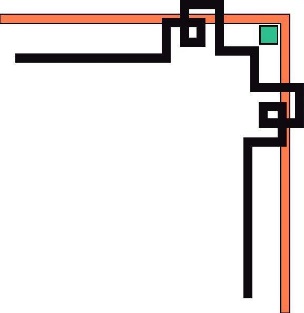
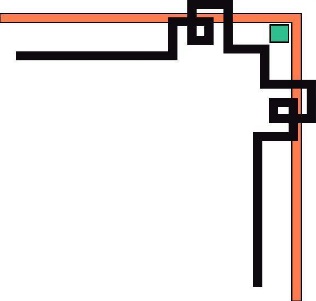
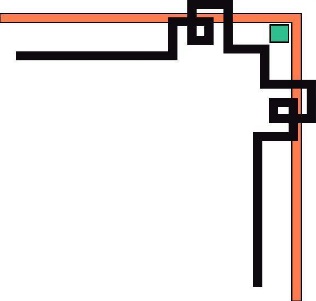
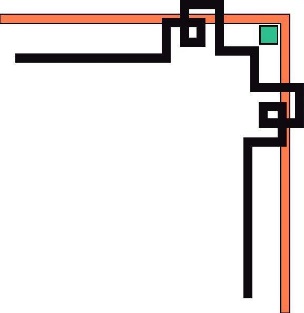
**H**



**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**KHOA KĨ THUẬT ĐIỆN TỬ 1**

****

**BÁO CÁO ĐỒ ÁN XỬ LÝ TÍN HIỆU SỐ**

**ĐỀ TÀI : HỆ THỐNG NHẬN DẠNG LOẠI QUẢ BẰNG MOBILENETV2**

**Giảng viên: Trần Thị Thục Linh**

**Họ và tên : Đoàn Công Minh**

**Mã sinh viên: B20DCDT134**

**Hà Nội 2024**

*Mục lục*

[*CHƯƠNG I: GIỚI THIỆU TỔNG QUAN………………………………………… 1*](#_Toc163924418)

[*1.1 Bài toán nhận dạng và phân loại hoa quả 1*](#_Toc163924419)

[*1.2 Các hướng tiếp cận và giải quyết bài toán 3*](#_Toc163924420)

[*1.2.1 Phương pháp học máy truyền thống 4*](#_Toc163924421)

[*1.2.1.1 Trích chọn đặc trưng 5*](#_Toc163924422)

[*1.2.1.2 Thuật toán 5*](#_Toc163924423)

[*1.2.2 Phương pháp học sâu 6*](#_Toc163924424)

[*CHƯƠNG II: MẠNG NƠ-RON TÍCH CHẬP 9*](#_Toc163924425)

[*2.1 Cấu trúc Mạng nơ-ron tích chập 9*](#_Toc163924426)

[*- Lớp tích chập: 10*](#_Toc163924427)

[*- Lớp kích hoạt phi tuyến ReLU: 11*](#_Toc163924428)

[*- Lớp lấy mẫu: 11*](#_Toc163924429)

[*- Lớp kết nối đầy đủ: 12*](#_Toc163924430)

[*2.2 Mô hình MobileNet 13*](#_Toc163924431)

[*2.2.1 Vấn đề của mạng tích chập thông thường 13*](#_Toc163924432)

[*2.2.2 Mô hình Tích chập tách biệt chiều sâu (Depthwise Separable Convolution) 14*](#_Toc163924433)

[*2.2.3 Mô hình kiến trúc 16*](#_Toc163924434)

[*2.3 Mô hình MobileNetV2 17*](#_Toc163924435)

[*2.3.1 Inverted Residual Block 17*](#_Toc163924436)

[*2.3.2 Loại bỏ Non-Linear 19*](#_Toc163924437)

[*CHƯƠNG III: THIẾT KẾ HỆ THỐNG 20*](#_Toc163924438)

[*3.1 Tổng quan hệ thống 20*](#_Toc163924439)

[*3.2 Dataset 21*](#_Toc163924440)

[*3.3 Các thư viện sử dụng 23*](#_Toc163924441)

[*3.3.1 TensorFlow 23*](#_Toc163924442)

[*3.3.2 TensorFlow Keras 24*](#_Toc163924443)

[*3.3.3 Numpy 24*](#_Toc163924444)

[*3.3.4 Python OpenCV 24*](#_Toc163924445)

[*3.4 Kiến trúc mô hình MobileNetV2 trong hệ thống 24*](#_Toc163924446)

[*3.5 Xây dựng mô hình huấn luyện 25*](#_Toc163924447)

[*CHƯƠNG IV: NHẬN XÉT CHUNG 31*](#_Toc163924448)

[*4.1 Kết quả thu được 31*](#_Toc163924449)

[*4.2 Đánh giá 31*](#_Toc163924450)

[*4.3 Nguyên nhân 31*](#_Toc163924451)

[*4.3 Kết luận 31*](#_Toc163924452)

# *Danh mục hình ảnh:*

*Hình 1.1 Các khó khăn trong bài toán nhận dạng vật thể trong ảnh*

*Hình 1.2 Sự đa dạng về chủng loại của một loại hoa quả*

*Hình 1.3 Các thông tin về hình học được tính toán bởi các thuật toán Xử lý ảnh*

*Hình 1.4 Mô hình hoạt động chung của các phương pháp Học máy*

*Hình 1.5 Mối quan hệ của Học sâu với các lĩnh vực liên quan*

*Hình 1.6 Mức độ trừu tượng tăng dần qua các tầng học của Học sâu*

*Hình 1.7 Bức ảnh quả tạ hai đầu sinh ra bởi mô hình dự đoán Học sâu*

*Hình 2.1 Kiến trúc cơ bản của một mạng tích chập*

*Hình 2.2 Ví dụ bộ lọc tích chập được sử dụng trên ma trận điểm ảnh*

*Hình 2.3 Trường hợp thêm/không thêm viền trắng vào ảnh khi tích chập*

*Hình 2.4 Phương thức Avarage Pooling và Max Pooling*

*Hình 2.5 Sơ đồ làm phẳng*

*Hình 2.6 Vấn đề của mạng tích chập thông thường*

*Hình 2.7 Tính toán với Depthwise convolution*

*Hình 2.8 Thay đổi độ sâu của output với Pointwise convolution*

*Hình 2.9 Mô hình kiến trúc mạng MobileNet*

*Hình 2.10 Convolution truyền thống (Trái), Depthwise separable convolution với BN và ReLU (Phải)*

*Hình 2.11 Một Residual Block*

*Hình 2.12 Inverted residual block trong các kết nối bottleneck*

*Hình 2.13 Tác động của phi tuyến tính trong bottleneck layers*

*Hình 3.1 Tập dataset kaggle*

*Hình 3.2 Folder của 10 class quả*

*Hình 3.3 Mô hình huấn luyện của kiến trúc MobileNetV2*

*Hình 3.4 Sơ đồ hiển thị hàm mất mát của model trên tập train và test*

*Hình 3.5 Sơ đồ hiển thị độ chính xác của model trên tập train và test*

*Hình 3.6 Thử nghiệm chương trình (Dưa hấu)*

*Hình 3.7 Thử nghiệm chương trình (Dưa hấu)*

# **LỜI NÓI ĐẦU**

Trong thời đại hiện đại, sự phát triển nhanh chóng của công nghệ đang mở ra những cánh cửa mới cho việc áp dụng trí tuệ nhân tạo và máy học vào các lĩnh vực đa dạng của cuộc sống. Trong lĩnh vực nông nghiệp và thực phẩm, việc nhận dạng và phân loại các loại quả là một phần quan trọng của quy trình sản xuất và tiêu thụ. Tuy nhiên, với sự đa dạng và phức tạp của các loại quả, việc tự động hóa quá trình này đòi hỏi sự phát triển của các công nghệ nhận dạng hình ảnh mạnh mẽ và chính xác.

Chính vì vậy, trong nghiên cứu này, chúng tôi chọn đề tài "Hệ thống nhận dạng loại quả bằng deep learning" nhằm mục đích xây dựng và phát triển một hệ thống có khả năng nhận biết và phân loại chính xác giữa các loại quả phổ biến. Với sự tiến bộ của deep learning và mạng nơ-ron sâu, chúng tôi tin rằng việc áp dụng công nghệ này vào việc nhận dạng loại quả sẽ mang lại những kết quả tích cực và ứng dụng rộng rãi trong thực tế.

Qua nghiên cứu này, chúng tôi hy vọng sẽ đóng góp vào sự phát triển của lĩnh vực trí tuệ nhân tạo và máy học, đồng thời cung cấp một công cụ hữu ích cho các lĩnh vực như nông nghiệp, công nghiệp thực phẩm và y tế. Chúng tôi cũng mong muốn rằng kết quả từ nghiên cứu này sẽ khơi dậy sự quan tâm và khám phá sâu hơn về ứng dụng của deep learning trong các lĩnh vực thực tiễn khác.

Trong phần tiếp theo của bài báo cáo, chúng tôi sẽ trình bày một cái nhìn tổng quan về vấn đề, phương pháp nghiên cứu, kết quả dự kiến và những đóng góp của nghiên cứu này. Mong rằng sẽ có những phát hiện và thông tin hữu ích từ nghiên cứu của chúng tô

# **CHƯƠNG I: GIỚI THIỆU TỔNG QUAN**

## **1.1 Bài toán nhận dạng và phân loại hoa quả**

Nhận dạng vật thể trong ảnh được coi là bài toán cơ bản nhất trong lĩnh vực Thị giác máy tính, là nền tảng cho rất nhiều bài toán mở rộng khác như bài toán phân lớp, định vị, tách biệt vật thể.... Tuy bài toán cơ bản này đã tồn tại hàng thế kỷ nhưng con người vẫn chưa thể giải quyết nó một cách triệt để, do tồn tại rất nhiều khó khăn để máy tính có thể hiểu được các thông tin trong một bức ảnh. Trong đó, những khó khăn tiêu biểu phải kể đến:

**Sự đa dạng trong điểm nhìn – Viewpoint:** Cùng một vật thể nhưng có thể có rất nhiều vị trí và góc nhìn khác nhau, dẫn đến các hình ảnh thu được về vật thể đó sẽ không giống nhau. Việc huấn luyện để máy tính có thể hiểu được điều này thực sự là một thách thức khó khăn.

**Sự đa dạng trong kích thước:** Các bức ảnh không có cách nào thể hiện trường thông tin về kích thước của vật thể trong đời thực, và máy tính cũng chỉ có thể tính toán được tỉ lệ tương đối của vật thể so với bức ảnh bằng cách đếm theo số lượng các điểm ảnh vật thể đó chiếm trong ảnh.

**Các điều kiện khác nhau của chiếu sáng:** Ánh sáng có ảnh hưởng mạnh mẽ đến thông tin thể hiện trong một bức ảnh, đặc biệt là ở mức độ thấp như mức độ điểm ảnh.

**Sự ẩn giấu một phần của vật thể sau các đối tượng khác trong ảnh:** Trong các bức ảnh, vật thể không nhất định phải xuất hiện với đầy đủ hình dạng mà có thể bị che lấp một phần nào đó bởi nền hoặc các vật thể xung quanh. Sự không đầy đủ về hình dạng của vật thể sẽ dẫn đến việc thiếu thông tin, đặc trưng và càng làm bài toán nhận dạng khó khăn hơn.

**Sự lộn xộn phức tạp của nền:** Trong nhiều trường hợp, vật thể cần nhận dạng bị lẫn gần như hoàn toàn vào nền của bức ảnh, sự lẫn lộn về màu sắc, họa tiết giữa vật thể và nền khiến cho việc nhận dạng trở nên vô cùng khó khăn, kể cả với thị giác con người.

**Sự đa dạng về chủng loại vật thể:** Vật thể cần nhận dạng có thể bao gồm nhiều chủng loại khác nhau, với hình dạng, màu sắc, kết cấu vô cùng khác biệt. Đây chính là một thách thức nữa với bài toán nhận dạng, đó là làm thế nào để các mô hình nhận dạng của máy tính có thể nhận biết được các biến thể về chủng loại của vật thể, ví dụ các loại ghế khác nhau, trong khi vẫn tách biệt được đâu là các vật thể khác loại, ví dụ phân biệt bàn với ghế...



*Hình 1.1: Các khó khăn trong bài toán nhận dạng vật thể trong ảnh*

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 hoa quả theo mùa, vùng miền, địa hình… với vô số loại hoa quả 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 quả do ảnh hưởng của thời tiết, điều kiện thổ nhưỡng và chế độ dinh dưỡng…

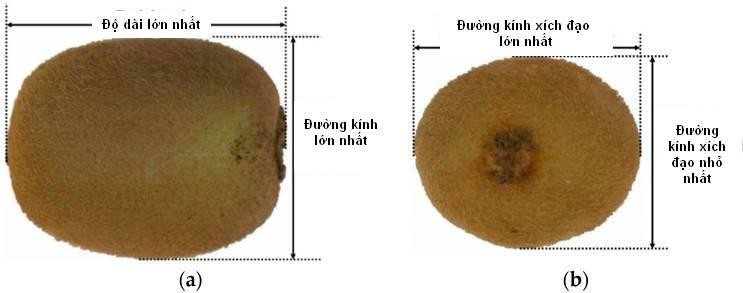
A group of different colored apples

Description automatically generated

*Hình 1.2: Sự đa dạng về chủng loại của một loại hoa quả*

## **1.2 Các hướng tiếp cận và giải quyết bài toán**

Bài toán tự động nhận dạng hoa quả đã xuất hiện từ lâu và đã có rất nhiều bài báo, công trình khoa học được đưa ra nhằm đề xuất hoặc cải tiến các thuật toán nhận dạng. Trong đó, xuất hiện sớm nhất là các phương pháp Xử lý ảnh – Image Processing, các phương pháp này tập trung vào phát triển các thuật toán nhằm trích xuất thông tin, ví dụ các tham số về màu sắc, hình dạng, kết cấu, kích thước…, từ bức ảnh đầu vào để nhận dạng hoa quả . Do chỉ đơn thuần xử lý trên một vài ảnh đầu vào trong khi sự biến thiên về màu sắc, hình dạng, kích thước… của hoa quả quá phức tạp, kết quả đạt được của các phương pháp này không được cao và phạm vi áp dụng trên số lượng loại hoa quả cũng bị hạn chế.

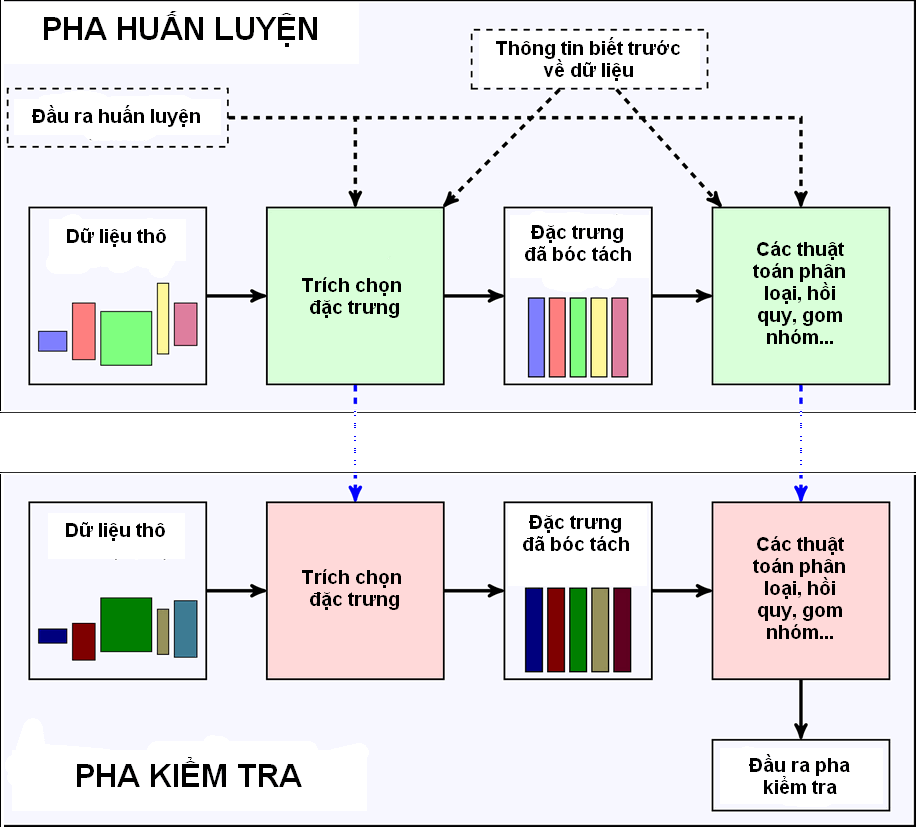


*Hình 1.3: Các thông tin về hình học được tính toán bởi các thuật toán Xử lý ảnh*

Bắt đầu từ những năm 2000s, sau khi xuất hiện một bài bài báo khoa học đề xuất áp dụng phương pháp Học máy - Machine Learning - vào bài toán nhận dạng hoa quả với độ chính xác cao, hướng giải quyết bài toán đã tập trung vào ứng dụng và cải tiến các thuật toán Học máy, cụ thể là nghiên cứu, thử nghiệm trích chọn các đặc trưng phù hợp nhất để đưa vào huấn luyện bộ nhận dạng tự động . Kết quả thu được tương đối khả quan, khả năng nhận dạng hoa quả tự động đã được cải thiện với số lượng loại hoa quả được mở rộng và độ chính xác của nhận dạng cao hơn nhiều so với các phương pháp thuần Xử lý ảnh ban đầu. Nối tiếp sự phát triển của Học máy, trong những năm gần đây, nhờ sự phát triển vượt bậc về sức mạnh tính toán của các máy tính cũng như sự bùng nổ dữ liệu trên Internet, một nhánh đặc biệt trong Học máy là Học sâu - Deep Learning đã đạt được nhiều thành tựu đáng kể, đặc biệt là trong lĩnh vực Xử lý ảnh và ngôn ngữ tự nhiên. Học sâu cũng đã được áp dụng rất thành công vào bài toán nhận dạng hoa quả, trong các thử nghiệm với phạm vi hạn chế về số lượng loại hoa quả cần nhận dạng, phương pháp này đã đạt được kết quả rất cao. Sau đây ta sẽ tìm hiểu sâu hơn về hai tiếp cận chính hiện nay để giải quyết bài toán nhận dạng hoa quả nói riêng và nhận dạng vật thể trong ảnh nói chung: phương pháp Học sâu và các phương pháp Học máy truyền thống không sử dụng Học sâu.

### **1.2.1 Phương pháp học máy truyền thống**

Mô hình hoạt động chung của các phương pháp Học máy truyền thống được thể hiện trong hình dưới đây:



*Hình 1.4: Mô hình hoạt động chung của các phương pháp Học máy*

Từ hình ta có thể thấy Học máy gồm hai giai đoạn chính là Huấn luyện – Training và Thử nghiệm – Testing, trong mỗi giai đoạn đều sử dụng hai thành phần quan trọng nhất do người xử lý bài toán thiết kế, đó là Trích chọn đặc trưng – Feature Engineering (hay còn gọi là Feature Extraction) và Thuật toán phân loại, nhận dạng… - Algorithms. Hai thành phần này có ảnh hưởng trực tiếp đến kết quả bài toán, vì thế được thiết kế rất cẩn thận, tốn nhiều thời gian, đòi hỏi người thiết kế phải có kiến thức chuyên môn và nắm rõ đặc điểm của bài toán cần xử lý.

**1.2.1.1 Trích chọn đặc trưng**

Trong các bài toán thực tế, ta chỉ có được những dữ liệu thô chưa qua chọn lọc xử lý, và để có thể đưa các dữ liệu này vào huấn luyện ta cần có những phép biến đổi để biến các dữ liệu thô thành dữ liệu chuẩn, với khả năng biểu diễn dữ liệu tốt hơn. Các phép biến đổi bao gồm loại bỏ dữ liệu nhiễu và tính toán để lưu lại các thông tin đặc trưng, có ý nghĩa từ dữ liệu thô ban đầu. Các thông tin đặc trưng này là khác nhau với từng loại dữ liệu và bài toán cụ thể, vì thế trong từng trường hợp phép biến đổi này cần phải được tùy biến một cách thích hợp để cải thiện độ chính xác của mô hình dự đoán. Quá trình này được gọi là *Trích chọn đặc trưng* – Feature Engineering, là một thành phần rất quan trọng trong các phương pháp Học máy truyền thống.

**Đầu vào:** Toàn bộ thông tin của dữ liệu, không có quy chuẩn về dạng thông tin (véc tơ, ma trận…) hay kích thước các chiều thông tin. Đồng thời, do chứa toàn bộ thông tin, gồm cả thông tin nhiễu và không có giá trị nên kích thước lưu trữ thường lớn và không có lợi cho tính toán sau này.

**Đầu ra:** Các thông tin hữu ích đã được tính toán, rút ra từ dữ liệu đầu vào, trong đó không còn các thành phần nhiễu hay vô nghĩa. Kích thước dữ liệu đầu ra đã được rút gọn rất nhiều so với kích thước dữ liệu đầu vào, giúp cho việc tính toán về sau trở nên nhanh gọn, thuận tiện hơn rất nhiều.

**Thông tin biết trước về dữ liệu:** Đây là thành phần tùy chọn, không bắt buộc với mọi bài toán, mà chỉ xuất hiện trong một số trường hợp cụ thể với những thông tin rõ ràng về đặc trưng hữu ích với mô hình dự đoán. Các thông tin biết trước này giúp người thiết kế có thể lựa chọn được những đặc trưng tốt nhất và các phương pháp tính toán phù hợp nhất để ra được mô hình dự đoán với độ chính xác cao.

* + - 1. **Thuật toán**

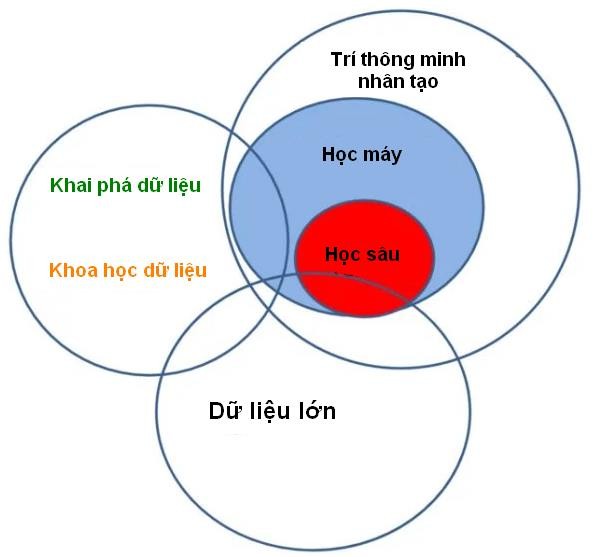
Sau quá trình trích chọn đặc trưng ở bước trước, ta có được các đặc trưng, được lưu trữ ở định dạng chuẩn về kiểu dữ liệu, kích thước dữ liệu…, và các thông tin đặc trưng này có thể được sử dụng cùng với các thông tin biết trước về dữ liệu (nếu có) để xây dựng ra các mô hình dự đoán phù hợp bằng các thuật toán khác nhau. Các thuật toán trong Học máy thường được phân loại theo hai cách phổ biến là theo phương thức học hoặc theo chức năng của thuật toán, ví dụ như:

* Phân nhóm theo phương thức học: Học giám sát và Học không giám sát (Supervised và Unsupervised Learning)
* Phân nhóm theo chức năng: Các thuật toán hồi quy, phân loại, gom nhóm…

Một đặc điểm nổi bật của các phương pháp Học máy truyền thống là độ chính xác của mô hình dự đoán phụ thuộc rất nhiều vào chất lượng các đặc trưng được lựa chọn, các đặc trưng này càng phù hợp với bài toán đưa ra thì kết quả thu được càng tốt. Đây là điểm mạnh, và cũng là điểm yếu của các phương pháp này, bởi việc trích chọn đặc trưng chính là sự đóng góp của bản tay con người trong việc cải tiến các mô hình, nó yêu cầu sự hiểu biết thấu đáo về bài toán cần giải quyết, các thuật toán sử dụng và các thông số trong mô hình huấn luyện. Các đặc trưng được thiết kế riêng cho từng bài toán khác biệt, do vậy hiếm khi chúng có thể được tái sử dụng với các bài toán mới mà cần phải được cải thiện hay thay thế bởi các đặc trưng khác.

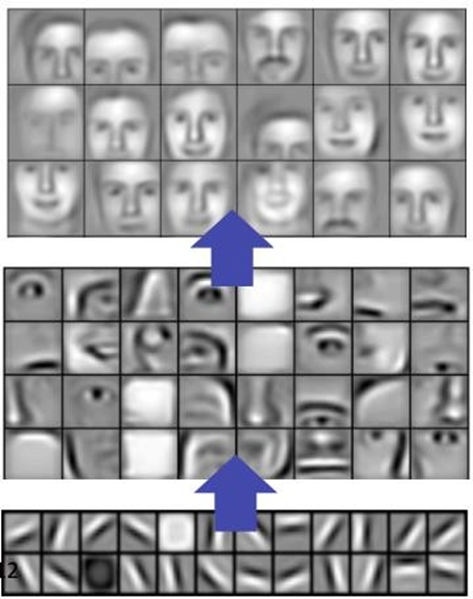
* + 1. **Phương pháp học sâu**

Học sâu là một nhánh đặc biệt của ngành Học máy, và bắt đầu trở nên phổ biến trong thập kỷ gần đây do các nhà khoa học đã có thể tận dụng khả năng tính toán mạnh mẽ của các máy tính hiện đại cũng như khối lượng dữ liệu khổng lồ (hình ảnh, âm thanh, văn bản,…) trên Internet. Ta có thể thấy rõ mối quan hệ giữa Học sâu với Học máy cũng như các lĩnh vực liên quan khác qua hình ảnh mô tả bên dưới:



*Hình 1.5: Mối quan hệ của Học sâu với các lĩnh vực liên quan*

Các mạng huấn luyện theo phương pháp Học sâu còn được gọi với cái tên khác là mạng nơ-ron sâu (Deep Neural Network) do cách thức hoạt động của chúng. Về cơ bản, các mạng này bao gồm rất nhiều lớp khác nhau, mỗi lớp sẽ phân tích dữ liệu đầu vào theo các khía cạnh khác nhau và theo mức độ trừu tượng nâng cao dần (xem [Hình](#_bookmark22) [1.6](#_bookmark22)).

****

*Hình 1.6: Mức độ trừu tượng tăng dần qua các tầng học của Học sâu*

Cụ thể, với một mạng Học sâu cho nhận dạng ảnh, các lớp đầu tiên trong mạng chỉ làm nhiệm vụ rất đơn giản là tìm kiếm các đường thẳng, đường cong, hoặc đốm màu trong ảnh đầu vào. Các thông tin này sẽ được sử dụng làm đầu vào cho các lớp tiếp theo, với nhiệm vụ khó hơn là từ các đường, các cạnh đó tìm ra các thành phần của vật thể trong ảnh. Cuối cùng, các lớp cao nhất trong mạng huấn luyện sẽ nhận nhiệm vụ phát hiện ra vật thể trong ảnh.

Với cách thức học thông tin từ ảnh lần lượt qua rất nhiều lớp, nhiều tầng khác nhau như vậy, các phương pháp này có thể giúp cho máy tính hiểu được những dữ liệu phức tạp bằng nhiều lớp thông tin đơn giản qua từng bước phân tích. Đó cũng là lý do chúng được gọi là các phương pháp Học sâu.

Tuy có nhiều điểm ưu việt trong khả năng huấn luyện máy tính cho các bài toán phức tạp, Học sâu vẫn còn rất nhiều giới hạn khiến nó chưa thể được áp dụng vào giải quyết mọi vấn đề. Điểm hạn chế lớn nhất của phương pháp này là yêu cầu về kích thước dữ liệu huấn luyện, mô hình huấn luyện Học sâu đòi hỏi phải có một lượng khổng lồ dữ liệu đầu vào để có thể thực hiện việc học qua nhiều lớp với một số lượng lớn nơ-ron và tham số. Đồng thời, việc tính toán trên quy mô dữ liệu và tham số lớn như vậy cũng yêu cầu đến sức mạnh xử lý của các máy tính server cỡ lớn. Quy trình chọn lọc dữ liệu cũng như huấn luyện mô hình đều tốn nhiều thời gian và công sức, dẫn đến việc thử nghiệm các tham số mới cho mô hình là công việc xa xỉ, khó thực hiện. Tuy nhiên, nhờ các phương pháp Học tập chuyển giao, hiện nay điểm hạn chế lớn nhất này đã không còn là vấn đề quá nghiêm trọng như trước – điều này sẽ được trình bày cụ thể trong các chương sau.

Ngoài hạn chế về kích thước dữ liệu đầu vào, Học sâu còn chưa đủ thông minh để nhận biết và hiểu được các logic phức tạp như con người, các tác vụ do chúng thực hiện vẫn tương đối máy móc và cần cải thiện để “thông minh” hơn nữa. Trong ví dụ [Hình 1.7](#_bookmark23), ta có thể nhận thấy sự vô lý trong bức ảnh về quả tạ hai đầu mà mạng Học sâu tạo ra sau khi được huấn luyện với hàng loạt ảnh mẫu. Bức ảnh có chứa các phần ảnh về cánh tay con người, là thành phần không phải thuộc về quả tạ. Việc hình ảnh cánh tay xuất hiện trong phần lớn các ảnh mẫu đã dẫn đến sự nhầm lẫn của mô hình dự đoán này.

A collage of a person lifting weights

Description automatically generated

*Hình 1.7: Bức ảnh quả tạ hai đầu sinh ra bởi mô hình dự đoán Học sâu*

Như đã trình bày trong phần mở đầu, mục đích của luận văn là tìm hiểu và ứng dụng một mô hình Học sâu vào bài toán nhận dạng, phân loại hoa quả, nguyên nhân chính khiến Học sâu được chọn làm giải pháp là bởi khả năng mạnh mẽ vượt trội của nó đối với các phương pháp Học máy truyền thống khi áp dụng vào các bài toán nhận dạng vật thể, trong đó vật thể là các đối tượng rất khó chọn lọc đặc trưng phù hợp, cụ thể với trường hợp này là các loại hoa quả. Để chứng minh cho nhận định này, luận văn đã thực hiện phép so sánh độ chính xác của hai mô hình nhận dạng, được huấn luyện lần lượt bởi hai phương pháp trên với cùng bộ dữ liệu đầu vào.

# **CHƯƠNG II: MẠNG NƠ-RON TÍCH CHẬP**

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 ảnh lớn của Facebook, Google hay Amazon… cho các mục đích khác nhau như các thuật toán tagging tự động, tìm kiếm ảnh hoặc gợi ý sản phẩm cho người tiêu dùng.

Sự ra đời của mạng CNN là dựa trên ý tưởng cải tiến cách thức các mạng nơ-ron nhân tạo truyền thống học thông tin trong ảnh. Do sử dụng các liên kết đầy đủ giữa các điểm ảnh vào node, các mạng nơ-ron nhân tạo truyền thẳng (Feedforward Neural Network) bị hạn chế rất nhiều bởi kích thước của ảnh, ảnh càng lớn thì số lượng liên kết càng tăng nhanh và kéo theo sự bùng nổ khối lượng tính toán. Ngoài ra sự liên kết đầy đủ này cũng là sự dư thừa khi với mỗi bức ảnh, các thông tin chủ yếu thể hiện qua sự phụ thuộc giữa các điểm ảnh với những điểm xung quanh nó mà không quan tâm nhiều đến các điểm ảnh ở cách xa nhau. Mạng CNN ra đời với kiến trúc thay đổi, có khả năng xây dựng liên kết chỉ sử dụng một phần cục bộ trong ảnh kết nối đến node trong lớp tiếp theo thay vì toàn bộ ảnh như trong mạng nơ-ron truyền thẳng.

**2.1 Cấu trúc Mạng nơ-ron tích chập**

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) và 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.

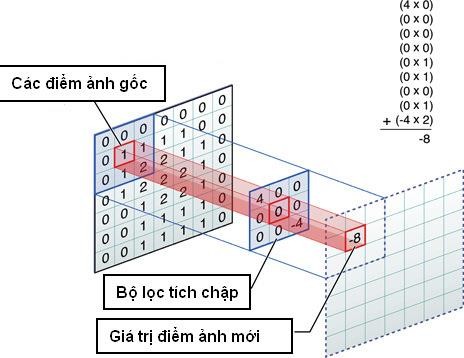
A diagram of a diagram

Description automatically generated

*Hình 2.1: Kiến trúc cơ bản của một mạng tích chập*

### - **Lớp tích chập**:

Đây là thành phần quan trọng nhất trong mạng CNN, cũng là nơi thể hiện tư tưởng xây dựng sự liên kết cục bộ thay vì kết nối toàn bộ các điểm ảnh. Các liên kết cục bộ này 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 2.2: Ví dụ bộ lọc tích chập được sử dụng trên ma trận điểm ảnh*

Trong ví dụ ở [Hình 2.2](#_bookmark27) , ta thấy bộ lọc được sử dụng là một ma trận có kích thước 3x3. Bộ lọc này được 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 (1) [13]:

A black text on a white background

Description automatically generated

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

A blue and green squares

Description automatically generated

*Hình 2.3: Trường hợp thêm/không thêm viền trắng vào ảnh khi tích chập*

Như vậy, sau khi đưa một bức ảnh đầu vào cho lớp Tích chập ta 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 cải thiện dần xuyên suốt quá trình huấn luyện.

### - **Lớp kích hoạt phi tuyến ReLU:**

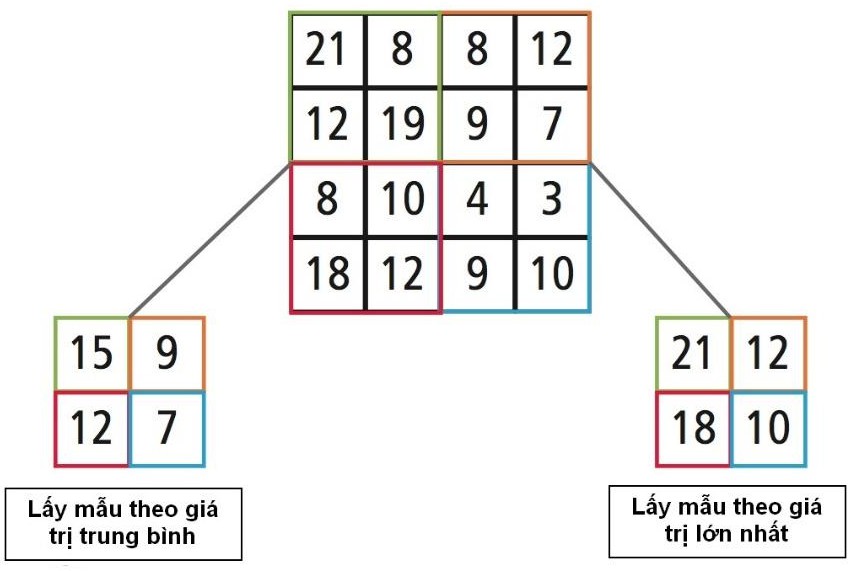
Lớp này được xây dựng với ý nghĩa đảm bảo tính phi tuyến của mô hình huấn luyện sau khi đã thực hiện một loạt các phép tính toán tuyến tính qua các lớp Tích chập. Lớp kích hoạt phi tuyến nói chung sử dụng các hàm kích hoạt phi tuyến như ReLU hoặc sigmoid, tanh… để giới hạn phạm vi biên độ cho phép của 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. Cụ thể, phép tính toán của hàm ReLU chỉ đơn giản là chuyển tất cả các giá trị âm thành giá trị 0.

𝑓(𝑥) = 𝑚𝑎𝑥(0, 𝑥)

Thông thường, 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.

- **Lớp lấy mẫu:**

Một thành phần tính toán chính khác trong mạng CNN là lấy mẫu (Pooling), thường được đặt sau lớp Tích chập và lớp ReLU để làm giảm kích thước 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 tương tự như lớp Tích chập, và thực hiện phép lấy mẫu thay vì phép tích chập – tức là ta sẽ chọn lưu lại một giá trị duy nhất đại diện cho toàn bộ thông tin của vùng ảnh đó.



*Hình 2.4: Phương thức Avarage Pooling và Max Pooling*

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:**

Lớp kết nối đầy đủ này được thiết kế hoàn toàn tương tự như trong mạng nơ-ron truyền thống, tức là tất cả các điểm ảnh được kết nối đầy đủ với node 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 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.

A close-up of a map

Description automatically generated

*Hình 2.5 Sơ đồ làm phẳng*

## 2.2 Mô hình MobileNet

Là 1 kiến trúc neural network được phát triển bởi nhóm các nhà nghiên cứu của Google. Kiến trúc này mang lại kết quả chính xác cao trong khi vẫn giữ các tham số và phép toán ở mức thấp nhất có thể để có khả năng chạy trên các thiết bị di động. MobileNets là các mô hình nhỏ, độ trễ thấp, công suất thấp được tham số hóa để đáp ứng các hạn chế về tài nguyên của nhiều trường hợp sử dụng khác nhau. Theo các bài báo nghiên cứu, MobileNetV2 cải thiện hiệu suất của các mô hình di động trên nhiều tác vụ cũng như trên nhiều kích thước mô hình khác nhau. Nó là một trình trích xuất tính năng rất hiệu quả để phát hiện và phân đoạn đối tượng.

* + 1. **Vấn đề của mạng tích chập thông thường**

Convolution ta có thể đơn giản nó là một cửa sổ trượt trên một ma trận, các lớp Convolution có các parameter (Kernel Matrix) sẽ tiến hành tính toán và đưa ra các giá trị đặc trưng nhờ đó có thể trích suất các điểm đặc trưng và giảm chiều tính toán. Nhưng tích chập 2 chiều thông thường sẽ được tính toán trên toàn bộ chiều sâu (channel). Do đó số lượng tham số của mô hình sẽ gia tăng đáng kể phụ thuộc vào độ sâu của layer trước đó.

A purple rectangular object with black arrows

Description automatically generated

*Hình 2.6 Vấn đề của mạng tích chập thông thường*

Như hình trên, ta có một đầu vào kích thước , tích chập thông thường sẽ cần tham số để thực hiện tích chập trên toàn bộ độ sâu của layers. Mỗi một bộ lọc sẽ tạo ra một ma trận output kích thước . Áp dụng bộ lọc khác nhau ta sẽ tạo ra đầu ra có kích thước . Khi đó số lượng tham số cần sử dụng cho một tích chập thông thường sẽ là: . Khi độ sâu tăng tiến dần về những layers cuối cùng của mạng CNN thì số lượng tham số của mô hình sẽ là một rất lớn. Sự gia tăng tham số này tạo ra những mô hình cồng kềnh làm chiếm dụng bộ nhớ và ảnh hưởng tới tốc độ tính toán.

Độ phức tạp tính toán trên khá lớn khó có thể áp dụng vào những dự án trên các thiết bị nhỏ gọn như Smartphone hay các thiết bị nhúng. Cần có một mô hình tích chập đơn giản hơn để khắc phục vấn đề này.

* + 1. **Mô hình Tích chập tách biệt chiều sâu (Depthwise Separable Convolution)**

Tích chập tách biệt chiều sâu sẽ tìm cách loại bỏ sự phụ thuộc vào độ sâu khi tích chập mà vẫn tạo ra được một output shape có kích thước tương đương so với tích chập thông thường. Ý tưởng của Depthwise Separable Convolution là chia phép convolution làm 2 bước: Depthwise convolution & Pointwise convolution.

* Depthwise convolution là một loại tích chập trong đó chúng ta áp dụng một bộ lọc tích chập duy nhất cho mỗi kênh đầu vào. Trong phép tích chập 2D thông thường được thực hiện trên nhiều kênh đầu vào, bộ lọc cũng sâu như đầu vào và cho phép chúng ta tự do trộn các kênh để tạo ra từng phần tử trong đầu ra. Ngược lại, sự biến đổi theo chiều sâu giữ cho mỗi kênh riêng biệt. Vì vậy ta có thể chia khối input tensor3D thành những lát cắt ma trận theo độ sâu. Thực hiện tích chập trên từng lát cắt như hình minh họa bên dưới. Các bước thực hiện:
* Tách đầu vào và lọc thành các kênh.
* Chuyển đổi từng đầu vào với bộ lọc tương ứng.
* Xếp chồng các kết quả đầu ra được biến đổi với nhau.

A diagram of a diagram of a diagram

Description automatically generated

*Hình 2.7 Tính toán với Depthwise convolution*

Mỗi một channel sẽ áp dụng một bộ lọc khác nhau và hoàn toàn không chia sẻ tham số. Điều này có ba tác dụng chính cho mô hình:

* Nhận diện đặc trưng: Quá trình học và nhận diện đặc trưng sẽ được tách biệt theo từng bộ lọc. Nếu đặc trưng trên các channels là khác xa nhau thì sử dụng các bộ lọc riêng cho channel sẽ chuyên biệt hơn trong việc phát hiện các đặc trưng. Chẳng hạn như đầu vào là ba kênh RGB thì mỗi kênh áp dụng một bộ lọc khác nhau chuyên biệt.
* Giảm thiểu khối lượng tính toán: Để tạo ra một điểm pixel trên output thì tích chập thông thường cần sử dụng phép tính trong khi tích chập chiều sâu tách biệt chỉ cần phép tính.
* Giảm thiếu số lượng tham số : Ở tích chập chiều sâu cần sử dụng tham số. Số lượng này ít hơn gấp lần so với tích chập chiều sâu thông thường.

Kết quả sau tích chập được concatenate lại theo độ sâu. Như vậy output thu được là một khối tensor3D có kích thước .

* Pointwise convolution Có tác dụng thay đổi độ sâu của output bước trên từ sang . Chúng ta sẽ áp dụng bộ lọc kích thước . Như vậy kích thước width và height không thay đổi mà chỉ độ sâu thay đổi.

A diagram of a diagram of a diagram

Description automatically generated with medium confidence

*Hình 2.8 Thay đổi độ sâu của output với Pointwise convolution*

Kết quả sau cùng chúng ta thu được là một output có kích thước  . Số lượng tham số cần áp dụng ở trường hợp này là .

Thử so sánh độ phức tạp giữa 2 mô hình bằng công thức với 64 filter (3x3) thì ta được kết quả đã giảm khoảng 8 lần so với việc sử dụng Convolution truyền thống. Các thực nghiệm trên tập dữ liệu cũng cho ra kết luận tương tự.

* + 1. **Mô hình kiến trúc**

A table of numbers and sizes

Description automatically generated with medium confidence

*Hình 2.9 Mô hình kiến trúc mạng MobileNet*

Mô hình có 30 lớp với các đặc điểm sau:

* Lớp 1: Convolution layer với stride bằng 2
* Lớp 2: Depthwise layer
* Lớp 3: Pointwise layer
* Lớp 4: Depthwise layer với stride bằng 2 (khác với bước 2, dw lớp 2 có stride size bằng 1)
* Lớp 5: Pointwise layer
* ...
* Lớp 30: Softmax, dùng để phân lớp.

Sau mỗi convolution, MobileNet sẽ sử dụng Batch Normalization (BN) và ReLU như hình bên dưới:

A table with text on it

Description automatically generated

*Hình 2.10 Convolution truyền thống (Trái), Depthwise separable convolution với BN và ReLU (Phải)*

* 1. **Mô hình MobileNetV2**

Mô hình MobileNetV2 có một số điểm cải tiến hơn so với phiên bản trước đó, giúp cho nó có độ chính xác cao hơn, số lượng tham số và số lượng các phép tính ít hơn nhờ vào việc sử dụng Inverted Residual Block và việc loại bỏ Non-Linear.

* + 1. **Inverted Residual Block**

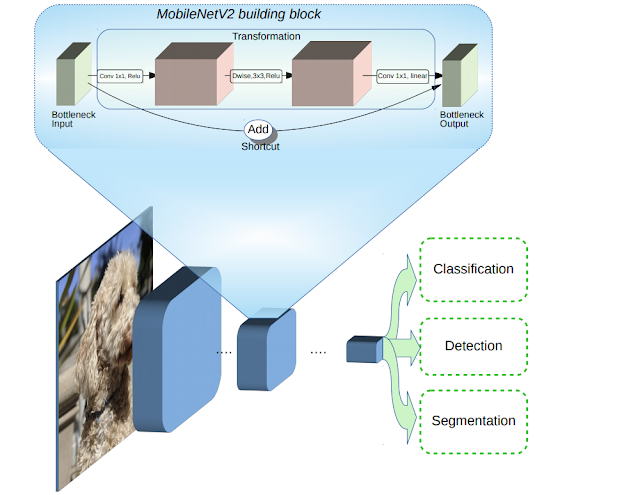
MobileNetV2 cũng sử dụng những kết nối tắt như ở mạng ResNet. Các khối ở layer trước được cộng trực tiếp vào layer liền sau. Nếu coi layer liền trước là x, sau khi đi qua các xử lý tích chập hai chiều ta thu được kết quả f(x) thì output cuối cùng là một residual block có giá trị x+f(x).

A diagram of a diagram

Description automatically generated

*Hình 2.11 Một Residual Block*

Trong MobileNetV2, kết nối tắt được điều chỉnh sao cho số kênh ở input và output của mỗi block residual được thắt hẹp lại, gọi là bottleneck layers. Kiến trúc này ngược lại so với các kiến trúc residual truyền thống vì các layer trung gian có số lượng kênh lớn hơn so với các layer input và output. Điều này tạo ra kiến trúc inverted residual block. Các layer trung gian trong một block biến đổi phi tuyến, nên cần dày hơn để thực hiện nhiều phép biến đổi hơn. Kết nối tắt được thực hiện trên bottleneck input và output chứ không trên các layer trung gian. Các layer bottleneck input và output chỉ ghi nhận kết quả mà không thực hiện biến đổi phi tuyến. Trong một inverted residual block, các layer ở giữa sử dụng Depthwise Separable Convolution để giảm thiểu số lượng tham số, giúp MobileNet có kích thước nhỏ gọn.



*Hình 2.12 Inverted residual block trong các kết nối bottleneck*

Các layer trung gian trong một block sẽ làm nhiệm vụ biến đổi phi tuyến nên cần dày hơn để tạo ra nhiều phép biến đổi hơn. Kết nối tắt giữa các block được thực hiện trên những bottleneck input và output chứ không thực hiện trên các layer trung gian. Do đó các layer bottleneck input và output chỉ cần ghi nhận kết quả và không cần thực hiện biến đổi phi tuyến.

Ở giữa các layer trong một block inverted residual block chúng ta cũng sử dụng những biến đổi tích chập tách biệt chiều sâu để giảm thiểu số lượng tham số của mô hình. Đây cũng chính là bí quyết giúp họ các model MobileNet có kích thước giảm nhẹ.

* + 1. **Loại bỏ Non-Linear**

Một trong những thực nghiệm được tác giả mô hình này ghi nhận đó là việc sử dụng các biến đổi phi tuyến (như biến đổi qua ReLu hoặc sigmoid) tại input và output của các residual block sẽ làm cho thông tin bị mất mát. Kết quả thực nghiệm được biểu diễn bằng biểu đồ bên dưới:

A graph of a line graph

Description automatically generated with medium confidence

*Hình 2.13 Tác động của phi tuyến tính trong bottleneck layers*

Chính vì thế trong kiến trúc của residual block trong mô hình MobileNetV2, tác giả đã loại bỏ hàm phi tuyến tại layer input và output và thay bằng các phép chiếu tuyến tính giúp mô hình đạt kết quả tốt hơn.

# **CHƯƠNG III: THIẾT KẾ HỆ THỐNG**

3.1 Tổng quan hệ thống

Dựa vào mô hình MobileNetV2 đã được huấn luyện để nhận dạng hoa quả từ camera webcam:

Thu thập Dữ liệu:

* Thu thập dữ liệu hình ảnh cho 10 loại trái cây cơ bản (Táo, Xoài, Cam, Cherry, Chuối, Dâu tây, Dứa, Bơ, Dưa hấu, Kiwi). Mỗi loại trái cây cần có một tập hình ảnh đủ lớn và đa dạng.
* Chia tập dữ liệu thành ba phần: tập huấn luyện, tập kiểm tra và tập validation.

Xây dựng mô hình huấn luyện mạng MobileNetV2:

* Sử dụng một mô hình CNN đã được huấn luyện trước của MobileNetV2
* Tùy chỉnh mô hình để phù hợp với dữ liệu bằng cách thay đổi lớp đầu ra (trong project là 224 x 224 x3 ) và thêm lớp Fully Connected mới.
* Tiến hành huấn luyện mô hình trên tập dữ liệu huấn luyện.

Triển khai ứng dụng:

* Sử dụng Flutter hoặc một framework khác để xây dựng giao diện người dùng.
* Tạo một giao diện đơn giản để hiển thị hình ảnh từ camera webcam.
* Tích hợp mô hình CNN đã được huấn luyện vào ứng dụng để nhận diện hoa quả từ hình ảnh trực tiếp từ camera.

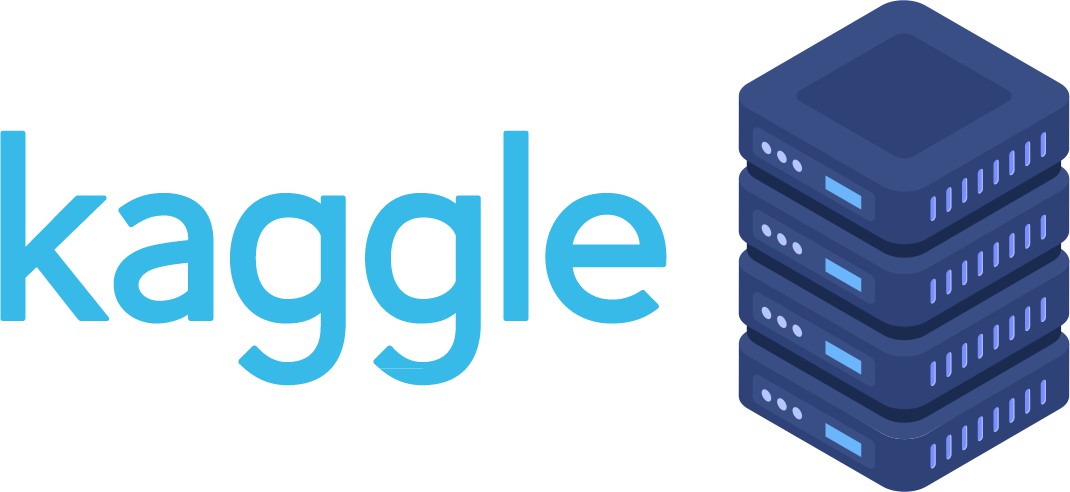
Nhận dạng thời gian thực:

* Sử dụng OpenCV hoặc một thư viện khác để truy cập và xử lý hình ảnh từ camera webcam.
* Truyền hình ảnh đã xử lý từ camera vào mô hình CNN để dự đoán loại trái cây.
* Hiển thị kết quả dự đoán trên giao diện người dùng.

Kiểm tra và tối ưu:

* Kiểm tra ứng dụng để đảm bảo rằng nó hoạt động mượt mà và đáng tin cậy trong các tình huống khác nhau.
* Tối ưu hóa hiệu suất của ứng dụng để đảm bảo thời gian đáp ứng nhanh chóng và tiêu thụ ít tài nguyên.
  1. Dataset

**Lựa chọn tập dữ liệu:**

Nhóm đã lựa chọn tập dữ liệu từ Kaggle vì đây là một nguồn thông tin phong phú chứa nhiều tập dữ liệu khác nhau, bao gồm cả dữ liệu hình ảnh về hoa quả.

*Hình 3.1 Tập dataset kaggle*

**Tìm kiếm và tải tệp dữ liệu:**

Nhóm đã thực hiện tìm kiếm trên Kaggle để tìm các tập dữ liệu hình ảnh về các loại trái cây cơ bản như Táo, Xoài, Cam, Cherry, Chuối, Dâu tây, Dứa, Bơ, Dưa hấu và Kiwi. Sau đó, chúng tôi đã tải xuống tập dữ liệu phù hợp nhất với nhu cầu của dự án.

A screenshot of a computer

Description automatically generated

*Hình 3.2 Folder của 10 class quả*

Dataset có tổng 3326 ảnh được chia ra như sau:

* Gồm 1841 ảnh trong file Train
* Gồm 459 ảnh trong file Validation
* Gồm 1025 ảnh trong tập Test

A black text on a white background

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
| Fruit | Train | Validation | Test |
| Bơ | 184 | 46 | 105 |
| Táo | 184 | 46 | 89 |
| Dứa | 184 | 46 | 105 |
| Dưa hấu | 184 | 46 | 105 |
| Chuối | 184 | 46 | 106 |
| Dâu tây | 184 | 46 | 103 |
| Kiwi | 184 | 46 | 105 |
| Cherry | 184 | 46 | 105 |
| Cam | 184 | 46 | 97 |
| Xoài | 185 | 45 | 105 |

**Kiểm tra và tiền xử lý:**

Sau khi tải xuống, chúng tôi đã tiến hành kiểm tra tập dữ liệu để đảm bảo rằng nó đầy đủ và không bị hỏng. Sau đó, chúng tôi đã tiến hành các bước tiền xử lý như:

* Chia tập dữ liệu thành các tập huấn luyện, kiểm tra và validation.
* Resize tất cả các hình ảnh về kích thước cố định ( 224x224 pixels).
* Chuẩn hóa dữ liệu hình ảnh

**Chuẩn bị nhãn:**

Chuẩn bị các nhãn cho từng hình ảnh, đảm bảo rằng mỗi hình ảnh được gán nhãn chính xác tương ứng với loại trái cây mà nó đại diện.

**Kiểm tra và tổng quan:**

Trước khi sử dụng tập dữ liệu, chúng tôi đã tiến hành kiểm tra và tổng quan nhanh chóng về các thông tin quan trọng như số lượng hình ảnh trong mỗi lớp, phân phối của dữ liệu.

**Lưu trữ và sắp xếp:**

Sau khi chuẩn bị xong, chúng tôi đã lưu trữ tập dữ liệu và nhãn vào các tệp hoặc cơ sở dữ liệu phù hợp để sử dụng trong quá trình huấn luyện mô hình.

**Kiểm tra và đánh giá:**

Cuối cùng, chúng tôi đã tiến hành kiểm tra lại tập dữ liệu và đánh giá xem liệu nó có phù hợp và đủ chất lượng cho mục đích nhận diện hoa quả hay không. Các bước này giúp đảm bảo rằng chúng tôi có một tập dữ liệu đáng tin cậy để sử dụng trong dự án của mình.

Qua quá trình trên, chúng tôi đã có được tập dữ liệu chất lượng và phù hợp để sử dụng trong dự án nhận diện hoa quả. Điều này là cơ sở quan trọng để huấn luyện mô hình nhận diện hoa quả đáng

* 1. Các thư viện sử dụng
     1. **TensorFlow**

TensorFlow là một thư viện mã nguồn mở phổ biến cho máy học và trí tuệ nhân tạo được phát triển bởi Google. Nó cung cấp một nền tảng mạnh mẽ cho việc xây dựng và huấn luyện mô hình máy học trên nhiều loại dữ liệu, từ dữ liệu số (như hình ảnh và văn bản) đến dữ liệu cấu trúc (như dữ liệu dạng bảng).

* + 1. **TensorFlow Keras**

TensorFlow Keras là một API cao cấp trong thư viện máy học TensorFlow, được tích hợp từ phiên bản 1.10 và trở thành API mặc định từ TensorFlow 2.0. Nó cung cấp giao diện đơn giản để xây dựng và huấn luyện mô hình mạng nơ-ron. Các tính năng bao gồm việc đóng gói các lớp, mô-đun và hàm tối ưu hóa thông dụng, cùng với ImageDataGenerator cho việc tạo dữ liệu tăng cường trong huấn luyện mô hình ảnh.

* + 1. **Numpy**

NumPy là một thư viện lập trình cho Python, hỗ trợ trong việc thực hiện các phép toán số học trên mảng và ma trận. Nó cung cấp các đối tượng mảng đa chiều (arrays) và các hàm để làm việc với chúng. NumPy thường được sử dụng để đọc ảnh từ file và biểu diễn ảnh dưới dạng mảng. Mảng NumPy thường được sử dụng để biểu diễn các kênh màu (RGB) của ảnh. Các phép toán NumPy có thể được sử dụng để chuyển đổi giữa các không gian màu và thực hiện các biến đổi màu sắc.

* + 1. **Python OpenCV**

OpenCV-Python là một thư viện mà các nhà phát triển sử dụng để xử lý hình ảnh cho các ứng dụng thị giác máy tính. Thư viện này cung cấp nhiều hàm cho các tác vụ xử lý hình ảnh như đọc và ghi hình ảnh cùng lúc, xây dựng môi trường 3D từ môi trường 2D cũng như chụp và phân tích hình ảnh từ video.

* 1. Kiến trúc mô hình MobileNetV2 trong hệ thống

Sơ đồ này minh họa một "bottleneck layer" cụ thể trong kiến trúc MobileNetV2. Dưới đây là phân tích chi tiết của mỗi thành phần trong sơ đồ nhóm sử dụng :

A diagram of a flowchart

Description automatically generated

A diagram of a computer flowchart

Description automatically generated

*Hình 3.3 Mô hình huấn luyện của kiến trúc MobileNetV2*

Model sẽ lấy dữ liệu huấn luyện mặc định trong data đã huấn luyện imagenet đã huấn luyện sẵn để cho vào

* 1. Xây dựng mô hình huấn luyện

Mô hình MobileNetV2 cho nhận diện Loại Quả, mô hình sử dụng ngôn ngữ python được sử dụng huấn luyện trên google colab trên môi trường CPU máy ảo miễn phí của google:

Google Colab là một môi trường phát triển trực tuyến miễn phí dựa trên trình duyệt được cung cấp bởi Google, cho phép bạn viết và chạy mã Python từ trình duyệt của mình mà không cần cài đặt bất kỳ phần mềm nào trên máy tính cá nhân. Điều này làm cho việc sử dụng các thư viện và gói phần mềm như keras-preprocessing trở nên dễ dàng và tiện lợi.

!pip install keras-preprocessing

Collecting keras-preprocessing

Downloading Keras\_Preprocessing-1.1.2-py2.py3-none-any.whl (42 kB)

━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 42.6/42.6 kB 334.2 kB/s eta 0:00:00

Requirement already satisfied: numpy>=1.9.1 in /usr/local/lib/python3.10/dist-packages (from keras-preprocessing) (1.25.2)

Requirement already satisfied: six>=1.9.0 in /usr/local/lib/python3.10/dist-packages (from keras-preprocessing) (1.16.0)

Installing collected packages: keras-preprocessing

Successfully installed keras-preprocessing-1.1.2

Cài đặt gói keras-preprocessing có thể import các module từ keras\_preprocessing và sử dụng chúng trong mã Python

Sử dụng gói keras-preprocessing trong Google Colab là một cách tiện lợi để xử lý dữ liệu cho việc huấn luyện mô hình neural network mà không cần phải cài đặt và cấu hình môi trường phát triển trên máy tính

import pandas as pd

import plotly.express as px

import seaborn as sns

import tensorflow as tf

from tensorflow import keras

from keras import Sequential

from keras.layers import Dense, Conv2D, MaxPooling2D, Flatten, AveragePooling2D, Dropout, BatchNormalization

from tensorflow.keras import regularizers

from tensorflow.keras.regularizers import l1

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from tensorflow.keras.callbacks import EarlyStopping

from tensorflow.keras.applications import MobileNetV2

import os

import matplotlib.pyplot as plt

import random

import numpy as np

from pathlib import Path

from tensorflow.keras.preprocessing.image import load\_img

import cv2

# mã hóa cả hai cột nhãn và loại

from sklearn.preprocessing import LabelEncoder

# cảnh báo

import warnings

warnings.filterwarnings('ignore')

pandas: Để làm việc với dữ liệu dạng bảng.

plotly.express và seaborn: Để trực quan hóa dữ liệu.

tensorflow và keras: Dùng để xây dựng và huấn luyện mô hình học sâu.

os: Để tương tác với hệ điều hành.

matplotlib.pyplot: Để trực quan hóa dữ liệu với Matplotlib.

numpy: Thư viện cho tính toán số học và ma trận.

pathlib.Path: Để làm việc với đường dẫn và tệp trong Python.

cv2: OpenCV cho xử lý ảnh và thị giác máy tính.

LabelEncoder: Để mã hóa các nhãn văn bản thành số nguyên.

warnings: Để quản lý cảnh báo.

Sau đó tạo ba đối tượng ImageDataGenerator để augmentation dữ liệu huấn luyện, validation và kiểm tra.

# Tạo các đối tượng ImageDataGenerator cho việc augmentation dữ liệu huấn luyện, validation và kiểm tra

train\_datagen = ImageDataGenerator(

    rescale=1./255,

    rotation\_range=40,

    width\_shift\_range=0.1,

    height\_shift\_range=0.1,

    horizontal\_flip=True,

    validation\_split=0.2

)

val\_datagen = ImageDataGenerator(

    rescale=1./255,

    validation\_split=0.2

)

test\_datagen = ImageDataGenerator(rescale=1./255)

Tiếp tạo ba đối tượng generator dữ liệu từ thư mục chứa ảnh, mỗi đối tượng tương ứng với dữ liệu huấn luyện, validation và kiểm tra. Đối tượng `train\_ds` và `validation\_ds` sử dụng `train\_datagen` và `val\_datagen` để augmentation dữ liệu, trong khi `test\_ds` không.

# Tạo các dữ liệu generator từ thư mục chứa dữ liệu ảnh

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

    directory='/content/drive/MyDrive/fruit\_cnn/train',

    batch\_size=32,

    target\_size=(224, 224),

    class\_mode='categorical',

    subset="training",

    seed=123

)

validation\_ds = val\_datagen.flow\_from\_directory(

    directory='/content/drive/MyDrive/fruit\_cnn/train',

    batch\_size=32,

    target\_size=(224, 224),

    class\_mode='categorical',

    subset="validation",

    seed=123

)

test\_ds = train\_datagen.flow\_from\_directory(

    directory='/content/drive/MyDrive/fruit\_cnn/test',

    batch\_size=32,

    target\_size=(224, 224),

    class\_mode='categorical'

)

Sau khi kiểm tra dữ liệu hoàn thành ta tải một mô hình MobileNetV2 được huấn luyện trước từ ImageNet mà không bao gồm lớp phân loại ở đầu (include\_top=False). Đồng thời, định dạng đầu vào của mô hình là (224, 224, 3) và sử dụng cơ chế pooling global average. Các lớp của mô hình đã được đóng băng, nghĩa là không cập nhật trọng số Fully connected của chúng trong quá trình huấn luyện tiếp theo.

# Tải mô hình MobileNetV2 được huấn luyện trước mà không có lớp phân loại top

MobileNetV2\_base = MobileNetV2(weights='imagenet', include\_top=False, input\_shape=(224, 224, 3),

                              pooling='avg')

# Đóng băng các lớp cơ sở đã được huấn luyện trước

MobileNetV2\_base.trainable = False

Xây dựng một mạng neural network trong Keras sử dụng cơ sở MobileNetV2 làm bộ trích xuất đặc trưng và thêm các lớp dense để phân loại 10 loại trái cây.

# Xây dựng mô hình

model = Sequential()

# Thêm cơ sở MobileNetV2 đã được huấn luyện trước

model.add(MobileNetV2\_base)

# Chuẩn hóa dữ liệu bằng Batch Normalization

model.add(BatchNormalization())

# Dropout Layer

model.add(Dropout(0.35))

# Thêm một lớp dense với 220 đơn vị và hàm kích hoạt ReLU

model.add(Dense(220, activation='relu'))

# Thêm một lớp dense với 60 đơn vị và hàm kích hoạt ReLU

model.add(Dense(60, activation='relu'))

# Thêm lớp output với 10 đơn vị và hàm kích hoạt softmax cho phân loại đa lớp

model.add(Dense(10, activation='softmax'))

A screenshot of a computer

Description automatically generated

Mô hình có tổng cộng 2,558,794 tham số (9.76 MB).

Có 2,262,544 tham số không thể huấn luyện lại (non-trainable params), chúng là trọng số của MobileNetV2\_base đã được huấn luyện trước.

Có 298,250 tham số có thể huấn luyện lại (trainable params), chúng là các trọng số của các lớp dense mà bạn đã thêm vào mô hình.

Mỗi lớp dense được mô tả bởi một số lượng đơn vị, ví dụ: lớp dense đầu tiên có 220 đơn vị.

Hàm kích hoạt của lớp cuối cùng là softmax, đảm bảo rằng đầu ra là một phân phối xác suất được chuẩn hóa (ở mô hình sẽ là 10 softmax tương ứng 10 class loại quả)

A screenshot of a computer program

Description automatically generated

Quá trình huấn luyện đã hoàn thành với tổng cộng 50 epochs.

Mô hình đã được huấn luyện qua 50 epoch với độ chính xác khá cao với 97.83 trên tập train và 92.81 trên tập dữ liệu mới, hàm mất mát cũng giảm xuống theo từng epoch nên không bị quá overfiting

Mô hình đã được huấn luyện 10 class loại quả ở mức tốt

A graph of a graph showing a loss

Description automatically generated with medium confidence

*Hình 3.4 Sơ đồ hiển thị hàm mất mát của model trên tập train và test*

A graph of a graph showing the value of a model

Description automatically generated with medium confidence

*Hình 3.5 Sơ đồ hiển thị độ chính xác của model trên tập train và test*

* 1. Thử nghiệm

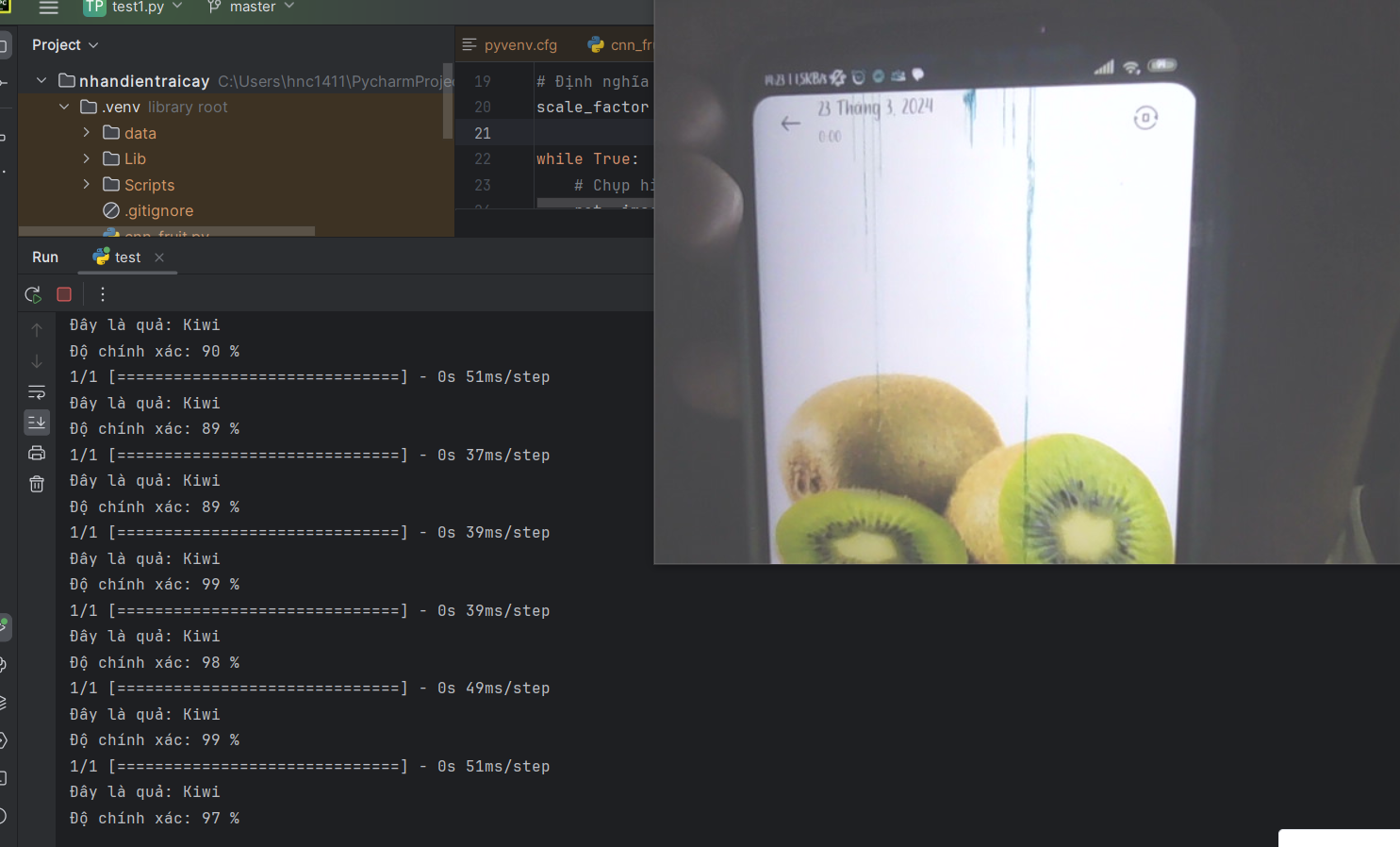
Chạy file test.py sau đó webcam của máy tính sẽ mở ra và nhận diện loại hoa quả theo thời gian thực khi từng hình ảnh đưa vào rồi nhận diện( nó sẽ hiển thị độ chính xác của từng 10 class quả )

VD:

A screenshot of a computer

Description automatically generated

Hình 3.6 Thử nghiệm chương trình (Dưa hấu)



Hình 3.7 Thử nghiệm chương trình (Kiwi)

# **CHƯƠNG IV: NHẬN XÉT CHUNG**

**4.1 Kết quả thu được**

Thử nghiện với test set có lệ chính xác khá cáo, đạt khoảng 92% với tệp validation và 97% với train.

Những ảnh có tỉ lệ nhận diện sai cao thường rơi vào 2 lớp:“cherri ” và “dâu tây”.

Những ảnh có tỉ lệ nhận diện đúng cao nằm ở các lớp 8 lớp khác.

**4.2 Đánh giá**

Model vẫn còn hạn chế, chưa nhận diện được loại quả quá chính xác nhưng không bị overfiting quá.

## **4.3 Nguyên nhân**

Một số loài có màu sắc, hình dáng tương tự nhau

Số lượng ảnh trong dataset còn ít, chỉ khoảng ~230 ảnh 1 dataset train 1 loại quả

Chất lượng ảnh kém, vỡ, màu sắc ảnh gần tương tự nhau

## **4.3 Kết luận**

Mô hình đã được huấn luyện và nhận diện 10 loại quả với độ chính xác tương đối cao 98% mắc dù lượng data sẽ còn hạn chế, Mô hình sử dụng MobileNetV2 có giá trị học và thực tiến ổn nếu phát triển thành 1 phầm mềm ứng dụng để nhận diện nhiều loại quả khác nhau nữa, nhưng yêu cầu khá nhiều về dataset.

# TÀI LIỆU THAM KHẢO

## *Tiếng Việt*

1. *Trần Tuấn Linh. (2017). Ứng dụng nhận dạng hoa quả cho điện thoại thông minh dựa trên hình ảnh.*
2. *Vũ Hữu Tiệp. (2017).Machine Learning cơ bản.* [*http://machinelearningcoban.com/general/2017/02/06/featureengineering/*](http://machinelearningcoban.com/general/2017/02/06/featureengineering/)

## *Tiếng Anh*

1. *Andrej Karpathy. CS231n Convolutional Neural Networks for Visual Recognition*

*- Image Classification.* [*http://cs231n.github.io/classification/*](http://cs231n.github.io/classification/)

1. *Sadrnia, H., Rajabipour, A., Jafary, A., Javadi, A., & Mostofi, Y. (2007). Classification and analysis of fruit shapes in long type watermelon using image processing. Int J Agric Biol, 9(1), 68–70.*
2. *Fu, L., Sun, S., Li, R., & Wang, S. (2016). Classification of kiwifruit grades based on fruit shape using a single camera. Sensors (Switzerland), 16(7), 1–14.*
3. *Seng, W. C., & Mirisaee, S. H. (2009). A new method for fruits recognition system. Proceedings of the 2009 International Conference on Electrical Engineering and Informatics, ICEEI 2009, 1, 130–134.*
4. *Arivazhagan, S., Shebiah, R. N., Nidhyanandhan, S. S., & Ganesan, L. (2010). Fruit Recognition using Color and Texture Features. Information Sciences, 1(2), 90–94.*
5. *Zhang, Y., & Wu, L. (2012). Classification of fruits using computer vision and a multiclass support vector machine. Sensors (Switzerland), 12(9), 12489–12505.*
6. *Naskar, S. (2015). A Fruit Recognition Technique using Multiple Features and Artificial Neural Network, 116(20), 23–28.*
7. *GilPress. (2016). Visually Linking AI, Machine Learning, Deep Learning, Big Data and Data Science What’s The Big Data ?* [*https://whatsthebigdata.com/2016/10/17/visually-linking-ai-machine-learning-*](https://whatsthebigdata.com/2016/10/17/visually-linking-ai-machine-learning-deep-learning-big-data-and-data-science/)[*deep-learning-big-data-and-data-science/*](https://whatsthebigdata.com/2016/10/17/visually-linking-ai-machine-learning-deep-learning-big-data-and-data-science/)
8. [*Mobilenetv2 next generation of On-Device computer vision networks - googleblog*](https://ai.googleblog.com/2018/04/mobilenetv2-next-generation-of-on.html)
9. [*Mobilenetv2 inverted residuals and linear bottlenecks -*](https://towardsdatascience.com/mobilenetv2-inverted-residuals-and-linear-bottlenecks-8a4362f4ffd5)
10. [*Squeezenet image classification*](https://towardsdatascience.com/review-squeezenet-image-classification-e7414825581a)
11. [*Review squeeze and excitation network*](https://towardsdatascience.com/review-senet-squeeze-and-excitation-network-winner-of-ilsvrc-2017-image-classification-a887b98b2883)
12. [*MobileNetV2: Inverted Residuals and Linear Bottlenecks - Mark Sandler, Andrew Howard*](https://arxiv.org/abs/1801.04381)