**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA HỆ THỐNG THÔNG TIN**



**BÁO CÁO ĐỒ ÁN**

**MÔN HỌC: KHAI THÁC DỮ LIỆU**

**Đề tài: Phân loại, nhận diện cảm xúc khuôn mặt qua hình ảnh**

GVHD: Nguyễn Hồ Duy Trí

**Nhóm sinh viên thực hiện:**

1.Nguyễn Văn Mạnh MSSV: 17520738

2. Nguyễn Kim Thảo MSSV: 17521065

3. Cao Thị Sin MSSV: 17520983

4. Nguyễn Ngọc Lan Thảo MSSV: 17521067

🙡🙢Tp. Hồ Chí Minh, 16/12/2019🙠🙣

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

*……., ngày……...tháng……năm 2019*

**Người nhận xét**

*(Ký tên và ghi rõ họ tên****)***

**MỤC LỤC**

[NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN 2](#_Toc13483716)

[MỤC LỤC 3](#_Toc13483717)

[LỜI MỞ ĐẦU 5](#_Toc13483718)

[DANH MỤC HÌNH ẢNH](#_Toc13483719)

[CHƯƠNG 1: TÌM HIỂU MẠNG CNNs 7](#_Toc13483720)

[1.1 Convolutional là gì? 7](#_Toc13483721)

[1.2 Cấu trúc của mạng CNN 8](#_Toc13483722)

[1.2.1 Trường tiếp nhận cục bộ (local receptive field) 9](#_Toc13483723)

[1.2.2 Trọng số chia sẻ (shared weight and bias) 11](#_Toc13483724)

[1.2.3 Lớp tổng hợp (pooling layer) 12](#_Toc13483725)

[1.3 Cách chọn tham số cho CNN 14](#_Toc13483726)

[CHƯƠNG 2: MÔ TẢ TẬP DỮ LIỆU, TIỀN XỬ LÍ DỮ LIỆU 15](#_Toc13483727)

[2.1 Mô tả tập dữ liệu 15](#_Toc13483728)

[2.2 Các bước tiền xử lí dữ liệu 16](#_Toc13483729)

[CHƯƠNG 3: NHẬN DIỆN VÀ PHÂN LOẠI CẢM XÚC VỚI CNNs 21](#_Toc13483730)

[3.1 Tập dữ liệu thuộc tính và tập phân lớp trên các tập training và test 21](#_Toc13483731)

[3.2 Xử lí dữ liệu đầu vào để xây dựng model 21](#_Toc13483732)

[3.3 Xây dựng model 22](#_Toc13483733)

[3.4 Đánh giá model 27](#_Toc13483734)

[CHƯƠNG 4: CHƯƠNG TRÌNH 32](#_Toc13483735)

[CHƯƠNG 5: KẾT LUẬN 33](#_Toc13483736)

[5.1 Kết luận chung 33](#_Toc13483737)

[5.2 Ưu điểm, nhược điểm 33](#_Toc13483738)

[5.2.1 Ưu điểm 33](#_Toc13483739)

[5.2.2 Nhược điểm 33](#_Toc13483740)

[5.3 Hướng phát triển 34](#_Toc13483741)

[TÀI LIỆU THAM KHẢO 35](#_Toc13483742)

**LỜI MỞ ĐẦU**

1. **Lý do chọn đề tài:**

Trong những năm qua, thế giới đã và đang được chứng kiến những bước tiến khổng lồ về chất lượng cùng “độ thần thánh” của các sản phẩm công nghệ chúng ta vẫn sử dụng hàng ngày. Bạn đã bao giờ đặt ra câu hỏi về những tiện ích đó hay deep learning chưa?

- Những trợ lý ảo giọng nói như Alexa của Amazon, Siri của Apple, Cortana của Microsoft cùng các hệ thống nhận diện giọng nói có mặt trên hầu như mọi sản phẩm của Google đang đồng loạt nở rộ trên nhiều nền tảng, giúp người dùng thực hiện nhiều tác vụ khác nhau

- Ngoài nhận diện giọng nói, dịch tự động và các công nghệ xử lý ngôn ngữ tự nhiên khác cũng đang ngày càng ưu việt hơn với những ông lớn như Google, Microsoft, Facebook, Baidu,… liên tục tung ra những tính năng độc đáo. Google Translate hiện nay có khả năng xử lý câu văn nói từ một sang 32 ngôn ngữ khác, dịch văn viết qua lại giữa 103 ngôn ngữ, thậm chí có thể dịch (tức thời) real-time ngay khi bạn lia ống kính điện thoại qua một tấm biển quảng cáo tiếng nước ngoài, như hình ảnh dưới đây.

- Và rồi chúng ta lại có công nghệ nhận diện hình ảnh – tính năng đã có mặt rộng rãi trên các sản phẩm của 4 gã khổng lồ nêu trên. Bạn có thể tìm kiếm và sắp xếp những bức ảnh của mình mà không cần phải gắn thẻ (tag) chúng, mà chỉ dựa cần trên những gì hiện diện trong hình, từ cụ thể như một chú chó, cảnh tuyết rơi cho đến trừu tượng như những cái ôm. Nhiều sản phẩm trong số này thậm chí còn có thể đọc mô tả lại các yếu tố trên bức ảnh cho người dùng khiếm thị

- Điều mà nhiều người không nhận ra là tất cả các công nghệ này, về mặt bản chất đều xuất phát từ cùng một nguồn gốc. Chúng được phát triển từ “deep learning”, một nhánh đặc biệt trong trí tuệ nhân tạo (AI). Trên thực tế, họ tạo ra một thuật toán giúp máy tính có khả năng tự học rồi cho nó tiếp xúc với hàng terabyte các dữ liệu liên quan – chẳng hạn như vài trăm ngàn bức ảnh các gương mặt, hay những băng ghi giọng nói kéo dài hàng năm trời.

- Convolutional Neural Network (CNNs – Mạng nơ-ron tích chập) là một trong những mô hình Deep Learning tiên tiến giúp cho chúng ta xây dựng được những hệ thống thông minh với độ chính xác cao như hiện nay. Trong bài thực hành này, em đi vào nghiên cứu về mạng neural cũng như mạng Convolution (tích chập) cũng như ý tưởng của mô hình CNNs trong phân lớp ảnh (Image Classification), và áp dụng trong việc xây dựng hệ thống nhận diện và phân loại cảm xúc con người qua hình ảnh.

1. **Các bước xử lí cho bài toán phân loại, nhận diện cảm xúc khuôn mặt qua hình ảnh:**

Ảnh đầu vào

Tiền xử lí

Nhận dạng

Huấn luyện

Xử lí ảnh

Kết quả nhận được

1. **Bố cục đề tài:**

- Ngoài phần lý do chọn đề tài, mục tiêu của đề tài, đề tài có bố cục gồm 5 chương:

* Chương 1: Tìm hiểu mạng CNNs
* 1.1 Sibling window là gì?
* 1.2 Convolutional là gì?
* 1.3 Mạng nơ-ron tích chập (CNNs)

1.3.1 Lớp tích chập (convolution layer)

1.3.2 Lớp tổng hợp (pooling layer)

1.3.3 Fully connected layer:

1.3.4 Cách chọn tham số cho CNN

* Chương 2: Mô tả tập dữ liệu và tiền xử lý
* 2.1 Mô tả tập dữ liệu
* 2.2 Tiền xử lí dữ liệu
* Chương 3: Nhận diện và phân loại cảm xúc con người với CNNs
* 3.1 Tập dữ liệu thuộc tính và tập phân lớp trên các tập training và test
* 3.2 Xử lí dữ liệu đầu vào để xây dựng model
* 3.3 Xây dựng model
* 3.4 Đánh giá model
* Chương 4: Chương trình ứng dụng
* Chương 5: Kết luận
* 5.1 Kết luận chung
* 5.2 Ưu điểm, nhược điểm

5.2.1 Ưu điểm

5.2.2 Nhược điểm

* 5.3 Hướng phát triển

**DANH MỤC HÌNH ẢNH**

Hình 1.1 Sibling window

Hình 1.2 Mô hình convolutional neural network

Hình 1.3 Các lớp trong CNNs

Hình 1.4 Kiến trúc CNN

Hình 1.5 max pooling

Hình 1.6 Các loại pooling layer

Hình 2.1: các tập dữ liệu

Hình 2.2 Thông tin dataset

Hình 2.3 biểu đồ biểu diễn số lượng tập dữ liệu

Hình 2.4 Thông tin dữ liệu sau khi bỏ PrivateTest

Hình 2.5 dataset sau khi loại bỏ private test

Hình 2.6 Biểu đồ biểu diễn số lượng dataset sau khi loại bỏ private test

Hình 2.7 Đếm số lượng các tập public test và training

Hình 2.8 Chia vào tập tương ứng

Hình 2.9 Số dòng dữ liệu theo cảm xúc phân lớp trong tập train

Hình 2.10 Số lượng của từng loại cảm xúc trong data\_train

Hình 2.11 Số lượng cảm xúc trong data\_test

Hình 2.12 Dòng dữ liệu của hình thứ 3

Hình 2.13 Ép kiểu dữ liệu

Hình 2.14 Dùng căn bậc 2 để xác định size ảnh

Hình 2.15 Hình ảnh thu được

Hình 3.1 Khởi tạo giá trị

Hình 3.2 Chuẩn hóa dữ liệu

Hình 3.3 Duyệt file và số lượng tập test và tập train

Hình 3.4 Thêm các thư viện cần thiết

Hình 3.5 Khởi tạo các tham số cần thiết

Hình 3.6 Sơ đồ khối xây dựng model

Hình 3.7 Xây dựng model

Hình 3.8: Compile model

Hình 3.9 Train model

Hình 3.10 Lưu model

Hình 3.11 Độ chính xác trên tập test

Hình 3.12 Độ chính xác, tỉ lệ mất mát của tập trainning và test

Hình 3.13 Đồ thị Độ chính xác, tỉ lệ mất mát của tập trainning và test

Hình 3.14 Ma trận nhầm lẫn

Hình 3.15 Biểu đồ ma trận nhầm lẫn

Hình 3.16 Một số độ đo khác

Hình 4.1 Kết quả chạy demo

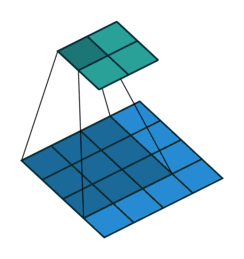
# TÌM HIỂU MẠNG CNNs

Convolutional Neural Networks là một trong những mô hình Deep Learning tiên tiến giúp cho chúng ta xây dựng được những hệ thống thông minh với độ chính xác cao như các 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.

Convolutional Neural Network là một kiến trúc lấy ý tưởng từ việc áp dụng các filters để tìm ra các features cho ảnh đầu vào, sau đó các sử dụng các features đó cho việc phân loại ảnh. Trong kiến trúc này có nhiều thành phần, nhưng core của nó là một thành phần được gọi là lớp Convolution (CONV). Lớp CONV này sẽ chứa các filters cho kiến trúc CNN. Tuy nhiên, thay vì phải tự đưa ra các filters thủ công như CV, các filters sẽ được tìm ra một cách tự động thông qua quá trình trainning. Có nghĩa là các giá trị trong mỗi một filter lúc này (chính là các weights) ban đầu sẽ được khởi tạo ngẫu nhiên, sau quá trình học mà chúng sẽ được cập nhật lại, và kết quả cố định cuối cùng được ổn định khi kết thúc training. Khi xây dựng CNN, chúng ta chỉ cần chỉ ra là ở mỗi lớp CONV cần số lượng bao nhiêu filter, size mỗi filter, giá trị stride, giá trị padding…

## Sibling window là gì?

Cửa sổ trượt (sliding window) là kỹ thuật mà ta dùng 1 cửa sổ (window, hay tên gọi khác là kernel) để trượt trên mỗi pixel của ảnh. Tại mỗi pixel trong quá trình trượt, ta áp dụng phép biến đổi giữa các pixel trên cửa sổ và các pixel tương ứng trên vùng ảnh.



*Hình 1.1 Sibling window*

Kỹ thuật sliding window thường ứng dụng để:

Phát hiện đối tượng: phát hiện khuôn mặt, phát hiện người đi bộ (pedestrian), phát hiện xe (bài toán giao thông), phát hiện biển báo giao thông, nhận diện kí tự, …

Nhận diện đối tượng: nhận diện khuôn mặt, nhận diện biển số xe, theo vết (tracking), …

Đếm đối tượng: trực tiếp hoặc gián tiếp.

## Convolutional là gì?

Convolution là một kỹ thuật quan trọng xử lý ảnh (digital image processing). Nó có mặt trong hầu hết các thuật toán làm mờ (Gausian Blur), hay làm rõ các đường (edge detector). Trong nhận dạng ảnh (deep learning image processing), convolution layer là một tầng biến đổi ma trận đầu vào để làm rõ và tách ra các đặc tính của hình ảnh mà vẫn bảo toàn tính tương quan không gian giữa đầu ra và đầu vào.

Vài khái niệm cơ bản:

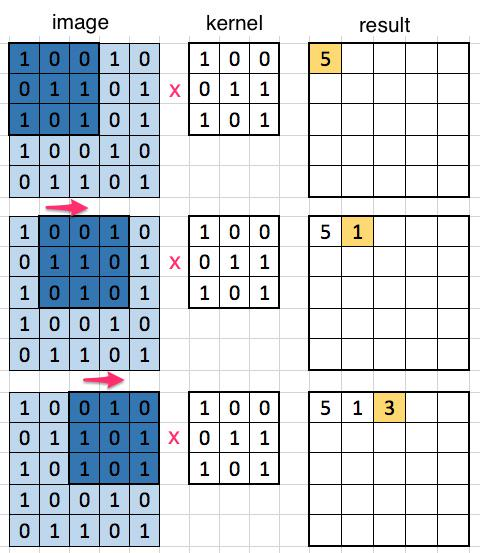
Filter, Kernel hay Feature Detector đều là cách gọi của ma trận lọc. Thông thường, ở các lớp đầu tiên của Conv Layer sẽ có kích thước là [5x5x3]

Convolved Feature, Activation Map hay Feature Map là đầu ra của ảnh khi cho bộ lọc chạy hết bức ảnh với phép tích vô hướng.

Depth là số lượng bộ lọc. Lưu ý: ở đây là số lượng bộ lọc (filter) chứ không phải số lượng kênh màu RBG như ở trên.

Stride được hiểu là khoảng cách dịch chuyển của bộ lọc sau mỗi lần tính. Ví dụ khi stride=2. Tức sau khi tính xong tại 1 vùng ảnh, nó sẽ dịch sang phải 2 pixel. Tương tự với việc dịch xuống dưới.

Là một cửa sổ trượt (Sliding Windows) trên một ma trận như mô tả hình dưới:



Bên trái là ma trận đầu vào, có thể là ảnh đã được số hóa.

Bên phải là *kernel* (filter hoặc feature detect) là một ma trận có kích thước nhỏ dùng để biến đổi ma trận đầu vào trong 2 vòng lặp lồng nhau convolution.

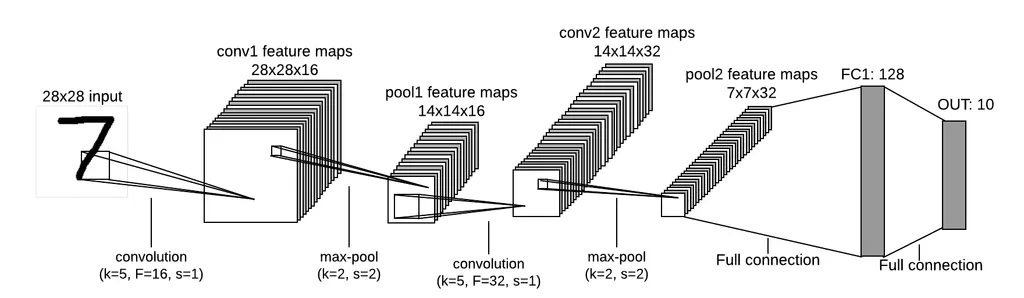
Từ trái qua phải, từ trên xuống dưới dịch chuyển cửa sổ có kích thước bằng với kernel trên ma trận đầu vào, thực hiện phép nhân từng phần tử cùng vị trí ở ma trận cửa số với kernel rồi tính tổng ra giá trị scalar điền vào ma trận kết quả được gọi là *Bản đồ đặc trưng (feature map)*.

## Mạng nơ-ron tích chập (CNNs)

Convolutional Neural Network (CNNs – Mạng nơ-ron tích chập) là một trong những mô hình Deep Learning tiên tiến. Nó giúp cho chúng ta 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.

CNN được sử dụng nhiều trong các bài toán nhận dạng các đối tượng trong ảnh. Để tìm hiểu tại sao thuật toán này được sử dụng rộng rãi cho việc nhận dạng (detection), chúng ta hãy cùng tìm hiểu về thuật toán này.

*Input image -> Convolutional layer (Conv) + Pooling layer (Pool) -> Fully connected layer (FC) -> Output.*

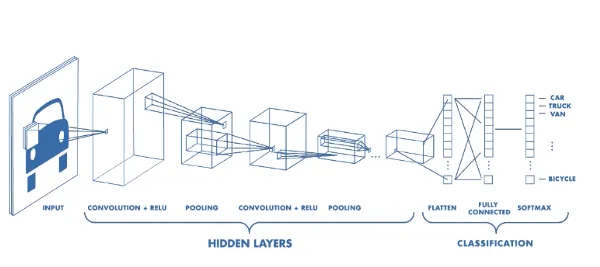


*Hình 1.2 Mô hình convolutional neural network*

CNNs gồm hai thành phần:

Phần tầng ẩn hay phần rút trích đặc trưng (hidden layers): trong phần này, mạng sẽ tiến hành tính toán hàng loạt phép **tích chập** và phép **hợp nhất** (pooling) để phát hiện các đặc trưng. Ví dụ: nếu ta có hình ảnh con ngựa vằn, thì trong phần này mạng sẽ nhận diện các sọc vằn, hai tai, và bốn chân của nó.

Phần phân lớp (classification): tại phần này, một lớp với các liên kết đầy đủ sẽ đóng vai trò như một bộ phân lớp các đặc trưng đã rút trích được trước đó. Tầng này sẽ đưa ra xác suất của một đối tượng trong hình.



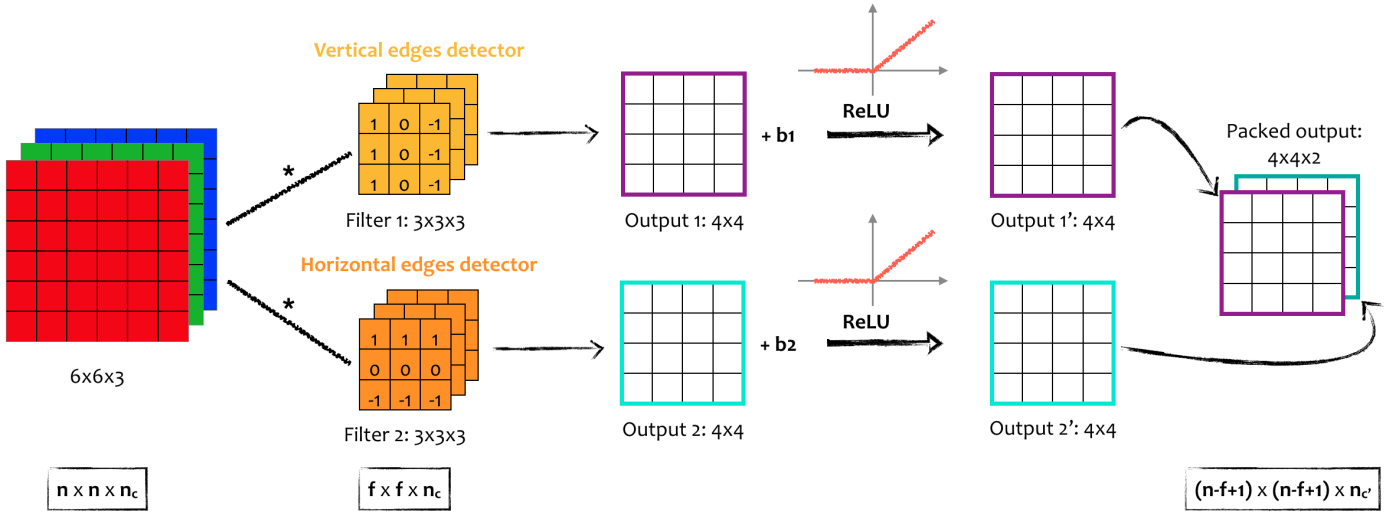
*Hình 1.3 Các lớp trong CNNs*

### Lớp tích chập (convolution layer)

Trong thực tế, tích chập được thực hiện hiện trên không gian 3 chiều. Vì mỗi hình ảnh được biểu diễn dưới dạng 3 chiều: rộng, cao, và sâu. Chiều sâu ở đây chính là giá trị màu sắc của hình (RGB).

Ta thực hiện phép tích chập trên đầu vào nhiều lần khác nhau. Mỗi lần sử dụng một kernel/filter khác nhau. Kết quả ta sẽ thu được những bản đồ đặc trưng khác nhau. Cuối cùng, ta kết hợp toàn bộ bản đồ đặc trưng này thành kết quả cuối cùng của tầng tích chập.

Tương tự như mạng nơ-ron thông thường, ta sử dụng một hàm kích hoạt (activate function) để có đầu ra dưới dạng phi tuyến. Trong trường hợp CNN, đầu ra của phép tích chập sẽ đi qua hàm kích hoạt nào đó ví dụ như hàm **ReLU** (rectified linear units).



*Hình 1.4 Kiến trúc CNN*

Trong quá trình trượt kernel/filter (sibling window) trên dữ liệu đầu vào, ta sẽ quy định một bước nhảy (stride) với mỗi lần di chuyển. Thông thường ta lựa chọn thường chọn bước nhảy là 1. Nếu kích thước bước nhảy tăng, kernel/filter sẽ có ít ô trùng lắp.

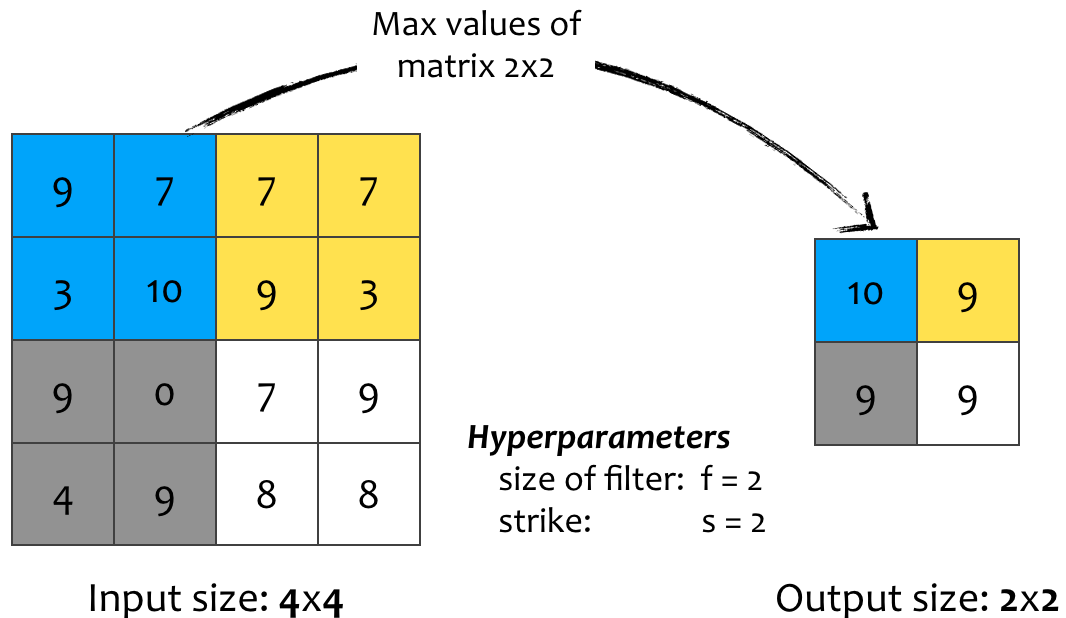
Bởi vì kích thước đầu ra luôn nhỏ hơn đầu vào nên ta cần một phép xử lí đầu vào để đầu ra không bị co giãn. Đơn giản ta chỉ cần thêm một lề nhỏ vào đầu vào. Một lề với giá trị 0 sẽ được thêm vào xung quanh đầu vào trước khi thực hiện phép tích chập.

Thông thường, sau mỗi tầng tích chập, ta sẽ cho kết quả đi qua một **tầng hợp nhất** (pooling layer). Mục đích của tầng này là để nhanh chóng giảm số chiều. Việc này giúp giảm thời gian học và hạn chế việc **overfitting**.

### Lớp tổng hợp (pooling layer)

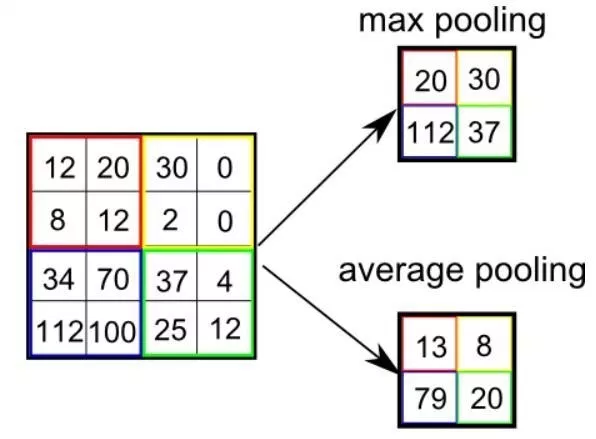
Pooling layer thường được dùng giữa các convolutional layer, để giảm kích thước dữ liệu, giảm số lượng neuron nhưng vẫn giữ được các thuộc tính quan trọng. Kích thước dữ liệu giảm giúp giảm việc tính toán trong model.

Gọi pooling size kích thước K\*K. Input của pooling layer có kích thước H\*W\*D, ta tách ra làm D ma trận kích thước H\*W. Với mỗi ma trận, trên vùng kích thước K\*K trên ma trận ta tìm maximum hoặc average của dữ liệu rồi viết vào ma trận kết quả. Quy tắc về stride và padding áp dụng như phép tính convolution trên ảnh.



*Hình 1.5 max pooling*

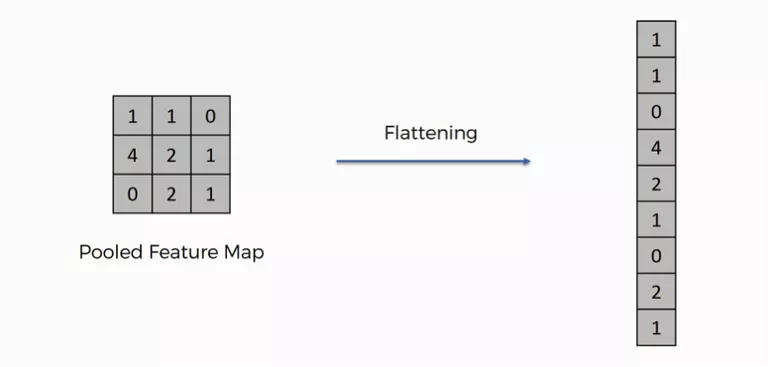
Có 2 loại pooling layer phổ biến là: max pooling và average pooling. Thủ tục thường được sử dụng nhất là max pooling.



*Hình 1.6 Các loại pooling layer*

### Fully connected layer:

Fully-Connected Layer (FC Layer) chính là một mạng NN được gắn vào phần cuối của CNNs. Phần FC-Layer này chính là nơi từ các feature được extract bởi phần convolution và pooling tạo ra kết quả cuối cùng (Classification hoặc Regression).



### Cách chọn tham số cho CNN

Số các convolution layer: càng nhiều các convolution layer thì performance càng được cải thiện.

Filter size: thường filter theo size 5×5 hoặc 3×3

Pooling size: thường là 2×2 hoặc 4×4 cho ảnh đầu vào lớn

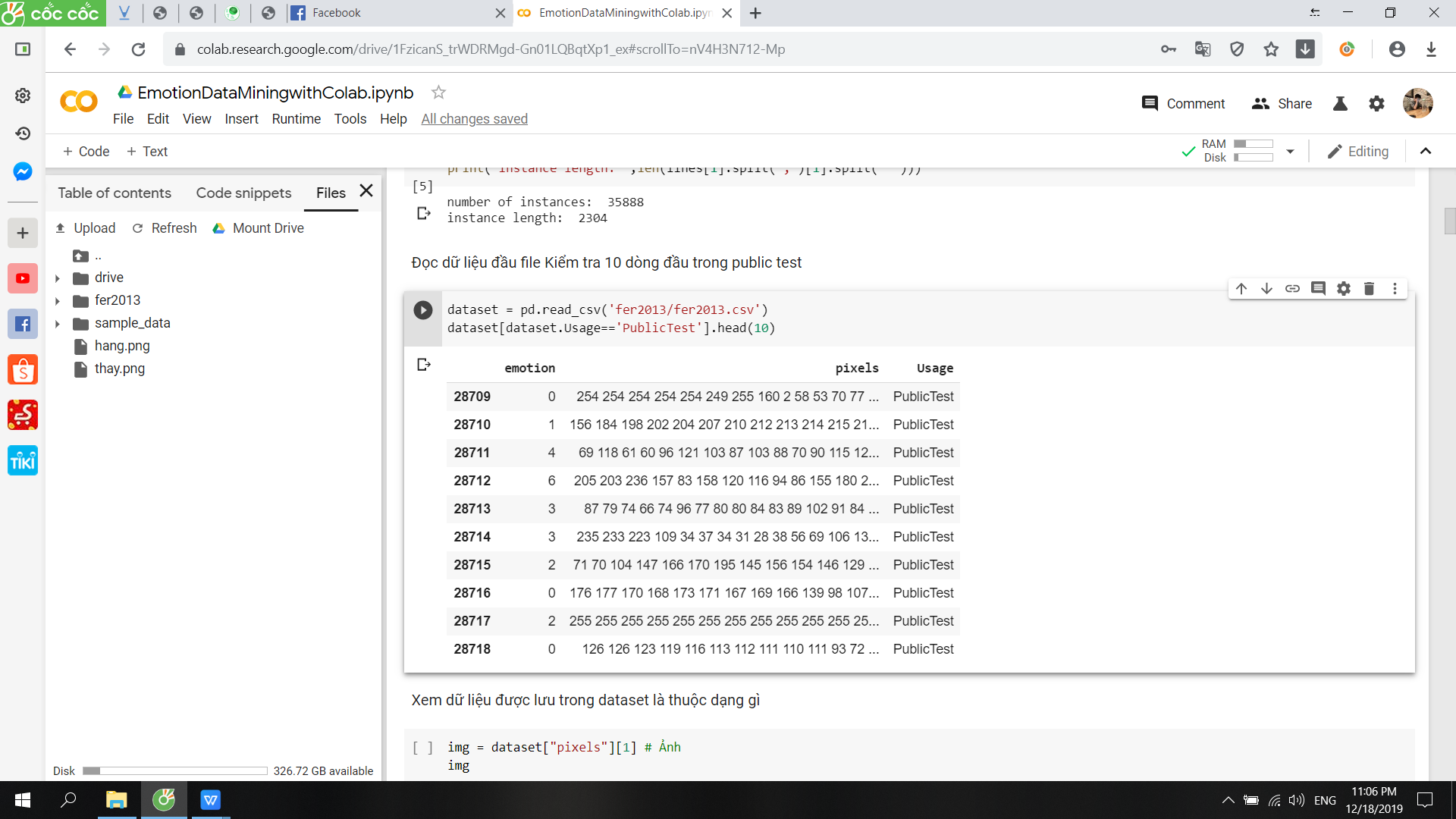
Cách cuối cùng là thực hiện nhiều lần việc train test để chọn ra được param tốt nhất.

# MÔ TẢ TẬP DỮ LIỆU, TIỀN XỬ LÍ DỮ LIỆU

Công việc đầu tiên chúng ta phải làm là chuẩn bị và thu thập tập một bộ dữ liệu. Tập dữ liệu yêu cầu phải phù hợp với vấn đề đang giải quyết. Nếu không tìm ra tập dữ liệu phù hợp thì phải tạo tập dữ liệu cho riêng mình. Sử dụng tập dữ liệu trên Kaggle:<https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition-challenge/data>

## Mô tả tập dữ liệu

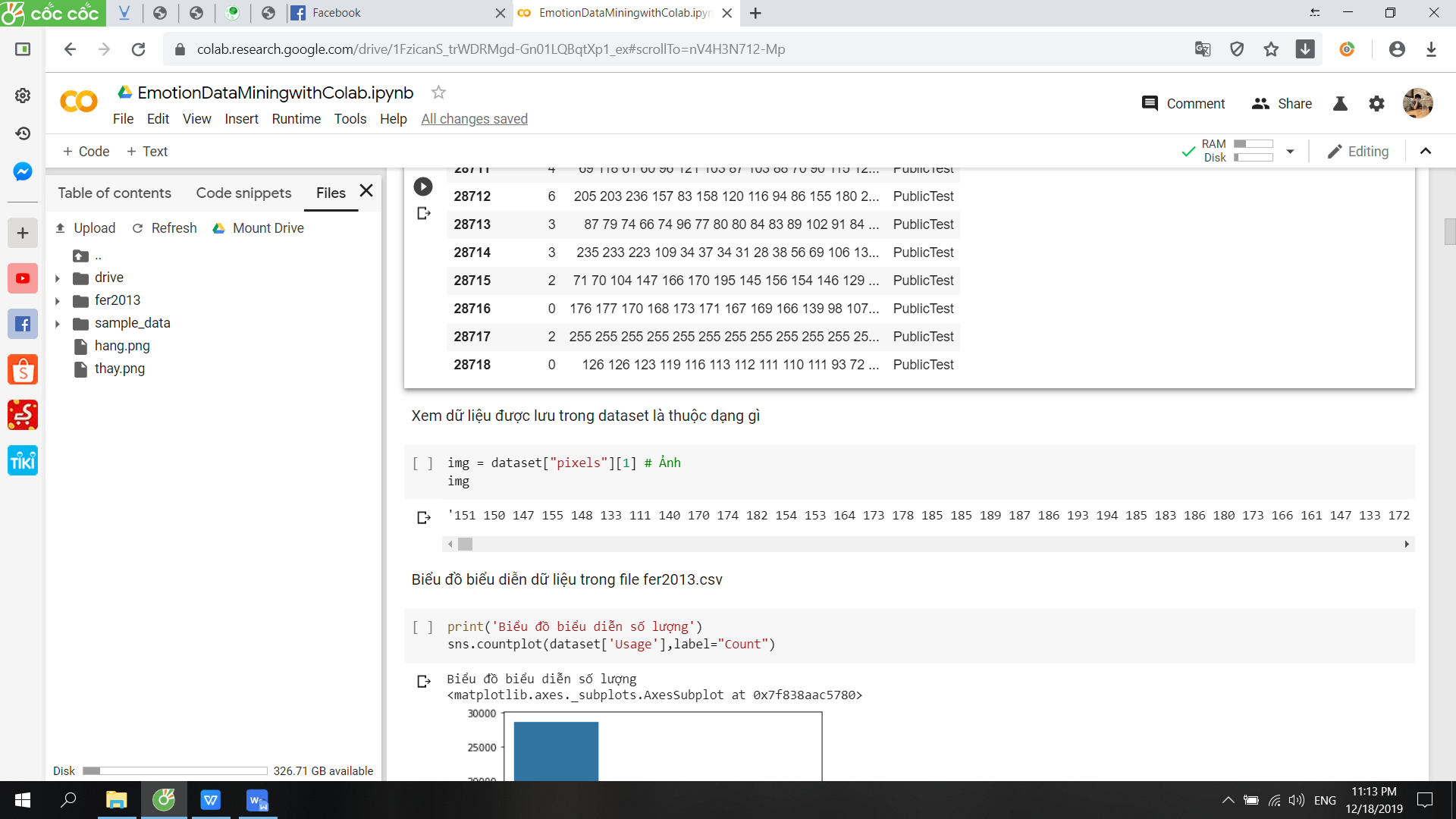
Dữ liệu gồm 48 x 48 pixel ảnh thang độ xám của khuôn mặt. Các hình ảnh đã được chỉnh sửa tự động để khuôn mặt được căn giữa và chiếm các khoảng trống như nhau ở mỗi hình ảnh.



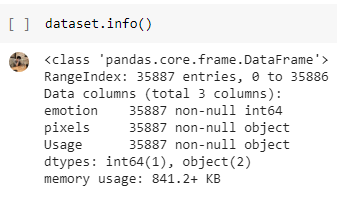
*Hình 2.1: các tập dữ liệu*

Tập dữ liệu chứa 3 cột là “emotion” ,“pixel” và “Usage” .

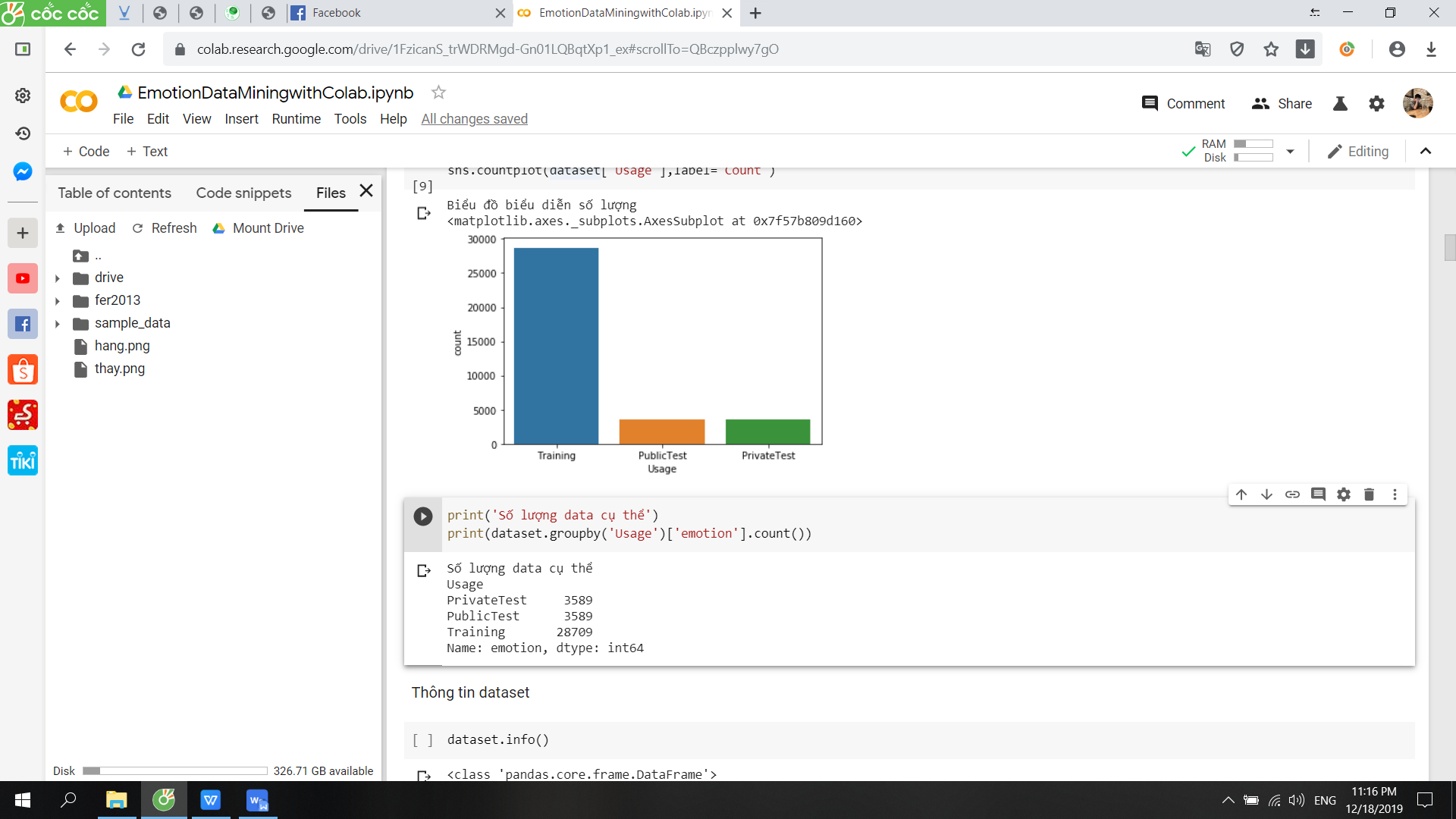
Cột “emotion” chứa số từ 0 đến 6 đại diện cho cảm xúc thể hiện trong biểu cảm khuôn mặt có trong hình ảnh theo một trong 7 loại biểu cảm sau: 0 = Giận dữ, 1 = Ghê tởm, 2 = Sợ hãi, 3 = Hạnh phúc, 4 = Buồn, 5 = Bất ngờ, 6 = Trung lập.



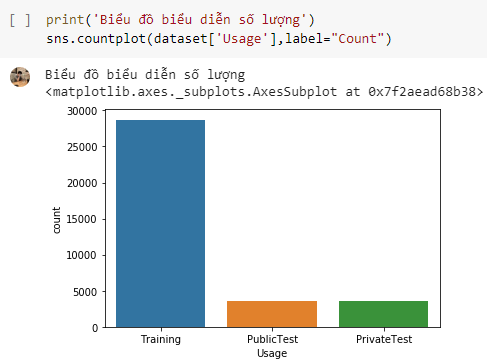
Cột “pixel” chứa một chuỗi các số được bao quanh trong dấu ngoặc kép cho mỗi hình ảnh. Nội dung của chuỗi này là giá trị các pixel của hình ảnh được phân biệt bằng dấu cách theo thứ tự chính hàng.



*Hình 2.2 Thông tin dataset*



Tập dữ liệu bao gồm 355887 dòng, chia làm 3 phần chính: Với 28709 dòng được sử dụng để Training, 3589 dòng được sử dụng làm public test và 3589 dòng còn lại dùng để private test.



*Hình 2.3 biểu đồ biểu diễn số lượng tập dữ liệu*

## Tiền xử lí dữ liệu

Việc chúng ta cần xử lí dữ liệu ở đây là sao cho nó phù hợp với thuật toán, training để tạo model một cách chính xác.

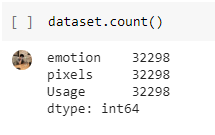
Việc xác định mô hình là bước quan trọng nhất. Ta phải chọn mô hình hay thuật toán để huấn luyện sao cho phù hợp với tập dữ liệu để đem lại độ chính xác cao nhất.

Trong tập dữ liệu này, chúng ta không sử dụng dữ liệu private test nên sẽ loại bỏ các dòng dữ liệu có thuộc tính Usage = “PrivateTest” khỏi dataset.

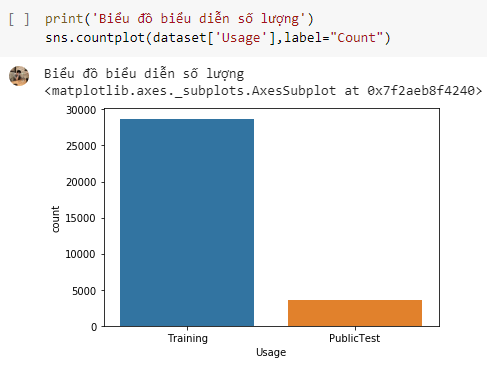
Loại bỏ các dòng private test (Có thể loại bỏ hoặc không, số lượng dòng private test không đáng kể)



Hình 2.4 Thông tin dữ liệu sau khi bỏ PrivateTest

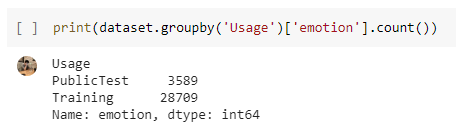


*Hình 2.5 dataset sau khi loại bỏ private test*



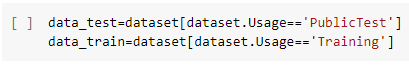
*Hình 2.6 Biểu đồ biểu diễn số lượng dataset sau khi loại bỏ private test*

Đếm số lượng và chia các tập test train tương ứng:



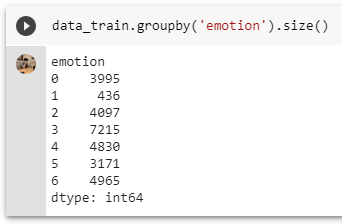
*Hình 2.7 Đếm số lượng các tập public test và training*

Chia vào các tập tương ứng:

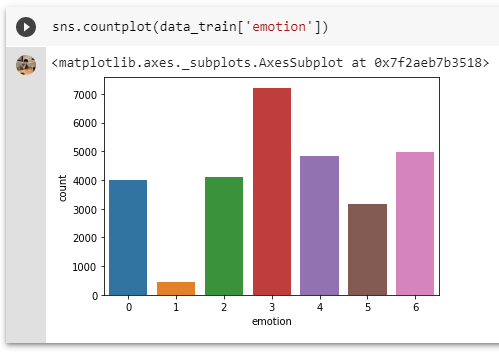


*Hình 2.8 Chia vào tập tương ứng*

Xem số lượng dữ liệu cảm xúc trong data\_train có mô tả qua biểu đồ:



*Hình 2.9 Số dòng dữ liệu theo cảm xúc phân lớp trong tập train*



*Hình 2.10 Số lượng của từng loại cảm xúc trong data\_train*

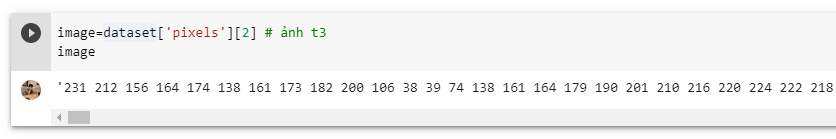
Xem số lượng cảm xúc trong data\_test và biểu đồ tương ứng:



*Hình 2.11 Số lượng cảm xúc trong data\_test*

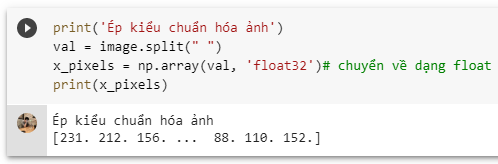
Kiểu dữ liệu ảnh:

Lấy một dòng dữ liệu của thuộc tính pixel kiểm tra xem kiểu dữ liệu và cấu trúc



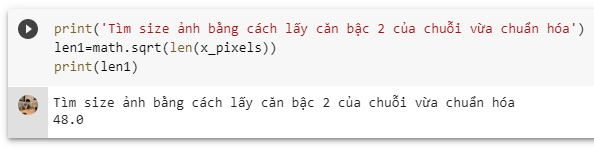
*Hình 2.12 Dòng dữ liệu của hình thứ 3*

Đưa dữ liệu từ chuỗi về mảng và ép kiểu dữ liệu về float:



*Hình 2.13 Ép kiểu dữ liệu*

Xác định size ảnh bằng cách lấy căn bậc 2 của chuỗi vừa chuẩn hóa:



*Hình 2.14 Dùng căn bậc 2 để xác định size ảnh*

Kết quả thu được 48, suy ra từ pixel ta thu được ảnh hiển thị có kích thước 48 x 48. Sau đó ta chuyển ảnh về dạng ma trận rồi hiển thị.



*Hình 2.15 Hình ảnh thu được*

# NHẬN DIỆN VÀ PHÂN LOẠI CẢM XÚC VỚI CNNs

## Tập dữ liệu thuộc tính và tập phân lớp trên các tập training và test

* Khởi tạo các giá trị chuẩn bị cho training:

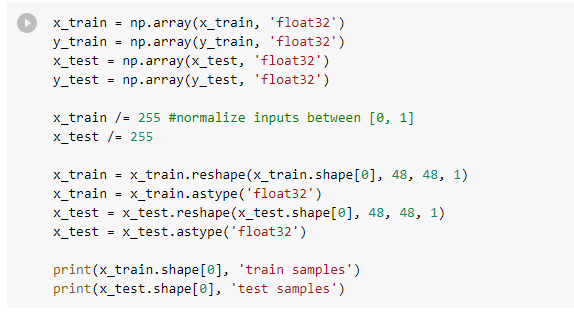


Hình 3.1 Hình Khởi tạo giá trị

* Phân chia thành các tập dữ liệu thuộc tính và tập phân lớp trên các tập training và test.

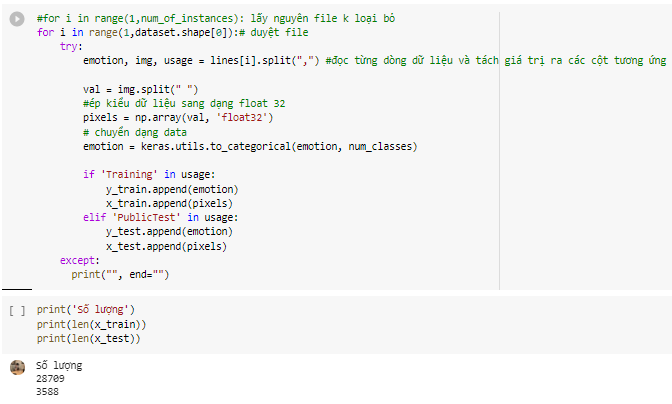
## Xử lí dữ liệu đầu vào để xây dựng model

* Chuẩn hóa các giá trị đầu vào về miền giá trị thuộc [0;1] và đưa các kiểu dữ liệu về float32. Đồng thời đưa về ma trận 2 chiều 48x48 với màu gray.



Hình 3.2 Chuẩn hóa dữ liệu

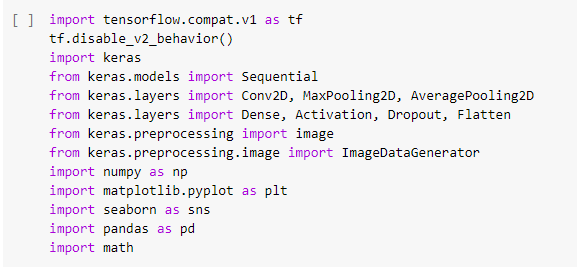
* Đọc từng dòng dữ liệu trong file, ép kiểu dữ liệu và chuyển dạng data:



Hình 3.3 Duyệt file và số lượng tập test và tập train

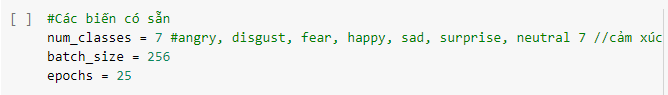
## Xây dựng model

* Load các thư viện cần thiết cho việc xây dựng model



Hình 3.4 Thêm các thư viện cần thiết

* Khởi tạo các biến để làm tham số cần thiết:



Hình 3.5 Khởi tạo các tham số cần thiết

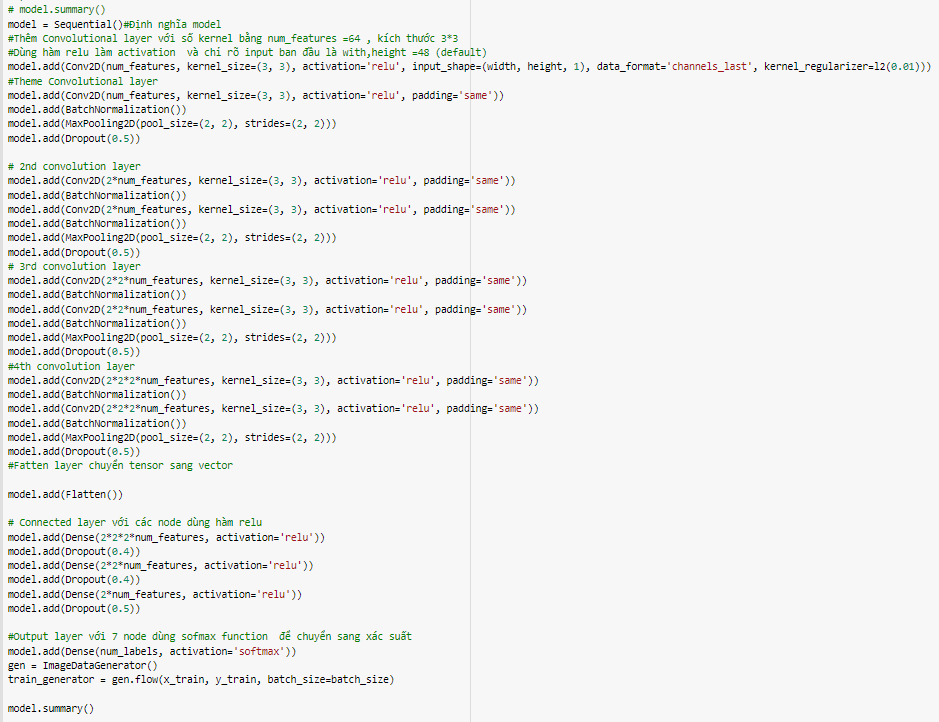
* Sơ đồ khối của model

Model nhận đầu vào là các ảnh (48 x 48 x 1), đầu ra là lớp FC softmax với nodes = 7. Ở giữa là 1 kiến trúc CNN với thứ tự các lớp như sau: [INPUT] => [CONV(relu)] => [CONV(relu)] => [BATCH NORM] => [MAXPOOL] => [DROPOUT] => [CONV (relu)] => [BATCH NORM]=> [CONV(relu)] => [BATCH NORM] => [MAXPOOL] => [DROPOUT] => [CONV(relu)] => [BATCH NORM] => [CONV(relu)] => [BATCH NORM] => [MAXPOOL] => [DROPOUT] => [CONV(relu)] => [BATCH NORM] => [CONV(relu)] => [BATCH NORM] => [MAXPOOL] => [DROPOUT] => [FLATTEN] => [FC(relu)] => [DROPOUT] => [FC(relu)] => [DROPOUT] => [FC(relu)] => [DROPOUT] => [FC(softmax)].=> [OUTPUT]

|  |  |
| --- | --- |
|  |  |

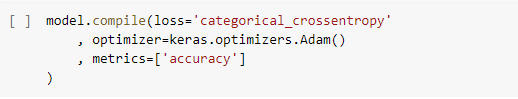
Hình 3.6 Sơ đồ khối xây dựng model

* Xây dựng model



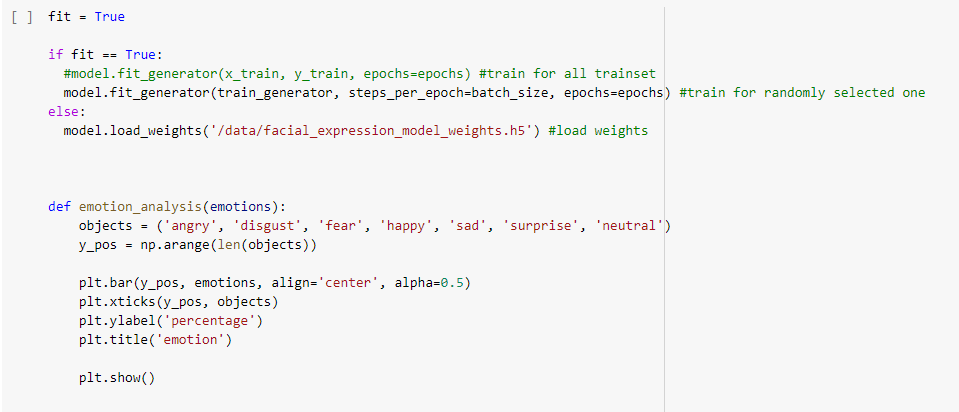
Hình 3.7 Xây dựng model

* Compile model



*Hình 3.8: Compile model*

* Training



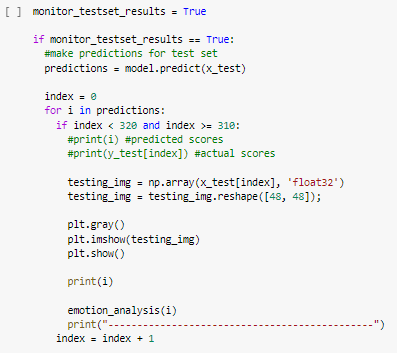
Hình 3.9 Train model

Khi khởi tạo model

* Lưu model:



Hình 3.10 Lưu model



*Hình 3.11: Load hình ảnh file nhận diện test sử dụng data trong csv*

Khi chúng ta khởi tạo một model thì có các method ta cần lưu ý là :

*compile*: Sau khi build model xong thì compile nó có tác dụng biên tập lại toàn bộ model của chúng ta đã build. Ở đây chúng ta có thể chọn các tham số để training model như : thuật toán training thông qua tham số *optimizer*, function loss của model chúng ta có thể sử dụng mặc định hoặc tự build thông qua tham số *loss*, chọn metrics hiện thị khi model được training

*summary*: method này giúp chúng ta tổng hợp lại model xem model có bao nhiêu layer, tổng số tham số bao nhiêu, shape của mỗi layer..

*fit*: dùng để đưa data vào training để tìm tham số model

*predict*: dùng để predict các new instance

*evaluate*: để tính toán độ chính xác của model

*history*: dùng để xem accuracy,loss qua từng epochs . Thường dùng với matplotlib để vẽ chart.

Sau khi xây dựng được model, ta cần định nghĩa hàm loss và 1 optimizer để có thể bắt đầu quá trình training. Keras hỗ trợ rất nhiều built-in loss function. Một số hàm loss phổ biến như: mean squared error với bài toán regression, binary crossentropy cho bài toán classification, categorical crossentropy với bài toán multiclass-classification. Đối với optimizers, tất cả các optimizer của Kera đều được xây dựng dựa trên thuật toán Gradient Descent. Trong số các optimizers mà Keras cung cấp thì Adam optimizer được sử dụng nhiều hơn cả do khả năng hội tụ nhanh của nó.

Giải thích một số layer trong model:

Conv2D: là convolution layer dùng để lấy feature từ image. tham số cần chú ý:

filters: số filter của convolution layer

kernel\_size: size window search trên image

strides: bước nhảy mỗi window search

padding: “same” là dùng padding, “valid” là không.

BatchNormalization: Giúp chuẩn hóa dữ liệu về một miền giá trị nhỏ hơn, phương pháp giúp mô hình có khả năng hội tụ nhanh hơn. Ngày nay, khi xây dựng các mô hình DL, phương pháp Batch Normalization được sử dụng rất nhiều vì hiệu quả rõ rệt mà nó mang lại.

MaxPooling2D:  Giảm số chiều mà vẫn giữ được những feature nổi bật(dùng max) và giúp giảm parameter khi training.

Flatten: nhiệm vụ của Flatten layer đúng như cái tên của nó, "trải phẳng" ma trận ra thành 1 vector, vd : shape : 20x20 qua layer này sẽ là 400x1

Dropout: layer này dùng để ẩn 1 số unit trong hidden trong quá trình training, với việc ẩn đi 1 số unit trong quá trình training sẽ tránh được việc mô hình quá fit với dữ liệu train để tránh overfitting. Với dropout, keras cung cấp sẵn 1 layer, chúng ta chỉ cần gọi layer ra, truyền giá trị dropout rate (0-1) và sử dụng.

Dense (FC): layer này sử dụng như một layer neural network bình thường.

Một số tham số cần chú ý:

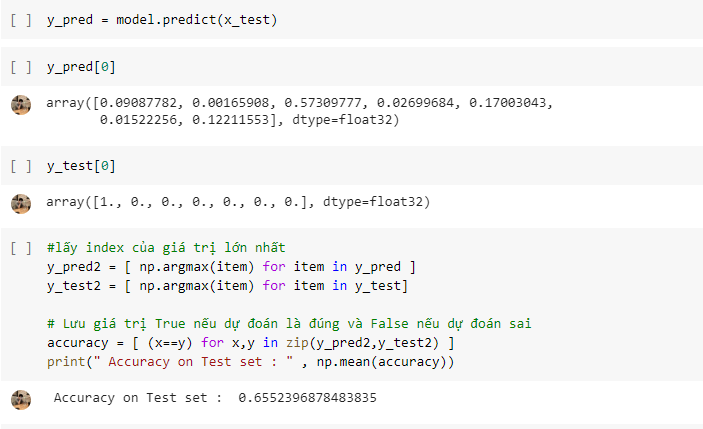
*Activation*: Hàm kích hoạt hay **activation functions** được sinh ra với mục đích **bẻ gãy sự tuyến tính** của mạng nơ ron. Các hàm này có thể hiểu đơn giản như một bộ lọc để quyết định xem thông tin có được đi qua nơ-ron hay không. Trong quá trình huấn luyện mạng nơ ron, các hàm kích hoạt đóng vai trò quan trọng trong việc điều chỉnh độ dốc của đạo hàm.

*RELU*: được dựa trên tư tưởng của việc loại bỏ bớt những tham số không quan trọng trong quá trình training và điều đó là cho mạng của chúng ta trở nên nhẹ hơn và việc training cũng nhanh chóng và có hiệu quả hơn. Hàm này thực hiện một việc rất đơn giản như sau: giữ nguyên những giá trị đầu vào lớn hơn 0, nếu giá trị đầu vào nhỏ hơn 0 thì coi là 0.

*Softmax:*  là một cách ràng buộc đầu ra của các mạng nơ ron phải có tổng bằng 1. Các giá trị đầu ra của hàm **softmax** có thể được coi như là một phân phối xác suất của các biến đầu ra. Nó rất hữu ích trong bài toán phân loại đa lớp.

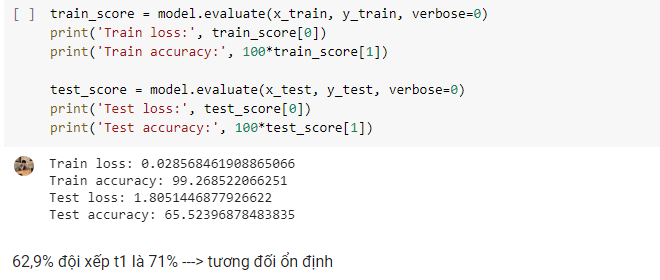
## Đánh giá model

* Độ chính xác của model trên tập test:



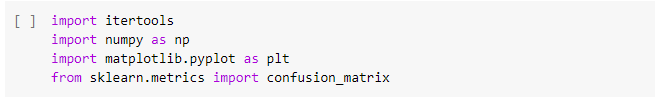
Hình 3.12 Độ chính xác trên tập test

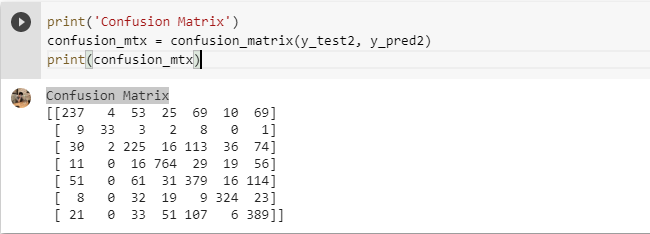
* Độ chính xác, tỉ lệ mất mất mát của tập training và test



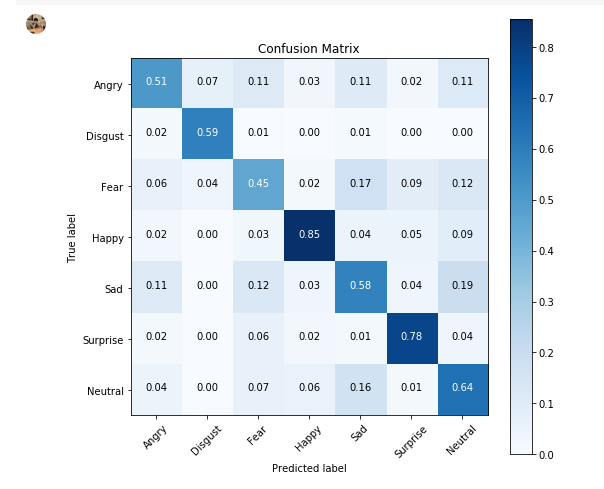
Hình 3.13 Độ chính xác, tỉ lệ mất mát của tập trainning và test

* Ma trận nhầm lẫn





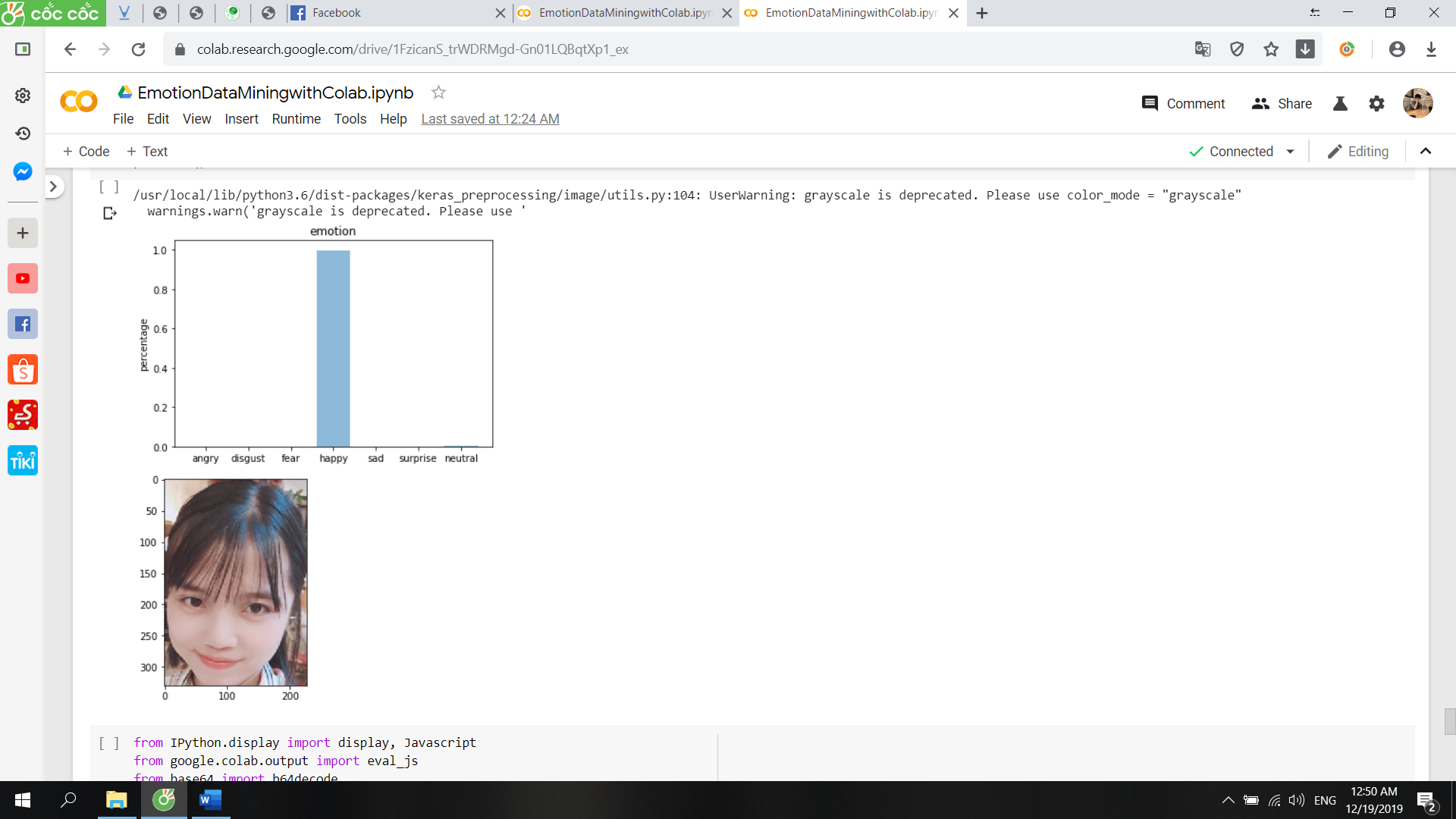
Hình 3.14 Ma trận nhầm lẫn



Hình 3.15 Biểu đồ ma trận nhầm lẫn

# CHƯƠNG TRÌNH

* Toàn bộ source code được lưu trữ trên google colab:
* Google Colab: <https://colab.research.google.com/drive/1FzicanS_trWDRMgd-Gn01LQBqtXp1_ex>
* Kết quả chạy demo một số hình:



Ảnh có chứa ảnh chụp màn hình

Mô tả được tạo tự động

Hình 4.1 Kết quả chạy demo

# KẾT LUẬN

## Kết luận chung

* Kết quả đồ án đã thực hiện được một số việc sau:
* Nhận diện và phân tích cảm xúc trên khuôn mặt con người qua hình ảnh với độ chính xác tương đối (Trên 60% với model đã sử dụng).
* Hiểu và áp dụng các kỹ thuật tiền xử lí trên dữ liệu.
* Áp dụng được thuật toán CNNs để giải quyết việc phân loại và nhận diện cảm xúc khuôn mặt
* Kết Luận: Xây dựng được chương trình có các tính năng cơ bản sau:
* Huấn luyện được mô hình cho bài toán trình bày
* Có thể dự đoán dựa vào một mẫu dữ liệu bất kì là hình khuôn mặt
* Đánh giá được kết quả cũng như thực nghiệm các độ đo
* Khó khăn khi thực hiện đề tài
* Kiến thức về deeplearning và neural network còn cơ bản.
* Thời gian thực hiện đề tài tương đối ngắn, không dài.

## Ưu điểm, nhược điểm

### Ưu điểm

* Chương trình đáp ứng được yêu cầu cơ bản được đặt ra
* Dễ dàng sử dụng cũng như kiểm tra, đánh giá kết quả
* Độ chính xác bài toán tương đối .Tỉ lệ chính xác tương đối (đội thắng cuộc là 71%)

### Nhược điểm

* Model chỉ có độ chính xác cao trên tập dữ liệu được test, thực tế thì do sự phân chia dữ liệu trong bộ dữ liệu không nhiều , nên kết quả có thể bị ảnh hưởng tương đối đáng kể
* Mô hình chưa thật sự tối ưu
* Thực thi chương trình phải chạy bằng lệnh, chưa có giao diện cho người dùng thao tác.
* Chưa có phần nhận diện khuôn mặt trong ảnh, cần phải xử lý ảnh để lấy được khuôn mặt trước khi cho vào kiểm tra.

## Hướng phát triển

Đề tài này tương đối phức tạp, nhóm chưa có nhiều kiến thức để vận dụng và xây dựng model hoàn chỉnh cho bài toán.Để hoàn thiện thêm, nhóm sẽ tiếp tục nghiên cứu thêm về các kiến thức deeplearning để hoàn thiện cho đồ án.

Nhóm sẽ tiếp tục tìm cách khắc phục hạn chế, nhược điểm và nâng cấp, mở rộng thêm bài toán, cụ thể:

* Xây dựng thuật toán nhận diện khuôn mặt trong ảnh
* Tìm cách giảm độ nhầm lẫn, tăng độ chính xác cho model khi dự đoán trong thực tế
* Xây dựng giao diện người dùng cho việc tương tác ứng dụng tốt hơn
* Tìm hiểu thêm về SVM để so sánh sự tối ưu của các thuật toán trong vấn đề xử lý ảnh để nhận diện cảm xúc .

# TÀI LIỆU THAM KHẢO

[1] <http://itechseeker.com/tutorials/tensorflow/xay-dung-mo-hinh-deep-learning-voi-keras/>

[2] <https://nttuan8.com/bai-7-gioi-thieu-keras-va-bai-toan-phan-loai-anh/>

[3] <https://thorpham.github.io/blog/2018/05/25/keras/>

[4] <https://nttuan8.com/bai-7-gioi-thieu-keras-va-bai-toan-phan-loai-anh/>

[5] <https://viblo.asia/p/deep-learning-qua-kho-dung-lo-da-co-keras-LzD5dBqoZjY>

[6] <https://www.youtube.com/watch?v=CS4cs9xVecg&list=PLkDaE6sCZn6Ec-XTbcX1uRg2_u4xOEky0>

[7] <https://www.youtube.com/channel/UCcIXc5mJsHVYTZR1maL5l9w>

[8] <https://www.youtube.com/channel/UCkw4JCwteGrDHIsyIIKo4tQ>