TRƯỜNG ĐẠI HỌC HỌC VĂN LANG

**KHOA CÔNG NGHỆ THÔNG TIN**

Ảnh có chứa biểu tượng, Phông chữ, Đồ họa, văn bản

Mô tả được tạo tự động

**BÁO CÁO ĐỒ ÁN MÔN HỌC HK242**

**NHẬP MÔN PHÂN TÍCH DỮ LIỆU VÀ HỌC SÂU**

**ĐỀ TÀI : NHẬN DẠNG HÌNH ẢNH BẰNG CÁC**

**MÔ HÌNH HUẤN LUYỆN VỚI DỮ LIỆU CIFAR-10**

**GVHD : Nguyễn Thị Mỹ Linh**

TP. Hồ Chí Minh – năm 2025

**LỜI CẢM ƠN**-------------

Lời đầu tiên, em xin gửi lời cảm ơn chân thành đến cô **Nguyễn Thị Mỹ Linh**. Trong suốt quá trình học tập và tìm hiểu bộ môn “Nhập môn phân tích dữ liệu và học sâu”, em đã nhận được sự quan tâm giúp đỡ, hướng dẫn tận tình và đầy tâm huyết từ cô. Cô đã giúp em hiểu rõ hơn về những kiến thức chuyên môn, định hướng cách tiếp cận và giải quyết vấn đề một cách logic và khoa học.

Bên cạnh đó, cô luôn tạo điều kiện thuận lợi để em có môi trường học tập, nghiên cứu tốt nhất, sẵn sàng giải đáp những thắc mắc dù là nhỏ nhất và truyền đạt những kinh nghiệm quý báu để em áp dụng vào thực tế.

Do thời gian và năng lực còn hạn chế, bài báo cáo không thể tránh khỏi những thiếu sót. Kính mong cô nhận xét, đóng góp ý kiến để em có thể học hỏi và hoàn thiện hơn trong tương lai.

Một lần nữa, em xin chân thành cảm ơn!

CÁC THÀNH VIÊN TRONG NHÓM

|  |  |  |  |
| --- | --- | --- | --- |
| **HỌ VÀ TÊN** | **MSSV** | **CÔNG VIỆC** | **MỨC ĐỘ HOÀN THÀNH** |
| TẠ GIA BẢO | 2274802010050 | LEADER | 100% |
| LÊ HỒNG PHÚC | 2274802010676 | A, B, E, F | 100% |
| NGUYỄN HOÀNG PHÚC | 2274802010681 | A, C, D, F, E | 100% |
| LÊ TRẦN KHÔI NGUYÊN | 2274802010584 | A, D, E | 100% |
| NGUYỄN GIA ĐẠI | 2274802010142 | A, D, E | 100% |

A: Tìm hiểu đề tài và khai thác

B: Triển khai code

C: Triển khai word

D: Triển Khai PPT

E: Điều chỉnh word và PPT

F: Kiểm tra code

Nhận xét của giảng viên

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………

**MỤC LỤC**

[CHƯƠNG 1: CƠ SỞ LÝ THUYẾT 6](#_Toc194443619)

[1. LÝ DO CHỌN ĐỀ TÀI 6](#_Toc194443620)

[2. MỤC TIÊU 6](#_Toc194443621)

[CHƯƠNG 2: PHÂN TÍCH TỔNG QUAN 7](#_Toc194443622)

[1. MÔ TẢ DỮ LIỆU 7](#_Toc194443623)

[2. CÁC MÔ HÌNH HUẤN LUYỆN 8](#_Toc194443624)

[2.1 Mạng nơ-ron tích chập (Convolutional Neural Network - CNN) 8](#_Toc194443625)

[2.2 ResNet (Residual Network) 10](#_Toc194443626)

[3. QUY TRÌNH PHÁT TRIỂN 11](#_Toc194443627)

[4. ĐÁNH GIÁ, NHẬN XÉT: 18](#_Toc194443628)

[4.1 Đánh giá 18](#_Toc194443629)

[4.2 Nhận xét 18](#_Toc194443630)

[CHƯƠNG 3: KẾT LUẬN 19](#_Toc194443631)

[1. KẾT QUẢ ĐẠT ĐƯỢC 19](#_Toc194443632)

[2. HẠN CHẾ VÀ HƯỚNG PHÁT TRIỂN 19](#_Toc194443633)

[3. KẾT LUẬN CHUNG 20](#_Toc194443634)

# CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

# LÝ DO CHỌN ĐỀ TÀI

# Trong bối cảnh trí tuệ nhân tạo và thị giác máy tính ngày càng phát triển, bài toán nhận dạng hình ảnh đóng vai trò quan trọng trong nhiều lĩnh vực như xe tự hành, y tế, thương mại điện tử và an ninh. Tập dữ liệu CIFAR-10 là một trong những bộ dữ liệu phổ biến và tiêu chuẩn để đánh giá hiệu suất của các mô hình học máy, đặc biệt là với mạng nơ-ron tích chập (CNN).

# Việc lựa chọn đề tài này xuất phát từ tính thực tiễn và thách thức của CIFAR-10. Với CIFAR-10 đòi hỏi mô hình phải có khả năng trích xuất đặc trưng mạnh mẽ để đạt độ chính xác cao. Hơn nữa, nghiên cứu trên tập dữ liệu này giúp làm quen với các kỹ thuật học sâu, tối ưu hóa mô hình, cũng như áp dụng vào các bài toán phức tạp hơn trong tương lai.

# Do đó, việc nghiên cứu và phát triển mô hình nhận dạng hình ảnh trên CIFAR-10 không chỉ giúp nâng cao kiến thức về trí tuệ nhân tạo mà còn tạo tiền đề cho các ứng dụng thực tế trong nhiều lĩnh vực công nghệ.

# MỤC TIÊU

Mục tiêu của tập dữ liệu CIFAR-10 là huấn luyện các mô hình học máy, đặc biệt là các mạng nơ-ron tích chập (CNN), để phân loại hình ảnh thành 10 lớp khác nhau, ví dụ như dự đoán các đối tượng như máy bay, xe hơi, chim, mèo, v.v. Mỗi hình ảnh trong CIFAR-10 có thể được gán một nhãn từ 1 trong 10 lớp này.

Input đầu vào: Các hình ảnh màu 32x32 pixels, là đầu vào cho mô hình.

Nhãn (Output): Nhãn phân loại của mỗi hình ảnh, xác định đối tượng mà hình ảnh đại diện (ví dụ, "dog", "airplane", "cat", v.v.).

Ý nghĩa: CIFAR-10 giúp nghiên cứu và phát triển các mô hình phân loại hình ảnh tự động. Nó cung cấp một nền tảng cho việc thử nghiệm và đánh giá các kỹ thuật học sâu, đặc biệt là các mạng nơ-ron tích chập (CNNs), trong việc xử lý và phân loại các hình ảnh nhỏ với độ phân giải thấp. Tập dữ liệu này giúp hiểu và cải thiện khả năng nhận diện hình ảnh trong các ứng dụng như nhận diện đối tượng, xe tự lái, và các hệ thống trí tuệ nhân tạo khác.

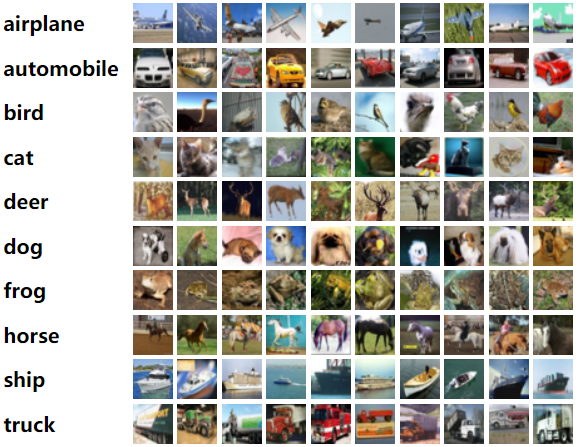
# CHƯƠNG 2: PHÂN TÍCH TỔNG QUAN

# MÔ TẢ DỮ LIỆU

Tập dữ liệu CIFAR-10 được phát triển bởi nhóm nghiên cứu tại **Canadian Institute for Advanced Research (CIFAR)**. Tập dữ liệu này được công bố lần đầu tiên vào năm 2009 bởi **Alex Krizhevsky**, Geoffrey Hinton và các cộng sự.

[CIFAR-10](http://www.cs.toronto.edu/~kriz/cifar.html)  là một tập dữ liệu thị giác máy tính đã được thiết lập được sử dụng để nhận dạng đối tượng. Đây là một tập hợp con của [tập dữ liệu 80 triệu hình ảnh nhỏ](http://groups.csail.mit.edu/vision/TinyImages/) và bao gồm 60.000 hình ảnh màu 32x32 chứa một trong 10 lớp đối tượng, với 6000 hình ảnh cho mỗi lớp.

Sau đây là các lớp trong bộ dữ liệu, cũng như 10 ảnh ngẫu nhiên từ mỗi lớp:



*Hình 1: Các lớp trong dữ liệu CIFAR-10*

Mỗi lớp có 6,000 hình ảnh, trong đó:

* Tập huấn luyện: 5,000 hình ảnh cho mỗi lớp (tổng cộng 50,000 hình ảnh).
* Tập kiểm tra: 1,000 hình ảnh cho mỗi lớp (tổng cộng 10,000 hình ảnh).

Tỷ lệ giữa tập huấn luyện và tập kiểm tra:

* Tập huấn luyện: 50,000 / 60,000 = 83.33%
* Tập kiểm tra: 10,000 / 60,000 = 16.67%

# CÁC MÔ HÌNH HUẤN LUYỆN

## Mạng nơ-ron tích chập (Convolutional Neural Network - CNN)

CNN (Mạng Nơ-ron Tích Chập) là một loại mạng nơ-ron nhân tạo đặc biệt mạnh mẽ trong xử lý ảnh và thị giác máy tính. Nó được thiết kế để tự động trích xuất đặc trưng từ hình ảnh mà không cần tiền xử lý phức tạp như các phương pháp truyền thống.

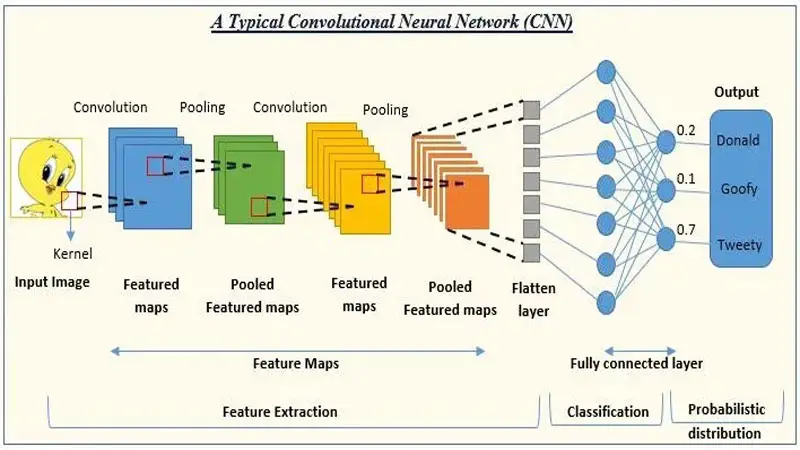
CNN có khả năng nhận dạng và học các đặc trưng không gian trong hình ảnh, chẳng hạn như các cạnh, kết cấu và hình dạng. Điều này giúp CNN rất mạnh mẽ trong việc phân loại hình ảnh.

Cấu trúc của CNN sẽ có 4 lớp:

* **Convolutional layer**: là thành phần cốt lõi trong CNN, đóng vai trò thực hiện các phép toán trên bốn đặc trưng quan trọng sau:
* Filter Map: Được áp dụng trực tiếp lên các vùng hình ảnh trong ma trận đầu vào. Cấu trúc bên trong của lớp này thường là một ma trận ba chiều chứa các tham số (Parameters).
* Stride: Là khoảng dịch chuyển của bộ lọc (Filter Map) trên ảnh, thường di chuyển từ trái sang phải và từ trên xuống dưới để quét toàn bộ dữ liệu.
* Padding**:** Là kỹ thuật thêm các giá trị 0 vào xung quanh ảnh đầu vào nhằm giữ nguyên kích thước hoặc kiểm soát thông tin biên
* Feature Map: Kết quả đầu ra sau khi Filter Map quét qua ảnh đầu vào. Mỗi lần quét sẽ tạo ra một kết quả khác nhau, phản ánh các đặc trưng khác nhau của ảnh.
* **Relu Layer** là một hàm kích hoạt trong mạng nơ-ron, giúp mô hình học hiệu quả hơn. Nó mô phỏng cách nơ-ron truyền tín hiệu qua axon và phổ biến cùng các hàm như Tanh, Sigmoid, Leaky ReLU. ReLU giúp tăng tốc tính toán nhưng cần chú ý các vấn đề như learning rate và dead unit để tránh ảnh hưởng hiệu suất.
* **Pooling Layer** giúp giảm số lượng tham số bằng cách tối ưu hóa việc sắp xếp các lớp Convolutional Layer. Hai loại phổ biến nhất hiện nay là Max Pooling (lấy giá trị lớn nhất trong vùng quét) và Average Pooling (lấy giá trị trung bình).
* **Fully Connected Layer** hoạt động sau khi **Convolutional Layer** và **Pooling Layer** xử lý dữ liệu, **Fully Connected Layer** (FC Layer) sẽ tổng hợp và đưa ra kết quả cuối cùng. Nếu đầu ra ở dạng mode, FC Layer giúp kết nối và mở rộng các kết quả, tạo ra nhiều output hơn.

Tóm tắt quy trình nhận diện hình ảnh của CNN:

* Ảnh đầu vào được xử lý qua nhiều lớp tích chập và pooling để trích xuất đặc trưng.
* Sau khi trích xuất, dữ liệu được chuyển sang dạng vector nhờ Flatten Layer.
* Fully Connected Layer phân tích và đưa ra dự đoán cuối cùng.
* Kết quả là xác suất của từng lớp (Tweety, Donald, Goofy).



*Hình 2: Ví dụ mô hình hoạt động của CNN*

## ResNet (Residual Network)

ResNet sử dụng một kiến trúc đặc biệt gọi là **residual block**. Thay vì học trực tiếp một phép biến đổi đầu ra, mỗi block học sự khác biệt (residual) giữa đầu vào và đầu ra. Điều này giúp giảm thiểu vấn đề gradient biến mất (vanishing gradient) khi đào tạo các mạng sâu.

Đặc điểm của ResNet là các skip connections, nơi thông tin từ các tầng trước được truyền trực tiếp tới các tầng sau. Điều này giúp tránh việc mất mát thông tin và cải thiện khả năng huấn luyện mạng nơ-ron sâu.

ResNet có thể được xây dựng với nhiều tầng khác nhau, ví dụ như ResNet-34, ResNet-50, ResNet-101, ResNet-110, với số lượng tầng ngày càng tăng. Các mô hình này đều sử dụng residual blocks để cải thiện hiệu quả huấn luyện.

# QUY TRÌNH PHÁT TRIỂN

Sử dụng bộ dữ liệu CIFAR-10 xuống và chia làm 10 folder ảnh tương ứng với 10 mục của CIFAR-10.

Thực hiện việc xử lý và làm sạch dữ liệu ảnh từ bộ dữ liệu CIFAR-10:

* Resize ảnh về kích thước 32x32.
* Chuẩn hóa giá trị pixel về khoảng 0-1.
* Chuyển đổi kiểu dữ liệu của ảnh sang float32.

import os

import cv2

import numpy as np

from tqdm import tqdm

# Thư mục chứa dữ liệu CIFAR-10 gốc

data\_path = "data"

# Thư mục để lưu ảnh sau khi làm sạch

cleaned\_path = "cleaned\_cifar10"

os.makedirs(cleaned\_path, exist\_ok=True)

# Duyệt qua từng thư mục con (các lớp ảnh)

for class\_name in os.listdir(data\_path):

    class\_path = os.path.join(data\_path, class\_name)

    cleaned\_class\_path = os.path.join(cleaned\_path, class\_name)

    # Tạo thư mục mới cho lớp nếu chưa tồn tại

    os.makedirs(cleaned\_class\_path, exist\_ok=True)

    if os.path.isdir(class\_path):

        for img\_name in tqdm(os.listdir(class\_path), desc=f"Processing {class\_name}"):

            img\_path = os.path.join(class\_path, img\_name)

            new\_img\_path = os.path.join(cleaned\_class\_path, img\_name)

            try:

                # Đọc ảnh

                img = cv2.imread(img\_path)

                # Kiểm tra ảnh lỗi

                if img is None:

                    print(f"Removing corrupted image: {img\_path}")

                    continue

                # Chuyển ảnh về RGB nếu cần

                if len(img.shape) == 2 or img.shape[2] != 3:

                    img = cv2.cvtColor(img, cv2.COLOR\_GRAY2RGB)

                # Resize ảnh về 32x32

                img = cv2.resize(img, (32, 32))

                # Chuẩn hóa giá trị pixel về 0-1

                img = img.astype(np.float32) / 255.0

                # Lưu ảnh vào thư mục mới

                cv2.imwrite(new\_img\_path, (img \* 255).astype(np.uint8))

            except Exception as e:

                print(f"Error processing {img\_path}: {e}")

Sau khi đã có ảnh làm sạch thì ta training các tập ảnh đó với các mô hình:

import tensorflow as tf

from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout, BatchNormalization, Input

from tensorflow.keras.models import Model

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from tensorflow.keras.optimizers import Adam

import pickle

# Xây dựng mô hình CNN

def build\_cnn(input\_shape, num\_classes):

    inputs = Input(shape=input\_shape)

    x = Conv2D(32, (3, 3), activation='relu', padding='same')(inputs)

    x = BatchNormalization()(x)

    x = MaxPooling2D((2, 2))(x)

    x = Conv2D(64, (3, 3), activation='relu', padding='same')(x)

    x = BatchNormalization()(x)

    x = MaxPooling2D((2, 2))(x)

    x = Conv2D(128, (3, 3), activation='relu', padding='same')(x)

    x = BatchNormalization()(x)

    x = MaxPooling2D((2, 2))(x)

    x = Flatten()(x)

    x = Dense(128, activation='relu')(x)

    x = Dropout(0.5)(x)

    outputs = Dense(num\_classes, activation='softmax')(x)

    model = Model(inputs, outputs)

    return model

# Đường dẫn tới thư mục chứa dữ liệu

train\_dir = 'backend/cifar-10-upscaled'

# Chuẩn bị dữ liệu

train\_datagen = ImageDataGenerator(

    rescale=1.0 / 255.0,

    rotation\_range=15,

    width\_shift\_range=0.1,

    height\_shift\_range=0.1,

    horizontal\_flip=True,

    validation\_split=0.2

)

train\_generator = train\_datagen.flow\_from\_directory(

    train\_dir,

    target\_size=(32, 32),

    batch\_size=32,

    class\_mode='categorical',

    subset='training'

)

validation\_generator = train\_datagen.flow\_from\_directory(

    train\_dir,

    target\_size=(32, 32),

    batch\_size=32,

    class\_mode='categorical',

    subset='validation'

)

# Xây dựng mô hình CNN

input\_shape = (32, 32, 3)

num\_classes = 10

model = build\_cnn(input\_shape, num\_classes)

# Biên dịch mô hình

model.compile(optimizer=Adam(learning\_rate=0.001),

              loss='categorical\_crossentropy',

              metrics=['accuracy'])

# Huấn luyện mô hình

history = model.fit(

    train\_generator,

    epochs=250,

    validation\_data=validation\_generator

)

# Lưu history vào file .pkl

with open('backend/models/history\_cnn2.pkl', 'wb') as f:

    pickle.dump(history.history, f)

# Lưu mô hình dưới dạng .h5

model.save('backend/models/cnn\_model.h5')

# Đánh giá mô hình

loss, accuracy = model.evaluate(validation\_generator)

print(f"Validation Loss: {loss}")

print(f"Validation Accuracy: {accuracy}")

Với mô hình CNN thì sau khi huấn luyện 80 vòng có kết quả như sau

Ảnh có chứa văn bản, biểu đồ, hàng, Sơ đồ

Nội dung do AI tạo ra có thể không chính xác.

Nhận dạng hình ảnh qua 2 tấm ảnh:

Ảnh có chứa động vật có vú, Con hươu, văn bản, động vật hoang dã

Nội dung do AI tạo ra có thể không chính xác.Ảnh có chứa văn bản, Phương tiện đường bộ, phương tiện, bánh xe

Nội dung do AI tạo ra có thể không chính xác.

import tensorflow as tf

from tensorflow.keras.layers import Conv2D, BatchNormalization, Activation, Add, GlobalAveragePooling2D, Dense, Input

from tensorflow.keras.models import Model

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from tensorflow.keras.optimizers import Adam

import pickle

# Hàm xây dựng một ResNet block

def resnet\_block(inputs, filters, strides=1):

    # Nhánh chính

    x = Conv2D(filters, (3, 3), strides=strides, padding="same")(inputs)

    x = BatchNormalization()(x)

    x = Activation('relu')(x)

    x = Conv2D(filters, (3, 3), strides=1, padding="same")(x)

    x = BatchNormalization()(x)

    # Nhánh shortcut

    if strides != 1 or inputs.shape[-1] != filters:

        shortcut = Conv2D(filters, (1, 1), strides=strides, padding="same")(inputs)

        shortcut = BatchNormalization()(shortcut)

    else:

        shortcut = inputs

    # Kết hợp nhánh chính và shortcut

    x = Add()([x, shortcut])

    x = Activation('relu')(x)

    return x

# Xây dựng ResNet

def build\_resnet(input\_shape, num\_classes):

    inputs = Input(shape=input\_shape)

    # Tầng đầu vào

    x = Conv2D(64, (3, 3), strides=1, padding="same")(inputs)

    x = BatchNormalization()(x)

    x = Activation('relu')(x)

    # Các ResNet blocks

    x = resnet\_block(x, 64, strides=1)

    x = resnet\_block(x, 64, strides=1)

    x = resnet\_block(x, 128, strides=2)

    x = resnet\_block(x, 128, strides=1)

    x = resnet\_block(x, 256, strides=2)

    x = resnet\_block(x, 256, strides=1)

    # Tầng đầu ra

    x = GlobalAveragePooling2D()(x)

    outputs = Dense(num\_classes, activation='softmax')(x)

    # Tạo mô hình

    model = Model(inputs, outputs)

    return model

# Đường dẫn tới thư mục chứa dữ liệu

train\_dir = 'backend/cleaned\_cifar10'

# Chuẩn bị dữ liệu

train\_datagen = ImageDataGenerator(

    rescale=1.0 / 255.0,

    rotation\_range=15,

    width\_shift\_range=0.1,

    height\_shift\_range=0.1,

    horizontal\_flip=True,

    validation\_split=0.2

)

train\_generator = train\_datagen.flow\_from\_directory(

    train\_dir,

    target\_size=(32, 32),

    batch\_size=32,

    class\_mode='categorical',

    subset='training'

)

validation\_generator = train\_datagen.flow\_from\_directory(

    train\_dir,

    target\_size=(32, 32),

    batch\_size=32,

    class\_mode='categorical',

    subset='validation'

)

# Xây dựng mô hình ResNet

input\_shape = (32, 32, 3)

num\_classes = 10

model = build\_resnet(input\_shape, num\_classes)

# Biên dịch mô hình

model.compile(optimizer=Adam(learning\_rate=0.001),

              loss='categorical\_crossentropy',

              metrics=['accuracy'])

# Huấn luyện mô hình

history = model.fit(

    train\_generator,

    epochs=50,

    validation\_data=validation\_generator

)

# Lưu history vào file .pkl

with open('backend/models/history\_resnet.pkl', 'wb') as f:

    pickle.dump(history.history, f)

# Lưu mô hình dưới dạng .h5

model.save('backend/models/resnet\_model.h5')

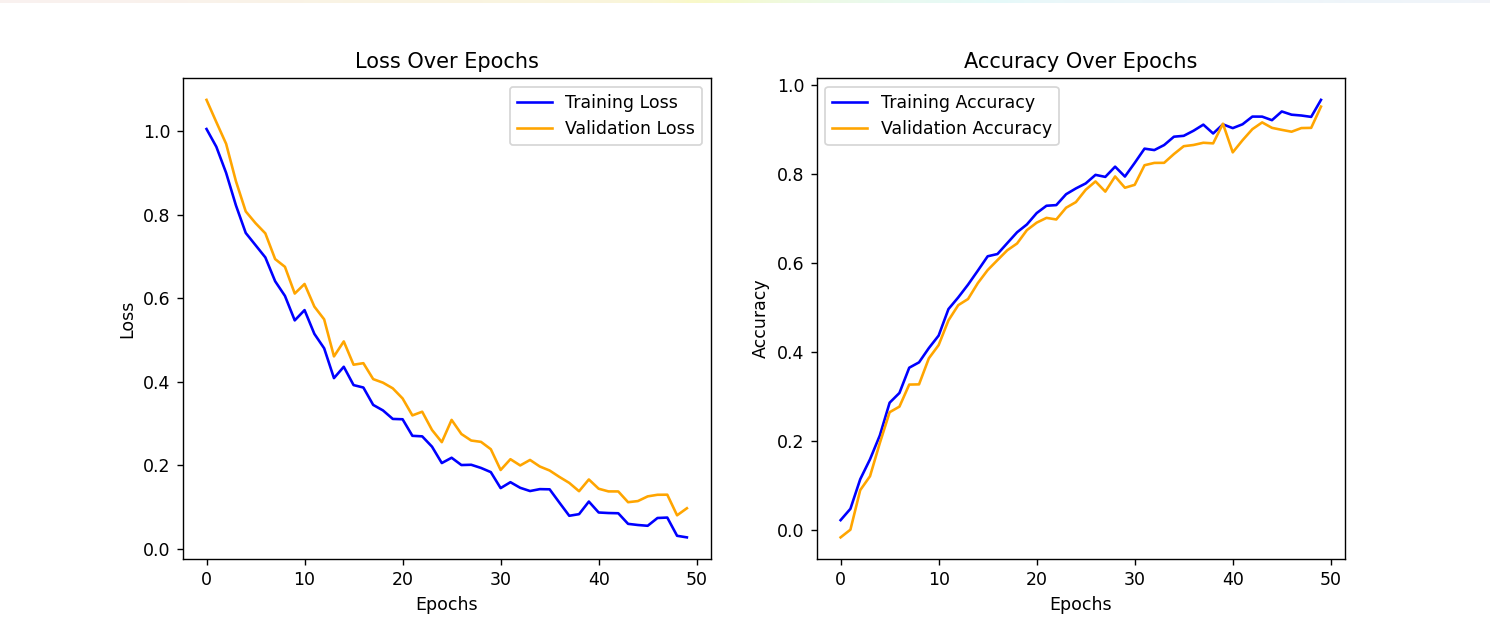
# Đánh giá mô hình

loss, accuracy = model.evaluate(validation\_generator)

print(f"Validation Loss: {loss}")

print(f"Validation Accuracy: {accuracy}")

Với mô hình ResNet sau khi huấn luyện 50 vòng có kết quả như sau:



Ảnh có chứa văn bản, phương tiện, Phương tiện đường bộ, bánh xe

Nội dung do AI tạo ra có thể không chính xác. Nhận dạng hình ảnh qua 2 tấm ảnh:

Ảnh có chứa động vật có vú, Con hươu, văn bản, động vật hoang dã

Nội dung do AI tạo ra có thể không chính xác.

# ĐÁNH GIÁ, NHẬN XÉT:

## 4.1 Đánh giá

Mô hình CNN:

* Mô hình CNN có một mức độ mất mát (loss) cao hơn một ít so với mô hình ResNet , nhưng vẫn duy trì một độ chính xác hợp lý. Điều này có thể chỉ ra rằng mô hình đang học được một số đặc trưng nhưng vẫn chưa hoàn toàn tối ưu.
* Có thể cần tinh chỉnh thêm các tham số (learning rate, batch size) hoặc cải thiện dữ liệu đầu vào để cải thiện độ chính xác.

Mô hình ResNet:

* ResNet đạt được kết quả tốt.
* ResNet có khả năng học các đặc trưng phức tạp nhờ vào các lớp residual (các lớp kết nối tắt), giúp giảm hiện tượng vanishing gradient và cải thiện hiệu suất học.
* Đây là mô hình đáng xem xét để triển khai thực tế, vì nó cung cấp một sự cân bằng tốt giữa độ chính xác và mất mát.

## 4.2 Nhận xét

ResNet là mô hình có kết quả tốt nhất, với độ chính xác cao và mất mát thấp. Đây là lựa chọn hợp lý nếu muốn có một mô hình ổn định và hiệu quả.

CNN cho thấy hiệu suất ổn định nhưng cần cải thiện để tối ưu hơn nữa, có thể thử tinh chỉnh thêm các siêu tham số hoặc áp dụng các kỹ thuật cải thiện độ chính xác như data augmentation hoặc dropout.

# CHƯƠNG 3: KẾT LUẬN

Trong báo cáo này, chúng tôi đã tiến hành nghiên cứu và so sánh hiệu suất của hai mô hình học sâu phổ biến là Mạng Nơ-ron Tích Chập (CNN) và ResNet trên tập dữ liệu CIFAR-10. Thông qua các bước phân tích dữ liệu, thiết lập mô hình, huấn luyện và đánh giá kết quả, chúng tôi đã rút ra một số kết luận quan trọng.

**1. KẾT QUẢ ĐẠT ĐƯỢC**

Mô hình CNN có khả năng nhận dạng hình ảnh ở mức độ tốt, tuy nhiên vẫn tồn tại một số hạn chế về độ chính xác do hiện tượng mất mát thông tin trong quá trình lan truyền.

Mô hình ResNet với kiến trúc Residual Block giúp cải thiện đáng kể khả năng học sâu, giảm thiểu hiện tượng vanishing gradient, từ đó đạt được kết quả tốt hơn về độ chính xác.

ResNet cho độ chính xác cao hơn so với CNN, đặc biệt trong các trường hợp hình ảnh có nhiều chi tiết phức tạp.

**2. HẠN CHẾ VÀ HƯỚNG PHÁT TRIỂN**

Hạn chế:

* Mô hình CNN có xu hướng hội tụ chậm hơn và cần tối ưu thêm các tham số như **learning rate, batch size** để đạt hiệu suất tốt hơn.
* Cả hai mô hình đều có thể gặp khó khăn khi xử lý dữ liệu nhiễu hoặc có độ phân giải cao hơn.

Hướng phát triển:

* Cải tiến mô hình bằng cách thử nghiệm với các kiến trúc Deep Learning hiện đại hơn như EfficientNet, Vision Transformer (ViT).
* Ứng dụng Data Augmentation để cải thiện độ chính xác và giảm overfitting.
* Sử dụng Transfer Learning để tận dụng sức mạnh của các mô hình đã được huấn luyện trước trên tập dữ liệu lớn hơn.

**3. KẾT LUẬN CHUNG**

Nhìn chung, kết quả nghiên cứu cho thấy ResNet là một lựa chọn hiệu quả hơn so với CNN trong bài toán phân loại hình ảnh với CIFAR-10. Tuy nhiên, hiệu suất của mỗi mô hình vẫn có thể cải thiện đáng kể nếu áp dụng các kỹ thuật tối ưu hóa phù hợp. Nghiên cứu này là tiền đề quan trọng để tiếp tục triển khai các ứng dụng nhận dạng hình ảnh trong thực tế, đặc biệt là trong các lĩnh vực như xe tự hành, giám sát an ninh, y tế và thương mại điện tử.