

Học viện Công nghệ Bưu chính Viễn thông cơ sở Hồ Chí Minh  
Khoa Công nghệ thông tin  
∞📖∞



**ĐỀ TÀI: NGHIÊN CỨU MACHINE LEARNING  
MÔ HÌNH PIPELINES (MODEL CNN)**

**Môn:** IOT và Ứng dụng  
**Giảng viên:** ThS Đàm Minh Linh  
**Lớp:** D20CQCNP01-N

Người thực hiện  
Trần Việt Anh N20DCCN087  
Châu Huy Diễn N20DCCN010

*TPHCM, ngày 05 tháng 01 năm 2024*

## MỤC LỤC

<b>LỜI CẢM ƠN.....</b>	<b>3</b>
<b>I. TỔNG QUAN.....</b>	<b>4</b>
1. Giới thiệu về dự án. ....	4
2. Giới thiệu về CNN.....	4
3. Ưu và nhược điểm của CNN trong phân loại hình ảnh. ....	9
<b>II. MÔ HÌNH ĐỀ XUẤT.....</b>	<b>10</b>
1. Kiến trúc mô hình CNN.....	10
2. Lựa chọn tham số cho mô hình CNN.....	10
3. PIPELINE .....	11
<b>III. TRIỂN KHAI.....</b>	<b>12</b>
1. Data validation.....	12
2. Data preprocessing.....	12
3. Feature engineering.....	13
<b>IV. TỔNG KẾT.....</b>	<b>15</b>
1. Đánh giá hiệu suất mô hình. ....	15
2. Nhận xét kết quả.....	17
3. Đề xuất hướng cải tiến cho tương lai.....	17
4. Ứng dụng demo.....	18
<b>TÀI LIỆU THAM KHẢO .....</b>	<b>22</b>

## **DANH SÁCH HÌNH ẢNH**

Hình ảnh 1 cấu trúc CNN. ....	5
Hình ảnh 2 Lớp tích chập. ....	6
Hình ảnh 3 Lớp tích chập bổ sung. ....	7
Hình ảnh 4 Lớp kết nối đầy đủ. ....	8
Hình ảnh 5 Cấu trúc của mô hình. ....	10
Hình ảnh 6 PIPELINE. ....	11
Hình ảnh 7 Tập dữ liệu đã được chia. ....	12
Hình ảnh 8 Biến đổi ảnh đầu vào. ....	13
Hình ảnh 9 Ảnh các lớp mô hình CNN. ....	14
Hình ảnh 10 Kết quả training Accuracy 89%. ....	15
Hình ảnh 11 kết quả Pre 90%, recall 89%, f1 89% ....	16

## **DANH SÁCH HÌNH ẢNH DEMO**

Ảnh demo 1 Mô tả hệ thống nhận diện chó và mèo. ....	18
Ảnh demo 2 Giao diện chính. ....	19
Ảnh demo 3 Chuẩn bị nhận diện chó. ....	19
Ảnh demo 4 Kết quả nhận diện chó. ....	20
Ảnh demo 5 Chuẩn bị nhận diện mèo. ....	20
ẢNH DEMO 6 KẾT QUẢ NHẬN DIỆN MÈO. ....	21

## LỜI CẢM ƠN

Trên thực tế không có sự thành công nào mà không gắn liền với những sự hỗ trợ, giúp đỡ dù ít hay nhiều, dù trực tiếp hay gián tiếp của người khác. Trong một khoảng thời gian từ khi bắt đầu học tập ở giảng đường đại học cho đến nay, chúng em luôn nhận được sự quan tâm, hướng dẫn và sự giúp đỡ tận tình của các thầy cô và cũng như sự động viên giúp đỡ của bạn bè. Để thực hiện và hoàn thành bài báo cáo cuối kỳ này, chúng em đã nhận được sự hỗ trợ, giúp đỡ cũng như là quan tâm rất nhiều từ giáo viên hướng dẫn môn học này. Bài tiểu luận cũng được hoàn thành dựa trên sự tham khảo, học tập kinh nghiệm từ các kết quả nghiên cứu liên quan, các sách, báo chuyên ngành của nhiều tác giả ở trường Đại học, các tổ chức nghiên cứu...

Trước hết, chúng em xin gửi lời cảm ơn sâu sắc đến thầy ThS. Đàm Minh Linh – người trực tiếp hướng dẫn, dạy học đã luôn dành nhiều thời gian, công sức hướng dẫn chúng em trong suốt quá trình thực hiện bài báo cáo này cũng như quá trình học tập vừa qua. Trong quá trình nghiên cứu, chúng em đã cố gắng để hoàn thành đề tài báo cáo, bằng việc tham khảo nhiều tài liệu, trao đổi, tiếp thu ý kiến của thầy cô và bạn bè.

Do điều kiện về thời gian và trình độ nghiên cứu của bản thân còn nhiều hạn chế, nên nghiên cứu khó tránh khỏi những thiếu sót. Vì vậy, chúng em rất mong nhận được những ý kiến đóng góp quý báu của thầy để kiến thức của chúng em trong lĩnh vực này được hoàn thiện hơn đồng thời có điều kiện bổ sung, nâng cao ý thức của mình. Xin trân trọng cảm ơn thầy!

## I. TỔNG QUAN.

### 1. Giới thiệu về dự án.

Dự án tập trung vào việc sử dụng Mạng Nơ-ron Tích chập (CNN) để phân loại hình ảnh, một lĩnh vực quan trọng và thú vị trong thị giác máy tính.

CNN là một loại mô hình học sâu được thiết kế đặc biệt để xử lý dữ liệu có cấu trúc lưới, như hình ảnh. CNN có khả năng tự học các đặc trưng từ dữ liệu, điều này giúp cho việc phân loại hình ảnh trở nên hiệu quả hơn so với các phương pháp truyền thống.

Trong dự án này, sẽ xây dựng và huấn luyện một mô hình CNN. Quá trình này bao gồm việc thiết kế kiến trúc của mô hình, lựa chọn các tham số, và sau đó huấn luyện mô hình với tập dữ liệu của kaggle về phân loại chó và mèo.

### 2. Giới thiệu về CNN.

- **CNN là gì**

CNN được viết tắt của Convolutional Neural Network hay còn được gọi là CNNS mang nơ-ron tích chập, là một trong những mô hình Deep Learning cực kỳ tiên tiến, bởi chúng cho phép bạn xây dựng những hệ thống có độ chính xác cao và thông minh. Nhờ khả năng đó, CNN có rất nhiều ứng dụng, đặc biệt là những bài toán cần nhận dạng vật thể (object) trong ảnh. CNN vô cùng quan trọng để tạo nên những hệ thống nhận diện thông minh với độ chính xác cao trong thời đại công nghệ ngày nay. Lý do cụ thể vì sao CNN đặc biệt phát huy hiệu quả trong việc nhận dạng (detection) [1].

- **CNN hoạt động như thế nào**

Mạng nơ-ron tích chập được phân biệt với các mạng thần kinh khác bởi hiệu suất vượt trội của chúng với đầu vào tín hiệu hình ảnh, lời nói hoặc âm thanh.

Chúng có ba loại lớp chính, đó là:

Lớp tích chập

Lớp gộp

Lớp kết nối đầy đủ (FC)

Lớp tích chập là lớp đầu tiên của mạng tích chập. Trong khi các lớp tích chập có thể được theo sau bởi các lớp tích chập bổ sung hoặc các lớp gộp, lớp được kết nối đầy đủ là lớp cuối cùng. Với mỗi lớp, CNN tăng độ phức tạp của nó, xác định các phần lớn hơn của hình ảnh. Các lớp trước đó tập trung vào các tính năng đơn giản, chẳng hạn như màu sắc và cạnh. Khi dữ liệu hình ảnh tiến triển qua các lớp của CNN, nó bắt đầu nhận ra các yếu tố hoặc hình dạng lớn hơn của đối tượng cho đến khi cuối cùng nó xác định được đối tượng dự định.

- **Cấu trúc CNN**

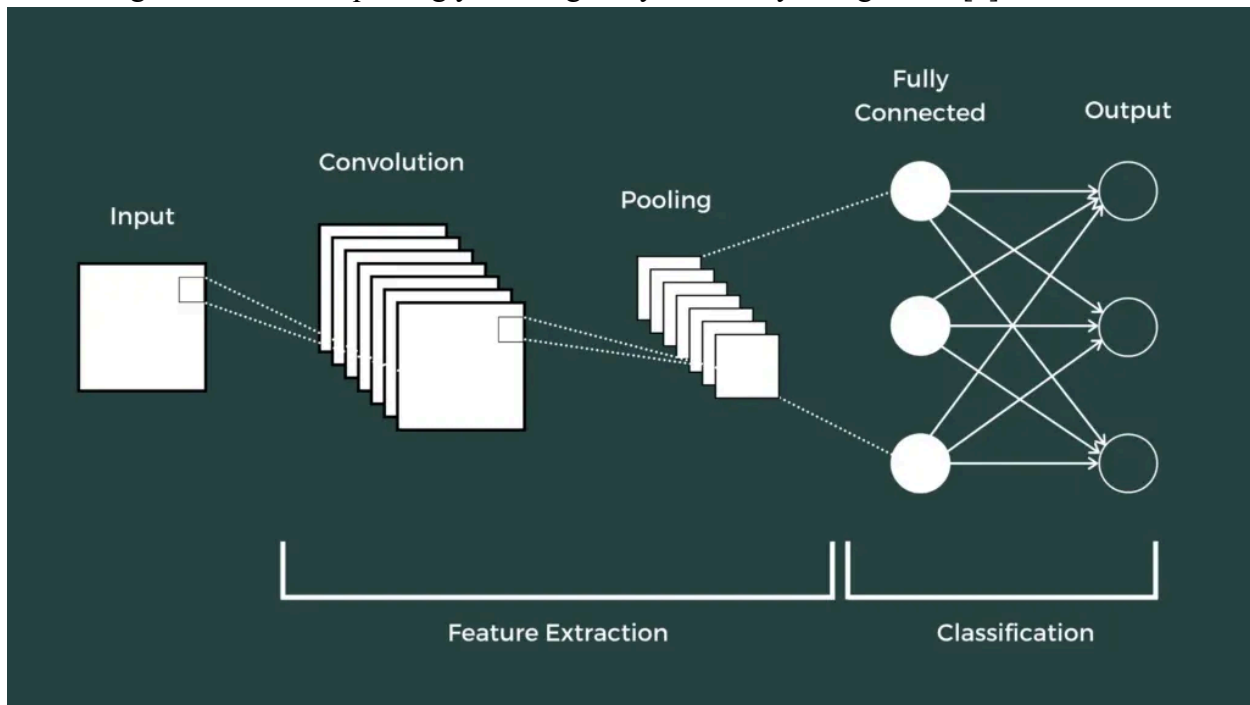
Một cấu trúc cơ bản nhất của CNN sẽ bao gồm 3 phần chủ yếu, đó là:

Local receptive field (trường cục bộ): Nhiệm vụ của trường cục bộ là phân tách và lọc dữ liệu cũng như thông tin ảnh, sau đó chọn ra các vùng ảnh có giá trị sử dụng cao nhất.

Shared weights and bias (trọng số chia sẻ): Trong mạng CNN, thành phần này có tác dụng giảm thiểu tối đa lượng tham số có tác dụng lớn. Trong mỗi convolution sẽ chứa nhiều feature map khác nhau, mỗi feature lại có khả năng giúp nhận diện một số feature trong ảnh.

Pooling layer (lớp tổng hợp): Pooling layer là lớp cuối cùng, với khả năng đơn giản hóa thông tin đầu ra. Khi đã hoàn tất tính toán và quét qua các lớp, pooling layer sẽ được tạo ra

nhằm mục đích lược bớt các thông tin không cần thiết và tối ưu đầu ra. Điều này giúp người dùng nhận được kết quả ưng ý và đúng với yêu cầu hay mong muốn [1].



Hình ảnh 1 cấu trúc CNN.

#### ○ Lớp tích chập

Lớp tích chập là khối xây dựng cốt lõi của CNN và là nơi phần lớn các tính toán xảy ra. Nó yêu cầu một vài thành phần, đó là dữ liệu đầu vào, bộ lọc và bản đồ tính năng. Giả sử rằng đầu vào sẽ là một hình ảnh màu, được tạo thành từ một ma trận các pixel trong 3D. Điều này có nghĩa là đầu vào sẽ có ba chiều — chiều cao, chiều rộng và chiều sâu — tương ứng với RGB trong hình ảnh. Chúng tôi cũng có một trình phát hiện tính năng, còn được gọi là hạt nhân hoặc bộ lọc, sẽ di chuyển qua các trường tiếp nhận của hình ảnh, kiểm tra xem tính năng có hiện diện hay không. Quá trình này được gọi là tích chập [2].

Máy dò tính năng là một mảng trọng lượng hai chiều (2-D), đại diện cho một phần của hình ảnh. Mặc dù chúng có thể khác nhau về kích thước, kích thước bộ lọc thường là ma trận 3x3; Điều này cũng xác định kích thước của trường tiếp nhận. Sau đó, bộ lọc được áp dụng cho một vùng của hình ảnh và tích chập được tính giữa các pixel đầu vào và bộ lọc. Sản phẩm chập này sau đó được đưa vào một mảng đầu ra. Sau đó, bộ lọc dịch chuyển theo một bước, lặp lại quá trình cho đến khi hạt nhân quét qua toàn bộ hình ảnh. Đầu ra cuối cùng từ chuỗi sản phẩm chập từ đầu vào và bộ lọc được gọi là bản đồ tính năng, bản đồ kích hoạt hoặc tính năng phức tạp.

Lưu ý rằng trọng số trong máy dò tính năng vẫn cố định khi nó di chuyển trên hình ảnh, còn được gọi là chia sẻ tham số. Một số thông số, như giá trị trọng lượng, điều chỉnh trong quá trình đào tạo thông qua quá trình lan truyền ngược và giảm độ dốc. Tuy nhiên, có ba siêu

tham số ảnh hưởng đến kích thước âm lượng của đầu ra cần được đặt trước khi bắt đầu đào tạo mạng thần kinh. Chúng bao gồm:

1. Số lượng bộ lọc ảnh hưởng đến độ sâu của đầu ra. Ví dụ: ba bộ lọc riêng biệt sẽ mang lại ba bản đồ địa vật khác nhau, tạo ra độ sâu là ba.
2. Stride là khoảng cách, hoặc số pixel, mà hạt nhân di chuyển qua ma trận đầu vào. Mặc dù giá trị sai phân từ hai trở lên là rất hiếm, nhưng sai phân lớn hơn mang lại đầu ra nhỏ hơn.
3. Zero-padding thường được sử dụng khi các bộ lọc không phù hợp với hình ảnh đầu vào. Điều này đặt tất cả các phần tử nằm ngoài ma trận đầu vào về không, tạo ra đầu ra lớn hơn hoặc có kích thước bằng nhau. Có ba loại đệm:

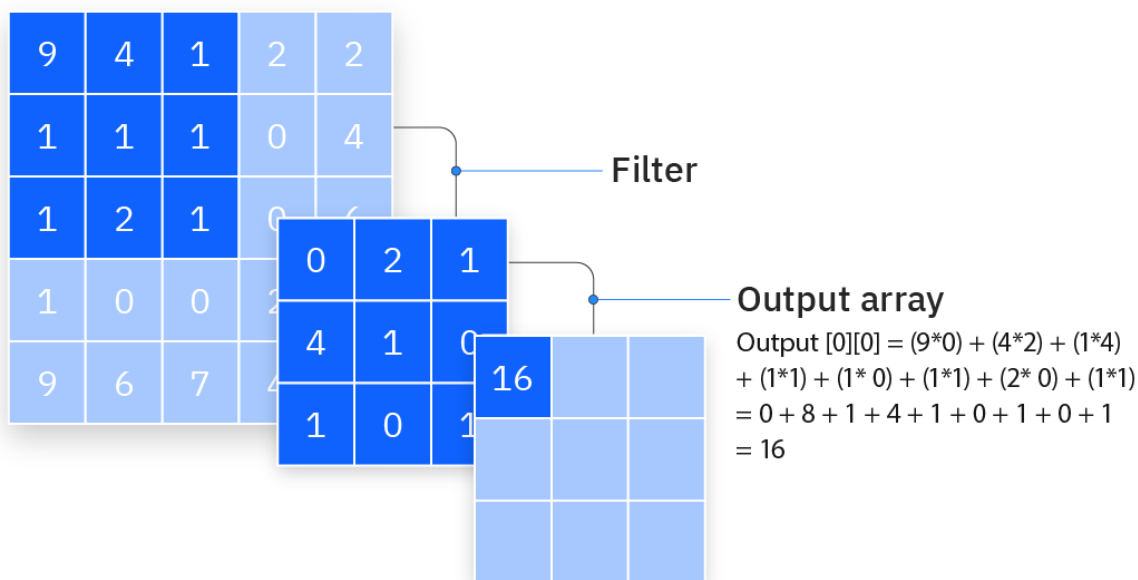
Đệm hợp lệ: Điều này còn được gọi là không có đệm. Trong trường hợp này, tích chập cuối cùng bị loại bỏ nếu kích thước không thẳng hàng.

Cùng một đệm: Phần đệm này đảm bảo rằng lớp đầu ra có cùng kích thước với lớp đầu vào

Đệm đầy đủ: Loại đệm này làm tăng kích thước của đầu ra bằng cách thêm số không vào đường viền của đầu vào.

Sau mỗi thao tác tích chập, CNN áp dụng phép chuyển đổi Đơn vị tuyến tính chỉnh lưu (ReLU) cho bản đồ tính năng, giới thiệu tính phi tuyến cho mô hình [2].

### Input image



Hình ảnh 2 Lớp tích chập.

#### ○ Lớp tích chập bổ sung

Như chúng tôi đã đề cập trước đó, một lớp tích chập khác có thể theo sau lớp tích chập ban đầu. Khi điều này xảy ra, cấu trúc của CNN có thể trở nên phân cấp vì các lớp sau có thể thấy các pixel trong các trường tiếp nhận của các lớp trước. Ví dụ: giả sử rằng chúng ta đang cố gắng xác định xem hình ảnh có chứa xe đạp hay không. Bạn có thể nghĩ về chiếc xe

đạp như một tổng hợp của các bộ phận. Nó bao gồm một khung, tay lái, bánh xe, bàn đạp, et cetera. Mỗi bộ phận riêng lẻ của xe đạp tạo thành một mô hình cấp thấp hơn trong mạng lưới thần kinh và sự kết hợp của các bộ phận của nó đại diện cho một mô hình cấp cao hơn, tạo ra một hệ thống phân cấp tính năng trong CNN. Cuối cùng, lớp tích chập chuyển đổi hình ảnh thành các giá trị số, cho phép mạng thần kinh diễn giải và trích xuất các mẫu có liên quan [2].



*Hình ảnh 3 Lớp tích chập bổ sung.*

- **Lớp Relu**

Còn có tên gọi khác là activation function, đây là một hàm được kích hoạt trong neural network. Nó có tác dụng mô phỏng các neuron có tỷ lệ truyền xung qua axon. Trong activation function chúng còn có hàm nghĩa là: Relu, Tanh, Sigmoid, Maxout, Leaky,... Relu layer được ứng dụng phổ biến trong việc huấn luyện nơ-ron do sở hữu nhiều ưu điểm tiên tiến.

- **Lớp gộp**

Các lớp gộp, còn được gọi là downsampling, tiến hành giảm kích thước, giảm số lượng tham số trong đầu vào. Tương tự như lớp tích chập, hoạt động gộp lại quét một bộ lọc trên toàn bộ đầu vào, nhưng sự khác biệt là bộ lọc này không có bất kỳ trọng số nào. Thay vào đó, kernel áp dụng một hàm tổng hợp cho các giá trị trong trường tiếp nhận, điền vào mảng đầu ra. Có hai loại gộp chính:

**Tổng hợp tối đa:** Khi bộ lọc di chuyển qua đầu vào, nó chọn pixel có giá trị tối đa để gửi đến mảng đầu ra. Ngoài ra, phương pháp này có xu hướng được sử dụng thường xuyên hơn so với gộp trung bình.

**Tổng hợp trung bình:** Khi bộ lọc di chuyển qua đầu vào, nó sẽ tính toán giá trị trung bình trong trường tiếp nhận để gửi đến mảng đầu ra.

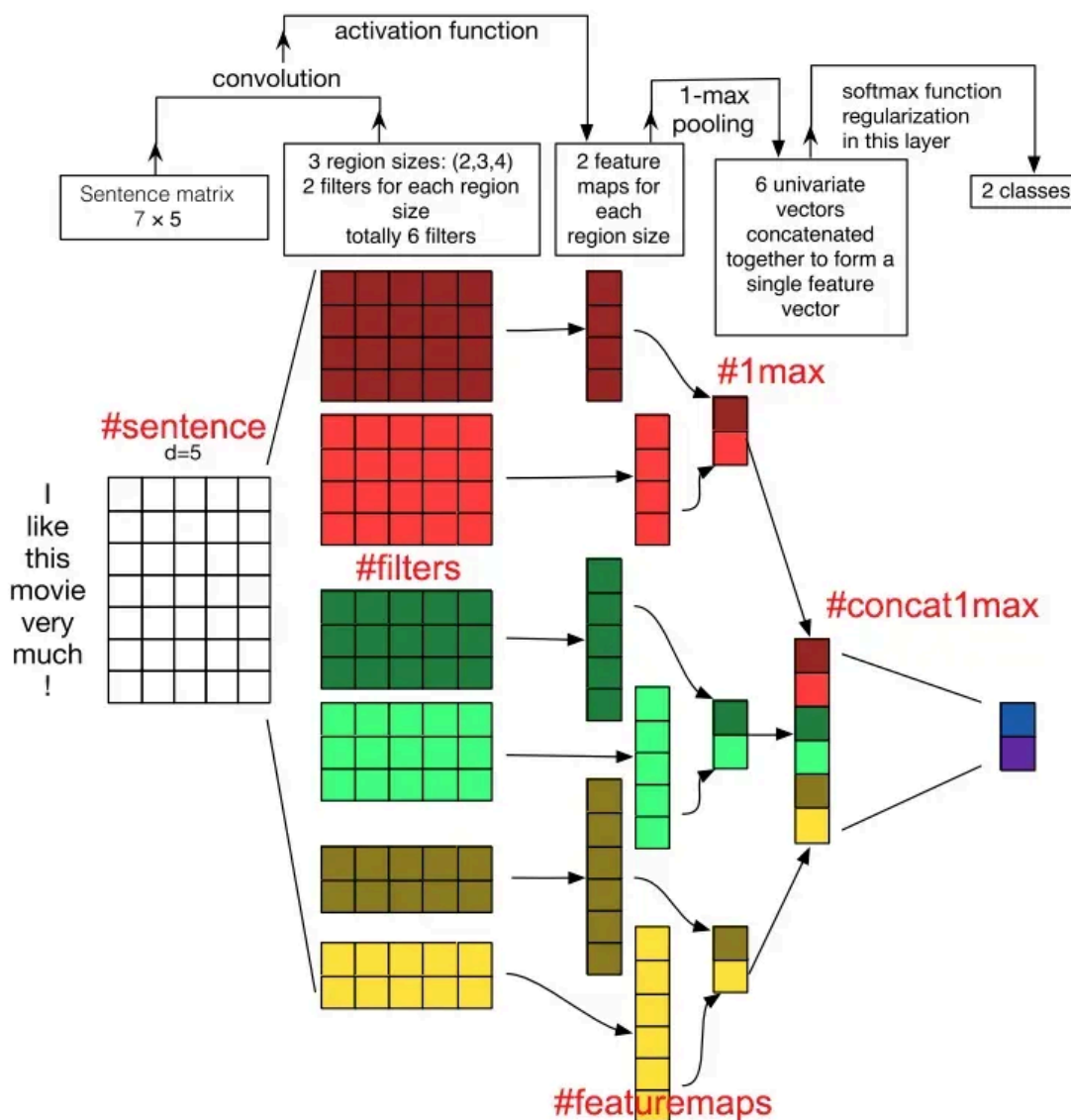


Trong khi rất nhiều thông tin bị mất trong lớp tổng hợp, nó cũng có một số lợi ích cho CNN. Chúng giúp giảm độ phức tạp, nâng cao hiệu quả và hạn chế nguy cơ quá tải [2].

#### ○ **Lớp kết nối đầy đủ**

Tên của lớp kết nối đầy đủ mô tả chính nó một cách khéo léo. Như đã đề cập trước đó, các giá trị pixel của hình ảnh đầu vào không được kết nối trực tiếp với lớp đầu ra trong các lớp được kết nối một phần. Tuy nhiên, trong lớp được kết nối đầy đủ, mỗi nút trong lớp đầu ra kết nối trực tiếp với một nút trong lớp trước.

Lớp này thực hiện nhiệm vụ phân loại dựa trên các tính năng được trích xuất thông qua các lớp trước và các bộ lọc khác nhau của chúng. Trong khi các lớp tích chập và gộp có xu hướng sử dụng các hàm ReLu, các lớp FC thường tận dụng chức năng kích hoạt softmax để phân loại đầu vào một cách thích hợp, tạo ra xác suất từ 0 đến 1 [2].



Hình ảnh 4 Lớp kết nối đầy đủ.

### **3. Ưu và nhược điểm của CNN trong phân loại hình ảnh.**

#### **Ưu điểm:**

- Mạng nơ-ron tích chập là khả năng chia sẻ thông tin giữa các lớp, trong khi nhược điểm là không có khả năng xác định số lần lặp tối ưu để tính toán chính xác [3].
- Độ chính xác cao hơn trong nhận dạng hình ảnh [4].
- Độ chính xác cao trong các tác vụ học máy khác nhau [5].
- Mạng nơ-ron tích chập đã cho thấy kết quả đáng chú ý trong các ứng dụng liên quan đến hình ảnh khác nhau [6].

#### **Nhược điểm:**

- Yêu cầu tính toán cao hơn và có thể dẫn đến mất thông tin [4].
- Kích thước mạng lớn và phức tạp [5].
- Thiết kế CNN đòi hỏi kiến thức đa miền và tốn nhiều công sức và thời gian [6].

## II. MÔ HÌNH ĐỀ XUẤT.

### 1. Kiến trúc mô hình CNN.

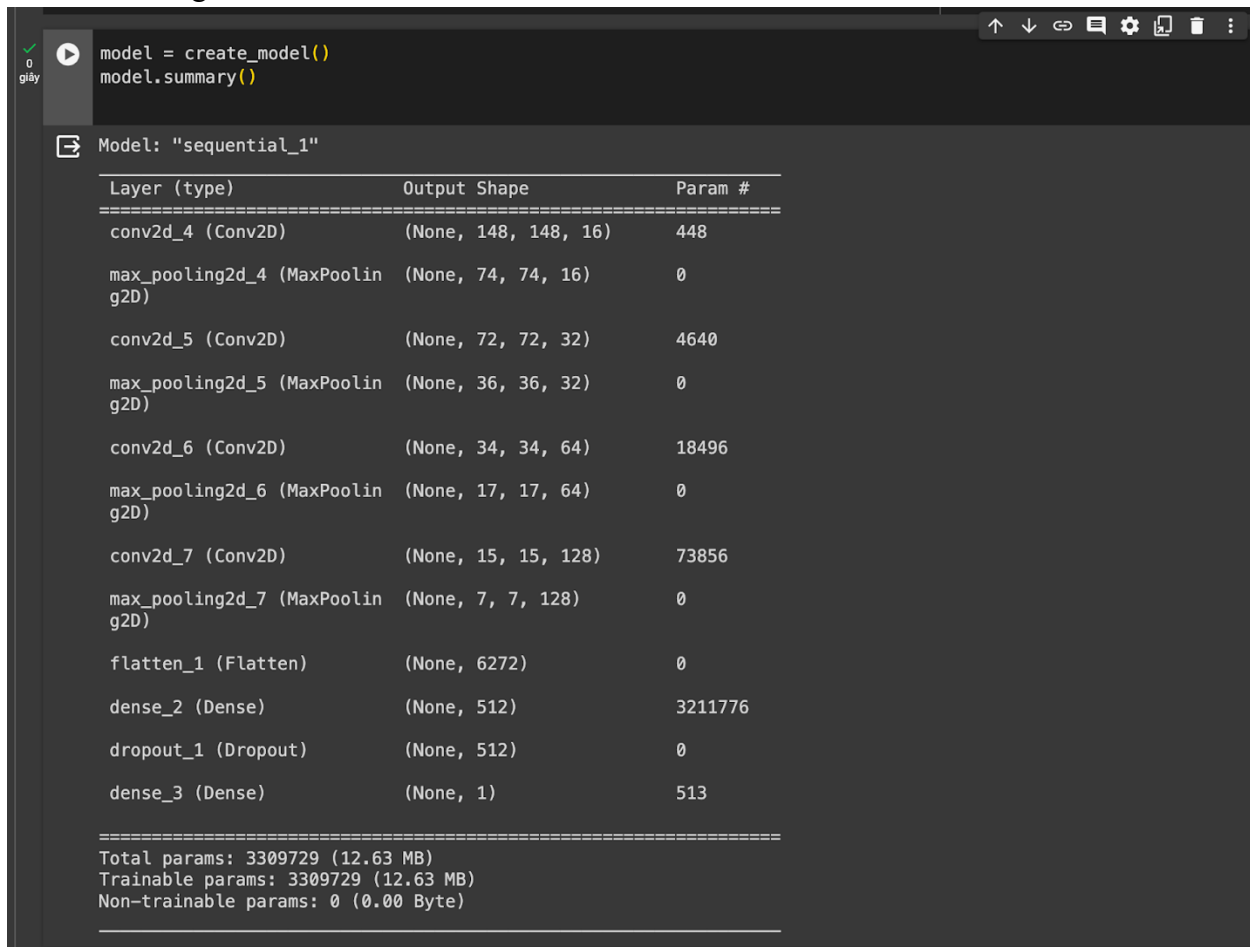
Bốn lớp Conv2D: Đây là các lớp tích chập, mỗi lớp sử dụng một bộ lọc kích thước (3,3) để quét qua hình ảnh đầu vào và thực hiện phép toán tích chập. Số lượng bộ lọc trong mỗi lớp lần lượt là 16, 32 và 64. Hàm kích hoạt 'relu' được sử dụng trong các lớp này.

Bốn lớp MaxPool2D: Đây là các lớp pooling, mỗi lớp giảm kích thước không gian của hình ảnh đầu vào bằng cách lấy giá trị lớn nhất trong mỗi bộ lọc kích thước (2,2).

Một lớp Dropout: Lớp này chống lại overfitting.

Một lớp Flatten: Lớp này chuyển đổi tensor đầu vào thành một vector 1D, chuẩn bị cho việc kết nối với các lớp fully connected.

Hai lớp Dense: Đây là các lớp fully connected. Lớp đầu tiên có 512 nút và sử dụng hàm kích hoạt 'relu'. Lớp thứ hai có 1 nút và sử dụng hàm kích hoạt 'sigmoid' để đưa ra dự đoán cuối cùng của mô hình.



The screenshot shows a Jupyter Notebook interface with a code cell containing the following code:

```
model = create_model()
model.summary()
```

Below the code cell, the model summary for 'sequential\_1' is displayed as a table:

Layer (type)	Output Shape	Param #
conv2d_4 (Conv2D)	(None, 148, 148, 16)	448
max_pooling2d_4 (MaxPooling2D)	(None, 74, 74, 16)	0
conv2d_5 (Conv2D)	(None, 72, 72, 32)	4640
max_pooling2d_5 (MaxPooling2D)	(None, 36, 36, 32)	0
conv2d_6 (Conv2D)	(None, 34, 34, 64)	18496
max_pooling2d_6 (MaxPooling2D)	(None, 17, 17, 64)	0
conv2d_7 (Conv2D)	(None, 15, 15, 128)	73856
max_pooling2d_7 (MaxPooling2D)	(None, 7, 7, 128)	0
flatten_1 (Flatten)	(None, 6272)	0
dense_2 (Dense)	(None, 512)	3211776
dropout_1 (Dropout)	(None, 512)	0
dense_3 (Dense)	(None, 1)	513

Below the table, the total parameters and trainable parameters are listed:

```
=====  
Total params: 3309729 (12.63 MB)  
Trainable params: 3309729 (12.63 MB)  
Non-trainable params: 0 (0.00 Byte)
```

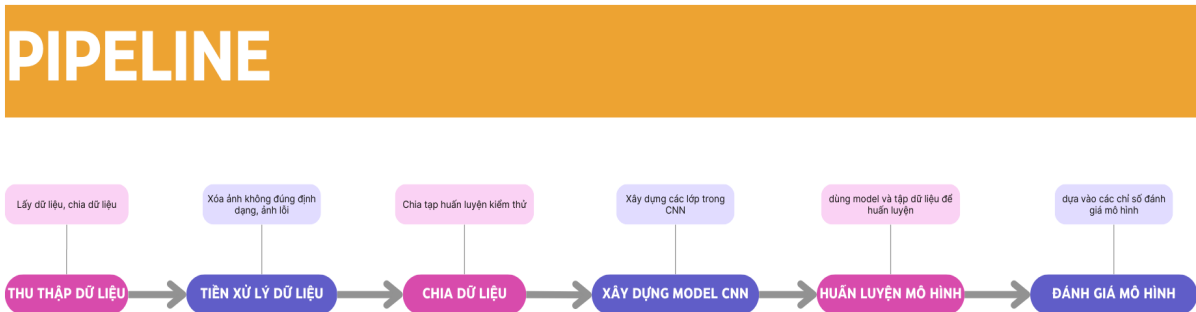
Hình ảnh 5 Cấu trúc của mô hình.

### 2. Lựa chọn tham số cho mô hình CNN.

Mô hình của bạn là một mạng nơ-ron tích chập (Convolutional Neural Network - CNN) được xây dựng bằng TensorFlow và Keras. Dưới đây là mô tả chi tiết về các lớp trong mô hình:

- Conv2D(16, (3,3), activation='relu', input\_shape=(150, 150, 3)): Lớp tích chập đầu tiên với 16 bộ lọc, kích thước bộ lọc là 3x3, hàm kích hoạt là ReLU, và kích thước đầu vào là 150x150x3.
- MaxPooling2D(2,2): Lớp gộp tối đa với kích thước cửa sổ gộp là 2x2.
- Conv2D(32, (3,3), activation='relu'): Lớp tích chập thứ hai với 32 bộ lọc và kích thước bộ lọc là 3x3.
- MaxPooling2D(2,2): Lớp gộp tối đa thứ hai.
- Conv2D(64, (3,3), activation='relu'): Lớp tích chập thứ ba với 64 bộ lọc.
- MaxPooling2D(2,2): Lớp gộp tối đa thứ ba.
- Conv2D(128, (3,3), activation='relu'): Lớp tích chập thứ tư với 128 bộ lọc.
- MaxPooling2D(2,2): Lớp gộp tối đa thứ tư.
- Flatten(): Lớp làm phẳng để chuyển đổi tensor 3D thành vector 1D.
- Dense(512, activation='relu'): Lớp kết nối đầy đủ với 512 nơ-ron.
- Dropout(0.5): Lớp Dropout với tỷ lệ dropout là 0.5 để giảm overfitting.
- Dense(1, activation='sigmoid'): Lớp kết nối đầy đủ cuối cùng với 1 nơ-ron và hàm kích hoạt là sigmoid.

### 3. PIPELINE



Hình ảnh 6 PIPELINE.

### III. TRIỂN KHAI.

#### 1. Data validation.

Dữ liệu được lấy từ Kaggle:

Với 25000 mẫu ảnh chó và mèo

Trong hàm `split_data`, có một phần của mã được sử dụng để tạo các tập dữ liệu huấn luyện, validation và kiểm thử từ các ảnh nguồn. Phần này chịu trách nhiệm chia dữ liệu thành các tập theo tỷ lệ đã được xác định (`split_size_train`, `split_size_val`, `split_size_test`). Dữ liệu được chia ngẫu nhiên và sau đó được sao chép vào các thư mục tương ứng.

```
split_data(CAT_SOURCE_DIR, TRAINING_CATS_DIR,  
VALIDATION_CATS_DIR, TEST_CATS_DIR, 0.8, 0.1, 0.1)
```

```
split_data(DOG_SOURCE_DIR, TRAINING_DOGS_DIR, VALIDATION_DOGS_DI  
R, TEST_DOGS_DIR, 0.8, 0.1, 0.1)
```

Trong đoạn mã này, các thư mục huấn luyện (`TRAINING_CATS_DIR`, `TRAINING_DOGS_DIR`), thư mục validation (`VALIDATION_CATS_DIR`, `VALIDATION_DOGS_DIR`), và thư mục kiểm thử (`TEST_CATS_DIR`, `TEST_DOGS_DIR`) được tạo và sau đó dữ liệu được chia và sao chép vào các thư mục này.

Sau khi chia thành 3 mục để train, test, validation:

```
❏ Không tìm thấy tệp: /content/PetImages/Dog/Thumbs.db  
Không tìm thấy tệp: /content/PetImages/Cat/Thumbs.db  
Không tìm thấy tệp: /content/PetImages/Cat/666.jpg  
Không tìm thấy tệp: /content/PetImages/Dog/11702.jpg  
  
Thư mục gốc của mèo có 12499 ảnh  
Thư mục gốc của chó có 12499 ảnh  
  
Số lượng ảnh mèo cho tập huấn luyện: 9999  
Số lượng ảnh chó cho tập huấn luyện: 9999  
Số lượng ảnh mèo cho tập validation: 1249  
Số lượng ảnh chó cho tập validation: 1249  
Số lượng ảnh mèo cho tập test: 1251  
Số lượng ảnh chó cho tập test: 1251
```

Hình ảnh 7 Tập dữ liệu đã được chia

#### 2. Data preprocessing.

Phần này đề cập đến việc làm sạch dữ liệu và chuẩn bị dữ liệu cho quá trình huấn luyện mô hình. Cụ thể, trong các hàm `remove_files` và `empty_directory`, có các bước sau:

##### 1. `remove_files`:

Xóa các tệp tin không mong muốn từ thư mục nguồn. Các đường dẫn cần được loại bỏ được xác định bởi `paths_to_remove`.

##### 2. `empty_directory`:

Tạo hoặc làm trống các thư mục để sẵn sàng nhận dữ liệu mới.

Các thư mục cụ thể được tạo hoặc làm trống là:

TRAINING\_CATS\_DIR, TRAINING\_DOGS\_DIR, VALIDATION\_CATS\_DIR, VALIDATION\_DOGS\_DIR, TEST\_CATS\_DIR, và TEST\_DOGS\_DIR.

### 3. Feature engineering.

Data Augmentation: Áp dụng thêm các biến đổi như xoay ảnh, lật ngang, zoom, để tăng kích thước của tập dữ liệu huấn luyện.

Những biến đổi này giúp mô hình học được từ nhiều góc độ, kích thước và tư thế khác nhau của ảnh, tăng khả năng tổng quát hóa và giảm nguy cơ overfitting.

```
train_datagen = ImageDataGenerator(  
    rescale=1/255,  
    rotation_range=40,  
    width_shift_range=0.2,  
    height_shift_range=0.2,  
    shear_range=0.2,  
    zoom_range=0.2,  
    horizontal_flip=True,  
    fill_mode='nearest'  
)
```

Hình ảnh 8 Biến đổi ảnh đầu vào

#### Conv2D Layers:

tf.keras.layers.Conv2D(16, (3,3), activation='relu', input\_shape=(150, 150, 3)): Lớp Conv2D đầu tiên với 16 bộ lọc kích thước (3,3), hàm kích hoạt là ReLU, và đầu vào có kích thước (150, 150, 3). Các bộ lọc này sẽ quét qua ảnh đầu vào để tìm ra các đặc trưng quan trọng.

#### MaxPooling2D Layers:

tf.keras.layers.MaxPooling2D(2,2): Lớp MaxPooling2D sau mỗi lớp Conv2D giảm kích thước của đầu ra và giữ lại thông tin quan trọng nhất. Kích thước cửa sổ là (2,2), nghĩa là sẽ giữ lại giá trị lớn nhất từ mỗi cửa sổ (2x2) của đầu vào.

#### Flatten Layer:

tf.keras.layers.Flatten(): Lớp này chuyển đổi tensor 2D thành vector 1D, tức là làm phẳng dữ liệu để chuẩn bị cho các lớp Dense.

#### Dense Layers:

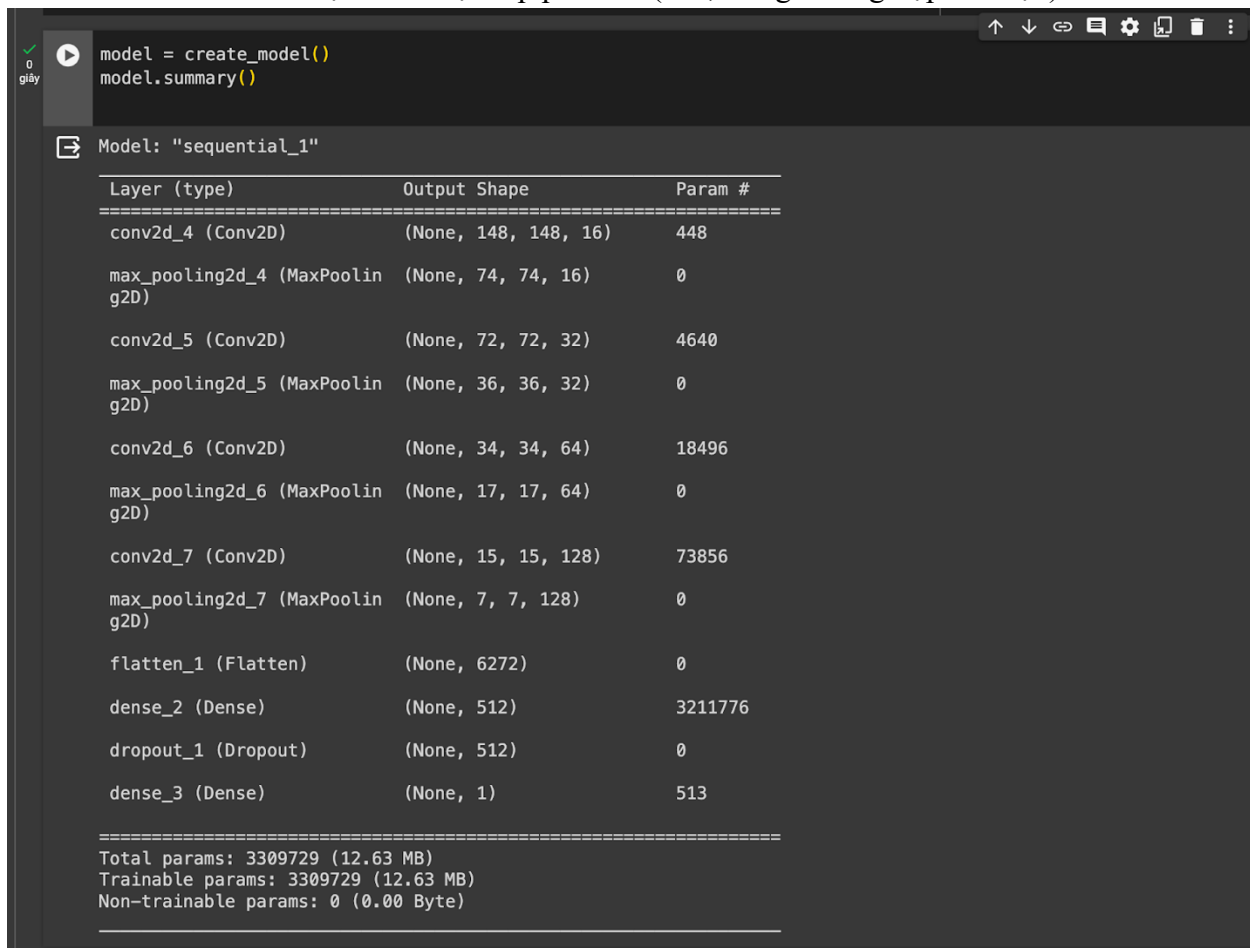
tf.keras.layers.Dense(512, activation='relu'): Lớp Dense với 512 nơ-ron và hàm kích hoạt là ReLU. Lớp này thực hiện kết nối đầy đủ giữa các nơ-ron.

Dropout Layer:

`tf.keras.layers.Dropout(0.5)`: Lớp Dropout giảm overfitting bằng cách tắt ngẫu nhiên 50% các nơ-ron trong quá trình huấn luyện. Điều này giúp mô hình trở nên chung chung hơn và có khả năng tổng quát hóa tốt hơn cho dữ liệu mới.

### Output Layer:

`tf.keras.layers.Dense(1, activation='sigmoid')`: Lớp đầu ra với 1 nơ-ron và hàm kích hoạt là sigmoid. Đây là một vấn đề phân loại nhị phân, nơ-ron này sẽ đưa ra đầu ra 0 hoặc 1, phản ánh xác suất của một ảnh thuộc lớp positive (chó, trong trường hợp của bạn).



The screenshot shows a Jupyter Notebook interface with a code cell containing the following code:

```
model = create_model()
model.summary()
```

Below the code cell, the model summary for "sequential\_1" is displayed as a table:

Layer (type)	Output Shape	Param #
conv2d_4 (Conv2D)	(None, 148, 148, 16)	448
max_pooling2d_4 (MaxPooling2D)	(None, 74, 74, 16)	0
conv2d_5 (Conv2D)	(None, 72, 72, 32)	4640
max_pooling2d_5 (MaxPooling2D)	(None, 36, 36, 32)	0
conv2d_6 (Conv2D)	(None, 34, 34, 64)	18496
max_pooling2d_6 (MaxPooling2D)	(None, 17, 17, 64)	0
conv2d_7 (Conv2D)	(None, 15, 15, 128)	73856
max_pooling2d_7 (MaxPooling2D)	(None, 7, 7, 128)	0
flatten_1 (Flatten)	(None, 6272)	0
dense_2 (Dense)	(None, 512)	3211776
dropout_1 (Dropout)	(None, 512)	0
dense_3 (Dense)	(None, 1)	513

Below the table, the following summary statistics are provided:

```
=====  
Total params: 3309729 (12.63 MB)  
Trainable params: 3309729 (12.63 MB)  
Non-trainable params: 0 (0.00 Byte)
```

Hình ảnh 9 Ảnh các lớp mô hình CNN.

## IV. TỔNG KẾT.

### 1. Đánh giá hiệu suất mô hình.

Chính xác (Accuracy): Mô hình có độ chính xác là khoảng 89%, tức là mô hình dự đoán đúng khoảng 89% trên tổng số các mẫu kiểm tra.

Precision: Đối với lớp 0 (chó), độ chính xác là 92%, tức là trong các trường hợp mà mô hình dự đoán là chó, thì khoảng 92% là đúng. Đối với lớp 1 (mèo), độ chính xác là 87%, có nghĩa là trong các trường hợp mà mô hình dự đoán là mèo, thì khoảng 87% là đúng.

Recall (Nhảy): Đối với lớp 0 (chó), độ nhảy là 86%, tức là mô hình có khả năng bắt được khoảng 86% số lượng chó thực sự trong tập kiểm tra. Đối với lớp 1 (mèo), độ nhảy là 93%, có nghĩa là mô hình có khả năng bắt được khoảng 93% số lượng mèo thực sự trong tập kiểm tra.

F1-score: F1-score là một trung bình độ chính xác và độ nhảy. Đối với lớp 0, F1-score là 0.89, và đối với lớp 1, F1-score là 0.90.

Macro Avg và Weighted Avg: Trung bình cộng của precision, recall và F1-score cho cả hai lớp. Macro avg tính trung bình không cân nặng giữa các lớp, trong khi weighted avg tính trung bình dựa trên sự xuất hiện thực sự của mỗi lớp.

```
Epoch 1/15  
436/1000 [=====] - ETA: 1:26 - loss: 0.6910 - accuracy: 0.5312/usr/local/lib/python3.10/dist-packages/PIL/TiffImagePlugin.py:858: UserWarning: Truncated File Read  
warnings.warn(str(msg))  
1000/1000 [=====] - 161s 154ms/step - loss: 0.6791 - accuracy: 0.5650 - val_loss: 0.6344 - val_accuracy: 0.6477  
Epoch 2/15  
1000/1000 [=====] - 152s 152ms/step - loss: 0.6452 - accuracy: 0.6243 - val_loss: 0.5849 - val_accuracy: 0.6833  
Epoch 3/15  
1000/1000 [=====] - 151s 151ms/step - loss: 0.6015 - accuracy: 0.6721 - val_loss: 0.5267 - val_accuracy: 0.7414  
Epoch 4/15  
1000/1000 [=====] - 151s 151ms/step - loss: 0.5616 - accuracy: 0.7108 - val_loss: 0.5254 - val_accuracy: 0.7322  
Epoch 5/15  
1000/1000 [=====] - 151s 151ms/step - loss: 0.5356 - accuracy: 0.7302 - val_loss: 0.4736 - val_accuracy: 0.7706  
Epoch 6/15  
1000/1000 [=====] - 152s 152ms/step - loss: 0.5130 - accuracy: 0.7500 - val_loss: 0.4335 - val_accuracy: 0.7918  
Epoch 7/15  
1000/1000 [=====] - 149s 149ms/step - loss: 0.4845 - accuracy: 0.7692 - val_loss: 0.4048 - val_accuracy: 0.8078  
Epoch 8/15  
1000/1000 [=====] - 147s 147ms/step - loss: 0.4584 - accuracy: 0.7816 - val_loss: 0.3869 - val_accuracy: 0.8223  
Epoch 9/15  
1000/1000 [=====] - 150s 150ms/step - loss: 0.4339 - accuracy: 0.7973 - val_loss: 0.3961 - val_accuracy: 0.8147  
Epoch 10/15  
1000/1000 [=====] - 150s 150ms/step - loss: 0.4119 - accuracy: 0.8103 - val_loss: 0.4107 - val_accuracy: 0.8143  
Epoch 11/15  
1000/1000 [=====] - 148s 148ms/step - loss: 0.3899 - accuracy: 0.8233 - val_loss: 0.3071 - val_accuracy: 0.8663  
Epoch 12/15  
1000/1000 [=====] - 148s 148ms/step - loss: 0.3806 - accuracy: 0.8293 - val_loss: 0.2962 - val_accuracy: 0.8699  
Epoch 13/15  
1000/1000 [=====] - 152s 152ms/step - loss: 0.3695 - accuracy: 0.8353 - val_loss: 0.3090 - val_accuracy: 0.8683  
Epoch 14/15  
1000/1000 [=====] - 152s 152ms/step - loss: 0.3555 - accuracy: 0.8425 - val_loss: 0.2715 - val_accuracy: 0.8875  
Epoch 15/15  
1000/1000 [=====] - 149s 149ms/step - loss: 0.3478 - accuracy: 0.8472 - val_loss: 0.2658 - val_accuracy: 0.8783  
126/126 [=====] - 4s 35ms/step - loss: 0.2438 - accuracy: 0.8949  
[*] Result:  
loss: 0.2438  
accuracy: 0.8949
```

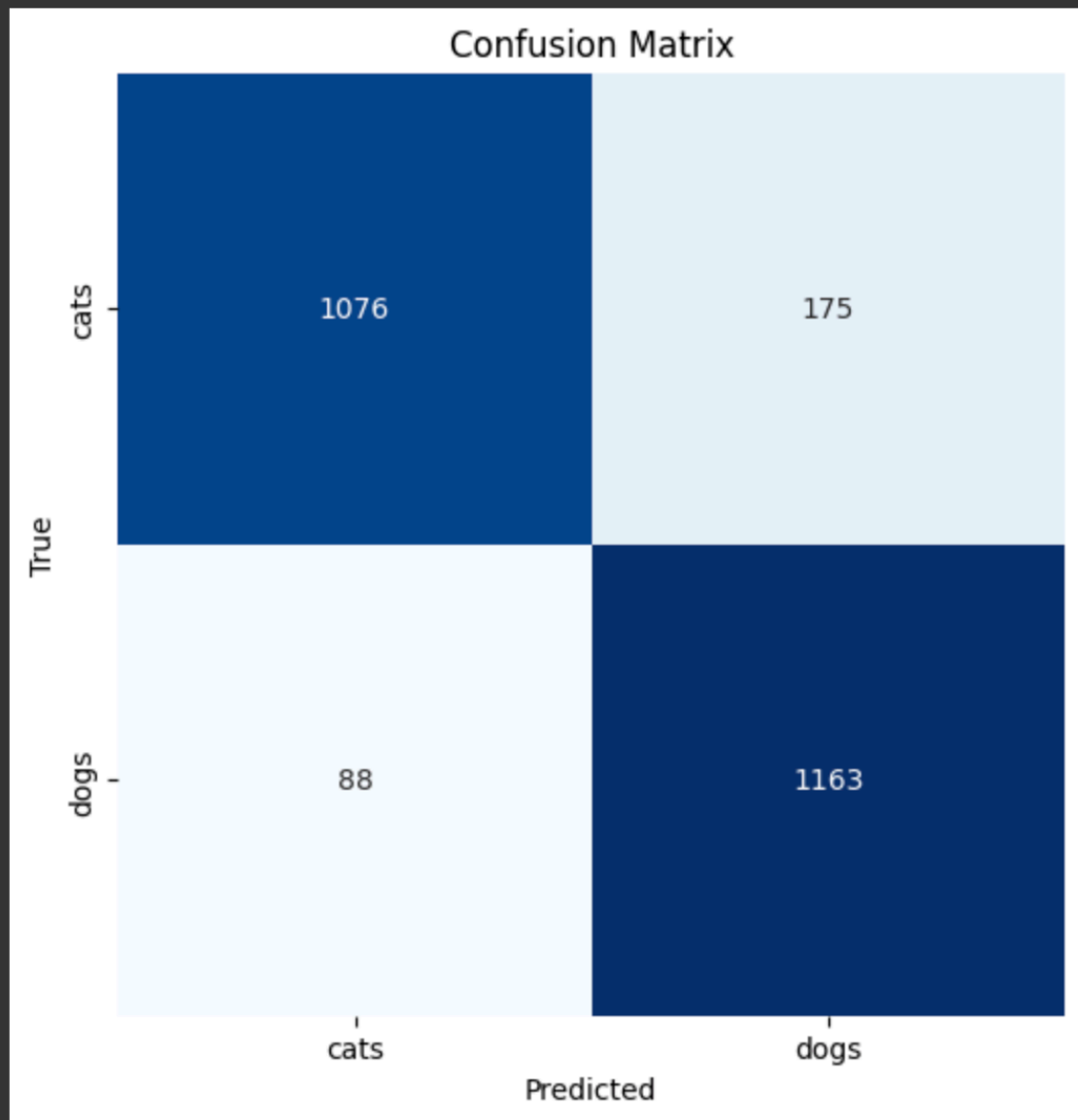
Hình ảnh 10 Kết quả training Accuracy 89%



126/126 [=====] - 13s 102ms/step

Classification Report:

	precision	recall	f1-score	support
0	0.92	0.86	0.89	1251
1	0.87	0.93	0.90	1251
accuracy			0.89	2502
macro avg	0.90	0.89	0.89	2502
weighted avg	0.90	0.89	0.89	2502



Hình ảnh 11 kết quả Pre 90%, recall 89%, f1 89%

## **2. Nhận xét kết quả.**

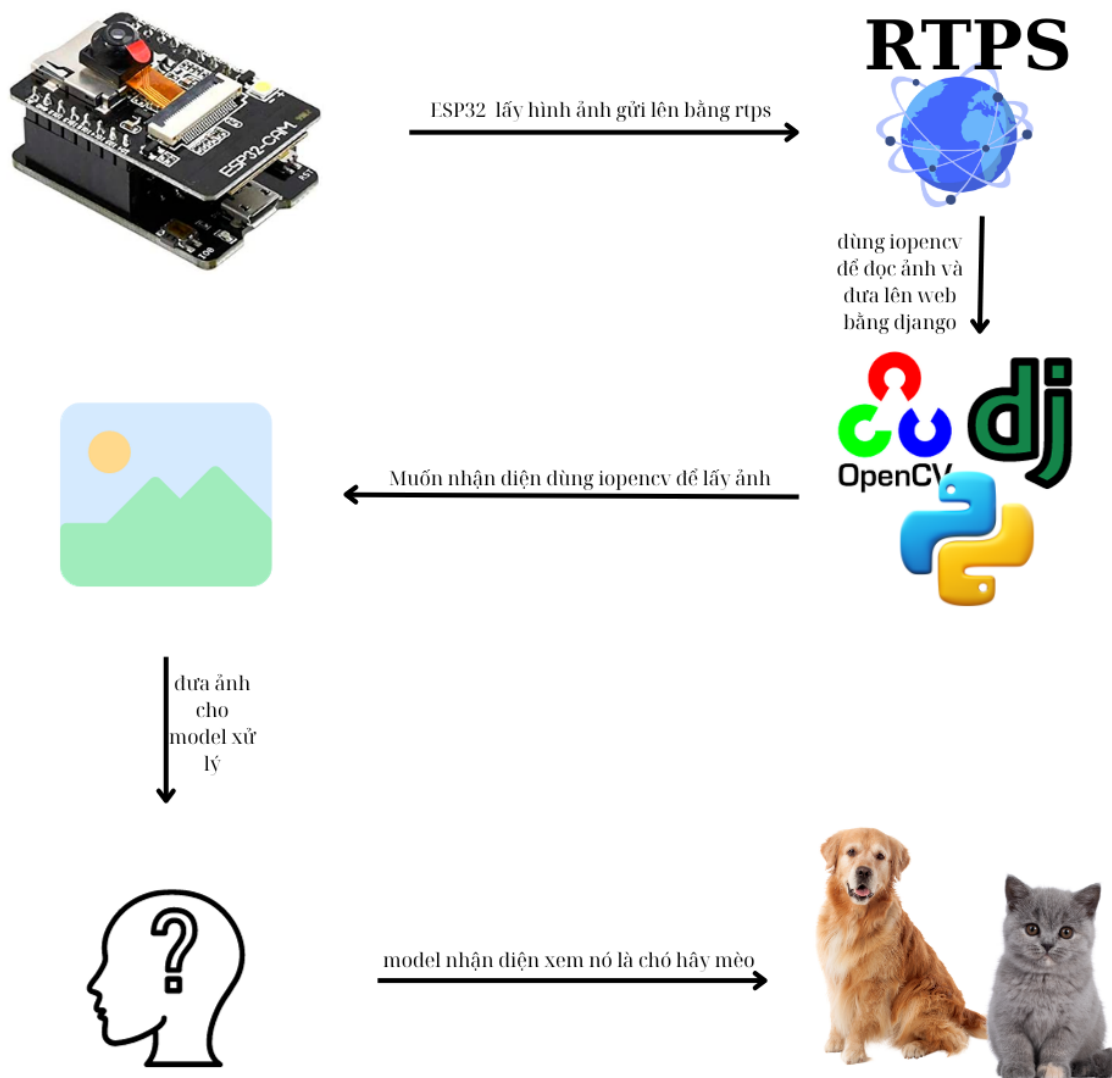
Mô hình có hiệu suất tốt với độ chính xác khá cao và cân bằng giữa precision và recall đối với cả hai lớp.

## **3. Đề xuất hướng cải tiến cho tương lai.**

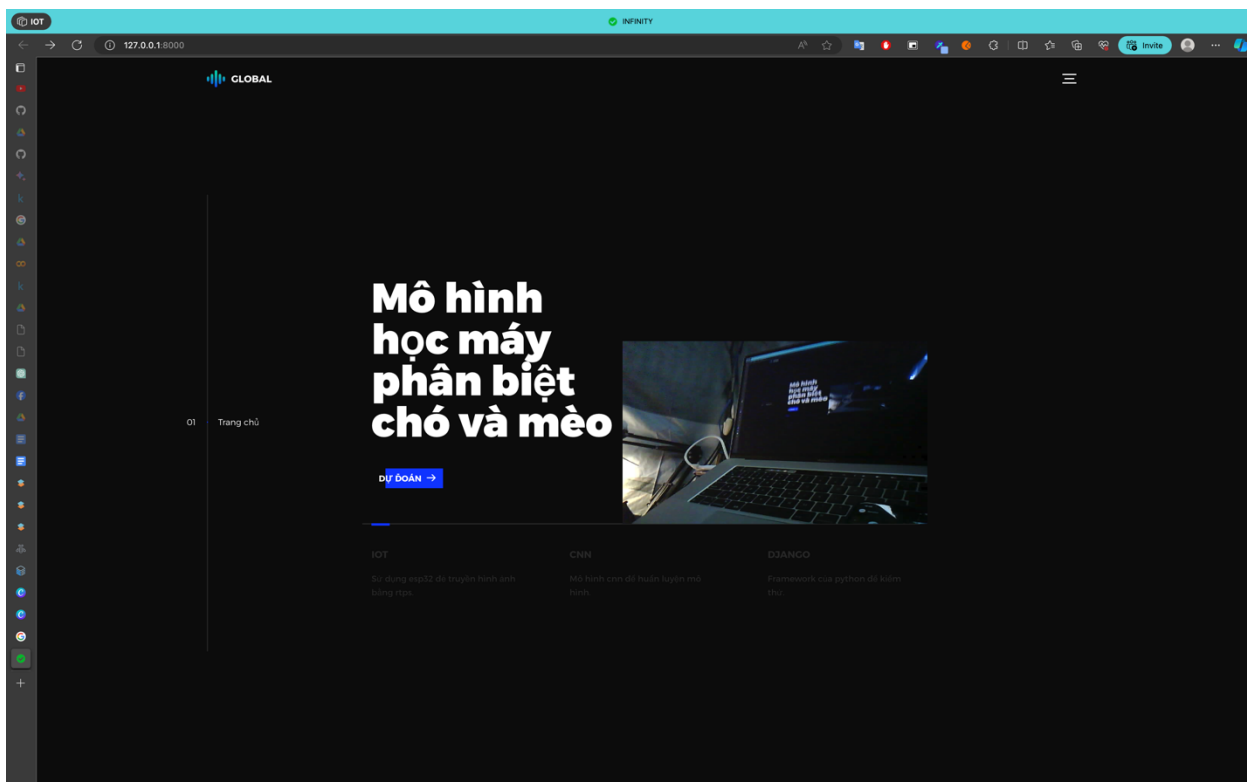
**Thu thập thêm dữ liệu:** Nếu có khả năng, việc thu thập thêm dữ liệu có thể cải thiện khả năng tổng quát hóa của mô hình. Điều này có thể giúp mô hình hiểu biết động vật và tình huống đa dạng hơn.

**Optimize và triển khai mô hình:** Nếu mô hình được đánh giá là đủ tốt, có thể tối ưu hóa và triển khai nó trên sản phẩm hoặc dịch vụ thực tế. Điều này có thể bao gồm việc chuyển đổi mô hình sang định dạng tối ưu hóa, tối ưu hóa để chạy trên các thiết bị có tài nguyên hạn chế, và quản lý hiệu suất trong môi trường triển khai.

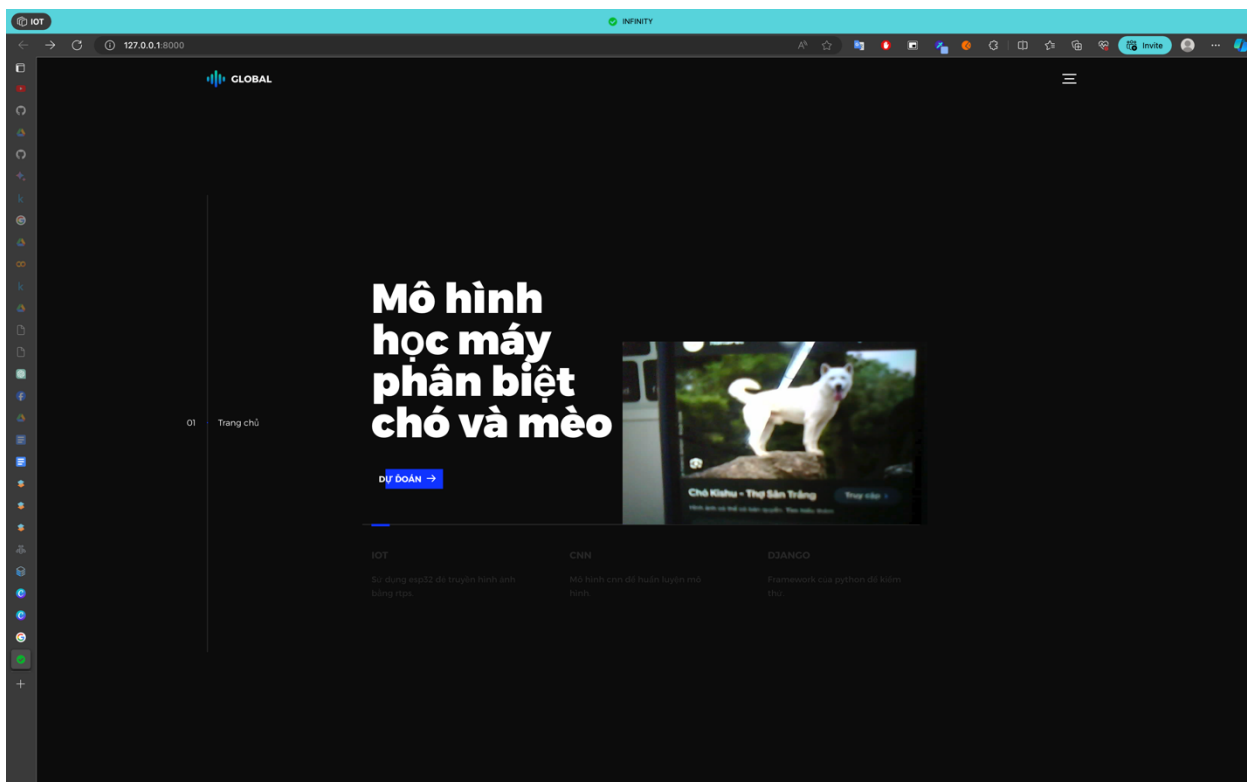
#### 4. Ứng dụng demo.



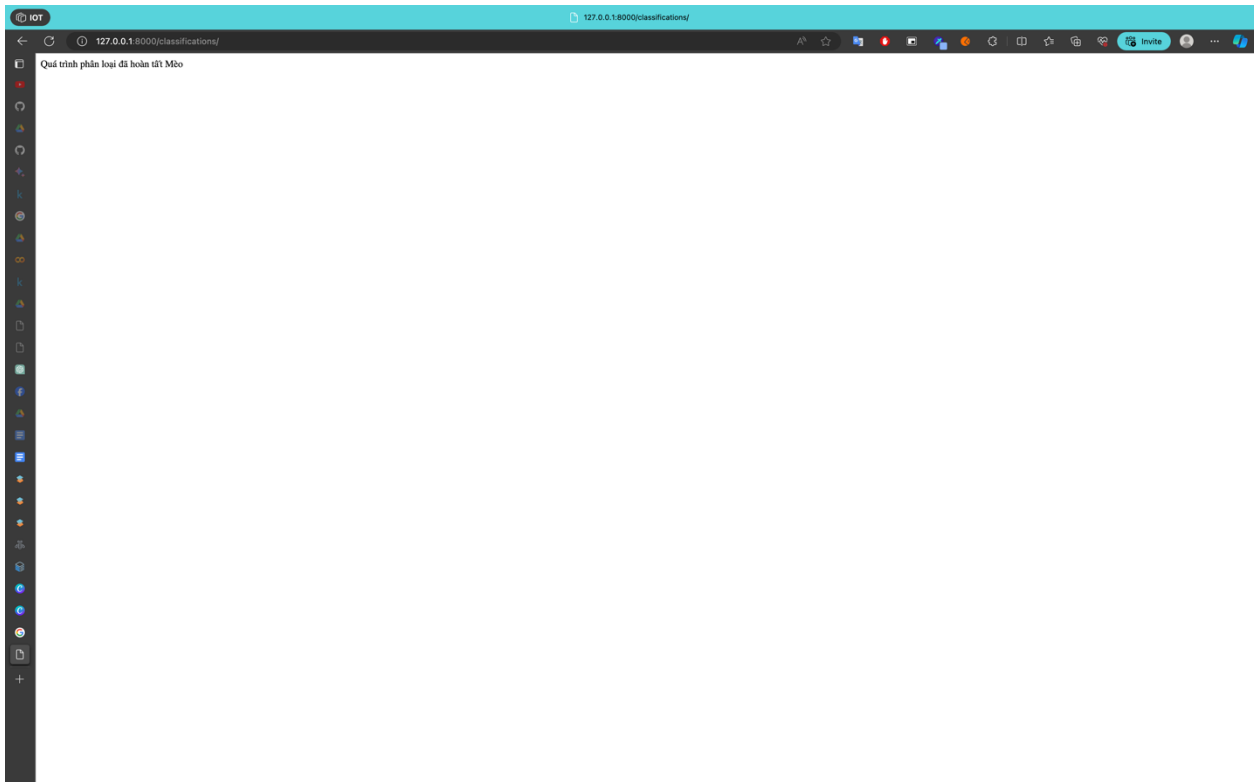
Ảnh demo 1 Mô tả hệ thống nhận diện chó và mèo.



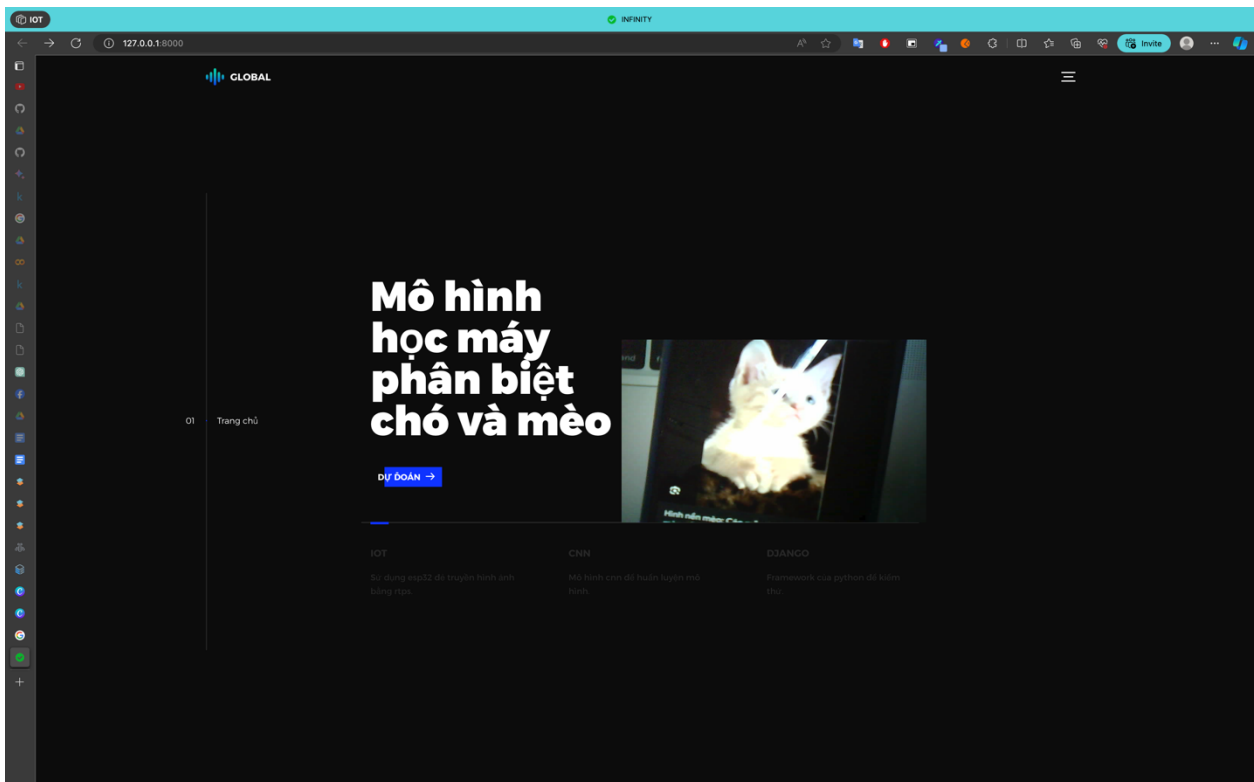
Ảnh demo 2 Giao diện chính.



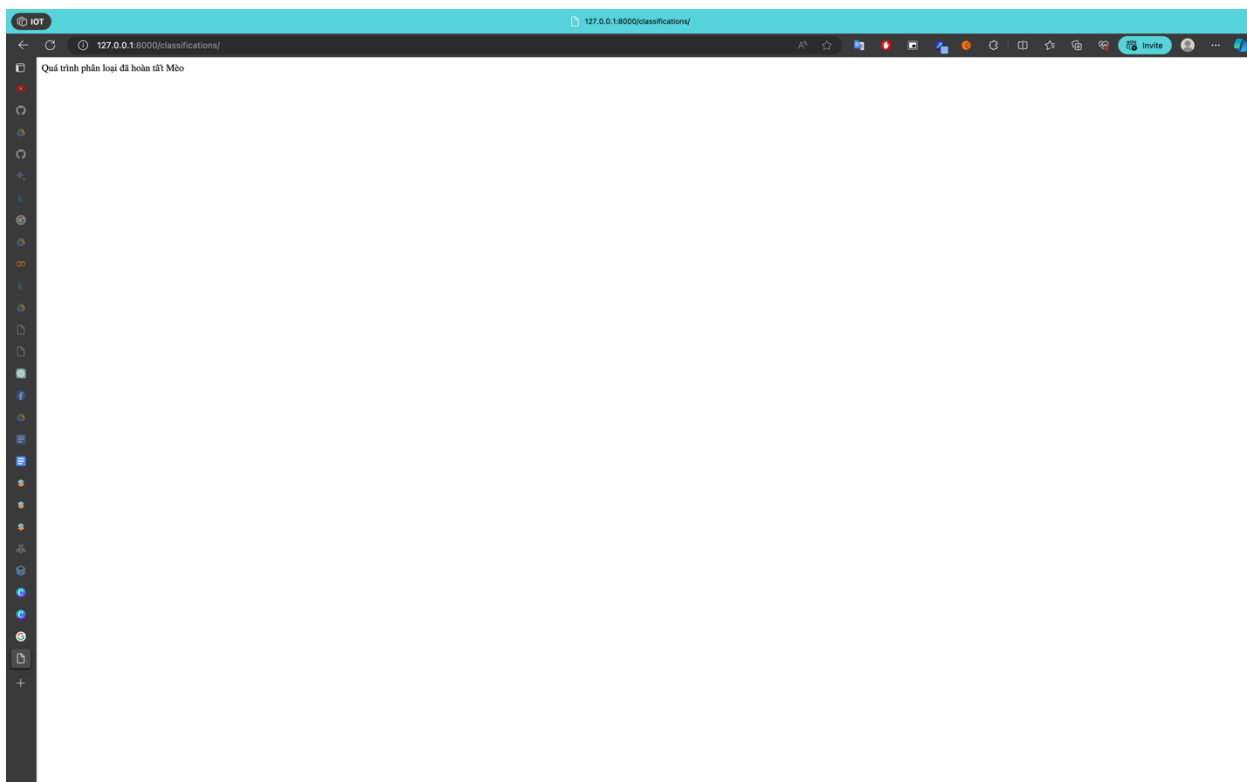
Ảnh demo 3 Chuẩn bị nhận diện chó.



*Ảnh demo 4 Kết quả nhận diện chó.*



*Ảnh demo 5 Chuẩn bị nhận diện mèo.*



ẢNH DEMO 6 KẾT QUẢ NHẬN DIỆN MÈO.

## TÀI LIỆU THAM KHẢO

- [1] H. Nguyễn, "Thuật toán CNN là gì? Tìm hiểu về Convolutional Neural Network," vietnix, 2020. [Online]. Available: <https://vietnix.vn/cnn-la-gi/>.
- [2] ibm, "What are convolutional neural networks?," ibm.
- [3] Z. A. S. S. Rui Yang, "Multilayer Extreme Learning Convolutional Feature Neural Network Model for the Weak Feature Classification and Status Identification of Planetary Bearing," Journal of Sensors-Vol. 2022, pp 7693393:1-7693393:11 , 2022.
- [4] S.-H. W. C.-Y. C. Shih-Chang Hsia, "Convolution neural network with low operation FLOPS and high accuracy for image recognition," Journal of Real-time Image Processing (Springer Berlin Heidelberg)-Vol. 18, Iss: 4, pp 1309-1319, 2021.
- [5] NEWCAS, "SqueezeVGGNet: A Methodology for designing low complexity VGG Architecture for Resource Constraint Edge Applications," IEEE Interregional NEWCAS Conference (NEWCAS), 2022.
- [6] Department of Computer Science and Engineering Florida Institute of technology Melbourne, "DQNAS: Neural Architecture Search using Reinforcement Learning," Department of Computer Science and Engineering Florida Institute of technology Melbourne, United States , 2023.