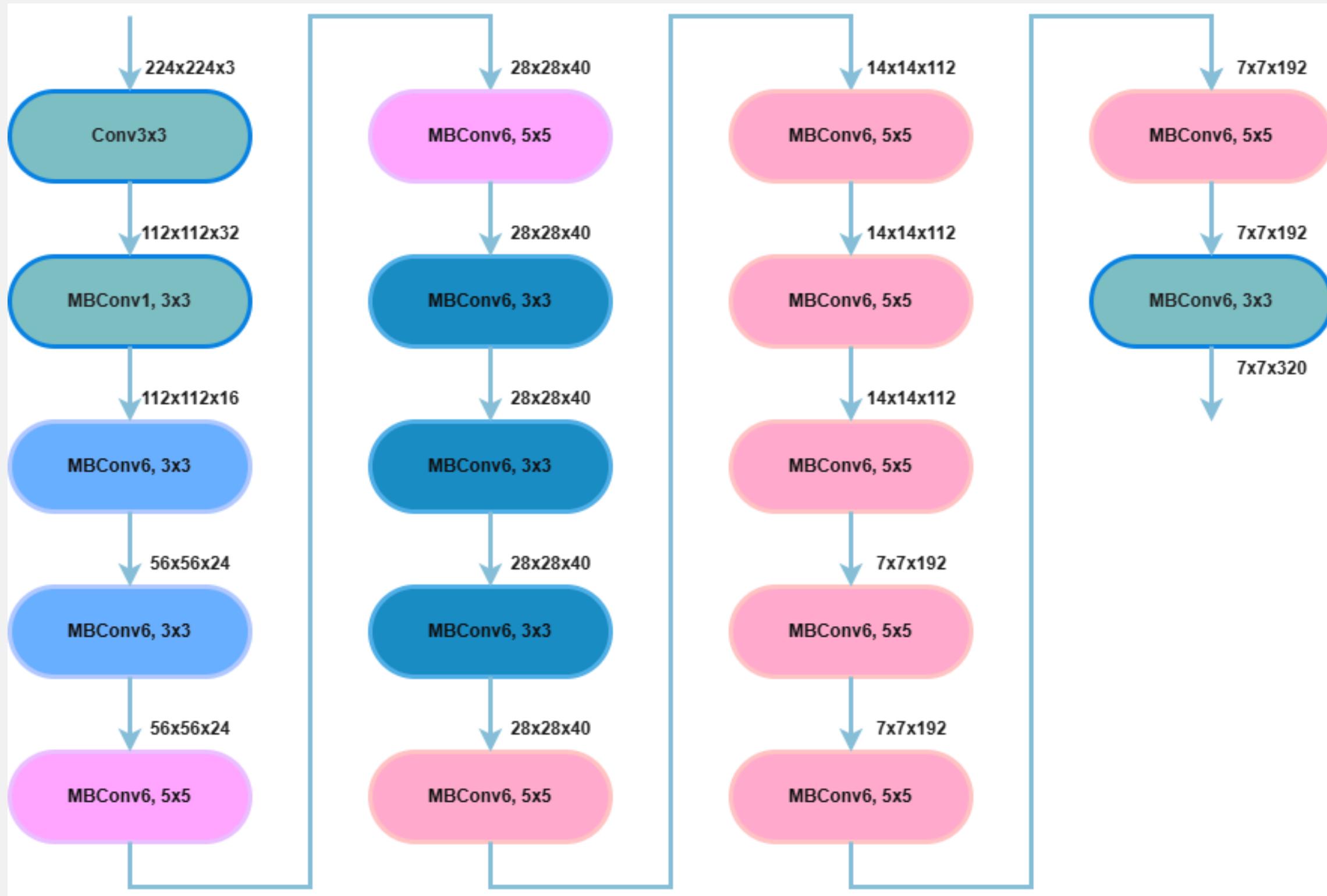
EfficientNetB3:



EfficientNetB3:



EfficientNetB3:



EfficientNetB3:

- **Tiền xử lý ảnh:**
 - Resize ảnh về kích thước 224×224 . (vì input yêu cầu 224×224)
 - Chuyển ảnh sang RGB, chuẩn hóa về $[0, 1]$.
- **Kiến trúc mô hình sử dụng:**
 - Sử dụng EfficientNetB3 từ keras.applications
 - Unfreeze 50 lớp cuối cùng để fine-tune.
 - Thêm các tầng:
 - **GlobalAveragePooling2D()**: trích xuất đặc trưng toàn cục.
 - **Dense(512, activation='relu')**: học các biểu diễn phân loại.
 - **Dropout(0.5)**: chống overfitting.
 - **Dense(num_classes, activation='softmax')**: lớp phân loại cuối cùng.

```
# Load the EfficientNetB3 model with pre-trained weights
base_model = EfficientNetB3(weights='imagenet', include_top=False, input_shape=(224, 224, 3))

# Unfreeze some of the top layers of EfficientNetB3
for layer in base_model.layers[-50:]: # Fine-tune top layers
    layer.trainable = True
```

Huấn luyện mô hình:

- **Loss:** sparse_categorical_crossentropy (vì label là số nguyên).
- **Optimizer:** Adam với learning_rate=1e-4.
- **Scheduler:** ReduceLROnPlateau theo dõi val_loss, giảm LR nếu không cải thiện sau 5 epoch.

```
# Learning rate scheduler
lr_scheduler = ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=5, min_lr=1e-6)

# Compile the model with a lower learning rate
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=1e-4),
               loss='sparse_categorical_crossentropy', metrics=['accuracy'])
```

Huấn luyện mô hình:

- Số huấn luyện:
- **Epoch:** 50.
- **Batch size:** 8 (do EfficientNetB3 khá nặng, batch nhỏ giúp tránh tràn RAM GPU).

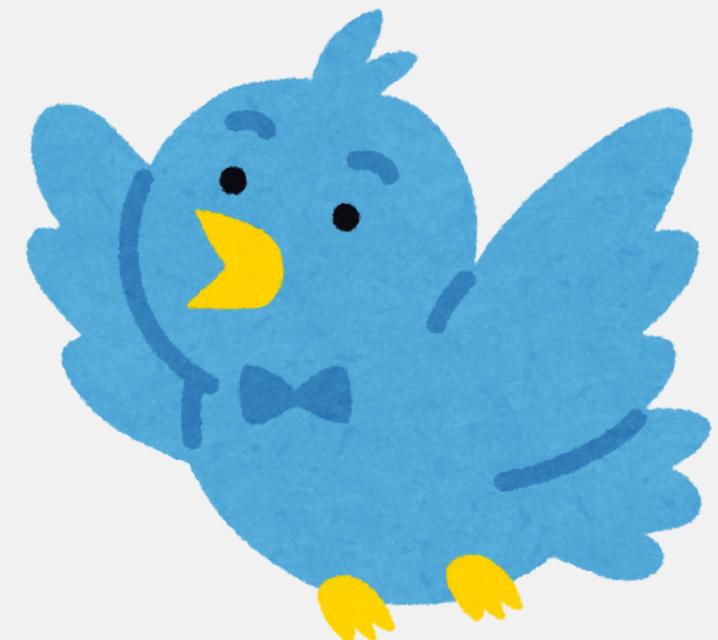
```
history = model.fit(datagen.flow(X_train, y_train, batch_size=8),
                     epochs=50,
                     validation_data=(X_test, y_test),
                     callbacks=[lr_scheduler])
```

Huấn luyện mô hình:

- Model: "sequential"

Layer (type)	Output Shape	Param #
efficientnetb3 (Functional)	(None, 7, 7, 1536)	10783535
global_average_pooling2d (G	(None, 1536)	0
lobalAveragePooling2D)		
dense (Dense)	(None, 512)	786944
dropout (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 90)	46170
<hr/>		
Total params: 11,616,649		
Trainable params: 11,529,346		
Non-trainable params: 87,303		
<hr/>		

Kết quả thực nghiệm

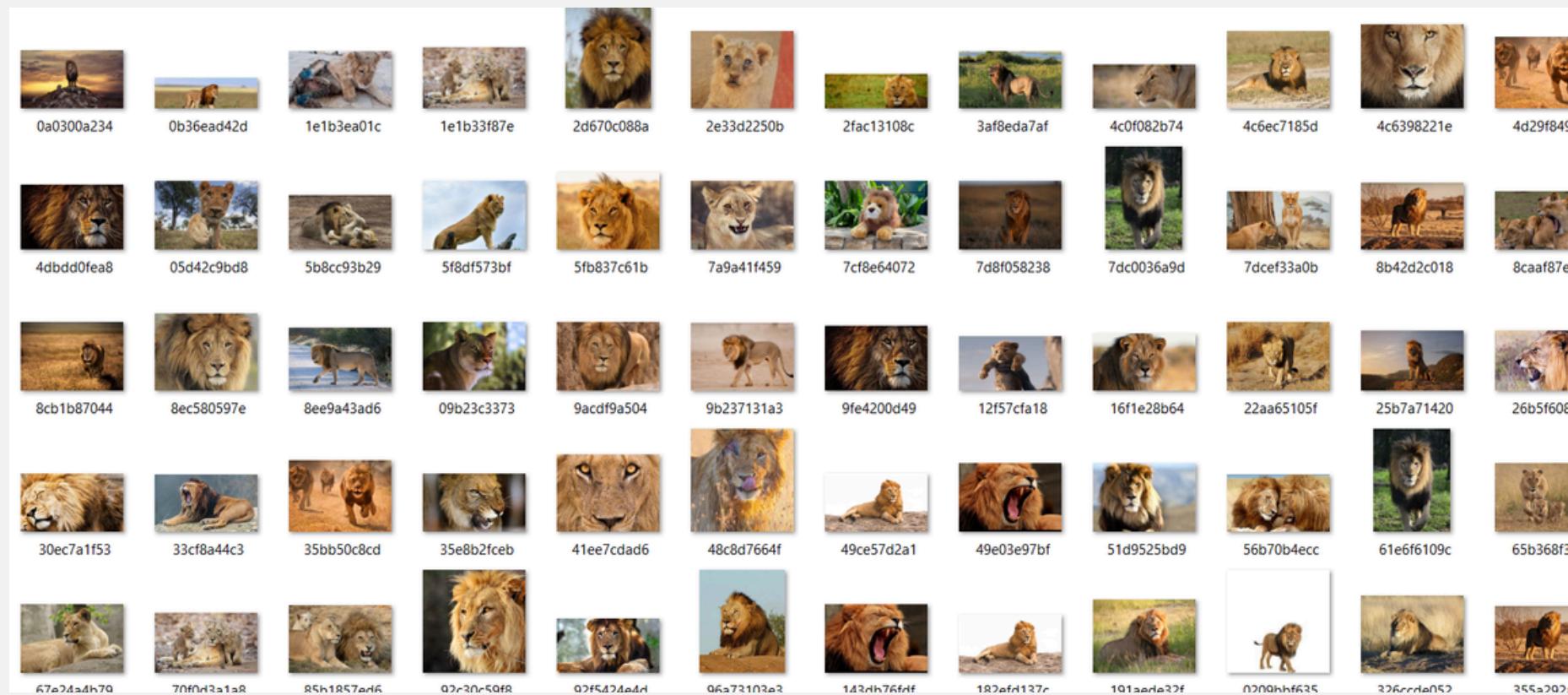


= ?



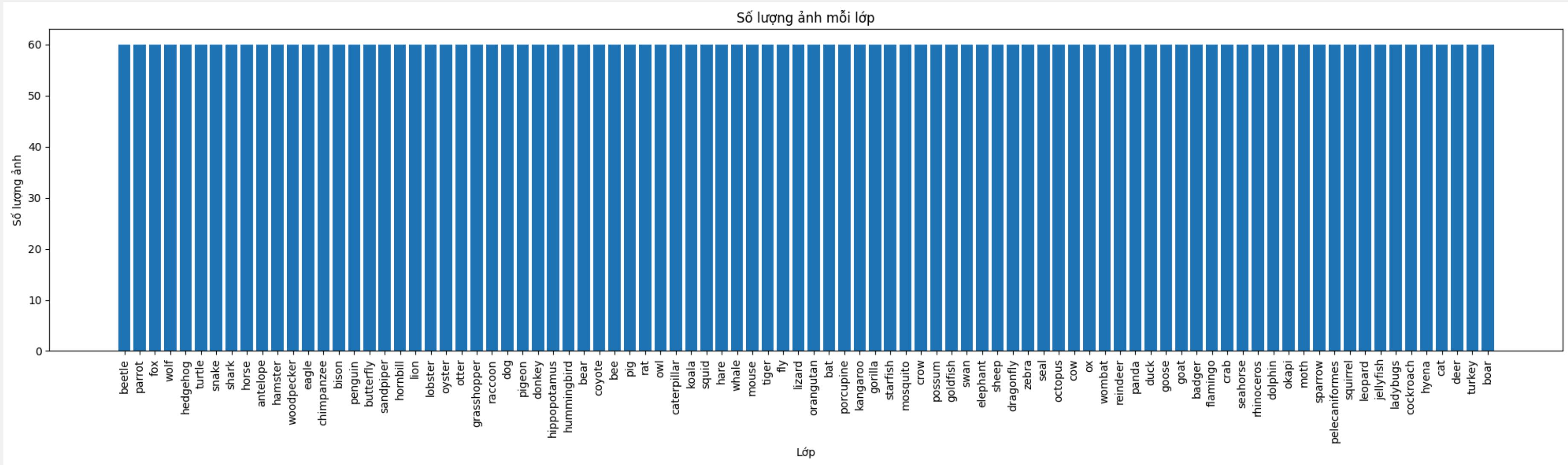
Dataset:

- **Dataset: Animal Image Dataset**, dữ liệu được thu thập từ **Google Images**
- **Vì sao chọn Dataset này?**
 - Có 90 loài động vật khác nhau, có các loài có hình dạng giống nhau như **báo-mèo-hổ** -> thử thách cho các ứng dụng thực tế



The screenshot shows a dataset page titled "Animal Image Dataset (90 Different Animals)". It includes a header with a "Coming in on Wildlife: 5400 Animal Images Across 90 Diverse Classes" message and a row of small animal icons. Below the header are tabs for "Data Card", "Code (95)", "Discussion (1)", and "Suggestions (0)". A large section titled "About Dataset" contains a "Text" field with detailed information about animals, their biology, and evolution. To the right, there are sections for "Usability" (10.00), "License" (Other), "Expected update frequency" (Never), and "Tags" (Biology, Image, Classification, Computer Vision, Animal). The bottom of the page features a large black bar with three white rectangular input fields.

Thống kê dataset



Sự cân bằng trong dataset, không có bias về 1 class nào

Kết quả thực nghiệm

- Độ đo: Accuracy

$$\text{Accuracy} = \frac{\text{Correct predictions}}{\text{All predictions}}$$

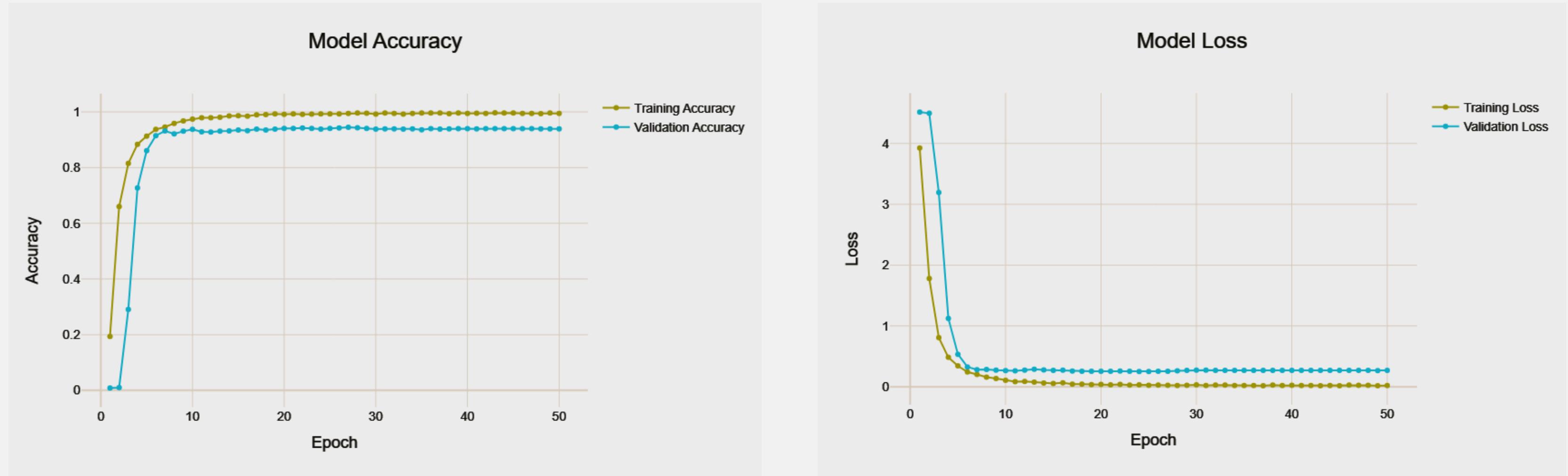
- **Accuracy** trên các phương pháp được đề xuất:

InceptionV3 with 1 phase training	InceptionV3 with 2 phase training	EfficientNetB3
0.83	0.89	0.95

- **Accuracy** trên phương pháp HOG + Machine learning model

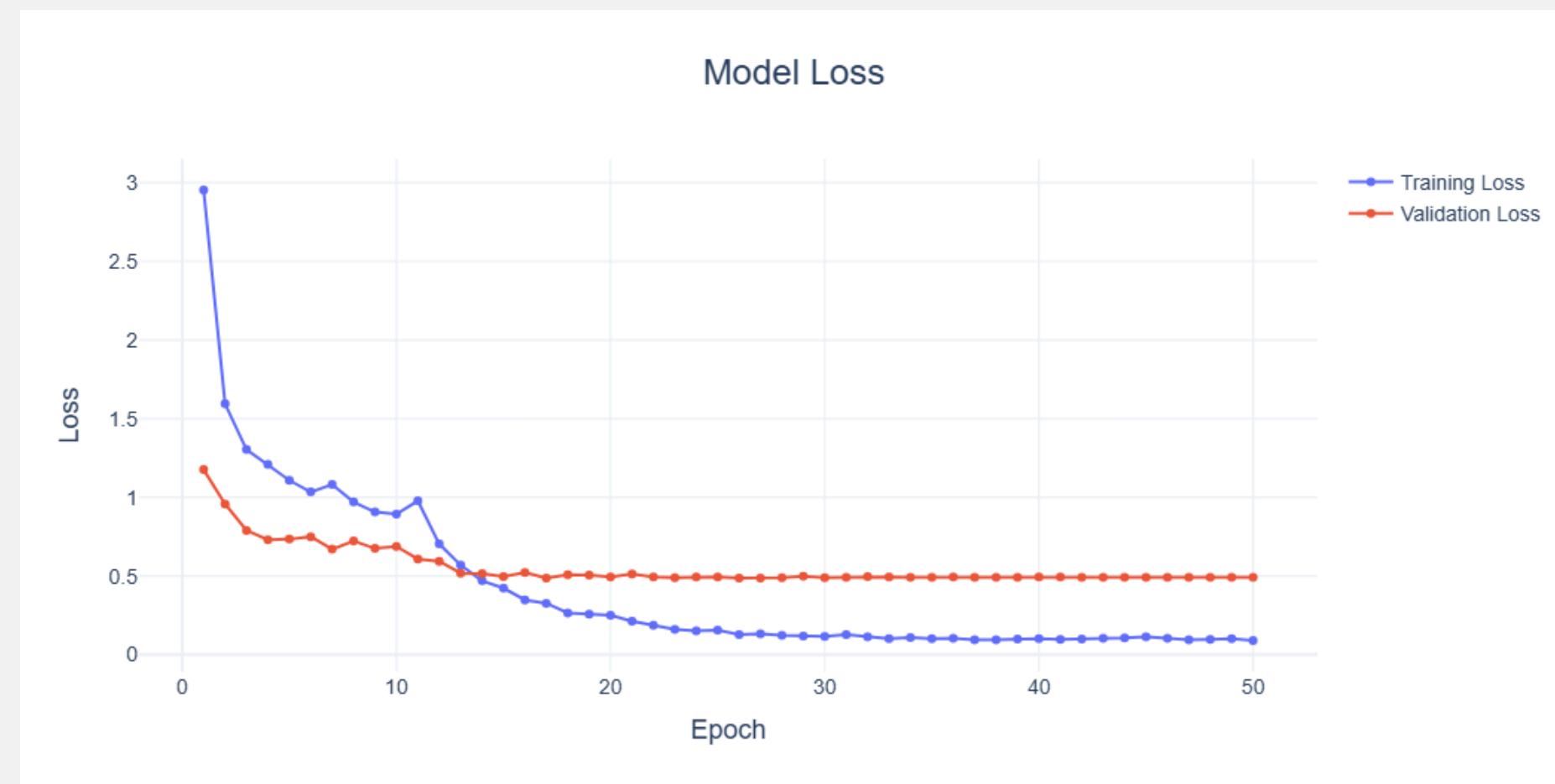
HOG + Linear SVM	HOG + Random Forest
0.27	0.28

Kết quả thực nghiệm



Accuracy và Loss của EfficientNetB3 qua từng epoch

Kết quả thực nghiệm



Accuracy và Loss của InceptionV3 qua từng epoch

Demo

Actual: owl
Predicted: owl



Actual: zebra
Predicted: zebra



Actual: sandpiper
Predicted: sandpiper



Actual: deer
Predicted: antelope



Actual: hyena
Predicted: hyena



Actual: bee
Predicted: bee



Actual: lizard
Predicted: lizard



Actual: reindeer
Predicted: reindeer



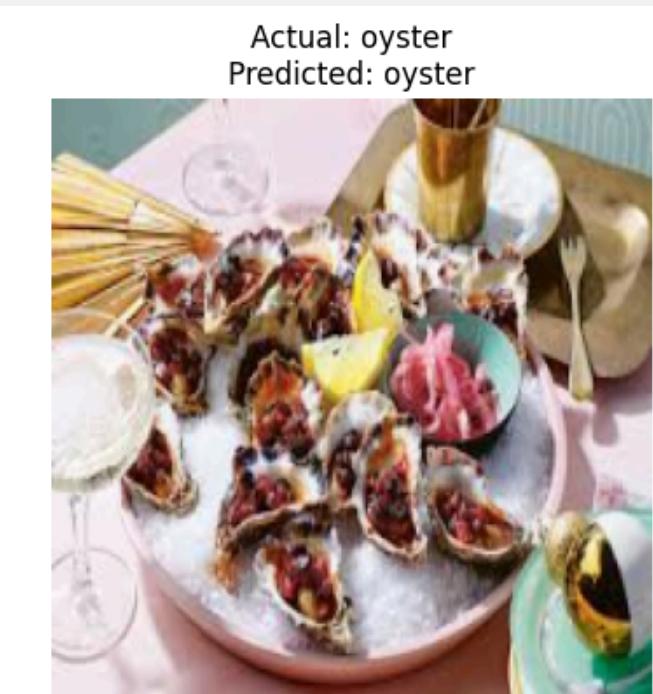
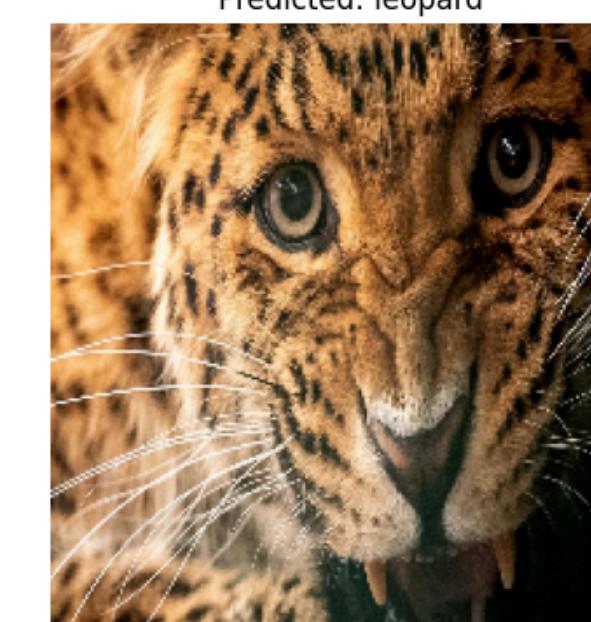
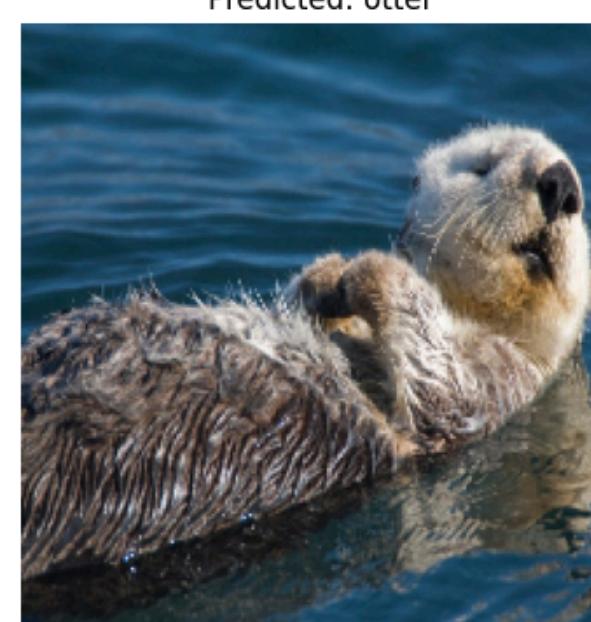
Actual: bear
Predicted: bear



Actual: shark
Predicted: shark



Demo



Nhận xét và mở rộng

Nhận xét:

- Thực nghiệm sử dụng đặc trưng HOG kết hợp với thuật toán máy học cho ra độ chính xác rất thấp. Lí do là vì HOG chỉ trích xuất thông tin gradient và cạnh, không đủ để phân biệt các đặc điểm phức tạp của 90 loài động vật
- Trái ngược với HOG, các mô hình học sâu pretrained cho hiệu suất vượt trội, accuracy lên đến 95% trong tập test.

Nhận xét và mở rộng

Hướng phát triển:

- So sánh đa mô hình:
 - Mở rộng bài toán bằng cách so sánh hiệu năng của nhiều mô hình pretrained khác nhau, như ResNet50, DenseNet121 hoặc MobileNetV2.
- Kết hợp mô hình (ensemble learning):
 - Một hướng tiềm năng là áp dụng ensemble learning - kết hợp đầu ra để cải thiện độ chính xác tổng thể.
- Phân tích lỗi và Confusion Matrix:
 - Từ ma trận nhầm lẫn, xác định những cặp loài thường bị nhầm lẫn với nhau để cải thiện tập dữ liệu (ví dụ: bổ sung ảnh), điều chỉnh loss function hoặc kiến trúc nhận dạng cụ thể hơn cho những lớp khó.

Tài liệu tham khảo

Tài liệu tham khảo:

- **[Animal Image Dataset - 90 Different Animals]**
- **[EfficientNet Paper]**
- **[TensorFlow Documentation]**
- **[OpenCV Documentation]**
- **[Complete Architectural Details of all EfficientNet Models]**
- **[Inception V3 CNN Architecture Explained]**
- **[Inception V2 and V3 - Inception Network Versions]**

**Cảm ơn thầy và
các bạn đã lắng nghe**

CS231 - COMPUTER VISION