TRƯỜNG ĐẠI HỌC CẦN THƠ

ĐƠN VỊ: TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

MINH CHỨNG SẢN PHẨM NGHIÊN CỨU KHOA HỌC

Sản phẩm giao nộp theo Hợp đồng thuê khoán chuyên môn 01/2023/HĐ-NCKH

Mã số: TSV2023-125

TÊN ĐỀ TÀI

NGHIÊN CỨU CHẾ TẠO XE LĂN ĐIỀU KHIỂN BẰNG MẮT

I. Khái niệm học sâu và thị giác máy tính

1. Giới thiệu về Deep Learning

Deep Learning là một phần quan trọng của lĩnh vực trí tuệ nhân tạo, là một phương pháp học máy mạnh mẽ được thúc đẩy bởi kiến thức về cấu trúc và hoạt động của não người, được xem là một lĩnh vực con của Machine Learning. Nó được thiết kế để mô hình hóa và giải quyết các vấn đề phức tạp bằng cách sử dụng các mạng neural nhân tạo sâu (Deep Neural Networks). Điều đặc biệt về Deep Learning là khả năng tự học cấu trúc và đặc điểm của dữ liệu thông qua việc tự điều chỉnh các trọng số và tham số của mạng neural.

Một điểm đặc biệt của Deep Learning là khả năng xử lý dữ liệu không cấu trúc, như hình ảnh, âm thanh, văn bản và video. Bằng cách sử dụng các lớp neural ẩn trong mạng, Deep Learning có khả năng tự động học lên các đặc trưng phức tạp từ dữ liệu đầu vào. Điều này đã tạo ra những đột phá lớn trong nhiều lĩnh vực, bao gồm nhận dạng hình ảnh, xử lý ngôn ngữ tự nhiên, dự đoán chuỗi thời gian và nhiều ứng dụng khác.

Các mô hình Deep Learning như Convolutional Neural Network (CNN) và Recurrent Neural Network (RNN) đã đạt được những thành tựu ấn tượng trong việc giải quyết các vấn đề phức tạp. Chẳng hạn, trong lĩnh vực nhận dạng hình ảnh, Deep Learning đã giúp máy tính nhận biết vật thể, khuôn mặt và thậm chí đạt độ chính xác cao hơn cả con người.

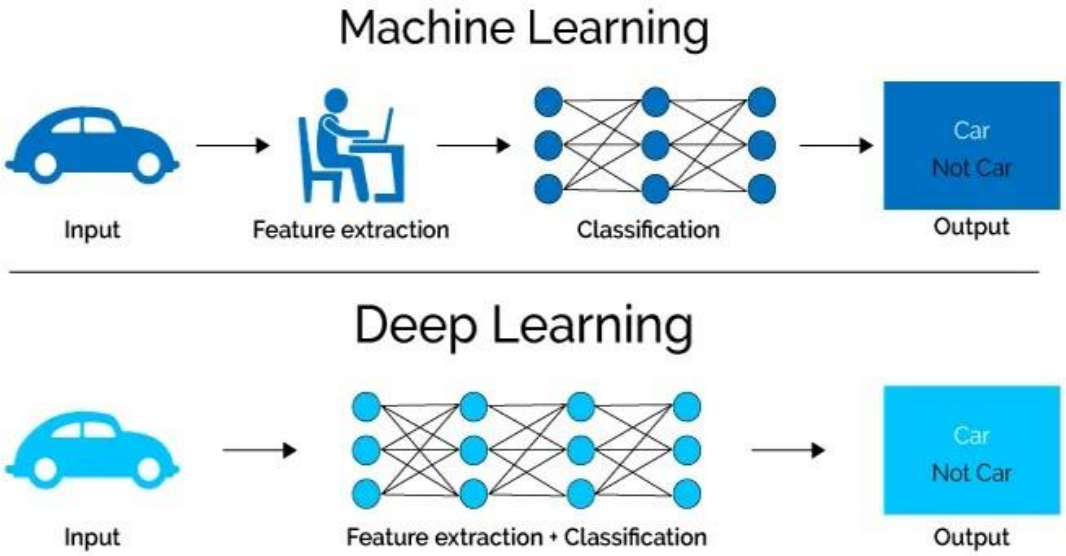
Tuy Deep Learning mang lại nhiều lợi ích, nhưng cũng đòi hỏi tài nguyên tính toán lớn, đặc biệt là khi xử lý các mạng neural sâu và dữ liệu lớn. Điều này đã thúc đẩy sự phát triển của các phần cứng và công nghệ mới để hỗ trợ việc huấn luyện và triển khai các mô hình Deep Learning.

Tóm lại, Deep Learning đang là một lĩnh vực đóng vai trò quan trọng trong cách chúng ta tiếp cận và giải quyết các vấn đề phức tạp sử dụng máy tính. Nhờ khả năng tự học và mô hình hóa dữ liệu phức tạp, nó đã đưa ra những giải pháp sáng tạo cho nhiều lĩnh vực và tiếp tục định hình tương lai của trí tuệ nhân tạo.

2. Cách thức hoạt động của Deep Learning

Deep Learning sử dụng mạng neural nhân tạo được xây dựng để mô phỏng khả năng tư duy của não bộ con người. Một mạng sẽ bao gồm nhiều lớp (layer) khác nhau, số lượng lớp càng nhiều mạng sẽ càng sâu. Trong từng lớp sẽ chứa các nút mạng (node) và các lớp được liên kết kề với nhau. Mỗi kết nối giữa các nút mạng chứa một trọng số tương ứng, trọng số này càng cao thì ảnh hưởng của kết nối này đến mạng neural càng lớn.

Mỗi neural chứa một hàm kích hoạt có nhiệm vụ chuẩn hóa đầu ra từ neural này. Dữ liệu được đưa vào mạng sẽ đi qua tất cả các lớp và trả về kết quả ở lớp cuối cùng, gọi là output layer.



Hình 1: Cách hoạt động của Deep Learning

Để dễ hình dung về Deep Learning, thông qua ví dụ sau:

Làm sao để nhận biết hình nào là hình vuông trong một tập hình? Để giải quyết vấn đề này điều cần thiết đầu tiên là kiến thức tối thiểu về hình vuông, cụ thể là các đặc trưng

về hình vuông khác biệt với các dạng hình khối khác. Khi đã có đủ điều kiện cần, bắt đầu kiểm tra:

- Bước 1: Hình này có 4 cạnh không?
- Bước 2: Nếu hình này có 4 cạnh thì 4 cạnh này có kết nối với nhau không?
- Bước 3: Nếu hình này có 4 cạnh kết nối với nhau, vậy các góc giữa 4 cạnh này có vuông không?
- Bước 4: Nếu hình này có 4 cạnh, vuông góc giữa các cạnh, vậy kích thích của 4 cạnh này có bằng nhau không?

Thông qua việc đi từng bước trong 4 bước cơ bản trên sẽ trả lời được hình nào là hình vuông. Deep Learning cũng hoạt động tương tự như vậy, nhưng ở một quy mô lớn hơn như nhận diện vật thể, nhận diện con người, ước tính khoảng cách,…

3. Những vấn đề Deep Learning có thể giải quyết

Một số vấn đề Deep Learning đã giải quyết được trong thực tế có thể kể đến:

Xe tự lái: Hiện nay một số mẫu xe tự lái đã được phép vận hành, các công nghệ tự lái sẽ được xây dựng dựa trên các mạng neural cao cấp, mô hình Deep Learning được sử dụng sẽ nhận diện các đối tượng, vật thể ở môi trường xung quanh xe, tính toán khoảng cách giữa xe so với các đối tượng, vật thể hay phương tiện xung quanh khác. Thêm vào đó là xác định làn đường, tín hiệu giao thông,… từ đó mô hình sẽ đưa ra các quyết định tối ưu và nhanh chóng nhất có thể. Một trong các hãng xe đi đầu trong việc sản xuất xe tự lái hiện nay chính là Tesla.

Phân tích cảm xúc: Deep Learning cũng có thể phân tích cảm xúc con người thông qua việc xử lý ngôn ngữ tự nhiên, phân tích văn bản và thống kê. Các mô hình này có thể được ứng dụng để phán đoán cảm xúc khách hàng thông qua các bình luận, đánh giá, tweet,… từ đó đưa ra những chiến lược kinh doanh và marketing phù hợp cho từng nhóm đối tượng cụ thể.

Trợ lý ảo: Trợ lý ảo hiện đang rất phổ biến trên toàn cầu, chúng được tích hợp trực tiếp vào điện thoại, máy tính, hay robot. Một trong số đó có thể kể đến như Cortana, Siri, Google Assistant, robot Vector,… Các trợ lý ảo này được xây dựng dựa trên Deep Learning với các thuật toán nhận diện văn bản, nhận diện giọng nói, xử lý ngôn ngữ tự nhiên,… từ đó phân tích và đưa ra kết quả tối ưu cho người dùng.

Mạng xã hội: Từ lâu các trang mạng xã hội như Facebook, Zalo, Tiktok, Twitter,.. đã trở lên quá phổ biến trong thời đại công nghệ 4.0. Các trang mạng này cũng áp dụng

các thuật toán Deep Learning để cải thiện dịch vụ của mình. Một số có thể kể đến như nhận diện khuôn mặt, khi một tấm hình được đăng tải lên, các mô hình sẽ được áp dụng để nhận diện các khuôn mặt xuất hình trong hình là ai dựa vào các dữ liệu về hình ảnh được tải lên trước đó. Thông qua các dữ liệu về tìm kiếm, cảm xúc, bình luận của người dùng. Các trang mạng này cũng phân tích và đề xuất, gợi ý các sản phẩm, mặt hàng liên quan phù hợp thông qua các sàn thương mại điện tử có liên kết.

Ngoài ra còn rất nhiều phương diện trong cuộc sống được áp dụng Deep Learning trên toàn thế giới. Sự phát triển của Deep Learning đã giải quyết được nhiều vấn đề liên quan, từ đó cải thiện được chất lượng sống, thời gian, nhân lực,…

4. Convolutional Neural Network (CNN) – Mạng neural tích chập

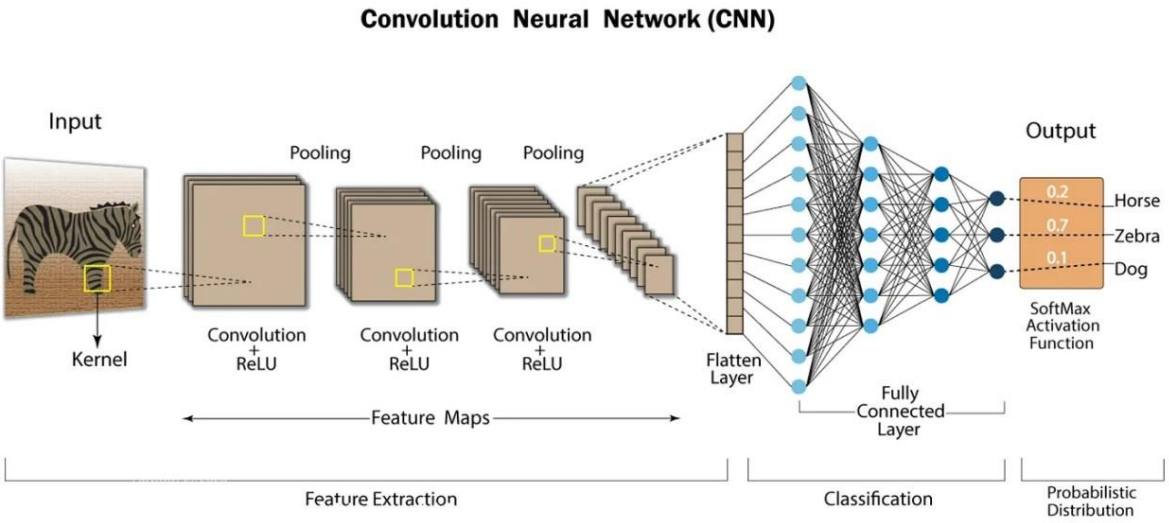
4.1. Giới thiệu

Convolutional Neural Network (CNN) là một loại mạng neural nhân tạo được thiết kế đặc biệt để xử lý và phân loại hình ảnh. CNN đã đạt được sự phát triển đáng kể trong lĩnh vực thị giác máy tính và xử lý hình ảnh, và chúng đã góp phần quan trọng vào sự thành công của nhiều ứng dụng thời gian thực như nhận dạng khuôn mặt, xe tự lái, và phát hiện đối tượng. CNN được lấy cảm hứng từ cách mà não người xử lý hình ảnh thông qua các tầng thụ động và tiếp thu thần kinh.

CNN đã đạt được kết quả ấn tượng trong nhiều cuộc thi và ứng dụng thực tế như phát hiện vật thể, phân loại hình ảnh y tế và thậm chí cả trong tự động đặt tên cho hình ảnh. Chúng đã trở thành một công cụ quan trọng trong lĩnh vực trí tuệ nhân tạo và thị giác máy tính, giúp chúng ta hiểu và xử lý hình ảnh một cách thông minh và hiệu quả.

4.2. Cấu trúc mạng CNN

Mạng CNN là một tập các lớp tích chập chồng lên nhau và sử dụng các hàm phi tuyến như ReLU và tanh để kích hoạt các trọng số trong các nút mạng. Mỗi một lớp sau khi thông qua các hàm kích hoạt sẽ tạo ra các thông tin trừu tượng hơn cho các lớp tiếp theo.

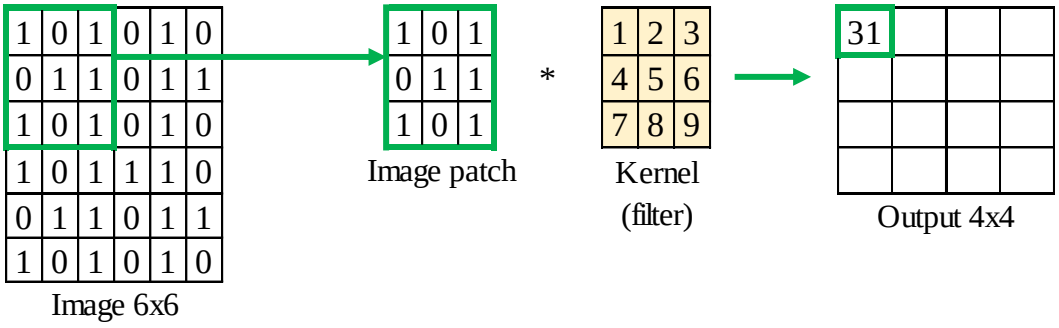


Hình 2: Cấu trúc mạng CNN

Số lượng các lớp tích chập sẽ tùy thuộc vào người xây dựng và sự phức tạp của dữ liệu đầu vào. Một mạng CNN cơ bản thường sẽ có 3 lớp chính: lớp tích chập (Convolutional layer), lớp pooling và lớp kết nối đầy đủ (Fully connected).

4.2.1. Lớp tích chập (Convolutional layer)

Lớp này sử dụng các bộ lọc để thực hiện phép tích chập khi đưa qua đầu vào theo các chiều của nó. Cụ thể, với đầu vào là một hình ảnh, được xem như một ma trận dữ liệu, lớp này sẽ là một ma trận quét qua ma trận dữ liệu, sau đó đưa vào các hàm kích hoạt phi tuyến như ReLU để thu lại một ma trận mới được gọi là feature map. Các siêu tham số của bộ lọc này là kích thước bộ lọc và bước nhảy (stride).



Hình 3: Ví dụ về lớp tích chập

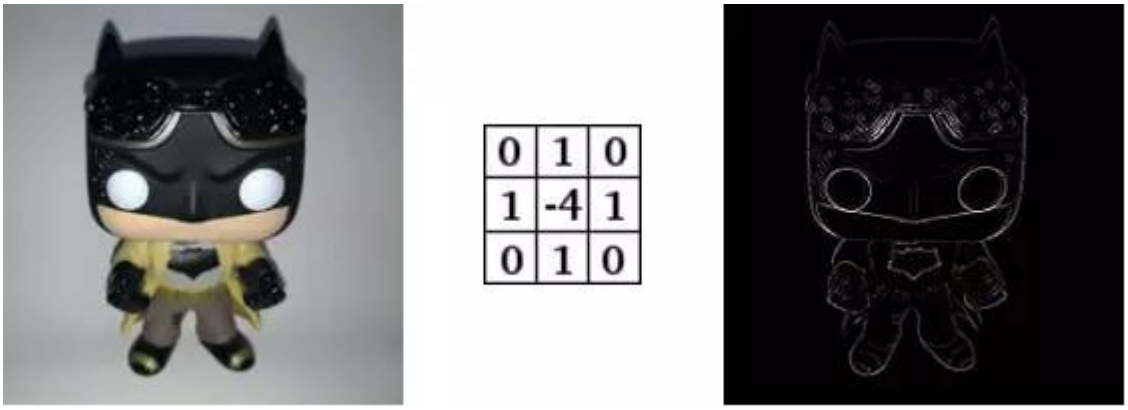
Giả sử đầu vào là một hình ảnh với ma trận dữ liệu tương đương kích thước 6x6. Với kernel là ma trận quét kích thước 3x3, ma trận quét sẽ quét lần lượt theo bước nhảy (stride) trên ma trận dữ liệu. Mỗi lần quét sẽ lấy được một ma trận con (Image patch) trong ma trận dữ liệu có kích thước bằng với kích thước kernel. Sau đó tiến hành tích chập ma trận con với ma trận quét và trả về kết quả cho ma trận đầu ra kích thước 4x4. Ví dụ với minh họa trên ta có:

Output[0][0] = (1x1)+(0x2)+(1x3)+(0x4)+(1x5)+(1x6)+(1x7)+(0x8)+(1x9)

= 1 + 0 + 3 + 0 + 5 + 6 + 7 + 0 + 9

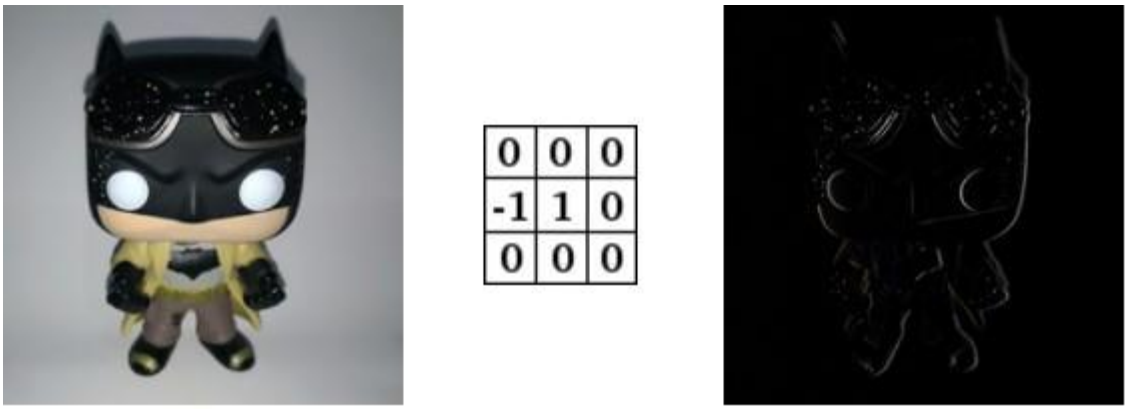
= 31

Các giá trị trong ma trận kernel không cố định, tùy vào mục đích trích xuất đặc tính hình ảnh để tìm các giá trị này. Ví dụ với 3 kernel bên dưới, khi tích chập sẽ cho ra 3 kết quả với đặc tính riêng biệt từ input ban đầu:



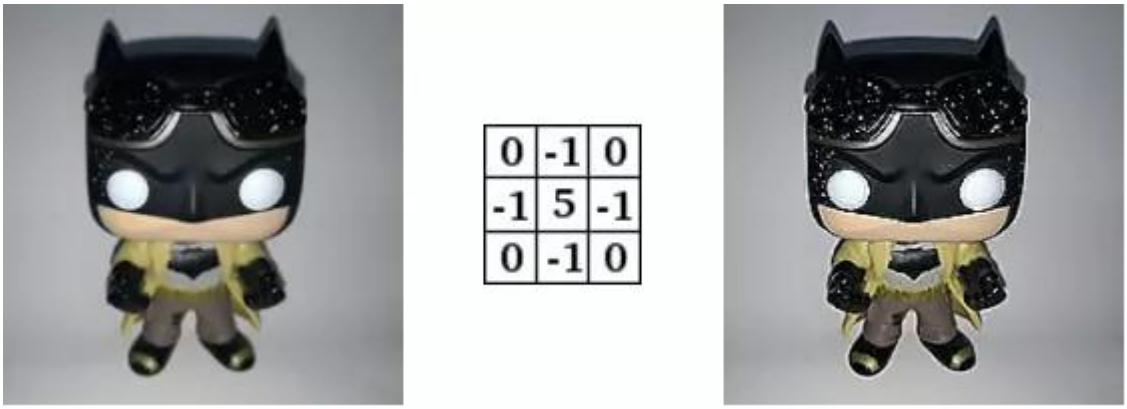
Hình 4: Ví dụ về kernel làm sắc nét

Convolution Sharpen (tạm dịch: Làm sắc nét bằng tích chập): là một kỹ thuật xử lý hình ảnh thường được sử dụng để tăng cường độ tương phản và sắc nét của một hình ảnh. Kernel được áp dụng có khả năng tạo ra một hiệu ứng tăng cường cạnh và độ tương phản của hình ảnh.



Hình 5: Ví dụ về kernel tăng cường biên độ

Convolution Edge Enhance (tạm dịch: Tăng cường biên độ bằng tích chập): đây là kỹ thuật xử lý hình ảnh dùng để tạo ra hiệu ứng tăng cường cạnh và biên độ trong hình ảnh. Kernel của Convolution Edge Enhance được thiết kế để tôn lên sự khác biệt giữa các vùng có độ tương phản cao và thấp, tạo ra hiệu ứng tăng cường cạnh.

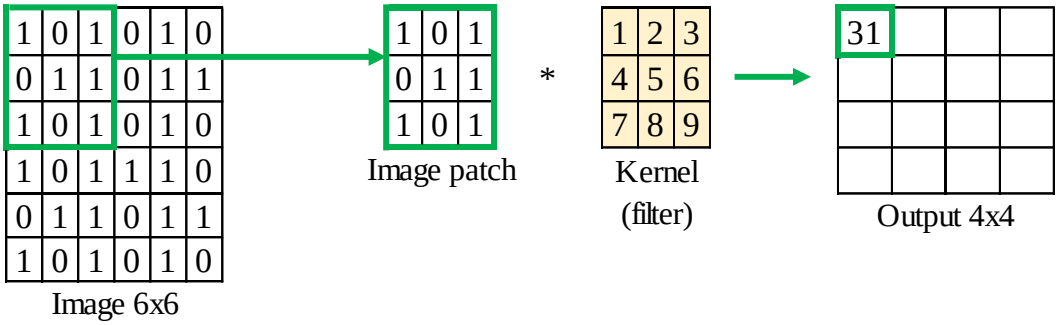


Hình 6: Ví dụ về kernel phát hiện biên độ

Convolution Edge Detect (tạm dịch: Phát hiện biên độ bằng phép tích chập) là một kỹ thuật xử lý hình ảnh dùng để tìm và làm nổi bật các biên hoặc đường nét trong hình ảnh. Kỹ thuật này sử dụng phép tích chập và một kernel được thiết kế đặc biệt để xác định các vùng có sự thay đổi đột ngột trong độ sáng hoặc màu sắc, coi chúng như là các đường biên. Khi kernel này được áp dụng lên hình ảnh, nó sẽ tạo ra một hình ảnh mới trong đó các biên sẽ được biểu thị bằng các điểm sáng hoặc tối tùy thuộc vào hướng thay đổi độ sáng hay màu sắc.

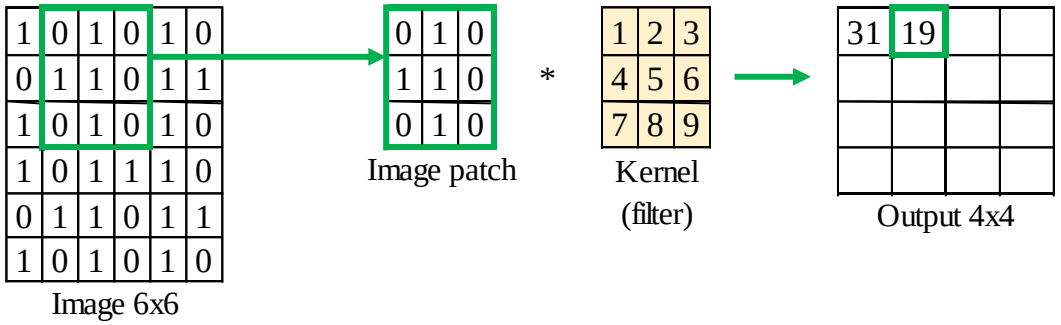
Các siêu tham số:

- Stride: là sải bước hay độ trượt của kernel khi quét qua ma trận dữ liệu đầu nào. Giả sử với Stride(1, 1), khi kernel quét qua ma trận đầu vào theo chiều ngang, mỗi lần trượt sẽ dịch kernel sang một ô, lần lượt như vậy cho đến cuối. Lúc này kernel sẽ trở lại vị trí ban đầu nhưng dịch xuống một ô và tiếp tục quét theo chiều ngang với Stride = 1. Quay lại ví dụ ban đầu:



Hình 7: Ví dụ về stride

Sau khi kernel tích chập với ma trận con 3x3 từ ma trận đầu vào 6x6 và trả về kết quả cho ma trận đầu ra 4x4: Output[0][0] = 31.



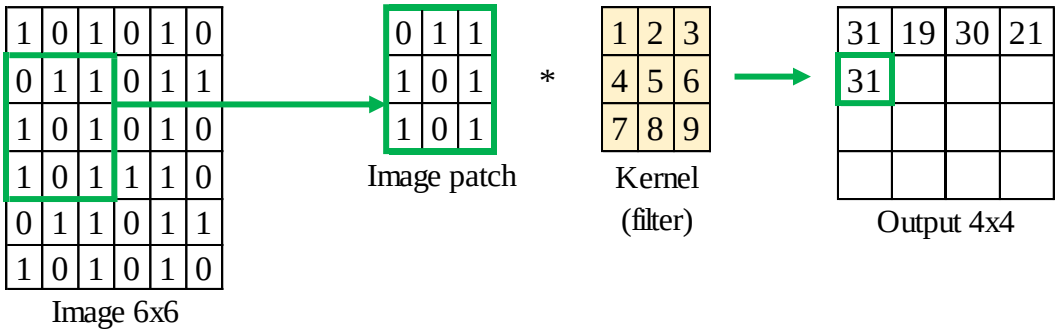
Hình 8: Ví dụ về stride

Trượt kernel sang phải một ô để lấy ma trận con tiếp theo, tiếp tục tích chập với kernel và trả về kết quả cho ma trận đầu ra:

Output[0][1] = (0x1)+(1x2)+(0x3)+(1x4)+(1x5)+(0x6)+(0x7)+(1x8)+(0x9)

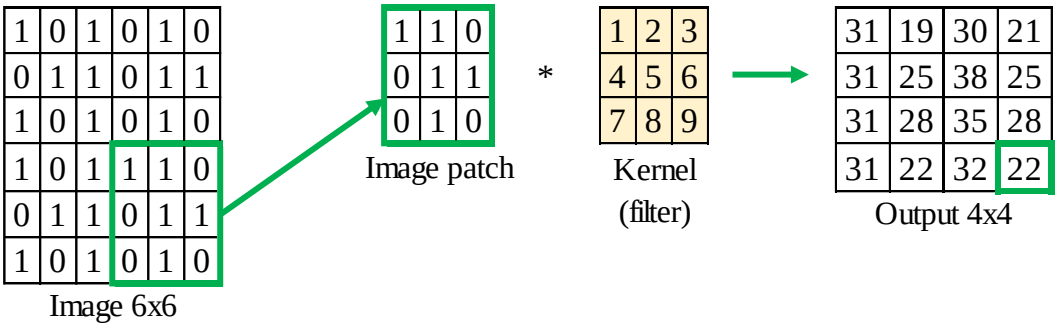
= 0 + 2 + 0 + 4 + 5 + 0 + 0 + 8 + 0

= 19



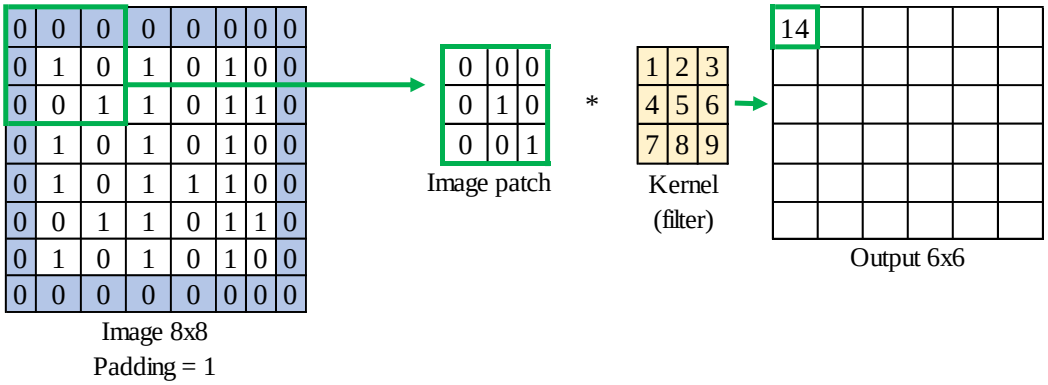
Hình 9: Ví dụ về stride

Sau khi trượt kernel đến cuối tương đương với hàng đầu của ma trận đầu ra. Kernel sẽ trở về vị trí ban đầu nhương trượt xuống một ô và tiếp tục tích chập theo chiều ngang để trả về kết quả cho hàng thứ hai của ma trận đầu ra. Liên tục quét kernel qua ma trận đầu vào cho đến hết sẽ thu được ma trận đầu ra đầy đủ:



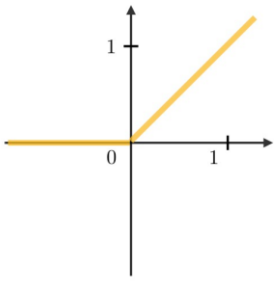
Hình 10: Ví dụ về stride

- **Padding:** Nếu chọn stride và kích thước của kernel càng lớn thì kích thước của ma trận đầu ra càng nhỏ, một vấn đề rắc rối có thể xảy ra là mất một số điểm ảnh trên biên của ảnh, sự mất mát này có thể tích lũy dần ra nhiều lớp tích chập. Và có một giải pháp cho vấn đề này là Padding, ta sẽ chèn các điểm ảnh bọc xung quanh đường biên của ảnh ảnh đầu vào, nhờ đó làm tăng kích thước sử dụng của bức ảnh. Ví dụ với Padding = 1, tiến hành bọc các điểm ảnh với giá trị thêm vào (thường là 0) xung quanh biên của ma trận ảnh đầu vào như sau:



Hình 11: Ví dụ về padding

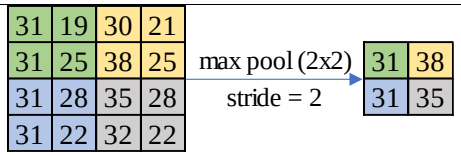
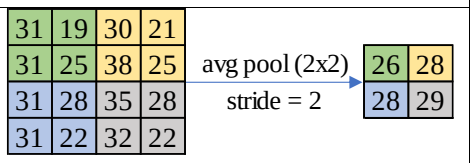
Các lớp tích chập có tác dụng trích xuất các đặc trưng của hình ảnh đầu vào, sau khi đi qua các lớp tích chập, các thông tin được trích xuất từ các lớp tích chập sẽ được đưa vào các hàm kích hoạt phi tuyến để tạo tính phi tuyến và không gian trước khi đi đến các lớp tiếp theo. Nếu không có các hàm kích hoạt phi tuyến này, dù mạng neural có nhiều lớp đến đâu vẫn chỉ có hiệu quả như một lớp tuyến tính. Một số hàm phi tuyến được sử dụng trong CNN như ReLU, tanh, sigmoid,… Trong đó ReLU thường được sử dụng vì nó có hiệu suất tốt.



Hình 12: Đồ thị hàm ReLU: g(z)=max(0,z)

4.2.2. Lớp Pooling

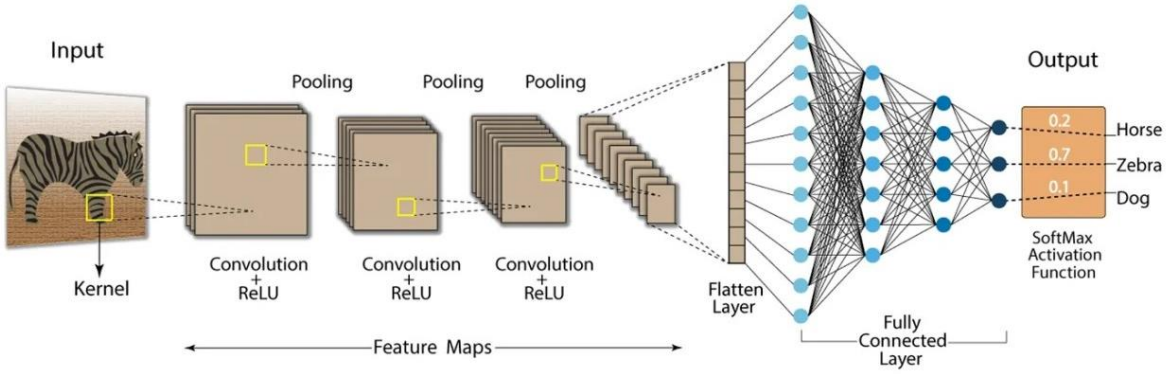
Lớp Pooling thường được sử dụng sau lớp tích chập và hàm kích hoạt phi tuyến, lớp này là một phép downsampling, giúp tăng tính bất biến không gian, giảm kích thước đầu vào và chi phí tính toán. Trong đó, max pooling và average pooling là những dạng pooling đặc biệt, hay được sử dụng (thường là max pooling). Cách thức hoạt động của lớp này gần giống với lớp tích chập. Cụ thể, lớp này sử dụng ma trận quét với stride tương tự như lớp tích chập, nhưng mỗi lần quét qua sẽ lấy giá trị lớn nhất (đối với max pooling) hoặc giá trị trung bình (đối với average pooling) để trả về ma trận đầu ra.

Kiểu	Max Pooling	Average Pooling
Chức năng	Từng phép pooling chọn giá trị lớn nhất trong khu mà nó đang được áp dụng	Từng phép pooling tính trung bình các giá trị trong khu vực mà nó được áp dụng
Minh họa		
Nhận xét	Bảo toàn các đặc trưng đã phát hiện Thường được sử dụng	Giảm kích thước feature map Được sử dụng trong mạng LeNet

4.2.3. Lớp kết nối đầy đủ (Fully Connected)

Các lớp kết nối đầy đủ thường được đặt ở cuối của kiến trúc CNN, sau một loạt các lớp tích chập và lớp pooling. Chúng thường được gọi là "dense" (đầy đủ) vì mọi neural trong một lớp này được kết nối với mọi neural trong lớp liền trước. Điều này có nghĩa là mỗi neural trong một lớp kết nối đầy đủ nhận đầu vào từ tất cả các neural trong lớp trước đó.

Lớp kết nối đầy đủ đóng vai trò quan trọng trong việc tổng hợp thông tin và học các mối quan hệ phức tạp giữa các đặc trưng để thực hiện các tác vụ như phân loại, dự đoán và điều chỉnh mô hình trong CNN.



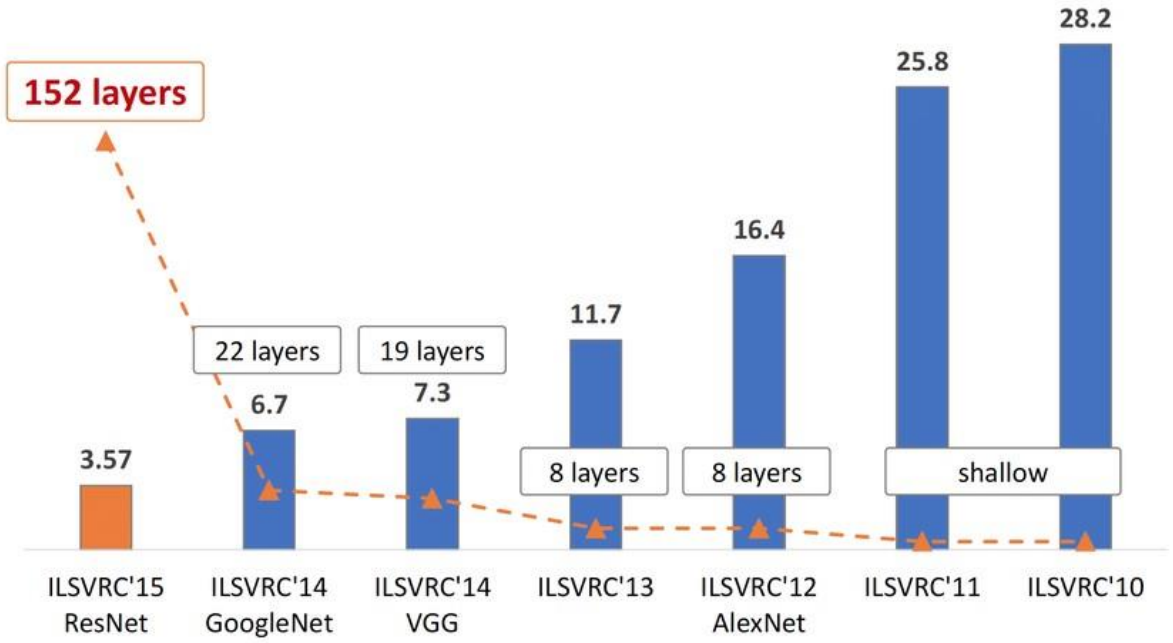
Hình 13: Cấu trúc mạng CNN

4.3. Mạng xương sống CNN (CNN backbone)

CNN backbone (Convolutional Neural Network backbone) là một phần quan trọng trong kiến trúc của mạng neural dùng cho các nhiệm vụ liên quan đến thị giác máy tính, chẳng hạn như nhận dạng vật thể trong hình ảnh, phân loại ảnh, hoặc các tác vụ khác trong lĩnh vực thị giác máy tính.

Backbone thường bao gồm một loạt các lớp convolutional và lớp pooling (thường là Convolutional Layers và MaxPooling Layers) được xếp chồng lên nhau. Các lớp này được thiết kế để trích xuất các đặc trưng quan trọng từ hình ảnh đầu vào, bằng cách áp dụng các phép tích chập và tổng hợp thông tin từ các vùng nhỏ của ảnh để tạo ra biểu đồ đặc trưng (feature map). Các feature map này sau đó được sử dụng bởi các lớp mạng neural sau để thực hiện các tác vụ như phân loại hoặc nhận dạng.

Một số CNN backbone phổ biến trong lĩnh vực thị giác máy tính bao gồm AlexNet, VGG, ResNet,... Mỗi loại backbone có các đặc điểm riêng biệt và hiệu suất khác nhau, tùy thuộc vào nhiệm vụ cụ thể và khả năng tính toán. Việc chọn backbone phù hợp là một phần quan trọng trong việc xây dựng các mô hình thị giác máy tính hiệu quả.



Hình 14: Biểu đồ phân bố kết quả sử dụng mạng CNN trên ImageNet

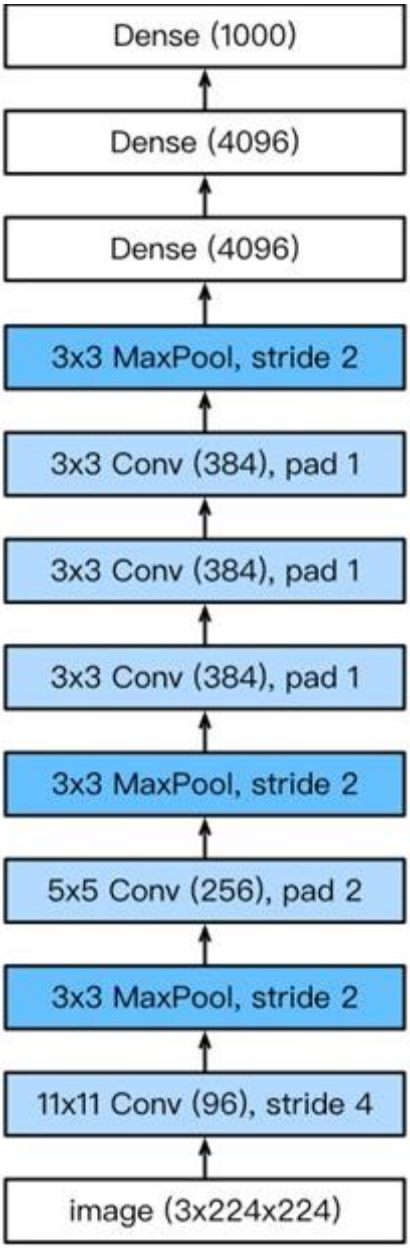
Biểu đồ trên phân bố kết quả sử dụng mạng CNN trong những năm từ 2010 đến 2015 trên bộ ImageNet. Từ năm 2012, AlexNet bắt đầu thắng thế giúp giảm xác suất lỗi trên bộ ImageNet xuống còn 16.4. Đến năm 2014 bắt đầu sử dụng mạng VGG và đã giảm

xác suất lỗi xuống 7.3, sau đấy là GoogleNet giảm xuống 6.7 và cuối cùng là ResNet giảm chỉ còn 3.57. Mạng ResNet cũng là mạng được sử dụng chính trong kết quả của bài nghiên cứu khoa học này.

4.3.1. Mạng neural tích chập sâu AlexNet

AlexNet là một mạng neural convolutional (CNN) nổi tiếng và đột phá trong lĩnh vực thị giác máy tính và học sâu. Đây là một trong những mạng neural đầu tiên và mạnh mẽ nhất được giới thiệu bởi Alex Krizhevsky, Ilya Sutskever và Geoffrey Hinton trong cuộc thi ImageNet Large Scale Visual Recognition Challenge (ILSVRC) năm 2012. Mạng này đã đặt nền móng cho sự phát triển mạnh mẽ của Deep Learning trong thị giác máy tính.

AlexNet đã giành chiến thắng trong cuộc thi ILSVRC 2012 với kết quả xuất sắc và đã mở đường cho việc sử dụng deep learning trong nhiều ứng dụng thị giác máy tính. Cấu trúc của AlexNet đã trở thành một kiểu mẫu cho các kiến trúc mạng CNN sau này và đóng vai trò quan trọng trong sự phát triển của lĩnh vực này.



Hình 15: Cấu trúc mạng AlexNet

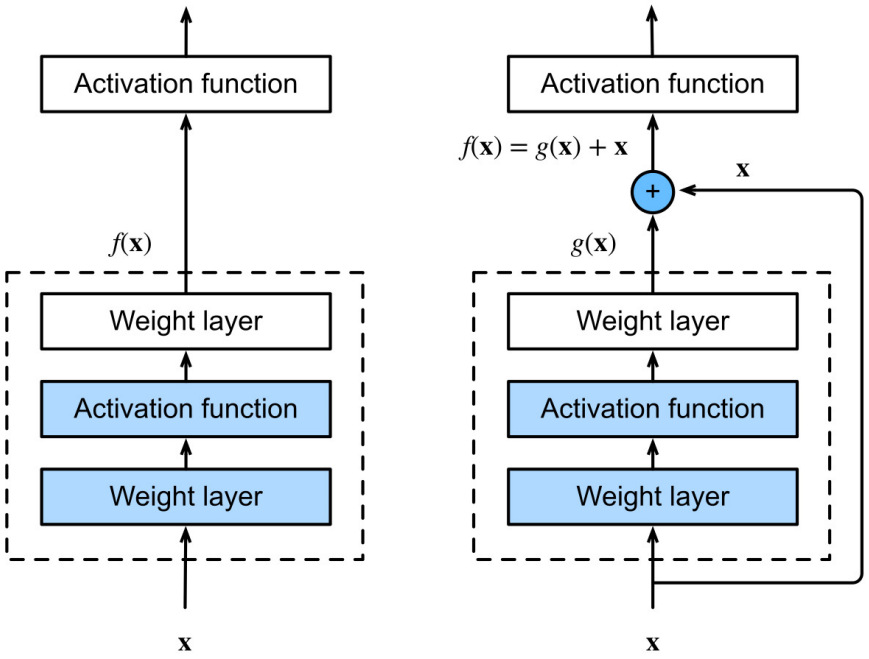
Kiến trúc AlexNet rất sâu so với các mạng trước đó tại thời điểm ra mắt, bao gồm 8 lớp tích chập và 3 lớp kết nối đầy đủ. Trong đó, ở tầng thứ nhất, kích thước cửa sổ tích chập là 11x11x3 (3 là kích thước của ảnh RGB). Kích thước cửa sổ tích chập trong tầng thứ hai được giảm xuống còn 5x5 và sau đó là 3x3. Ngoài ra, theo sau các tầng chập thứ nhất, thứ hai và thứ năm là các tầng Max Pooling với kích thước 3x3 và stride bằng 2.

Sau tầng tích chập cuối cùng là hai tầng kết nối đầy đủ với đầu ra là 4096, hai tầng này tạo ra gần 1GB các tham số mô hình. Cuối cùng là tầng kết nối đầy đủ với đầu ra 1000 dùng cho việc phân loại các đối tượng.

Alexnet cũng đã thay hàm kích hoạt sigmoid bằng ReLU. Điều này vừa giúp giảm việc tính toán vì ReLU không có phép lũy thừa như hàm kích hoạt sigmoid, vừa giúp việc huấn luyện mô hình trở nên dễ dàng hơn vì sử dụng các phương thức khởi tạo tham số khác nhau.

4.3.2. Mạng phần dư ResNet

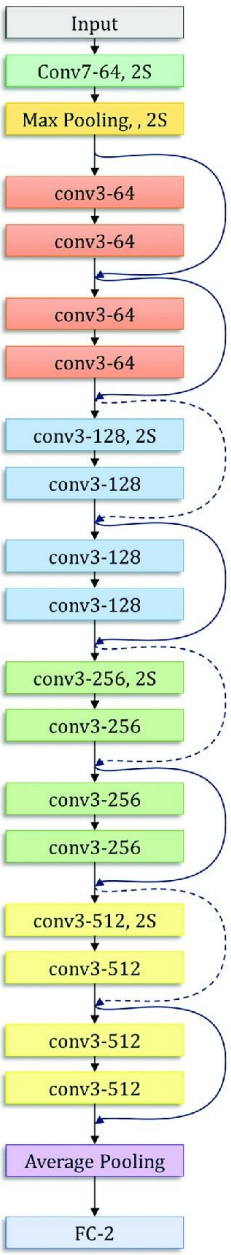
Mạng ResNet (Residual Network) là một kiểu kiến trúc mạng neural sâu (Deep Neural Network) chú trọng vào việc xử lý sự đánh mất gradient (vanishing gradient) trong quá trình đào tạo các mạng sâu. Ý tưởng chính của ResNet là sử dụng các khối còn gọi là "residual blocks" để giảm thiểu sự mất mát thông tin trong quá trình lan truyền ngược (backpropagation) và cho phép xây dựng các mạng sâu hơn mà vẫn có khả năng học tốt.



Hình 16: Sự khác nhau giữa khối thông thường (trái) và khối phần dư (phải)

Với đầu vào là x, giả sử hàm ánh xạ lý tưởng muốn học là f(x) được dùng là đầu vào của hàm kích hoạt. Phần nằm trong viền nét đứt (hình bên trái) phải khớp với hàm ánh xạ f(x). Để thực hiện việc đó có thể không đơn giản nếu không cần khối đó

và muốn giữ lại đầu vào x. Khi đó, chỉ cần tham số hóa độ lệch khỏi giá trị x bằng cách trả về f(x) + x, việc này giúp chống lại đạo hàm bằng 0 do vẫn còn cộng thêm x. Giải pháp này gọi là sử dụng kết nối "tắt" đồng nhất để xuyên qua nhiều lớp, dùng để giải quyết hiện tượng vanishing gradient. Và một khối như vậy được gọi là khối phần dư (Residual Block).



Hình 17: Cấu trúc mạng ResNet-18

Hai tầng đầu tiên trong kiến trúc của ResNet giống với kiến trúc GoogleNet, tầng đầu tiên là tầng tích chập 7x7 với 64 kênh đầu ra và stride bằng 2, theo sau là tầng

Max Pooling 3x3 với stride bằng 2. Theo sau là những tầng tích chập kết hợp với khối phần dư. Số lượng các lớp theo sau tùy thuộc vào kiểu ResNet sử dụng. Một số kiểu ResNet như ResNet-18, ResNet-34, ResNet-50, ResNet-101, ResNet-101, ResNet-152. Các con số theo sau ResNet tượng trưng cho số lượng lớp có trong kiến trúc ResNet đó.

5. Giới thiệu thị giác máy tính

Thị giác máy tính (Computer Vision) là một lĩnh vực trong trí tuệ nhân tạo (AI) liên quan đến việc giải quyết các vấn đề liên quan đến xử lý hình ảnh và video bằng máy tính. Nó có mục tiêu là cho máy tính khả năng nhận biết, hiểu và xử lý thông tin từ hình ảnh và video giống như con người. Một số khía cạnh quan trọng của thị giác máy tính:

Phân đoạn hình ảnh (Image Segmentation): Quá trình chia một hình ảnh thành các phần khác nhau dựa trên các đặc điểm như màu sắc, kích thước, hình dạng, và vị trí. Điều này thường được sử dụng trong việc nhận dạng đối tượng trong hình ảnh.

Phát hiện đối tượng (Object Detection): Nhận biết và xác định vị trí của các đối tượng cụ thể trong hình ảnh hoặc video. Điều này có ứng dụng trong việc theo dõi đối tượng, phân loại đối tượng, và nhiều ứng dụng khác.

Nhận dạng đối tượng (Object Recognition): Xác định xem đối tượng trong hình ảnh hoặc video là gì. Điều này có thể bao gồm việc phân loại đối tượng thành các danh mục cụ thể hoặc nhận dạng đối tượng theo tên riêng.

Phân tích khuôn mặt (Face Analysis): Phân tích các đặc điểm của khuôn mặt như nhận dạng khuôn mặt, nhận dạng biểu cảm, và theo dõi chuyển động khuôn mặt.

Xử lý video (Video Processing): Liên quan đến xử lý và phân tích video, bao gồm việc theo dõi đối tượng trong video, nhận biết hành động, và xử lý video theo thời gian thực.

Tạo hình ảnh tổng hợp (Image Generation): Sử dụng máy tính để tạo ra hình ảnh hoặc video mới dựa trên các mô hình học máy.

Nhận dạng chữ viết (Optical Character Recognition - OCR): Nhận biết và chuyển đổi văn bản từ hình ảnh hoặc tài liệu in thành dạng văn bản có thể xử lý.

Tăng cường thực tế (Augmented Reality - AR): Kết hợp thông tin thị giác với thế giới thực để tạo ra trải nghiệm tăng cường thực tế, ví dụ như trò chơi AR hoặc ứng dụng hướng dẫn sử dụng.

Xử lý hình ảnh y tế (Medical Image Processing): Áp dụng thị giác máy tính trong lĩnh vực y tế để phát hiện bệnh, theo dõi tiến triển của bệnh, và hỗ trợ trong quá trình chẩn đoán.

Tổng hợp dữ liệu từ nhiều nguồn (Multimodal Fusion): Kết hợp thông tin từ nhiều nguồn dữ liệu như hình ảnh, âm thanh, và văn bản để đưa ra quyết định hoặc tạo ra hiểu biết phức tạp hơn về môi trường.

Thị giác máy tính là một lĩnh vực đang phát triển mạnh mẽ và có nhiều ứng dụng thú vị trong nhiều lĩnh vực khác nhau như công nghiệp, y tế, xe tự động, giám sát an ninh, và nhiều ứng dụng khác. Để đạt được các mục tiêu trong thị giác máy tính, các nhà nghiên cứu thường sử dụng các mô hình học máy và mạng neural sâu (deep learning) để giải quyết các vấn đề phức tạp liên quan đến xử lý hình ảnh và video.

6. Camera Calibration

Camera calibration (hiệu chỉnh máy ảnh) là một quá trình quan trọng trong lĩnh vực thị giác máy tính, được thực hiện để biến đổi hình ảnh từ camera thành hình ảnh thế giới thực, giúp máy tính hiểu và xử lý thông tin trong không gian ba chiều (3D). Quá trình này đặc biệt quan trọng khi bạn muốn đo đạc hoặc theo dõi các đối tượng trong thế giới thực dựa trên hình ảnh chụp từ camera. Một số khía cạnh quan trọng trong quá trình calibrate camera:

Nhiệm vụ chính: Nhiệm vụ chính của camera calibration là tìm ra các thông số và biểu đồ biến đổi giữa không gian 3D (thế giới thực) và không gian hình ảnh (hình ảnh trên camera). Cụ thể, điều này bao gồm xác định các thông số như tỷ lệ tiêu cự, distortion coefficients (hệ số méo), và vị trí của trung tâm của ống kính.

Biến đổi 2D sang 3D: Sau khi calibrate camera, có thể biến đổi các điểm trong hình ảnh 2D thành không gian 3D dựa trên các thông số đã xác định. Điều này cho phép đo đạc các kích thước và khoảng cách trong không gian thực.

Loại bỏ méo ảnh (Distortion Correction): Hệ số méo (distortion) là sự méo mó không mong muốn trong hình ảnh chụp từ camera và có thể làm sai lệch các thông tin về khoảng cách và hình dạng. Calibration giúp loại bỏ hoặc giảm thiểu méo ảnh này, đảm bảo hình ảnh đúng với thực tế.

Ứng dụng trong thị giác máy tính: Camera calibration được sử dụng rộng rãi trong thị giác máy tính để các ứng dụng như theo dõi chuyển động, phát hiện đối tượng, tái tạo mô hình 3D và thực hiện đo lường trong không gian thực.

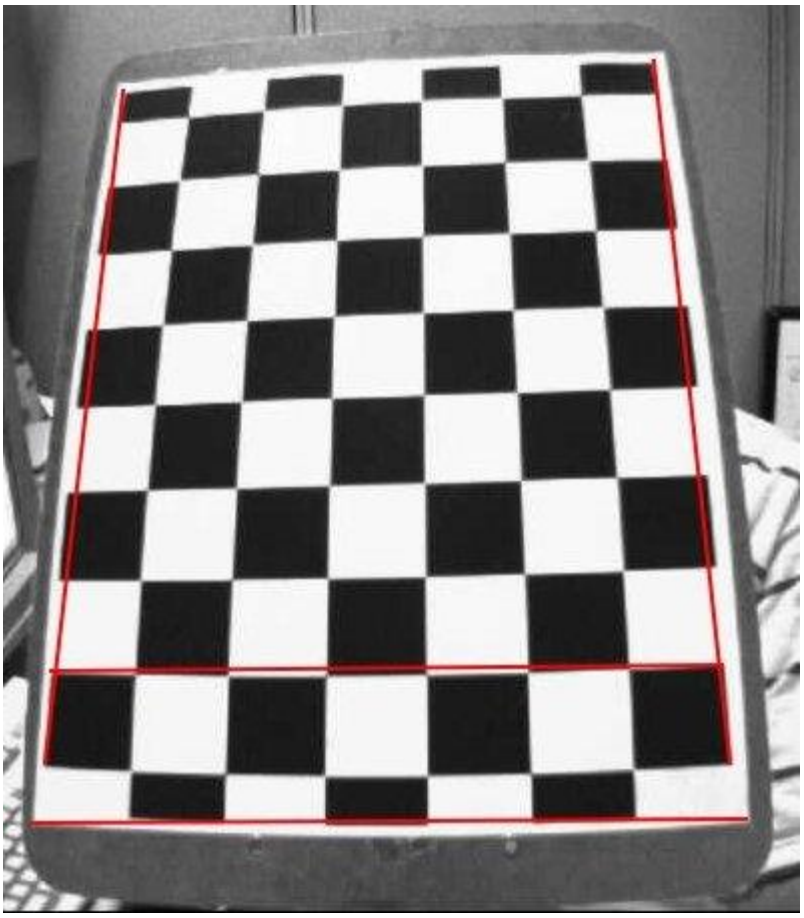
Phương pháp calibration: Có nhiều phương pháp calibrate camera, bao gồm việc sử dụng mẫu đặc biệt (chẳng hạn như một dấu chấm hoặc một bảng chứa các hình dạng đã biết trước) và thu thập các hình ảnh từ nhiều góc độ khác nhau.

Công cụ phần mềm: Có nhiều công cụ phần mềm và thư viện như OpenCV, MATLAB, và Python với các thư viện như Camera Calibration Toolbox cho việc thực hiện camera calibration.

Quá trình calibration là bước quan trọng để đảm bảo rằng thông tin từ camera có thể được sử dụng chính xác trong các ứng dụng thị giác máy tính, đặc biệt là khi cần đo lường và phân tích không gian ba chiều.

Một số máy ảnh lỗ kim gây ra hiện tượng méo hình ảnh đáng kể và hai loại biến dạng phổ biến nhất là biến dạng radial và biến dạng tangential.

Biến dạng radial làm cho các đường thẳng trở nên cong. Biến dạng radial trở nên lớn hơn khi các điểm xa hơn khỏi trung tâm của hình ảnh. Ví dụ, bên dưới đây là một hình ảnh trong đó hai cạnh của bàn cờ đang được đánh dấu bằng các đường đỏ. Nhưng có thể thấy rằng biên của bàn cờ không phải là một đường thẳng và không khớp với đường đỏ. Tất cả các đường thẳng đều bị nở ra.



Hình 18: Hình ảnh bị biến dạng

Biến dạng radial có thể được biểu diễn như sau:

$$x_{distorted} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$
$$y_{distorted} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

Tương tự vậy, hiện tượng biến dạng tangential xảy ra do thấu kính chụp ảnh không được đặt song song hoàn hảo với mặt phẳng hình ảnh. Dẫn đến việc một số vùng trong ảnh có thể trong gần hơn. Mức độ biến dạng tangential có thể được biểu diễn như sau:

$$x_{distorted} = x + [2p_1 xy + p_2(r^2 + 2x^2)]$$
$$y_{distorted} = y + [p_1(r^2 + 2y^2) + 2p_2 xy]$$

Tóm lại, để có thể hiệu chỉnh máy ảnh, cần tìm năm tham số gọi là hệ số biến dạng được cho bởi:

$$Distortion\ coefficients = (k_1\ \ k_2\ \ p_1\ \ p_2\ \ k_3)$$

Ngoài ra, cần một số thông tin khác như các thông số biên trong và ngoài máy ảnh. Các thông số đặc biệt dành riêng cho máy ảnh bao gồm như tỷ lệ tiêu cự (f_x, f_y) và vị

trí tâm (c_x, c_y). Các thông số này sẽ được biểu diễn dưới dạng ma trận gọi là ma trận camera như sau:

$$camera\ matrix = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

Hình 19: Hình ảnh sau khi hiệu chỉnh

Dựa vào các tham số camera matrix và distortion coefficients kết hợp với những thư viện hỗ trợ như OpenCV trong python có thể hiệu chỉnh hình ảnh, loại bỏ các biến dạng. Đây là tiền đề để tăng sự chính xác trong các vấn đề liên quan đến thị giác máy tính.

7. Phát hiện khuôn mặt và các điểm mốc khuôn mặt (face landmark)

Phát hiện khuôn mặt và các điểm mốc khuôn mặt (face landmark) là hai nhiệm vụ quan trọng trong lĩnh vực thị giác máy tính và xử lý ảnh, đặc biệt trong lĩnh vực trí tuệ nhân tạo và ứng dụng của nó, như nhận dạng khuôn mặt, dự đoán cảm xúc, tự động nhận diện và điều khiển gương mặt trên các thiết bị di động.

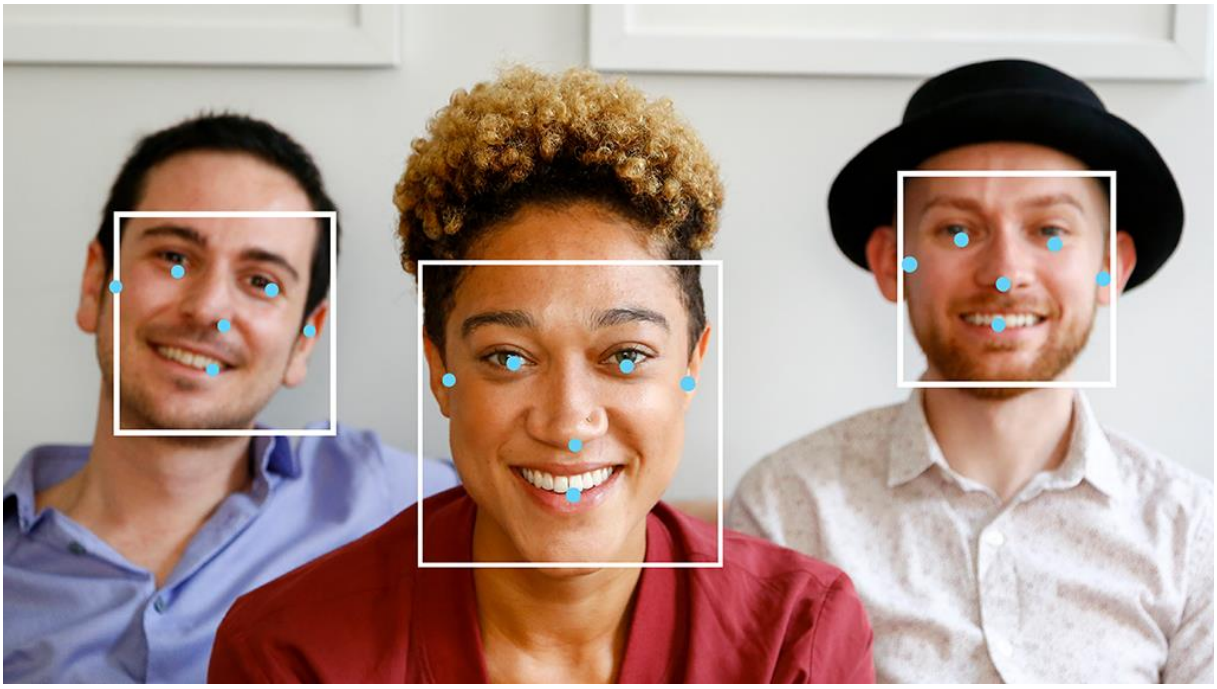
Hiện nay có rất nhiều nền tảng được xây dựng và phát triển cho các mục đích liên quan đến thị giác máy tính nói chung và phát hiện khuôn mặt, điểm mốc khuôn mặt nói riêng. Một trong các nền tảng được sử dụng phổ biến nhất là MediaPipe.

MediaPipe là một nền tảng mã nguồn mở được phát triển bởi Google, là tập hợp các giải pháp Machine Learning và trí tuệ nhân tạo như phát hiện vật thể, phân loại hình

ảnh, phát hiện khuôn mặt, phát hiện các điểm mốc trên khuôn mặt, phân loại văn bản,… Mỗi giải pháp bao gồm một hoặc nhiều mô hình được huấn luyện trước với độ chính xác cao.

7.1. MediaPipe – Phát hiện khuôn mặt

Phát hiện khuôn mặt là một bài toán cực kỳ quen thuộc và lâu đời tính đến thời điểm hiện tại. Nhiệm vụ của bài toán này là làm sao từ một ảnh, video hoặc webcam trực tiếp làm đầu vào, tìm ra được vị trí và bounding box những khuôn mặt con người xuất hiện trong đầu vào đấy, sau đấy đánh dấu các điểm quan trọng trên khuôn mặt như mũi, miệng, mắt,…(MediaPipe sử dụng 5-landmark). MediaPipe Face Detection sử dụng mạng BlazeFace làm nền tảng nhưng thay đổi backbones. Ngoài ra, thuật toán NMS (non-maximum suppression) cũng được thay thế bởi một chiến thuật khác, giúp giảm đáng kể thời gian xử lý.

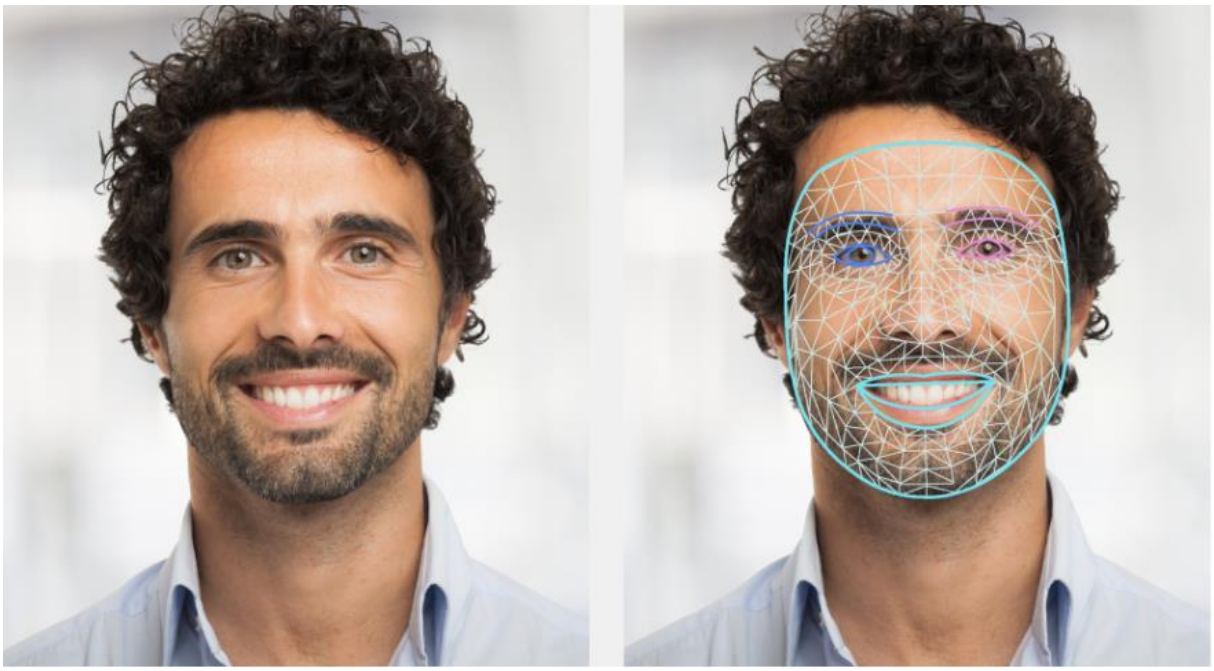


Hình 20: Ví dụ về phát hiện khuôn mặt

7.2. MediaPipe – Phát hiện các điểm mốc trên khuôn mặt

Khác với phát hiện khuôn mặt, thay vì tìm vị trí và bounding box bao quanh mặt, MediaPipe Face Landmark Detection sẽ nhận diện một loạt các điểm trên khuôn mặt để tạo thành một lưới (mesh) của mặt. Lưới này thường được áp dụng vào các bài toán chỉnh sửa mặt 3D hay các tác vụ liên quan đến 3D Alignment và Anti-spoofing.

Giải pháp của MediaPipe sinh ra 468 điểm trên mặt và tạo thành lưới mà không đòi hỏi quá nhiều năng lực tính toán cũng như số lượng camera.



Hình 21: Ví dụ về phát hiện các điểm mốc trên khuôn mặt

II. Khái niệm lập trình nhúng, lập trình Arduino, thiết kế mạch

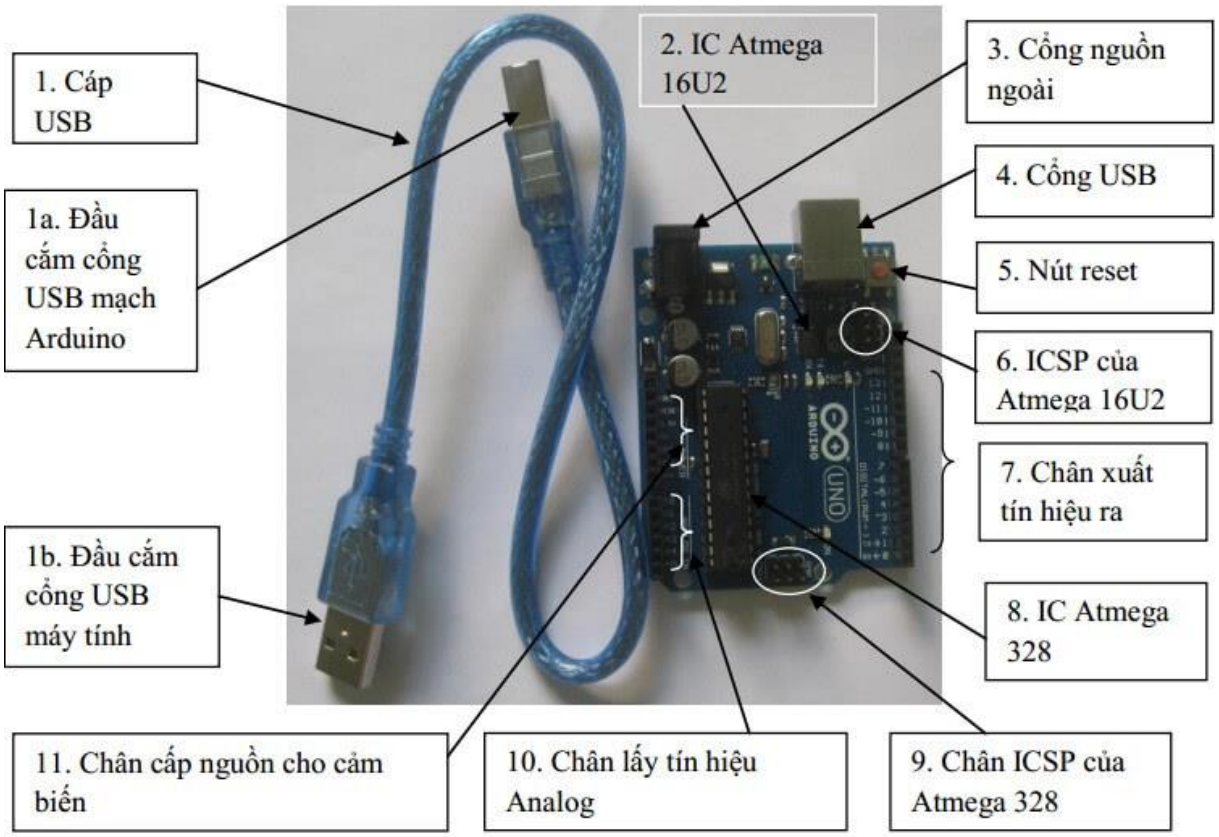
1. Giới thiệu

Arduino là một nền tảng phát triển điện tử mã nguồn mở phổ biến, được sáng tạo để đơn giản hóa quá trình phát triển các dự án và ứng dụng điện tử. Mạch Arduino được sử dụng để cảm nhận và điều khiển nhiều đối tượng khác nhau. Nó có thể thực hiện nhiều nhiệm vụ lấy tín hiệu từ cảm biến đến điều khiển đèn, động cơ, và nhiều đối tượng khác. Ngoài ra mạch còn có khả năng liên kết với nhiều module khác nhau như module đọc thẻ từ, ethernet shield, sim900A, ….để tăng khả ứng dụng của mạch.

Phần cứng bao gồm một board mạch nguồn mở được thiết kế trên nền tảng vi xử lý AVR Atmel 8bit, hoặc ARM, Atmel 32-bit,…. Hiện phần cứng của Arduino có tất cả 6 phiên bản, Tuy nhiên phiên bản thường được sử dụng nhiều nhất là Arduino Uno và Arduino Mega.

2. Cấu tạo của Arduino

Hiện Arduino có rất nhiều loại, ví dụ với Arduino Uno R3 sẽ bao gồm cấu tạo và các thông số cơ bản sau:



Hình 22: Cấu tạo của Arduino Uno R3

Vi điều khiển	Atmega 328 (họ 8 bit)
Điện áp hoạt động	5V – DC (cấp qua cổng USB)
Tần số hoạt động	16 MHz
Dòng tiêu thụ	30mA
Điện áp vào khuyên dùng	7 – 12V – DC
Điện áp vào giới hạn	6 – 20V – DC
Số chân Digital I/O	14 (6 chân PWM)
Số chân Analog	6 (độ phân giải 10 bit)
Dòng tối đa trên mỗi chân I/O	30 mA
Dòng ra tối đa (5V)	500 mA
Dòng ra tối đa (3.3V)	50 mA
Bộ nhớ flash	32 KB (Atmega328) với 0.5KB dùng bởi bootloader
SRAM	2KB (Atmega328)
EEPROM	1KB (Atmega328)

3. Ứng dụng Arduino

Arduino có nhiều ứng dụng trong đời sống, trong việc chế tạo các thiết bị điện tử chất lượng cao. Một số ứng dụng có thể kể đến như:

Lập trình robot: Arduino chính là một phần quan trọng trong trung tâm xử lý giúp điều khiển được hoạt động của robot.

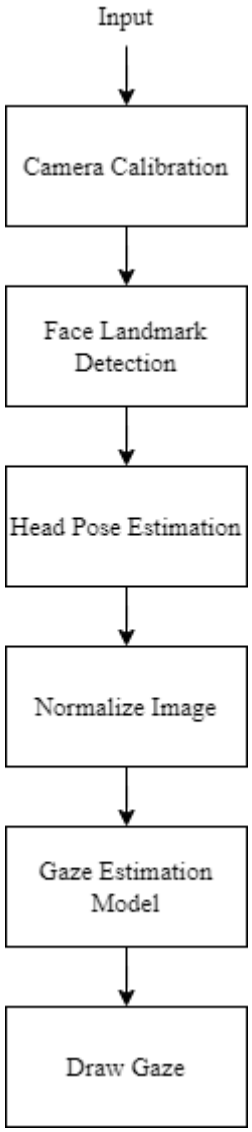
Lập trình máy bay không người lái. Có thể nói đây là ứng dụng có nhiều kỳ vọng trong tương lai.

Game tương tác: chúng ta có thể dùng Arduino để tương tác với Joystick, màn hình,… để chơi các trò như Tetrix, phá gạch, Mario…

Arduino điều khiển thiết bị ánh sáng cảm biến tốt. Là một trong những bộ phần quan trọng trong cây đèn giao thông, các hiệu ứng đèn nháy được cài đặt làm nổi bật các biển quảng cáo.

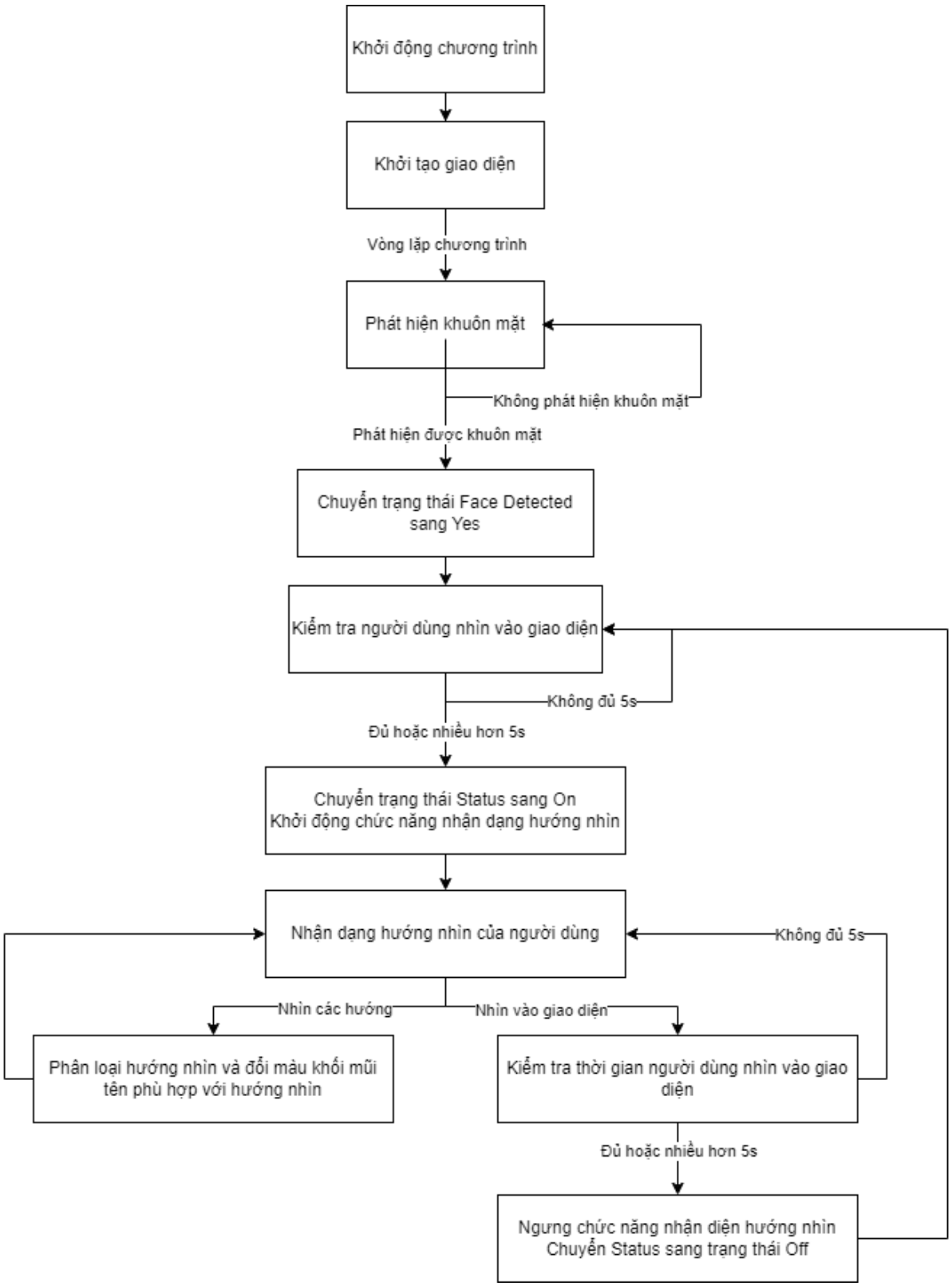
Arduino cũng được ứng dụng trong máy in 3D và nhiều ứng dụng khác tùy thuộc vào khả năng sáng tạo của người sử dụng.

III. Mô hình nhận diện hướng mắt, phần mềm nhận diện hướng mắt từ mô hình đã tạo

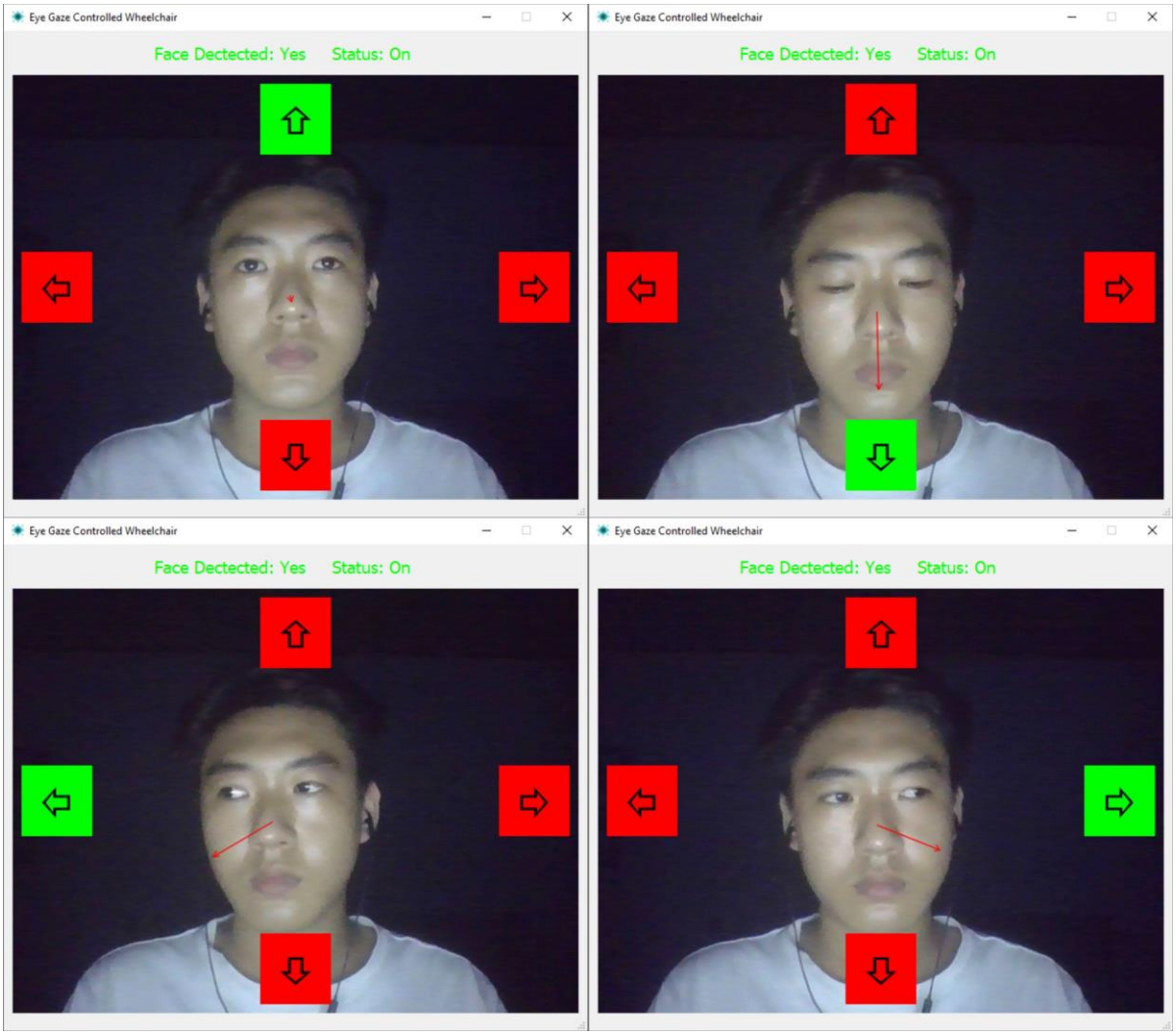


Hình 23: Sơ đồ hoạt động của module ước tính hướng mắt

Hình 24: Mô hình nhận diện hướng mắt

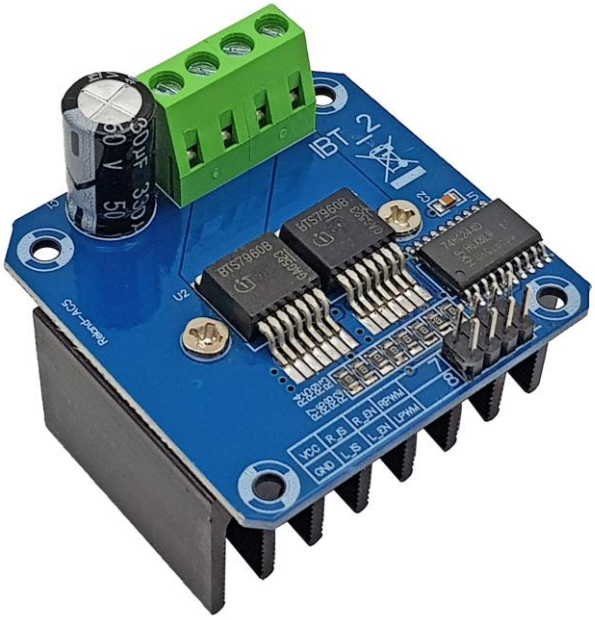


Hình 25: Sơ đồ hoạt động của phần mềm nhận diện hướng mắt



Hình 26: Phần mềm nhận diện hướng mắt hoàn chỉnh

IV. Mạch điều khiển xe lăn



Hình 27: Mạch điều khiển động cơ BTS7960 43A

V. Xe lăn điều khiển bằng mắt

Hình 28: Xe lăn điều khiển bằng mắt

Thực nghiệm tại trường Công nghệ thông tin & Truyền thông – Đại học Cần Thơ