BÁO CÁO PROJECT II:

NHẬN DIỆN KHUÔN MẶT NGƯỜI BẰNG THUẬT TOÁN CNN

# A, Cơ sở lý thuyết và công nghệ sử dụng

# I, Giới thiệu về thuật toán CNN - Convolutional Neural Network

Convolutional Neural Network (CNN – Mạng neural tích chập) là một trong những mô hình Deep Learning tiên tiến giúp xây dựng được những hệ thống thông minh với độ chính xác cao như hiện nay như hệ thống xử lý ảnh lớn như Facebook, Google hay Amazon đã đưa vào sản phẩm của mình những chức năng thông minh như nhận diện khuôn mặt người dùng, phát triển xe hơi tự lái hay drone giao hàng tự động.

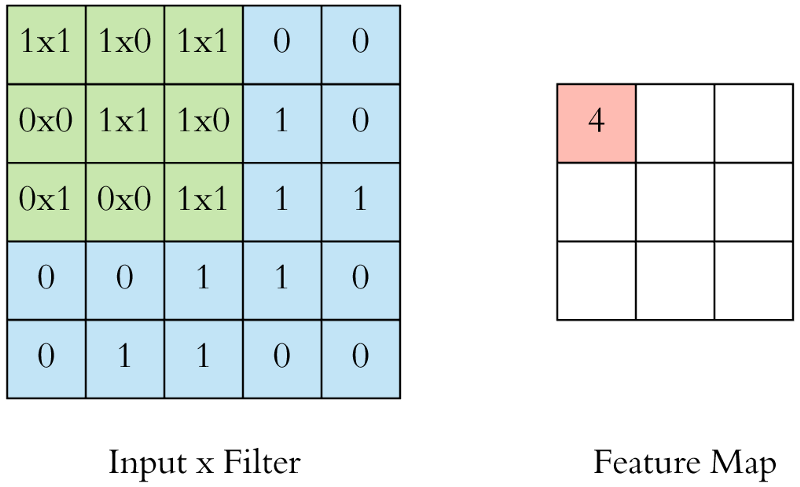
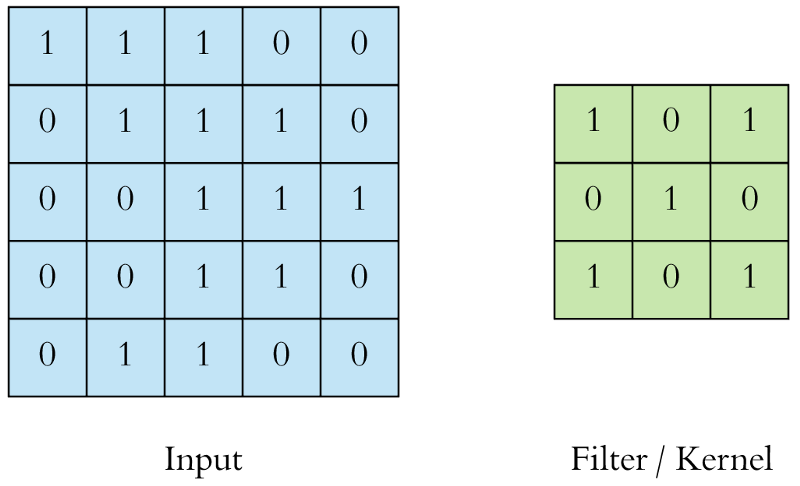
Trong mạng neural, mô hình mạng neural tích chập (CNN) là 1 trong những mô hình để nhận dạng và phân loại hình ảnh. Trong đó, xác định đối tượng và nhận dạng khuôn mặt là 1 trong số những lĩnh vực mà CNN được sử dụng rộng rãi.

# II, Các khái niệm liên quan đến thuật toán CNN

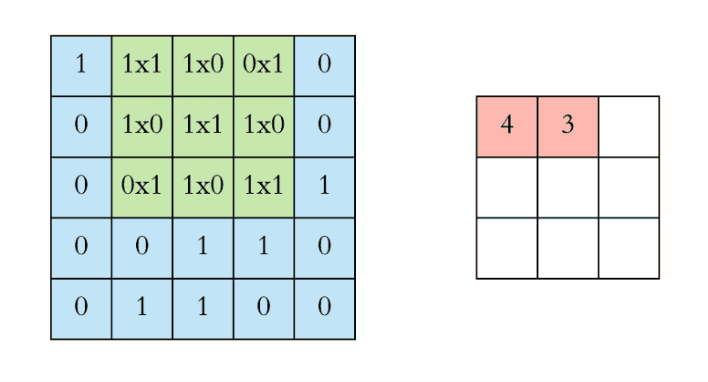
## 1, Convolution

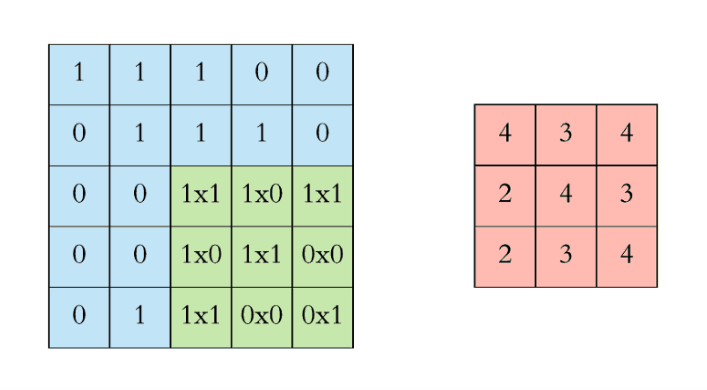
Để đơn giản hóa việc định nghĩa và giải thích, ta coi Convolution bao gồm 2 khái niệm khác là Convolution Filter và Convolutional Layer.

* Trong mạng neural network thông thường, từ input, ta cho qua các hidden layer rồi ra được output. Đối với CNN, Convolutional Layer cũng chính là hidden layer, khác ở chỗ, Convolutional Layer là một tập các feature map và mỗi feature map này là một bản scan của input ban đầu, nhưng được trích xuất ra các feature (đặc tính) cụ thể.
* Việc scan như thế nào dựa vào Convolution Filter hay Kernel. Đây là một ma trận sẽ quét qua ma trận dữ liệu đầu vào, từ trái qua phải, trên xuống dưới, và nhân tương ứng từng giá trị của ma trận đầu vào và ma trận Kernel rồi cộng tổng lại, đưa qua các activation function (sigmoid, relu, elu, ...), kết quả sẽ là một con số cụ thể, tập hợp các con số này lại tạo ra 1 ma trận khác, chính là feature map.



Cơ chế scan của ma trận Kernel trên ma trận Input tương tự như kĩ thuật Sliding Window (Cửa sổ trượt):

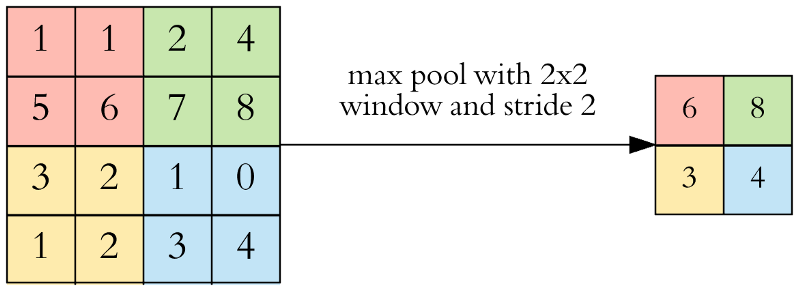




## 2, Pooling

Mục đích sử dụng Pooling rất đơn giản, đó là làm giảm số hyperparameter mà ta cần phải tính toán, từ đó giảm thời gian tính toán và tránh overfitting. Loại Pooling thường gặp nhất là Max Pooling, lấy giá trị lớn nhất trong một Pooling Window.

Pooling hoạt động gần giống với Convolution, sử dụng 1 cửa sổ trượt gọi là Pooling Window, cửa sổ này trượt qua từng giá trị của ma trận dữ liệu đầu vào (thường là các feature map trong Convolutional Layer), chọn ra một giá trị từ các giá trị nằm trong cửa sổ trượt (với Max Pooling sẽ lấy giá trị lớn nhất).



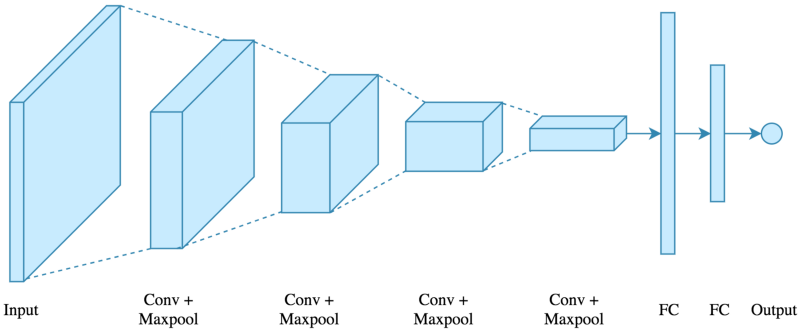
Cửa sổ trượt Max Pooling ở trên có kích thước 2x2 với bước trượt là 2.

Như vậy qua lớp Max Pooling thì số lượng neural giảm đi phân nửa. Trong một mạng CNN có nhiều Feature Map nên mỗi Feature Map chúng ta sẽ cho mỗi Max Pooling khác nhau. Chúng ta có thể thấy rằng Max Pooling là cách hỏi xem trong các đặc trưng này thì đặc trưng nào là đặc trưng nhất.

## 3, Fully Connected

Tương tự như 1 neural network thông thường, layer này cũng chính là 1 fully connected ANN. Thường thì sau các lớp Conv + Pooling thì sẽ là 2 lớp Fully connected: 1 layer để tập hợp các feature layer mà ta đã tìm ra, chuyển đổi dữ liệu từ 3D, hoặc 2D thành 1D, tức chỉ còn là 1 vector; 1 layer nữa là output, số neural của layer này phụ thuộc vào số output mà ta muốn tìm ra.

## 4, CNN Structure



Hình ảnh trên minh họa kiến trúc của 1 mạng CNN.

Ta có thể thấy mạng CNN là một tập hợp các Convolution layer chồng lên nhau, 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. Mỗi một lớp được sử dụng các filter khác nhau, thông thường có hàng trăm hàng nghìn filter như vậy và kết hợp kết quả của chúng lại.

Ngoài ra, còn có một số layer khác như pooling dùng để chắt lọc lại các thông tin hữu ích hơn (hay loại bỏ các thông tin nhiễu). Trong quá trình huấn luyện mạng (training) CNN tự động học các giá trị qua các lớp filter dựa vào cách thức mà ta thực hiện.

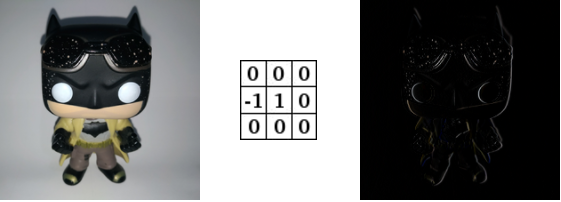
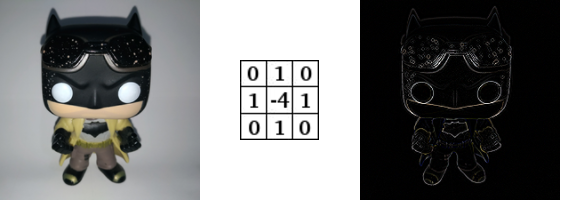
# III, Áp dụng thuật toán CNN trong xử lý ảnh

## 1, Phép tích chập trong xử lý ảnh

Trong toán học, tích chập là phép toán tuyến tính cho ra kết quả là một hàm mới bằng việc tính toán dựa trên hai hàm đã cho.

Theo lĩnh vực xử lý ảnh, ảnh kỹ thuật số (ảnh được số hóa) có thể coi là một ma trận 2 chiều chứa hữu hạn các phần tử gọi là điểm ảnh (pixel). Mỗi pixel mang tọa độ (x, y) và giá trị số l(x, y) là độ xám của điểm ảnh đó.

Đối với bài toán nhận diện vùng ảnh, phép tích chập giữa ma trận lọc (kernel) và ma trận ảnh input cho ra kết quả ảnh đã được trích xuất ra các đặc tính (features) cụ thể:



Ảnh minh họa phép tích chập của ma trận input và ma trận kernel

Công thức tích chập giữa hàm ảnh f(x, y) và bộ lọc k(x, y) (kích thước mxn):



Thành phần không thể thiếu của phép tích chập là ma trận kernel (bộ lọc). Điểm neo (anchor point) của kernel sẽ quyết định vùng ma trận tương ứng trên ảnh để tích chập, thông thường anchor point được chọn là tâm của kernel. Giá trị mỗi phần tử trên kernel được xem như là hệ số tổ hợp với lần lượt từng giá trị độ xám của điểm ảnh trong vùng tương ứng với kernel.

Từ đó, ta có thể hình dung phép tích chập được thực hiện bằng việc dịch chuyển ma trận kernel lần lượt qua tất cả các điểm ảnh trong ảnh, bắt đầu từ góc bên trái trên của ảnh. Và đặt anchor point tương ứng tại điểm ảnh đang xét. Ở mỗi lần dịch chuyển, thực hiện tính toán kết quả mới cho điểm ảnh đang xét bằng công thức tích chập.

Mạng CNN sẽ được train để tự động tìm các kernel phù hợp cho việc dò ra các features của ảnh.

## 2, Chọn các tham số phù hợp cho mạng CNN

* Convolution layers: càng nhiều các convolution layers thì performance càng được cải thiện. Ta chọn sau khoảng 3 hoặc 4 layers, các tác động được giảm một cách đáng kể.
* Kernel size: Thường chọn size 3x3 hoặc 5x5.
* Pooling size: Thường chọn size 2x2 hoặc 4x4 cho ảnh đầu vào có kích thước lớn.

# IV, Một số công nghệ hỗ trợ

## 1, Ngôn ngữ lập trình: Python 3

## 2, OpenCV2

Là một thư viện mã nguồn mở hàng đầu cho thị giác máy tính (computer vision), xử lý ảnh và máy học, và các tính năng tăng tốc GPU trong hoạt động thời gian thực.

## 3, TensorFlow

Là một thư viện mã nguồn mở dùng cho tính toán số học sử dụng đồ thị luồng dữ liệu phát triển bởi Google – Google Brain. TensorFlow được sử dụng phổ biến do:

* Tích hợp sẵn rất nhiều các thư viện Machine Learning.
* Có khả năng tương thích và mở rộng tốt. Được Google phát triển cho Machine Learning phục vụ cả nghiên cứu lẫn xây dựng các ứng dụng thực tế.

# B, Áp dụng vào tiến hành xử lý bài toán cụ thể

# I, Giới thiệu bài toán nhận diện khuôn mặt người

Ngày nay, nhận dạng khuôn mặt người (hay Facial detection) đã không còn là một khái niệm quá xa lạ đối với bất kỳ một ai. Đây là một kỹ thuật nhằm xác định khuôn mặt một người từ nhiều góc độ trong một hình ảnh hoặc một khung hình trong video thu thập được. Công nghệ nhận diện khuôn mặt giờ đã trở nên rất quen thuộc, và được áp dụng phổ biến trong các hệ thống an ninh ở nhiều nơi trên thế giới, trong đó có cả Việt Nam. Ưu điểm của công nghệ này so với các công nghệ nhận dạng khác (nhận dạng vân tay, nhận dạng giọng nói, nhận dạng mống mắt) chính là việc nó không đòi hỏi sự hợp tác đến từ người dùng.

Hiện nay, có rất nhiều kỹ thuật để thực hiện việc nhận dạng khuôn mặt người, tuy nhiên cơ sở chung của các kỹ thuật này là đều sẽ phải tiến hành qua 3 bước:

* Xác định và lấy ra (các) khuôn mặt người có trong hình ảnh.
* Từ hình ảnh các khuôn mặt người được lấy ra, thực hiện việc phân tích, trích xuất các đặt trưng (features) của từng khuôn mặt.
* Từ các thông tin có được sau khi phân tích, kết luận và xác minh danh tính người dùng tương ứng.

Bài viết này nhằm xây dựng một hệ thống tự động nhận diện khuôn mặt người hoàn chỉnh dựa trên ứng dụng của thuật toán CNN.

# II, Mô tả hướng triển khai và dữ liệu đầu vào, đầu ra của sản phẩm

## 1, Hướng triển khai thuật toán để xây dựng sản phẩm

Dựa theo cơ sở chung của các kỹ thuật xử lý bài toán nhận dạng khuôn mặt người đã nêu, quá trình triển khai thuật toán CNN để xây dựng chương trình sản phẩm chia thành các bước:

* **Bước 1:** Chuẩn bị dataset gồm tập nhiều hình ảnh có chứa/không chứa dữ liệu khuôn mặt người được đánh nhãn tương ứng (positive/negative), từ đó chia thành 2 tập training mô hình CNN và testing kết quả training.
* **Bước 2:** Thiết lập mạng CNN dựa trên cơ sở lý thuyết đã trình bày và thực hiện train model bằng dữ liệu từ tập training, ta sẽ áp dụng thuật toán tối ưu Adam ([Adam optimizer](https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/)) để tối ưu và đánh giá độ chính xác (accuracy) của mạng neural này.
* **Bước 3:** Thực hiện test model vừa training bằng dữ liệu trong tập testing, đưa ra output của sản phẩm để nhận xét độ chính xác của model.
* **Bước 4:** Áp dụng model vào nhận diện các vùng ảnh chứa mặt người của 1 bức ảnh bất kì để đánh giá độ hiệu quả của model.

## 2, Dữ liệu đầu vào của sản phẩm

Là một hoặc nhiều hình ảnh trong 1 tệp có chứa dữ liệu về khuôn mặt người.



Ảnh minh họa cho dữ liệu đầu vào

## 3, Dữ liệu đầu ra của sản phẩm

Sản phẩm đầu ra là một chương trình tự động nhận diện và đóng khung tag chữ nhật các vùng hình ảnh chứa thông tin về khuôn mặt người từ các hình ảnh đầu vào.



Ảnh minh họa cho dữ liệu đầu ra

# III, Cài đặt môi trường

* Tải và cài đặt môi trường Python 3.6: <https://www.python.org/downloads/>
* Cài đặt PIP3 trên Windows 7, 8, 10: Tải xuống tệp lệnh [get-pip.py](https://quantrimang.com/url?q=aHR0cHM6Ly9ib290c3RyYXAucHlwYS5pby9nZXQtcGlwLnB5), trên cửa sổ **Command Prompt** điều hướng đến tệp chạy dòng lệnh:

*python get-pip.py*

* Cài đặt các thư viện hỗ trợ NumPy, Pandas, Matplotlib, Scikit-Learn:

*pip3 install –upgrade matplotlib numpy pandas scipy scikit-learn*

* Tải và cài đặt OpenCV2 trên Windows:

*pip install opencv-python*

* Tải và cài đặt TensorFlow 1.5:

*pip install tensorflow-gpu==1.5*

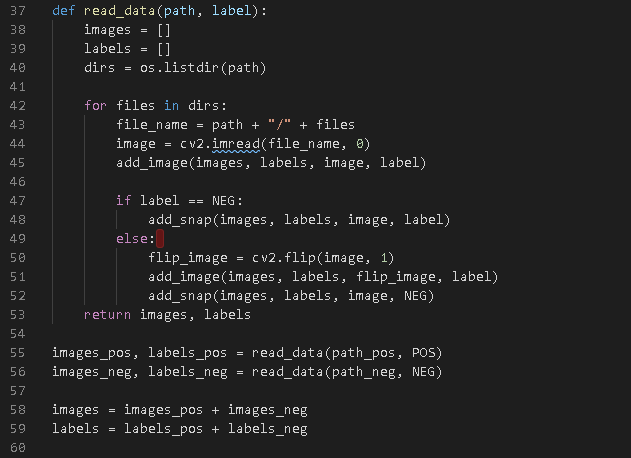
# IV, Chuẩn bị dataset và xây dựng các tập training/testing data

Dữ liệu dataset có chứa/không chứa thông tin về khuôn mặt người và mô tả cụ thể được lấy tham khảo từ các nguồn:

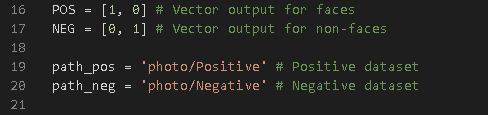
<https://www.kaggle.com/prasunroy/natural-images>

<https://figshare.com/articles/dataset/15-Scene_Image_Dataset/7007177>

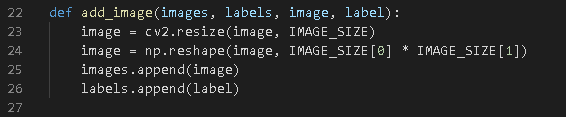
Đầu tiên, ta cần load dữ liệu từ các folder chứa ảnh và đánh nhãn (label) tương ứng với mỗi bức ảnh.



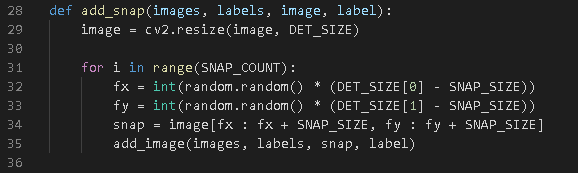
Trong đó, ảnh chứa mặt người có label là POS (positive), ảnh không chứa mặt người có label là NEG (negative).



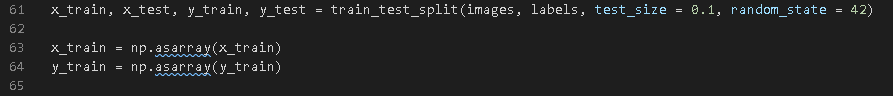
Các bức ảnh sau khi load cần được resize về cùng 1 kích thước chuẩn trước khi đưa vào tập dữ liệu ảnh để chia thành 2 tập training và testing data ở bước tiếp theo.



Ngoài ra, để đa dạng hóa dữ liệu negative, với mỗi bức ảnh ta tạo thêm tập các ảnh snapshot trích xuất từ ảnh đó rồi thêm các ảnh này vào tập dữ liệu ảnh với label là NEG.



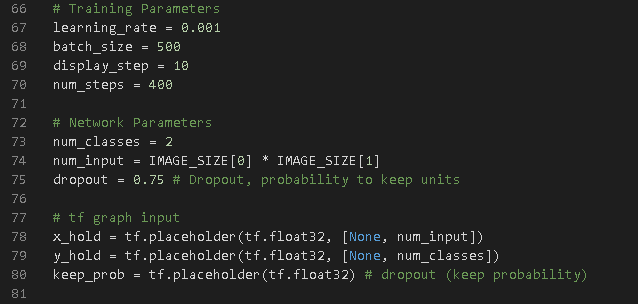
Sau khi tập dữ liệu ảnh đã hoàn thiện, ta chia nó thành 2 tập dữ liệu training và testing data.



# V, Thiết lập mạng CNN và thực hiện train model

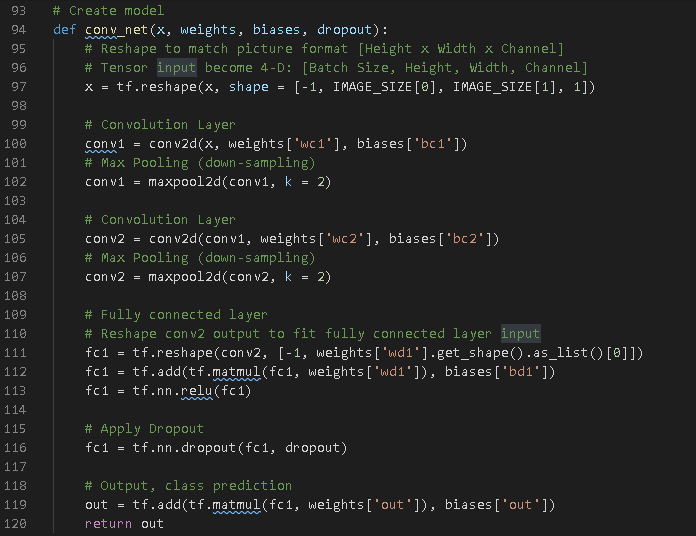
## 1, Thiết lập mạng CNN

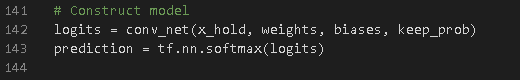
Trước hết, ta sẽ thiết lập các giá trị parameters và input holder cho CNN như sau:



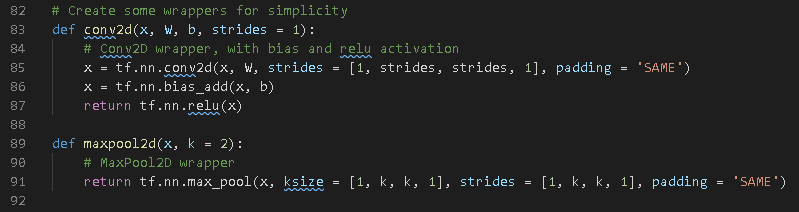
Thông thường, một CNN bao gồm một stack chứa các convolutional module, mỗi module thực hiện công việc extract các features từ dữ liệu đầu vào tương ứng. Mỗi module cũng bao gồm một convolutional layer và theo sau là một pooling layer.

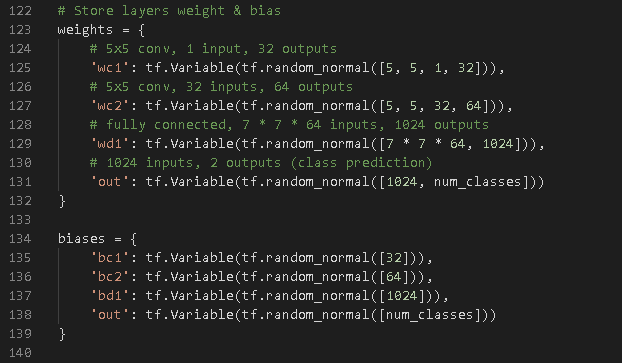
Convolutional module cuối cùng gồm một hoặc nhiều các dense layer (hay còn gọi là fully-connected layer) để thực hiện việc phân lớp dữ liệu. Dense layer cuối cùng trong một CNN gồm một single node cho mỗi một lớp cụ thể trong model (tất cả các class khả thi mà model có thể dự đoán được). Với mỗi node này, ta sẽ sử dụng hàm kích hoạt softmax để tạo ra giá trị predict.





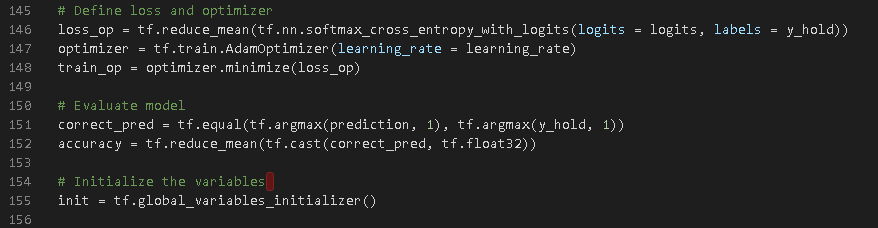
Trong đó, các hàm xây dựng layer và các hằng số ứng với mỗi layer (weight, bias) được thiết lập như sau:



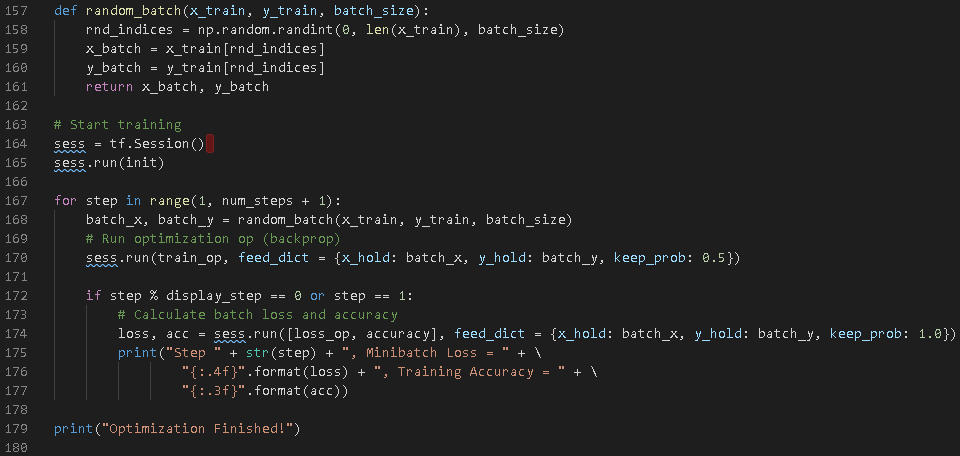


## 2, Thực hiện training mạng CNN với thuật toán Adam optimizer

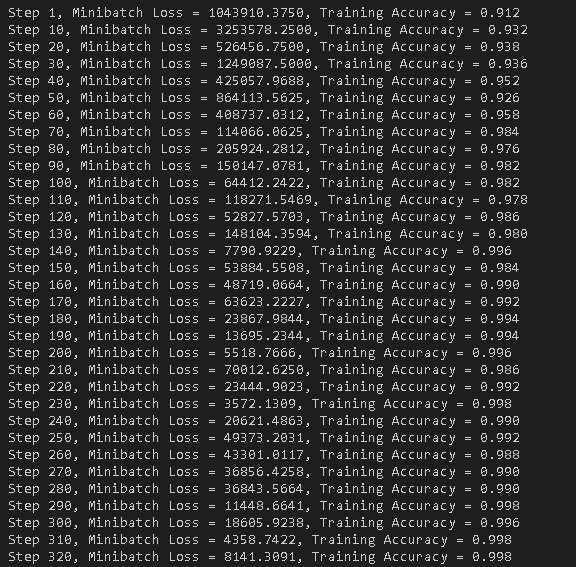
Đầu tiên, ta cần tìm correct và accuracy để đánh giá độ chính xác của mạng CNN vừa được thiết lập. Ta sẽ thiết lập hàm mất mát loss\_op của mạng, sau đó đưa hàm này vào giải thuật Adam optimizer. Cũng giống như Gradient Descent, Adam optimizer tìm ra bộ tham số để minimize giá trị hàm mất mát loss\_op. Phần code phía dưới tìm correct và accuracy để đánh giá độ chính xác của mạng CNN và cuối cùng là dòng code khởi tạo các variable trong tensorflow.

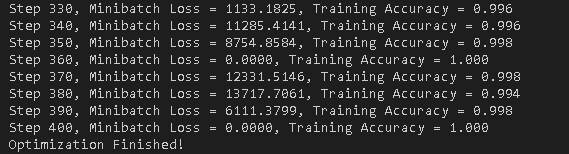


Tiếp theo, ta truyền các giá trị x, y đã chuẩn bị trước để bắt đầu train mạng CNN:



Kết quả in ra sau khi thực hiện optimize:



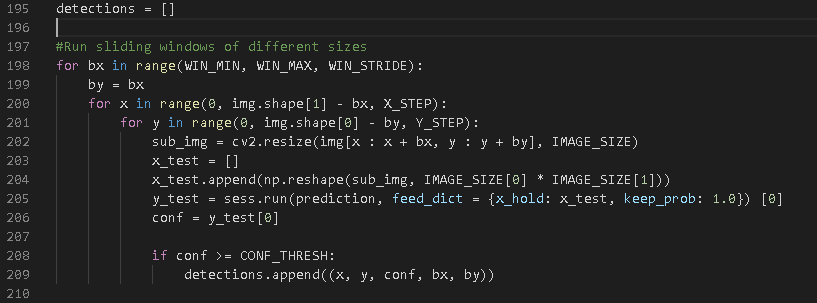


Nhận xét thấy kết quả training mạng CNN khá khả quan khi accuracy xấp xỉ 1, ta sẽ test model để kiểm tra tính chính xác của mạng.

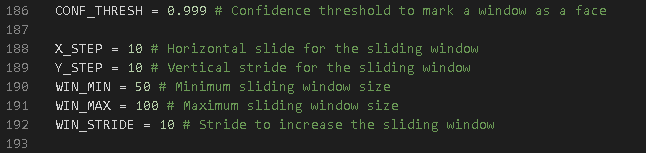
# VI, Test model vừa train và đưa ra output

## 1, Kĩ thuật Sliding window

Để tìm ra các vùng ảnh có chứa mặt người trong 1 bức ảnh, ta cần sử dụng các cửa sổ trượt (sliding window) với kích thước cố định, trượt trên bức ảnh để lần lượt khoanh vùng ảnh muốn predict label. Với mỗi vùng ảnh được khoanh vùng, ta resize về kích thước chuẩn và đưa vào mạng CNN vừa train, so sánh giá trị conf trả lại từ hàm softmax với ngưỡng tin cậy (confidence thresh) để predict label cho vùng ảnh này.

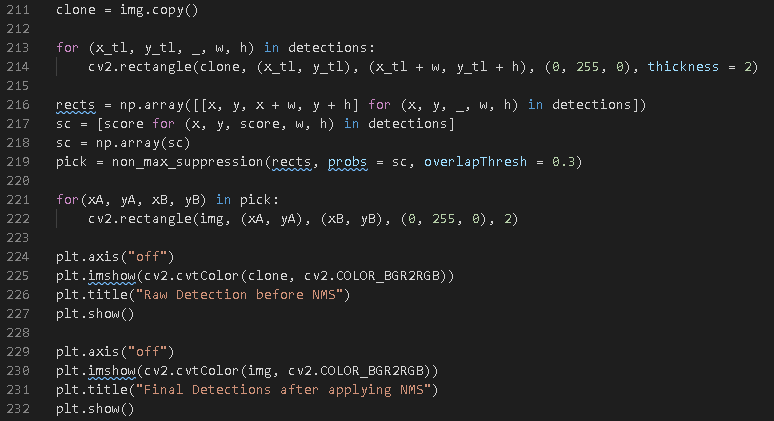


Trong đó, các tham số của sliding window được thiết lập như sau:



## 2. Thuật toán Non-maximum Suppression

Sau khi đưa ảnh qua mạng CNN, ta sẽ thu được rất nhiều proposals là boding box cho 1 mặt người duy nhất, dẫn đến việc dư thừa. Ta sẽ sử dụng thuật toán Non-maximum Suppression (NMS) để giải quyết vấn đề này. Tư tưởng chính của thuật toán là giữ lại box có giá trị confidence cao nhất cho mỗi mặt người và loại bỏ các box dư thừa với ngưỡng overlap threshold chọn trước.

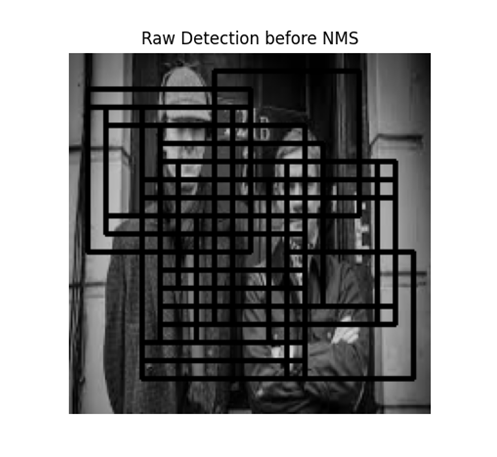


## 3. Thực hiện testing model

Ta sẽ xử lý ảnh dưới đây bằng 2 thuật toán đã nêu cùng mạng CNN vừa train và quan sát kết quả:



Kết quả khi chưa áp dụng thuật toán NMS:



Kết quả sau khi áp dụng thuật toán NMS:



# VII, Nhận xét về kết quả sản phẩm, độ hiệu quả của thuật toán

Từ kết quả đầu ra của quá trình test model, ta nhận thấy các vùng ảnh được khoanh vùng có chứa mặt người tuy nhiên chưa thực sự sát với kì vọng và còn 1 vài vùng ảnh nhận diện sai có thể do training data chưa quá đa dạng. Để khắc phục, ta có thể thu thập thêm nhiều nguồn dữ liệu ảnh đa dạng hơn trên Internet hoặc từ thực tế. Ngoài ra, quá trình training model tốn lượng thời gian tính toán khá lớn khi batch\_size và num\_steps lớn.

# VIII, Kết luận và hướng phát triển

Thay vì tìm ra phương hướng phát triển công nghệ nhận diện khuôn mặt người với độ chính xác cao, bài viết tập trung vào việc đưa ra hướng tiếp cận và cái nhìn trực quan về quá trình thiết kế, đào tạo mô hình mạng CNN trong lĩnh vực Machine Learning/Deep Learning. Có thể thấy từ kết quả đầu ra của mạng được thiết kế trong bài viết này, mô hình CNN có tiềm năng giải quyết các bài toán có dữ liệu đầu vào mang nhãn hoặc được đào tạo qua nhiều lớp khá dễ dàng, tuy nhiên đòi hỏi thời gian tính toán lớn. Hiện nay, để cải thiện về hiệu suất tính toán đối với lớp bài toán Object Detection sử dụng mô hình CNN, ta có thể tham khảo thuật toán R-CNN và Faster R-CNN.

Nguồn code tham khảo: <https://github.com/dacthai2807/Face-detection-CNN.git>