**MỤC LỤC**

[CHƯƠNG 1: GIỚI THIỆU 4](#_Toc165742588)

[1.1. Giới thiệu chung 4](#_Toc165742589)

[1.2. Nêu bài toán 4](#_Toc165742590)

[1.3. Phương pháp tiếp cận bài toán 5](#_Toc165742591)

[1.4. Giới hạn đề tài 6](#_Toc165742592)

[1.5. Đối tượng và phạm vi nghiên cứu 6](#_Toc165742593)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 7](#_Toc165742594)

[2.1. Giới thiệu về mạng CNN 7](#_Toc165742595)

[2.2. Kiến trúc của CNN 7](#_Toc165742596)

[2.3. Hàm OpenCV 10](#_Toc165742597)

[2.4. Hàm OS 11](#_Toc165742598)

[2.5. TesnsorFlow 12](#_Toc165742599)

[2.6. Keras 13](#_Toc165742600)

[CHƯƠNG 3: XÂY DỰNG CHƯƠNG TRÌNH 14](#_Toc165742601)

[3.1. Xây dựng mô hình bài toán 14](#_Toc165742602)

[3.2. Các bước thực hiện 14](#_Toc165742603)

[3.2.1. Xử lý tập dữ liệu 14](#_Toc165742604)

[3.2.2. Huấn luyện dữ liệu đưa vào và trích chọn các đặc trưng. 15](#_Toc165742605)

[3.2.3. Hiển thị dữ liệu đầu ra 20](#_Toc165742606)

[CHƯƠNG 4 KẾT QUẢ THÍ NGHIỆM 22](#_Toc165742607)

[4.1. Kết quả sau quá trình huấn luyện 22](#_Toc165742608)

[4.2. Đưa hình ảnh cần nhận diện 24](#_Toc165742609)

[KẾT LUẬN 29](#_Toc165742610)

[TÀI LIỆU THAM KHẢO 30](#_Toc165742611)

**DANH MỤC HÌNH ẢNH**

[Hình 1.1. Sự đa dạng về chủng loại của 1 loại rau củ 5](#_Toc165742146)

[Hình 1.2 .Các thành phần trong tập dữ liệu 5](#_Toc165742147)

[Hình 2.1. Kiến trúc cơ bản của mạng CNN 8](file:///C:\Users\admin\Documents\Zalo%20Received%20Files\BIA.docx#_Toc165742036)

[Hình 2.2. Bộ lọc tích chập được sử dụng trên ma trận điểm ảnh 8](#_Toc165742037)

[Hình 2.3. Các phương thức lấy mẫu 10](#_Toc165742038)

[Hình 2.4. Cấu trúc cơ bản của OpenCV 11](file:///C:\Users\admin\Documents\Zalo%20Received%20Files\BIA.docx#_Toc165742039)

[Hình 3.1. Mô hình tổng quan hệ thống 14](#_Toc165742102)

[Hình 3.2. Mô hình mạng neuron network 16](#_Toc165742103)

[Hình 3.3. Ảnh đầu vào 17](#_Toc165742104)

[Hình 3.4. Hình ảnh sau khi xử lý của khối B2 18](#_Toc165742105)

[Hình 3.5. Hình ảnh sau khi xử lý của khối B3 18](#_Toc165742106)

[Hình 3.6. Dữ liệu phân lớp của mô hình CNN 20](#_Toc165742107)

[Hình 3.7. Tên ứng với ảnh 20](#_Toc165742108)

[Hình 4.1. Đoạn mã xây dựng và huấn luyện mô hình 22](#_Toc165742111)

[Hình 4.2. Trọng số được huấn luyện 23](#_Toc165742112)

[Hình 4.3. Độ chính xác và giá trị hàm loss theo từng epoch 24](#_Toc165742113)

**CHƯƠNG 1: GIỚI THIỆU**

* 1. **Giới thiệu chung**

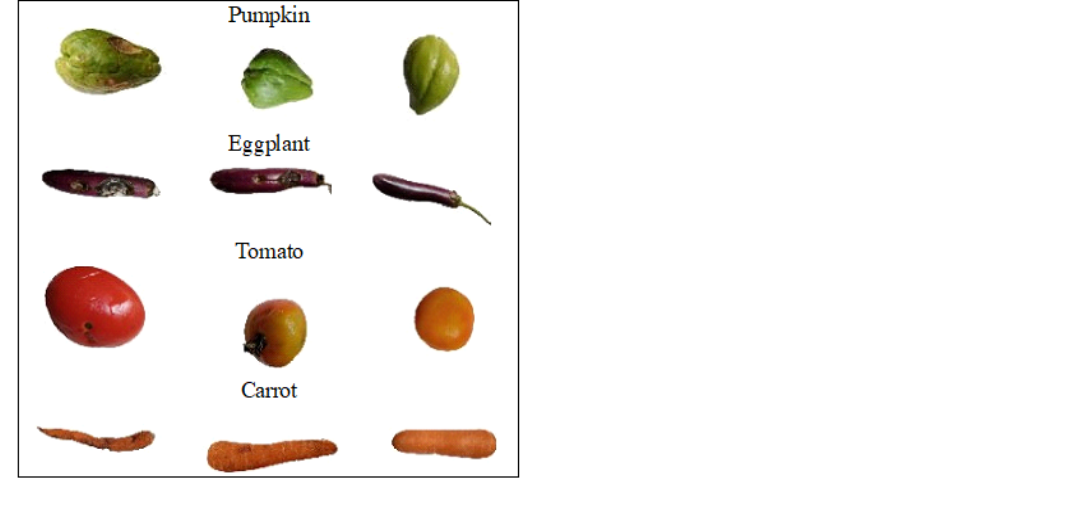
Trong thời đại công nghệ ngày nay, việc áp dụng trí tuệ nhân tạo vào lĩnh vực thị giác máy tính đang trở nên ngày càng phổ biến và quan trọng. Một trong những ứng dụng tiêu biểu của trí tuệ nhân tạo là nhận diện và phân loại đối tượng trong hình ảnh, trong đó bao gồm cả việc nhận diện rau củ .

Bài toán “Nhận diện rau củ bằng mạng neural convolutional (CNN)” nhằm mục đích tự động phân loại các loại rau quả từ hình ảnh. Điều này có thể được áp dụng trong nhiều ứng dụng thực tế như hệ thống quản lý kho hàng, ứng dụng di động hỗ trợ việc mua sắm, hoặc trong lĩnh vực nghiên cứu về dinh dưỡng. Quá trình này bắt đầu từ việc xây dựng một tập dữ liệu chứa các hình ảnh của các loại rau quả khác nhau. Sau đó, một mô hình CNN được huấn luyện trên tập dữ liệu này để học cách phân biệt giữa các loại rau quả. Quá trình huấn luyện mô hình này thường bao gồm các bước như trích xuất đặc trưng, tối ưu hóa tham số mô hình và đánh giá hiệu suất.   
  
Khi mô hình đã được huấn luyện, nó có thể được triển khai vào các hệ thống thực tế để nhận diện và phân loại rau quả từ hình ảnh chụp từ các thiết bị như máy ảnh hoặc điện thoại di động. Điều này mang lại lợi ích không chỉ trong việc tăng cường trải nghiệm người dùng mà còn trong việc tối ưu hóa quản lý và phân phối rau củ.

* 1. **Nêu bài toán**

Ở đây, chúng ta có đầu vào là các hình ảnh các loại rau củ và qua quá trình xử lý ta được đầu ra là các nhãn của rau củ tương ứng với hình ảnh đó.

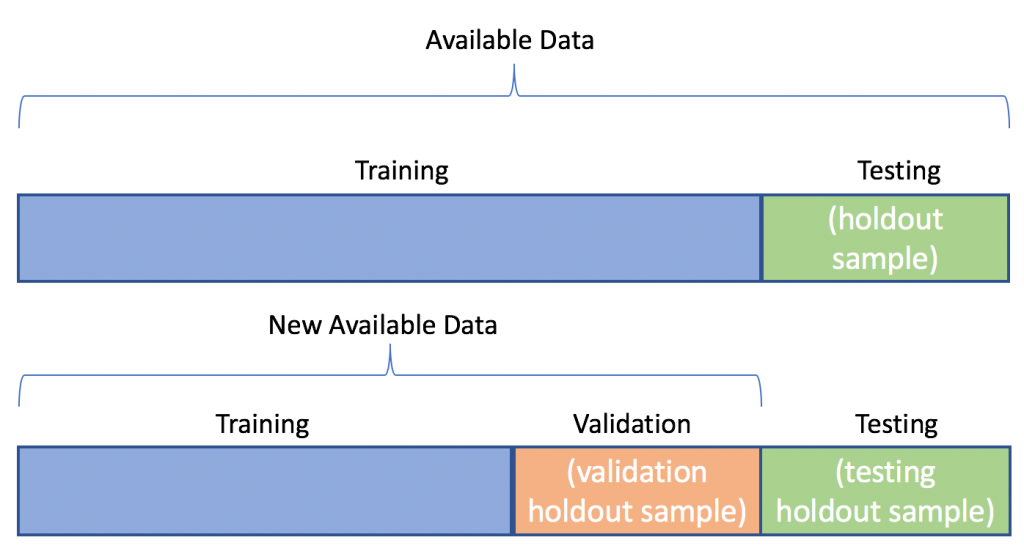
Là một trường hợp cụ thể của bài toán nhận dạng và phân lớp, bài toán nhận dạng hoa quả kế thừa các khó khăn vốn có của bài toán gốc, và kèm theo là các khó khăn riêng của chính nó, như: số lượng khổng lồ về chủng loại rau củ theo mùa, vùng miền, địa hình… với vô số loại có hình dáng, màu sắc, kết cấu giống nhau, dải biến thiên màu sắc theo chu kỳ phát triển của quả từ lúc còn xanh đến lúc chín, hay sự đa dạng về hình dạng của cùng một loại rau củ do ảnh hưởng của thời tiết,….



Hình 1.1: Sự đa dạng về chủng loại của 1 loại rau củ

Hình 1 1

* 1. **Phương pháp tiếp cận bài toán**
* Thu thập dữ liệu: Bước đầu tiên là thu thập một bộ dữ liệu ảnh chứa các hình ảnh của các loại rau củ khác nhau. Dữ liệu này cần phải đủ lớn và đa dạng để mô hình có thể học được các đặc điểm của từng loại rau củ.
* Xử lý dữ liệu : Dữ liệu được chia thành các tập : train, test, validation

 Hình 1 2

*Hình 1.2 Các thành phần trong tập dữ liệu*

+ Tập Train để huấn luyện mô hình học

+ Tập Test sử dụng để đánh giá hiệu suất của mô hình đã huấn luyện. Sau khi mô hình đã học từ tập train, chúng ta sử dụng tập Test để kiểm tra xem mô hình có dự đoán đúng trên dữ liệu mà nó chưa từng thấy hay không.

+ Tập Validation được sử dụng trong quá trình huấn luyện để đánh giá hiệu suất của mô hình mỗi vòng lặp. Thông qua việc so sánh kết quả trên tập validation, chúng ta có thể điều chỉnh các siêu tham số của mô hình tránh việc mô hình bị overfitting (quá khớp) hoặc underfitting (không khớp đủ) với dữ liệu.

Sau đó tiến hành các bước tiền xử lý như resize ảnh về kích thước chuẩn, chuẩn hóa dữ liệu, xoay, dịch chuyển, thay đổi độ sáng để tăng tính đa dạng của dữ liệu.

* Xây dựng mô hình mạng CNN : gồm các lớp tích chập, pooling, kết nối đầy đủ.
* Huyến luyện mô hình CNN
* Đánh giá mô hình
* Tinh chỉnh mô hình
* Dự đoán : sử dụng mô hình để dự đoán loại rau củ trong các hình ảnh mới không được sử dụng trong quá trình huấn luyện.
  1. **Giới hạn đề tài**

Phạm vi giới hạn của đề tài:

* Đề tài chủ yếu sử dụng các thư viện có sẵn trong lập trình python, đồng thời ứng dụng các mô hình toán học trong mạng nơ ron nhân tạo. Điều kiện đầu vào của mô hình là các tệp hình ảnh có giới hạn, đầu ra phân tích hoa quả được đưa vào có độ chính xác tương đối.
* Đối với chức năng thì có sự giới hạn do hiểu biết về lập trình và cách ứng dụng của mô hình vào các hệ thống lớn.
  1. **Đối tượng và phạm vi nghiên cứu**

Các đối tượng cần nghiên cứu và phạm vi nghiên cứu có thể giải quyết được đề tài:   
- Đối tượng nghiên cứu: Trí tuệ nhân tạo, lập trình python.   
- Phạm vi nghiên cứu: Mạng CNN, các module trong python : opencv, numpy, os, tensorflow, image…

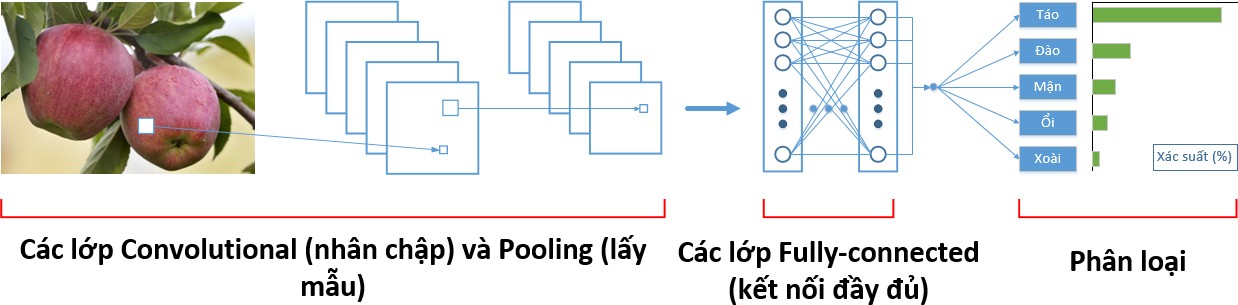
# **CHƯƠNG 2: CƠ SỞ LÝ THUYẾT**

## **2.1. Giới thiệu về mạng CNN**

* Mạng nơ-ron tích chập (CNN - Convolutional Neural Network) là một trong những mô hình mạng Học sâu phổ biến nhất hiện nay, có khả năng nhận dạng và phân loại hình ảnh với độ chính xác rất cao, thậm chí còn tốt hơn con người trong nhiều trường hợp. Mô hình này đã và đang được phát triển, ứng dụng vào các hệ thống xử lý toán tagging tự động, tìm kiếm ảnh hoặc gợi ý sản phẩm cho người tiêu dùng, ảnh lớn của Facebook, Google hay Amazon… cho các mục đích khác nhau.
* Mạng nơ-ron tích chập (CNN) ra đời từ nỗ lực cải tiến các mô hình nơ-ron nhân tạo truyền thống để xử lý thông tin từ hình ảnh. Trong các mạng nơ-ron truyền thẳng, sự kết nối đầy đủ giữa mỗi điểm ảnh và mỗi node làm giảm hiệu suất khi xử lý ảnh lớn, vì số lượng liên kết tăng theo cấp số nhân và gây ra vấn đề về khối lượng tính toán. Hơn nữa, việc kết nối đầy đủ này cũng không hiệu quả, vì thông tin chủ yếu thường phụ thuộc vào mối quan hệ giữa các điểm ảnh lân cận, không cần thiết phải liên quan đến mọi điểm ảnh trong hình ảnh. CNN được thiết kế để giải quyết vấn đề này bằng cách thay đổi cấu trúc của mạng.
* Thay vì kết nối đầy đủ, CNN chỉ sử dụng một phần nhỏ của hình ảnh, được gọi là cửa sổ trượt (convolutional window), và kết nối các điểm ảnh trong cửa sổ này với các node trong lớp tiếp theo. Điều này giúp giảm đáng kể số lượng tham số cần học và tăng cường khả năng của mạng trong việc nhận diện các đặc trưng cục bộ trong hình ảnh, đồng thời giảm thiểu sự dư thừa và tăng tốc quá trình tính toán.

**2.2. Kiến trúc của CNN**

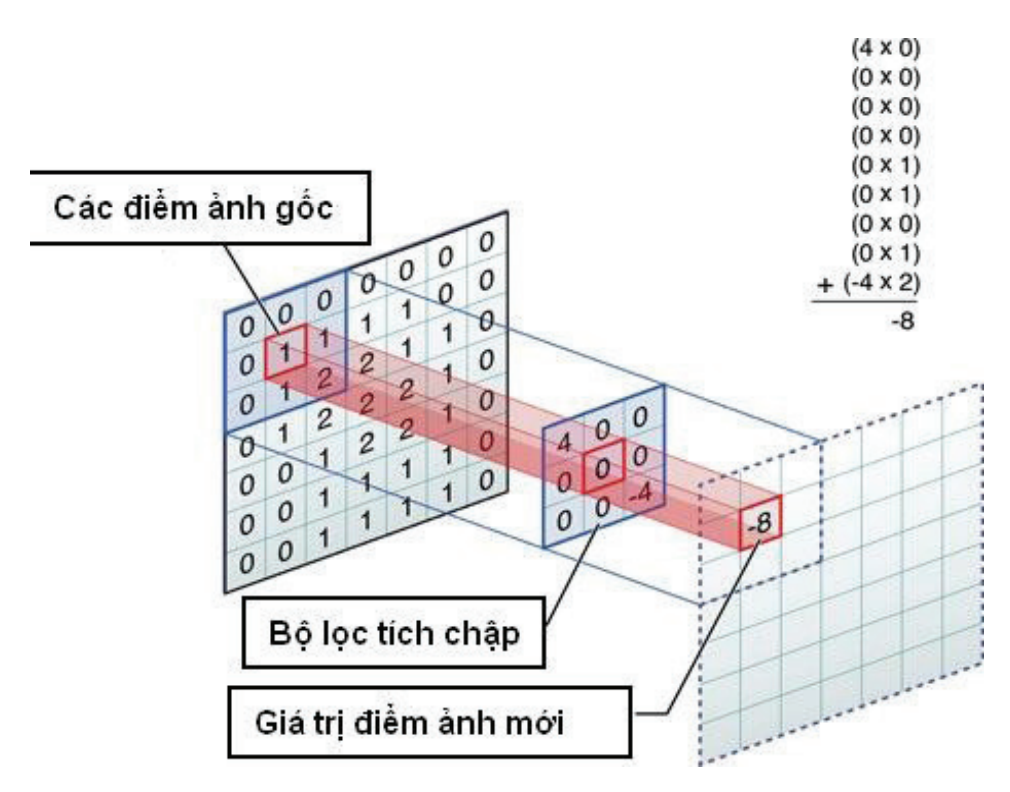
* Hình 2.1 trình bày một kiến trúc mạng CNN, các lớp cơ bản trong một mạng CNN bao gồm: lớp tích chập (Convolutional); lớp kích hoạt phi tuyến ReLU (Rectified Linear Unit); lớp lấy mẫu (Pooling); lớp kết nối đầy đủ (Fully connected) được thay đổi về số lượng và cách sắp xếp để tạo ra các mô hình huấn luyện phù hợp cho từng bài toán khác nhau. Các lớp tích chập (Convolutional), kích hoạt phi tuyến ReLU và lấy mẫu (Pooling) Các lớp kết nối đầy đủ (Fully connected)

*Hình 2.1. Kiến trúc cơ bản của mạng CNN*

Hình 1 4

1

* Lớp tích chập : đây là thành phần quan trọng nhất trong mạng CNN, thể hiện sự liên kết cục bộ thay vì kết nối toàn bộ các điểm ảnh. Các liên kết cục bộ được tính toán bằng phép tích chập giữa các giá trị điểm ảnh trong một vùng ảnh cục bộ với các bộ lọc filters có kích thước nhỏ.



Hình 2 2Hình 1 5

*Hình 2.2. Bộ lọc tích chập được sử dụng trên ma trận điểm ảnh*

* Trong hình 2.2, bộ lọc được sử dụng là một ma trận có kích thước 3x3, bộ lọc này dịch chuyển lần lượt qua từng vùng ảnh đến khi hoàn thành quét toàn bộ bức ảnh, tạo ra một bức ảnh mới có kích thước nhỏ hơn hoặc bằng với kích thước ảnh đầu vào. Kích thước này được quyết định tùy theo kích thước các khoảng trắng được thêm ở viền bức ảnh gốc và được tính theo công thức sau:   
   O = (i+2\*p-k)/s+1 (2.1)

Trong đó: O: kích thước ảnh đầu ra;

i: kích thước ảnh đầu vào;

p: kích thước khoảng trắng phía ngoài viền của ảnh gốc;

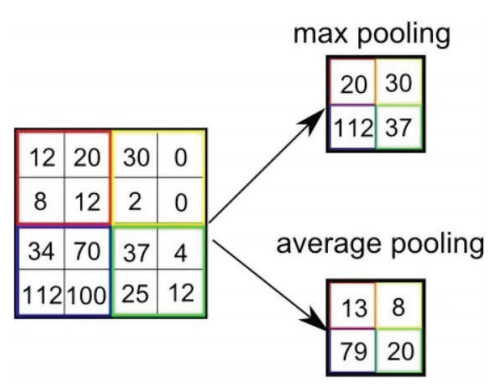
k: kích thước bộ lọc;

s: bước trượt của bộ lọc

* Như vậy, sau khi đưa một bức ảnh đầu vào cho lớp tích chập nhận được kết quả đầu ra là một loạt ảnh tương ứng với các bộ lọc đã được sử dụng để thực hiện phép tích chập. Các trọng số của các bộ lọc này được khởi tạo ngẫu nhiên trong lần đầu tiên và sẽ được cập nhật trong quá trình huấn luyện
* Lớp kích hoạt phi tuyến ReLU : được xây dựng để đảm bảo tính phi tuyến của mô hình huấn luyện sau khi đã thực hiện một loạt các phép tính toán tuyến tính qua các lớp tích chập. Lớp kích hoạt phi tuyến sử dụng các hàm kích hoạt phi tuyến như ReLU hoặc sigmoid, tanh… để giới hạn phạm vi biên độ cho phép của giá trị đầu ra. Trong số các hàm kích hoạt này, hàm ReLU được chọn do cài đặt đơn giản, tốc độ xử lý nhanh mà vẫn đảm bảo được tính toán hiệu quả.
* Phép tính toán của hàm ReLU chỉ đơn giản là chuyển tất cả các giá trị âm thành giá trị 0. Lớp ReLU được áp dụng ngay phía sau lớp tích chập, với đầu ra là một ảnh mới có kích thước giống với ảnh đầu vào, các giá trị điểm ảnh cũng hoàn toàn tương tự, trừ các giá trị âm đã bị loại bỏ.

𝑓(𝑥) = 𝑚𝑎𝑥(0, 𝑥) (2.2)

* Lớp được đặt sau lớp tích chập và lớp ReLU để làm giảm kích thước ảnh đầu ra trong khi vẫn giữ được các thông tin quan trọng của ảnh đầu vào. Việc giảm kích thước dữ liệu có tác dụng làm giảm được số lượng tham số cũng như tăng hiệu quả tính toán. Lớp lấy mẫu cũng sử dụng một cửa sổ trượt để quét toàn bộ các vùng trong ảnh như lớp tích chập, và thực hiện phép lấy mẫu thay vì phép tích chập, sẽ chọn lưu lại một giá trị duy nhất đại diện cho toàn bộ thông tin của vùng ảnh đó.   
  - Hình 2.3 thể hiện các phương thức lấy mẫu thường được sử dụng nhất hiện nay, đó là Max Pooling (lấy giá trị điểm ảnh lớn nhất) và Avarage Pooling (lấy giá trị trung bình của các điểm ảnh trong vùng ảnh cục bộ).



Hình 2 3 1 6

*Hình 2.3. Các phương thức lấy mẫu*

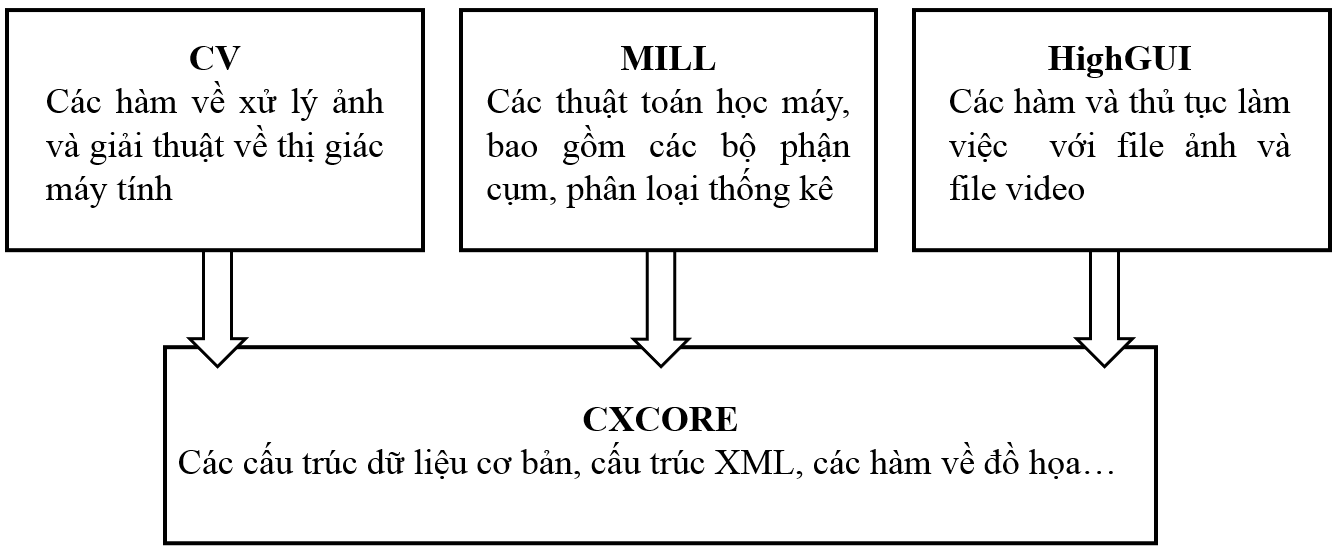
* Như vậy, với mỗi ảnh đầu vào được đưa qua lấy mẫu ta thu được một ảnh đầu ra tương ứng, có kích thước giảm xuống đáng kể nhưng vẫn giữ được các đặc trưng cần thiết cho quá trình tính toán sau này.
* Lớp kết nối đầy đủ : được thiết kế tương tự như trong mạng nơ ron truyền thống, tất cả các điểm ảnh được kết nối đầy đủ với nơ ron trong lớp tiếp theo.   
  So với mạng nơ ron truyền thống, các ảnh đầu vào của lớp này đã có kích thước được giảm bớt rất nhiều, đồng thời vẫn đảm bảo các thông tin quan trọng của ảnh cho việc nhận dạng. Do vậy, việc tính toán nhận dạng sử dụng mô hình truyền thẳng đã không còn phức tạp và tốn nhiều thời gian như trong mạng nơ ron truyền thống.

**2.3. Hàm OpenCV**

- OpenCV là một thư viện mã nguồn mở hàng đầu cho thị giác máy tính (computer vision), xử lý ảnh và máy học, và các tính năng tăng tốc GPU trong hoạt động thời gian thực.   
- OpenCV được phát hành theo giấy phép BSD, do đó nó hoàn toàn miễn phí cho cả học thuật và thương mại. Nó có các interface C++, C, Python, Java và hỗ trợ Windows, Linux, Mac OS, iOS và Android.  
- OpenCV được thiết kế để tính toán hiệu quả và với sự tập trung nhiều vào các ứng dụng thời gian thực. Được viết bằng tối ưu hóa C/C++, thư viện có thể tận dụng lợi thế của xử lý đa lõi.

- OpenCV có cộng đồng hơn 47 nghìn người dùng và số lượng download vượt quá 6 triệu lần. Phạm vi sử dụng từ nghệ thuật tương tác, cho đến lĩnh vực khai thác mỏ, bản đồ trên web hoặc công nghệ robot.

* Một số tính năng chính của OpenCV:
* Đọc và Ghi hình ảnh và Video
* Xử lý hình ảnh cơ bản (cắt, phóng to, thu nhỏ, thay đổi độ sáng, tương phản)
* Phân Loại và Nhận Diện Đối Tượng: đi kèm với các thuật toán phân loại và nhận diện đối tượng như phát hiện khuôn mặt (face detection), phát hiện đối tượng (object detection), phát hiện và nhận diện vật thể (object recognition).
* Xử Lý Video và Thị Giác Máy Tính : hỗ trợ xử lý video và thị giác máy tính như theo dõi vật thể trong video, xác định chuyển động, và xác định vị trí và hướng của vật thể.
* Cấu trúc của OpenCV: Cấu trúc của opencv được chia làm 5 phần chính, 4 trong số đó được chia ra như trong hình sau:



Hình 1 7

Hình 2 4

*Hình 2.4. Cấu trúc cơ bản của OpenCV*

* CV (computer vision) là thành phần chữa những xử lý ảnh cơ sở và thuật toán thị giác máy tính mức cao.
* MLL (machine learning library) là thư viện machine learning, cái này bao gồm rất nhiều lớp thống kê và gộp công cụ xử lý.
* HighGUI chứa các thủ tục vào ra và các hàm dùng cho việc lưu trữ và tải ảnh video. CXCore chứa cấu trúc và nội dung dữ liệu cơ sở.

**2.4. Hàm OS**

* Khái niệm: Là một module bao gói các lệnh gọi POSIX hoặc mô phỏng chúng trên một số nền tảng mà không có POSIX. Bạn sẽ thấy các hàm gần như được đặt tên giống hệt như trong C và các ngôn ngữ khác cũng như cách thực hiện tương tự. Có thể dễ dàng nhận ra các hằng số như os.R\_OK, os.WNOHANG ... được đặt tên chính xác theo C/POSIX
* Một số hàm os :
* os.getcwd(): Trả về đường dẫn của thư mục làm việc hiện tại.
* os.path.exists(path): Kiểm tra xem một đường dẫn có tồn tại không.
* os.mkdir(path): Tạo một thư mục mới tại đường dẫn path.
* os.getcwd(): Trả về đường dẫn của thư mục làm việc hiện tại.

…….

**2.5. TesnsorFlow**

TensorFlow là một thư viện mã nguồn mở của Google được sử dụng để xây dựng và huấn luyện mô hình học máy và mạng nơ-ron.

* Kiếntrúc:  
  + Phần 1: Tiền xử lý dữ liệu: Trong phần này, dữ liệu được chuẩn bị và tiền xử lý để sẵn sàng cho quá trình huấn luyện mô hình. Các bước tiền xử lý có thể bao gồm: chuẩn hóa dữ liệu, chia dữ liệu thành batch, mã hóa dữ liệu, xử lý dữ liệu bị thiếu, và các biến đổi dữ liệu khác.

+ Phần 2: Dựng model: Trong phần này, một biểu đồ tính toán (graph) được tạo ra để định nghĩa mô hình học máy. Biểu đồ này bao gồm các node đại diện cho các phép toán toán học (operations) và các cạnh đại diện cho dữ liệu chạy qua các phép toán (tensors). Người dùng xây dựng biểu đồ này bằng cách sử dụng các API của TensorFlow để định nghĩa các layer, activation functions, loss functions và metrics.   
+ Phần 3: Train và ước tính model: Trong phần này, mô hình được huấn luyện trên dữ liệu đã được tiền xử lý bằng cách sử dụng các thuật toán tối ưu hóa như gradient descent. Sau khi huấn luyện xong, mô hình có thể được sử dụng để ước tính (predict) trên dữ liệu mới hoặc đánh giá hiệu suất của mô hình trên dữ liệu kiểm tra.

* Cách hoạt động : Khi Tensorflow hoạt động sẽ cho phép các lập trình viên có thể tạo ra dataflow graph, cũng như cấu trúc mô tả làm sao để cho dữ liệu có thể di chuyển qua 1 biểu đồ; hoặc di chuyển qua 1 seri mà các node đang xử lý. Mỗi một node có trong đồ thị thường đại diện cho 1 operation toán hoặc và mỗi kết nối thường hay edge giữa các node với nhau. Từ đó, mỗi kết nối hoặc edge giữa các node được xem là mảng dữ liệu đa chiều. Tensorflow sẽ cung cấp tất cả mọi điều đến cho lập trình viên dựa theo phương thức của ngôn ngữ Python. Ngôn ngữ này sẽ cung cấp nhiều cách tiện lợi để ta có thể hiểu được nên làm thế nào cho các high-level abstractions có thể kết hợp được với nhau. Node cũng như tensor có trong Tensorflow chính là đối tượng của Python. Và, mọi ứng dụng Tensorflow bản thân chúng chính là một ứng dụng Python

## **2.6. Keras**

* Keras là một API cấp cao của nền tảng TensorFlow, được phát triển để giải quyết các vấn đề trong machine learning (ML), tập trung vào học sâu hiện đại. Nó cung cấp một giao diện dễ tiếp cận và hiệu quả để xây dựng và huấn luyện các mô hình ML, từ xử lý dữ liệu đến điều chỉnh siêu tham số cho đến triển khai.
* Kiến trúc :

1. Sequential Model:

Sequential Model là một kiểu model cơ bản trong Keras, cho phép bạn xây dựng một chuỗi các lớp một cách tuần tự. Cách này thích hợp cho các mô hình có cấu trúc đơn giản, mà dữ liệu di chuyển tuần tự qua các lớp.

2. Functional API:

Functional API là một phương pháp mạnh mẽ hơn trong Keras, cho phép bạn xây dựng các mô hình có cấu trúc phức tạp hơn. Với Functional API, bạn có thể tạo ra các mô hình có nhiều input và output, chia sẻ lớp giữa các nhánh của mạng, và thực hiện các phức tạp hơn về logic kiến trúc.

* Cách hoạt động : Keras cung cấp giao diện đơn giản để xây dựng và huấn luyện mô hình ML (machine learning). Sử dụng các lớp và mô hình để tạo biểu đồ tính toán, trong đó mỗi lớp đại diện cho một phép biến đổi và mô hình là tập hợp các lớp kết nối với nhau

# **CHƯƠNG 3: XÂY DỰNG CHƯƠNG TRÌNH**

## **3.1. Xây dựng mô hình bài toán**

A diagram of a graph

Description automatically generated with medium confidence

Hình 3 1

*Hình 3.1. Mô hình tổng quan hệ thống*

Mô hình nhận dạng được chia thành 3 bước chính:

* Bước 1: Xử lý tập dữ liệu
* Bước 2: Huấn luyện dữ liệu đưa vào và trích chọn các đặc trưng.
* Bước 3: Phân loại ảnh rau củ dựa trên đặc trưng được trích chọn và đưa ra kết quả.

**3.2. Các bước thực hiện**

**3.2.1. Xử lý tập dữ liệu**

Các dữ liệu dùng để huấn luyện có thể lấy từ ảnh có sẵn

Tập dữ liệu gồm các ảnh của các loại rau củ khác nhau và đảm bảo các hình ảnh này không có cùng kích thước.

Bước 1: Tiền xử lý

A screen shot of a computer

Description automatically generated

* Trong mã code, bước này được thực hiện thông qua việc sử dụng các đối tượng train\_data\_gen và validation\_data\_gen từ module ImageDataGenerator của thư viện Keras Preprocessing:
* Cụ thể, khi khởi tạo ImageDataGenerator đã sử dụng tham số rescale = 1./255 để chuẩn hóa giá trị các pixel trong hình ảnh về khoảng giá trị từ 0 đến 1. Đồng thời, sử dụng các tham số như: zca\_epsilon, rotation\_range, vertical\_flip,... nhằm tăng tính đa dạng của dữ liệu và cải thiện hiệu suất và tính tổng quát của mô hình

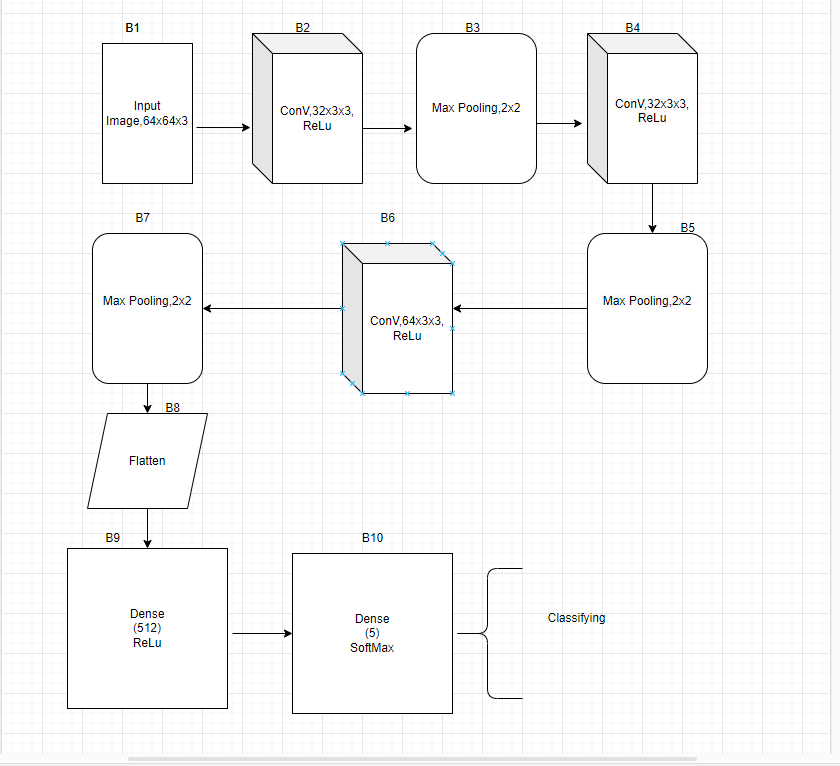
A screen shot of a computer

Description automatically generated

* Sau đó sử dụng hàm flow\_from\_directory để tạo ra các dữ liệu huấn luyện (training\_set) và dữ liệu xác thực (validation\_set) từ thư mục chứa các hình ảnh của hoa quả.
* Trong quá trình này, hình ảnh được đọc từ thư mục và được tự động chia thành các batch với kích thước đã được xác định (batch\_size = 12) và được resize về kích thước mục tiêu là (64, 64) bằng tham số target\_size. Những batch dữ liệu này sau đó được sử dụng để huấn luyện và đánh giá mô hình nhận diện và phân loại hoa quả, đảm bảo rằng mô hình được huấn luyện trên những phần quan trọng nhất của hình ảnh, từ đó đạt được kết quả tốt nhất khi áp dụng vào dữ liệu mới.

### 3.2.2. Huấn luyện dữ liệu đưa vào và trích chọn các đặc trưng.

Mô hình CNN được thiết kế gồm hai phần chức năng là trích chọn đặc trưng của ảnh và phân lớp đối tượng dựa trên đặc trưng đã chọn. Mô hình CNN bao gồm nhiều lớp, số lớp nơron và độ lớn (số nơron) của mỗi lớp ảnh hưởng đến chất lượng cũng như độ phức tạp trong tính toán của mạng nơron. Các nghiên cứu thường điều chỉnh hai yếu tố này tuỳ theo bài toán ứng dụng để đạt được chất lượng mong muốn và đồng thời đảm bảo sự phức tạp tính toán chấp nhận được. diễn đạt theo cách khác



Hình 3 2Hình 3.2 Mô hình mạng neuron network

Mỗi lớp nơron trong mô hình CNN lấy một mảng nhiều chiều gồm các số làm đầu vào và tạo ra một mảng số nhiều chiều khác ở đầu ra (sau đó trở thành đầu vào của lớp tiếp theo). Khi phân loại hình ảnh khuôn mặt, đầu vào của lớp nơron đầu tiên là kích thước hình ảnh đầu vào. Kích thước đầu ra của lớp cuối cùng là tập hợp các khả năng của các lớp khác nhau được phân loại cho mỗi ảnh đầu vào. Cả ba loại lớp nơron để xây dựng kiến trúc của CNN bao gồm: 3 lớp tích chập (CONV), 2 lớp nơron Maxpooling và 2 lớp nơron kết nối đầy đủ để phân loại (gọi là lớp Dense). Mỗi lớp CONV được kết nối theo sau nó bởi một lớp Maxpooling, áp dụng cơ chế kích hoạt ReLu (Rectified Linear Unit, mặc định là max(x,0)) sau mỗi lớp CONV để đảm bảo đầu vào không âm cho lớp nơron kế tiếp. Theo nguyên tắc xếp chồng các lớp nơron và giảm không gian mẫu (downsampling) tại các kết quả đầu ra của chúng, CNN thực hiện trích xuất các đặc trưng ngày càng trừu tượng và phức tạp hơn, đồng thời, là bất biến đối với các phép biến dạng và chuyển đổi. Mạng tích chập CNN trong mô hình này được chia thành 10 khối (Hình 3.1).

* Khối B1 là ảnh đầu vào có kích thước H×W×1 (cao × rộng × sâu). Trong bài này, ta sẽ sử dụng ảnh có kích thước 64x64x3

1. Hình vẽ sau minh hoạ cho một ảnh đầu vào:



Hình 3 3

Hình 3.3: Ảnh đầu vào

* Khối B2, B4 là lớp nơron tích chập có 32 bộ lọc đặc trưng với kích thước cửa sổ hàm nhân là 3×3. Hàm kích hoạt ReLu được sử dụng trong lớp nơron này. Hiệu ứng này nhằm cải thiện các đặc trưng thưa của toàn mạng và tránh sự phụ thuộc vào việc truyền tham số giữa các nơron. Khối B6 tương tự khối B2 nhưng số các bộ lọc (filter) tăng lên 64

A collage of images of different colors

Description automatically generated

Hình 3 4

Hình 3.4 Hình ảnh sau khi xử lý của khối B2

* Khối B3, B5, B7 là lớp Maxpooling, cửa sổ xử lý có kích thước 2×2 được sử dụng. Lớp giảm không gian mẫu (downsampling) này sử dụng phương pháp MaxPooling với việc có thể giữ lại thông tin hữu ích và cắt giảm lượng dữ liệu cần xử lý ở bước tiếp theo.

A collage of images of a person's body

Description automatically generated

Hình 3 5

Hình 3.5 Hình ảnh sau khi xử lý của khối B3

* Khối B9 là lớp nơron kết nối đầy đủ (fully connected layers). Lớp này cùng với lớp trong khối B10 nhằm mục tiêu phân lớp các đặc trưng được trích chọn ở các lớp trước, do đó chúng tôi thiết kế số nơron đủ lớn. Hàm kích hoạt ReLu cũng được áp dụng. Dữ liệu sau đây là đầu ra của 512 nơron đầu tiên trong khối B9, các giá trị ở đây được xem như biểu diễn dạng số của các đặc trưng khuôn mặt tương ứng với ảnh đầu vào.
* Khối B10 là lớp nơron đầu ra cuối cùng, đây chính là một phân bố cho phân loại của các lớp đối tượng khác nhau với tính năng của hàm kích hoạt Softmax. Áp dụng phương pháp hồi quy Softmax ở lớp đầu ra của mạng CNN nhằm thẩm định cho quá trình huấn luyện mạng. Cơ chế thẩm định này đảm bảo mô hình CNN không bị quá khớp (overfiting) dữ liệu học và có khả năng dự đoán tốt hơn. Cơ chế này được thực hiện thông qua việc chia dữ liệu học thành hai phần, một phần để tính toán cập nhật và điều chỉnh trọng số mạng, một phần để tính toán sai số và cũng đưa vào pha cập nhật thay đổi trọng số mạng. Đầu ra phân lớp cuối cùng của mô hình được xác định dựa trên giá trị cực đại hàm Softmax của nơron tương ứng, với mô hình CNN có lớp (tức là có nơron ở lớp ra) thì ta có công thức xác định như sau:

outputclassified = argmaxCk{Ok¬¬: k = 1, …, |Ok|}

* Trong đó, Ok là đầu ra của nơron thứ ở lớp nơron cuối cùng và tương ứng với nó là lớp đối tượng Ck.

Bảng dưới đây minh hoạ dữ liệu gồm 5 giá trị sau là kết quả đầu ra của lớp nơron cuối cùng (khối B9), tương ứng với kết quả phân lớp của ảnh đầu vào. Kết quả đầu ra của mô hình CNN tương ứng với mỗi ảnh đầu vào gồm 5 giá trị (ở đây chỉ minh hoạ với 5 lớp). Giá trị cao nhất ở vị trí nào (tính theo chỉ số từ 0 và từ trái sang phải) trong bộ 5 giá trị đầu ra chính là định danh cá nhân (vị trí tương ứng giá trị và số in đậm) của dữ liệu ảnh đầu vào.

A close-up of a chart

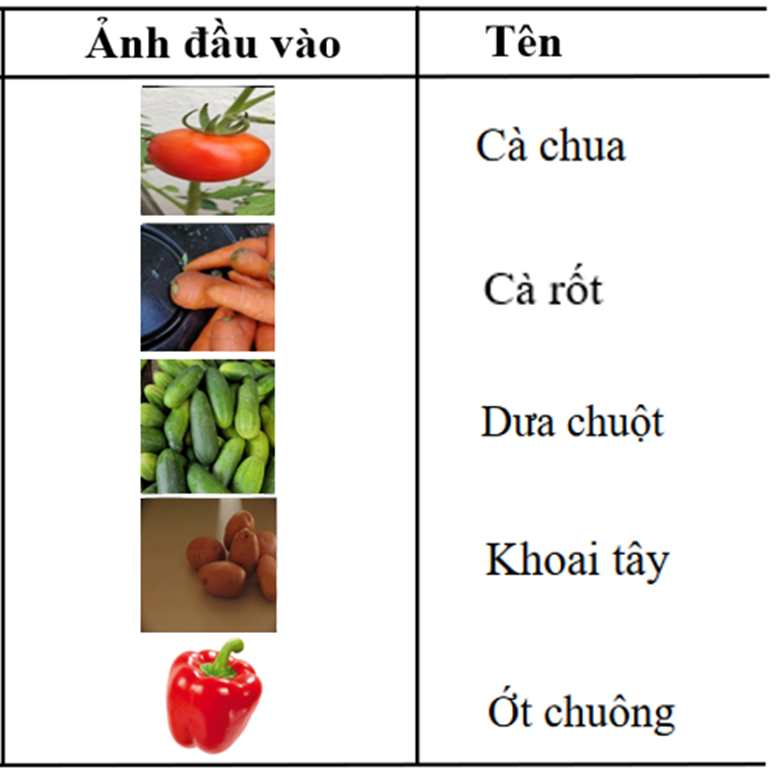
Description automatically generated

Hình 3.6 Dữ liệu phân lớp của mô hình CNN

### 3.2.3. Hiển thị dữ liệu đầu ra

Sau khi mô hình được huấn luyên, lưu module vào file “**model.keras**”, sau đó dùng file này để nhận diện và dự đoán tên loại rau củ của hình ảnh đưa vào.

Từ dữ liệu đầu ra của module, tạo 1 mảng label để đặt tên cho loại rau củ được nhận diện. Cách đặt tên tương ứng với hình ảnh như hình 3.6.

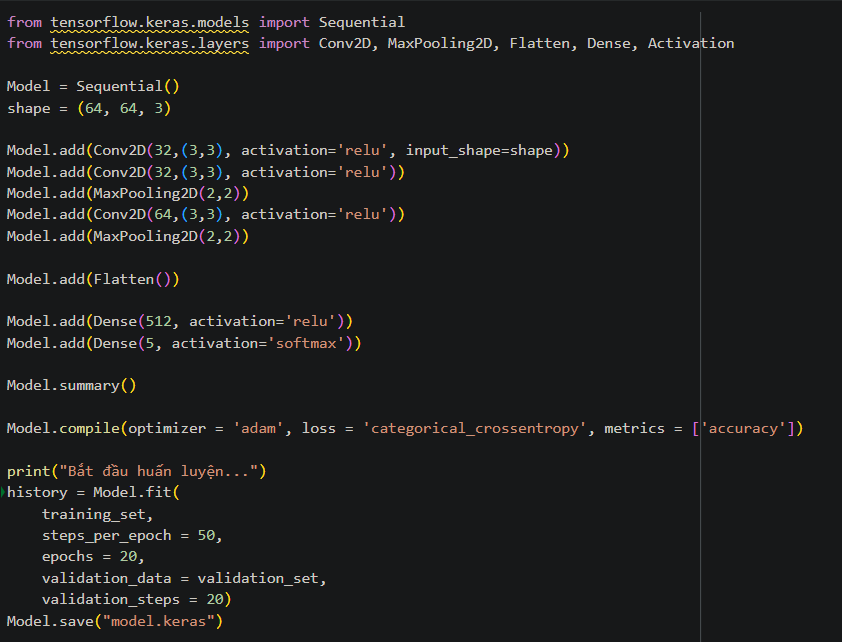


Hình 3 8

*Hình 3.7 Tên ứng với ảnh*

# **CHƯƠNG 4 KẾT QUẢ THÍ NGHIỆM**

## **4.1. Kết quả sau quá trình huấn luyện**



Hình 4 1

*Hình 4.1. Đoạn mã xây dựng và huấn luyện mô hình*

- Quá trình huấn luyện tập dữ liệu

A screenshot of a computer program

Description automatically generated

Hình 4 2

*Hình 4.2. Trọng số được huấn luyện*

A screenshot of a computer screen

Description automatically generated

Hình 4 3

*Hình 4.3. Độ chính xác và giá trị hàm loss theo từng epoch*

*\*\*Nhận xét :*

* Giá trị của hàm lỗi giảm dần theo từng epoch. Giá trị của hàm loss ở epoch 5 là 0.8733, giá trị hàm loss ở epoch 20 giảm xuống còn 0.0938.
* Độ chính xác tăng dần qua từng epoch, cao nhất là 99,58%.

## **4.2. Đưa hình ảnh cần nhận diện**

Với mỗi loại rau củ , chúng ta sẽ kiểm thử với 10 ảnh ngẫu nhiên bất kì. Sau đây là kết quả thực nghiệm sau quá trình huấn luyện :

* Với hình ảnh quả cà chua

A screenshot of a computer program

Description automatically generated

* Cà rốt

A screenshot of a computer program

Description automatically generated

* Dưa chuột

A screenshot of a computer program

Description automatically generated

Trường hợp chọn các ảnh rau củ ngẫu nhiên

A screenshot of a computer program

Description automatically generated

# **KẾT LUẬN**

Trong báo cáo này, chúng em đã đề xuất một mô hình dựa trên mạng nơron tích chập (CNN) để nhận dạng rau củ quả. Mô hình này có 3 lớp nơron tích chập (Convolution) và 2 lớp nơron liên kết đầy đủ (Fully Connected), tổng số tham số là khoảng hơn 6 triệu.   
  
Như vậy, có thể khẳng định mô hình có độ phức tạp ở mức vừa phải, phù hợp với các hệ thống xử lý ở mức trung bình và đem lại tiềm năng khả thi trong ứng dụng thực tiễn. Hiện nay do điều kiện tính toán nên chỉ áp dụng số lần huấn luyện còn thấp, nếu được huấn luyện ở mức độ sâu hơn thì kỳ vọng sẽ đem lại kết quả cao hơn nữa.

Để phát triển thêm cho mô hình, nhóm em sẽ tìm hiểu và thiết kế một hệ thống thu thập dữ liệu hình ảnh để tạo bộ dữ liệu huấn luyện đa dạng cho mô hình, từ đó xây dựng một ứng dụng cho bài toán thực tiễn ví dụ như : mô hình có thể được tích hợp vào các hệ thống nông nghiệp thông minh để giúp nhận biết các loại cỏ dại, sâu bệnh hoặc những vấn đề khác gặp phải trong nông nghiệp hay tích hợp vào các ứng dụng thương mại điện tử hoặc cửa hàng trực tuyến để giúp người mua nhận diện và chọn lựa các sản phẩm rau củ quả một cách thuận tiện và chính xác,….

TÀI LIỆU THAM KHẢO

* PGS. TS Trương Ngọc Sơn, *Trí tuệ nhân tạo cơ sở và ứng dụng,* Đại học quốc gia TPHCM, 2020.
* Q. Zhang, M. Zhang, T. Chen, Z. Sun, Y. Ma, and B. Yu, “Recent advances in convolutional neural network acceleration,” *Neurocomputing*, vol. 323, pp. 0–38, 2019, doi: 10.1016/j.neucom.2018.09.038.
* Andrej Karpathy. CS231n Convolutional Neural Networks for Visual Recognition - Image Classification. http://cs231n.github.io/classification/
* Dumoulin, V., & Visin, F. (2016). A guide to convolution arithmetic for deep learning.
* Andrej Karpathy. (n.d.). CS231n Convolutional Neural Networks for Visual Recognition - Visualizing what ConvNets learn. http://cs231n.github.io/understanding-cnn/