

TRƯỜNG ĐẠI HỌC SÀI GÒN

KHOA CÔNG NGHỆ THÔNG TIN

**----------****----------**

BÁO CÁO ĐỒ ÁN CUỐI KỲ

HỌC PHẦN: CƠ SỞ TRÍ TUỆ NHÂN TẠO

ĐỀ TÀI:

**XÂY DỰNG MÔ HÌNH NHẬN DIỆN HÌNH ẢNH ĐỘNG VẬT**

|  |  |  |
| --- | --- | --- |
| **MÃ HỌC PHẦN:** | **841110** | |
| **NHÓM MÔN HỌC:** | **04** | |
| **HỌC KỲ:** | **02** | |
| **GVHD**: | **PHẠM HOÀNG VƯƠNG** | |
| **THÀNH VIÊN**: | **3122410149** | **Nguyễn Quốc Huy**  **(Nhóm trưởng)** |
|  | **3122410098** | **Nguyễn Vũ Hào** |
|  | **3122410216** | **Đoàn Hoàng Long** |
|  | **3123410316** | **Nguyễn Huỳnh Tài** |
|  | **3122410191** | **Phạm Tấn Khương** |

**TP HỒ CHÍ MINH, THÁNG 4 NĂM 2023**

**MỤC LỤC**

[PHÂN CÔNG VÀ ĐÁNH GIÁ 1](#_Toc164901243)

[1. Chuẩn bị cho phần thuyết trình và báo cáo: 1](#_Toc164901244)

[2. Phần kĩ thuật và thực hành đồ án: 2](#_Toc164901245)

[NHẬN XÉT CỦA GIẢNG VIÊN 3](#_Toc164901246)

[LỜI MỞ ĐẦU 4](#_Toc164901247)

[CHƯƠNG 1: CƠ SỞ LÝ THUYẾT 4](#_Toc164901248)

[1. Giới thiệu về ngôn ngữ Python 4](#_Toc164901249)

[2. Các thư viện của Python được sử dụng 5](#_Toc164901250)

[2.1 Back-end 5](#_Toc164901251)

[2.2 Giao diện demo: 5](#_Toc164901252)

[4. Lý thuyết CNN 8](#_Toc164901253)

[4.1 Convolutional Neural Network (CNN): 8](#_Toc164901254)

[Quy trình Huấn luyện (Training Process): 9](#_Toc164901255)

[CHƯƠNG 2: XÂY DỰNG CHƯƠNG TRÌNH NHẬN DIỆN ĐỘNG VẬT 11](#_Toc164901256)

[1.Phân tích bài toán ứng dụng AI (Nhận diện hình ảnh) 11](#_Toc164901257)

[CHƯƠNG 3: CÀI ĐẶT, DEMO CHƯƠNG TRÌNH 12](#_Toc164901258)

[1. Cài đặt 12](#_Toc164901259)

[2. Demo chương trình 14](#_Toc164901260)

[KẾT LUẬN 16](#_Toc164901261)

[1. ĐÁNH GIÁ NHỮNG NỘI DUNG ĐÃ THỰC HIỆN 16](#_Toc164901262)

[2. ĐÁNH GIÁ NHỮNG NỘI DUNG CHƯA THỰC HIỆN 16](#_Toc164901263)

[3. HƯỚNG PHÁT TRIỂN HOÀN THIỆN CHƯƠNG TRÌNH 16](#_Toc164901264)

# PHÂN CÔNG VÀ ĐÁNH GIÁ

## Chuẩn bị cho phần thuyết trình và báo cáo:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Thứ tự** | **Họ tên** | **Mã số sinh viên** | **Nội dung công việc** | **Mức độ đóng góp** | **Giảng viên đánh giá** |
| 1 | Nguyễn Quốc Huy | 3122410148 | Soạn lý thuyết báo cáo, chuẩn bị Powerpoint | 20% |  |
| 2 | Đoàn Hoàng Long | 3122410216 | Soạn lý thuyết báo cáo, chuẩn bị Powerpoint | 20% |  |
| 3 | Nguyễn Huỳnh Tài | 3123410316 | chuẩn bị Powerpoint | 20% |  |
| 4 | Nguyễn Vũ Hào | 3122410098 | Soạn lý thuyết báo cáo, chuẩn bị Powerpoint | 20% |  |
| 5 | Phạm Tấn Khương | 3122410191 | Soạn lý thuyết báo cáo, chuẩn bị Powerpoint | 20% |  |

**\*NOTE:** Cả nhóm ngồi thảo luận về làm ppt và phân công hợp lý để hoàn thành đúng tiến độ báo cáo

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Thứ tự** | **Họ tên** | **Mã số sinh viên** | **Nội dung công việc** | **Mức độ đóng góp** | **Giảng viên đánh giá** |
| 1 | Nguyễn Quốc Huy | 3122410148 | Train model | 20% |  |
| 2 | Đoàn Hoàng Long | 3122410216 | Train model | 20% |  |
| 3 | Nguyễn Huỳnh Tài | 3123410316 |  |  |  |
| 4 | Nguyễn Vũ Hào | 3122410098 | Train model | 20% |  |
| 5 | Phạm Tấn Khương | 3122410191 | Chuẩn bị dataset, và nhãn | 20% |  |

## Phần kĩ thuật và thực hành đồ án:

# NHẬN XÉT CỦA GIẢNG VIÊN

# LỜI MỞ ĐẦU

## CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

### 1. Giới thiệu về ngôn ngữ Python

#### 1.1 Ngôn Ngữ Python là gì?

Python là một ngôn ngữ lập trình bậc cao, được tạo ra bởi Guido van Rossum vào những năm 1980 và phát triển tiếp theo trong thập niên 1990. Python là một ngôn ngữ lập trình đơn giản, dễ học và dễ sử dụng, có thể được sử dụng để xây dựng các ứng dụng máy tính, trang web, các hệ thống đám mây, trí tuệ nhân tạo, máy học và các ứng dụng phân tích dữ liệu. Python cũng là một ngôn ngữ lập trình đa nền tảng, có thể chạy trên các hệ điều hành khác nhau như Windows, Linux và MacOS. Python có một cộng đồng lớn và đa dạng, với hàng ngàn thư viện và framework hỗ trợ cho việc phát triển ứng dụng.

#### 1.2 Lợi ích của Python

+ *Dễ học và sử dụng:* Python có cú pháp đơn giản và dễ hiểu, giúp người mới bắt đầu học lập trình dễ dàng tiếp cận và làm quen với các khái niệm lập trình cơ bản.

+ *Đa năng:* Python có thể được sử dụng để phát triển các loại ứng dụng khác nhau như ứng dụng máy tính, web, trí tuệ nhân tạo, máy học, phân tích dữ liệu, đám mây, hệ thống nhúng và nhiều hơn nữa.

+ *Cộng đồng lớn và đa dạng:* Python có một cộng đồng phát triển lớn và đa dạng, với hàng ngàn thư viện và framework hỗ trợ cho việc hát triển ứng dụng.

+ *Tính tương thích cao:* Python là một ngôn ngữ đa nền tảng, có thể chạy trên nhiều hệ điều hành khác nhau như Windows, Linux và MacOS.

+ *Hiệu suất cao:* Python có thể được tối ưu hóa để đạt được hiệu suất cao, đặc biệt là khi sử dụng các thư viện và framework được viết bằng C/C++.

+ *Dễ dàng mở rộng:* Python có tính mở rộng cao, cho phép người dùng tạo ra các module và thư viện riêng để mở rộng chức năng của Python.

+ *Tương tác tốt với các ngôn ngữ khác:* Python có thể tương tác tốt với các ngôn ngữ khác như C/C++, Java, và .NET, giúp cho việc tích hợp các phần mềm và dịch vụ khác nhau trở nên dễ dàng hơn.

##### 1.3 Ứng dụng của Python

Dưới đây là một số ví dụ về các ứng dụng của Python:

+ *Phát triển ứng dụng web*: Python có thể được sử dụng để phát triển các ứng dụng web, với các framework như Django, Flask và Pyramid.

+ *Trí tuệ nhân tạo và máy học*: Python là một trong những ngôn ngữ phổ biến nhất trong lĩnh vực trí tuệ nhân tạo và máy học, với các thư viện như TensorFlow, Keras và PyTorch.

+ *Phân tích dữ liệu*: Python có các thư viện như Pandas và NumPy để xử lý và phân tích dữ liệu, và Matplotlib và Seaborn để tạo đồ thị và biểu đồ.

+ *Game development*: Python có thể được sử dụng để phát triển game với các thư viện như Pygame và PyOpenGL.

+ *Đám mây*: Python có thể được sử dụng để quản lý và triển khai các hệ thống đám mây với các framework như OpenStack và Ansible.

+ *Hệ thống nhúng*: Python có thể được sử dụng để phát triển các hệ thống nhúng, với các framework như MicroPython và CircuitPython.

+ *Ứng dụng máy tính*: Python có thể được sử dụng để phát triển các ứng dụng máy tính trên nhiều nền tảng khác nhau, với các thư viện như PyQt và wxPython.

## 2. Các thư viện của Python được sử dụng

Các thư viện được sử dụng trong đồ án:

* *Import os*
* *Import cv2*
* *Import numpy*

### 2.1 Back-end

#### 2.1.1 Import os

Thư viện os cung cấp các phương thức để tương tác với hệ thống tệp và thư mục trong Python. Bạn có thể sử dụng nó để thực hiện các hoạt động như tạo, xóa, di chuyển hoặc đổi tên tệp.

Trong code của nhóm em thì có sử dụng **os:**

+ Để tạo đường dẫn đến thư mục chứa biển báo giao thông trong biến image\_dir.

+ Mở file ảnh biển báo giao thông từ thư mục tạo đường dẫn đến file ảnh có thể là một cách thực tiễn khi cần tương tác với nhiều hệ điều hành khác nhau. Tuy nhiên việc này không được thực hiện trực tiếp.

+ Kiểm tra sự tồn tại của thư mục khi truy cập từng biển báo để tránh lỗi không tồn tại hoặc đã bị xóa đi.

#### 2.1.2 Import cv2

Thư viện OpenCV(cv2) là một thư viện mã nguồn mở chuyên dụng cho xử lý hình ảnh và thị giác máy tính. Nó cung cấp các công cụ để đọc, ghi, xử lý và phân tích hình ảnh, cũng như các chức năng nhận dạng đối tượng, phát hiện khuôn mặt, và nhiều hơn nữa.

Cung cấp các chức năng để thực hiện các bước sau:

+ Đọc và xử lý hình ảnh trong đường dẫn img\_path.

+ Chuyển đổi định dạng màu sắc hình ảnh từ RGBA sang RGB.

+ Phân loại hình ảnh, hình ảnh được truyền vào 1 mô hình đã được huấn luyện để phân loại.

#### 2.1.3 Import numpy

Thư viện numpy là một thư viện Python mạnh mẽ cho tính toán khoa học và toán học. Nó cung cấp một đối tượng mảng nhiều chiều (ndarray), các hàm để làm việc với mảng này, và một loạt các công cụ để làm việc với mảng số.

Cung cấp các chức năng để thực hiện các bước sau:

+ *Chuyển đổi hình ảnh*: NumPy cung cấp các phương pháp để chuyển đổi hình ảnh từ một loạt các định dạng khác nhau thành các mảng NumPy, thuận tiện cho việc xử lý tiếp theo.

+ *Thao tác ma trận*: NumPy cung cấp một loạt các hàm và phương thức để thực hiện các phép toán ma trận, như cộng, trừ, nhân, chia, transpoze, và nhiều phép toán khác. Điều này hữu ích trong việc xử lý dữ liệu hình ảnh, ví dụ như khi áp dụng các bộ lọc hình ảnh hoặc biến đổi hình ảnh.

+ *Xử lý dữ liệu số*: NumPy cung cấp các công cụ mạnh mẽ để thực hiện các phép toán số học và tính toán khoa học trên dữ liệu số.Chúng tôi có thể sử dụng NumPy để xử lý các dữ liệu kết quả từ việc nhận diện biển báo giao thông hoặc xử lý hình ảnh.

### 2.2 Giao diện demo:

#### 2.2.1 Widget

Widget là lớp cơ sở cho tất cả các thành phần giao diện người dùng (GUI) trong Kivy.

+ Bất kỳ thành phần giao diện nào, chẳng hạn như nút, nhãn, hộp thoại, hoặc bất kỳ thành phần tùy chỉnh nào tạo ra trong Kivy, đều phải kế thừa từ lớp Widget.

+ Widget cung cấp các tính năng cơ bản như định vị, kích thước, và chứa các thành phần con bên trong.

+ Có thể tùy chỉnh và mở rộng lớp Widget để tạo ra các thành phần giao diện phức tạp và đa dạng cho ứng dụng.

#### 2.2.2 Camera

Camera:

+ Lớp Camera trong Kivy được sử dụng để hiển thị hình ảnh từ máy ảnh của thiết bị hoặc từ nguồn video khác.

+ Khi sử dụng Camera trong ứng dụng, nó sẽ kích hoạt máy ảnh của thiết bị và hiển thị hình ảnh trực tiếp từ máy ảnh đó trên giao diện người dùng.

+ Sử dụng Camera để tạo các ứng dụng liên quan đến việc chụp ảnh hoặc quay video, hoặc để thực hiện nhận dạng hình ảnh hoặc các tác vụ xử lý hình ảnh.

#### 2.2.3 Image, AsyncImage

Image:

+ Lớp Image trong Kivy được sử dụng để hiển thị hình ảnh tĩnh.

+ Sử dụng Image để hiển thị hình ảnh từ các tệp tin cục bộ trên thiết bị hoặc từ URL trên Internet.

+ Đối với hình ảnh tĩnh, cần tải hình ảnh hoặc chỉ định đường dẫn đến hình ảnh muốn hiển thị.

AsyncImage:

+ Lớp AsyncImage trong Kivy cung cấp khả năng hiển thị hình ảnh từ URL một cách bất đồng bộ.

+ Khi sử dụng AsyncImage để hiển thị hình ảnh từ một URL, quá trình tải hình ảnh sẽ không làm đóng băng giao diện người dùng, điều này giúp tránh tình trạng giao diện đóng băng khi tải hình ảnh từ Internet.

+ Chỉ cần chỉ định URL của hình ảnh muốn hiển thị, và AsyncImage sẽ tự động tải và hiển thị hình ảnh đó.

#### 2.2.4 Button

Button: là một thành phần giao diện người dùng cho phép người dùng tương tác bằng cách nhấn vào.

Các đặc điểm chính:

+ *Văn bản*: Bạn có thể đặt văn bản trên nút bằng cách sử dụng thuộc tính text.

+ *Hành vi*: Bạn có thể gán một hàm hoặc phương thức để được gọi khi người dùng nhấn vào nút, thông qua sự kiện on\_press hoặc on\_release.

+ *Màu sắc và giao diện*: Button có các thuộc tính cho phép bạn tùy chỉnh màu sắc, kích thước, hình dạng và kiểu chữ.

+ *Trạng thái*: Bạn có thể kích hoạt hoặc vô hiệu hóa một nút, hoặc thay đổi trạng thái của nó, như nút nhấn hoặc nút chức năng thông qua thuộc tính disabled.

Button là một phần cơ bản nhưng quan trọng trong việc xây dựng giao diện người dùng trong Kivy, và nó thường được sử dụng để tạo ra các tương tác người dùng cơ bản như kích hoạt hành động hoặc chức năng.

#### 2.2.5 Label (MDLabel)

Label(MDLabel): Là một thành phần giao diện người dùng được sử dụng để hiển thị văn bản tĩnh trên màn hình.

Các đặc điểm chính:

+ *Văn bản*: Bạn có thể đặt văn bản để hiển thị trên Label thông qua thuộc tính text.

+ *Căn chỉnh và định dạng văn bản*: Label cho phép bạn căn chỉnh văn bản theo chiều ngang và dọc, thay đổi kích thước của văn bản, font chữ, màu sắc và kiểu chữ.

+ *Trạng thái*: Tương tự như Button, bạn cũng có thể vô hiệu hóa hoặc kích hoạt một Label thông qua thuộc tính disabled.

Label thường được sử dụng để hiển thị tiêu đề, thông điệp hoặc nội dung tĩnh khác trên giao diện người dùng, và nó là một thành phần cơ bản nhưng quan trọng trong việc xây dựng giao diện người dùng trong Kivy.

## 4. Lý thuyết CNN

### 4.1 Convolutional Neural Network (CNN):

+ CNN là một dạng của mạng nơ-ron nhân tạo (neural network) được thiết kế đặc biệt cho việc xử lý dữ liệu không gian như hình ảnh.

+ Cấu trúc của CNN bao gồm các lớp chính như lớp convolution (tích chập), lớp pooling (tổng hợp), và các lớp kết nối đầy đủ (fully connected layers).

+ Lớp convolution giúp mô hình học được các đặc trưng cục bộ từ dữ liệu đầu vào bằng cách áp dụng các bộ lọc (kernels) qua ảnh.

+ Lớp pooling giúp giảm kích thước của đầu vào và giữ lại các đặc trưng quan trọng.

Các lớp kết nối đầy đủ được sử dụng để kết nối các đặc trưng đã học được với các lớp đầu ra để thực hiện phân loại.

**+** *Lớp Convolutional (Convolutional Layer):*

* Lớp này chịu trách nhiệm xử lý đầu vào bằng cách áp dụng các bộ lọc (kernels) qua ảnh đầu vào để tạo ra các đặc trưng cục bộ.
* Các bộ lọc này giúp phát hiện các đặc trưng như cạnh, góc, hoặc các hình dạng cụ thể trong ảnh.
* Kết quả của lớp convolution thường được gọi là các feature map.

**+** *Lớp Tổng hợp (Pooling Layer):*

* Lớp này giúp giảm kích thước của feature map, đồng thời giữ lại các thông tin quan trọng.
* Các phương pháp phổ biến trong pooling là max pooling và average pooling, trong đó max pooling giữ lại giá trị lớn nhất trong mỗi vùng và bỏ qua các giá trị khác

.

**+** *Lớp Kích hoạt (Activation Layer):*

* Lớp này thường được đặt sau lớp convolution và pooling để tạo ra các biểu diễn phi tuyến tính từ đầu vào.
* Hàm kích hoạt phổ biến nhất là ReLU (Rectified Linear Activation), nhưng cũng có thể sử dụng các hàm khác như tanh hoặc sigmoid.
* Lớp Kết nối Đầy đủ (Fully Connected Layer):
* Lớp này thường được sử dụng ở cuối mạng để kết nối các đặc trưng đã học được với các lớp đầu ra để thực hiện phân loại.
* Các lớp fully connected được sử dụng để học các mối quan hệ phức tạp giữa các đặc trưng và lớp đầu ra.

### Quy trình Huấn luyện (Training Process):

- Quy trình huấn luyện của mạng CNN thường bao gồm các bước như sau:

* *Định nghĩa mạng*: Xác định kiến trúc mạng CNN bao gồm số lượng và kiểu của các lớp, kích thước của các bộ lọc, số lượng lớp và kích thước lớp đầu ra.
* *Chuẩn bị dữ liệu*: Tiền xử lý dữ liệu, chia dữ liệu thành các batch, chuẩn hóa dữ liệu nếu cần.
* *Chọn hàm mất mát và thuật toán tối ưu*: Phổ biến nhất là cross-entropy loss function và thuật toán gradient descent hoặc các biến thể như Adam.
* *Huấn luyện mô hình*: Đưa dữ liệu vào mô hình để huấn luyện, điều chỉnh các tham số (weights) sao cho hàm mất mát được giảm thiểu.
* *Đánh giá mô hình*: Sử dụng tập dữ liệu kiểm tra để đánh giá hiệu suất của mô hình, thực hiện đánh giá về độ chính xác hoặc các metric khác.
* Trong mô hình CNN của bạn, quy trình huấn luyện sẽ được thực hiện để học các biểu diễn của các loại biển báo giao thông từ dữ liệu huấn luyện, và sau đó mô hình được sử dụng để phân loại các biển báo mới từ các hình ảnh mới.

Trong đồ án này, mô hình CNN và deep learning được sử dụng để học các mẫu và đặc trưng từ các hình ảnh của các biển báo giao thông, từ đó có thể phân loại chúng thành các loại biển báo khác nhau.

# CHƯƠNG 2: XÂY DỰNG CHƯƠNG TRÌNH NHẬN DIỆN ĐỘNG VẬT

## 1.Phân tích bài toán ứng dụng AI (Nhận diện hình ảnh)

Về bài toán nhận diện hình ảnh, chúng em sử dụng mô hình CNN(Convolutional Neural Network).

Cách sử dụng CNN vào chương trình huấn luyện mô hình nhận diện động vật: chim, mèo, hưu, chó, ếch, ngựa.

Bước 1: Set up

+ Chuẩn bị dataset: cifar10 là một bộ thư dữ liệu của keras, chỉ lấy từ mẫu 3 đến mẫu 8(chim, mèo, hưu, chó, ếch, ngựa.)

+ Chuẩn bị nhãn(label): các tên của biển báo được quy định dựa trên dataset đã chuẩn bị.

Bước 2: Thực hiện

+ Chuyển hóa hình ảnh thành ma trận, kèm theo đó sử dụng thuật toán convolutional layer để tạo ra một ma trận ba chiều(convoled feature).

+ Sử dụng thuật toán Kernel relu, padding: tối ưu hóa dữ liệu đầu vào. Kernel relu làm cho kết quả không âm, còn padding thì làm cho những pixel ở viền ma trận được tính toán nhiều hơn.

+ Max Pooling 2D: tối ưu về kích thước của convoled feature, tăng tốc độ xử lý (giảm dữ liệu một nửa).

+ Drop out: chuyển tham số bằng đơn vị phần trăm để giảm số lượng pixel của convoled feature.

Bước 3: Lặp lại bước 2 bốn lần.

Bước 4: Fully connection (Dense: mạng chằn chịt): Trải dài ma trận được tạo NxMx3 thành ma trận một chiều (N\*M)x1x3. Sử dụng thuật toán trong keras tạo ra 3 ma trận với ma trận thứ 2 có kích thước là 6000x1x3, ma trận thứ 3 là 600x1x3 (bằng với số label đã được chuẩn bị ở bước 1), và ma trận thứ 4 có 60x1x3, ma trận cuối có 6x1x3. Có xảy ra mất mát. Các ma trận một chiều này được liên kết 1-N (phần tử ma trận phía trước liên kết với N phần tử của ma trận phía sau) theo công thức:

tổng số lượng liên kết ma trận phía trước = số phần tử phía trước\*số phần tử ma trận phía sau + số phần từ phía trước

Bước 5: Lưu mô hình.

# CHƯƠNG 3: CÀI ĐẶT, DEMO CHƯƠNG TRÌNH

## 1. Cài đặt

Môi trường thực hiện train và các bước thực hiện:

Môi trường: PyCharm, Visual Studio Code

Các bước train:

* Import các thư viện cần thiết và tải dữ liệu:

*+ import numpy as np*: Import thư viện numpy để làm việc với mảng nhiều chiều.

*+ import matplotlib.pyplot as plt*: Import thư viện matplotlib để trực quan hóa dữ liệu.

*+ from PIL import Image*: Import thư viện PIL để làm việc với hình ảnh.

*+ from sklearn.model\_selection import train\_test\_split*: Import hàm train\_test\_split từ thư viện sklearn để chia dữ liệu thành tập huấn luyện và tập kiểm tra.

*+ from keras.utils import to\_categorical*: Import hàm to\_categorical từ thư viện keras.utils để chuyển đổi nhãn thành dạng one-hot encoding.

*+ from keras.models import Sequential, load\_model*: Import lớp Sequential và hàm load\_model từ thư viện keras.models để xây dựng và tải mô hình.

*+ from keras.layers import Conv2D, MaxPool2D, Dense, Flatten, Dropout*: Import các lớp Conv2D, MaxPool2D, Dense, Flatten, Dropout từ thư viện keras.layers để xây dựng kiến trúc mạng nơ-ron.

* Chuẩn bị dữ liệu:

+Kích thước dữ liệu huấn luyện: (30000, 32, 32, 3)

+Kích thước dữ liệu kiểm tra: (6000, 32, 32, 3)

+ Số lượng nhãn biển báo: 6

+ Các nhãn gồm: chim, mèo, hưu, chó, ếch, ngựa.

Chuyển đổi dữ liệu thành mảng numpy:

+ Chuyển đổi data và labels từ danh sách thành mảng numpy bằng cách sử dụng np.array().

* Chia dữ liệu thành tập huấn luyện và tập kiểm tra:

+ Sử dụng hàm train\_test\_split() để chia data và labels thành X\_train, X\_test, y\_train, y\_test với tỷ lệ tập kiểm tra là 0.2 (20%).

+ Kết quả được lưu vào các biến tương ứng.

* Chuyển đổi nhãn thành dạng one-hot encoding:

+ Sử dụng hàm to\_categorical() từ thư viện keras.utils để chuyển đổi y\_train và y\_test thành dạng one-hot encoding với số lượng lớp là 6.

* Xây dựng mô hình:

+ Tạo một đối tượng mô hình Sequential bằng cách sử dụng model = Sequential().

+ Thêm các lớp Conv2D, MaxPool2D, Dropout, Flatten, và Dense vào mô hình theo kiến trúc đã được định nghĩa.

+ Lớp cuối cùng sử dụng softmax activation để đưa ra dự đoán cho 6 lớp.

* Biên dịch mô hình:

+ Sử dụng compile() để biên dịch mô hình với hàm mất mát là categorical\_crossentropy và tối ưu hóa bằng adam optimizer.

+ Đánh giá mô hình dựa trên chỉ số độ chính xác (accuracy).

* Huấn luyện mô hình:

+ Sử dụng fit() để huấn luyện mô hìnhvới dữ liệu huấn luyện X\_train và nhãn y\_train trong epochs epochs.

+ Sử dụng kích thước batch là 32 và sử dụng tập kiểm tra X\_test và y\_test để đánh giá mô hình sau mỗi epoch.

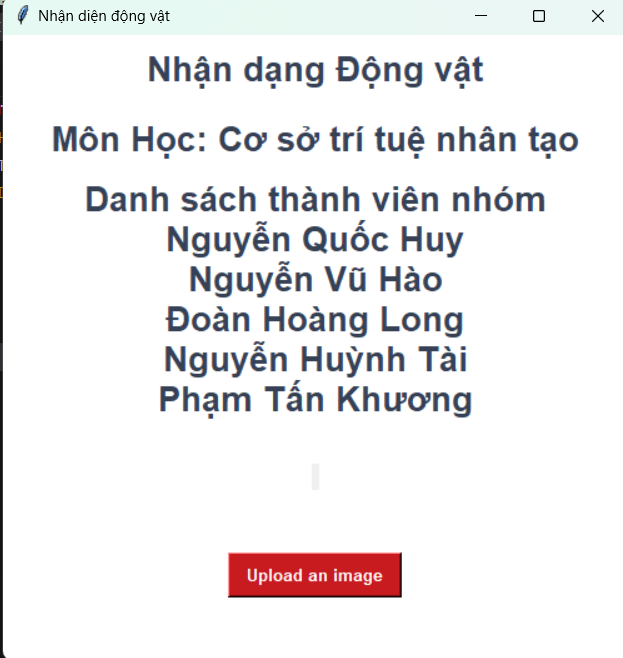
* Lưu mô hình:

+ Sử dụng save() để lưu mô hình vào tệp "my\_model.keras".

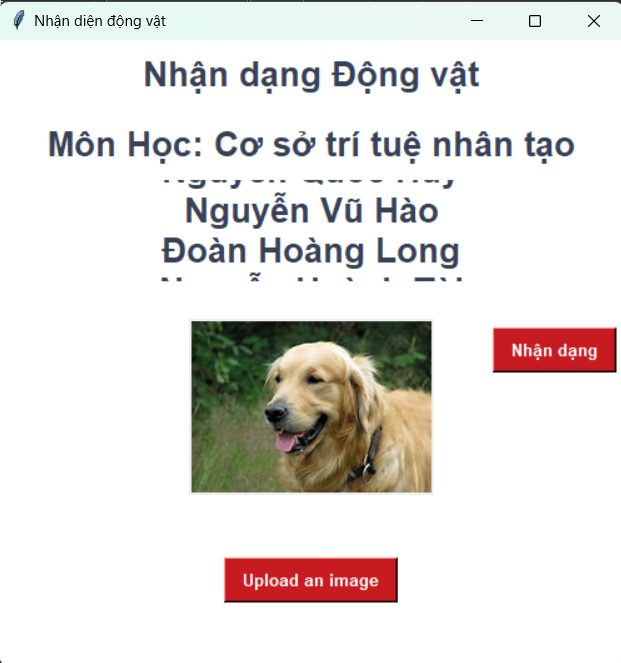
* Trực quan hóa kết quả:

+ Sử dụng matplotlib để vẽ đồ thị cho độ chính xác (accuracy) và hàm mất mát (loss) trên tập huấn luyện và tập kiểm tra.

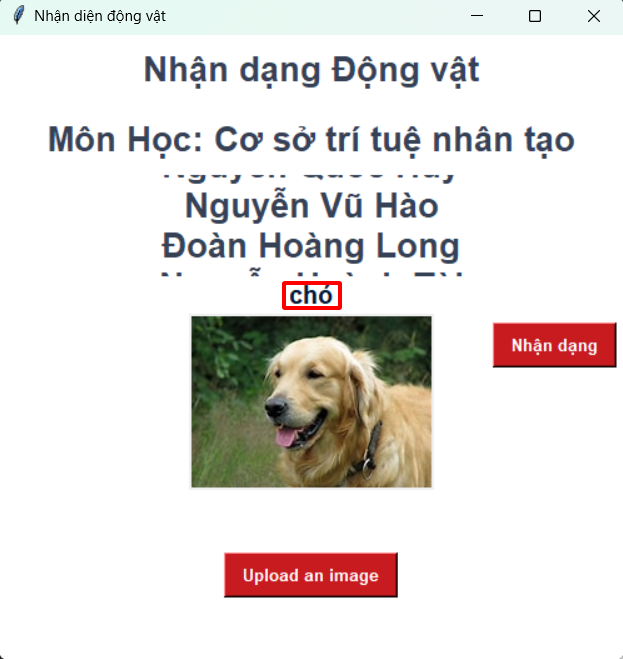
## 2. Demo chương trình



Hình 1. Giao diện ở trang chủ khi khởi động



Hình 2. Giao diện khi upload 1 tấm ảnh bất kì



Hình 3. Giao diện khi nhận diện xong và trả về kết quả

# KẾT LUẬN

### ĐÁNH GIÁ NHỮNG NỘI DUNG ĐÃ THỰC HIỆN

**-**Đã huấn luyên được mô hình và tỉ lệ chính xác tầm 85% với các hình test.

**-** Tạo một giao diện đơn giản để upload hình ảnh cho người dùng sử dụng

### ĐÁNH GIÁ NHỮNG NỘI DUNG CHƯA THỰC HIỆN

* Mô hình dữ liệu chưa đủ lớn
* Nhận diện chưa chính xác so với kì vọng
* Giao diện còn thô sơ

### HƯỚNG PHÁT TRIỂN HOÀN THIỆN CHƯƠNG TRÌNH

* Mở rộng dữ liệu: Để cải thiện mô hình, bạn có thể tăng cường dữ liệu bằng cách thu thập thêm hình ảnh và nhãn để huấn luyện. Điều này giúp mô hình học được nhiều đặc trưng và tăng khả năng nhận diện chính xác.
* Tăng cường độ chính xác: Nếu mô hình chưa đạt đến mức chính xác như kỳ vọng, bạn có thể thử các phương pháp khác nhau để cải thiện hiệu suất nhận diện. Điều này có thể bao gồm việc sử dụng các kiến trúc mạng học sâu tiên tiến hơn, tinh chỉnh siêu tham số của mô hình hiện tại hoặc thử các phương pháp khác nhau để xử lý dữ liệu đầu vào.
* Cải thiện giao diện: Tạo ra một giao diện người dùng trực quan và thân thiện hơn sẽ tăng khả năng sử dụng của hệ thống. Bạn có thể cải thiện giao diện bằng cách thêm tính năng như tải lên nhiều hình ảnh cùng một lúc, hiển thị kết quả nhận diện một cách hợp lý và tối ưu hóa trải nghiệm người dùng chung**.**