BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC CẦN THƠ**

**TRƯỜNG CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG**

**🙞 🕮 🙜**



**NIÊN LUẬN NGÀNH**

**NGÀNH KHOA HỌC MÁY TÍNH**

**Đề tài**

**XÂY DỰNG ỨNG DỤNG**

**NHẬN DẠNG CÂY THUỐC NAM**

**Sinh viên thực hiện: Nguyễn Khắc Duy**

**Mã số: B2017027**

**Khóa: 46**

Cần Thơ, 04/2024

BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC CẦN THƠ**

**KHOA CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG**

**🙞 🕮 🙜**



**NIÊN LUẬN CƠ SỞ NGÀNH**

**NGÀNH KHOA HỌC MÁY TÍNH**

**Đề tài**

**XÂY DỰNG ỨNG DỤNG**

**NHẬN DẠNG CÂY THUỐC NAM**

**Giáo viên hướng dẫn: Sinh viên thực hiện:**

**TS.Lưu Tiến Đạo Nguyễn Khắc Duy**

**Mã số: B2017027**

**Khóa: 46**

Cần Thơ, 04/2024

**NHẬN XÉT CỦA GIẢNG VIÊN**

**--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------**

**LỜI CẢM ƠN**

Để có được bài niên luận này, em xin được bày tỏ lòng biết ơn chân thành và sâu sắc đến Thầy Lưu Tiến Đạo – người đã trực tiếp tận tình hướng dẫn,giúp đỡ em. Trong suốt quá trình thực hiện niên luận, nhờ những sự chỉ bảo và hướng dẫn quý giá đó mà bài niên luận này được hoàn thành một cách tốt nhất.

Em cũng xin gửi lời cám ơn chân thành đến các Thầy Cô Giảng viên Đại học Cần Thơ, đặc biệt là các Thầy Cô ở Trường CNTT & TT, những người đã truyền đạt những kiến thức quý báu trong thời gian qua.

Em cũng xin chân thành cảm ơn bạn bè cùng với gia đình đã luôn động viên, khích lệ và tạo điều kiện giúp đỡ trong suốt quá trình thực hiện để em có thể hoàn thành bài niên luận một cách tốt nhất.

Tuy có nhiều cố gắng trong quá trình thực hiện niên luận, nhưng không thể tránh khỏi những sai sót. Em rất mong nhận được sự đóng góp ý kiến quý báu của quý Thầy và các bạn để bài niên luận hoàn thiện hơn.

|  |  |
| --- | --- |
|  | Cần Thơ, ngày 03 tháng 04năm 2024  Người viết  Nguyễn Khắc Duy |

**MỤC LỤC**

[PHẦN GIỚI THIỆU 9](#_Toc18325)

[1. Đặt vấn đề 9](#_Toc20360)

[2. Lịch sử giải quyết vấn đề 9](#_Toc14450)

[3. Mục tiêu đề tài 10](#_Toc14939)

[4. Đối tượng và phạm vi nghiên cứu 10](#_Toc474)

[5. Phương pháp nghiên cứu 10](#_Toc31258)

[6. Kết quả đạt được 11](#_Toc16845)

[7. Bố cục niên luận 11](#_Toc17231)

[PHẦN NỘI DUNG 12](#_Toc14121)

[CHƯƠNG 1 12](#_Toc11245)

[NỘI DUNG LÝ THUYẾT 12](#_Toc28765)

[I. Cơ sở khoa học 12](#_Toc16327)

[1. Tìm hiểu về Deep Learning 12](#_Toc11400)

[1.1. Lịch sử hình thành 12](#_Toc438)

[1.2. Mạng nơ-ron nhân tạo 13](#_Toc27279)

[2. Sử dụng Deep Learning trong nhận dạng hình ảnh 20](#_Toc1737)

[2.1. Biểu diễn hình ảnh trong máy tính 20](#_Toc15865)

[2.2. Tìm hiểu về mạng nơ-ron tích chập (CNNs) 22](#_Toc1090)

[2.3. Hàm lỗi 27](#_Toc32653)

[3. Phương pháp Transfer Learning 28](#_Toc11818)

[3.1. Pre-Trained Model 28](#_Toc24585)

[3.2. Mô hình ResNet50 29](#_Toc29626)

[3.3. Mô hình MobileNetV2 30](#_Toc21869)

[4. Thư viện Tensorflow 32](#_Toc20469)

[II. Mô tả chi tiết bài toán 34](#_Toc24486)

[1. Mô hình nhận dạng cây thuốc Nam 34](#_Toc22565)

[2. Xây dựng ứng dụng di động với Flutter 34](#_Toc18343)

[CHƯƠNG 2 XÂY DỰNG MÔ HÌNH VÀ ỨNG DỤNG NHẬN DẠNG CÂY THUỐC NAM 36](#_Toc16145)

[I. Xây dựng mô hình 36](#_Toc21553)

[1. Quy trình xây dựng mô hình 36](#_Toc31534)

[2. Thu thập dữ liệu 36](#_Toc2509)

[3. Tiền xử lý dữ liệu 38](#_Toc28445)

[4. Huấn luyện mô hình 39](#_Toc30676)

[4.1. Mô hình ResNet50 39](#_Toc12071)

[4.2. Mô hình MobileNetV2 43](#_Toc25206)

[5. Đánh giá kết quả 46](#_Toc5023)

[II. Xây dựng ứng dụng nhận dạng cây thuốc Nam 47](#_Toc23062)

[1. Sơ đồ hoạt động của ứng dụng 47](#_Toc1871)

[2. Thiết kế cơ sở dữ liệu 47](#_Toc4732)

[3. Ứng dụng 49](#_Toc14942)

[3.1. Chức năng tìm kiếm 50](#_Toc8364)

[3.2. Chức năng dự đoán 51](#_Toc25936)

[CHƯƠNG 3 55](#_Toc15463)

[KIỂM THỬ VÀ ĐÁNH GIÁ => giới thiệu demo 55](#_Toc25868)

[1. Kết quả kiểm tra 55](#_Toc19090)

[PHẦN KẾT LUẬN 56](#_Toc12)

[1. Kết quả đạt được 56](#_Toc24616)

[2. Hướng phát triển 56](#_Toc26152)

[TÀI LIỆU THAM KHẢO 57](#_Toc10587)

**DANH MỤC HÌNH**

[Hình 1 .1: Mô hình perceptron. 14](#_Toc3485)

**DANH MỤC BẢNG**

**DANH MỤC CÁC TỪ VIẾT TẮT**

|  |  |  |
| --- | --- | --- |
| **STT** | **Từ viết tắt** | **Diễn giải** |
|  | AI | Artificial Intelligence |
|  | Sắp xếp theo ABC |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**TÓM TẮT**

Việt Nam là quốc gia có truyền thống lâu đời trong sử dụng cây cỏ trong việc phòng và trị bệnh. Nắm bắt được sự cần thiết và giá trị dược liệu lớn của cây thuốc nam trong y học dân tộc, việc xây dựng một ứng dụng nhận dạng cây thuốc nam có thể giúp người dùng có thể nhận biết và sử dụng các loại cây thuốc một cách chính xác và hiệu quả.

Nghiên cứu này sẽ tập trung trong việc thu thập dữ liệu hình ảnh các loại cây thuốc nam, xây dựng mô hình phân loại bằng phương pháp học sâu và cuối cùng là phát triển một ứng dụng di động có khả năng nhận dạng và phân loại các loại cây thuốc một cách chính xác.

Kết quả của nghiên cứu này sẽ cung cấp một công cụ hữu ích cho người dùng có thể nhận dạng và tra cứu thông tin cây thuốc nam để có phương pháp sử dụng hiệu quả.

**ABSTRACT**

Vietnam is a country with a long-standing tradition of using herbs and plants for prevention and treatment of diseases. Recognizing the necessity and immense value of herbal medicine in traditional healthcare, the development of a medicinal plant recognition application can assist users in accurately and effectively identifying and utilizing various medicinal plants.

This research focuses on gathering image data of medicinal plants, constructing classification models using deep learning methods, and ultimately developing a mobile application capable of accurately identifying and classifying different types of medicinal plants.

The outcomes of this study will provide a useful tool for users to identify and look up information on medicinal plants, enabling them to utilize them effectively.

# PHẦN GIỚI THIỆU

## 1. Đặt vấn đề

Việt Nam là một quốc gia có nguồn tại nguyên cây dược liệu rất lớn, nhiều loài đặc hữu quý hiếm, bên cạnh đó dược phẩm còn một trong các ngành kinh tế quan trọng của Việt Nam, bởi nó góp phần giúp người dân, người lao động được đảm bảo sức khỏe, từ đó hỗ trợ cho các ngành, lĩnh vực khác thực hiện tốt các sứ mệnh của mình.

Với vị trí địa lý nằm trong khu vực Indo - Burma có khí hậu nhiệt đới cận xích đạo, là 1 trong 25 điểm nóng về đa dạng sinh học toàn cầu. Đa dạng sinh học ở Việt Nam đứng thứ 16 trên thế giới với trên 36.000 loài thực vật, động vật, vi tảo, vi sinh vật biển trong đó có nhiều loài quý hiếm. Với chiều dài lịch sử hơn 4.000 năm hình thành và phát triển của cộng đồng các dân tộc, đã kết tinh nhiều giá trị về văn hóa, kinh tế, xã hội. Nền y dược học cổ truyền, dược liệu được hình thành và phát triển cho tới ngày nay, trong số 13.766 loài thực vật được ghi nhận ở Việt Nam, đã ghi nhận sử dụng 5.117 loài thuộc 1.823 chi của 362 họ thực vật có giá trị làm thuốc chữa bệnh và chăm sóc, bảo vệ sức khỏe nhân dân.

Tuy nhiên với sự đa dạng về loài cũng như tên gọi từng vùng miền làm cho việc nhận biết các loại cây thuốc nam gặp không ít khó khăn đối với những người không chuyên, đặc biệt là khi phải xác định chúng từ các tài liệu hoặc nguồn thông tin trực tuyến không đáng tin cậy.

Nắm bắt được những thông tin trên nên việc có một ứng dụng nhận dạng cây thuốc nam là hết sức cần thiết và có thể đem lại nhiều lợi ích. Và đó cũng là lý do cho việc xây dựng ứng dụng “**Nhận dạng cây thuốc nam với Deep Learning**” trong nghiên cứu này.

## 2. Lịch sử giải quyết vấn đề

Trên thực tế đã có rất nhiều bài nghiên cứu, đề tài trong nhận dạng hình ảnh nói chung và nhận dạng cây thuốc nam nói riêng. Trong phạm vi nghiên cứu của đề tài nhận dạng cây thuốc nam này thì đã có đề tài “Nhận dạng lá cây thuốc nam với deep learning” của anh Lâm Thanh Phương trong Luận văn Thạc sĩ Công nghệ thông tin của Trường Đại học Cần Thơ do Thầy TS. Dương Trung Nghĩa hướng dẫn. Cùng với đó là nhóm tác giả thuộc Viện Nghiên cứu Quốc tế MICA(thuộc Trường Đại học Bách Khoa Hà Nội) đã cho ra đời Ứng dụng tra cứu đa phương thức cây thuốc Việt Nam dựa trên kỹ thuật Thị giác máy tính.

Nhìn chung các đề tài đã được phát triển rất tốt và mang tính ứng dụng cao khi được phát triển trên thiết bị di động mang lại sự linh hoạt và tiện dụng trong việc tra cứu thông tin cây thuốc nam. Nhưng các đề tài chỉ dừng lại ở việc huấn luyện mô hình và nhận dạng hình ảnh trong phạm vi lá của cây thuốc.

## 3. Mục tiêu đề tài

Mục tiêu của đề tài này là mang lại một ứng dụng nhận dạng, tra cứu thông tin cây thuốc nam kế thừa từ những đề tài đã có nhưng sẽ được phát triển trong quá trình huấn luyện hình ảnh cây thuốc từ phạm vi chỉ từ lá của cây thuốc lên thành toàn bộ thân cây thuốc.

Vì lá cây chỉ là một phần nhỏ của cây và không phản ánh đầy đủ đặc điểm của cây, và trong tự nhiên lá cây có thể có các đặc điểm hình thái phụ thuộc vào điều kiện tự nhiên hoặc các giai đoạn phát triển khác nhau. Cho nên việc mở rộng phạm vi nhận dạng có thể giúp người dùng xác định chính xác loại cây cho dù chúng đang ở giai đoạn phát triển nào.

Bên cạnh việc nhận dạng cây thuốc nam, đề tài cũng mang lại một cơ sở dữ liệu phục vụ cho tra cứu thông tin cây thuốc, các thông tin này được tổng hợp từ nhiều nguồn đáng tin cậy. Từ đó mang đến cho người dùng những thông tin về dược liệu, cách dùng một cách hữu ích và chính xác nhất.

## 4. Đối tượng và phạm vi nghiên cứu

Đối tượng nghiên cứu: là 2.840 hình ảnh được phân bố cho 30 loại cây thuốc nam.

Phạm vi nghiên cứu: Nhận dạng phân loại cây thuốc nam dựa trên hình ảnh.

## Phương pháp nghiên cứu

* Về lý thuyết:
* Tìm hiểu về Deep Learning trong phân loại hình ảnh, tìm hiểu về mô hình ResNet50 và MobileNet-v2.
* Về thu thập dữ liệu: Dữ liệu hình ảnh các loại cây thuốc nam được thu thập từ Internet.
* Về thực nghiệm: Huấn luyện mô hình ResNet50 và MobileNet-v2 từ dữ liệu thu thập được từ đó đưa ra đánh giá kiểm thử và chọn ra mô hình tối ưu hơn để xây dựng ứng dụng trên di động.

## Kết quả đạt được

## Bố cục niên luận

Niên luận bao gồm có 3 phần:

**Phần giới thiệu**

Giới thiệu chung về đề tài, lịch sử giải quyết vấn đề, đối tượng và phạm vi nghiên cứu, các phương pháp nghiên cứu, mục tiêu mong muốn đạt được của đề tài.

**Phần nội dung**

**Chương 1**: Nội dung lý thuyết.

Nội dung của Chương 1 chủ yếu đề cập đến cơ sở lý thuyết của Deep Learning và sử dụng Deep Learning trong nhận dạng hình ảnh. Lý thuyết về các mô hình ResNet50 và MobileNet-v2.

**Chương 2**: Xây dựng và huấn luyện mô hình.

Chương 2 tập trung vào việc thu thập và xử lý dữ liệu hình ảnh, xây dựng mô hình và huấn luyện mô hình dựa trên hình ảnh đã có và chọn ra mô hình tối ưu.

**Chương 3**: Xây dựng ứng dụng.

Từ kết quả huấn luyện có ở Chương 2, trong Chương 3 sẽ xây dựng ứng dụng nhận dạng cây thuốc nam trên thiết bị di động.

**Phần kết luận**

Trình bày kết quả đạt được và hướng phát triển hệ thống.

# PHẦN NỘI DUNG

# CHƯƠNG 1

# NỘI DUNG LÝ THUYẾT

## Cơ sở khoa học

### Tìm hiểu về Deep Learning

Deep Learning (học sâu) là một phần trong một nhánh rộng hơn các phương pháp học máy dựa trên mạng thần kinh nhân tạo kết hợp với việc học biểu diễn đặc trưng (representation learning). Mục tiêu của phương pháp này là tạo ra các mô hình máy học có khả năng học và hiểu dữ liệu phức tạp, đặc biệt là dữ liệu không có cấu trúc hoặc dữ liệu có cấu trúc phức tạp như hình ảnh, văn bản và âm thanh.

#### Lịch sử hình thành

Được biết đến với cái tên mạng nơ-ron nhân tạo (ANN) từ những năm 1959, được đề xuất bởi David H.Hubel và Torsten Wiesel. Mạng nơ-ron nhân tạo được hai người lấy cảm hứng từ các mô hình sinh học khi tìm thấy hai loại tế bào trong vỏ não thị giác sơ cấp: các tế bào đơn giản và các tế bào phức tạp. Nghiên cứu này của hai ông được xem như những viên gạch đầu tiên đặt nền móng cho con đường hình thành Deep Learning sau này.

Nhìn chung, Deep Learning được phát triển từ các nghiên cứu về mạng nơ-ron nhân tạo. Từ khi ra đời, các kiến trúc học sâu được xây dựng và phát triển liên tục từ mạng này, tiêu biểu là mô hình mạng nơ-ron có cấp bậc Neocognition của Kunihiko Fukushima ra mắt năm 1979 dùng trong nhận dạng ký tự viết tay của Nhật Bản. Mô hình này là ý tưởng để phát triển về một trong những kiến trúc mô hình phổ biến của mạng nơ-ron sau này đó là mạng nơ-ron tích chập (CNNs). Thời điểm đó, một trong những thách thức là làm thế nào để huấn luyện mô hình này với nhiều lớp hơn. Từ thách thức đó nên năm 1989, một nghiên cứu quan trọng của Yann Le Cun cùng các cộng sự đã cho ra đời một mạng nơ-ron tích chập áp dụng thuật toán lan truyền ngược (back-propagation) để để áp dụng vào vấn đề phân loại chữ số viết tay. Mạng này được đặt tên là LeNet và đã được Cục Bưu điện Hoa Kỳ sử dụng trong những năm1990 để tự động hóa việc đọc các mã ZIP trên phong bì thư.

Khoảng năm 2010, mặc dù mạng nơ-ron gần như bị quên lãng hoàn toàn bởi cộng đồng khoa học do sự phát triển bởi các phương pháp như Support Vector Machines (SVM), Decision Trees,… một số người vẫn kiên trì nghiên cứu với mạng nơ-ron đã đạt được những thành tựu đột phá.

Năm 2011, Dan Ciresan từ IDSIA chiến thắng các cuộc thi phân loại hình ảnh với các mạng nơ-ron sâu được huấn luyện bằng GPU. Nhưng thời điểm quan trọng nhất đến vào năm 2012, với sự tham gia của nhóm của Hinton trong thách thức phân loại hình ảnh quy mô lớn hàng năm ImageNet (ImageNet Large Scale Visual Recognition Challenge – ILSVRC). Thách thức ImageNet lúc đó được biết đến với độ khó cao, bao gồm việc phân loại hình ảnh màu có độ phân giải cao thành 1.000 danh mục khác nhau sau khi được huấn luyện trên 1,4 triệu hình ảnh. Năm 2011, độ chính xác top-5 của mô hình chiến thắng, dựa trên các phương pháp cổ điển của thị giác máy tính, chỉ là 74,3%.

Sau đó, vào năm 2012, một đội dẫn đầu bởi Alex Krizhevsky và được tư vấn bởi Geoffrey Hinton đã đạt được độ chính xác top-5 là 83,6% – một đột phá đáng kể. Từ đó, mỗi năm cuộc thi đã bị chi phối bởi các mạng nơ-ron tích chập sâu. Đến năm 2015, người chiến thắng đạt được độ chính xác 96,4%, và nhiệm vụ phân loại trên ImageNet được coi là một vấn đề được giải quyết hoàn toàn.

Từ năm 2012 trở đi, mạng nơ-ron tích chập sâu (CNNs) đã trở thành thuật toán chính cho tất cả các nhiệm vụ thị giác máy tính. Đồng thời, học sâu cũng đã được ứng dụng vào nhiều loại vấn đề khác nhau, như xử lý ngôn ngữ tự nhiên. Nó đã hoàn toàn thay thế SVMs và cây quyết định trong nhiều ứng dụng.

#### Mạng nơ-ron nhân tạo

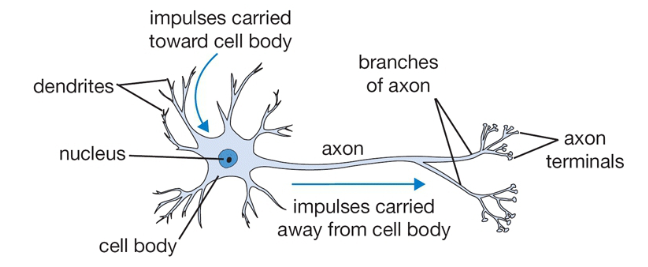
Mạng nơ-ron nhân tạo được chia thành nhiều loại khác nhau, sử dụng cho những mục đích khác nhau. Có thể kể đến như:

* Mạng Perceptron.
* Mạng nơ-ron truyền thẳng.
* Mạng nơ-ron tích chập.
* Mạng nơ-ron hồi quy.

Trong phạm vi nghiên cứu lần này sẽ tìm hiểu về phiên bản có cấu trúc nguyên thủy nhất của mạng nơ-ron nhân tạo đó là mạng Perceptron. Từ đó sẽ mở rộng sang tìm hiểu mạng nơ-ron tích chập trong xử lý hình ảnh.

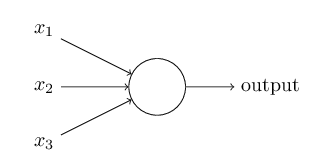
##### Perceptron

Một mạng nơ-ron được cấu thành từ các nơ-ron đơn lẻ được gọi là các perceptron, được lấy cảm hứng từ nơ-ron sinh học như mô tả dưới đây:



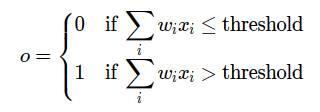
Hình 1. 1: Minh hoạt mạng nơ-ron sinh học.

Như hình trên có thể thấy trong một nơ-ron sinh học có thể nhận nhiều đầu vào và cho ra một kết quả duy nhất, nên mô hình của một perceptron cũng tương tự như vậy:

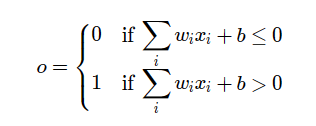


Hình 1.2: Mô hình perceptron.

Một perceptron sẽ nhận một hoặc nhiều đầu vào X dạng nhị phân và cho ra một kết quả O duy nhất dạng nhị phân. Các đầu vào có tầm ảnh hưởng được quy định bởi các tham số trọng lượng W tương ứng, còn kết quả đầu ra được xác định bằng một ngưỡng quyết định *b* nào đó:

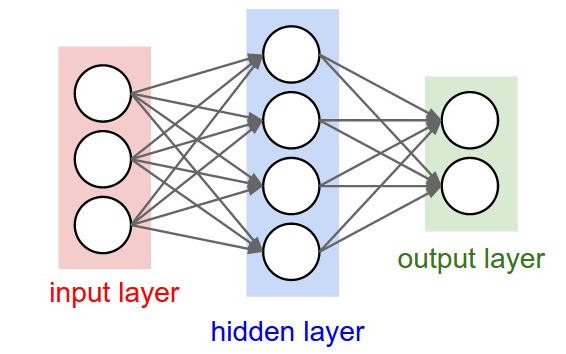


Đặt *b* = - threshold, ta có thể viết lại thành:



##### Kiến trúc mạng nơ-ron:

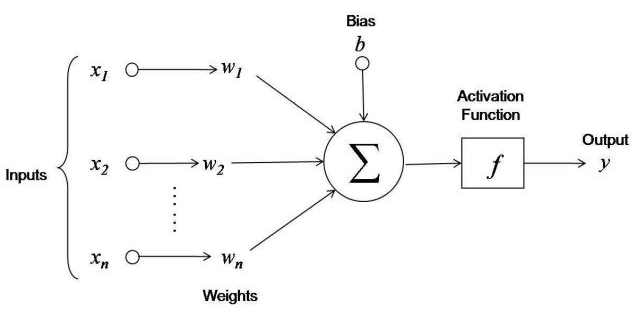
Thực tế mạng nơ-ron được cấu thành từ rất nhiều perceptron như thế, các perceptron liên kết với nhau thành các tầng perceptron hay còn được gọi là perceptron đa tầng:



Một mạng nơ-ron sẽ có 3 kiều tầng:

* Tầng đầu vào (input layer): Thông tin cần được xử lý được đưa vào lớp đầu vào. Nút đầu vào tiếp nhận, phân loại, phân tích dữ liệu và sau đó chuyển dữ liệu sang lớp tiếp theo.
* Tầng ẩn (hidden layer): Dữ liệu chuyển từ lớp đầu vào sang lớp ẩn, hoặc từ lớp ẩn này sang lớp ẩn khác. Mạng nơ-ron nhân tạo có thể có một số lượng lớn lớp ẩn, mỗi lớp ẩn phân tích dữ liệu đầu ra từ lớp trước, xử lý dữ liệu đó sâu hơn rồi chuyển sang lớp tiếp theo. Mạng nơ-ron nhiều lớp ẩn như thế còn được gọi là mạng nơ-ron sâu (DNN - Deep Neural Network).
* Tầng đầu ra (output layer): Lớp đầu ra cho kết quả cuối cùng của tất cả dữ liệu được xử lý trước đó, lớp này có thể có một hoặc nhiều nút. Đối với bài toán phân loại nhị phân sẽ có một nút đầu ra, nút này sẽ cho kết quả 0 hoặc 1. Đối với bài toán phân loại nhiều lớp sẽ có sốt nút bằng với số lớp cần phân loại.

Các tầng được kết nối với nhau theo kiểu đầu ra của nút ở tầng hiện tại sẽ là một trong những đầu vào của nút ở tầng tiếp theo. Giá trị của một nút được tính như sau:



Hình 1. 3: Tính toán trong một nơ-ron.

Để tính giá trị của một nơ-ron ta có công thức sau:

y

Trong đó:

* y là kết quả của nơ-ron.
* wi là trọng số của nơ-ron đầu vào.
* xi là nơ-ron đầu vào.
* b là độ lệch bias.
* f là hàm kích hoạt.

##### Hàm kích hoạt

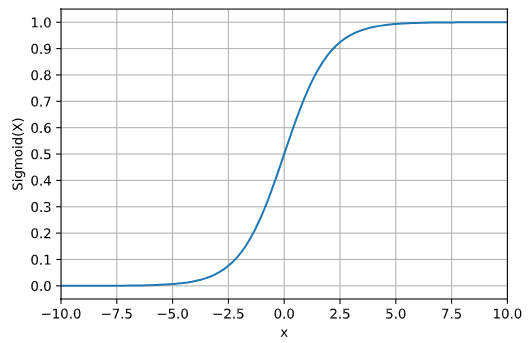
Trong mạng nơ-ron, hàm kích hoạt đóng vai trò quan trọng trong việc biểu diễn đầu ra của mỗi nơ-ron. Cụ thể, hàm kích hoạt thường được áp dụng sau khi tổng trọng số đầu vào được tính toán thông qua một phép toán tuyến tính. Khi đó hàm kích hoạt được sử dụng để chuyển đổi đầu ra của mạng thành dạng xác suất hoặc thang đo chuẩn giúp trong việc giải quyết các vấn đề phân loại.

Một số hàm kích hoạt phổ biến được sử dụng trong mạng nơ-ron như hàm Sigmoid, hàm Tanh, hàm ReLU, hàm Softmax,… Việc lựa chọn đúng hàm kích hoạt là rất quan trọng để huấn luyện mạng nơ-ron có khả năng dự đoán tốt.

* **Hàm Sigmoid**

Có công thức:

Đồ thị của hàm như sau:



Hình 1. 4: Đồ thị hàm Sigmoid.

Hàm sigmoid nhận đầu vào là một số thực và chuyển thành một giá trị trong khoảng (0;1). Nếu đầu vào là một số thực âm rất nhỏ sẽ cho đầu ra tiệm cận với 0, ngược lại nếu là một số thực dương lớn thì đầu ra sẽ tiệm cận với 1.  
 Nhưng có một số nhược điểm khiến cho hàm sigmoid dần ít được sử dụng hơn đó là vấn đề mất mát gradient (vanishing gradient). Vì mạng nơ-ron sử dụng lan truyền ngược để cập nhật trọng số trong quá trình học. Gradient của hàm mất mát (loss function) đối với các trọng số được lan truyền từ lớp đầu ra trở về lớp đầu vào để cập nhật. Tuy nhiên một số trường hợp, đạo hàm của hàm kích hoạt sigmoid có giá trị rất nhỏ (gần bằng 0) ở một số điểm dữ liệu đầu vào. Khi các gradient này nhỏ, các trọng số không được cập nhật một cách hiệu quả, dẫn đến việc mạng không học được một mô hình tốt.

* **Hàm softmax:**

Hàm softmax thường được sử dụng trong các lớp đầu ra của mạng nơ-ron, đặc biệt là trong bài toán phân loại nhiều lớp. Hàm softmax chuyển đổi đầu vào thành một phân phối xác suất nằm trong khoảng (0;1] sao cho các đầu ra có tổng bằng 1. Giả sử nếu ta có *n* đầu ra thì sẽ được định nghĩa như sau:

, với

Ví dụ:

softmax

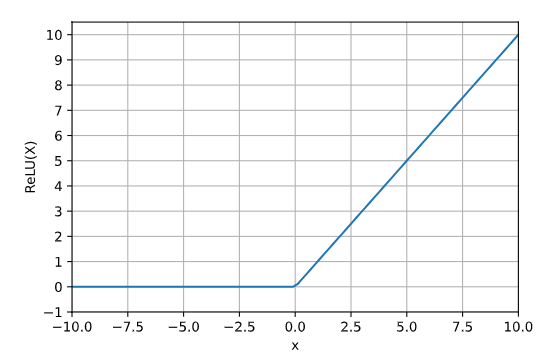
Việc chuyển đổi các giá trị đầu ra thành phân phối xác suất giúp hiểu được mức độ chắc chắn của dự đoán cho mỗi lớp, giúp thuận tiện trong việc so sánh và giải thích kết quả.

* **Hàm ReLU:**

Hàm kích hoạt ReLU (Rectified Linear Unit) là một trong những hàm kích hoạt phổ biến nhất được sử dụng trong mô hình mạng nơ-ron, đặc biệt là sử dụng trong các lớp ẩn. Hàm ReLU có công thức như sau:

ReLU(x) = max(0,x)

Đồ thị biểu diễn hàm ReLU như sau:



Hình 1. 5: Đồ thị hàm ReLU.

Hàm ReLU đơn giản chỉ là lọc các giá trị 0 từ đầu vào. Có nghĩa là đầu ra của hàm ReLU sẽ bằng 0 nếu đầu vào mang giá trị âm, còn nếu đầu vào không âm thì đầu ra sẽ là chính nó. Và hàm kích hoạt này cũng khắc phục vấn đề mất gradient ở hàm kích hoạt sigmoid.

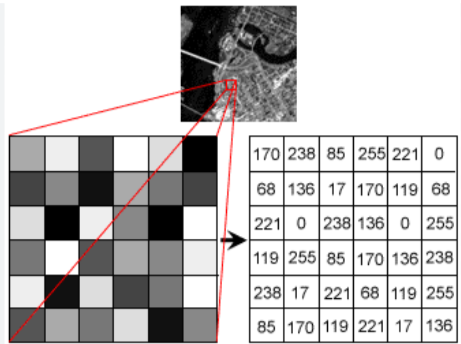
### Sử dụng Deep Learning trong nhận dạng hình ảnh

Như đã đề cập ở trên, quá trình phát triển của Deep Learning đã cho ra đời các mô hình phổ biến như CNNs, RNNs, LSTM,… và trong số các mô hình đó, CNNs được coi là nhân tốt chủ chốt tác động lớn đến việc biến Deep Learning trở nên phổ biến và mang tính ứng dụng cao trong thực tế với khả năng xử lý hình ảnh mạnh mẽ của mình.

#### Biểu diễn hình ảnh trong máy tính

Một hình ảnh được biểu diễn bằng kích thước (chiều cao và chiều rộng) dựa trên số lượng pixel, và pixel đóng vai trò như một đơn vị cốt lõi để tạo thành hình ảnh. Ví dụ, nếu kích thước của hình ảnh là 500 x 400 (chiều rộng x chiều cao) thì tổng số pixel trong hình ảnh là 200.000. Mỗi pixel đại diện cho một thông tin quan trọng về sắc độ và độ sáng tạo thành sự thể hiện trực quan. Có các loại biểu diễn trực quan hình ảnh như: Ảnh mức xám, RGB, RGBA,… .

* **Ảnh mức xám:**

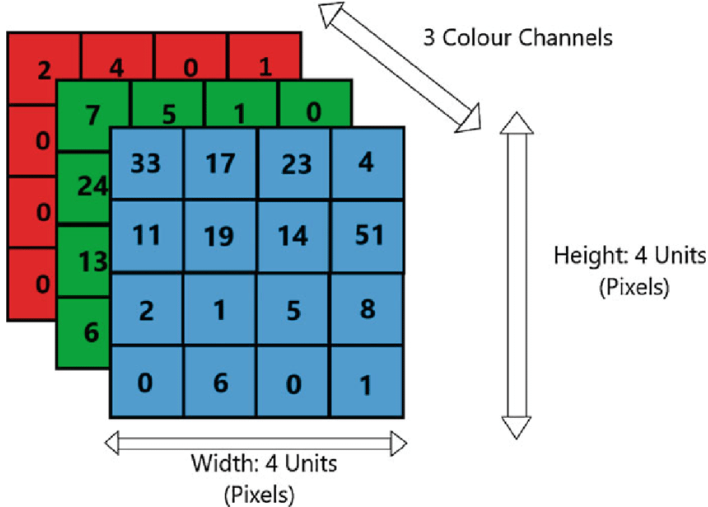


Hình 1. 6: Minh hoạt cấu trúc ảnh mức xám.

Trong hình ảnh dạng mức xám, mỗi pixel chỉ chưa một giá trị duy nhất biểu diễn mức độ sáng tương ứng với mức độ xám của điểm ảnh đó. Thông thường, giá trị này được biểu diễn dưới dạng số nguyên từ 0 đến 255, trong đó 0 đại diện cho màu đen tuyệt đối và 255 đại diện cho màu trắng tuyệt đối.

* **Ảnh RGB:**

Ảnh RGB (Red, Green, Blue), mỗi điểm ảnh được biểu diễn bằng một vector có 3 thành phần bao gồm: giá trị mức độ của màu đỏ (R), màu xanh lá cây (G), màu xanh dương (B). Hình ảnh được biểu diễn dưới dạng một ma trận ba chiều, trong đó kích thước của 2 chiều đầu tiên là chiều rộng và chiều cao của hình ảnh và chiều thứ ba sẽ là 3, biểu thị cho các thành phần màu RGB. Ví dụ một hình ảnh có kích thước 100x100 pixel thì sẽ được biểu dinex bằng một ma trận có kích thước 100x100x3.



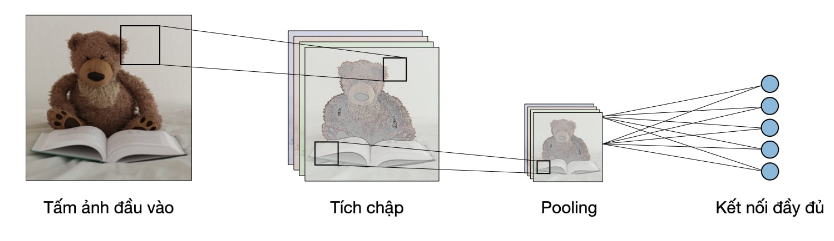
Hình 1. 7: Minh hoạt cấu trúc ảnh RGB.

Từ hình ảnh màu ban đầu, các điểm màu Red, Green, Blue được tách riêng ra thành từng ma trận biểu diễn cường độ màu sắc của từng kênh màu.

#### Tìm hiểu về mạng nơ-ron tích chập (CNNs)

Mạng nơ-ron tích chập (Convolutional Neural Network - CNNs), là một loại mạng nơ-ron nhân tạo, hiệu quả trong các tác vụ hình ảnh khác nhau, bao gồm phân loại hình ảnh, phân đoạn hình ảnh, phát hiện đối tượng,… .

Kiến trúc truyền thống của một mạng CNN được cấu thành từ những lớp như sau:



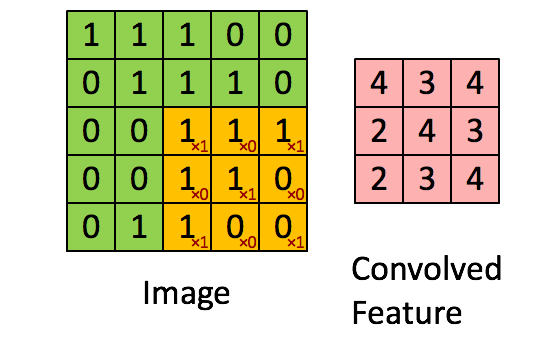
Hình 1. 8: Mô hình mạng nơ-ron tích chập (CNNs).

Một mạng CNN cơ bản được cấu thành từ các lớp như:

* Lớp tích chập (Convolution), viết tắt là Conv.
* Lớp Pooling.
* Lớp kết nối đầy đủ (Fully Connected, viết tắt là FC.

##### Lớp tích chập (ConV)

Lớp tích chập là một phần quan trọng của CNN chịu trách nhiệm trong việc phát hiện đặc trưng trong dữ liệu. Lớp tích chập sử dụng các bộ lọc để quét qua dữ liệu đầu vào và tính toán độ tương đồng giữa từng phần tử trong bộ lọc và dữ liệu.



Hình 1. 9: Phép toán tích chập.

Trong hình ảnh minh họa trên, ta đang có một ma trận ảnh đầu vào có kích thước 5x5 và một bộ lọc có kích thước 3x3, bộ lọc này sẽ di chuyển từng bước một trên ma trận ảnh 5x5 theo thứ tự từ trái qua phải và từ trên xuống dưới hay còn gọi là bước trược (stride). Sau khi thực hiện phép nhân tích chập trên một lần dịch chuyển sẽ cho ra một giá trị số tương ứng. Kết quả đầu ra của phép tích chập này là một ma trận 2-D có kích thước 3x3.

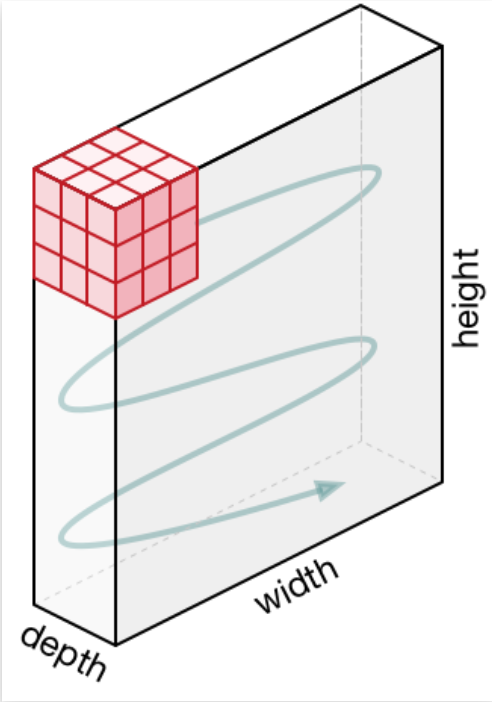
Ví dụ cho vị trí Output [0][0] được tính như sau:

4 = (1\*1) + (1\*0) + (1\*1) + (0\*0) + (1\*1) + (1\*0) + (0\*1) + (0\*0) + (1\*1)

Sau đó bộ lọc sẽ tiếp tục dịch chuyển sang trái một bước, có nghĩa là sẽ bỏ cột đầu tiên và thực hiện phép tích chập tiếp theo cho vị trí Output [0][1]:

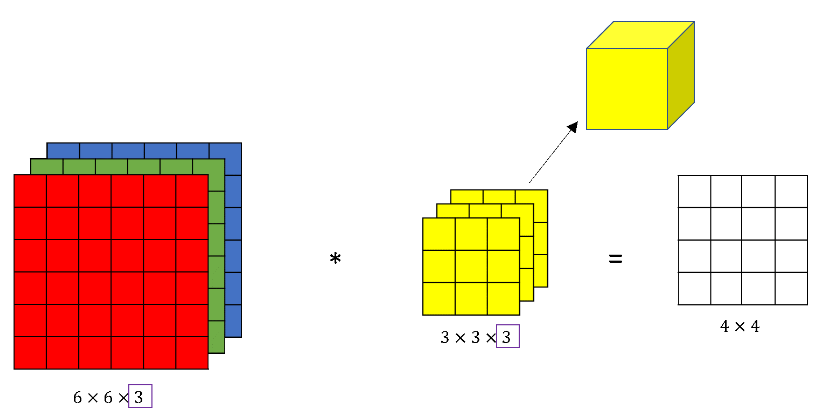
3 = (1\*1) + (1\*0) + (0\*1) + (1\*0) + (1\*1) + (1\*0) + (0\*1) + (1\*0) + (1\*1)

Trong ví dụ trên, quá trình tính toán được thực hiện trên đầu vào là một ảnh xám, nên số kênh màu của ma trận chỉ là một. Trong thực tế đa số bài toán sẽ được tính toán trên ảnh màu cho nên số kênh màu của những hình ảnh màu sẽ có 3 channels đó là: red, green, blue. Vì vậy hình ảnh màu sẽ được biểu diễn dưới dạng tensor 3 chiều có kích thước k\*k\*3.



Hình 1. 10: Tích chập trên ảnh RGB.

Quá trình tích chập trên ảnh màu cũng làm tương tự như trên ảnh xám với mỗi kênh màu sẽ có một filter riêng. Sau mỗi lần di chuyển khối filter trên 3 kênh màu, giá trị của 3 phép tích chập đó sẽ được tính tổng với nhau và cộng thêm một bias để cho ra output với vị trí tương ứng.



Hình 1. 11: Minh hoạt tích chập trên ma trận ảnh RGB.

.

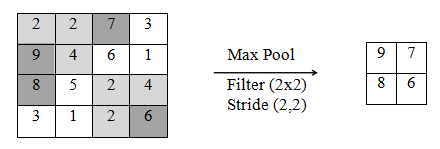
Trong mạng CNN, mỗi lớp tích chập có thể có một hoặc nhiều bộ lọc để trích xuất được các đặc trưng khác nhau của ảnh. Kích thước ma trận đầu ra (feature map) phụ thuộc vào kích thước ma trận đầu vào (input\_shape), kích thước bộ lọc (filter\_size) và bước nhảy (stride) của bộ lọc. Các siêu tham số này có thể được điều chỉnh phụ thuộc vào mục đích trích xuất đặc trưng ảnh.

Đầu ra của lớp tích chập sẽ đi qua hàm kích hoạt (activation function) trước khi trở thành đầu vào của lớp tích chập tiếp theo.

##### Lớp Pooling

Lớp Pooling (lớp gộp) hay còn gọi là downsampling (lấy mẫu giảm) thường được sử dụng ngay sau lớp Conv để đơn giản hóa thông tin đầu ra bằng cách giảm kích thước, giảm số lượng tham số đầu vào từ đó có thể giảm overfitting. Tương tự như lớp tích chập, lớp Pooling này sẽ quét bộ lọc trên toàn bộ ma trận đầu vào, nhưng điểm khác biệt là bộ lọc này không có bất kỳ trọng số nào. Có nhiều cách tính toán ở lớp Pooling như gộp theo giá trị lớn nhất (Max Pooling), gộp theo giá trị trung bình (Average Pooling), gộp theo giá trị toàn cục (Global Pooling). Trong đó Max Pooling và Average Pooling là hai phương pháp được sử dụng phổ biến nhất.

* Max Pooling: phương pháp gộp này thực hiện bằng cách lấy giá trị lớn nhất trong khu vực mà bộ lọc đi qua:



Hình 1. 12: Lấy mẫu giảm với Max Pooling.

* Average Pooling: mỗi phần tử của giá trị đầu ra được tính toán bằng cách tính trung bình các giá trị trong khu vực mà bộ lọc đi qua:

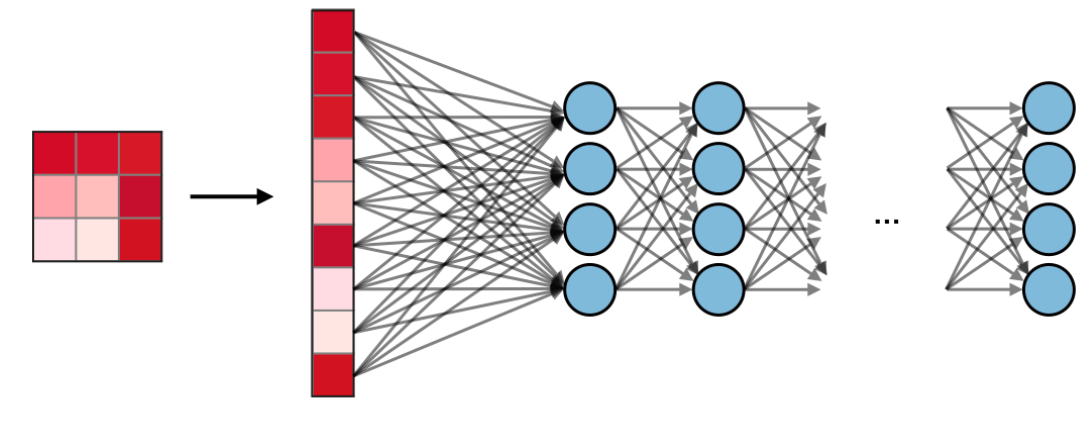


Hình 1. 13: Lấy mẫu giảm bằng Average Pooling.

Phương pháp Max Pooling thường phổ biến hơn so với Average Pooling vì nó giữ lại thông tin quan trọng nhất từ mỗi vùng pool giúp tăng cường tính nổi bật của đặc trưng. Trong khi Average Pooling có thể làm giảm thông tin quan trọng vì nó không phân biệt được các giá trị lớn và nhỏ, nhưng thường tạo ra một biểu diễn mượt mà và ít bị nhiễu hơn.

##### Lớp kết nối đầy đủ (Fully Connected)

Lớp kết nối đầy đủ (FC) nhận vào các dữ liệu đã được làm phẳng trước đó nhờ vào lớp Flatten, mỗi đầu vào đó được kết nối đến tất cả các nơ-ron. Trong mô hình mạng CNNs, các lớp kết nối đầy đủ thường được tìm thấy ở cuối mạng. Lớp này thực hiện nhiệm vụ phân loại dựa trên các đặc điểm được trích xuất qua các lớp trước và các bộ lọc khác nhau của chúng.



Hình 1. 14: Minh hoạt lớp kết nối đầy đủ.

* Biểu diễn:
* Các nơ-ron trong lớp kết nối đầy đủ thường được kích hoạt bằng hàm kích hoạt phi tuyến tính như ReLU hoặc sigmoid.
* Mỗi nơ-ron thực hiện một phép tổ hợp tuyến tính của các đầu vào của nó, sau đó truyền qua hàm kích hoạt để tạo đầu ra.
* Huấn luyện:
* Trong quá trình huấn luyện, trọng số của mỗi kết nối được điều chỉnh thông qua thuật toán lan truyền ngược và thuật toán tối ưu hóa gradient descent.
* Mục tiêu là điều chỉnh trọng số sao cho mô hình có thể dự đoán đầu ra chính xác từ các đầu vào đã cho.
* Đầu ra:
* Đầu ra của lớp kết nối đầy đủ thường là một vector có kích thước tương ứng với số lượng lớp đầu ra.
* Thông thường, một hàm softmax được sử dụng để chuyển đổi các giá trị đầu ra thành xác suất tương ứng với mỗi lớp.

#### Hàm lỗi

Hàm lỗi hay còn gọi là hàm mất mát (loss function) là hàm dùng để so sánh giá trị dự đoán với giá trị thực tế. Từ đó có thể đánh giá được mức độ hiệu quả của mạng nơ-ron khi được học trên dữ liệu huấn luyện. Vì trong quá trình huấn luyện, chúng ta sử dụng phương pháp lan truyền ngược để cập nhật trọng số, mục tiêu để cho mô hình tốt hơn. Cụ thể là cập nhật trọng số sao cho có thể giảm bớt đi giá trị của hàm mất mát.

Có các hàm lỗi phổ biến trong lĩnh vực máy học có thể kể đến như hàm lỗi MSE, MAE,… dùng trong bài toán hồi quy. Cross-Entropy dùng trong bài toán phân loại. Và với tác vụ phân loại hình ảnh trong chủ đề lần này thì chúng ta sẽ đề cập đến hàm lỗi Cross-Entropy. Vì đây là hàm lỗi được sử dụng phổ biến trong phân loại hình ảnh. Có 2 biến thể của hàm này đó là:

* **Binary Cross-Entropy:** Dùng cho bài toán phân lớp nhị phân (hai lớp).

Trong đó:

- m: là số lượng mẫu huấn luyện.

- yi : là nhãn thực tế của mẫu thứ *i*.

- y\_hati : là nhãn dự đoán của mẫu thứ *i*.

* **Categorical Cross-Entropy:** Dùng cho bài toán phân lớp đa lớp.

Hàm lỗi này được sử dụng khi đầu ra được chuẩn hóa theo hàm kích hoạt softmax. Tại đó dự đoán của từng lớp sẽ được chuyển về theo phân phối xác suất, và hàm lỗi sẽ tính toán giá trị xác suất của dự đoán so với giá trị thực tế từ đó đưa ra kết quả phân lớp cuối cùng.

### Phương pháp Transfer Learning

Transfer learning hay còn gọi là học chuyển giao. Đây là một lĩnh vực con trong ngành học máy và trí tuệ nhân tạo với mục đích áp dụng kiến thức thu được từ một tác vụ nguồn cho một tác vụ tương tự khác. Hiểu một cách đơn giản thì một mô hình được phát triển cho một nhiệm vụ được sử dụng lại và trở thành điểm bắt đầu cho một mô hình ở nhiệm vụ thứ hai.

#### Pre-Trained Model

Pre-trained model là một mô hình máy học đã được huấn luyện sẵn trên một tập dữ liệu trước đó. Các mô hình pre-trained thường được huấn luyện trên các tập dữ liệu có kích thước lớn, ví dụ như tập dữ liệu ImageNet có hàng triệu hình ảnh với hàng ngàn nhãn khác nhau.

Sau khi được huấn luyện, mô hình pre-trained có thể được sử dụng như một bộ trích xuất đặc trưng hoặc một mô hình phân loại để giải quyết các tác vụ khác nhau. Việc sử dụng mô hình pre-trained giúp giảm thời gian và chi phí huấn luyện, đồng thời cải thiện độ chính xác của mô hình so với việc huấn luyện từ đầu trên một tập dữ liệu nhỏ hơn.

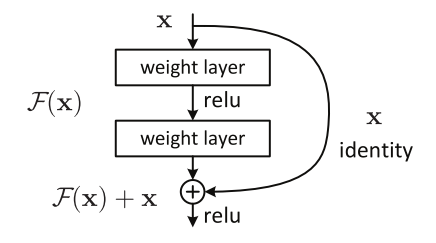
Một số mô hình pre-trained phổ biến nhất được phát triển và có thể sử dụng cho các tác vụ trong transfer learning như: VGG19, Inceptionv3, ResNet50, EfficientNet,… .

#### Mô hình ResNet50

##### Tìm hiểu về ResNet

ResNet (Residual Network) được giới thiệu đến công chúng vào năm 2015 và thậm chí đã giành được vị trí thứ 1 trong cuộc thi ILSVRC 2015 với tỉ lệ lỗi top 5 chỉ 3.57%. Không những thế nó còn đứng vị trí đầu tiên trong cuộc thi ILSVRC and COCO 2015 với ImageNet Detection, ImageNet localization, Coco detection và Coco segmentation. Hiện tại thì có rất nhiều biến thể của kiến trúc ResNet với số lớp khác nhau như ResNet-18, ResNet-34, ResNet-50, ResNet-101, ResNet-152,… . Với tên là ResNet theo sau là một số chỉ kiến trúc ResNet với số lớp nhất định.

Một vấn đề khi xây dựng mạng CNN nhiều lớp chập đó chính là xảy ra hiện tượng Vanishing Gradient. Nên mạng ResNet ra đời để giải quyết vấn đề đó. Giải pháp mà ResNet đưa ra là sử dụng kết nối tắt đồng nhất để xuyên qua một hay nhiều lớp. Một khối bị xuyên qua như vây được gọi là một residual block (hay còn gọi là một khối dư) như trong hình sau:

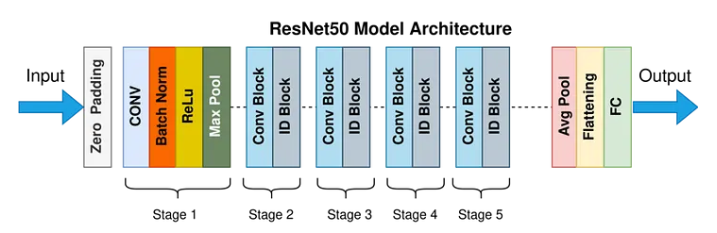


Hình 1. 15: Residual Block.

##### Tìm hiểu vè mô hình ResNet50

Mô hình ResNet50 là một dạng của ResNet với độ sâu 50 lớp. Mô hình này cho phép truy cập 1 triệu hình ảnh từ mô hình gốc là ImageNet, có thể phân loại 1000 đối tượng, bao gồm cả hình ảnh màu với kích thước 224×224 pixel. Mô hình này mang lại hiệu quả cải thiện hơn nhiều lần so với VGG19 dù mức độ phức tạp thấp.

ResNet50 bao gồm 16 khối residual block, trong đó mỗi khối bao gồm một số lớp chập và một ID Block, mỗi khối ID Block chứa 3 lớp tích chập, mỗi lớp sẽ được chuẩn hóa theo phương pháp Batch Normalization (BN). Kiến trúc cũng bao gồm các lớp gộp (Polling), lớp kết nối đầy đủ (Fully Connected) và lớp đầu ra softmax để phân loại.



Hình 1. 16: Kiến trúc mạng ResNet50.

#### Mô hình MobileNetV2

Được kế thừa phương pháp Depthwise Separable Convolutions từ mô hình MobileNet, ở phiên bản MobileNetV2 này còn đề xuất thêm hai phương pháp đó là: Linear bottlenecks và Inverted Residual Block.

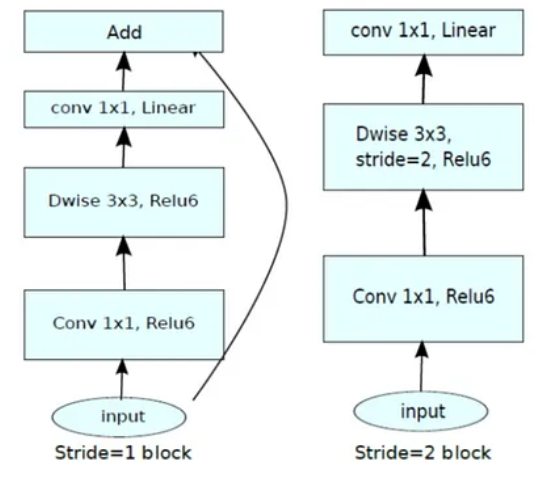
* **Inverted Residual Block:**

MobileNetV2 cũng sử dụng những kết nối tắt như ở mạng ResNet. Tuy nhiên kết nối tắt ở MobileNetV2 được điều chỉnh sao cho số kênh (hoặc chiều sâu) ở input và output của mỗi block residual được thắt hẹp lại. Chính vì thế nó được gọi là các Bottleneck Layers.

Residual block của MobileNetV2 ngược lại so với các kiến trúc residual truyền thống, vì kiến trúc residual truyền thống có số lượng kênh ở input và output của một block lớn hơn so với các layer trung gian. Chính vì vậy nó còn được gọi là Inverted residual block.

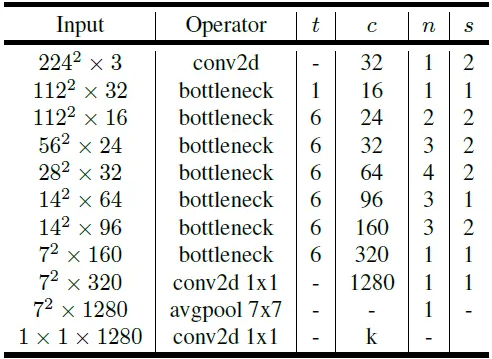
* **Depthwise Separable Convolutions:**

Trong một lớp convolutional thông thường, mỗi kernel được áp dụng cho toàn bộ các kênh đầu vào. Tuy nhiên, trong Depthwise Separable Convolutions, bước này được chia thành hai bước riêng biệt là: convolution theo chiều sâu (depthwise convolution) và sau đó là một convolution 1x1 thông thường để kết hợp các kênh.



Hình 1. 17: Kiến trúc mạng MobileNetV2.

Ta có kiến trúc tổng thể của mạng MobileNetV2 như sau:



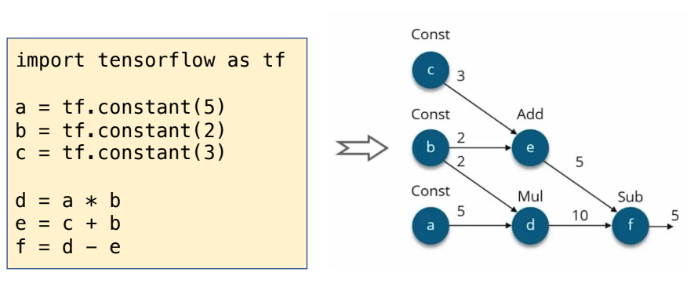
Hình 1. 18: Tổng thể mạng MobileNetV2.

Với t là hệ số mở rộng, c là số kênh đầu ra, n là số lượng khối cho một lớp và s chính là số lượng kernel có kích thước 3x3 được sử dụng trong tích chập.

### Thư viện Tensorflow

TensorFlow là một thư viện mã nguồn mở được phát triển bởi Google, được sử dụng để xây dựng các mô hình học máy và mạng nơ-ron. TensorFlow sẽ giúp giải quyết các bài toán nhanh chóng và đơn giản hơn thông qua việc tạo các mô hình tính toán trong Machine Learning trên máy tính.

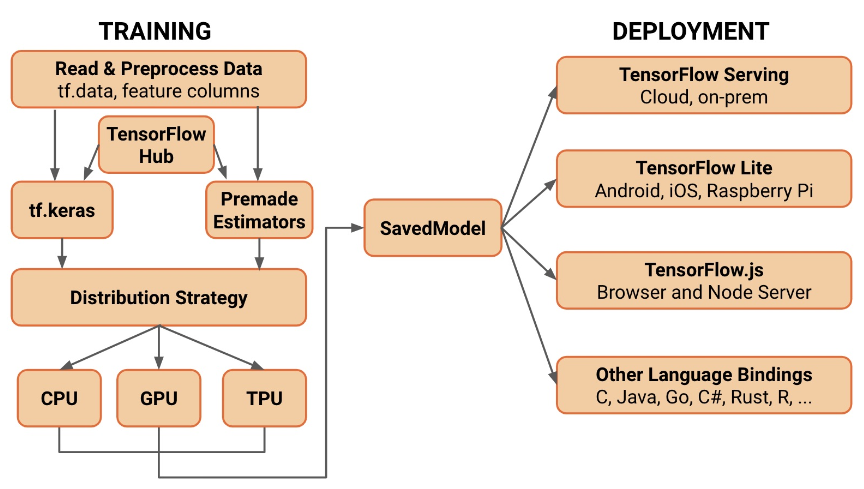
TensorFlow là một thư viện tính toán số và biểu diễn dữ liệu bằng cấu trúc đồ thị (graph) để tạo và huấn luyện các mô hình học máy. Các đồ thị này bao gồm các nút (nodes) và các cạnh (edges) được sử dụng để biểu diễn các phép tính và dữ liệu tương ứng trong mô hình. Cụ thể một đồ thị tính toán được thể hiện như sau:



Hình 1. 19: Đồ thị tính toán trong Tensorflow.

Một cách tổng quát, quá trình huấn luyện mô hình học máy trong TensorFlow trải qua các bước sau:

* Xây dựng mô hình tính toán: Người dùng xác định cấu trúc đồ thị tính toán bằng cách khai báo các biến (variables) và các phép tính (operations) trong mô hình.
* Định nghĩa hàm mất mát: Hàm mất mát (loss function) được định nghĩa để đo lường sự khác biệt giữa đầu ra dự đoán của mô hình và giá trị thực tế.
* Tối ưu hóa mô hình: Quá trình tối ưu hóa được sử dụng để tìm ra các giá trị tham số tối ưu nhằm giảm thiểu hàm mất mát.
* Huấn luyện mô hình: Dữ liệu huấn luyện được đưa vào mô hình để huấn luyện và cập nhật các giá trị tham số.
* Đánh giá mô hình: Dữ liệu kiểm tra được sử dụng để đánh giá hiệu suất của mô hình.
* Sử dụng mô hình: Mô hình đã huấn luyện được sử dụng để dự đoán và phân loại các dữ liệu mới.



Hình 1. 20: Huấn luyện mô hình học máy trong Tensorflow.

## Mô tả chi tiết bài toán

### Mô hình nhận dạng cây thuốc Nam

Với mục tiêu đề tài và kiến thức có được từ nghiên cứu về DeepLearning, cụ thể trong mô hình CNNs và sử dụng phương pháp Transfer Learning trong phát triển một mô hình có khả năng nhận dạng hình ảnh. Với dữ liệu hình ảnh đầu vào là các cây thuốc Nam đã được gán nhãn, mong muốn sau khi xây dựng mô hình có thể từ tập dữ liệu đầu vào, mô hình có thể học được những đặc trưng dữ liệu và phân loại được hình ảnh cây thuốc Nam. Từ đó có thể thực hiện nhiệm vụ khi có một hình ảnh cây thuốc Nam mới, mô hình có thể dự đoán cho ra kết quả đáp ứng yêu cầu bài toán.

### Xây dựng ứng dụng di động với Flutter

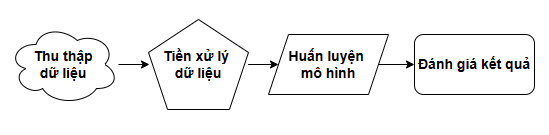
Mô hình tốt nhất sau khi huấn luyện sẽ được chọn để xây dựng một ứng dụng di động thực hiện chức năng phân loại hình ảnh cây thuốc Nam. Và Flutter chính là framework được chọn để thực hiện công việc này.

Flutter là một framework phát triển ứng dụng di động mã nguồn mở, được phát triển bởi Google. Nó cho phép xây dựng các ứng dụng di động đẹp và linh hoạt trên nhiều nền tảng, bao gồm cả iOS và Android từ một mã nguồn duy nhất. Flutter cung cấp một bộ công cụ phong phú để xây dựng giao diện người dùng đẹp và linh hoạt. Từ đó có thể dễ dàng tạo các giao diện phức tạp và tương tác dễ dàng. Sau khi có mô hình được lưu sau khi huấn luyện, chúng ta có thể tích hợp và sử dụng trực tiếp trên ứng dụng thông qua thư viện TensorFlow Lite hoặc có thể sử dụng thông qua một dịch vụ API. Ứng dụng hoàn thiện sẽ cho chúng ta tiện ích có thể tải ảnh hoặc chụp ảnh của cây thuốc Nam, sau đó ứng dụng sẽ sử dụng mô hình phân loại để cho ra dự đoán nhãn của cây thuốc Nam đó cùng với thông tin dược liệu.

# CHƯƠNG 2 XÂY DỰNG MÔ HÌNH VÀ ỨNG DỤNG NHẬN DẠNG CÂY THUỐC NAM

## Xây dựng mô hình

### Quy trình xây dựng mô hình



Đối với bất kỳ bài toán nào trong lĩnh vực học máy, điều quan trọng đầu tiên là phải có dữ liệu. Dữ liệu sẽ được thu thập và tiến hành tiền xử lý dữ liệu. Khi dữ liệu đã được chuẩn hóa và phân chia sẽ thực hiện xây dựng mô hình và huấn luyện trên tập dữ liệu đó. Một mô hình đảm bảo đáp ứng yêu cầu là một mô hình có kết quả tốt trong khâu đánh giá kết quả cuối cùng.

### Thu thập dữ liệu

Dữ liệu được thu thập là hình ảnh của 30 trên 70 loại cây thuốc Nam theo Quyết định số 4664/QĐ-BYT của Bộ Y tế phát hành bộ tranh minh hoạt 70 cây thuốc mẫu được sử dụng trong cơ sở khám chữa bệnh bằng y học cổ truyền.

Hình ảnh được thu thập trong các kho dữ liệu trên Internet tại các trang như istockphoto.com, eol.org,… . Với mỗi loại cây có khoảng 80-110 ảnh màu với nhiều kích cỡ khác nhau.

| **STT** | **Tên cây thuốc Nam** | **Số lượng ảnh** | **Dung lượng** |
| --- | --- | --- | --- |
| 1 | Bạc hà | 100 | 3.08 MB |
| 2 | Bạch đồng nữ | 100 | 2.53 MB |
| 3 | Cam thảo đất | 64 | 1.53 MB |
| 4 | Cây ổi | 100 | 2.64 MB |
| 5 | Cỏ mần trầu | 100 | 2.51 MB |
| 6 | Cỏ mực | 100 | 2.48 MB |
| 7 | Dành dành | 100 | 2.36 MB |
| 8 | Dâu tằm | 100 | 2.21 MB |
| 9 | Địa liên | 95 | 2.44 MB |
| 10 | Diếp cá | 104 | 2.42 MB |
| 11 | Diệp hạ châu | 100 | 2.62 MB |
| 12 | Đinh lăng | 100 | 3.42 MB |
| 13 | Đơn lá đỏ | 70 | 2.33 MB |
| 14 | Dừa cạn | 100 | 2.31 MB |
| 15 | Húng chanh | 101 | 2.65 MB |
| 16 | Ké đầu ngựa | 100 | 2.62 MB |
| 17 | Kim hoa thảo | 69 | 1.54 MB |
| 18 | Kim ngân | 90 | 2.67 MB |
| 19 | Kim tiền thảo | 88 | 2.30 MB |
| 20 | Lá lốt | 101 | 2.87 MB |
| 21 | Mã đề | 94 | 2.90 MB |
| 22 | Ngải cứu | 100 | 3.03 MB |
| 23 | Nhân trần | 80 | 1.94 MB |
| 24 | Phèn đen | 99 | 2.25 MB |
| 25 | Rau má | 80 | 2.26 MB |
| 26 | Sài đất | 94 | 2.55 MB |
| 27 | Sò huyết | 100 | 3.17 MB |
| 28 | Tía tô | 102 | 3.40 MB |
| 29 | Trinh nữ hoàng cung | 100 | 2.93 MB |
| 30 | Vú sữa đất | 109 | 3.24 MB |
| Tổng cộng | | 2840 | 77.2 MB |

### Tiền xử lý dữ liệu

Quá trình tiền xử lý dữ liệu được thực hiện bằng việc gán nhãn dữ liệu thủ công. Hình ảnh sẽ được lưu vào các thư mục với nhãn đại diện của cây thuốc Nam đó. Cụ thể như bảng … dưới đây:

| **STT** | **Tên cây thuốc Nam** | **Tên nhãn** |
| --- | --- | --- |
| 1 | Bạc hà | BacHa |
| 2 | Bạch đồng nữ | BachDongNu |
| 3 | Cam thảo đất | CamThaoDat |
| 4 | Cây ổi | CayOi |
| 5 | Cỏ mần trầu | CoManTrau |
| 6 | Cỏ mực | CoMuc |
| 7 | Dành dành | DanhDanh |
| 8 | Dâu tằm | DauTam |
| 9 | Địa liên | DiaLien |
| 10 | Diếp cá | DiepCa |
| 11 | Diệp hạ châu | DiepHaChau |
| 12 | Đinh lăng | DinhLang |
| 13 | Đơn lá đỏ | DonLaDo |
| 14 | Dừa cạn | DuaCan |
| 15 | Húng chanh | HungChanh |
| 16 | Ké đầu ngựa | KeDauNgua |
| 17 | Kim hoa thảo | KimHoaThao |
| 18 | Kim ngân | KimNgan |
| 19 | Kim tiền thảo | KimTienThao |
| 20 | Lá lốt | LaLot |
| 21 | Mã đề | MaDe |
| 22 | Ngải cứu | NgaiCuu |
| 23 | Nhân trần | NhanTran |
| 24 | Phèn đen | PhenDen |
| 25 | Rau má | RauMa |
| 26 | Sài đất | SaiDat |
| 27 | Sò huyết | SoHuyet |
| 28 | Tía tô | TiaTo |
| 29 | Trinh nữ hoàng cung | TrinhNuHoangCung |
| 30 | Vú sữa đất | VuSuaDat |

### Huấn luyện mô hình

Quá trình huấn luyện mô hình được thực hiện trên môi trường máy ảo được cung cấp bởi Google Colab.

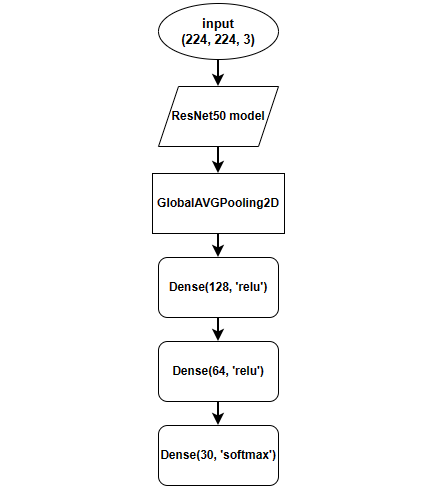
Dữ liệu được phân chia thành 2 tập train và test theo tỉ lệ 80% cho huấn luyện và 20% dành cho kiểm tra.

|  |  |
| --- | --- |
| **Số lượng hình ảnh cho huấn luyện** | **Số lượng hình ảnh cho kiểm thử** |
| 2264 | 572 |

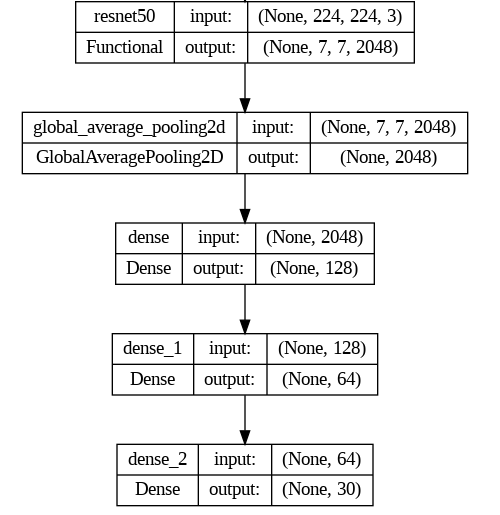
#### Mô hình ResNet50

* **Xây dựng mô hình:**

Đầu vào của mô hình là ảnh có kích thước (224, 224, 3), một kích thước phổ biến để sử dụng trong các mô hình ResNet.



Áp dụng phương pháp transfer learning để xây dựng mô hình sử dụng kiến trúc của mô hình ResNet50 và cấu hình thêm một tầng Pooling cùng 3 tầng Dense với số lượng kernel lần lượt là 128 - 64 - 30. Tầng Dense cuối cùng có tác dụng phân lớp nên sẽ có hàm kích hoạt là ‘softmax’ chịu trách nhiệm chuyển các giá trị đầu ra thành phân phối xác suất của 30 lớp. Chi tiết về đầu vào và đầu ra của từng tầng được minh họa trong hình ảnh sau:



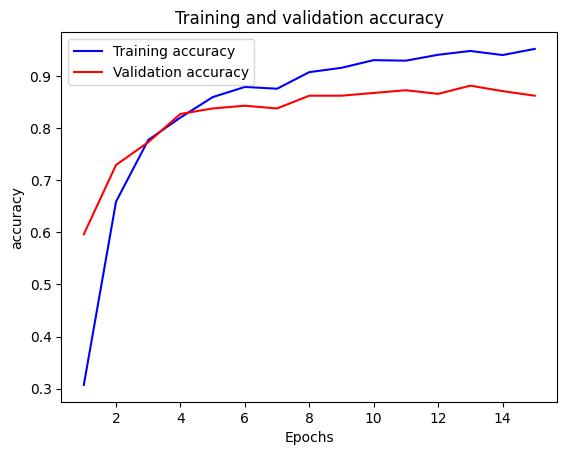
Hình ảnh sau khi đi qua kiến trúc mô hình ResNet50 từ kích thước ban đầu là (None, 224, 224, 3) sẽ trở thành (None, 7, 7, 2048). Tổng tham số của tầng ResNet50 là 23.587.712 tham số.

Tiếp theo hình ảnh sẽ được giảm chiều và giảm tham số khi đi qua tầng Pooling để mỗi đầu vào có kích thước 7x7 sẽ được chuyển thành một vector 1 chiều có kích thước (2048,). Qua đó có thể phù hợp với kích thước đầu vào của các tầng Dense bên dưới. Với tầng Dense đầu tiên có 128 kernel và tổng tham số tầng này là 262.272. Tầng Dense thứ 2 có 64 kernel có tổng tham số là 8256. Và tầng Dense cuối cùng có 30 kernel với tổng tham số là 1950.

Tổng cộng mô hình có 23.860.190 tham số với dung lượng là 91.02 MB.

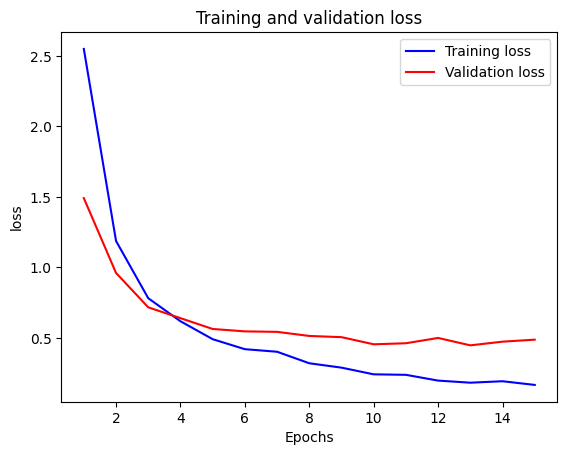
* **Huấn luyện:**

Mô hình được huấn luyện với 15 epochs và batch\_size là 32, cùng với tốc độ học là 0.001. Ghi nhận thời gian huấn luyện 8 phút 52 giây với độ chính xác cao nhất đạt được là 0.8811(tương đương 88.1 %). Biểu đồ biểu diễn độ chính xác trong quá trình huấn luyện như hình dưới đây:



Có thể thấy độ chính xác trên tập kiểm tra (đường màu đỏ) không có sự chênh lệnh quá nhiều so với tập huấn luyện (đường màu xanh dương). Nhưng có phần giảm ở những epochs cuối.

Đồng thời ta có biểu đồ hàm mất mát như sau:

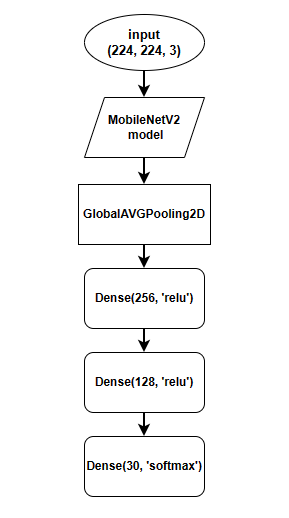


Giá trị mất mát có xu hướng giảm nhưng có thể thấy chưa có sự hội tụ mạnh đối với tập huấn luyện và tập kiểm tra. Nhưng nhìn chung mô hình đã tương đối có thể chấp nhận được.

#### Mô hình MobileNetV2

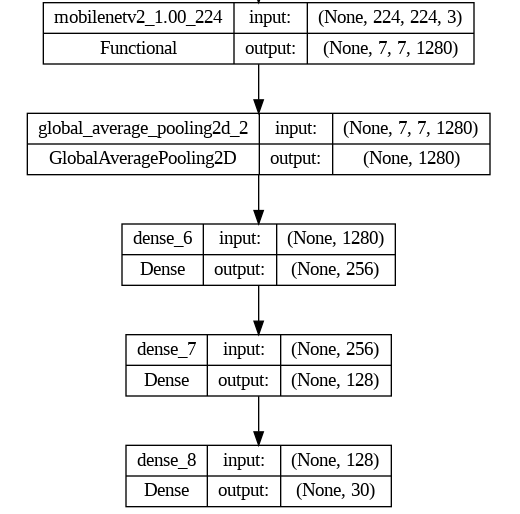
* **Xây dựng mô hình:**

Tương tự như mô hình ResNet50 ở trên, mô hình MobileNetV2 cũng nhận đầu vào là ảnh màu với kích thước (224, 224, 3). Có kiến trúc như sau:



Dữ liệu sau khi đi qua tầng mô hình MobileNetV2 sẽ tiếp tục được giảm chiều ở lớp Pooling. Sau đó sẽ lần lượt đi qua 2 tầng Dense với số kernel lần lượt là 256 - 128 với hàm kích hoạt là ReLU và tầng Dense cuối cùng với kernel 30 đại diện cho 30 nơ-ron của 3 lớp với hàm kích hoạt softmax.

Về chi tiết đầu vào đầu ra và tham số của từng tầng như hình sau:

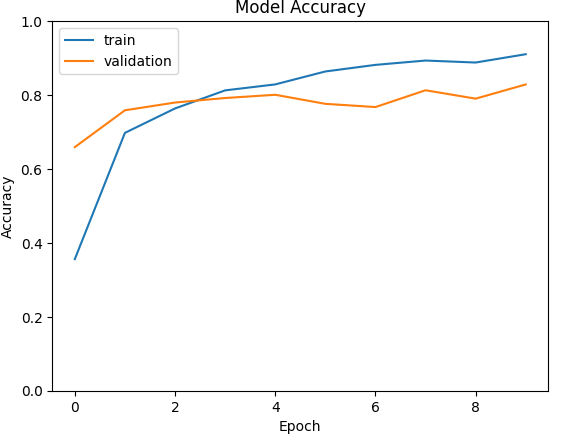


Hình ảnh đầu vào sau khi qua mô hình MobileNetV2 từ kích thước (224,224,3) giảm xuống còn (7,7,1280), và tổng tham số của tầng này là 2.257.984. Tiếp theo dữ liệu sẽ được giảm chiều xuống còn (None, 1280) ở tầng Pooling, tầng này không có tham số. Từ đầu ra ở tầng Pooling, kích thước dữ liệu còn (None, 256) ở tầng Dense thứ nhất, với tham số ở tầng này là 327.936. Tham số ở tầng Dense thứ 2 và tầng Dense cuối cùng lần lượt là 32.896 và 3870.

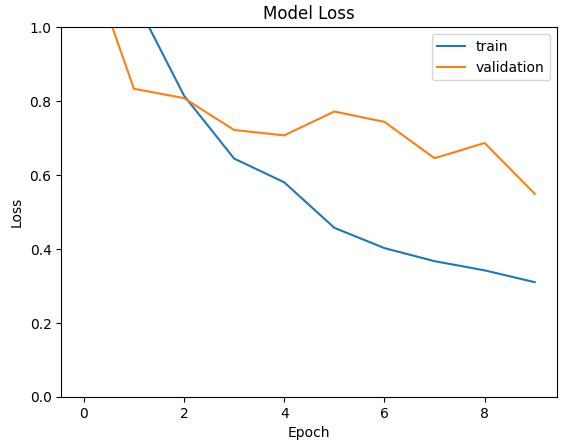
Tổng tham số cho toàn bộ mô hình là: 2.622.686.

* **Huấn luyện:**

Mô hình được huấn luyện với 10 epochs và batch\_size là 32, cùng với tốc độ học là 0.001. Ghi nhận thời gian huấn luyện 21 phút 8 giây với độ chính xác cao nhất đạt được là 0.8287(tương đương 82.87 %). Biểu đồ biểu diễn độ chính xác trong quá trình huấn luyện như hình dưới đây:



Tương tự ta cũng có biểu đồ hàm mất mát như sau:



Có thể thấy cho dù độ chính xác có vẻ tốt nhưng đối với hàm mất mát như trên thì giá trị mất mát trên tập dự đoán còn chênh lệch rất nhiều so với tập dữ liệu huấn luyện.

### Đánh giá kết quả

Qua kết quả huấn luyện mô hình cho thấy mô hình được huấn luyện với kiến trúc ResNet50 cho kết quả tốt hơn so với kiến trúc MobileNetV2. Chi tiết được thống kê trong bảng sau:

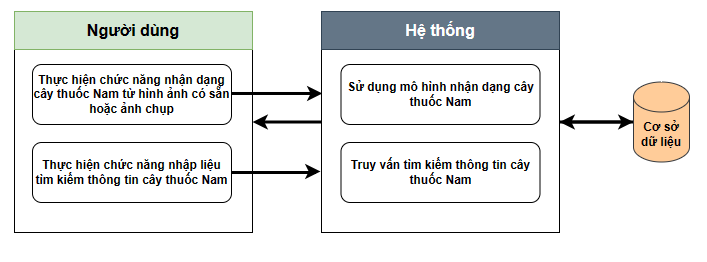
|  |  |  |
| --- | --- | --- |
|  | **ResNet50** | **MobileNetV2** |
| Thời gian huấn luyện | 8 phút 52 giây | 21 phút 8 giây |
| Độ chính xác | 88.1% | 82.87% |
| Tổng tham số | 23.587.712 | 2.622.686 |

Từ thông số trên nên quyết định chọn mô hình được huấn luyện với ResNet50 để xây dựng ứng dụng di động trong nhận dạng cây thuốc Nam.

## Xây dựng ứng dụng nhận dạng cây thuốc Nam

Linh hoạt trong sử dụng, tra cứu nhanh chóng và chính xác thông tin cây thuốc Nam đó là những gì đề tài này mong muốn thực hiện. Mô hình đã được huấn luyện với độ chính xác khá cao và có khả năng sử dụng trong thực tế. Từ đó cần thiết xây dựng một ứng dụng đáp ứng nhu cầu tra cứu thông tin cây thuốc Nam của người dùng dựa trên mô hình đã huấn luyện.

### Sơ đồ hoạt động của ứng dụng



Ứng dụng cho phép người dùng thực hiện hai chức năng đó là:

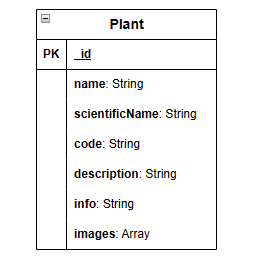
* Nhận dạng cây thuốc Nam:
* Từ ảnh chụp.
* Từ thư viện ảnh.
* Tra cứu thông tin cây thuốc Nam.

Khi nhận yêu cầu từ người dùng, ứng dụng sẽ tiến hành thực hiện dự đoán hoặc tra cứu sau đó sẽ thực hiện tìm kiếm trên cơ sở dữ liệu và trả về thông tin trên ứng dụng.

### Thiết kế cơ sở dữ liệu

Cơ sở dữ liệu được sử dụng trong việc lưu trữ thông tin cây thuốc Nam là MongoDB. Một trình quản lý dữ liệu không quan hệ có tốc độ truy xuất dữ liệu nhanh. Cơ sở dữ liệu lưu trữ các thông tin của cây thuốc nam bao gồm tên thường gọi, tên khoa học, đặc điểm hình thái, thông tin dược liệu và ảnh minh họa. Cụ thể thông tin của một cây thuốc được định nghĩa trong cơ sở dữ liệu như sau:

* Mô hình dữ liệu ở mức quan niệm:



Các trường thông tin cơ bản đều là kiểu dữ liệu dạng chuỗi (String), riêng hình ảnh sẽ lưu dạng mảng, vì mỗi phần tử là một đường liên kết đến kho lưu trữ đám mây là Cloudinary.

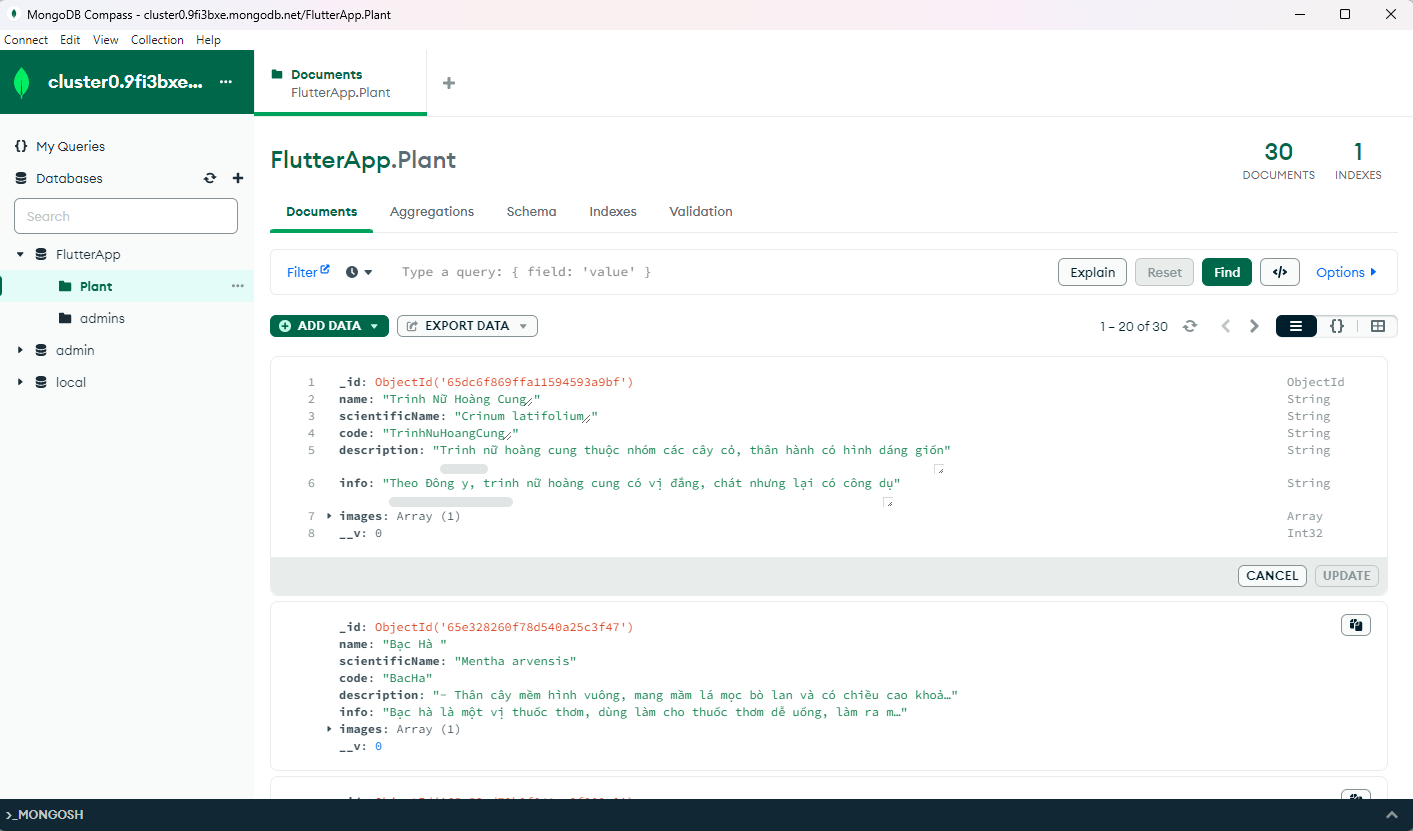
* Mô hình dữ liệu ở mức logic:

Plant(\_id,name, scientificName, code, description, info, images).

* Mô hình dữ liệu ở mức vật lý:

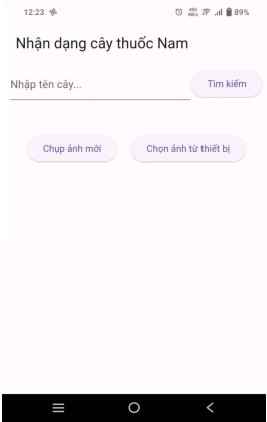
|  |  |  |
| --- | --- | --- |
| **Tên thuộc tính** | **Kiểu dữ liệu** | **Ghi chú** |
| \_id | ObjectID | Khóa chính |
| name | String |  |
| scientificName | String |  |
| code | String |  |
| description | String |  |
| info | String |  |
| images | Array |  |

Minh hoạ giao diện tương tác với cơ sở dữ liệu cây thuốc Nam thông qua ứng dụng MongoDB Compass:



### Ứng dụng

Giao diện chính của ứng dụng



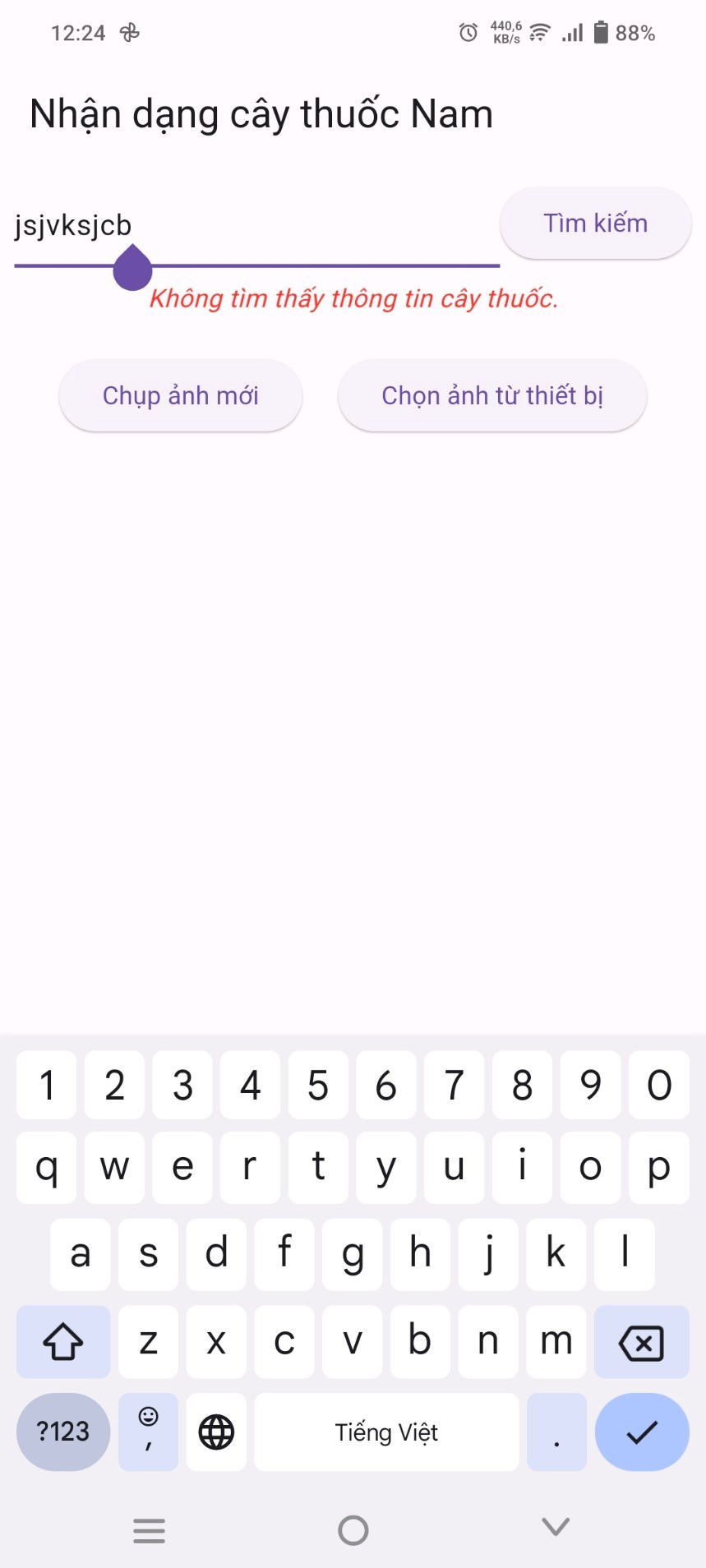
Ứng dụng được thiết kế đơn giản để sử dụng. Tại giao diện chính của ứng dụng người dùng có thể thực hiện tra cứu thông tin cây thuốc Nam, thực hiện dự đoán bằng cách chụp ảnh trực tiếp hoặc chọn ảnh từ thiết bị.

#### Chức năng tìm kiếm

Đối với chức năng tìm kiếm, người dùng cần nhập tên của cây thuốc vào ô nhập liệu. Khuyến khích người dùng nhập tiếng Việt có dấu để có thể truy vấn chính xác nhất. Sau đó khi chọn Tìm kiếm sẽ có hai sự kiện xảy ra:

* Khi không tìm thấy thông tin cây thuốc:

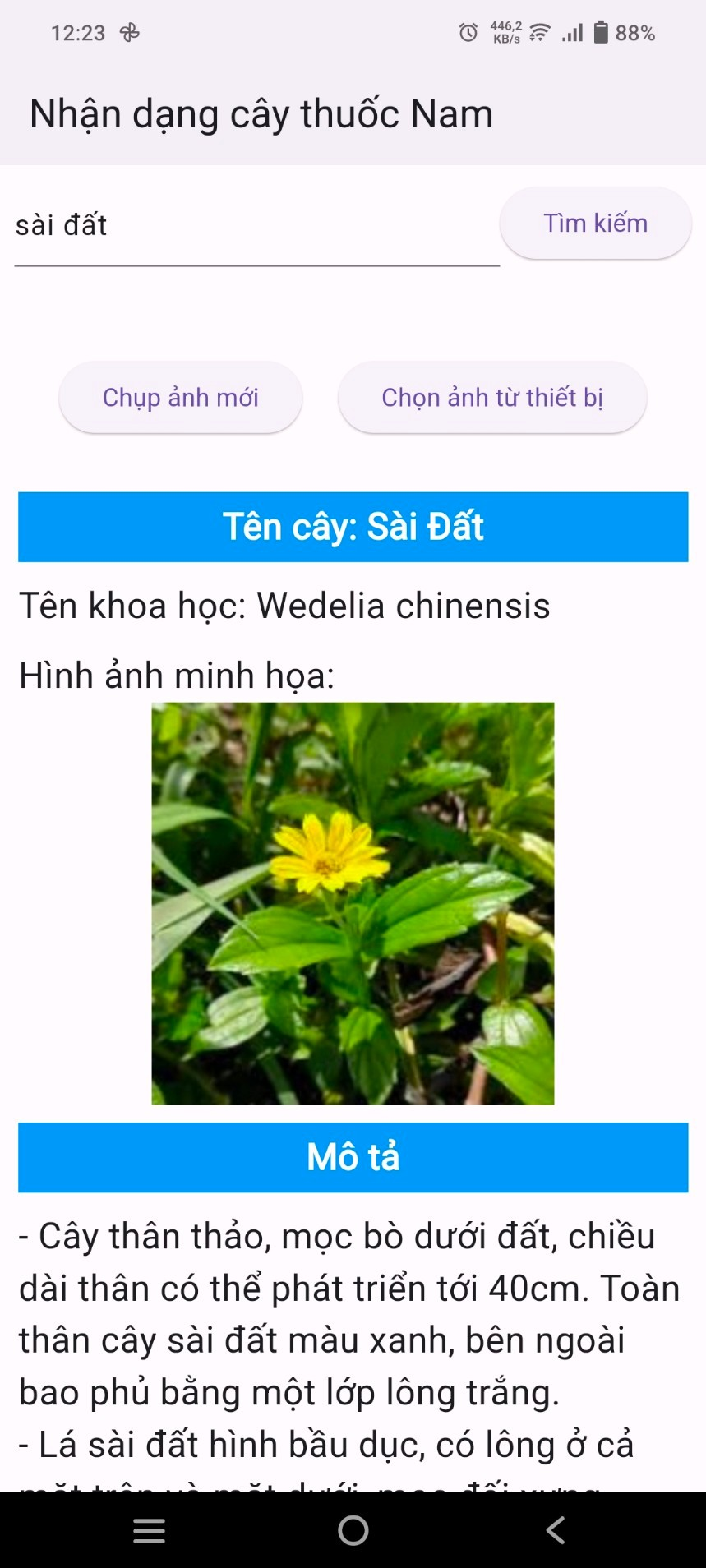
Do một số lí do nào đó như thao tác nhập liệu có vấn đề chính tả, hoặc thông tin cây thuốc hiện chưa có trên CSDL. Lúc này hệ thống sẽ trả về lỗi ‘Không tìm thấy thông tin cây thuốc’.



Người dùng cần nhập lại thông tin tìm kiếm chính xác hơn để có thể lấy thông tin từ CSDL.

* Tìm thấy thông tin cây thuốc:

Khi công việc tìm kiếm thành công dựa trên tên cây thuốc Nam mà người dùng nhập. Lúc này hệ thống tiến hành truy xuất các thông tin liên quan và trả về giao diện người dùng như hình dưới đây:

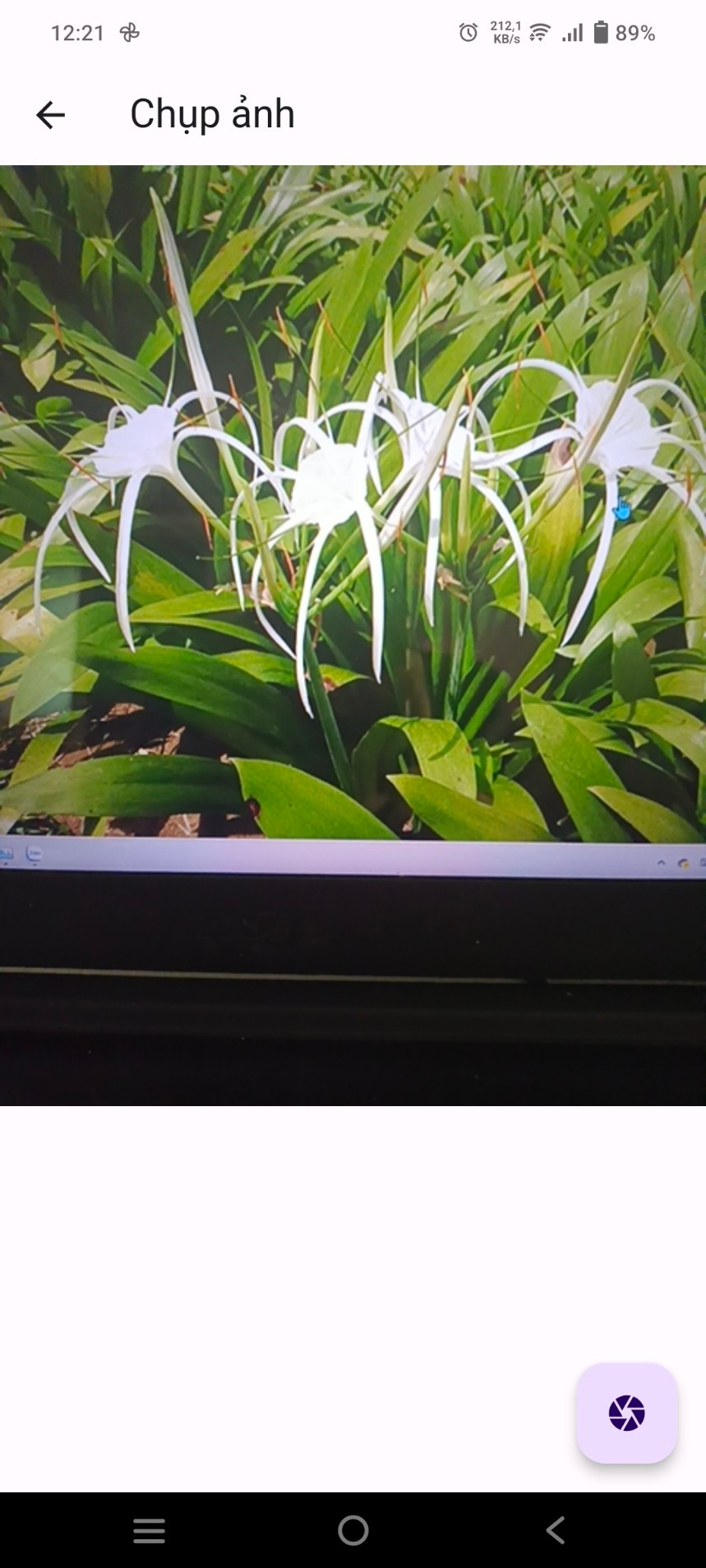


#### Chức năng dự đoán

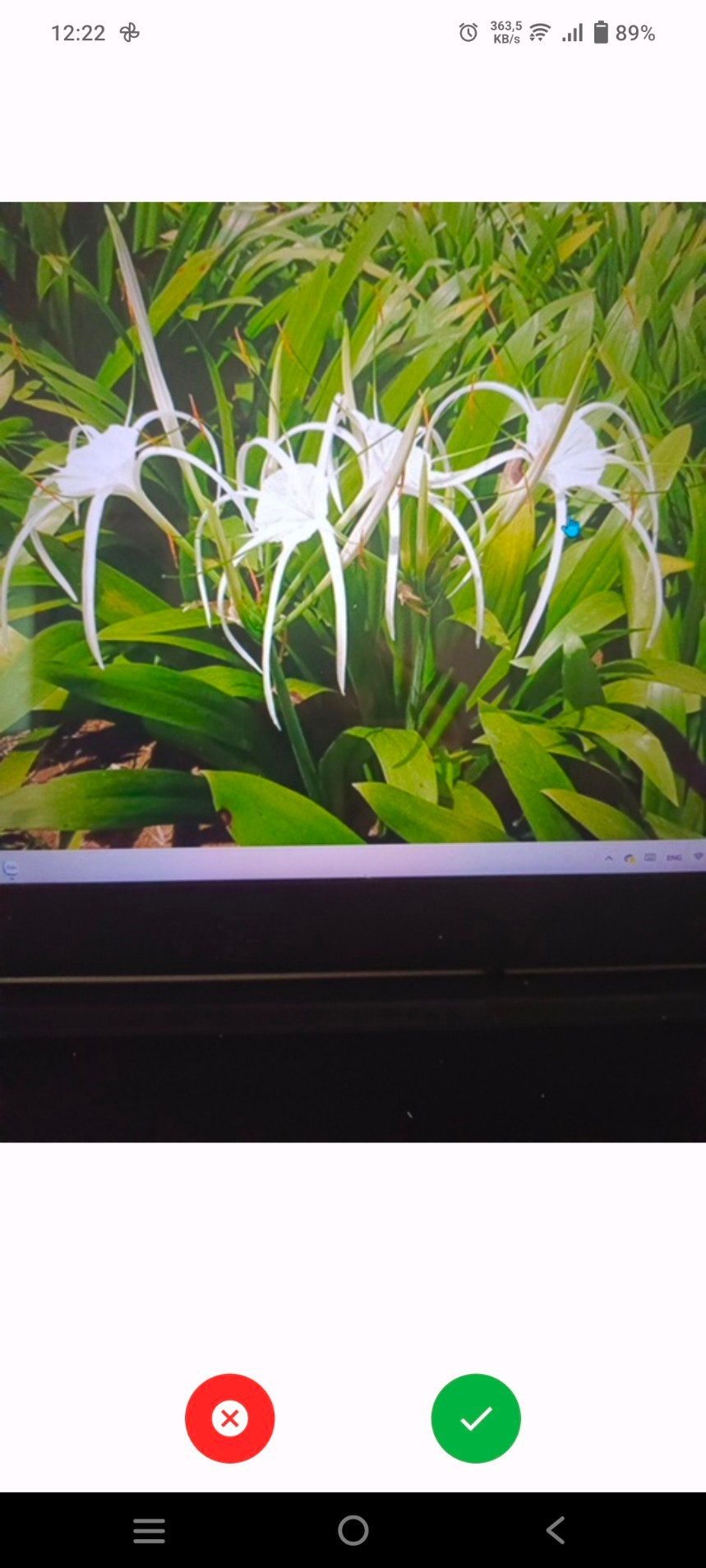
Ở chức năng dự đoán người dùng có 2 lựa chọn đó là: Chụp ảnh mới hoặc chọn ảnh từ thiết bị.

* **Chụp ảnh mới:**

Tại trang chính của ứng dụng, người dùng chọn ‘Chụp ảnh mới’, lúc này ứng dụng sẽ chuyển đến giao diện chụp ảnh như sau:



Người dùng sẽ chọn nút chụp ảnh dưới góc màn hình để chụp ảnh, sau khi chụp ảnh xong, người dùng sẽ có thể lựa chọn tiếp tục dự đoán hoặc chụp lại ảnh khác như sau:



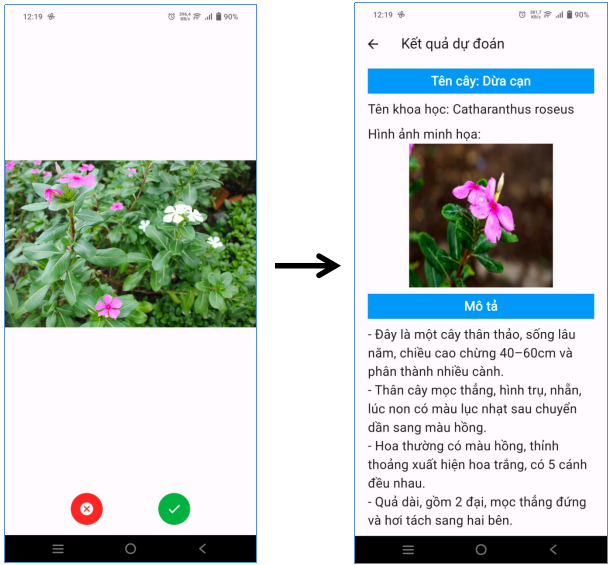
Khi người dùng chọn nút X màu đỏ thì sẽ thực hiện lại chức năng chụp ảnh, hoặc khi người dùng chọn nút tích màu xanh thì lúc này yêu cầu dự đoán với hình ảnh hiện tại được thực hiện. Và người dùng sẽ chuyển đến trang kết quả dự đoán như sau:



Sẽ có hiệu ứng Loading vì khi dự đoán, tại giao diện người dùng sẽ gửi yêu cầu dự đoán đến server thông qua API, khi server dự đoán có kết quá thì thông tin dự đoán mới được gửi lại ứng dụng để hiển thị.

* **Chọn ảnh từ thư viện:**

Khi người dùng chọn chức năng dự đoán bằng hình ảnh từ thư viện, giao diện sẽ hiển thị thư viện ảnh và người dùng sẽ chọn ảnh muốn dự đoán, tiếp theo là giao diện giống lúc duyệt ảnh khi chụp như sau:



# CHƯƠNG 3

# KIỂM THỬ VÀ ĐÁNH GIÁ

## Kết quả kiểm tra

## 

# PHẦN KẾT LUẬN

## 1. Kết quả đạt được

* Xây dựng được phần mềm chỉnh sửa ảnh với các chức năng cơ bản như: đọc ảnh chụp từ camera hoặc ảnh có sẵn, lưu ảnh lại sau khi chỉnh sửa.
* Xây dựng được các hiệu ứng chỉnh sửa cho ảnh.

## 2. Hướng phát triển

* Xây dựng thêm nhiều hiệu ứng ảnh.
* Cải thiện thời gian cũng như chất lượng đầu ra các loại ảnh.

# TÀI LIỆU THAM KHẢO

1. Kkk