**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**KHOA CÔNG NGHỆ THÔNG TIN**

****

**BÁO CÁO MÔN HỌC**

**CÁC HỆ THỐNG ĐA PHƯƠNG TIỆN**

**ĐỀ TÀI: NHẬN DIỆN CẢM XÚC**

**KHUÔN MẶT CON NGƯỜI**

**GVHD : TS Huỳnh Hữu Hưng**

**SVTH : Nguyễn Định An**

**Vương Lữ Trân Châu**

**Lê Trọng Hiếu**

**Đà Nẵng, 3/2019**

**Ý KIẾN ĐÁNH GIÁ CỦA GIẢNG VIÊN**

# MỤC LỤC

[DANH SÁCH HÌNH ẢNH ii](#_Toc6991651)

[DANH SÁCH BẢNG iv](#_Toc6991652)

[MỞ ĐẦU v](#_Toc6991653)

[GIỚI THIỆU ĐỀ TÀI 1](#_Toc6991654)

[Phân chia công việc 3](#_Toc6991655)

[Chương 1: Hướng giải quyết và cơ sở lý thuyết, mô hình. 3](#_Toc6991656)

[1.1 Bài toán và hướng giải quyết 3](#_Toc6991657)

[1.2 Lý thuyết về mạng neural tích chập 4](#_Toc6991658)

[1.2.1 Neural nhân tạo 4](#_Toc6991659)

[1.2.2 Mạng neural kinh điển 5](#_Toc6991660)

[1.2.3 Mạng neural tích chập 6](#_Toc6991661)

[1.3 Thư viện OpenCV 10](#_Toc6991662)

[1.3.1 Tổng quan về OpenCV 10](#_Toc6991663)

[1.3.2 Ứng dụng nhận diện khuôn mặt bằng OpenCV 10](#_Toc6991664)

[Chương 2: TRIỂN KHAI THỰC HIỆN 14](#_Toc6991665)

[2.1 Các bước triển khai thực hiện 14](#_Toc6991666)

[2.1.1 Xác định các loại cảm xúc khuôn mặt 14](#_Toc6991667)

[2.1.2 Thu thập bộ dữ liệu hình ảnh khuôn mặt 16](#_Toc6991668)

[2.1.3 Xây dựng mô hình mạng neural tích chập và huấn luyện mô hình này bằng bộ dữ liệu đã thu thập. 17](#_Toc6991669)

[2.1.4 Xây dựng chương trình chính 21](#_Toc6991670)

[Chương 3: KẾT QUẢ VÀ ĐÁNH GIÁ 22](#_Toc6991671)

[3.1 Kết quả training mạng neural tích chập 22](#_Toc6991672)

[3.2 Kết quả chương trình 23](#_Toc6991673)

[3.3 Đánh giá 27](#_Toc6991674)

[Tài liệu tham khảo 28](#_Toc6991675)

# DANH SÁCH HÌNH ẢNH

Hình 1.1 Sơ đồ tổng quát các bước xử lý bài toán 3

Hình 1.2 Cấu trúc một neural 4

Hình 1.3 Sự liên kết của các neural 5

Hình 1.4 Cấu trúc layer trong mạng neural 5

Hình 1.5 Ví dụ về mạng neural kinh điển gồm 3 layer 6

Hình 1.6 Mô hình chung mạng neural tích chập 6

Hình 1.7 Một bức ảnh thông qua nhiều góc nhìn khác nhau (các feature map) 8

Hình 1.8 Minh họa max pooling 9

Hình 1.9 Minh họa average pooling 9

Hình 1.10 Drop-out thực hiện loại bỏ một số neural trong quá trình tính toán 9

Hình 1.11 Các đặc trưng Haar cơ bản 11

Hình 1.12 Cách tính toán hình ảnh tích phân 12

Hình 1.13 Cách tính tổng vùng pixel nhanh 12

Hình 1.14 Các đặc trưng có ích 13

Hình 2.1 Sơ đồ tổng quá các bước triển khai thực hiện 14

Hình 2.2 Một số hình ảnh và cảm xúc mẫu 15

Hình 2.3 Dữ liệu hình ảnh khuôn mặt dưới dạng file .csv 16

Hình 2.4 Giao diện Anaconda Navigator 19

Hình 2.5 Chạy lệnh huấn luyện trên terminal (windows cmd) 20

Hình 2.6 Huấn luyện trên Google Collaborator 20

Hình 2.7 Các file lưu bộ phân lớp đã huấn luyện 21

Hình 3.1 Kết quả train thực tế 22

Hình 3.2 Biểu đồ thể hiện sự thay đổi độ chính xác qua từng epoch 23

Hình 3.3 Bản mẫu giao diện chương trình ban đầu 23

Hình 3.4 Nhận diện cảm xúc giận dữ 24

Hình 3.5 Nhận diện cảm xúc ngạc nhiên 24

Hình 3.6 Nhận diện cảm xúc nhiều khuôn mặt (điều kiện sáng) 25

Hình 3.7 Nhận diện cảm xúc nhiều khuôn mặt (điều kiện sáng) 25

Hình 3.8 Nhận diện cảm xúc nhiều khuôn mặt (điều kiện tối) 26

Hình 3.9 Trường hợp nghiêng mặt sang một bên 26

# DANH SÁCH BẢNG

[Bảng 2‑1 Bảng mô tả mô hình Mạng neural tích chập 19](#_Toc7647612)

# MỞ ĐẦU

Công nghệ thông tin nói chung và trí tuệ nhân tạo nói riêng đã có những sự tiến bộ vượt bậc. Chúng đã giúp chúng ta giải quyết nhiều vấn đề một cách nhanh chóng và hiệu quả mà trước đây chúng ta không thể xử lý được.

Đề tài Nhận diện các cảm xúc khuôn mặt của môn học Đa phương tiện hướng tới việc nhận diện được cảm xúc khuôn mặt, với dữ liệu đầu vào là các camera quay khuôn mặt. Đây là một đề tài thiết thực, hỗ trợ cho việc nhận diện được khuôn mặt, đồng thời cũng giúp sinh viên áp dụng những kiến thức đã học được vào thực hành thực tế.

Qua đề tài này, chúng em không những củng cố được kiến thức về mạng neutron và xử lý hình ảnh mà còn được tiếp xúc với nhiều công nghệ mới trong ngành từ đó áp dụng tạo ra một sản phẩm cơ bản.

Chúng em xin chân thành cảm ơn TS. Huỳnh Hữu Hưng đã hướng dẫn và giúp đỡ chúng em trong quá trình thực hiện đề tài.

Trong quá trình thực hiện đề tài, chúng em không thể tránh khỏi những sơ suất, kính mong thầy thông cảm cho chúng em.

Chúng em xin chân thành cảm ơn.

# GIỚI THIỆU ĐỀ TÀI

Trong vài năm gần đây, cùng với sự phát triển về khoa học và công nghệ, đặc biệt là trong lĩnh vực trí tuệ nhân tạo, máy móc đã dần dần có được các khả năng nhận biết như con người. Một trong những khả năng đó là khả năng nhận biết các vật thể xung quanh thông qua hình ảnh, hay còn gọi là thị giác máy tính. Để tìm hiểu rõ hơn về thị giác máy tính, trong môn học Các hệ thống đa phương tiện lần này, nhóm quyết định chọn một đề tài áp dụng Xử lý ảnh và Học máy để thực hiện. Đề tài được lựa chọn là Nhận diện cảm xúc khuôn mặt con người.

Hơn 2 thập kỉ qua đã có rất nhiều nghiên cứu về đề tài này bằng nhiều hướng tiếp cận khác nhau. Có thể kể đến như hướng tiếp cận sử dụng Mạng Bayesian của nhóm tác giả Yoshihiro Miyakoshi, và Shohei Kato [1], hoặc hướng tiếp tiếp cận sử dụng Support Vector Machine và Deep Boltmann Machine của nhóm tác giả Ma Xiaoxi, Lin Weisi [2]. Ngoài ra còn có Pyramid Histogram of Gradients (PHOG) [3], AU aware facial features [4], boosted LBP descriptors [5], hay RNNs [6]. Tuy nhiên gần đây, các bài dự thi [7], [8] đứng đầu cuộc thi 2015 Emotions in the Wild – xác định cảm xúc khuôn mặt trên ảnh tĩnh - đều sử dụng Mạng neural tích chập (Convolutional neural network), đạt độ chính xác lên đến 62%.

Một nghiên cứu gần đây của G. Levi và cộng sự [9] cho thấy sự cải thiện đáng kể trong việc sử dụng CNN để nhận diện cảm xúc khuôn mặt. Nhóm tác giả đã giải quyết 2 vấn đề nổi bật: 1) huấn luyện mạng CNN bằng một lượng dữ liệu không lớn và 2) ngoại hình thay đổi do sự chiếu sáng thay đổi. Họ đã sử dụng Local Binary Pattern (LBP) để biến bức ảnh thành nhiều bức ảnh khác dựa trên sự thay đổi chiếu sáng. Bước tiền xử lý dữ liệu này đã được áp dụng cho nhiều mô hình khác như VGG\_S. Kết quả cuối cùng cho thấy độ chính xác kiểm tra lên tới 54,56%.

Một cài đặt thực tế trong việc áp dụng CNN cho việc xác định cảm xúc khuôn mặt thời gian thực được thực hiện bởi S. Oullet [10]. Tác giả đã tạo ra một trò chơi, ở đó, CNN được áp dụng để trực tiếp nhận dạng cảm xúc khuôn mặt người chơi từ camera rồi điều khiển trò chơi dựa trên cảm xúc.

Hiện nay Mạng neural tích chập cũng là một hướng tiếp cận được áp dụng rất nhiều trong thực tế cho nhiều bài toán xử lý ảnh khác, đồng thời cũng được hỗ trợ ở mức cao với rất nhiều thư viện lập trình xây dựng sẵn, nên nhóm quyết định chọn nghiên cứu thực hiện đề tài theo hướng tiếp cận này. Nhóm sử dụng một nguồn tài liệu khá tốt từ đại học Stanford [11] để tìm hiểu về CNN và thực hiện báo cáo này.

Bố cục bài báo cáo được trình bày như sau:

Chương 1 sẽ trình bày về hướng giải quyết, cơ sở lý thuyết, mô hình. Với nội dung chính là sơ đồ tổng quát các bước xử lý bài toán đặt ra; cơ sở lý thuyết về mạng Neural tích chập và Thư viện Thị giác máy tính OpenCV cũng như thuật toán sử dụng để nhận dạng khung hình chứa khuôn mặt trong một khung hình lớn hơn.

Chương 2 sẽ trình bày chi tiết từng bước triển khai thực hiện theo mô hình ở chương 1.

Chương 3 là phần kết quả và đánh giá thực hiện chương trình.

## Phân chia công việc

Một số công việc chính được phân chia như sau:

- Nguyễn Định An:

+ Thực hiện các bước tiền xử lý với bộ dữ liệu hình ảnh.

+ Xây dựng môi trường training trên máy cá nhân và Google Collaboration.

+ Xử lý video vào để phù hợp với mục đích riêng, ...

- Lê Trọng Hiếu:

+ Tìm kiếm bộ dữ liệu.

+ Xây dựng mô hình mạng neural tích chập

+ Huấn luyện mạng neural tích chập và lưu lại model để tái sử dụng.

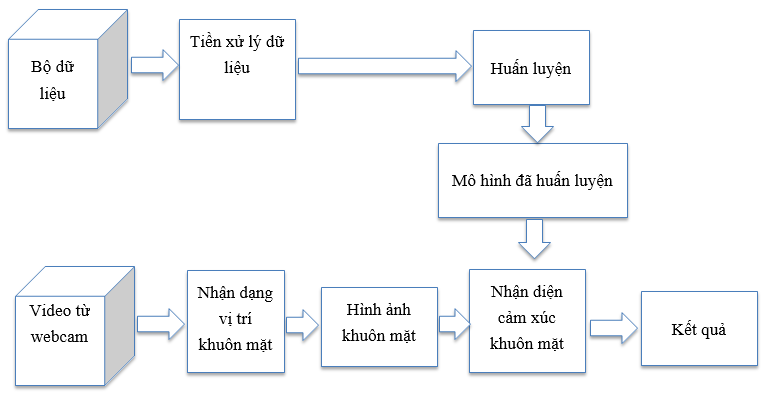
- Vương Lữ Trân Châu:

+ Tìm hiểu thuật toán.

+ Viết báo cáo.

# Hướng giải quyết và cơ sở lý thuyết, mô hình.

## 1.1 Bài toán và hướng giải quyết



Hình . Sơ đồ tổng quát các bước xử lý bài toán

Cụ thể bài toán đặt ra là, xây dựng một chương trình nhận diện cảm xúc khuôn mặt trực tiếp trên video thu bằng webcam. Với bài toán như vậy, các bước giải quyết nhóm đặt ra là:

* Xác định các loại cảm xúc trên khuôn mặt.
* Thu thập bộ dữ liệu hình ảnh khuôn mặt.
* Xây dựng mô hình **mạng neural tích chập** và huấn luyện mô hình này bằng bộ dữ liệu đã thu thập.
* Xây dựng chương trình xác định được vị trí khuôn mặt người trong từng khung hình thu được bởi webcam. Vị trí khuôn mặt sẽ được xác định bằng các đánh dấu một vùng hình chữ nhật chứa khuôn mặt trên khung hình.
* Đưa dữ liệu vị trí khuôn mặt đó vào mạng neural đã huấn luyện để xác định cảm xúc khuôn mặt trong hình.

Theo các bước đặt ra như trên, đầu tiên là vấn đề về việc xác định các loại cảm xúc trên khuôn mặt. Theo một nghiên cứu thì có chủ yếu 7 loại cảm xúc khuôn mặt (0=Tức giận, 1=Chán nản, 2=Lo sợ, 3=Vui vẻ, 4=Buồn, 5=Ngạc nhiên, 6=Bình thường). Theo đó có thể xác định được cần phải xây dựng một mạng neural có khả năng phân lớp được 7 lớp cảm xúc như trên.

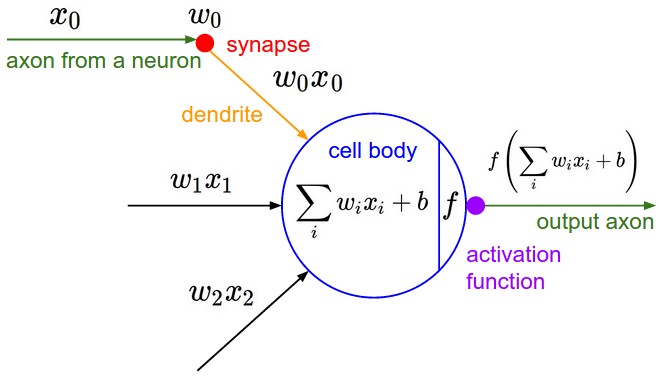
Về việc thu thập bộ dữ liệu, hiện tại có sẵn khá nhiều bộ dữ liệu khác nhau, nhóm quyết định sẽ tìm bộ dữ liệu phù hợp sau khi tìm hiểu về mạng neural tích chập. Ngoài ra, để xử lý các vấn đề liên quan đến việc thu hình từ webcam máy tính, xác định khuôn mặt,... nhóm quyết định sử dụng thư viện Thị giác máy tính được sử dụng khá rộng rãi hiện nay, đó là OpenCV.

Phần tiếp theo trình bày cơ sở lý thuyết của Mạng neural tích chập được sử dụng để xây dựng **Mô hình huấn luyện** và Thư viện Thị giác máy tính OpenCV cùng với thuật toán HaarCascade được sử dụng ở bước **Nhận dạng vị trí khuôn mặt**.

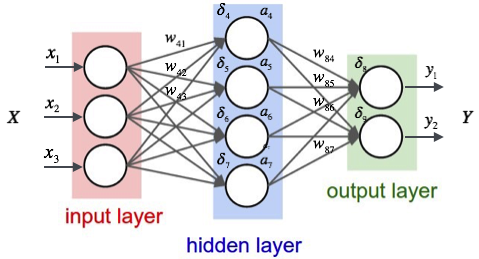
## 1.2 Lý thuyết về mạng neural tích chập

### 1.2.1 Neural nhân tạo

Mạng neural được xây dựng từ một hay nhiều thành phần cơ bản gọi là neural (artificial neural). Chúng nhận một hay nhiều **input**, thay đổi trạng thái nội tại của nó (gồm một bộ các số gọi là **weight** và **bias** kèm một hàm gọi là **activation function**) dựa theo input tương ứng, trả về **output**. Mạng neural hình thành bằng sự liên kết của các neural, khi đó output của neural này là input của neural khác.



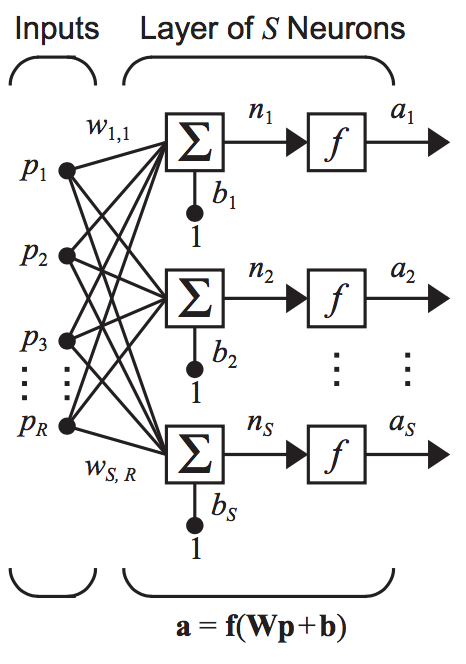
Hình . Cấu trúc một neural



Hình . Sự liên kết của các neural

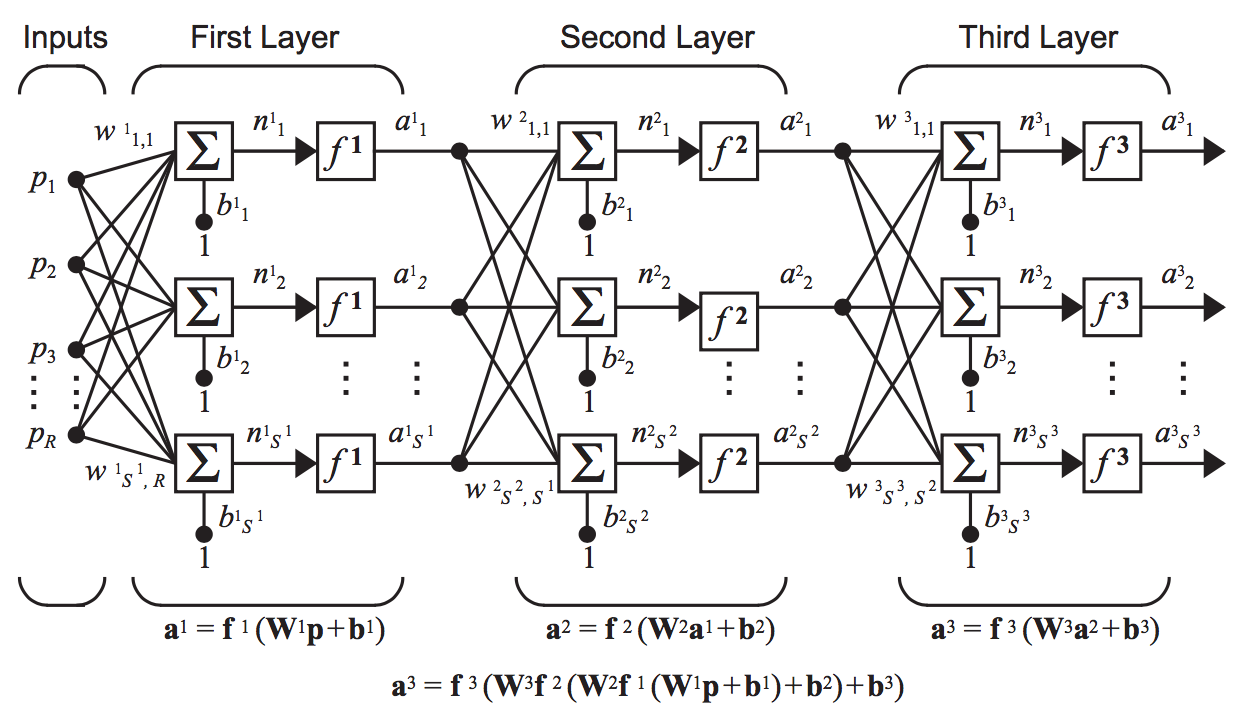
### 1.2.2 Mạng neural kinh điển

Thành phần cơ bản của mạng neural kinh điển là các lớp (layer). Mỗi layer gồm S multiple-input neural, thường có cùng activation function và cùng nhận input **p** gồm **R** thành phần.



Hình . Cấu trúc layer trong mạng neural

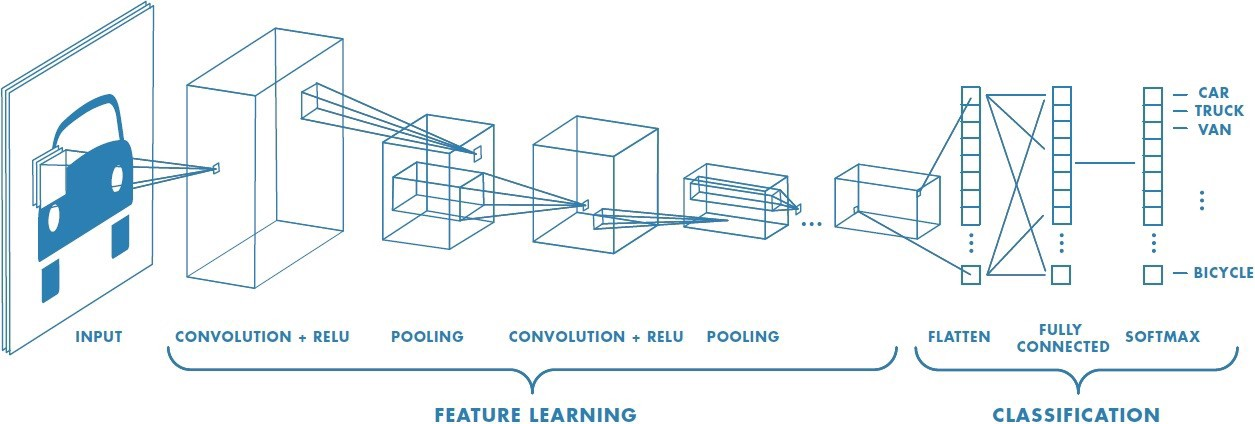
Mạng neural cơ bản bao gồm nhiều layer xếp liên tiếp nhau. Mỗi layer có ma trận weight, vector bias của riêng nó. Input của layer này là output của layer kế trước nó, trừ layer đầu tiên nhận input của mạng neural làm input của nó.



Hình . Ví dụ về mạng neural kinh điển gồm 3 layer

Trên đây là cấu trúc kinh điển và đơn giản nhất của mạng neural với các neural được liên kết với toàn bộ các neural trước nó, do đó còn có tên gọi là fully- connected Neural Network và các layer của nó là fully-connected layer hay FC layer.

### 1.2.3 Mạng neural tích chập



Hình . Mô hình chung mạng neural tích chập

Mạng neural tích chập (Convolutional Neural Network - CNN) hiện nay là một trong những phương pháp thông dụng nhất trong nhận diện hình ảnh (Image recognition), liên quan mật thiết với thị giác máy tính (Computer Vision) khởi nguồn từ những năm 1980 khi Kunihiko Fukushima đề xuất một cấu trúc mạng neural mới lấy cảm hứng từ cơ quan thị giác của mèo và khỉ.

Những thuật toán ở thời điểm đó có thể nhận diện một số cấu trúc của hình ảnh tuy nhiên khi chúng được biến đổi như bị dịch chuyển ra cạnh của ảnh, bị lật ngược hoặc bị che khuất thì độ chính xác giảm đi nhiều. Cấu trúc được đề xuất bởi Fukushima có thể tổng quát hóa hình ảnh cũng như loại bỏ những vật thể thừa, tránh được tình trạng over-fitting của mạng neural truyền thống khi tính toán rất tốt những ảnh mà nó đã học nhưng thể hiện rất tệ đối với những ảnh chúng chưa từng gặp bằng cách giảm thiểu số lượng tham số của mạng neural.

Năm 1998, LeCun và cộng sự xây dựng mạng neural chồng chập LeNet-5 nhằm nhận dạng chữ số viết tay với kích thước ảnh 32x32. Tuy nhiên vì sự hạn chế của phần cứng máy tính lúc đó nên không thể phát triển cho ảnh có độ phân giải cao hơn.

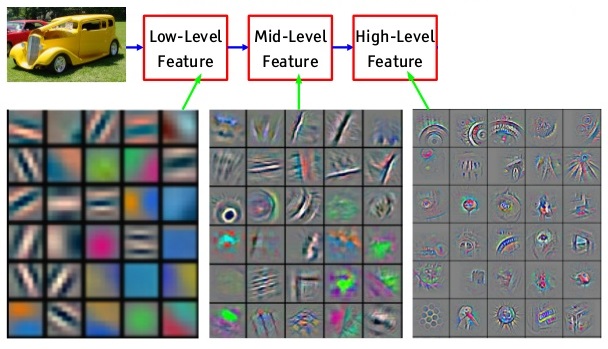
Đến năm 2005, tại Hội nghị quốc tế về Phân tích và Nhận dạng tài liệu ICDAR, Dave Steinkraus và cộng sự đề xuất giải pháp sử dụng GPU cho các giải thuật học máy và đã được áp dụng thành công trên các cơ sở dữ liệu ảnh lớn như MNIST, ImageNet vào năm 2012.

Các layer trong mạng neural tích chập:

* Convolutional layer

Convolutional layer (gọi tắt là lớp conv) chứa các “dây thần kinh” của mạng neural được gọi là feature map. Các feature map này có nhiệm vụ tính toán các dữ liệu input (đối với lớp conv đầu tiên thì là giá trị của sample trong dataset, đổi với các lớp conv khác là output của các hidden layer liền trước nó) thông qua phép tích chập.

Feature map thể hiện khả năng nhận biết các đặc trưng của input. Các feature map tựa như các góc độ quan sát của con người khi nhìn vào một bức ảnh, càng quan sát dưới nhiều góc độ, khả năng phát hiện các đặc điểm nhận diện càng cao. Đây cũng chính là điểm vượt trội so với mạng neural kinh điển chỉ xét bức ảnh hoặc dữ liệu dưới một góc độ duy nhất.



Hình . Một bức ảnh thông qua nhiều góc nhìn khác nhau (các feature map)

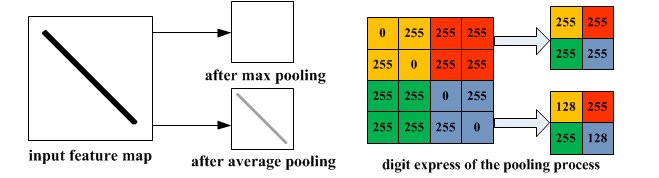
Quá trình train mạng neural tích chập thực chất là quá trình update các tham số của các feature map (các weight và bias).

* ReLU layer

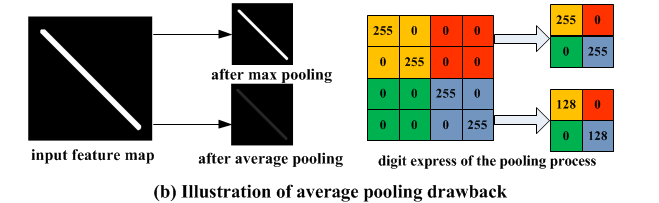
ReLU layer thực chất là một activation function định nghĩa output của convolutional layer liền trước nó. Activation function có nhiệm vụ chuẩn hóa ouput này vào một khoảng nhất định chẳng hạn như [-1, 1] hoặc [0, 1], …Đối với ReLU, output sẽ được chuẩn hóa thành số không âm và nằm trong khoảng [0, x] với x là giá trị của output. ReLU mang ý nghĩa quan trọng đối với khả năng học của mạng neural tích chập.

* Pooling layer

Mục đích của pooling rất đơn giản là làm giảm số parameter mà ta cần phải tính toán, từ đó giảm thời gian tính toán, tránh overfitting. Loại pooling ta thường gặp nhất là *max pooling*, lấy giá trị lớn nhất trong một *pooling window*. Pooling hoạt động gần giống với convolution, nó cũng có 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 gía trị nằm trong cửa sổ trượt (với max pooling ta sẽ lấy giá trị lớn nhất). Ngoài max pooling còn có average pooling, thay vì lấy giá trị lớn nhất ta sẽ tính giá trị trung bình cộng trong cửa số trượt *pooling window.*



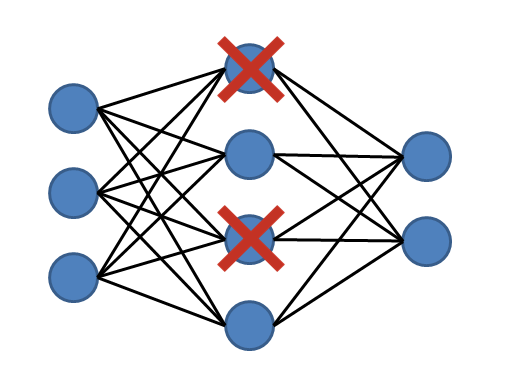
Hình . Minh họa max pooling



Hình . Minh họa average pooling

* Drop-out layer

Drop-out là một kĩ thuật Regularization để chống lại vấn đề overfitting. Dropout layer là cách thức chúng ta giả định một phần dữ liệu bị ẩn đi theo tỉ lệ định sẵn trong quá trình training. Drop-out giúp cho mạng neural network có khả năng nhận diện mang tính tổng quát và khách quan hơn, tránh tình trạng học vẹt và quá phụ thuộc vào bộ train dẫn đến việc sai lệch tính toán khi áp dụng vào thực tế.



Hình . Drop-out thực hiện loại bỏ một số neural trong quá trình tính toán

* Softmax layer

Softmax là hàm kích hoạt ở ouput layer trong mạng neural tích chập được sử dụng với bài toán phân loại nhị phân với nhiều classes (multi-class classification problems) ở lớp ra. Với các bài toán classification, output layer thường là một softmax layer giúp tính xác suất để một điểm dữ liệu rơi vào mỗi class.

Ở bài toán phân loại cảm xúc khuôn mặt, có tất cả 7 cảm xúc ta xem như 7 class trong mạng neural tích chập. Mục tiêu chính của softmax layer là thể hiện được phân phối xác suất trên 7 class, từ đó ta chọn class có xác suất cao nhất tức là kết quả mong muốn của mạng neural tích chập.

## 1.3 Thư viện OpenCV

### 1.3.1 Tổng quan về OpenCV

OpenCV (Open Source Computer Vision Library) được sử dụng rộng rãi trong Thị giác máy tính. Đây là thư viện mở, được sử dụng miễn phí cho mục đích nghiên cứu học thuật và cả thương mại. OpenCV có giao diện lập trình hỗ trợ ngôn ngữ C++, Python, Java và hỗ trợ hệ điều hành Windows, Linux, MacOS, iOS và Android. OpenCV được thiết kế để có hiệu quả tính toán cao và hướng đến các ứng dụng thời gian thực. OpenCV được viết và tối ưu bằng ngôn ngữ C++.

Các ứng dụng chính của OpenCV là:

* Xử lý hình ảnh 2D, 3D.
* Nhận diện khuôn mặt.
* Nhận diện cử chỉ.
* Nhận diện tương tác người – máy.
* Nhận diện vật thể.
* Theo dõi cử chỉ.

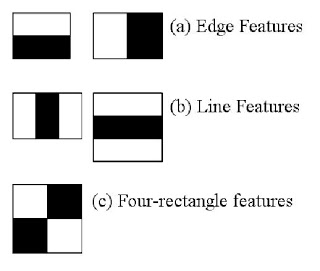
### 1.3.2 Ứng dụng nhận diện khuôn mặt bằng OpenCV

Nhận diện vật thể sử dụng bộ phân lớp Haar cascade là một phương pháp nhận diện vật thể hiệu quả được đề xuất bởi Paul Viola và Michael Jones trong một bài báo [12] của họ vào năm 2001. Thuật toán sử dụng phương pháp tiếp cận dựa trên học máy, ở đó, một hàm cascade được huẩn luyện từ rất nhiều hình ảnh. Sau đó các hàm này được sử dụng để nhận diện vật thể trong các hình ảnh khác.

Ở đây chúng ta chỉ nói đến nhận diện khuôn mặt. Thuật toán gồm 4 bước chính:

* Lựa chọn các đặc trưng Haar.
* Tạo các ảnh tích phân (Integral Image).
* Huấn luyện bằng Adaboost.
* Xếp tầng các bộ phân lớp (Cascading Classifiers).

Bước đầu tiên là tìm các đặc trưng Haar. Các đặc trưng Haar là sự kết hợp của 2, 3 hình chữ nhật “trắng”, “đen” như hình bên dưới.

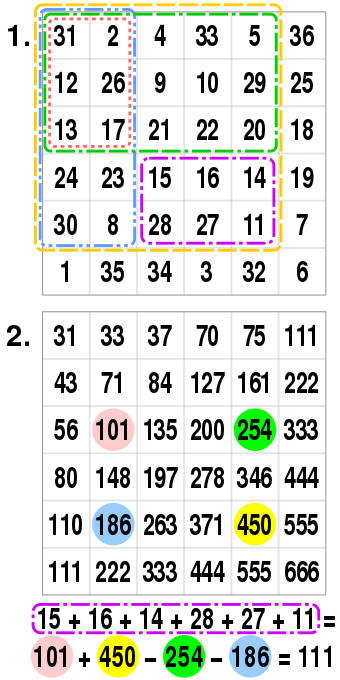


Hình . Các đặc trưng Haar cơ bản

Hình trên nêu lên 3 loại đặc trưng Haar cơ bản. Sử dụng các đặc trưng trên làm 1 nhân đặc trưng (kernel) như trong lý thuyết mạng neural tích chập, ta tính giá trị của đặc trưng là sự chênh lệch giữa tổng của các pixel ở các vùng đen và trắng như sau:

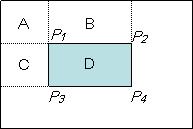
Ta cần tính toán các đặc trưng bằng cách dịch chuyển nhân đặc trưng này “trượt” trên bức ảnh. Tưởng tượng chỉ với một bức ảnh 24x24 đã tạo ra hơn hơn 160000 đặc trưng. Vì vậy cần có phương pháp tăng tốc việc tính toán này.

Bước thứ 2 là tạo các ảnh tích phân. Mục đích của bước này là để tăng tốc việc tính toán đặc trưng Haar. Việc tạo ảnh tích phân được tính như hình dưới.



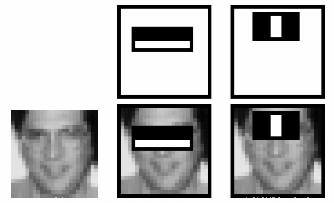
Hình . Cách tính toán hình ảnh tích phân

Từ ảnh tích phân này ta có thể tính toán tổng của một vùng pixel hình chữ nhật bất kì chỉ với độ phức tạp O(1).



Hình . Cách tính tổng vùng pixel nhanh

Ví dụ như để tính tổng giá trị của vùng chữ nhật D, ta chỉ cần lấy giá trị . Tuy nhiên trong hầu hết các đặc trưng tính toán ra đều không có nhiều giá trị. Ví dụ như hình dưới, chỉ có những đặc trưng đầu tiên là có giá trị xác định vị trí mắt, vì thường khu vực mắt tối màu hơn những vùng khác trên khuôn mặt. Nhưng ví dụ cũng lấy chính nhân đặc trưng đó tính toán ở vùng má hoặc miệng thì không cho ra kết quả khả quan gì.



Hình . Các đặc trưng có ích

Vậy chúng ta cần có phương pháp để lựa chọn các đặc trưng tốt nhất trong số hơn 160000 đặc trưng có thể tính toán ra bằng các sử dụng thuật toán Adaboost. Sau đó sử dụng phương pháp xếp tầng các bộ phân lớp (cascade classifiers) để tạo nên một bộ phân lớp mạnh cuối cùng.

Chúng ta có thể tự huấn luyện lại các bộ phân lớp Haar này. Tuy nhiên thư viện OpenCV có chứa khá nhiều bộ phân lớp đã được huấn luyện sẵn có khả năng xác định khuôn mặt, mắt, mũi, miệng,... nên trong đề tài lần này, nhóm sẽ sử dụng bộ phân lớp đã được huấn luyện sẵn trong OpenCV.

# TRIỂN KHAI THỰC HIỆN

## 2.1 Các bước triển khai thực hiện

### 

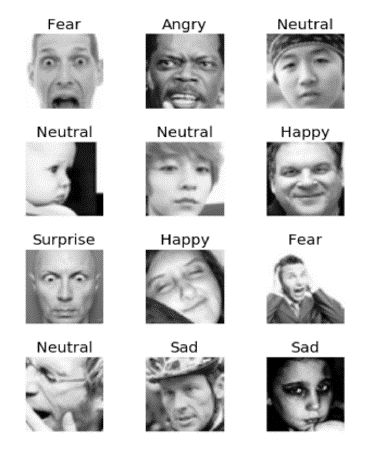
Hình . Sơ đồ tổng quá các bước triển khai thực hiện

### Kí hiệu (2.x.x) ở sơ đồ trên là phần trình bày cho bước thực hiện đó.

### Sơ đồ tổng quát các bước triển khai thực hiện trên tương đối giống với sơ đồ tổng quát của phương pháp Học có giám sát (Supervised learning). Tuy nhiên nhóm đã bỏ qua bước Tách đặc trưng (Feature Extraction) mà đưa trực tiếp dữ liệu hình ảnh vào mô hình CNN để huấn luyện.

### 2.1.1 Xác định các loại cảm xúc khuôn mặt và bộ dữ liệu hình ảnh khuôn mặt

Theo một nghiên cứu thì có chủ yếu 7 loại cảm xúc khuôn mặt (0=Tức giận, 1=Chán nản, 2=Lo sợ, 3=Vui vẻ, 4=Buồn, 5=Ngạc nhiên, 6=Bình thường).



Hình . Một số hình ảnh và cảm xúc mẫu

Vì thế tổng cộng có 7 phân lớp cảm xúc cho mỗi hình ảnh khuôn mặt.

Trên trang Kaggle đã có sẵn bộ dữ liệu cảm xúc khuôn mặt phù hợp với 7 cảm xúc nêu trên. Bộ dữ liệu gồm hơn 30000 bức ảnh khuôn mặt với độ phân giải 48x48 được lưu dưới định dạng file csv.



Hình . Dữ liệu hình ảnh khuôn mặt dưới dạng file .csv

Mỗi hàng dữ liệu bao gồm 3 cột:

* “emotion” với giá trị từ 0-6 tương ứng với từng loại cảm xúc.
* “pixels” bao gồm các giá trị pixel tương ứng với từng điểm ảnh của bức ảnh.
* “Usage” để đánh dấu phần dữ liệu sử dụng để huấn luyện hay để test.

Ngoài ra, nhằm mục đích tăng lượng dữ liệu có sẵn, ở bước này, nhóm còn sử dụng cách làm đó là xoay hoặc phóng to, thu nhỏ dữ liệu có sẵn, tạo ra các dữ liệu hình ảnh mới bằng cách sử dụng lớp ImageDataGenerator có sẵn trong thư viện Keras.

### 2.1.2 Tiền xử lý dữ liệu

Bước tiền xử lý dữ liệu chỉ bao gồm việc chuẩn hóa dữ liệu cụ thể là phương pháp feature scaling. Như đã trình bày, bộ dữ liệu có sẵn mang giá trị mỗi điểm ảnh từ 0-255. Chuẩn hóa dữ liệu đưa miền giá trị của dữ liệu trở về vùng -1 đến 1 bằng cách lấy giá trị gốc chia 255, trừ 0.5 rồi nhân 2. Công thức tổng quát như sau:

Bởi vì giá trị của dữ liệu thường phân phối trên một khoảng rất lớn nên trong một số thuật toán học máy, hàm mục tiêu (loss function) có thể hoạt động không đúng cách. Nếu như một trong các features mang một dải giá trị quá rộng, sự chính xác của feature sẽ bị ảnh hưởng xấu bởi khoảng cách giữa các giá trị điểm ảnh. Một lý do khác chúng ta áp dụng phương pháp feature scaling này chính là *gradient descent* sẽ hội tụ nhanh hơn, đồng nghĩa với việc tốc độ học tăng lên nhanh hơn.

### 2.1.3 Xây dựng mô hình mạng neural tích chập và huấn luyện mô hình này bằng bộ dữ liệu đã thu thập.

Mạng neural tích chập được xây dựng dựa vào thư viện keras. Ta thực hiện xây dựng các khuôn mẫu mạng neural tích chập trong file *myCNN.py.*

Vì mạng neural tích chập là một chuỗi các lớp tuần tự liên tiếp nhau nên đầu tiền thực hiện khai báo:

|  |
| --- |
| model = Sequential() |

Sau đó ta lần lượt thêm các layer Convolution2D, theo sau lần lượt là các layer Activation tương ứng hoặc Pooling2D tùy thuộc vào mô hình mạng neural tích chập mà chúng ta xây dựng. Ta cũng có thể thêm vào các layer Dropout với tỉ lể drop các neural tương ứng hoặc layer BatchNormalization nhằm mục đích tối ưu hóa các thông số trong mạng. Mã giả thể hiện một phần của mạng neural tích chập được biểu diễn như sau:

|  |
| --- |
| model.add(Convolution2D(filters=16, kernel\_size=(7, 7), padding='same',  name='image\_array', input\_shape=input\_shape))  model.add(BatchNormalization())  model.add(Convolution2D(filters=16, kernel\_size=(7, 7), padding='same'))  model.add(BatchNormalization())  model.add(Activation('relu'))  model.add(AveragePooling2D(pool\_size=(2, 2), padding='same'))  model.add(Dropout(.5)) |

Ở đây chúng ta khai báo layer đầu tiền của mạng neural tích chập là Convolution2D với input\_shape là kích thước của ảnh đầu vào 48x48, 16 features (filters), mỗi features có kích thước 7x7, ta cũng có thể tích tổng các tham số có trong layer đơn giản như sau:

*Số params = (Số features \* Kích thước features) + Số bias*

Như vậy ở layer đầu tiên:

*Số params = 16 \* 7 \* 7 + 16 = 800 (tham số)*

Theo sau là layer BatchNormalization nhằm chuẩn hóa các tham số đầu ra ở lớp Convolution2D theo các batch nhằm ổn định các tham số của một bức ảnh đối với các bức ảnh khác. Theo sau nữa là layer Activation với hàm ReLU chuẩn hóa các output vào trong một khoảng xác định.

Tiếp nối là layer AveragePooling2D giảm kích thước output xuống còn một nửa. Sau khi đi qua AveragePooling2D kích thước của output ban đầu 48x48x16 sẽ giảm xuống còn 24x24x16.

Cuối cùng là layer Dropout 0.5 đưa một nửa ngẫu nhiên các output về 0 tránh quá trình học vẹt trên một bức ảnh. Layer này giúp mạng neural đưa ra các kết quả chính xác hơn khi gặp phải những bức ảnh nằm ngoài tập train, hay nói cách khác là kết quả do mạng neural tích chập đưa ra một cách khách quan hơn.

Ở cuối mạng neural tích chập chúng ta khai báo layer như sau:

|  |
| --- |
| model.add(Activation('softmax', name='predictions'))  return model |

Layer softmax tính toán xác suất để output của mạng neural tích chập rơi vào từng class. Kết quả của layer softmax này là 7 xác suất tương ứng với xác suất xuất hiện các cảm xúc khuôn mặt.

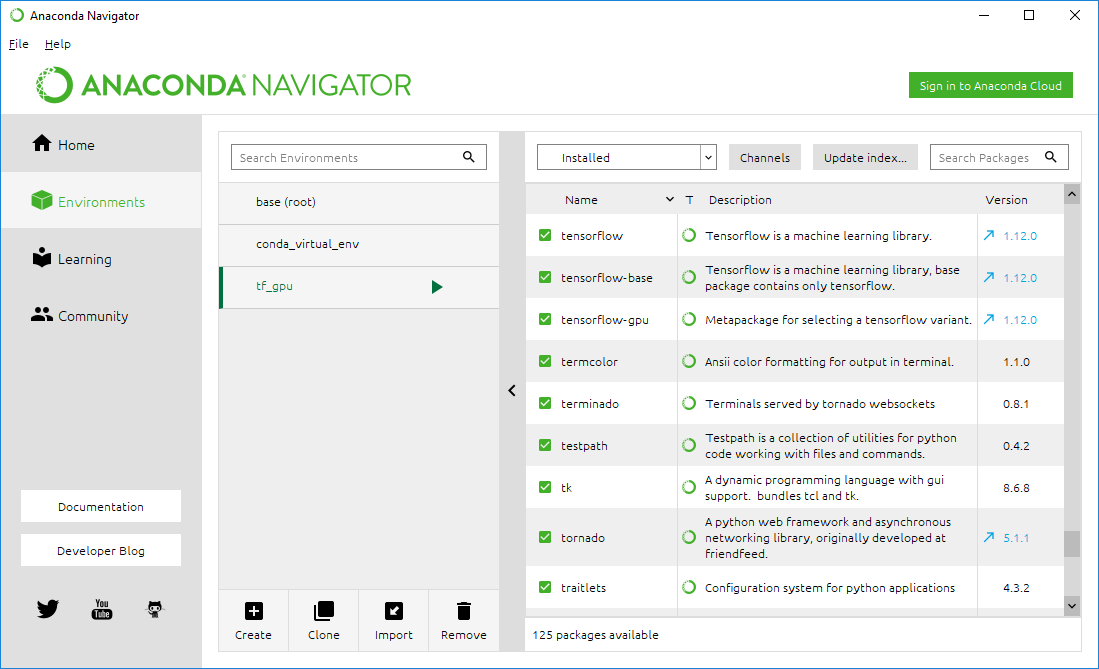
Thông số mạng neural mà nhóm đã xây dựng được thể hiện bằng bảng sau:

|  |  |  |
| --- | --- | --- |
| **Layer (type)** | **Output Shape** | **Param #** |
| image\_array (Conv2D) | (None, 48, 48, 16) | 800 |
| batch\_normalization\_1 | (Batch (None, 48, 48, 16) | 64 |
| conv2d\_1 (Conv2D) | (None, 48, 48, 16) | 12560 |
| batch\_normalization\_2 | (Batch (None, 48, 48, 16) | 64 |
| activation\_1 (Activation) | (None, 48, 48, 16) | 0 |
| average\_pooling2d\_1 | (Average (None, 24, 24, 16) | 0 |
| dropout\_1 (Dropout) | (None, 24, 24, 16) | 0 |
| conv2d\_2 (Conv2D) | (None, 24, 24, 32) | 12832 |
| batch\_normalization\_3 | (Batch (None, 24, 24, 32) | 128 |
| conv2d\_3 (Conv2D) | (None, 24, 24, 32) | 25632 |
| batch\_normalization\_4 | (Batch (None, 24, 24, 32) | 128 |
| activation\_2 (Activation) | (None, 24, 24, 32) | 0 |
| average\_pooling2d\_2 | (Average (None, 12, 12, 32) | 0 |
| dropout\_2 (Dropout) | (None, 12, 12, 32) | 0 |
| conv2d\_4 (Conv2D) | (None, 12, 12, 64) | 18496 |
| batch\_normalization\_5 | (Batch (None, 12, 12, 64) | 256 |
| conv2d\_5 (Conv2D) | (None, 12, 12, 64) | 36928 |
| batch\_normalization\_6 | (Batch (None, 12, 12, 64) | 256 |
| activation\_3 (Activation) | (None, 12, 12, 64) | 0 |
| average\_pooling2d\_3 | (Average (None, 6, 6, 64) | 0 |
| dropout\_3 (Dropout) | (None, 6, 6, 64) | 0 |
| conv2d\_6 (Conv2D) | (None, 6, 6, 128) | 73856 |
| batch\_normalization\_7 | (Batch (None, 6, 6, 128) | 512 |
| conv2d\_7 (Conv2D) | (None, 6, 6, 128) | 147584 |
| batch\_normalization\_8 | (Batch (None, 6, 6, 128) | 512 |
| activation\_4 (Activation) | (None, 6, 6, 128) | 0 |
| average\_pooling2d\_4 | (Average (None, 3, 3, 128) | 0 |
| dropout\_4 (Dropout) | (None, 3, 3, 128) | 0 |
| conv2d\_8 (Conv2D) | (None, 3, 3, 256) | 295168 |
| batch\_normalization\_9 | (Batch (None, 3, 3, 256) | 1024 |
| conv2d\_9 (Conv2D) | (None, 3, 3, 7) | 16135 |
| global\_average\_pooling2d\_1 | (None, 7) | 0 |
| predictions (Activation) | (None, 7) | 0 |

Bảng ‑ Bảng mô tả mô hình Mạng neural tích chập

Tổng số tham số: 642,935

Nhóm đã thử tự cài đặt các công cụ cần thiết để huấn luyện trên máy của mình. Sử dụng Anaconda để tạo môi trường python ảo, cài đặt các driver cần thiết để có thể tận dụng nhân CUDA trong GPU Nvidia của máy.



Hình . Giao diện Anaconda Navigator

Công cụ Anaconda khá tiện dụng, có thể lưu lại môi trường đã cài đặt thành một file yml để có thể sử dụng ở máy khác một cách dễ dàng. Sử dụng lệnh:

*conda env export > environment.yml*

Sau đó, khi cần tạo lại môi trường ở máy khác, sử dụng lệnh:

*conda env create -f environment.yml*

Để kích hoạt terminal sử dụng môi trường này, sử dụng lệnh

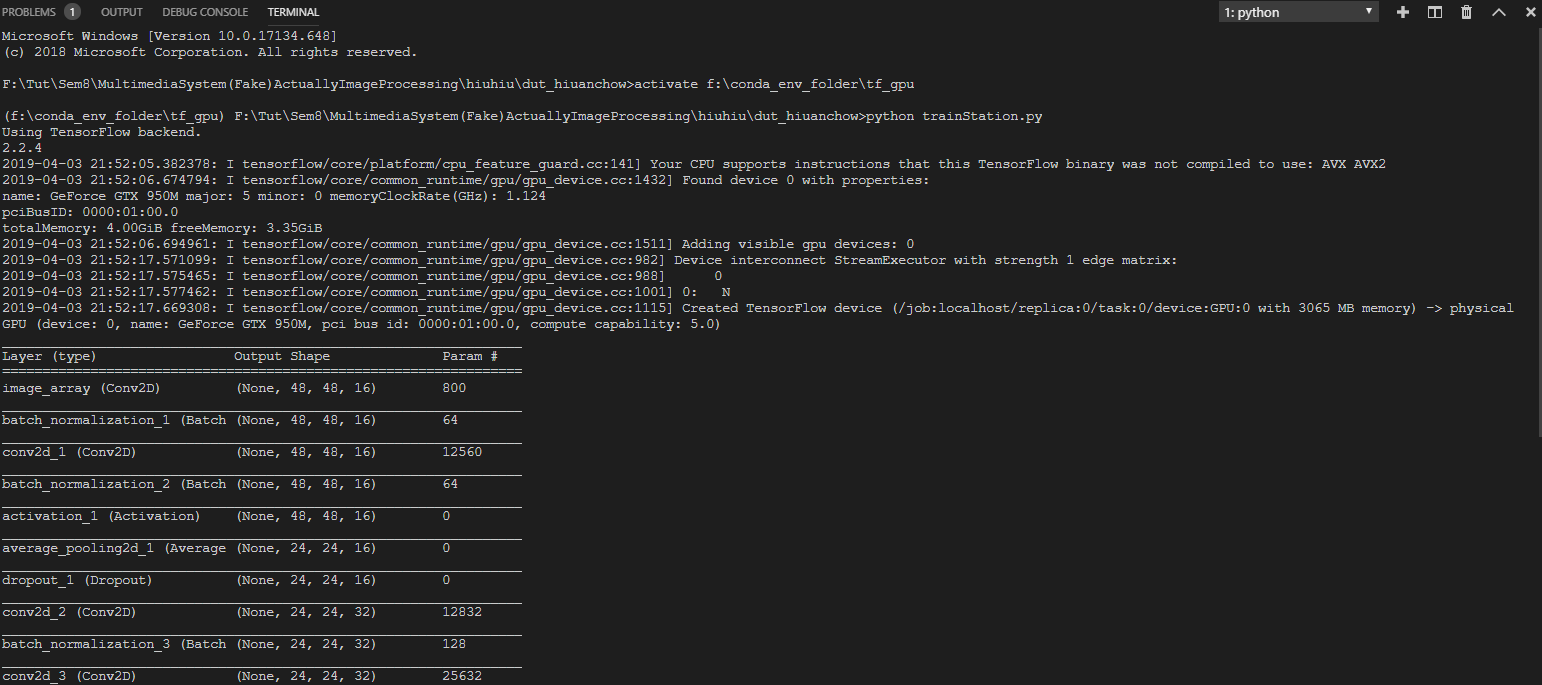
*activate <tên môi trường>*

Ở đây nhóm lưu môi trường này ở ổ F nên câu lệnh kích hoạt là:

*activate f:\conda\_env\_folder\tf\_gpu*

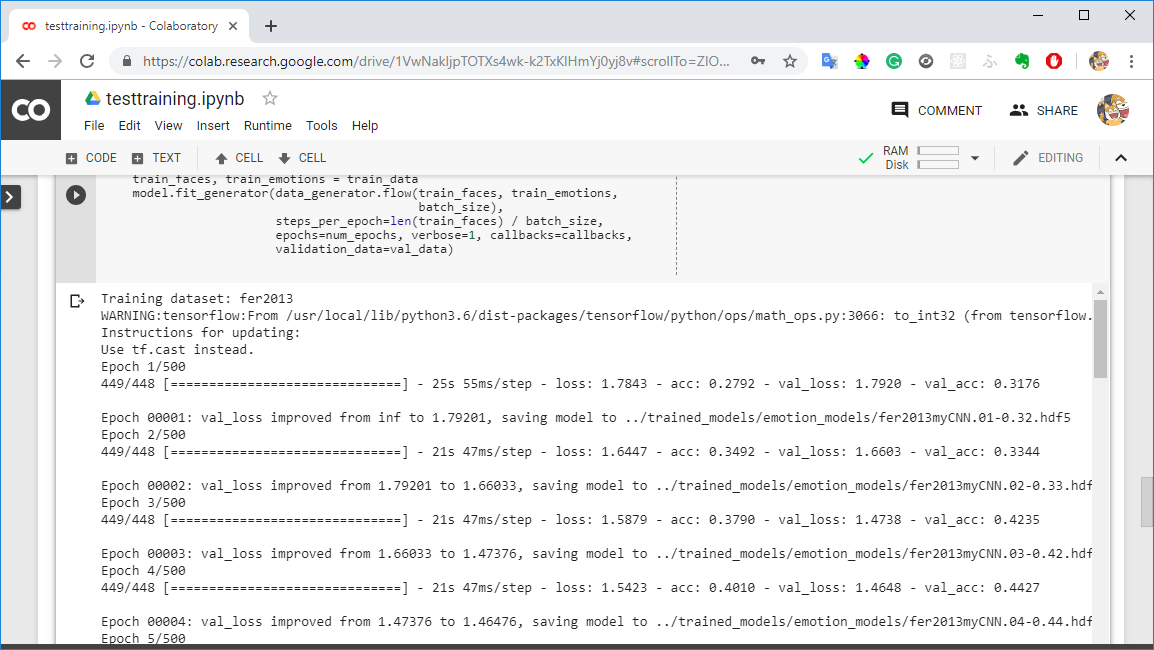
Sau đó để bắt đầu huấn luyện, chạy lệnh:

*python trainStation.py*



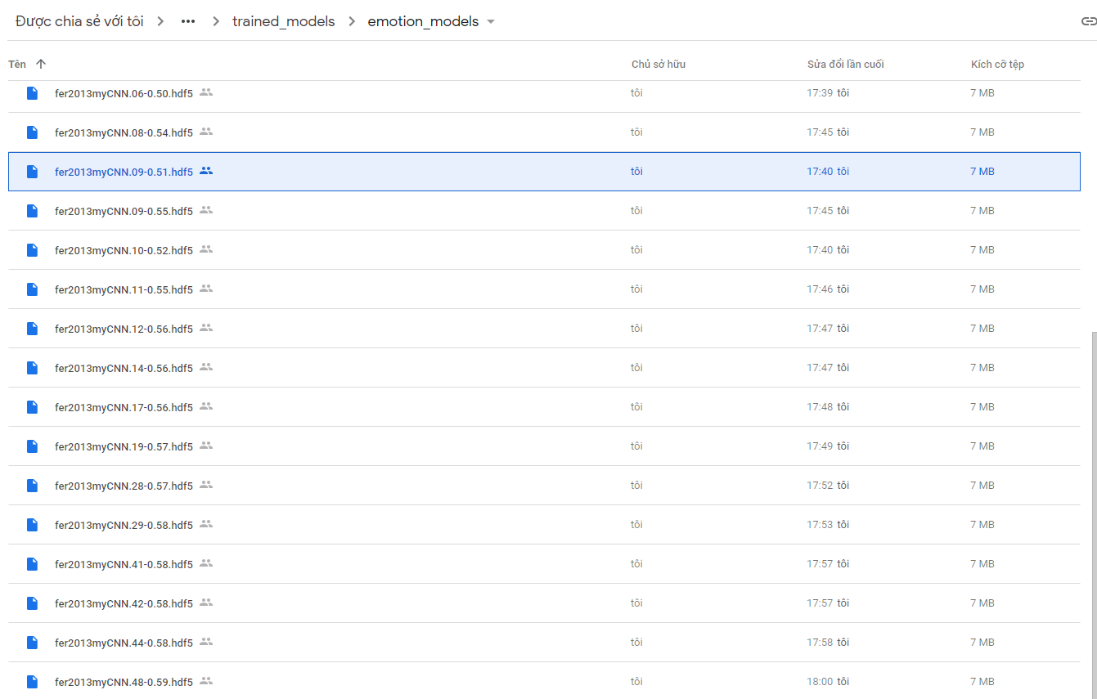
Hình . Chạy lệnh huấn luyện trên terminal (windows cmd)

Tuy nhiên GPU laptop khá yếu nên không thể huấn luyện nhanh, đồng thời nhiệt độ máy lên >90 độ C nên đã dừng việc huấn luyện và đã cố gắng chuyển dữ liệu lên Google Collaboration để huấn luyện. Môi trường Google Collaboration đã được cài đặt sẵn tất cả các thư viện cần thiết, chỉ cần thiết lập thêm một số bước để liên kết đến Google Drive là có thể chạy chương trình bình thường.



Hình . Huấn luyện trên Google Collaborator

Mỗi lần thực hiện huấn luyện sẽ tự đồng bộ dữ liệu lên google drive khi độ chính xác được cải thiện so với lần tước trước đó.. File lưu với dạng: fer2013<tên mạng>.<epoch>-<accuracy>.hdf5



Hình . Các file lưu bộ phân lớp đã huấn luyện

### 2.1.4 Xây dựng chương trình chính

- Sử dụng OpenCV ghi hình trực tiếp từ webcam của laptop.

- Sử dụng Haarcacasde xác định khuôn mặt trên video.

- Với mỗi khuôn mặt xác định được, chuyển sang dạng gray và resize bức ảnh xuống 48x48.

- Sau khi xác định được vị trí khuôn mặt người trong từng khung hình thu được bởi webcam. Đưa dữ liệu vị trí khuôn mặt đó vào mạng neural đã huấn luyện để xác định cảm xúc khuôn mặt trong hình.

Tất cả các bước trên được thực hiện trong file chương trình detectEmo.py

Để chạy chương trình chỉ cần kích hoạt môi trường Anaconda, sau đó chạy lênh:

*python detectEmo.py*

# KẾT QUẢ VÀ ĐÁNH GIÁ

## 3.1 Kết quả training mạng neural tích chập

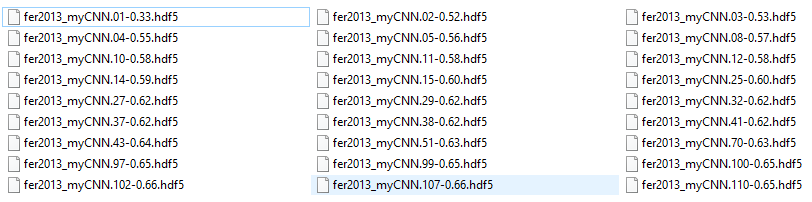
Nhóm thực hiện train mạng neural tích chập với tập dataset gồm 28709 hình. Quá trình huấn luyện bao gồm 110 epoch và tập dataset thành các batch có số lượng ảnh là 64.

Phương pháp huấn luyện này được gọi là Mini-Batch Gradient Descent (tạm dịch là phương pháp huấn luyện theo lô mini). Ở phương pháp này chúng ta thực hiện chia tập dataset này thành các batch nhỏ hơn ngẫu nhiên để cập nhật các tham số của mô hình. Batch size thể hiện số lượng ảnh sử dụng cho một lần cập nhật tham số. Mỗi một epoch là một lần duyệt qua tất cả các ảnh trong tập dataset, và iteration là số lượng batch mà thuật toán phải duyệt trong một epoch.

Như vậy chúng ta mất tổng cộng 28709/64 = 448 iteration để cập nhật tham số mạng neural tích chập trong một epoch.

Sau mỗi epoch nhóm lại xuất ra file có định dạng *\*.hdf5* để lưu lại tham số của mạng neural nhằm mục đích theo dõi độ chính xác của quá trình training qua các epoch. Các thông số epoch và độ chính xác được thể hiện ngay trên tên file như đã được giới thiệu ở phần trước.

Dưới đây là kết quả mà nhóm thu được khi thực hiện train thực tế:



Hình . Kết quả train thực tế

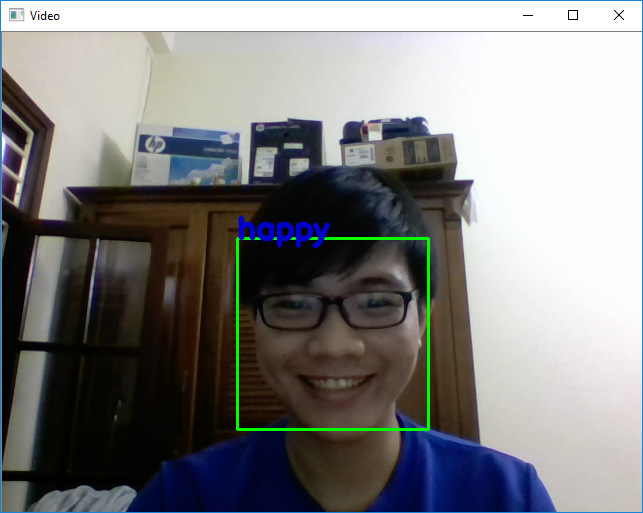
Nhằm thể hiện rõ hơn kết quả của quá trình trainning, nhóm xây dựng biểu đồ tổng quan sự cải thiện độ chính xác của mạng neural tích chập. Trục tung thể hiện độ chính xác đạt được trên tập trainning, trục hoành thể hiện số epoch tương ứng.

Có thể thấy được ở epoch đầu tiên, độ chính xác đạt được chỉ là 32%, tuy nhiên mạng neural tích chập lại cho ra các kết quả tốt hơn ở những epoch sau lần lượt là 55% và 58%. Kết quả tốt nhất cuối cùng mà nhóm đạt được là độ chính xác 65% sau 110 epoch.

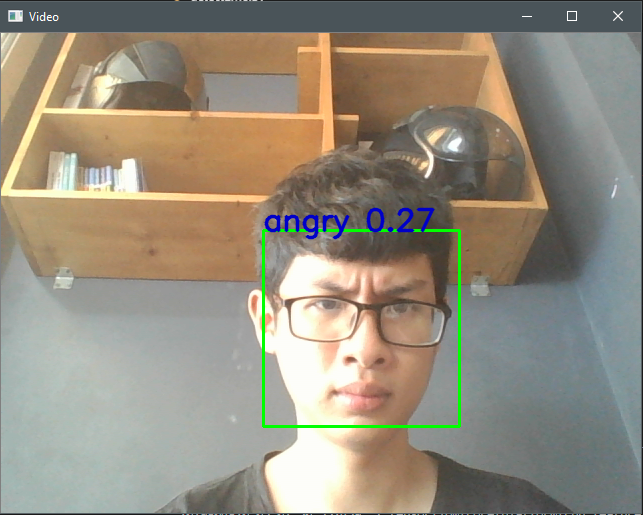
Hình . Biểu đồ thể hiện sự thay đổi độ chính xác qua từng epoch

## 3.2 Kết quả chương trình

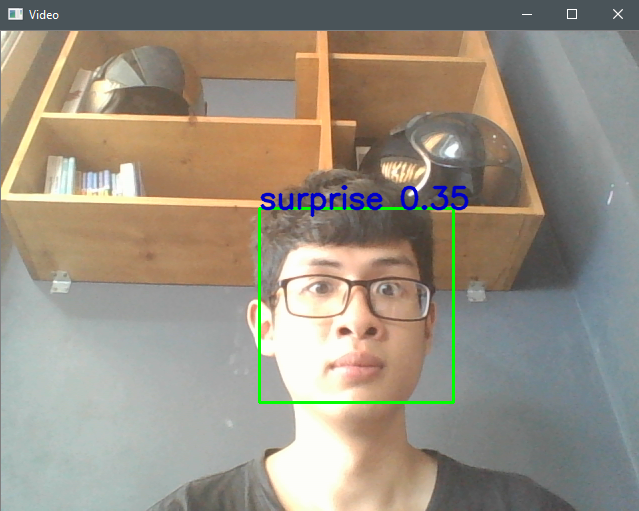
Xây dựng được chương trình nhận diện cảm xúc khuôn mặt *detectEmo.py*. Chương trình này yêu cầu máy tính phải có webcam, có khả năng phát hiện được khuôn mặt và cảm xúc theo thời gian thực. Phần khuôn mặt sẽ được bao quanh bởi ô vuông viền xanh và kèm theo đó là cảm xúc mà mạng neural tích chập phân tích được. Chương trình này còn có thể hiển thị độ chắc chắn (hay xác suất) của cảm xúc mà mạng neural tích chập dự đoán.



Hình . Bản mẫu giao diện chương trình ban đầu

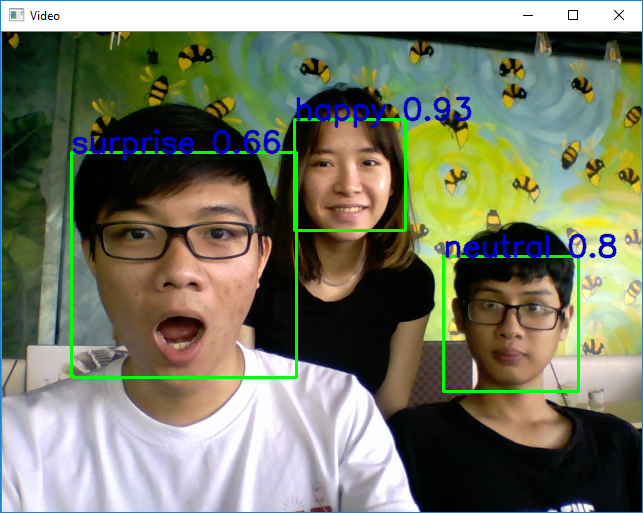


Hình . Nhận diện cảm xúc giận dữ

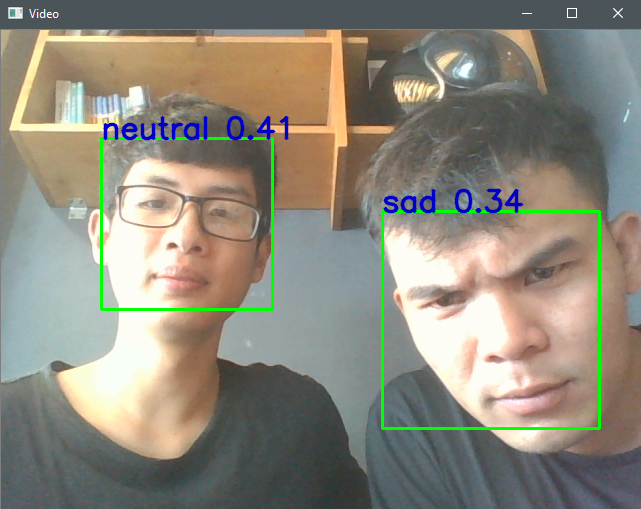


Hình . Nhận diện cảm xúc ngạc nhiên

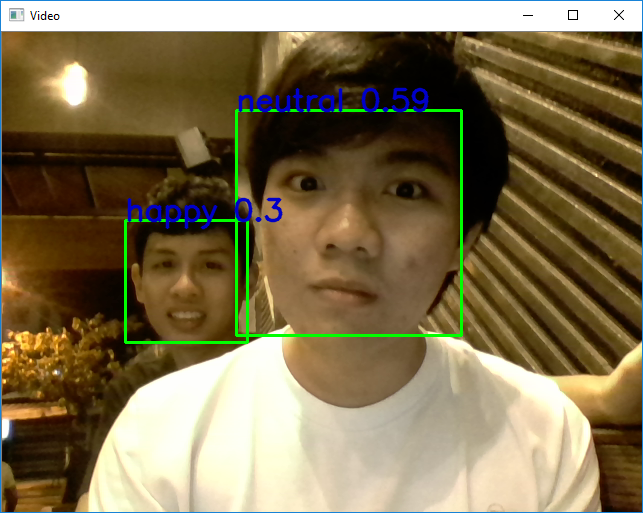
Cũng có thể nhận diện cùng lúc nhiều khuôn mặt xuất hiện trên webcam.



Hình . Nhận diện cảm xúc nhiều khuôn mặt (điều kiện sáng)



Hình . Nhận diện cảm xúc nhiều khuôn mặt (điều kiện sáng)



Hình . Nhận diện cảm xúc nhiều khuôn mặt (điều kiện tối)



Hình . Trường hợp nghiêng mặt sang một bên

Chương trình chỉ phát hiện những khuôn mặt đối diện trực tiếp với webcam, các trường hợp mặt nghiêng sang một bên hoặc chỉ có một nửa thì xem như không phát hiện được khuôn mặt.

Có thể nhận xét rằng chương trình đưa ra các kết quả khá chính xác theo thời gian thực dù điều kiện tối hay sáng. Tuy nhiên cảm xúc Hoảng sợ rất khó để nhận diện.

## 3.3 Đánh giá

So sánh với các chương trình khác đã được xây dựng với cùng đề tài nhận diện cảm xúc khuôn mặt là tạm chấp nhận được. Tuy nhiên các cảm xúc khuôn mặt chủ yếu nhận diện đúng chỉ bao gồm Vui vẻ, Bình thường, và Bất ngờ.

Có thể giải thích cho kết quả trên vì 3 phân lớp cảm xúc trên có các đặc điểm nhận dạng dễ nhận ra. Ví dụ khuôn mặt vui vẻ thường có phần miệng hình chữ U, hay bất ngờ là miệng chữ O,...

Với đề tài lần này, kết quả đạt được của nhóm là đã xây dựng được chương trình cơ bản nhận diện được khuôn mặt và cảm xúc khuôn mặt sử dụng OpenCV và mạng neural tích chập. Tuy nhiên mức độ tìm hiểu vẫn chưa sâu, chỉ mới ở mức sử dụng thư viện và xây dựng lại theo các mô hình đã được nghiên cứu sẵn.

Bên cạnh những kết quả đạt được còn có những vấn đề mà nhóm chưa giải quyết được:

* Việc nghiên cứu về một số giải thuật trích chọn đặc trưng.
* Khảo sát sự ảnh hưởng của các optimizer (tham số tối ưu hóa) trong mạng neural tích chập giúp mạng có khả năng cải thiện nhanh hơn
* Phát hiện được khuôn mặt ở nhiều góc độ khác nhau, hiện nay nhóm chỉ dừng lại ở mức độ khuôn mặt đầy đủ và đối diện với webcam.

# Tài liệu tham khảo

|  |  |
| --- | --- |
| [1] | Y. Miyakoshi and S. Kato, "Facial Emotion Detection Considering Partial Occlusion of Face Using Bayesian Network," 2011. |
| [2] | M. Xiaoxi and L. Weisi, "Facial Emotion Recognition," *IEEE 2nd International Conference on Signal and Image Processing,* 2017. |
| [3] | A. Krizhevsky, I. Sutskever and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," vol. 1, p. 4, 2012. |
| [4] | A. Yao, J. Shao, N. Ma and Y. Chen, "Capturing au-aware facial features and their latent relations for emotion recognition in the wild," *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction,* pp. 415-458, 2015. |
| [5] | C. Shan, S. Gong and P. W. McOwan, "Facial expression recognition based on local binary patterns: A comprehensive study," 2009. |
| [6] | S. Kahou, V. Michalski, K. Konda, R. Memisevic and a. C. Pal, "Recurrent neural networks for emotion recognition in video," *ICMI,* pp. 467-474, 2015. |
| [7] | C. Zhang and Z. Yu, "Image based static facial expression recognition with multiple deep network learning," *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction,* 2015. |
| [8] | B. Kim, J. Roh, S. Dong and S. Lee, "Hierarchical committee of deep convolutional neural networks for robust facial expression recognition," *Journal on Multimodal User Interfaces,* 2016. |
| [9] | G. Levi and T. Hassner, "Emotion recognition in the wild via convolutional neural networks and mapped binary patterns," *Proc. ACM International Conference on Multimodal Interaction,* November 2015. |
| [10] | S. Ouellet, "Real-time emotion recognition for gaming using deep convolutional network features," 2014. |
| [11] | S. University, "CS231n: Convolutional Neural Networks for Visual Recognition," [Online]. Available: http://cs231n.stanford.edu/. |
| [12] | P. Viola and M. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features," *ACCEPTED CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION 2001,* 2001. |