TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI

BÁO CÁO TỔNG KẾT

ĐỀ TÀI NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN
NĂM HỌC 2020-2021

Học máy để nhận biết biểu cảm trên khuôn mặt con người

Sinh viên thực hiện
Lê Quang Duy Lớp: CNTT1-K59 Khoa: Công nghệ thông tin

(in đậm Sinh viên chịu trách nhiệm chính thực hiện đề tài)

Người hướng dẫn: ThS. Phạm Xuân Tích

HÀ NỘI, 2021

TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI

BÁO CÁO TỔNG KẾT

ĐỀ TÀI NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN
NĂM 2020

Học máy để nhận biết biểu cảm trên khuôn mặt con người

Sinh viên thực hiện
Lê Quang Duy Nam, Nữ: Nam Dân tộc: Kinh
Lớp: CNTT1-K59 Khoa: CNTT Năm thứ:3/4
Ngành học: Công nghệ thông tin

(in đậm Sinh viên chịu trách nhiệm chính thực hiện đề tài)

Người hướng dẫn: ThS. Phạm Xuân Tích

HÀ NỘI, 2021

Mục lục

Danh mục bảng biểu

Danh mục những từ viết tắt

CNN – Convolution Newron Network

1. Mở đầu:

Biểu cảm khuôn mặt là yếu tố chính giúp ta nhận biết được cảm xúc của con người, đóng vai trò rất quan trọng trong giao tiếp xã hội, chúng được tạo nên bởi sự chuyển động của các cơ kết nối với da trên khuôn mặt, dựa vào nét mặt, các hình thái biểu cảm của khuôn mặt khác nhau mà biết được một người đang hạnh phúc, giận dữ, hay lo buồn. Nó là một ngôn ngữ chung để nhận biết cảm xúc, ý nghĩa thật sự của mọi người, khó chi phối bởi vùng miền, địa lí, và văn hóa.

Những biểu cảm này thường diễn ra rất nhanh, chỉ từ 1/15 đến 1/25 của một giây, có 7 loại biểu cảm khuôn mặt phổ biến là giận dữ, lo lắng, buồn rầu, ngạc nhiên, hạnh phúc, bình thường và khinh bỉ. Từ trước đến nay, đã có nhiều công trình nghiên cứu về cảm xúc sinh học của con người nhằm xây dựng một phương pháp chung để nhận biết cảm xúc thật sự của con người qua khuôn mặt như phân tích hình ảnh, phân tích video.

Trong đề tài này chỉ phân tích biểu cảm khuôn mặt con người qua hình ảnh, sử dụng mạng noron nhân tạo để phân loại cảm xúc của con người.

2. Tổng quan tình hình nghiên cứu thuộc lĩnh vực công trình/đề tài:

Từ khi học máy trở nên phát triển và dễ tiếp cận nhờ vào khả năng của phần cứng cũng như tập dữ liệu ngày càng lớn và dễ tìm, các đề tài nghiên cứu cũng phát triển không ngừng.

Nhận biết cảm xúc con người được mang vào một cuộc thi lần đầu năm 2013 ở hội nghị international Conference on Machine Learning – ICML trên bộ dữ liệu fer2013 được Ian Goodfellow và các cộng sự xây dựng với mô hình chiến thắng dự đoán tỉ lệ chính xác tới 70.22%. Ngoài ra, còn có nhiều bộ dữ liệu được mang vào các cuộc thi khác nhau như CK, CK+, EULA, v..v

Kết quả chung đều tìm được model đạt đến chất lượng mức con người (human accuracy) trên các tập dữ liệu. Vì đề tài này chọn nghiên cứu trên tập dữ liệu ferplus, được microsoft gán nhãn lại trên tập dữ liệu gốc fer2013 ngoài 7 biểu cảm cơ bản còn có thêm nhãn “conttemp” (khinh thường) nên xin điểm qua một số model tiêu biểu đạt chất lượng state-of-the-art trong tập dữ liệu ferplus.

1. Đào tạo trên kiến trúc mạng LResNet50E-IR, với độ chính xác 89.257%
2. Đào tạo trên kiến trúc VGG16 với độ chính xác đạt được 89.16%

Đặc điểm chung để có được độ chính xác cao là các mạng trên đều là những kiến trúc mạng lớn, với đề tài nghiên cứu tìm hiểu về deeplearning, mạn noron tích chập, tôi mong muốn xây dựng một model từ kiến trúc CNN đơn giản, không đặt mục tiêu đạt độ chính xác quá cao, nhưng model nhẹ, ít tham số và độ chính xác ở mức chấp nhận được.

3. Lý do lựa chọn công trình/đề tài:

Như đã nói ở trên, ngành phân tích biểu cảm khuôn mặt đã phát triển từ rất lâu, trước cả khi mạng noron ra đời vì tính thực dụng, cần thiết của nó ứng dụng được vào nhiều hoàn cảnh trong đời sống hằng ngày từ giao tiếp, tương tác giữa con người với con người, đến các ngành tâm thân học như phát hiện dấu hiệu bất thường của trẻ tự kỷ, trong lĩnh vực vận chuyển như kiểm soát tài xế khi đang vận hành xe, trong lĩnh vực giáo dục, trong lĩnh robot chăm sóc y tế v..v

Với những lợi ích thực tế như vậy, em đã lựa chọn đề tài: “Học máy để nhận biết biểu cảm trên khuôn mặt con người”.

4. Mục tiêu, nội dung, phương pháp nghiên cứu của công trình/đề tài:

4.1 Mục tiêu:

Cùng với những kết quả nghiên cứu đã được công bố trước đây trong kiến trúc mạng noron học sâu, tôi sẽ tập trung khảo sát, đánh giá, và thử nghiệm tìm ra một mô hình mạng noron có độ chính xác ở mức chấp nhận được với chất lượng ở mức con người, cố gắng đơn giản hóa mô hình để dễ dàng ứng dụng vào các vấn đề thực tế.

4.2 Nội dung, phương pháp nghiên cứu

Về mặt nội dung và phương pháp nghiên cứu, đề tài sẽ tìm hiểu và sử dụng mạng noron tích chập CNN (Convolutional Neural Networks), kho dữ liệu được dùng để đào tạo mạng là ferplus được microsoft gán nhãn lại trên tập dữ liệu fer2013.

Sau cùng, với model đạt được, sẽ triển khai thử nghiệm kết quả bằng dự đoán cảm xúc thời gian thực camera điện thoại, cũng như cung cấp API dành cho các ứng dụng thực tế về sau.

5. Đối tượng và phạm vi nghiên cứu

Vì đề tài tập trung nghiên cứu hình ảnh tĩnh thông qua mạng học sâu noron tích chập, nên dữ liệu đầu vào sẽ là hình ảnh, đối tượng ở đây chính là biểu cảm trên khuôn mặt người, với tập dữ liệu ferplus, phạm vi nghiên cứu chỉ dừng lại ở trên tập dữ liệu là 8 cảm xúc của con người bao gồm: tức giận, ghê tởm, sợ hãi, hạnh phúc, buồn rầu, ngạc nhiên, bình thường, khinh thường.

6. Kết quả nghiên cứu và thảo luận:

6.1. Tổng quan về học máy (deep learning):

Deep learning là một tập con của lĩnh vực nghiên cứu về trí tuệ nhân tạo (AI) và thuộc nhóm học có giám sát đã giúp máy tính làm những việc tưởng như là không thể trong thập kỉ trước như phân loại hàng ngàn vật thể trong bức ảnh, tự tạo chú thích cho ảnh, bắt trước giọng nói, hay thậm chí sán tác tác phẩm nghệ thuật. Sử dụng mạng lưới thần kinh với nhiều lớp, qua các lớp, mạng phân tích đặc trưng của dữ liệu và đưa ra dự đoán kết quả, liên tục lặp lại quá trình trên tập đào tạo.

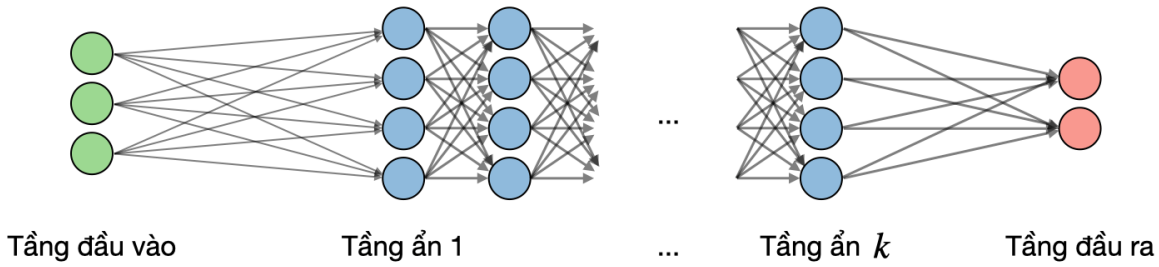


Figure 1Mạng noron

Với mỗi nút trên mạng noron là một hàm số được gọi là hàm kích hoạt (activation function, được tính lần lượt theo công thức dưới đây:

$$z^{(i)} = w^T x^{(i)} + b$$
$$\hat{y}^{(i)} = a^{(i)} = \text{sigmoid}(z^{(i)})$$

Với $x^{(i)}$ là dữ liệu được đưa vào, ma trận w và b được khởi tạo ngẫu nhiên ban đầu và quá trình học của model sẽ là quá trình tìm các tham số của ma trận w và b sao cho với đầu vào là dữ liệu, đầu ra sẽ trả về kết quả đúng như nhãn đã được gắn sẵn $y^{(i)}$.

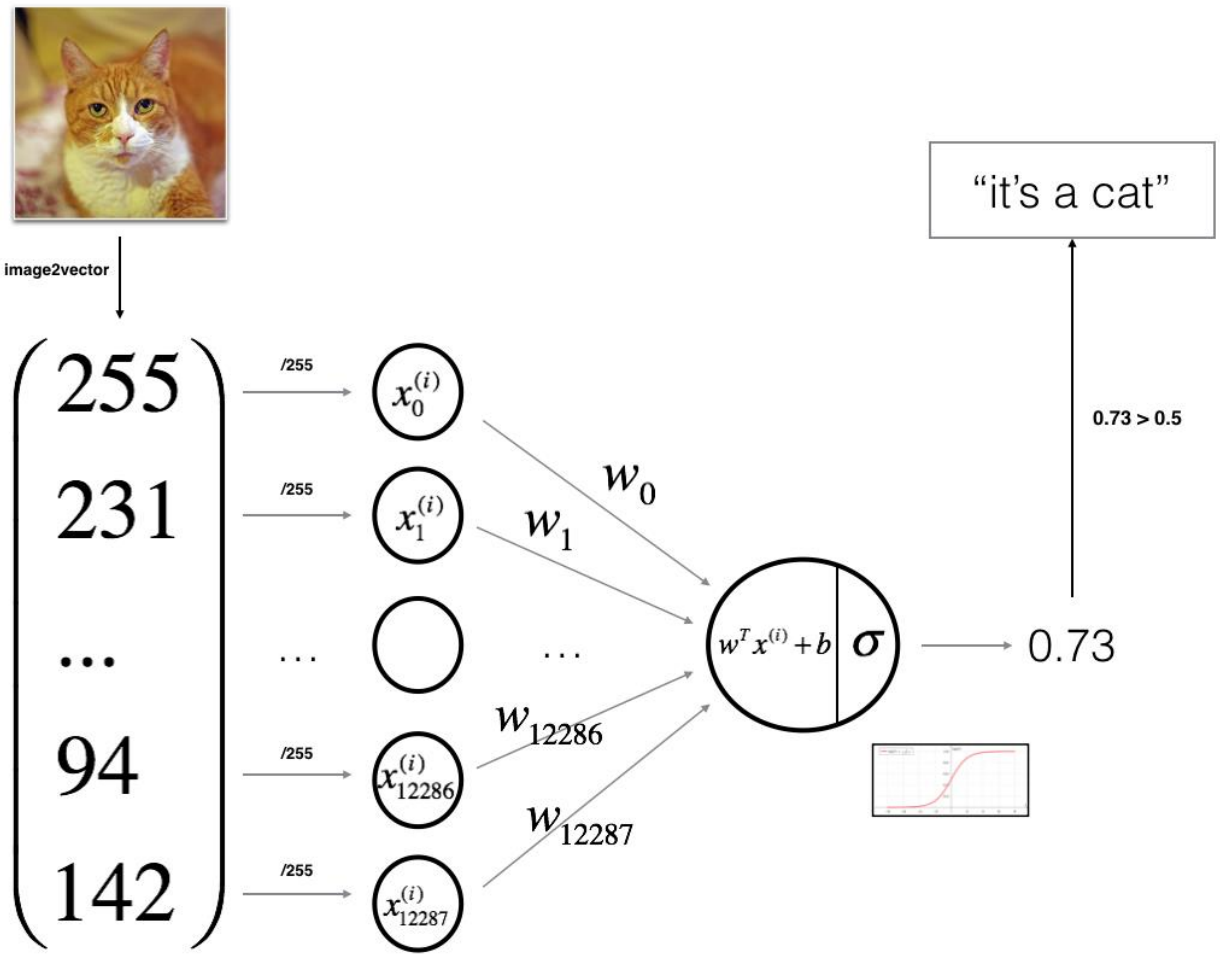


Figure 2 Bên trong mỗi nút noron

- Hàm kích hoạt ở ví dụ trên đây là hàm sigmod, còn có nhiều hàm kích hoạt khác nhau tuỳ thuộc vào kết quả muốn có, xu hướng gần đây hay sử dụng hàm relu làm hàm kích hoạt ở các lớp trung gian cho lĩnh vực nhận diện hình ảnh.

- Để biết được kết quả dự đoán của model và kết quả thật sự khác nhau thế nào, ta định nghĩa hàm lỗi cross-entropy theo một quy tắc nào đó, giả sử được định nghĩa sau đây:

$$L(a^{(i)}, y^{(i)}) = -y^{(i)} \log(a^{(i)}) - (1 - y^{(i)}) \log(1 - a^{(i)})$$

- Hàm L (loss function) trả về một số thực không âm thể hiện độ chênh lệnh giữa giá trị dự đoán và giá trị thực tế, tất nhiên ta luôn muốn tối thiểu hoá hàm lỗi, vì vậy, quá trình đào tạo model là quá trình thay đổi tham số để giảm thiểu giá trị của hàm lỗi.

- Hàm lỗi trên toàn bộ giữa liệu được gọi là hàm giá trị (cost function), công thức được thể hiện như sau:

$$J = \frac{1}{m}\sum_{i=1}^{m} \mathcal{L}\left(a^{(i)}, y^{(i)}\right)$$

- Để có thể cập nhật, thay đổi giá trị của các tham số W và b sao cho hàm lỗi giảm dần, ta dùng thuật toán gradient denscent, qua vô số các vòng lặp, mỗi lần lặp ta sẽ đi ngược lại hướng tăng của hàm lỗi phía trên bằng cách cập nhật lại W và b cộng hay trừ một lượng nào đó theo hệ số gọi là learning rate alpha.

$$W^{[l]} = W^{[l]} - \alpha\, dW^{[l]}$$
$$b^{[l]} = b^{[l]} - \alpha\, db^{[l]}$$

- Quá trình tìm ra được dW và db dựa vào một thuật toán gọi là backpropagation, dịch ra tiếng Việt là lan truyền ngược, ngược lại của bước forwardpropagation, bước tính toán ra giá trị hàm lỗi.

- Quá trình lan truyền ngược được tiến hành dựa trên một quy tắc đạo hàm gọi là chain rule sau đây:

$$\frac{\partial f}{\partial y} = \frac{\partial f}{\partial q}\frac{\partial q}{\partial y}$$

- Hình ảnh dưới đây mô phỏng lại quá tình lan truyền xuôi và lan truyền ngược:

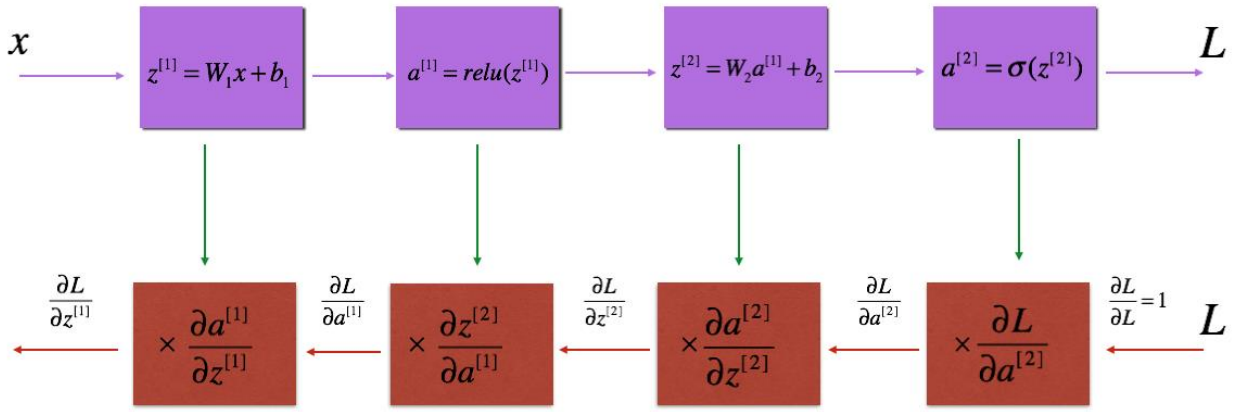


Figure 3 Forwark propagation và Backpropagation

- Như vậy, ta tính được dW và db, từ đó cập nhật được giá trị phù hợp cho W và b:

$$dW^{[l]} = \frac{\partial \mathcal{L}}{\partial W^{[l]}}$$

$$db^{[l]} = \frac{\partial \mathcal{L}}{\partial b^{[l]}}$$

- Quá trình lan truyền ngược sẽ cập nhật lại model từ nút cuối cùng đến nút ban đầu lần lượt, với mỗi nút có thể được biểu thị như sau:

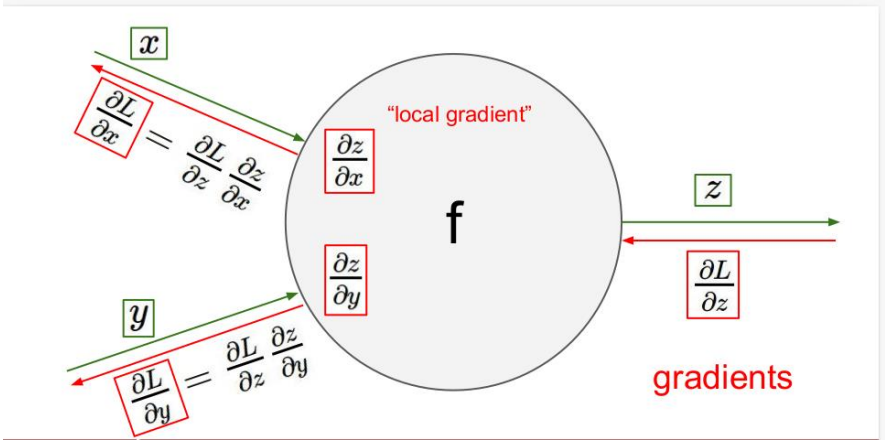


Figure 4 Quá trình lan truyền ngược với từng nút

- Như vậy, trong mạng noron, quá trình đào tạo được diễn ra như sau:

+ Bước 1: Lấy một tập dữ liệu huấn luyện.

+ Bước 2: Thực thi lan truyền xuôi để lấy được giá trị hàm lỗi.

+ Bước 3: Lan truyền ngược lỗi để lấy được gradient độ dốc (chính là đạo hàm).

+ Bước 4: Sử dụng gradients để cập nhật trọng số của mạng.

6.2 Mạng noron tích chập CNN:

- Thực tế với dữ liệu là ảnh, không thể áp dụng được mạng noron truyền thống ở trên vì lượng tham số quá lớn, ảnh hưởng nặng nề tới tốc độ học của chương trình, vì thế, trong lĩnh vực thị giác máy tính, có một mạng được đặc trưng bởi phép tích chập ta có thể giải quyết được vấn đề lượng lớn tham số mà vẫn trích xuất ra được đặc trưng của ảnh gọi là CNN, mạng noron tích chập được cấu thành bởi các tầng sau:

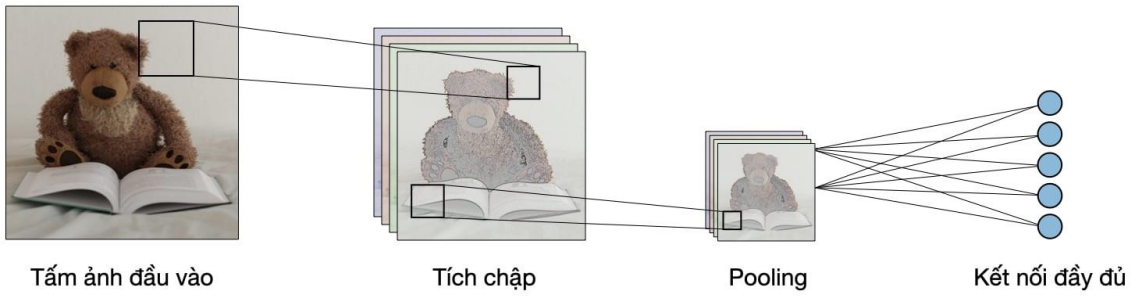


Figure 5 Một mạng CNN đơn giản

- Trước tiên, phép tích chập trên một ảnh là phép trượt một ma trận 2 chiều kenel đặt tại vị trí bên trên trái của ma trận ảnh đã cho rồi thực hiện phép nhân theo từng phần tử trong 2 ma trận, kết quả là lấy tổng của tất cả phép nhân đó, thuật toán được thực hiện cụ thể như dưới đây là phép tích chập trên ảnh X với bộ lọc kenel K:

```
def corr2d(X, K):
    h, w = K.shape
    Y = np.zeros((X.shape[0] - h + 1, X.shape[1] - w + 1))
    for i in range(Y.shape[0]):
        for j in range(Y.shape[1]):
            Y[i, j] = (X[i: i + h, j: j + w] * K).sum()
    return Y
```

- Với ảnh màu có tới 3 kênh RGB, nên khi biểu diễn ma trận ảnh cũng là ma trận 3 chiều, mỗi kenel cũng là 3 chiều kích thước k*k*3.

- Tầng tích chập là đặc trưng của mạng CNN, sử dụng các bộ lọc (filter) có thể học được và phép tích chập để trích xuất đặc trưng của ảnh, được gọi là feature map, ví dụ dưới đây là một bộ lọc trích xuất cạnh dọc của một ảnh.

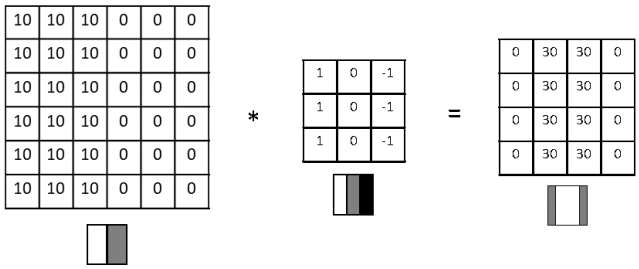


Figure 6 Bộ lọc tích chập trích xuất cạnh dọc của một ảnh

- Như vậy, filter chính là khả năng "nhìn" của mạng, trích xuất các đặc trưng của ảnh tương tự như mắt người, việc học trong mạng CNN là việc đi tìm ra bộ lọc phù hợp với đặc trưng của bài toán.

- Tầng pooling thường được xử dụng sau tầng tích chập, dùng để giảm kích thước mô hình cũng như giảm đi mọt phần lớn tham số trong mạng, do đó cũng kiểm soát tình trạng quá khớp sẽ được nói ở phía dưới đề tài, tầng pooling là tầng không có tham số sinh thêm, có hai dạng phổ biến là max pooling bảo toàn các đặc trưng đã phát hiện, average pooling.

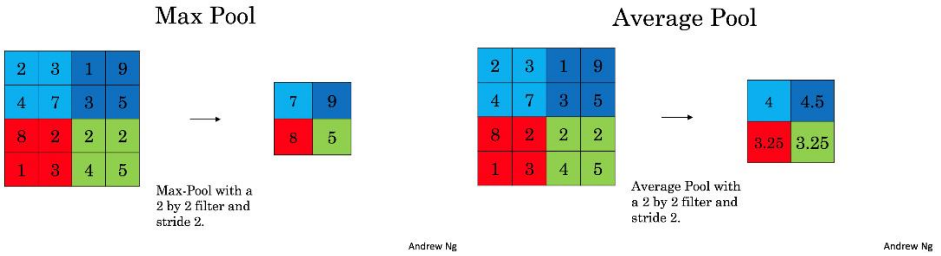


Figure 7 Max pool, Figure 8 Average pool

- Tầng kết nối đầy đủ nhận đầu vào là các dữ liệu đã được làm phẳng (flatten) mà mỗi đầu vào kết nối đến tất cả noron của lớp cuối cùng. Trong mô hình mạng CNN, tầng này thường xuất hiện ở cuối mạng.

- Một số khái niệm khác trong mạng như strike, padding trong đó strike là bước nhảy của bộ lọc trên khối đầu vào activation map trước nó, mặc định trong ví dụ về phép tích chập ở

trên là stride=1, padding có thể hiểu như vùng đệm đầu vào trước đó, nó thường được dùng để điều chỉnh kích cỡ của khối đầu ra.

- Công thức để tính khối đầu ra cho lớp tích chập: W1xH1xD1

+ Gọi K là số filter, tức chiều sâu của khối tích chập

+ F là kích thước của filter, ví dụ 3x3

+ S là bước nhảy stride

+ P là tham số padding

Kích cỡ đầu ra W2xH2xD2 được tính theo công thức:

$$W2 = \frac{W1 - F + 2P}{S} + 1$$

$$H2 = \frac{H1 - F + 2P}{S} + 1$$

$$D2 = K$$

Mỗi noron trong tập W2xH2xD2 kết nối với một vùng có kích thước FxFxK.

- Để hiểu hơn về kiến trúc này, ta có thể xem qua một kiến trúc đơn giản đầy đủ các phần đã kể trên của mạng CNN:

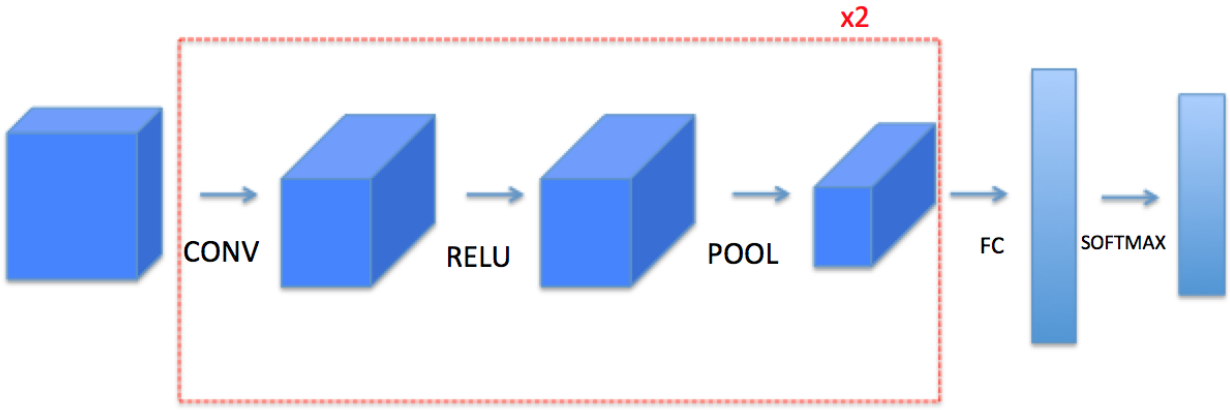


Figure 9 Kiến trúc mạng CNN đơn giản

- Như vậy, các noron trong mạng CNN không kết nối hoàn toàn với các noron trước đó mà chỉ kết nối một phần gọi là trường thụ cảm (receptive field) như trong hình vẽ dưới đây:

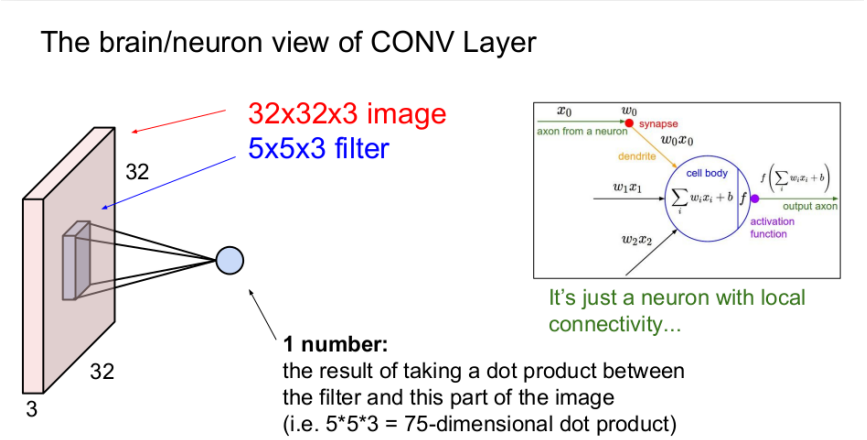


Figure 10 Trường thụ cảm

- Việc tất cả các noron trong cùng một lớp cùng một toạ độ đều cùng chỉ kết nối đến một phần của đầu vào trước đó làm giảm đáng kể lượng tham số mô hình, và đó cũng là một đặc trưng của mạng norron tích chập.

6.3 Tập dữ liệu lựa chọn

Lý do dựa chọn tập dữ liệu fer2013 là bởi số lượng dữ liệu cũng như chất lượng gán nhãn đạt độ chính xác cao của nó đã được kiểm chứng nhiều lần.

Bộ dữ liệu ban đầu có 7 nhãn cơ bản, tuy nhiên, có nhiều ảnh không phải mặt người, cũng như bị gãn nhãn sai, vì vậy, đề tài này sử dụng bộ dữ liệu 2013 nhưng sử dũng nhãn được microsoft gán lại, có thêm 1 biểu cảm khuôn mặt, cũng như loại bỏ ảnh không liên quan tới biểu cảm con người.

Vì microsoft chỉ cung cấp file csv chứa tên file và nhãn mới nên ta cần viết chương trình để chia các file vào các thư mục khác nhau phục vụ cho mục đích học của model sau này:

```
def copy_image(self):
    for od, pd in self.fer_dirs.items():
        src = os.path.join(self.image, pd)
        dst = os.path.join(self.dst, od)
        with open(os.path.join(self.label, pd, 'label.csv')) as f:
            lines = csv.reader(f)
            for line in lines:
                img,_,n,h,s,sad,a,ds,fear,c,_,_=line
                if img:
```

```
arr = [n,h,s,sad,a,ds,fear,c]
m = max(arr)
idx = arr.index(m)
em = ferplus_em[idx]
i = fer2013_em.index(em) if em != 'contempt' else
shutil.copyfile(os.path.join(src, img), os.path.join(dst, str(i), img))
```

Dưới đây là bảng so sánh giữa bộ dữ liệu FER(trên) và FER+(dưới):

Figure 11 Giới thiệu về tập dữ liệu ferplus

Bộ dữ liệu FER+ bao gồm 28273 bức ảnh tập đào tạo (train), 3534 bức ảnh tập xác minh và 3579 bức ảnh thuộc tập kiểm thử độc lập, phân bổ theo đồ thị sau:

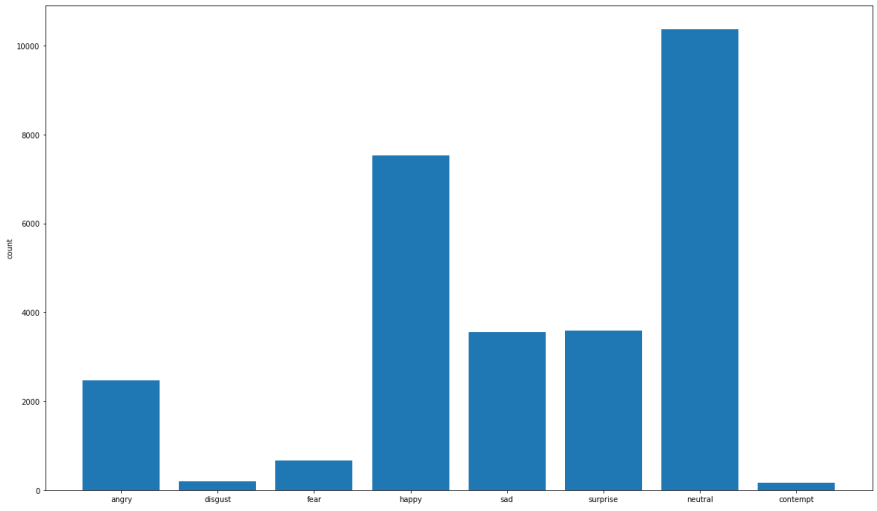


Figure 12 Biểu đồ phân phối tập đào tạo

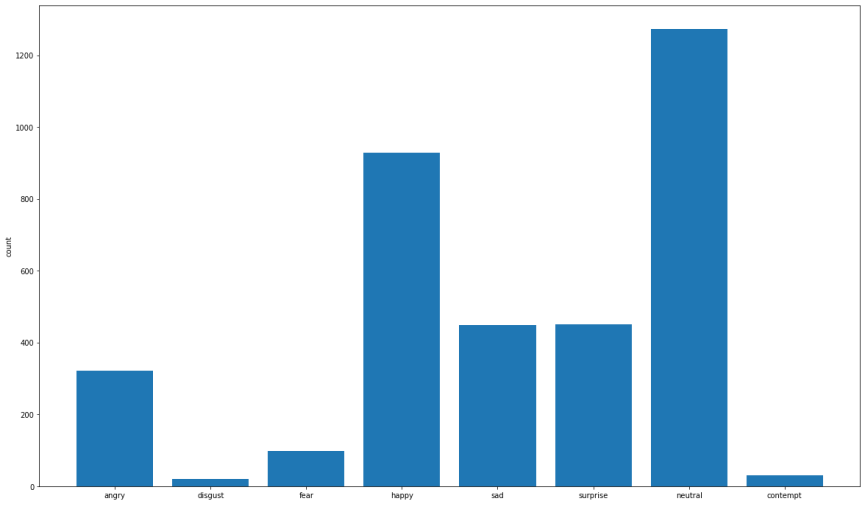


Figure 13 Biểu đồ phân phối tập xác minh

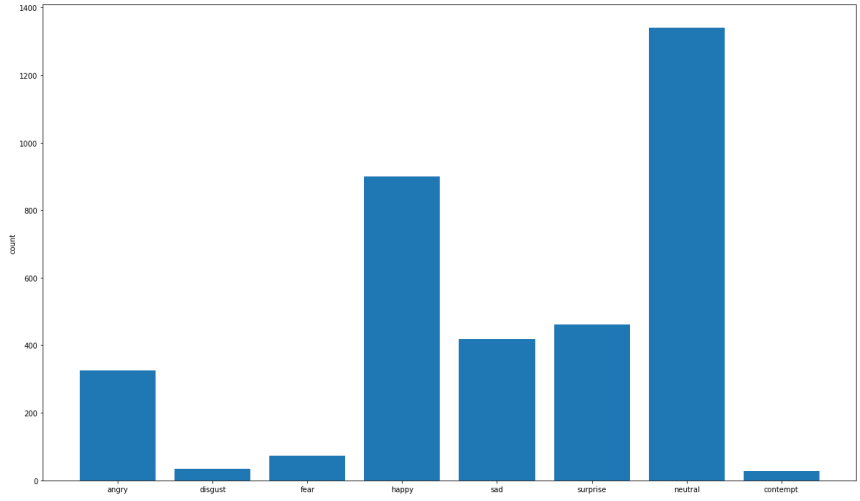


Figure 14 Biểu đồ phân phối tập kiểm tra

6.4 Xây dựng model

1. Hàm softmax

Vì đây là bài toán phân loại, bên cạnh SVM (Support Vector Machine), hàm softmax thường được chọn để đưa ra kết quả dự đoán. Được định nghĩa như sau:

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_j}}$$

2. Mini batch

Khác với Batch gradient descent, Mini-batch là phương pháp dùng 1 lượng dữ liệu vừa đủ trong tập đào tạo để thực hiện bước tính đạo hàm, áp dụng cho những tập dữ liệu lớn.

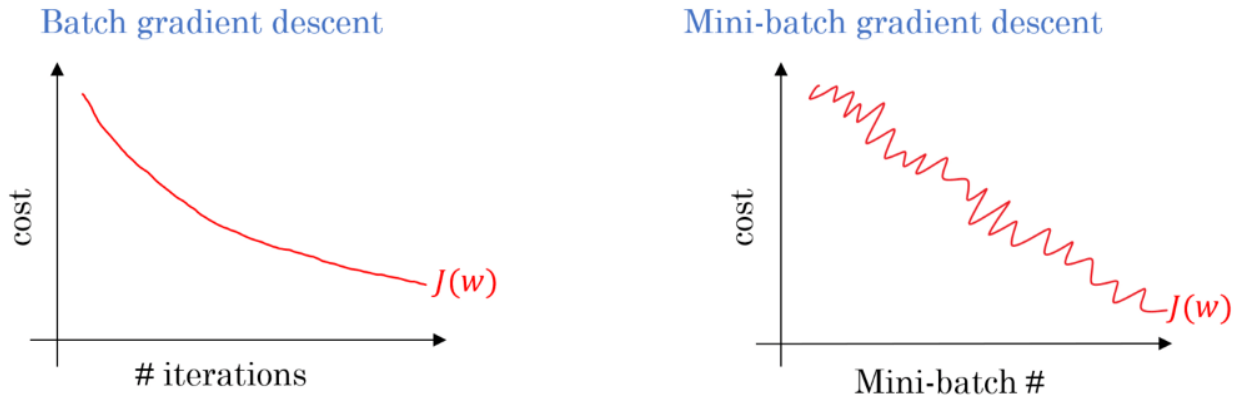


Figure 15 Batch vs Mini-Batch

Áp dụng vào tập dữ liệu FER+, việc tính toán 1 lúc hơn 28000 bức ảnh là rất lâu, vì vậy,trong quá trình đào tạo model chọn BatchSize=16.

3. Tránh hiện tượng overfitting

Hiện tượng overfitting là hiện tượng mà mô hình tìm được quá khớp với dữ liệu đào tạo (training), việc quá khớp này có thể dẫn đến việc dự đoán nhầm, nhiễu, và chất lượng mô hình không còn tốt trên tập kiểm tra nữa, vì vậy, để tránh hiện tượng quá khớp, ta áp dụng một số phương pháp vào mô hình:

3.1 Dropout

Trong quá trình training, khi thực hiện bước lan truyền xuôi, đến layer mà có dropout, nó sẽ tắt ngẫu nhiên một số nút trước đó, sử dụng dropout đã được chứng minh có hiệu quả trong việc tránh tình trạng overfitting với các model hiện đại.

3.2 Batch Normalization

Batch Normalization đưa dữ liệu về phân phối chuẩn Zero Mean (trung bình bằng không và phương sai đơn vị), việc đưa về phân phối chuẩn giúp đặc trưng về độ lớn của dữ liệu sẽ không bị mất đi khi trainning, quá trình trainning sẽ diễn ra nhanh hơn.

Ta có, với mỗi mini-batch (lô nhỏ) có giá trị đầu vào gọi là B = $\{x_{i...m}\}$, tính trung bình và phương sai trên đó:

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i$$

$$\partial_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2$$

Tiến hành chuẩn hoá các x_i :

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\partial_B^2 + \epsilon}}$$

Tham số ϵ rất nhỏ được thêm vào để tránh phép chia 0. Đầu ra sau khi chuẩn hoá là:

$$y_i = \gamma\hat{x}_i + \beta$$

Với gamma và beta là hai tham số cần học trong suốt quá trình training, vì vậy ở bước gradient denscent, cần tính đạo hàm của gamma và beta rồi cập nhật.

3.3 Early stopping, checkpoint

Thực tế cho thấy, khi epochs (số lần lặp lại tất cả dữ liệu của tập đào tạo) quá lớn, độ chính xác với tập kiểm tra bị giảm đi, như vậy, để khắc phục tình đạng ta cần tìm một thồi điểm thích hợp để dừng quá trình đào tạo model lại, để thực hiện điều này, cần chia tập dữ liệu thành 3 tập khác nhau là tập huấn luyện, tập kiểm định và tập kiểm tra như đã nói ở trên, trong quá trình đào tạo model, độ chính xác của mô hình liên tục được đánh giá trên tập kiểm tra khi cảm thấy model không còn khả năng cải thiện được nữa, thì dừng sớm để tránh tăng phương sai.

Với đề tài này, tham số theo dõi trong quá trình training là monitor=val_accuracy, muốn quá trình đào tạo tìm model có độ chính xác trên tập kiểm định đạt cao nhất, vì nó cũng cùng phân phối trên tập kiểm tra.

Model checkpoint thường được sử dụng kèm Early stopping, để cập nhật lại mô hình sau khi hoàn thành 1 epoch, tránh khi quá trình đào tạo bị lỗi không tìm được model.

4. Lựa chọn thuật toán tối ưu

Trong phần nghiên cứu này sẽ điểm qua các thuật toán tối ưu nhằm tìm kiếm thuật toán phù hợp nhất cho model.

4.1. Gradient descent

Gradient descent một thuật toán tối ưu đơn giản, với mỗi lớp của layer thứ l, cập nhật W và b tất cả các nút theo công thức:

$$W^{[l]} = W^{[l]} - \alpha\, dW^{[l]}$$

$$b^{[l]} = b^{[l]} - \alpha\, db^{[l]}$$

Giả sử chúng ta đang cố tối ưu hoá hàm chi phí có mặt cắt theo trục Ox như dưới đây, chấm đỏ là điểm cực tiểu.

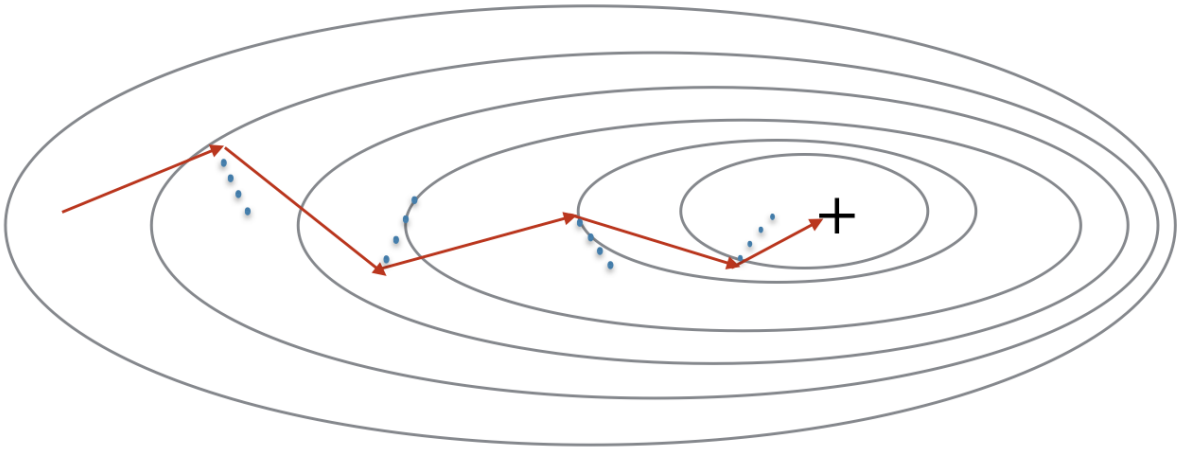


Figure 16 Quá trình tìm kiếm điểm global minimum của gradient descent

Thuật toán gradient descent dao động lên xuống theo trục Oy nhiều khiến thuật toán tối ưu bị chậm đi, thêm nữa, nếu hàm mất mát có nhiều điểm cực tiểu, có thể thuật toán sẽ không tìm được vị trí nơi giá trị hàm đạt nhỏ nhất.

4.2 Gradient descent momentum

Để khắc phục nhược điểm trên của gradient descent, một thuật toán khác áp dụng nguyên lí: trung bình trọng số theo cấp số nhân (Exponentially Weighted Averages) giúp quá trình tìm kiếm điểm global minimum diễn ra nhanh hơn gọi là gradient descent momentum.

Với l là số lớp layer của mạng noron, quá trình cập nhật trọng số sẽ diễn ra như sau:

$$v_{dW^{[l]}} = \beta v_{dW^{[l]}} + (1 - \beta) dW^{[l]}$$

$$W^{[l]} = W^{[l]} - \alpha v_{dW^{[l]}}$$

$$v_{db^{[l]}} = \beta v_{db^{[l]}} + (1 - \beta) db^{[l]}$$

$$b^{[l]} = b^{[l]} - \alpha v_{db^{[l]}}$$

Với hình vẽ về quá trình tiến tới điểm global minimum ở trên, ta sẽ giảm thiểu sự dao động của trục Oy và tăng nhanh sự giao động ở trục Ox, bằng cách áp dụng trung bình trọng số cấp số nhân cho hai giá trị dW và db, thay thế gradient bằng trung bình của các gradient trong quá khứ với số gradient hiệu dụng là $\frac{1}{1-\beta}$ giúp thuật toán hội tụ nhanh hơn.

4.3. Thuật toán RMSprop

Gần tương tự như momentum ở trên, thuật toán RMSprop cũng hoạt động dựa trên trung bình trọng số theo cấp số nhân:

$$s_{dW} = \beta s_{dW} + (1 - \beta) dW^2$$

$$s_{db} = \beta s_{db} + (1 - \beta) db^2$$

$$W = W - \propto \frac{dW}{\sqrt{s_{dw} + \epsilon}}$$

$$b = b - \propto \frac{db}{\sqrt{s_{db} + \in}}$$

Với ∈ là hằng số vô cùng nhỏ thêm vào để tránh trường hợp chia 0. Khi dW^2, db^2 lớn thì $\sqrt{s_{dw} + \in}$, $\sqrt{s_{dw} + \in}$ lớn hơn nhiều, vì vậy sẽ giảm thiểu dao động, tương tự, khi dW^2, db^2 nhỏ thì $\sqrt{s_{dw} + \in}$, $\sqrt{s_{dw} + \in}$, trong thực tế, dW và dB là 2 tham số có nhiều chiều và khi nó tăng ta sẽ giảm sự dao động của nó, từ đó có thể tăng tham số alpha khiến quá trình tìm ra điểm global minimum sẽ diễn ra nhanh hơn.

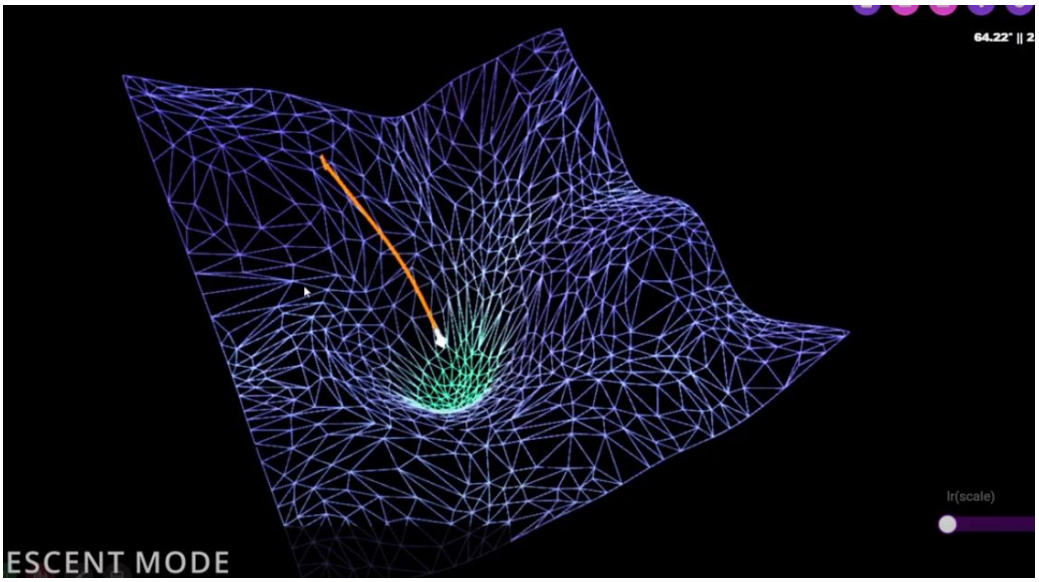


Figure 17 Quá trình đi tìm điểm local minimum của thuật toán tối ưu

4.4 Adam

Thuật toán Adam là sự kết hợp giữa các thuật toán đã đề cập trước, mang tính hiệu quả cao đã được thử nghiệm trên nhiều mạng khác nhau.

Bước đầu tiên, thuật toán sử dụng tính v và s dựa trên trung bình trọng số cấp số nhân theo gradient.

$$v_{dW^{[l]}} = \beta_1 v_{dW^{[l]}} + (1 - \beta_1)\frac{\partial \mathcal{J}}{\partial W^{[l]}}$$

$$s_{dW^{[l]}} = \beta_2 s_{dW^{[l]}} + (1 - \beta_2)\left(\frac{\partial \mathcal{J}}{\partial W^{[l]}}\right)^2$$

Tham số β_1đặc trưng cho thuật toán gradient descent momentum, vì vậy, thường chọn $\beta_1 = 0.9$, β_2đặc trưng cho thuật toán RMSprop, thường chọn $\beta_2 = 0.999$. Điều này làm cho phương sai di chuyển chậm hơn nhiều so với động lượng, vì vậy, ta cần chuẩn hoá lại thành 2 biến dưới đây:

$$v_{dW^{[l]}}^{corrected} = \frac{v_{dW^{[l]}}}{1 - (\beta_1)^t}$$

$$s_{dW^{[l]}}^{corrected} = \frac{s_{dW^{[l]}}}{1 - (\beta_2)^t}$$

Như vậy, tương tự RMSpops, cập nhật trọng số W:

$$W^{[l]} = W^{[l]} - \alpha \frac{v_{dW^{[l]}}^{corrected}}{\sqrt{s_{dW^{[l]}}^{corrected}} + \varepsilon}$$

Kết luận, nếu momentum được ví như một viên bi nặng lăn theo đà thì RMS lại là một viên bi rất nặng có ma sát, kết hợp lại thành Adam, nó dễ dàng vượt qua các điểm local minimum tiến tới global minimum, vì vậy, trong model nhận diện cảm xúc con người cũng lựa chọn sử dụng Adam làm thuật toán tối ưu.

5. Quá trình xây dựng model:

Với các nghiên cứu lý thuyết cần thiết để xây dựng model ở trên, việc thiết kế model tối ưu là một quá trình lặp lại liên tục tuần hoàn từ lên ý tưởng, đào tạo mạng, kiểm tra đến sửa sai.

Qua quá trình xem những nhãn mà model dự đoán sai có đặc điểm chung thế nào, áp dụng cách tính độ chệch và phương sai để liên tục mở rộng mô hình, tăng độ chính xác khi có thể, nhưng cũng giảm thiểu hiện tượng overfitting, underfiting, vanishing gradient. Sau cùng, đây là model tối ưu nhất mà đề tài nghiên cứu này tìm được:

5.1 Tiển xử lý, làm giàu dữ liệu:

Ta sẽ sử dụng kỹ thuật data augmentation để làm giàu tập dữ liệu hiện có bằng các phép xoay ảnh, dịch trái, dịch phải, phóng to ảnh, đảo ngược ảnh trên tập train, còn dữ nguyên trên tập validation và tập test.

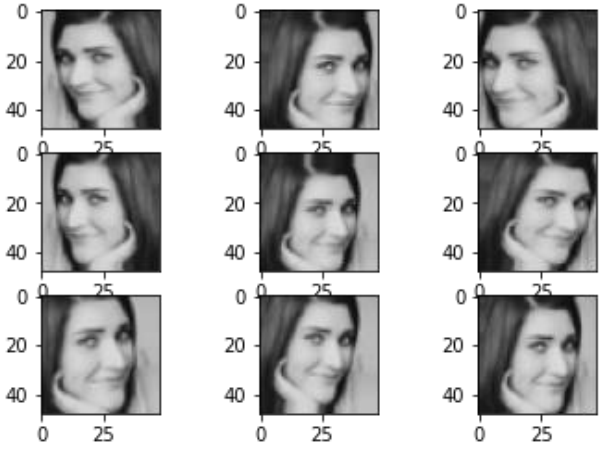


Figure 18 Làm giàu dữ liệu

5.2 Kiến trúc model

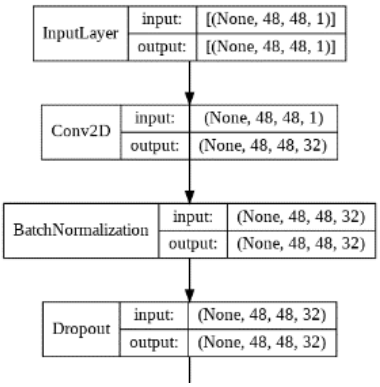


Figure 19 Khối thứ nhất

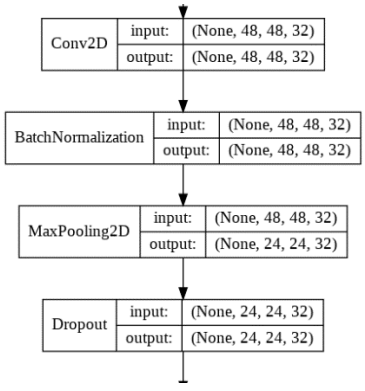


Figure 20 5 khối tiếp theo gần tương tự, chỉ khác số lớp filter tăng dần 32,64,64,128,256

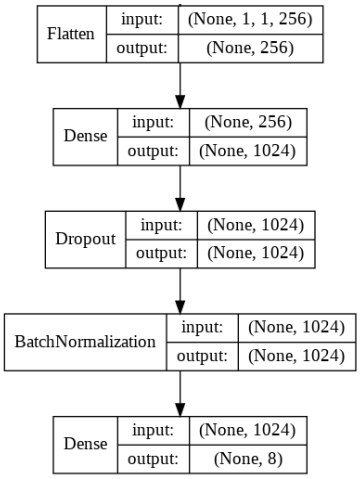


Figure 21 Khối cuối cùng

Số lượng tham số: 711.784, trong đó số lượng tham số đào tạo là 708.584.

5.3 Các siêu tham số khác

+ Kích cỡ ảnh đầu vào : 48x48x1
+ Tốc độ học: 0.001
+ Momentum: 0.9
+ Batch Size:16
+ Số lượng epoches tối đa: 60/100
+ Patience: 10

5.4 Đào tạo mạng

Quá trình đào tạo mạng được hiện thực bằng ngôn ngữ Python, framework tensorflow, keras.

Python là ngôn ngữ bậc cao, dễ dùng, thuận tiện, tensorflow là một framework nổi tiếng về huấn luyện model, keras là một API bậc cao chạy trên nền tensorflow, model chủ yếu được huấn luyện thông qua gọi API bậc cao từ keras.

Phát hiện vị trí khuôn mặt, sử dụng OpenCV2.

5.4 Đánh giá kết quả trên tập dữ liệu:

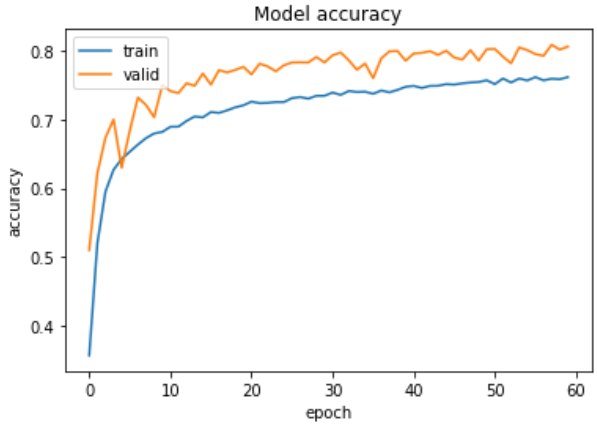


Figure 22 Biểu đồ quá trình trainning

Với độ chính xác đạt 81.11 trên tập xác minh, dưới đây là ma trận bối rối trên tập xác minh:

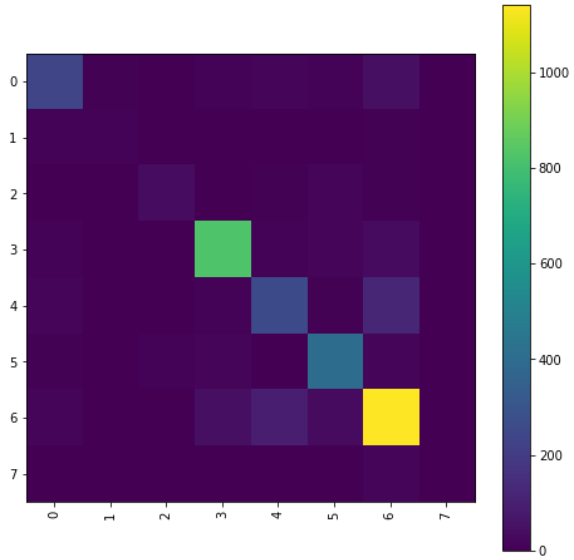
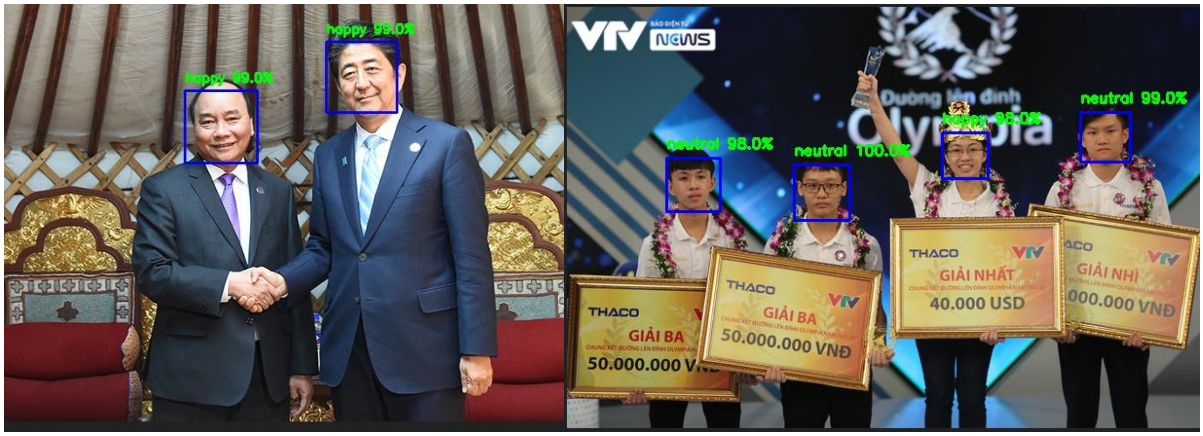


Figure 23 Ma trận bối rối của mạng tự làm

Trên tập kiểm định, độ chính xác đạt 80%. Như vậy, không có sự khác biệt quá lớn giữa độ chính xác các tập đào tạo, tập kiểm định và tập xác minh, model hoàn toàn có thể mở rộng hơn nữa.

6. Một số kết quả dự đoán trong thực tế:



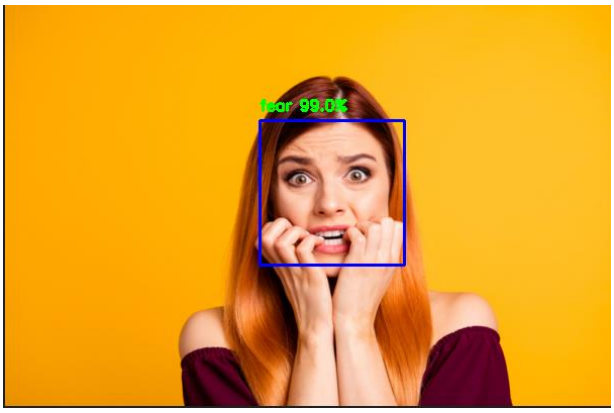


Figure 24 Một số hình ảnh kiểm định trong thực tế

7. Ứng dụng thực tế, tạo API, web cho phép nhận diện cảm xúc con người realtime qua camera trước:

Ngôn ngữ backend: python, thư viện frontend: reactjs.
Ứng dụng thư viện flask trong python để viết API đơn giản gửi ảnh lên server, ảnh trả về sẽ là ảnh dự đoán.

Đầu tiên chúng ta cần dùng thư viện opencv để định vị tất cả khuôn mặt có trong bức ảnh rồi chạy model dự đoán trên đó. Cụ thể như sau:

```
def face_detector(img):
    # Convert image to grayscale
    gray = cv2.cvtColor(img.copy(), cv2.COLOR_BGR2GRAY)
    faces = face_classifier.detectMultiScale(gray, 1.2, 4)
    if faces is ():
        return (0, 0, 0, 0), np.zeros((48, 48), np.uint8), img

    allfaces = []
    print(len(faces))
    rects = []
    for (x, y, w, h) in faces:
        cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 2)
        roi_gray = gray[y:y + h, x:x + w]
        roi_gray = cv2.resize(roi_gray, (48, 48), interpolation=cv2.INTER_AREA)
        allfaces.append(roi_gray)
        rects.append((x, w, y, h))
    return rects, allfaces, img


@app.route('/uploader', methods=['GET', 'POST'])
def upload_file():
    if request.method == 'POST':
        f = request.files['file']
        f.save(f.filename)

        img = cv2.imread("./" + f.filename)
        rects, faces, image = face_detector(img)

        i = 0
        for face in faces:
            roi = face.astype("float") / 255.0
            roi = img_to_array(roi)
            roi = np.expand_dims(roi, axis=0)

            preds = classifier.predict(roi)[0]

            label = class_labels[preds.argmax()] + " " +str(round(preds[preds.argmax()]*100))+"%"

            label_position = (rects[i][0], abs(rects[i][2] - 10))

            i+= 1
            cv2.putText(image,label,label_position, cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0),2)

            cv2.imwrite(f.filename,image)

        return send_file(f.filename, mimetype='image/gif')
```

Tiến hành chạy thử API trên máy local:

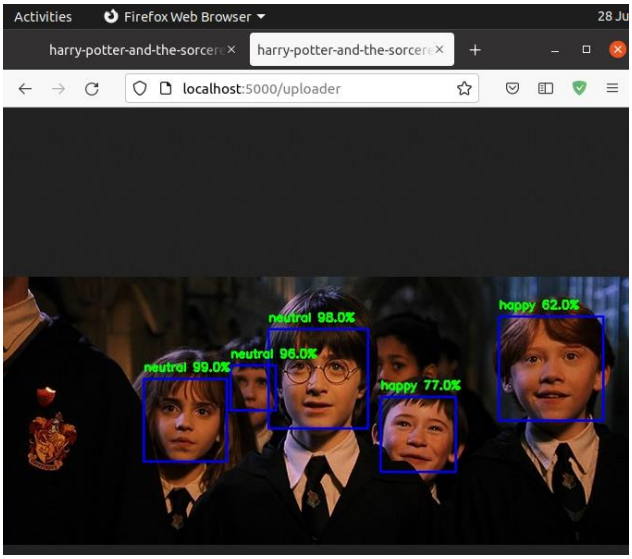


Figure 25 API

Do khó khăn trong việc tìm kiếm máy chủ để chạy realtime camera, nên web realtime sẽ được viết trên tensorflowjs, ta chỉ cần convert model về dạng json là xử lý được, thuật toán cũng tương tự như ở trên, nên xin phép sẽ không trình bày lại.

Kết quả cuối cùng: model nhận diện nhanh chóng, đạt độ chính xác cao.

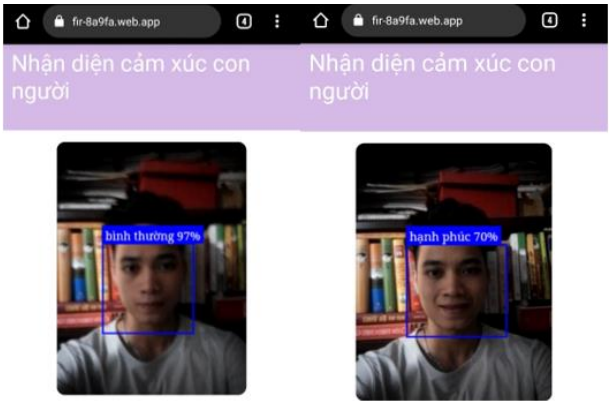


Figure 26 Thử model trên web

8. Kết luận và kiến nghị:

a. Phần kết luận:

Qua quá trình tìm hiểu về mạng noron tích chập, sau đó áp dụng vào để xây dựng một model đơn giản, gọn nhẹ, ít tham số nhưng chất lượng vẫn ở mức cao, và phát triển API cũng như ứng dụng web, em hi vọng với công cụ này sẽ áp dụng được vào vấn đề thực tiễn, ứng dụng hoá công nghệ này vào đời sống.

b. Phần kiến nghị:

Từ kết quả nghiên cứu, có thể dễ dàng nhận thấy muốn cải thiện chất lượng nhận diện này không chỉ dừng lại ở việc phát triển model mà còn là phát triển dữ liệu, khắc phục sự mất cân đối trong tập dataset đã dùng ở trên. Mở rộng hơn có thể cá nhân hoá trên tập dữ liệu người Việt để đưa ra những dự đoán phù hợp với đặc trưng, tính cách con người Việt Nam.

12. Tài liệu tham khảo:

[1] Andrew Ng. "Course Deep Learning Specialization"

[2] Stanford University. "CS230, CS231N"

[3] Book Dive into Deep Learning.

[4] Vũ Hữu Tiệp, blog machinelearningcoban

[5] Tuấn Nguyễn, blog nttuan.com

[6] Isha Talegaonkar, Kalyani Joshi, Shreya Valunj, Rucha Kohok, Anagha Kulkarni. "Real Time Facial Expression Recognition using Deep Learning"

[7] Amil Khanzada, Charles Bai, Ferhat Turker Celepcikay, "Facial Expression Recognition with Deep Learning"

[8] Akash Saravanan. "Facial Emotion Recognition using Convolutional Neural Networks"