

**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

## **ĐỒ ÁN TỐT NGHIỆP**

### **Xây dựng hệ thống chấm công dựa trên công nghệ nhận dạng khuôn mặt**

**PHAN ĐĂNG HOÀNG**

hoang.pd155650@sis.hust.edu.vn

**Giảng viên hướng dẫn:** TS. Nguyễn Đình Thuận

---

Chữ ký của GVHD

**Bộ môn:** Kỹ thuật máy tính

**Viện:** Công nghệ thông tin và Truyền thông

**HÀ NỘI, 12/2020**

# PHIẾU GIAO NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

## 1 Thông tin về sinh viên

Họ và tên sinh viên: Phan Đăng Hoàng

Điện thoại liên lạc: 0936169578

Email: *hoang.pdang578@gmail.com*

Lớp: CN-CNTT 01 K60

Hệ đào tạo: Cử nhân công nghệ

Đồ án tốt nghiệp được thực hiện tại: Viện Công nghệ thông tin và Truyền thông - Trường đại học Bách Khoa Hà Nội.

Thời gian làm ĐATN: từ 9/2020 đến 30/12/2020.

## 2 Mục đích nội dung của ĐATN

Xây dựng, thử nghiệm hệ thống chấm công dựa trên việc nhận dạng khuôn mặt bằng phương pháp học sâu.

## 3 Các nhiệm vụ cụ thể của ĐATN

- Tìm hiểu tổng quan xử lý ảnh.
- Tìm hiểu mạng nơ-ron nhân tạo, mạng nơ-ron tích chập và mạng hồi quy.
- Tìm hiểu về thuật toán phát hiện khuôn mặt Histogram of Oriented Gradient.
- Tìm hiểu, nghiên cứu các mô hình học sâu đã đạt hiệu quả tốt cho việc phân loại khuôn mặt.
- Ứng dụng thử nghiệm vào của bài toán trong thực tế.

## 4 Lời cam đoan của sinh viên

Tôi - Phan Đăng Hoàng – cam kết ĐATN là công trình nghiên cứu của bản thân tôi dưới sự hướng dẫn của TS. Nguyễn Đình Thuận.

Các kết quả nêu trong ĐATN là trung thực, không phải sao chép toàn văn của bất kỳ công trình nghiên cứu nào khác.

Hà Nội, ngày 31 tháng 12 năm 2020

Tác giả đồ án tốt nghiệp

Phan Đăng Hoàng

## 5 Xác nhận của giáo viên hướng dẫn về mức độ hoàn thành của ĐATN và cho phép bảo vệ

Hà Nội, ngày 31 tháng 12 năm 2020

Giáo viên hướng dẫn

TS. Nguyễn Đình Thuận

## **Lời cảm ơn**

Lời cảm ơn chân thành em xin được gửi đến các thầy cô trường Đại học Bách Khoa Hà Nội và đặc biệt các thầy cô trong Viện Công nghệ thông tin và Truyền thông. Trong hơn 5 năm qua, em đã học được không những rất nhiều kiến thức mà còn trang bị cho em những hành trang vững chắc bước đi sau này. Qua những thời gian học tập, sống với Bách Khoa đã giúp em trở nên không ngại khó khăn thử thách, luôn sẵn sàng với tinh thần chiến đấu, học hỏi, kiên trì và không ngừng theo đuổi ước mơ, đam mê của bản thân.

Đặc biệt nhất, em xin gửi lời cảm ơn chân thành nhất tới thầy Nguyễn Đình Thuận, thầy đã giảng dạy trong quá trình học tập các học phần tại trường và hướng dẫn, giúp đỡ em hoàn thành ĐATN. Mặc dù còn nhiều thiếu sót nhưng dưới sự giúp đỡ tận tình của thầy giúp em có động lực hoàn thành đề tài “Xây dựng hệ thống chấm công dựa trên công nghệ nhận dạng khuôn mặt” một cách tốt nhất. Ngoài ra, em cũng xin chân thành cảm ơn những người bạn đã luôn ở bên cạnh giúp đỡ em trong quá trình thực hiện ĐATN.

Cuối cùng, xin cảm ơn tới gia đình em đã luôn là hậu phương vững chắc để em có thể triển khai theo đuổi sự nghiệp học tập tại trường. Mặc dù thành tích học không được như mong muốn nhưng gia đình luôn là nguồn động lực to lớn, luôn ở cạnh động viên em để em hoàn thành chương trình học.

Sinh viên

Phan Đăng Hoàng

## **Tóm tắt nội dung đề án**

Đề án này nhằm mục đích áp dụng các phương pháp học sâu thử nghiệm xây dựng hệ thống chấm công, nhằm tạo ra sản phẩm phục vụ cho các doanh nghiệp, tổ chức.

Bố cục của đề án gồm có 5 chương :

*Chương 1. Giới thiệu chung về lĩnh vực.* Chương này sẽ giới thiệu chung về lĩnh vực phát hiện, nhận dạng khuôn mặt. Trình bày vấn đề cần giải quyết, giới hạn phạm vi đề tài, hướng giải pháp cho vấn đề trong khuôn khổ đề án.

*Chương 2. Cái nhìn tổng thể và các hướng tiếp cận.* Chương này sẽ giới thiệu tổng thể về bài toán, trình bày các hướng nghiên cứu, sẽ trình bày sơ lược về các hướng tiếp cận.

*Chương 3. Một số kiến thức liên quan.* Trong chương này sẽ nói một số vấn đề xử lý dữ liệu ảnh trong bài toán. Phần sau của chương trình bày các thuật toán sử dụng cho bài toán và một số vấn đề liên quan.

*Chương 4. Xây dựng ứng dụng.* Nội dung của chương sẽ trình bày về việc phân tích thiết kế chức năng, thiết kế giao diện, cài đặt và kiểm thử chương trình.

*Chương 5. Kết luận và hướng phát triển trong tương lai.* Nội dung của chương sẽ trình bày về những điểm đã đạt được, những điểm còn chưa giải quyết được và hướng phát triển trong quá trình nghiên cứu tiếp theo.

Sinh viên thực hiện

## MỤC LỤC

<b>CHƯƠNG 1. GIỚI THIỆU CHUNG VỀ LĨNH VỰC.....</b>	<b>1</b>
1.1 Các khái niệm.....	1
1.1.1 Hệ thống nhận dạng khuôn mặt .....	1
1.1.2 Hệ thống xác thực .....	1
1.1.3 Ứng dụng trong thực tế .....	2
1.2 Phạm vi đề tài.....	1
1.3 Bài toán và hướng giải pháp .....	1
<b>CHƯƠNG 2. CÁI NHÌN TỔNG THỂ VÀ CÁC HƯỚNG TIẾP CẬN.....</b>	<b>4</b>
2.1 Ý tưởng chung của các phương pháp .....	4
2.2 Các công trình nghiên cứu .....	5
2.3 Một số phương pháp cổ điển.....	6
2.3.1 Trích chọn đặc trưng Weber local Descripor .....	6
2.3.2 Viola Jone Face Detection .....	7
<b>CHƯƠNG 3. MỘT SỐ KIẾN THỨC LIÊN QUAN.....</b>	<b>11</b>
3.1 Convolution neural network .....	11
3.1.1 Giới thiệu tổng quan .....	11
3.1.2 Convolution neural Network .....	11
3.2 Một số vấn đề liên quan .....	15
<b>CHƯƠNG 4. XÂY DỰNG ỨNG DỤNG.....</b>	<b>19</b>
4.1 Thực hiện phát hiện khuôn mặt.....	19
4.2 Thực hiện phân loại, dự đoán .....	27
4.3 Ứng dụng và thực nghiệm .....	33
<b>CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN .....</b>	<b>39</b>
<b>TÀI LIỆU THAM KHẢO.....</b>	<b>40</b>
<b>PHỤ LỤC.....</b>	<b>41</b>



## DANH MỤC HÌNH VẼ

<b>Hình 1.1</b>	Hệ thống xác thực khuôn mặt .....	1
<b>Hình 1.2</b>	Tổng quan quá trình huấn luyện và nhận dạng của hệ thống .....	2
<b>Hình 2.1</b>	Các yếu tố quan trọng trong hệ thống nhận dạng .....	4
<b>Hình 2.2</b>	Quá trình tính toán của phương pháp WLD.....	7
<b>Hình 2.3</b>	Haar Features được sử dụng trong Viola Jones .....	8
<b>Hình 2.4</b>	Áp dụng trên ảnh .....	8
<b>Hình 2.5</b>	Ví dụ ảnh Intergral .....	9
<b>Hình 2.6</b>	Các bước phân loại khuôn mặt sử dụng Cascade .....	10
<b>Hình 2.7</b>	Kết quả của Viola Jones Face Detection .....	10
<b>Hình 3.1</b>	Ví dụ về một mạng Neural Network .....	11
<b>Hình 3.2</b>	Công thức tính tích chập .....	12
<b>Hình 3.3</b>	Ma trận đầu vào và kernel .....	13
<b>Hình 3.4</b>	Ma trận Input x Kernel và Feature map .....	13
<b>Hình 3.5</b>	Ví dụ với stride=1 và padding =1 .....	14
<b>Hình 3.6</b>	Công thức tổng quát tính kích thước Feature map .....	14
<b>Hình 3.7</b>	Quá trình chuyển Feature map thành Vector .....	15
<b>Hình 3.8</b>	Ví dụ về mô hình Covolution Neural Network .....	15
<b>Hình 3.9</b>	Confusion maxtric tổng quát .....	16
<b>Hình 4.1</b>	HOG trong ứng dụng phát hiện người đi bộ .....	20
<b>Hình 4.2</b>	Trực quan các tính năng sau khi sử dụng HOG.....	20
<b>Hình 4.3</b>	HOG trong việc trích xuất tính năng .....	21
<b>Hình 4.4</b>	Quy trình phát hiện đối tượng sử dụng phương pháp Histogram of Oriented Gradient .....	21
<b>Hình 4.5</b>	Một ảnh sau khi áp dụng Sobel mask .....	23
<b>Hình 4.6</b>	Kết quả của một ảnh sau khi thực hiện phép tính Gradient .....	24
<b>Hình 4.7</b>	Hình ảnh đối tượng được chia thành nhiều các ô. Sau đó thực hiện tính đạo hàm trên từng ô .....	24
<b>Hình 4.8</b>	Ánh xạ độ lớn gradients vào các bins .....	25
<b>Hình 4.9</b>	Ví dụ về việc ánh xạ các giá trị cường độ Gradient tương ứng .....	26
<b>Hình 4.10</b>	Biểu đồ histogram của Gradient gồm 9 bins tương ứng với mỗi một ô vuông trong lưới .....	26
<b>Hình 4.11</b>	Biểu diễn các vector histogram trên các lưới ô vuông của hình ảnh. . gốc .....	27
<b>Hình 4.12</b>	Kiến trúc mô hình VGG16 .....	29
<b>Hình 4.13</b>	Những mô hình khác nhau của VGG .....	30
<b>Hình 4.14</b>	Minh họa kỹ thuật Transfer Learning .....	31
<b>Hình 4.15</b>	Bộ cơ sở dữ liệu 87 nhân viên .....	33
<b>Hình 4.16</b>	Ảnh chân dung khuôn mặt được chụp ở nhiều góc độ, biểu cảm khác nhau .....	33
<b>Hình 4.17</b>	Webcam laptop Dell E7 sử dụng cho bài toán.....	34
<b>Hình 4.18</b>	Kết quả quá trình phát hiện khuôn mặt dựa trên thuật toán HOG .....	34
<b>Hình 4.19</b>	Hình ảnh khuôn mặt thu được sau quá trình phát hiện khuôn mặt ...	35
<b>Hình 4.20</b>	Quá trình đưa ra đánh giá, dự đoán của mô hình .....	35
<b>Hình 4.21</b>	Kiến trúc mô hình pre-trained VGGFace .....	36
<b>Hình 4.22</b>	Quá trình huấn luyện trên tập dữ liệu thực .....	36
<b>Hình 4.23</b>	Hình ảnh hiển thị kết quả phân loại .....	37

<b>Hình 4.24</b>	<i>Giao diện đăng nhập .....</i>	<i>37</i>
<b>Hình 4.25</b>	<i>Hiển thị danh sách nhân viên .....</i>	<i>38</i>
<b>Hình 4.26</b>	<i>Giao diện hiển thị thông tin về lịch sử làm việc của nhân viên.....</i>	<i>38</i>



## DANH MỤC TỪ VIẾT TẮT

<b>API</b>	Application Programming Interface Giao diện lập trình ứng dụng
<b>HOG</b>	Histogram of Oriented Gradient
<b>CNN</b>	Convolutional Neural Network Mạng rơ-ron tích chập
<b>CNTT</b>	Công nghệ thông tin
<b>PCA</b>	Principal Component Analysis
<b>DNN</b>	Deep Neural Network Mạng nơ-ron sâu
<b>SVM</b>	Support Vector Machine
<b>DCT</b>	Discrete Cosine Transform Biến đổi Cosin rời rạc
<b>DFT</b>	Discrete Fourier Transform Biến đổi Fourier rời rạc
<b>ĐATN</b>	Đồ án tốt nghiệp
<b>RNN</b>	Recurrent Neural Network Mạng hồi quy

## DANH MỤC THUẬT NGỮ

<b>Computer Vision</b>	Thị giác máy tính
<b>Deep Learning</b>	Học sâu
<b>Classifier</b>	Bộ phân lớp có nhiệm vụ xác định ảnh đầu vào có phải là khuôn mặt hay không-phải-là-mặt.
<b>Face detection</b>	Phát hiện khuôn mặt nhằm tìm ra vị trí và kích thước của những khuôn mặt hiện diện trong ảnh.
<b>Face image hoặc positive example</b>	Ảnh mẫu khuôn mặt người.
<b>Cascaded Classifier</b>	Bộ phân lớp xếp tầng, mỗi tầng là một bộ phân lớp mạnh được xếp từ đơn giản đến phức tạp.
<b>Haar feature</b>	Những chi tiết được định nghĩa giống hàm sóng Haar được sử dụng để xây dựng những bộ phân lớp yếu trong giải thuật AdaBoost
<b>Knowledge</b>	Tri thức
<b>File</b>	Tệp
<b>Nature Language Processing</b>	Xử lý ngôn ngữ tự nhiên
<b>Neural Network</b>	Mạng nơ-ron
<b>Network</b>	Mạng
<b>End-to-end</b>	Đầu cuối
<b>Vector</b>	Vec-tơ
<b>Smart phone</b>	Điện thoại thông minh

## CHƯƠNG 1. GIỚI THIỆU CHUNG VỀ LĨNH VỰC

### 1.1 Các khái niệm

#### 1.1.1 Hệ thống nhận dạng khuôn mặt

Hệ thống nhận dạng khuôn mặt là một ứng dụng thị giác máy tính cho phép tự động xác định hoặc nhận dạng một người nào đó từ một bức ảnh kỹ thuật số hoặc một khung hình được truyền vào máy tính từ một nguồn video. Một trong những cách để thực hiện được điều này là so sánh các đặc điểm khuôn mặt chọn trước từ hình ảnh và một cơ sở dữ liệu về khuôn mặt.

Hệ thống này thường được sử dụng trong các hệ thống an ninh, giám sát và có thể được tích hợp cùng với nhiều công nghệ nhận dạng sinh trắc học khác như: tròng mắt, vân tay,..... cảm biến.

#### 1.1.2 Hệ thống xác thực

Là hệ thống được thiết kế để xác minh thông tin của một người. Kỹ thuật xác minh ở đây là dựa trên một phép so sánh, kiểm tra về mặt thông tin truyền vào hệ thống của người đó, với thông tin đã được lưu trữ trên cơ sở dữ liệu. Đánh giá và so sánh về mức độ trùng khớp.



Hình 1.1 Hệ thống xác thực khuôn mặt.

#### 1.1.3 Ứng dụng trong thực tế

Từ những năm 1990 trở lại đây, chúng ta đã chứng kiến sự phát triển vượt bậc của các ngành điện tử, công nghiệp, công nghệ thông tin,... đặc biệt là ngành chế tạo điện tử. Nhiều thiết bị điện tử cao cấp ra đời như máy tính, camera kỹ thuật số, smartphone,... và nhiều sản phẩm khác. Hiện nay các sản phẩm này không chỉ có mặt tại các phòng thí nghiệm, do chi phí sở hữu những sản phẩm này trong những năm gần đây phù hợp với người dùng. Nên đồng nghĩa với việc mở ra cánh cửa lớn để ứng dụng những nghiên cứu về thị giác máy tính trong đời sống. Đồng thời là sự phát triển về mặt giao tiếp giữa con người và máy tính mà trong đó hệ thống nhận dạng khuôn mặt đóng một vai trò không nhỏ. Dưới

đây là một ố ứng dụng thực tế của nó:

- Các ứng dụng chuyên biệt trong ngành hàng không.
- Ứng dụng trong việc xây dựng hệ thống nhà thông minh.
- Kết hợp với những hệ thống trong quân sự,....

## **1.2 Phạm vi đề tài**

Đồ án hướng tới việc xây dựng hệ thống nhận dạng nhân viên trong một đơn vị tổ chức, doanh nghiệp. Cơ sở dữ liệu cho đồ án này còn gặp nhiều hạn chế trong việc thu thập dữ liệu thực tế để huấn luyện, thử nghiệm cho mô hình. Do đồ án mang tính chất thử nghiệm và có nhiều hạn chế về thời gian, số lượng người được thu thập nên đồ án chỉ thực hiện trên bộ dữ liệu nhỏ.

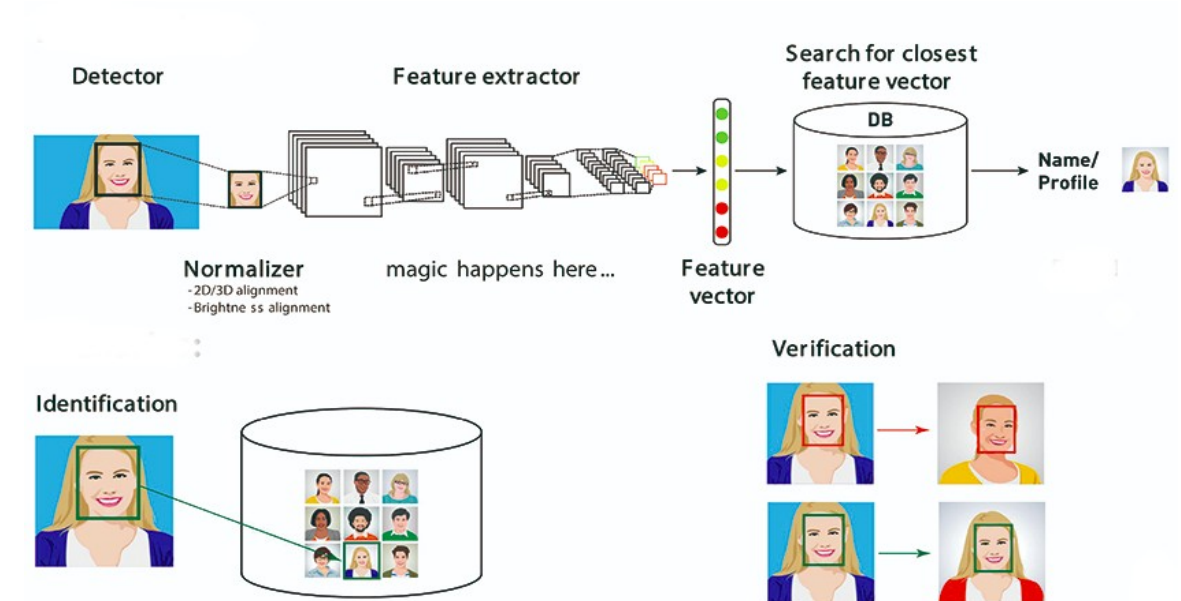
## **1.3 Bài toán và hướng giải pháp**

Nhận dạng khuôn mặt là một bài toán lâu đời và được nghiên cứu rộng rãi trong khoảng hơn 30 năm trở lại đây. Bài toán nhận dạng khuôn mặt có thể áp dụng rộng rãi trong nhiều lĩnh vực khác nhau. Các ứng dụng liên quan đến nhận dạng khuôn mặt có thể kể đến như: Hệ thống phát hiện tội phạm, hệ thống theo dõi nhân sự trong một đơn vị, hệ thống tìm kiếm trên ảnh, tìm kiếm thông tin dựa trên nội dung,....Hiện nay, bài toán nhận dạng khuôn mặt còn gặp nhiều thách thức về độ chính xác, dữ liệu, chất lượng hình ảnh.

Hiện nay, các phương pháp đều nhận diện đều có đặc điểm chung là sử dụng các thuật toán chuyển miền không gian( *Facenet*, *DeepFace*,....) ban đầu về miền không gian để trích xuất ra những đặc trưng quan trọng. Sau đó sử dụng một miền không gian cho phép những đặc trưng đó đi qua và so sánh với vị trí trong miền không gian của những người đã biết.

Trong những năm gần đây, với những thành công vượt trội của việc áp dụng các phương pháp học sâu(*Deep Learning*) trong giải quyết các bài toán phức tạp trong xử lý ngôn ngữ tự nhiên(*Natural Language Processing*), thị giác máy tính(*Computer Vision*), *Parttern Recognize* .... và nhiều lĩnh vực khác. Trong năm 2014, và năm 2015 hai mô hình được giới thiệu cho nhận dạng khuôn mặt đó là *Ensemble of Regression Trees*[1], *FaceNet* [2], hoặc một số phương pháp đạt hiệu quả cao có thể kể đến như *HOG*[3]. Vì thế hiện nay có thể coi học sâu là một trong những phương pháp lựa chọn thử nghiệm tối ưu trong việc giải quyết bài toán nhận dạng khuôn mặt.

Tôi xin được giới thiệu tổng thể về những phương pháp trên ở chương sau. Để từ đó đưa ra những so sánh về độ hiệu quả của chúng trong mô hình hệ thống thức tế.

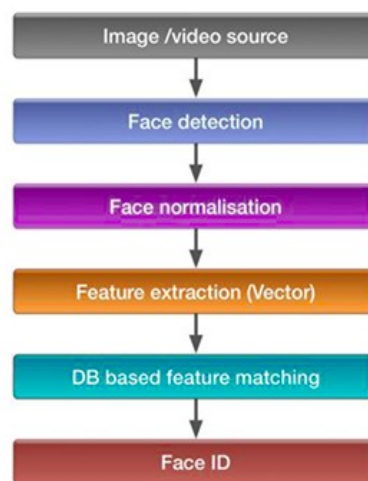


Hình 1.2 Tổng quan quá trình huấn luyện và nhận dạng của hệ thống.

## CHƯƠNG 2. CÁI NHÌN TỔNG THỂ VÀ CÁC HƯỚNG TIẾP CẬN

### 2.1 Ý tưởng chung của các phương pháp

Những tiến bộ gần đây trong phân tích khuôn mặt tự động, nhận mẫu và học máy đã giúp chúng ta có thể xây dựng, phát triển các hệ thống nhận dạng một cách đơn giản. Nhưng bên cạnh đó, việc nhận dạng trong thực tế giữa con người với con người là một quá trình tự nhiên, bởi vì mọi người thường làm điều đó một cách dễ dàng mà không cần ý thức nhiều. Mặt khác, việc áp dụng quy trình này vào lĩnh vực thị giác máy tính vẫn là một vấn đề khó khăn. Là một phần của công nghệ sinh trắc học, nhận dạng khuôn mặt tự động cần phải có được những đặc tính mong muốn riêng của từng người, từng nhóm người. Chúng dựa vào những đặc điểm quan trọng, không nhầm lẫn.



Hình 2.1 Các yếu tố quan trọng trong hệ thống nhận dạng.

Hiệu suất nhận diện khuôn mặt là một vấn đề then chốt, vì vậy kỹ thuật xử lý rút tính năng trên khuôn mặt là một bước vô cùng quan trọng để hệ thống đạt hiệu suất cao, cũng như về mặt chính xác trong dự đoán. Phương pháp chuyển miền không gian của ảnh ban đầu về miền không gian nhỏ hơn nhưng vẫn giữ lại những thông tin quan trọng là nền tảng cho nhiều kỹ thuật nhận dạng hiện tại. Khám phá không gian con này để trích xuất tính năng một cách hiệu quả và xây dựng các bộ phân loại một cách mạnh mẽ là một thách thức hiện tại.

Giai đoạn đầu tiên là nhận diện khuôn mặt trong các hình ảnh thu được bất kể tỷ lệ và vị trí của nó trong bức ảnh. Nó thường sử dụng một quy trình lọc (*filtering*) để phân biệt các vị trí đại diện trên khuôn mặt và lọc chúng bằng các bộ phân loại chính xác. Đáng chú ý là cho dù dịch trái, tất cả những thay đổi về kích cỡ, tỷ lệ, xoay phải, trái,... của đối tượng đều được xử lý ở trong giai đoạn này.

Trong bước tiếp theo hệ thống dựa trên tập dữ liệu để dự đoán vị trí gần đúng của các đặc điểm chính như mắt, mũi, miệng. Tất nhiên, toàn bộ quy trình này được lặp lại để dự đoán các tính năng phụ. liên quan đến các tính năng chính và

sẽ được thống kê lại để loại bỏ đối với các tính năng được phân bổ sai.

Các điểm neo(*anchor*) chuyên dụng được tạo ra là kết quả của sự kết hợp hình học trong hình ảnh khuôn mặt và sau đó bắt đầu quá trình nhận dạng thực tế. Nó được thực hiện bằng cách tìm biểu diễn cục bộ của hình dáng khuôn mặt tại mỗi điểm neo. Sơ đồ đại diện phụ thuộc vào cách tiếp cận. Để đối phó với sự phức tạp như vậy và tìm ra bất biến thực sự của các mẫu, các nhà nghiên cứu đã nghiên cứu và phát triển các thuật toán khác nhau.

Có một số thuật toán hiện nay đã cải tiến được hiệu suất cũng như hiệu quả của bài toán, nhưng với điều kiện là ở trong điều kiện lý tưởng(ánh sáng, độ phân giải, khoảng cách...). Những thuật toán đó khi áp dụng vào thực tế gặp sự thay đổi về điều kiện lý tưởng thì hiệu suất cũng như độ chính xác giảm tương đối đáng kể. Một số vấn đề khác là cách hiệu quả để lưu trữ là truy cập *csdl* khuôn mặt. Cái mà được trích xuất từ hình ảnh, *video* dưới dạng một tập hợp các tính năng.

Xem xét cách tiếp cận được trình bày ngắn gọn ở trên của quá trình phức tạp trong nhận diện khuôn mặt thì có thể thấy một số hạn chế và chưa hoàn hảo. Chương tiếp theo sẽ trình bày sơ lược về các thuật toán các hướng giải quyết được đánh giá cao hiện nay.

## 2.2 Các công trình nghiên cứu

Bài toán nhận dạng khuôn mặt cần xác định hai vấn đề chính: dùng thông tin nào để nhận dạng: mắt, môi, mũi, miệng, tai, hay kết hợp toàn bộ những thông tin trên. Vấn đề thứ hai là dùng phương pháp nào để huấn luyện cho máy tính để tốc độ và độ chính xác trong mức chấp nhận được. Nhiều năm qua lĩnh vực thị giác máy tính đã có nhiều bước tiến nổi bật như:

➤ *Image-based Illumination Inspired by Using Decomposition of Local Singular Value and Discrete Wavelet Transformation*[4] kỹ thuật này là so với chung biểu đồ tiêu chuẩn phân phối (GHE) và biểu đồ địa phương cân bằng (LHE).

➤ *Antonio J.Colmenarez (1998)[5]* sử dụng kỹ thuật *Pattern Recognition*. Ông cho rằng bài toán dò tìm khuôn mặt là thao tác phân loại khuôn mặt trong đó khuôn mặt thuộc về một lớp và các đối tượng khác thuộc về lớp còn lại bằng phương pháp ước lượng mô hình xác suất cho mỗi lớp, và việc dò tìm sử dụng ước lượng hợp lý cực đại (*Maximum-likelihood*).

➤ *Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks*[6] sử dụng một bộ face detector để tách phần ảnh có gương mặt người ra. Các phương pháp hiện đại sử dụng MTCNN (Multi-task Cascaded Convolutional Networks), ưu điểm là mô hình này detect được gương mặt ở nhiều góc độ, và có thể nhận diện cả face landmark với thời gian xử lý khá tốt.

- *Kazunori Okada, Johannes Steffens, Thomas Maurer, Hai Hong, Egor Elagin, Hart Neyven (1998)[7]* dựa vào sóng *Gabor* để nhận dạng khuôn mặt và phương pháp đồ thị. Với ý tưởng dùng đồ thị để biểu diễn khuôn mặt, ảnh khuôn mặt được đánh dấu tại các vị trí đã được xác minh trước trên khuôn mặt, gọi các vị trí này là các vị trí chuẩn. Khi thực hiện thao tác so khớp đồ thị với một ảnh, các điểm chuẩn(*Jets*) sẽ trích ra từ ảnh và so sánh các điểm chuẩn này với tất cả các điểm chuẩn tương ứng có trong đồ thị, và so sánh tương quan giữa chúng sau đó lựa chọn ảnh nào có độ chính xác, ăn khớp cao nhất.
- *Dual Shot Face Detector(2019)[8]* Một hệ nhận diện gương mặt mới với 3 phát hiện mới về 3 yếu tố chính trong nhận diện gương mặt, theo thứ tự là: khả năng học đặc điểm gương mặt tốt hơn, thiết kế hạn chế mất dữ liệu, và tăng cường dữ liệu dựa trên phân bố mẫu neo. Mô hình/Kiến trúc sử dụng: Khung DSFD sử dụng module tăng cường đặc điểm (Feature Enhance Module), kết hợp kiến trúc VGG/ResNet để tổng hợp các đặc điểm được tăng cường so với gốc (a), cùng 2 lớp first shot PAL và second shot PAL.
- *Additive Angular Margin Loss for Deep Face Recognition(2019)[9]* ArcFace có thể nhận diện những đặc điểm sâu, có khả năng phân biệt cao và cho kết quả tối ưu, có thể làm lại trong thử thách MegaFace Challenge. Mô hình: Additive Angular Margin Loss (ArcFace) là công trình sử dụng để tăng cường độ gọn trong cùng lớp, cũng như sự khác biệt đa lớp, bằng các sử dụng khoảng cách giữa sample và tâm, giúp tăng độ chính xác cho các mô hình nhận diện gương mặt. Độ chính xác của mô hình: Qua các thử nghiệm toàn diện, có thể kết luận ArcFace chính xác hơn cả những mô hình tối ưu nhất hiện tại.

## 2.3 Một số phương pháp cổ điển

### 2.3.1 Trích chọn đặc trưng Weber local Descripor-WLD

*Weber local Description(WLD)*: việc con người nhìn nhận một đối tượng không chỉ phụ thuộc vào sự thay đổi kích thước, mà còn phụ thuộc vào cường độ gốc của kích thước. *WLD* gồm hai thành phần chính: *Differential Excitation* và *Gradient Orientation* của ảnh và xây dựng histogram dựa trên thành phần đó:

*Different Excitaions*:

- Sử dụng sự khác nhau giữa cường độ pixel trung tâm và điểm lân cận để miêu tả sự thay đổi của pixel hiện tại. Qua đó ta có thể thấy những đặc điểm nổi bật. Quá trình đó mô phỏng quá trình nhận dạng đối tượng của con người.

*Orientation*:

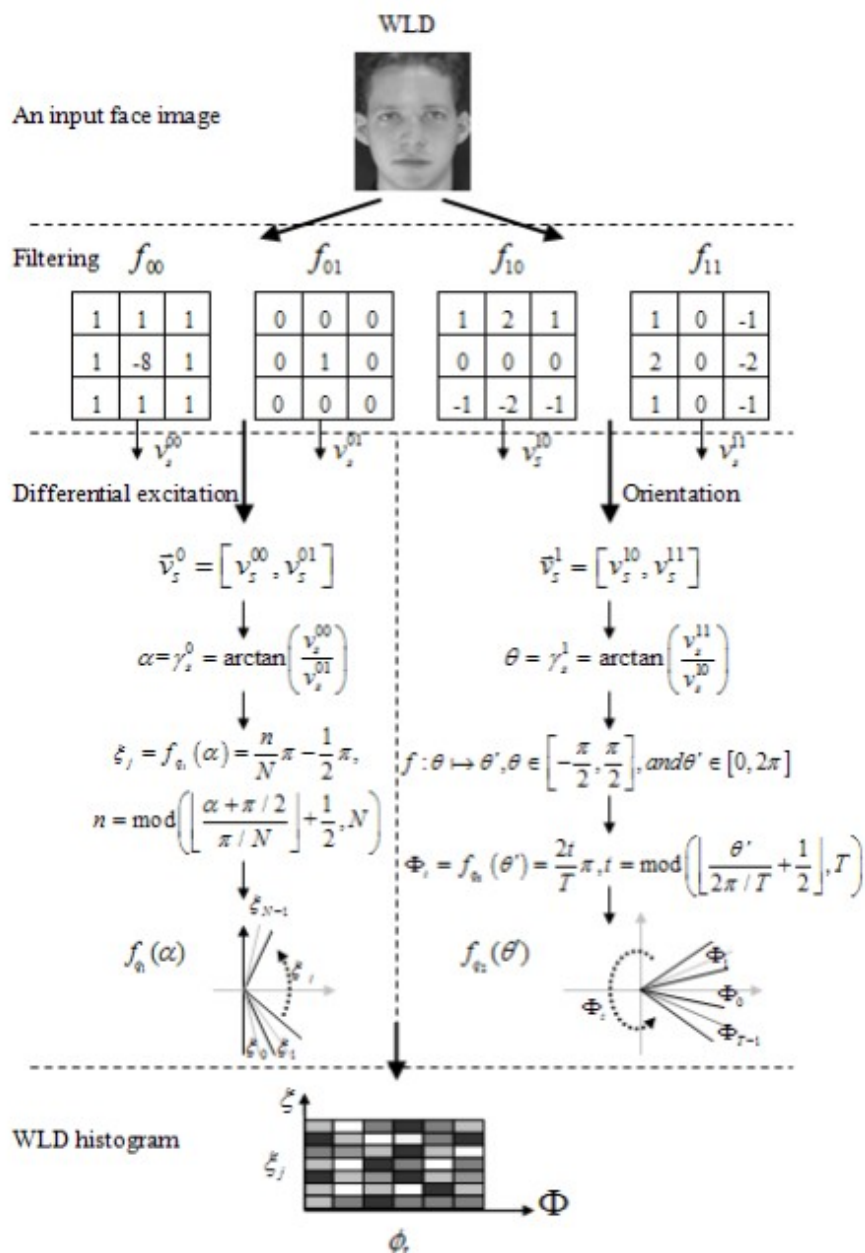
- Đề cập tới việc thay đổi tỷ lệ của ảnh theo phương ngang, phương đứng. Quá trình được đề cập tổng quan ở hình bên dưới.

*WLD Histogram*:

- Quá trình biến đổi biểu đồ *2D WLD histogram* thành *vector* hàng với độ dài



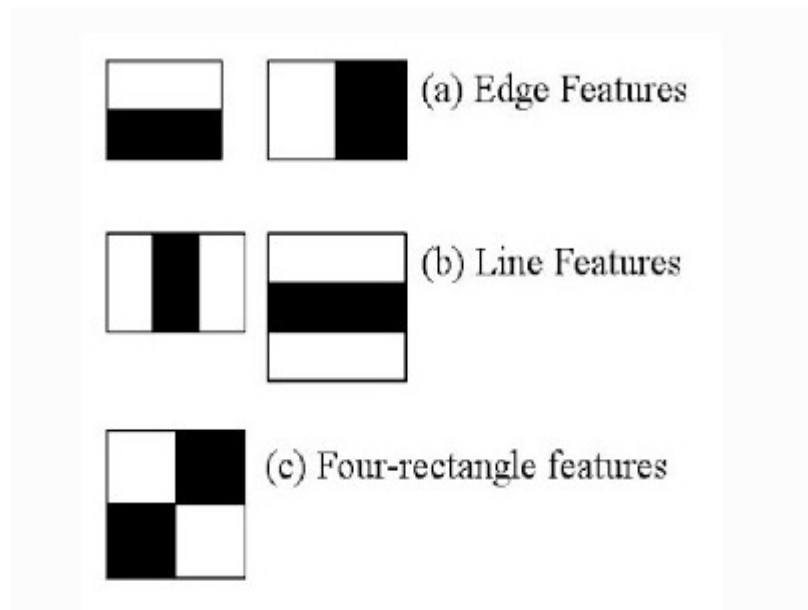
$T \times N$ . Trong đó  $N$  là số “Different Excitations” chiếm ưu thế.  $T$  là số “Orientation” chiếm ưu thế.



Hình 2.2 Quá trình tính toán của phương pháp WLD

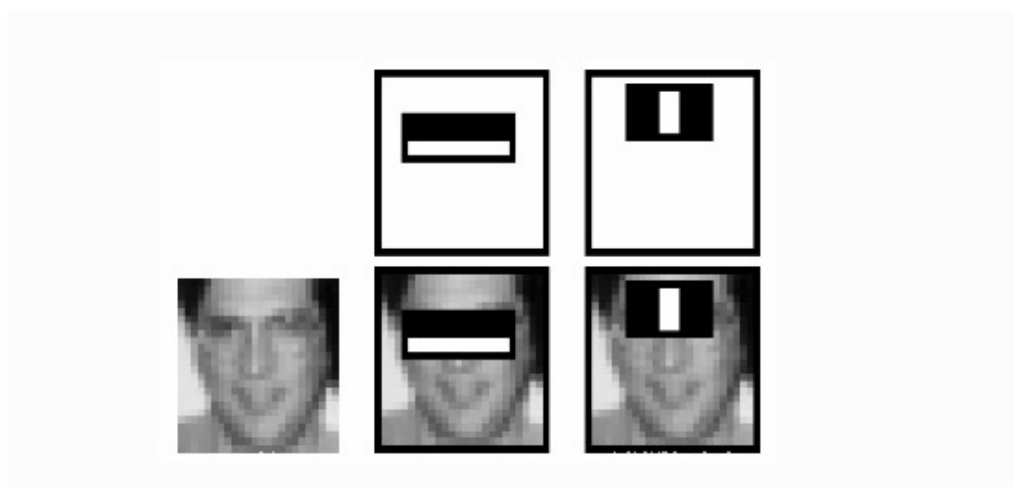
### 2.3.2 Viola Jone Face Detection

Ý tưởng của phương pháp: Độ sáng tối của mỗi vùng trên khuôn mặt là khác nhau. Ví dụ phần môi sẽ sẫm màu hơn vùng má xung quanh, vùng quanh mắt sẽ tối hơn vùng trán,... Với việc tổng hợp kết quả của mỗi đặc trưng được tính bằng hiệu của tổng các pixel trong các miền ô trắng trừ đi tổng pixel trong các miền ô đen.



Hình 2.3 Haar Features được sử dụng trong Viola Jones

Thuật toán *Viola Jones* sử dụng cử số trượt có kích thước  $24 \times 24$  (có thể thay đổi tùy thuộc vào từng yêu cầu bài toán) để đánh giá các đặc trưng của ảnh. Nếu xem tất cả các tham số của các *Features* ta sẽ tính được khoảng hơn 160000 đặc trưng cho mỗi cửa sổ.



Hình 2.4 Áp dụng trên ảnh

*Integral Image*: giá trị ở  $pixel(x, y)$  là tổng của các pixel ở trên và bên trái  $(x, y)$ . Cho phép tính tổng của các *pixel* trong bất kì kernel nào chỉ với 4 giá trị ở góc.

1	1	1
1	1	1
1	1	1

**Ảnh gốc ban đầu**

1	2	3
2	4	6
3	6	9

**Ảnh Integral**

*Hình 2.5 Ví dụ ảnh Integral*

Trong quá trình tính toán, ta thu được nhiều đặc trưng từ một cửa sổ 24x24 nhưng chỉ số ít trong đó có thực sự hữu ích trong việc nhận dạng khuôn mặt.

Để tối ưu, giảm thiểu thời gian quan tâm tới những tính năng không quan trọng. Ta cần sử dụng một thuật toán để tìm ra những đặc trưng có giá trị tốt nhất. Sau đó gán trọng số cho các đặc trưng này để tạo nên một hàm đánh giá để xác thực xem trong cửa sổ đó có phải là khuôn mặt hay không. Các đặc trưng đó được gọi là bộ phân lớp yếu. Sau đó chúng được tổ hợp tuyến tính để tạo ra một bộ phân lớp mạnh.

$$F(x) = \alpha_1 f_1(x) + \alpha_2 f_2(x) + \alpha_3 f_3(x)$$

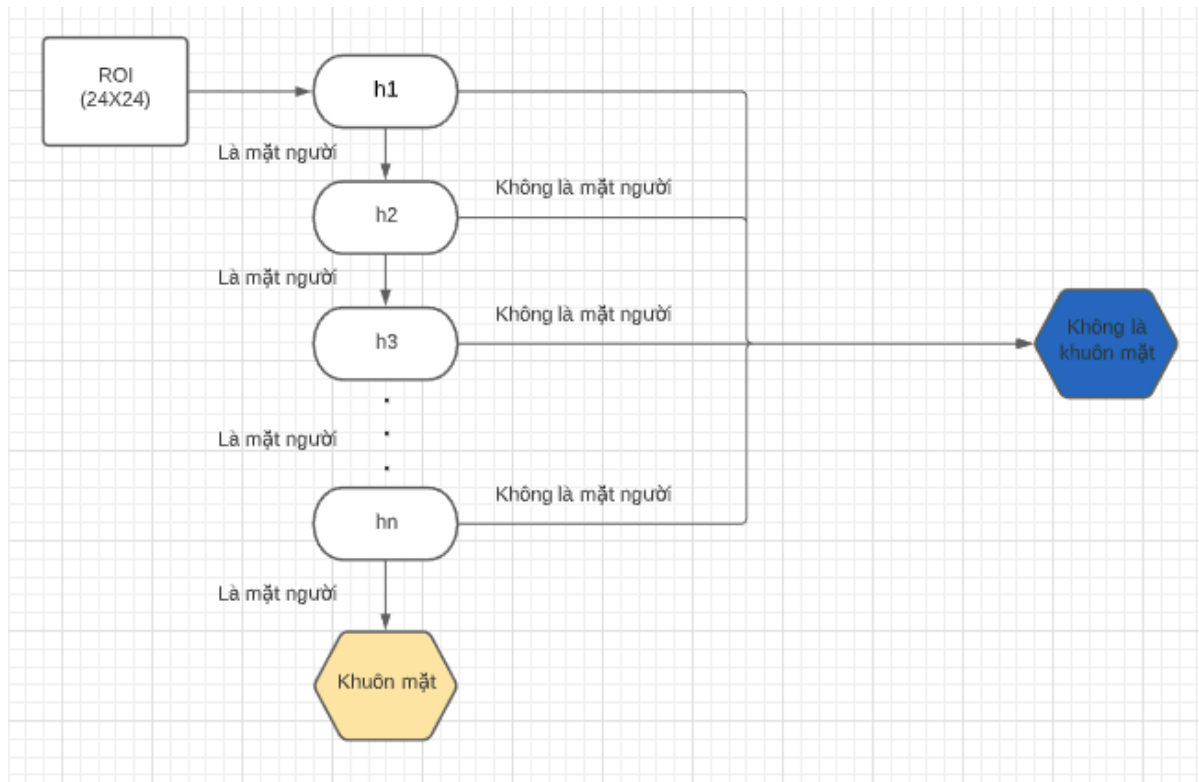
$\Delta$

$\Delta$

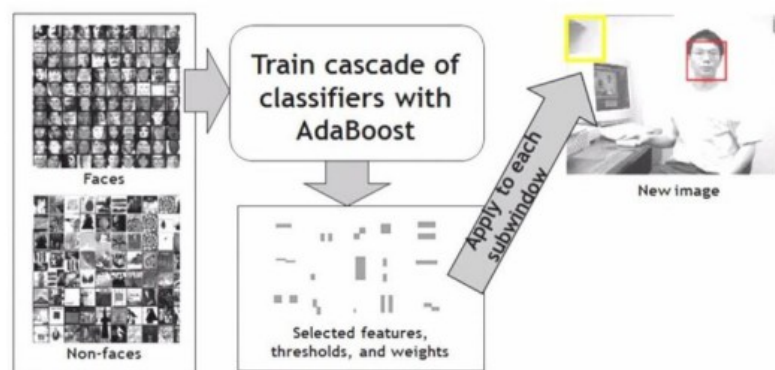
*Strong classifier Weak classifier*

Mặc dù một ảnh có thể chứa một hay nhiều khuôn mặt nhưng số lượng vật không phải là khuôn mặt vẫn tương đối nhiều. Do đó thuật toán cần tập trung vào việc loại bỏ những đối tượng không phải là mặt người một cách nhanh chóng.

Một bộ phân lớp *Cascade(cascade classifier)* được đưa vào sử dụng với mục đích mau chóng phân loại các đối tượng không phải mặt người. Các đặc trưng được nhóm vào vài “stage”. Mỗi “stage” gồm một số các đặc trưng, được sử dụng để xác định một cửa sổ có phải là khuôn mặt hay không.



Hình 2.6 Các bước phân loại khuôn mặt sử dụng Cascade.



Hình 2.7 Kết quả của Viola Jones Face Detection

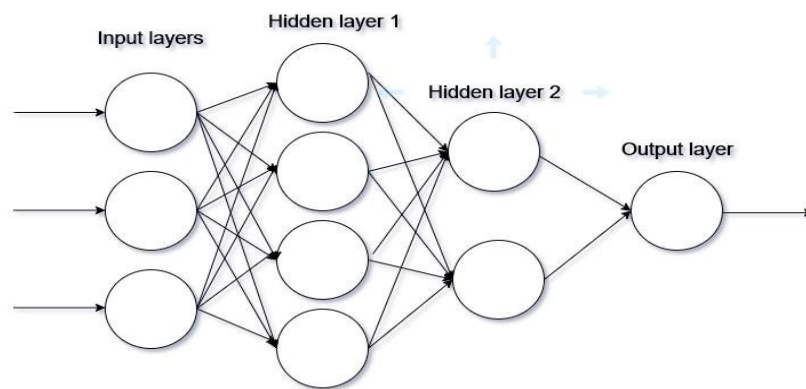
Bên dưới là kết quả được công bố trên bài báo *Viola Jones Face Detection*. Thời điểm công bố kết quả nghiên cứu, thuật toán được giới chuyên gia, học giả đánh giá cao về độ chính xác cũng như tốc độ. Và được kì vọng là một bước đột phá mới trong lĩnh vực thị giác máy tính.

## CHƯƠNG 3. MỘT SỐ KIẾN THỨC LIÊN QUAN

### 3.1 Convolution neural network

#### 3.1.1 Giới thiệu tổng quan

*Neural* là hay *neuron*(*nơ-ron*) là một thuật ngữ chỉ bộ phận tế bào hệ thần kinh của con người và những những loài động vật có khả năng suy nghĩ. Chúng là một thành phần quan trọng để cấu tạo nên bộ não. Để giúp con người và động vật có khả năng suy nghĩ, nhận thức, nhận biết, ra quyết định,..... *Network* là mạng cấu trúc đồ thị. vì thế *neural network* là hệ thống tính toán lấy cảm hứng từ sự hoạt động của các *nơ-ron* trong hệ thống thần kinh. Kiến trúc của một mạng *nơ-ron* sẽ gồm nhiều *nơ-ron* được kết nối với nhau. Và về cơ bản nó được chia thành 3 lớp chính: lớp đầu vào(*Input layer*), lớp ở giữa(*hidden layer*), các lớp đầu ra(*output layer*). Dưới đây là một mô hình đơn giản về mạng *nơ-ron*.



Hình 3.1 Ví dụ về một mạng *neural network*

Trong một kiến trúc của một *nơ-ron* ta có thể xây dựng nhiều lớp ẩn nhưng chỉ có một lớp đầu vào và một lớp đầu ra. Trong mạng thì có các *node*, mỗi *node* này chứa một hàm kích hoạt có thể khác giống hoặc khác nhau. Tuy nhiên trong thực tế người ta sử dụng chung một hàm kích hoạt để tối thiểu hóa độ phức tạp tính toán.

#### 3.1.2 Convolution Neural Network

*CNN* là một trong những mô hình *deeplearning* tiên tiến giúp xây dựng hệ thống nhận dạng, phân loại với tốc độ nhanh và độ chính xác cao trong xử lý ảnh. Những ứng dụng thực tiễn của nó là áp dụng vào trong các bài toán nhận dạng đối tượng, phân loại,... trong một ảnh. Tuy nhiên phương pháp này lại có một số nhược điểm là phải xây dựng một tập cơ sở dữ liệu lớn thì mới đạt được độ chính

xác cao và đồng thời việc xử lý khối dữ liệu này đòi hỏi phải nhanh và chính xác.

Về bản chất một mạng *CNN* là một kiến trúc mạng tuần tự(truyền thẳng), một tập hợp các lớp chồng lên nhau trong đó kiến trúc gồm nhiều thành phần(các *Layer*) được ghép nối với nhau theo cấu trúc: lớp tích chập(*Convolutional*), *Pooling*, *Relu*, và *Fully Connected*. Trong đó các lớp *Convolution* sử dụng các hàm *nonlinear* như *Relu*, *Tanh*,... để kích hoạt các trọng số trong các *Node*. Những layer sau khi được tính toán thông qua một hàm kích hoạt thì sẽ cho ra một chuỗi các thông tin, kể đó những thông tin này sẽ được truyền đến các lớp phía dưới. Trong mô hình mạng truyền ngược(*feedforward neural network*) thì mỗi *neural* đầu vào(*input node*) cho mỗi đầu ra trong các lớp tiếp theo. Mô hình này gọi là mô hình mạng kết nối đầy đủ(*fully connected layer*) hay mạng toàn vẹn(*affine layer*).

Cơ chế trong hoạt động trong kiến trúc *CNN* thì hoàn toàn trái ngược. Sự liên kết giữa các layer thông qua phép tích chập *Convolution*. Những layer kế tiếp là đầu ra của phép *Convolution* trên layer trước đó, nhờ đó hình thành những mối liên kết, kết cấu nội bộ. Như vậy mỗi neuron ở mỗi lớp kế tiếp sinh ra từ kết quả của filter áp dụng lên một vùng ảnh cục bộ của *neuron* trước đó.

### 3.1.2.1 Convolution Layer

Trong mạng *CNN*, lớp tích chập chính là những *hidden layer*, và mỗi lớp tích chập là một tập các *feature map* và mỗi này là một bản *scan* của hình ảnh đầu vào, được trích xuất ra thành các *feature map* đặc trưng cụ thể. Việc thực hiện phụ thuộc vào bộ lọc và nhân.

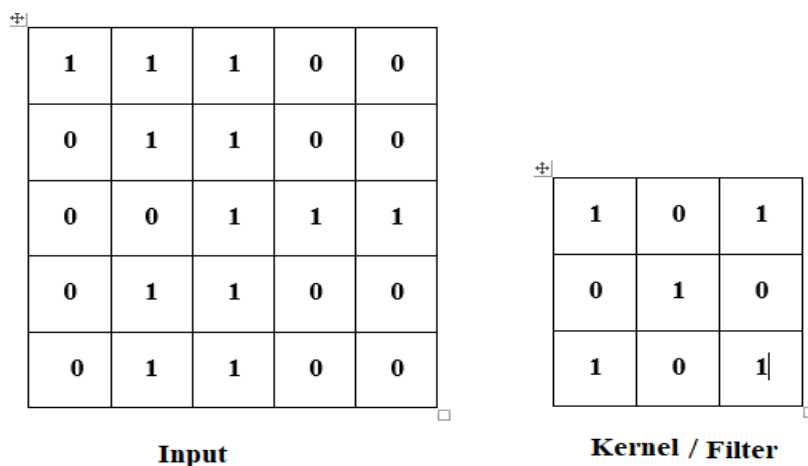
Công thức tính tích chập giữa hàm ảnh  $f(x, y)$  và bộ lọc  $k(x, y)$  (kích thước  $m \times n$ ):

$$k(x, y) \star f(x, y) = \sum_{u=-m/2}^{m/2} \sum_{v=-n/2}^{n/2} k(u, v) f(x - u, y - v)$$

Hình 3.2 Công thức tính tích chập

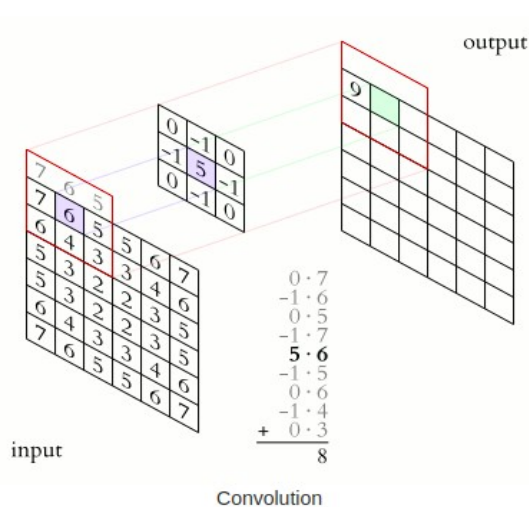
Thành phần không thể thiếu của phép tích chập là ma trận *kernel*(bộ lọc). Điểm *neo(anchor point)* của *kernel* sẽ quyết định vùng ma trận tương ứng trên hình ảnh để tích chập, thông thường *anchor point* sẽ được chọn là tâm của *kernel*. Giá trị của mỗi phần tử trên *kernel* được xem như là hệ số tổ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

Phép tích chập được hình dung bằng việc thực hiện dịch chuyển ma trận *kernel* lần lượt qua tất cả các điểm ảnh trong ảnh, bắt đầu từ góc bên trái trên cùng của ảnh. Và đặt *anchor point* tương ứng tại điểm ảnh đang xét. Ở mỗi lần dịch chuyển, thực hiện tính toán kết quả mới cho điểm ảnh đang xét bằng công thức tích chập. Sau đó, nó đưa qua hàm kích hoạt thu được một con số cụ thể. Sau quá trình đó ta thu được một ma trận, nó chính là *feature map*.



Hình 3.3 Ma trận đầu vào và kernel

Sau quá trình quét qua ảnh thông qua *kernel* thực hiện đồng thời các phép toán như đã nói trên ta thu được giá trị tại các *feature map*.

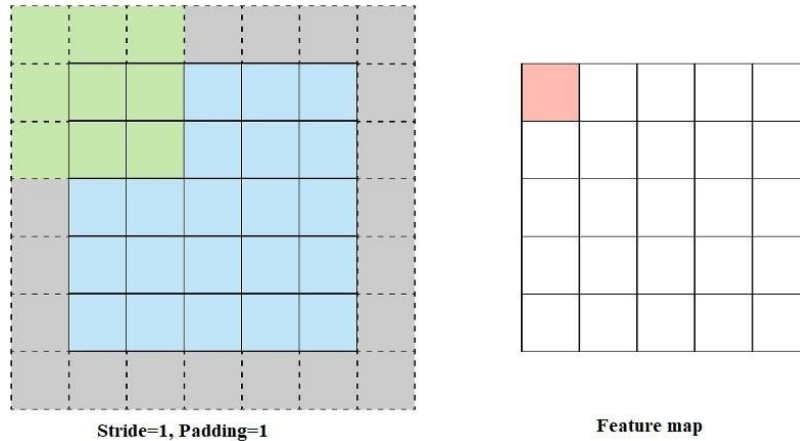


Hình 3.4 Ma trận  $Input \times Kernel$  và Feature map.

Sau quá trình quét qua ảnh thông qua *kernel* thực hiện đồng thời các phép toán như đã nói trên ta thu được giá trị tại các *feature map*. Tổng quát hóa, khi ta nhân tích chập ma trận *input* kích thước  $n \times n$  vào bộ lọc kích thước  $f \times f$ , ta thu được một ma trận kích thước:

$$(n - f + 1) \times (n - f + 1).$$

Mỗi lần sử dụng phép tích chập, kích thước *feature map* bị giảm xuống. Chúng ta có thể sử dụng padding để giữ nguyên kích thước của *feature map* so với ban đầu. Khi ta thực hiện phép điều chỉnh *padding=p*, tức là ta đã thêm *p* ô bao bọc xung quanh các cạnh của ma trận *input*.



Hình 3.5 Ví dụ với *stride=1* và *padding=1*

Ngoài tác dụng giữ nguyên kích thước ban đầu của *feature map*. *Padding* còn giúp tránh, giảm việc mất mát thông tin tại các vùng lân cận ảnh. Bởi đôi khi những thông tin ở vùng lân cận của cạnh lại là những thông tin quan trọng. Công thức tổng quát khi sử dụng với *padding=p*, *stride=s* được tính theo công thức:

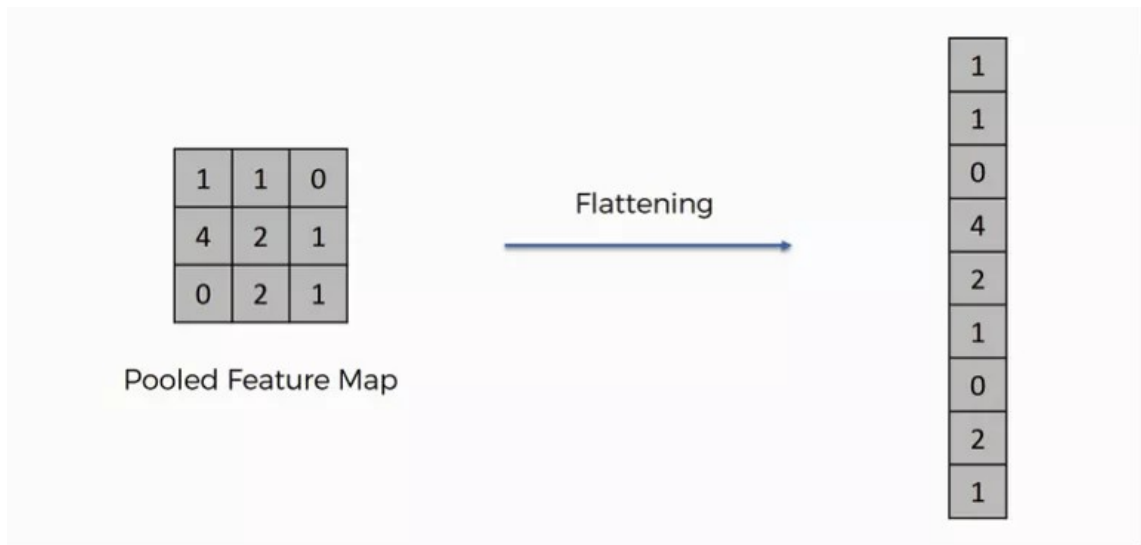
$$\left(\frac{n - f + 2p}{s} + 1\right) \times \left(\frac{n - f + 2p}{s} + 1\right)$$

Hình 3.6 Công thức tổng quát tính kích thước *Feature map*

### 3.1.2.2. Fully-Connected

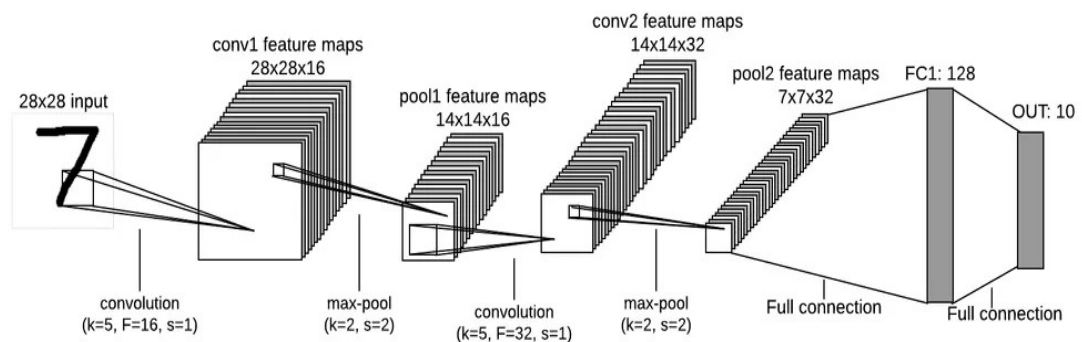
Lớp *Fully-Connected* là lớp mà các *nơ-ron* liên kết đầy đủ với nhau. Lớp này cũng tương tự như một *fully-connected* trong mạng *nơ-ron* nhân tạo. Sau khi ảnh được truyền qua nhiều *convolution layer* và *pooling layer* thì *model* đã học được tương đối các đặc điểm của ảnh( ví dụ như mắt, mũi, khung mặt, đường nét, góc, cạnh,..) thì *tensor* của *output layer* cuối cùng sẽ được chuyển về dạng một *vector* kích.





Hình 3.7 Quá trình chuyển Feature map thành vector

Sau đó ta dùng các *fully-connected layer* để kết hợp các đặc điểm của ảnh để ra được *output* của *model*.



Hình 3.8 Ví dụ về mô hình Convolution neural network

## 3.2 Một số vấn đề liên quan

### 3.2.1 Những tiêu chí đánh giá mô hình

Trong những năm gần đây với sự bùng nổ về các phương pháp tiếp cận bài toán nhận dạng khuôn mặt. Việc so sánh đánh giá các mô hình thông thường được các chuyên gia và giới nghiên cứu dựa trên những tiêu chí sau:

- **Tốc độ(Speed)**

Tốc độ là chi phí về thời gian liên quan đến quá trình thiết kết và sử dụng mô hình.

- **Sức mạnh(Robustness)**

Sức mạnh là khả năng phù hợp khái quát hóa. Khi mô hình có thể dự đoán đúng với các mẫu dữ liệu khó, chứa nhiễu hay bị mất một phần thông tin.

- **Độ đo(Metrics)**

Độ đo là những con số đưa ra dựa trên cách đánh giá về khả năng của mô hình cho biết độ chính xác khi mẫu vào thử nghiệm. Trong bài toán phân loại người ta thường sử dụng *Confusion matrix* hay *Error matrix*.

*Confusion matrix* là một ma trận tổng quát thể hiện kết quả phân loại chính xác và kết quả phân loại không chính xác được tạo bởi mô hình phân loại bằng cách so sánh với giá trị của biến mục tiêu (biến phân loại) của dữ liệu test. Ma trận  $N \times N$  với  $N$  là tổng số giá trị của biến mục tiêu (số *class/nhóm* của biến phân loại). Bảng dưới đây là *Confusion matrix* tổng quát.

		Predicted Category		
		0	1	Total
Actual category	0	True negatives: Predicted 0 Actually 0	False positives: Predicted 1 Actually 0	Total actually negative
	1	False negatives: Predicted 0 Actually 1	True positives: Predicted 1 Actually 1	Total actually positive
	Total	Total Predicted negative	Total Predicted positive	Grand total

Hình 3.9 *Confusion matrix* tổng quát.

- **Khả năng mở rộng (Scalability)**

Khả năng mở rộng là khả năng mà mô hình có thể xây dựng hiệu quả với các miền dữ liệu khác nhau.

- **Tính hiểu được (Interpretability)**

Tính hiểu được là mức độ chi tiết của thông tin được cung cấp bởi mô hình.

- **Tính đơn giản (Simplicity)**

Tính đơn giản là mức độ đơn giản, kích thước nhẹ của mô hình.

### 3.2.2 Vanishing Gradient, Exploding Gradient

Một khó khăn mà chúng ta sẽ phải đối mặt khi đào tạo những mô hình deep learning là *Vanishing Gradient* và *Exploding Gradient*. Trong một thời gian dài trước kia, hai vấn đề này từng là một trở ngại lớn cho việc xây dựng và phát triển các mô hình học sâu phức tạp.

Khi đào tạo một mạng *nơ-ron* sâu với phương pháp học dựa trên *gradient*

và lan truyền ngược(*backward propagation*) chúng ta xây dựng và tìm các đạo hàm riêng bằng cách duyệt mạng từ layer cuối cùng( $\hat{y}$ ) đến những *layer* ban đầu. Sử dụng quy tắc chuỗi tuần tự, các lớp nằm ở càng sâu trong mạng sẽ trải qua quá trình nhân liên tục các ma trận để tính đạo hàm của chúng. Trong một mạng gồm  $n$  lớp ẩn,  $n$  đạo hàm sẽ được nhân với nhau. Nếu các đạo hàm lớn thì gradient sẽ tăng theo cấp số nhân khi chúng ta truyền xuống các layer ở phía dưới và khi đủ lớn chúng ta sẽ bắt gặp vấn đề *Exploding Gradient*. Còn khi khác đạo hàm nhỏ thì các gradient sẽ giảm theo cấp số nhân, khi đủ nhỏ sẽ xảy ra hiện tượng *Vanishing Gradient*.

Trong trường hợp *Exploding Gradient*, sự phát triển nhanh của đạo hàm dẫn đến mô hình không ổn định và không có khả năng học tập hiệu quả, sự thay đổi lớn trong trọng số của mô hình tạo ra mạng lưới không ổn định, mà ở các giá trị cực trị thì trọng số trở nên quá lớn. Dẫn đến các giá trị trọng số không thể cập nhật được nữa.

Mặt khác, việc đạo hàm nhỏ dần rồi biến mất *Vanishing Gradient* dẫn đến một mô hình không có khả năng học thông tin chi tiết, có ý nghĩa vì vốn các trọng số và *bias* của các layer ban đầu vốn có xu hướng tìm hiểu các tính năng cốt lõi của đối tượng trong ảnh( đầu vào), sẽ không được cập nhật một cách hiệu quả. Trong trường hợp xấu nhất, gradient sẽ là 0, đến lượt nó sẽ dừng lại và không tiếp tục đào tạo nữa.

#### 3.2.2.1 Exploding Gradient

Có một số phương pháp mà ta có thể sử dụng để xác định liệu xem mô hình của mình có đang gặp vấn đề *Exploding Gradient*:

- Mô hình không học được nhiều về dữ liệu đào tạo do đó dẫn đến *loss* nhỏ.
- Mô hình có những thay đổi lớn về *loss* trong mỗi lần cập nhật do sự không ổn định của mô hình
- Trong quá trình đào tạo các *loss* sẽ là *NaN*

Khi đối mặt với vấn đề này, để xác nhận xem vấn đề có phải là do *Exploding Gradient* hay không có một số dấu hiệu ta có thể quan sát thấy được:

- Trọng số của mô hình phát triển theo cấp số nhân và trở nên rất lớn.
- Trọng số có thể trở thành *NaN* trong giai đoạn đào tạo

#### 3.2.2.2 Vanishing Gradient

Có một số cách thức để phát hiện vấn đề *Vanishing Gradient*:

- Mô hình sẽ cải thiện rất chậm trong giai đoạn đào tạo và cũng có thể việc đào tạo dừng lại rất sớm, nghĩa là bất kì đào tạo nào nữa cũng không cải thiện mô hình.
- Các trọng số gần lớp đầu ra của mô hình ta sẽ thấy chúng có sự thay đổi rõ rệt hơn so với những lớp gần đầu vào sẽ không thay đổi nhiều.
- Trọng số của mô hình sẽ thu nhỏ theo cấp số nhân và trở nên rất nhỏ khi huấn luyện.
- Trọng số mô hình trở về 0 trong giai đoạn huấn luyện.

### 3.2.2.3 Giải pháp

Có nhiều cách để tiếp cận cũng như giải quyết 2 vấn đề này. Dưới đây là một số cách tiếp cận.

- *Giảm số lượng các layer*  
 Đây là giải pháp có thể được sử dụng trong cả hai trường hợp. Bằng cách giảm số lượng lớp trong mạng để loại bỏ độ phức tạp của mô hình, vì có nhiều lớp làm tăng độ phức tạp, làm mô hình biểu diễn các ánh xạ một cách phức tạp
- *Gradient Clipping*  
 Phương pháp này được sử dụng để giải quyết vấn đề *Exploding Gradients*. Bằng cách kiểm tra và giới hạn kích thước của các gradient trong quá trình đào tạo.
- *Khởi tạo trọng số*  
 Lựa chọn khởi tạo các trọng số ban đầu giúp khi đào tạo mô hình chúng ta có thể tránh được hai vấn đề trên. Nhưng giải pháp này không giải quyết được hoàn toàn vấn đề này

## CHƯƠNG 4. XÂY DỰNG ỨNG DỤNG

### 4.1 Thực hiện phát hiện khuôn mặt

#### 4.1.1 Histogram of Oriented Gradients

##### 4.1.1.1 Giới thiệu chung

Hiện nay có rất nhiều các phương pháp khác nhau trong thị giác máy tính cho phép chúng ta có thể phát hiện các vật thể từ ảnh đầu vào. Các phương pháp hiện đại ứng dụng *deep learning* làm cốt lõi có thể kể đến như *YoLo*, *SSD*, *Faster RCNN*,.... Nhược điểm của những phương pháp tiếp cận này là cần đến những kiến trúc máy tính đủ mạnh để có thể đào tạo cũng như giải quyết vấn đề.

Vậy câu hỏi đặt ra ở đây là trước khi thế giới biết đến các mô hình học sâu thì con người đã dùng cách gì để xử lý vấn đề này. *HOG*(*histogram of oriented gradient*) được biết đến là một trong các phương pháp cổ điển để giải quyết vấn đề này. Mặc dù không thể bằng các phương pháp hiện đại về tốc độ cũng như độ chính xác khi phát hiện vật thể nhưng việc *HOG* lại vô cùng hoàn hảo để áp dụng vào cho những hệ thống vừa và nhỏ.

Thuật toán này sẽ tạo ra các bộ mô tả đặc trưng(*feature descriptor*) nhằm mục đích phát hiện vật thể(*object detection*). Từ một bức ảnh, ta sẽ lấy ra 2 ma trận quan trọng giúp lưu trữ thông tin của ảnh đó là độ lớn gradient(*gradient magnitude*) và phương của gradient(*gradient orientation*). Bằng việc kết hợp hai thông tin này vào một biểu đồ phân phối *histogram*, trong đó độ lớn gradient được điểm theo các nhóm khối chồng(*bins*) của phương gradient. Cuối cùng ta sẽ thu được vector đặc trưng *HOG* đại diện cho *histogram*. Nhìn chung, thuật toán trong thực tế còn hoạt động phức tạp hơn khi vector *HOG* sẽ được tính trên từng vùng cục bộ như mạng *CNN* và sau đó sử dụng phép chuẩn hóa để đồng nhất. Cuối cùng ta thu được một vector tổng hợp từ các vector trên vùng cục bộ.

##### 4.1.1.2 Mô tả chung về thuật toán

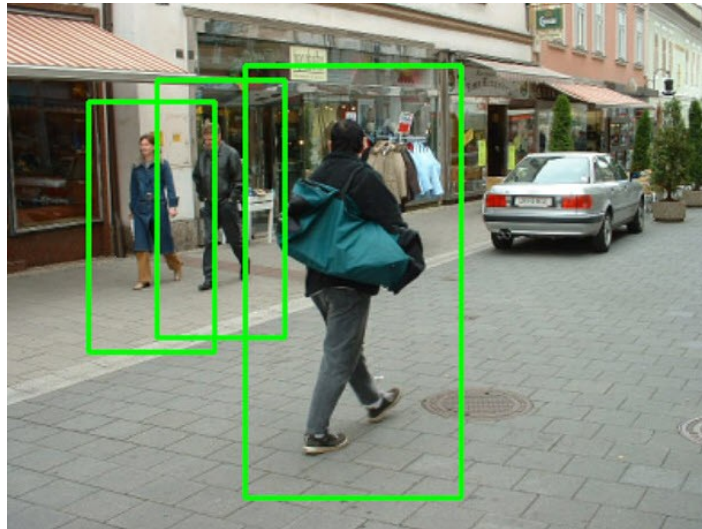
Điểm cốt lõi trong nguyên lý hoạt động của *HOG* đó là cấu trúc của một vật thể được mô tả thông qua hai ma trận đó là ma trận độ lớn gradient(*gradient magnitude*) và ma trận phương của gradient(*gradient direction*). Việc tạo ra hai ma trận trên dựa trên việc xác định các vùng cục bộ liên kề hoặc xếp chồng lên nhau. Một vùng cục bộ bao gồm nhiều ô cục bộ( trong thuật toán *HOG* là 4) có kích thước là  $8 \times 8$  pixels. Sau đó, một biểu đồ histogram thống kê độ lớn gradient được tính toán. Sau đó ta sẽ nối liền 4 vector histogram ứng với mỗi ô thành một vector tổng hợp. Vector tổng hợp đó người ta gọi là bộ mô tả *HOG*(*Hog description*). Để cải thiện độ chính xác, mỗi giá trị của vector histogram trên vùng cục bộ sẽ được chuẩn hóa theo norm chuẩn bậc 2 hoặc 1 nhằm tạo ra sự bất biến tốt hơn đối với những thay đổi trong chiếu sáng và đổ bóng.

Bộ mô tả *HOG* có một vài điểm nổi bật. Vì nó hoạt động trên các ô cục bộ, nó bất biến đối với các phép biến đổi hình học, thay đổi độ sáng. Hơn nữa, như 2 tác giả đã đề cập, khi sử dụng chuẩn hóa trên vùng cục bộ sẽ cho phép chuyển động cơ thể của người đi bộ được loại bỏ miễn là họ duy trì được tư thế thẳng đứng. Do đó bộ *HOG* đặc biệt phù hợp để phát hiện người trong ảnh.

#### 4.1.1.3 Ứng dụng của HOG

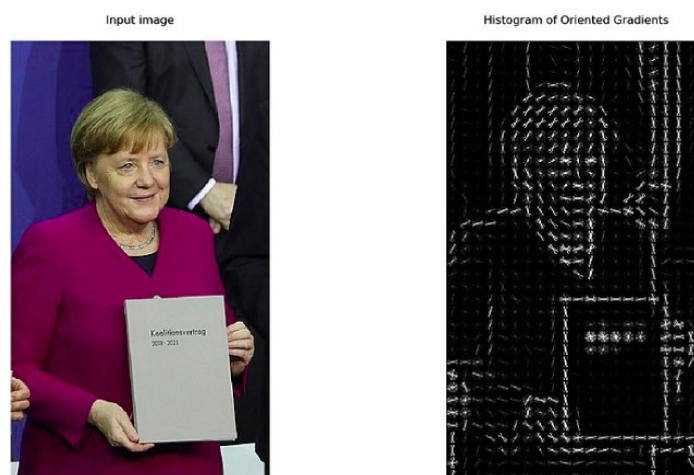
Một số ứng dụng cụ thể của *HOG* trong thực tế và đã mang lại hiệu quả cao có thể kể đến như:

- **Nhận diện người:** Tác giả đã có thí nghiệm công bố về độ hiệu quả của thuật toán trong việc phát hiện người. Thí nghiệm này được tác giả giới thiệu cũng với giải thích chi tiết về thuật toán được giới thiệu trong bài báo *Histograms of Oriented Gradient for Human Detection* của *Dalal* và *Trigg*. *HOG* có thể phát hiện được một hoặc nhiều người đi bộ trên cùng một bức hình.



Hình 4.1 *HOG* trong ứng dụng phát hiện người đi bộ

- **Nhận diện khuôn mặt:** Thông thường khi nhắc đến tác vụ nhận diện khuôn mặt chúng ta có thể nghĩ ngay đến *Haar Cascade Classifier*. Tuy nhiên cũng không thể phủ nhận độ hiệu quả của *HOG* trong việc xử lý tác vụ này. Bởi vì *HOG* có khả năng biểu diễn các đường nét chính của khuôn mặt dựa trên phương và độ lớn của gradient thông qua các vector trên từng cell như mô tả hình bên dưới:

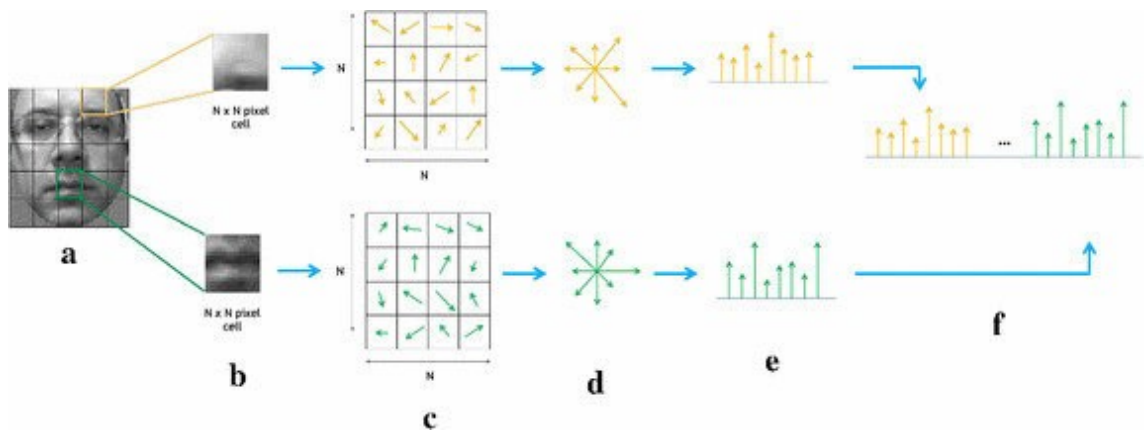


Hình 4.2 Trực quan các tính năng sau khi sử dụng *HOG*.

- **Phát hiện các vật thể khác:** Trong thực tế ta cũng có thể sử dụng *HOG* để

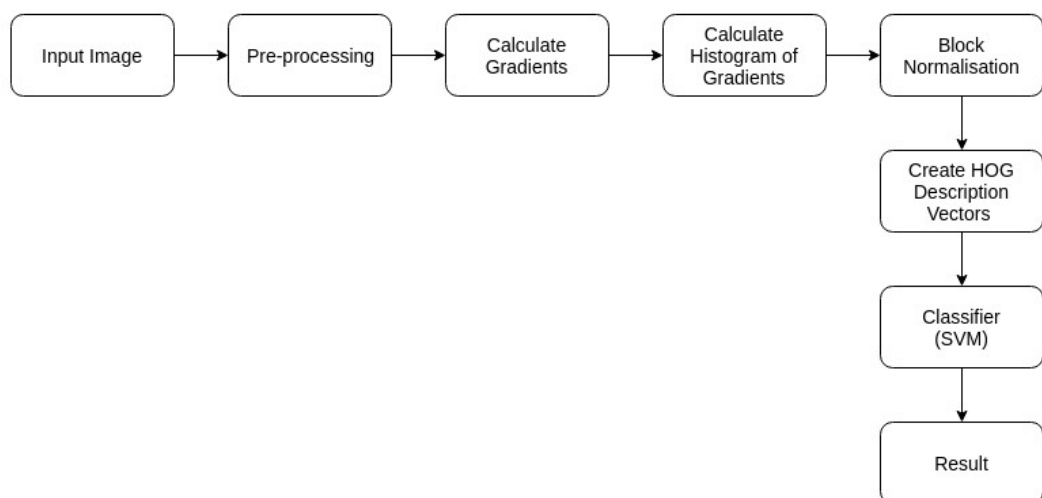
phát hiện vật thể như động vật, đồ vật,...Thậm chí ta còn có thể sử dụng để nhận dạng ảnh động trên Video.

- **Trích xuất tính năng:** Trong thực tế với bài toán phân loại, để có thể áp dụng các mô hình học sâu ta cần phải xây dựng một cơ sở dữ liệu đủ lớn thì mới đạt được hiệu quả tốt. Việc tạo ra các bộ dữ liệu lớn đó trong thực tế thì tốn nhiều chi phí trong việc thu thập, lưu trữ. Và hơn thế nữa việc đào tạo những mô hình đó cũng mất rất nhiều thời gian. Khi đó sử dụng *HOG* sẽ đem lại hiệu quả tốt hơn.



Hình 4.3 HOG trong việc trích xuất tính năng

#### 4.1.2 Quy trình phát hiện khuôn mặt



Hình 4.4 Quy trình phát hiện đối tượng sử dụng phương pháp Histogram of Oriented Gradient(HOG)

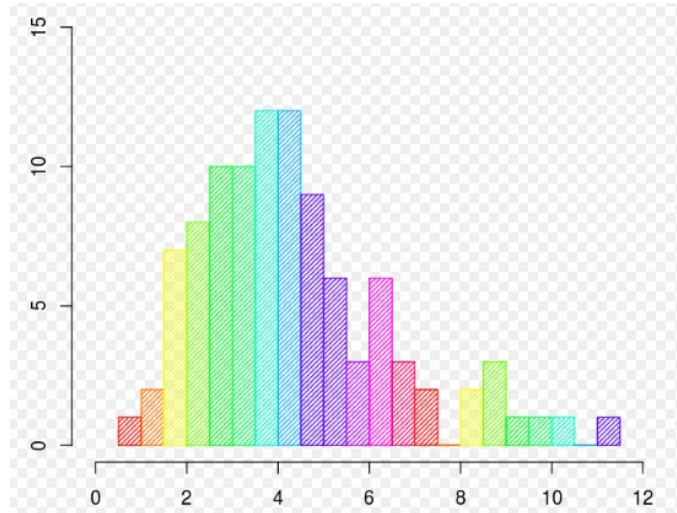
Dựa vào sơ đồ bên trên ta cũng đã có thể hình dung được từng bước quan trọng trong quá trình phát hiện khuôn mặt trên hình ảnh. Bằng việc chuyển dữ liệu đầu vào thành các bộ miêu tả *HOG*(hog descriptions vector) sau đó đưa các bộ vector mô tả này qua trình phân loại *SVM* ta có thể phát hiện được những đối tượng liên qua. Tuy nhiên trong khi triển khai thuật toán chi tiết thuật toán phức tạp hơn nhiều.



### 4.1.3 Chi tiết thuật toán HOG

#### 4.1.3.1 Một số thuật ngữ chính

- **Biểu đồ(histogram):** Là biểu đồ biểu diễn phân phối của các cường độ màu sắc theo khoảng giá trị.



Hình 2.8 Một ví dụ về histogram

- **Ô cục bộ(local cell):** Trong thuật toán HOG, một hình ảnh được chia thành nhiều cells bởi một lưới ô vuông. Mỗi cell được gọi là một ô cục bộ.
- **Vùng cục bộ(local portion):** là một block được trích xuất ra từ ô vuông trên hình ảnh.
- **Chuẩn hoá cục bộ(local normalization):** phép chuẩn hoá được thực hiện trên một vùng cục bộ. Thường là chia cho norm chuẩn bậc 2 hoặc bậc 1. Mục đích của việc chuẩn hóa là duy trì tính đồng nhất của các giá trị cường độ màu sắc về cùng một phân phối.
- **Gradient:** Là đạo hàm của vector cường độ màu sắc giúp phát hiện hướng di chuyển của các đối tượng trong hình ảnh.
- **Bộ miêu tả đặc trưng(feature description):** Là một phép biến đổi dữ liệu đầu vào thành các kiểu đặc trưng giúp ích cho phân loại hoặc nhận diện vật thể. Các phương pháp có thể kể đến như HOG, SUFT, SHIFT.
- **Cường độ Gradient(gradient magnitude):** Là chiều dài của vector gradient theo phương x, y. Biểu diễn phân phối histogram của vector này theo vector phương gradient sẽ thu được vector mô tả đặc trưng của HOG. Được tính theo công thức bên dưới với  $G_x$ ,  $G_y$  lần lượt là giá trị gradient lần lượt theo phương x và y của hình ảnh.

$$|G| = \sqrt{G_x^2 + G_y^2}$$

- **Phương Gradient(gradient direction):** Là độ lớn góc giữa vector gradient x, y giúp xác định phương thay đổi cường độ màu sắc hay chính là phương đổ bóng của hình ảnh. Với  $G_x$ ,  $G_y$  lần lượt là giá trị gradient lần lượt theo phương x

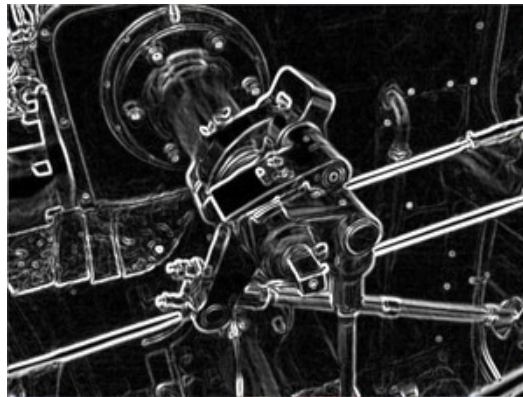


- và y của hình ảnh.

$$\theta = \arctan\left(\frac{G_y}{G_x}\right)$$

#### 4.1.3.2 Tính toán Gradient

Phương pháp phổ biến nhất để tính toán các giá trị gradient là áp dụng một mặt nạ đạo hàm rời rạc (*discrete derivative mask*) theo một hoặc cả hai chiều ngang và dọc. Cụ thể, phương pháp sẽ lọc ma trận cường độ ảnh với các bộ lọc như *Sobel mask* hoặc *Scharr*.



Hình 4.5 Một ảnh sau khi áp dụng Sobel mask

Để áp dụng bộ lọc *sobel*, ta sử dụng phép tính tích chập của *kernel* kích thước 3x3 được thực hiện với hình ảnh ban đầu. Nếu chúng ta kí hiệu *I* là ma trận gốc của  $G_x$ ,  $G_y$  là 2 ma trận ảnh mà mỗi điểm trên nó lần lượt là đạo hàm theo trục x và trục y. Chúng ta có thể tính toán được *kernel* như sau:

- Đạo hàm theo chiều ngang:

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * I$$

- Đạo hàm theo chiều dọc:

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * I$$

Kí hiệu  $*$  tương tự như phép tính tích chập giữa bộ lọc bên trái và ảnh đầu vào bên phải.

Giá trị độ lớn của gradient (*gradient magnitude*) và phương hướng gradient (*gradient direction*) có thể được tạo từ 2 đạo hàm  $G_x$ ,  $G_y$  theo công thức ở phần mô tả về thuật ngữ bên trên.



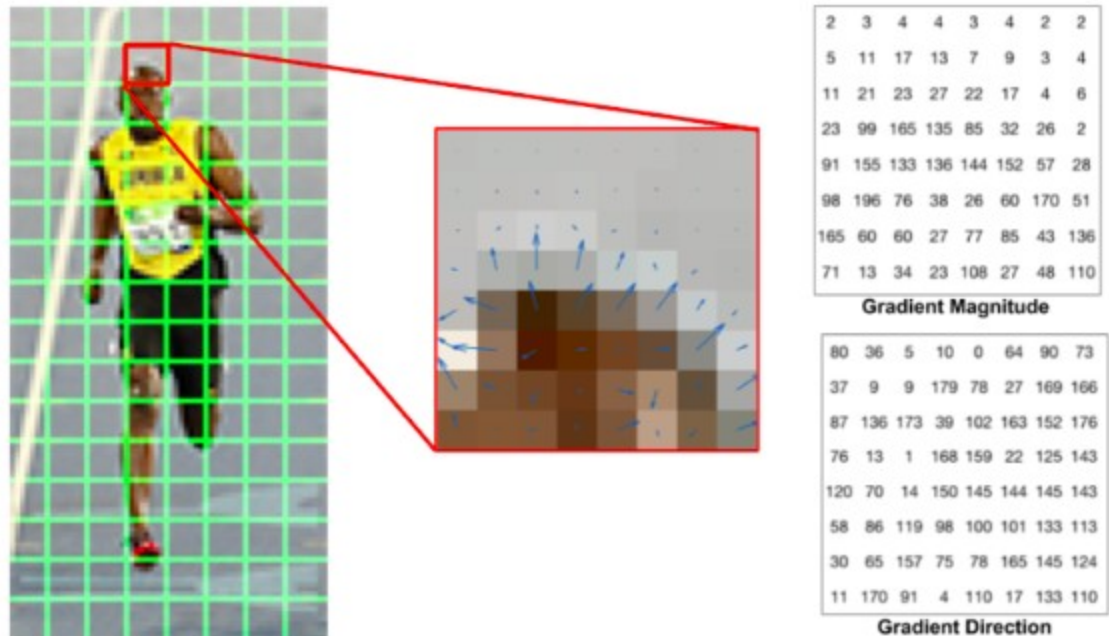
Hình 4.6 Kết quả của một ảnh sau khi thực hiện phép tính Gradient

#### 4.1.3.3 Các bước tính HOG

Đặc trưng của mỗi bức ảnh được biểu diễn thông qua 2 thông số là mức độ thay đổi cường độ màu sắc và hướng thay đổi cường độ màu sắc. Do đó chúng ta cần tạo ra một bộ mô tả đặc trưng (*feature descriptor*) sao cho biến đổi bức ảnh thành một vector mà thể hiện được cả 2 thông tin này.

Việc đầu tiên ta chia hình ảnh thành một lưới các ô vuông mà mỗi một ô có kích thước  $8 \times 8$  pixels. Như vậy chúng ta có tổng cộng 64 ô pixels tương ứng với một ô. Trên mỗi ô trong 64 pixels ta cần tính ra 2 tham số đó là độ lớn gradient (*gradient magnitude*) và phương gradient (*gradient direction*). Như vậy tổng cộng chúng ta  $8 \times 8 \times 2 = 128$  giá trị cần tính bao gồm 64 giá trị độ lớn gradient, và 64 giá trị phương gradient.

Trên mỗi ô chúng ta thực hiện tính đạo hàm thông qua bộ lọc Sobel để thu được 2 ma trận bên độ lớn về phương, và cường độ của gradient.



Hình 4.7 Hình ảnh đối tượng được chia thành các ô. Sau đó thực hiện tính đạo hàm trên từng ô.

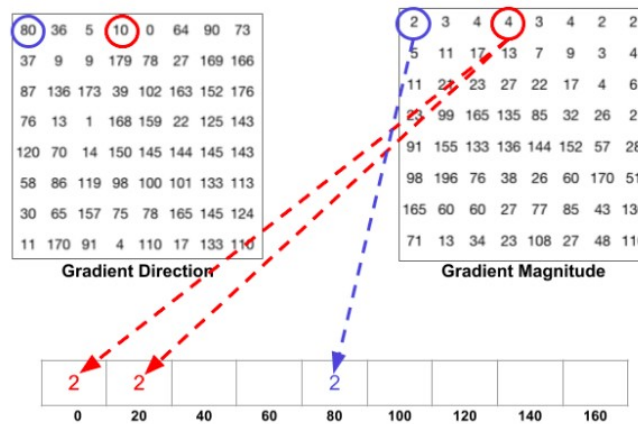
Sau quá trình trên ta sẽ biến đổi các ma trận thành các *vector histogram*. Theo các bước sau :

- **Ánh xạ độ lớn gradient vào các bins tương ứng của phương gradient:**

Xếp các giá trị phương gradient theo thứ tự từ nhỏ đến lớn và chia chúng vào 9 bins. Độ lớn của phương gradient sẽ nằm trong khoảng  $[0, 180]$  nên mỗi bin sẽ có độ giá là 20 như hình bên dưới.

Mỗi một phương gradient sẽ ghép cặp với một độ lớn gradient ở cùng vị trí tọa độ. khi biết được phương gradient thuộc bins nào trong vector *bin*, ta sẽ điền vào giá trị cường độ gradient của chính bins đó.

Chẳng hạn trong hình bên dưới có 2 ô được khoanh tròn viền xanh tương ứng với phương gradient là 80 và độ lớn gradient là 2. Khi đổ tại vector *bins* của HOG, phương gradient bằng 80 sẽ rơi vào vị trí thứ 5 nên tại ô này chúng ta điền giá trị 2 ứng với độ lớn gradient.



Hình 4.8 Ảnh xạ độ lớn gradients vào các bins.

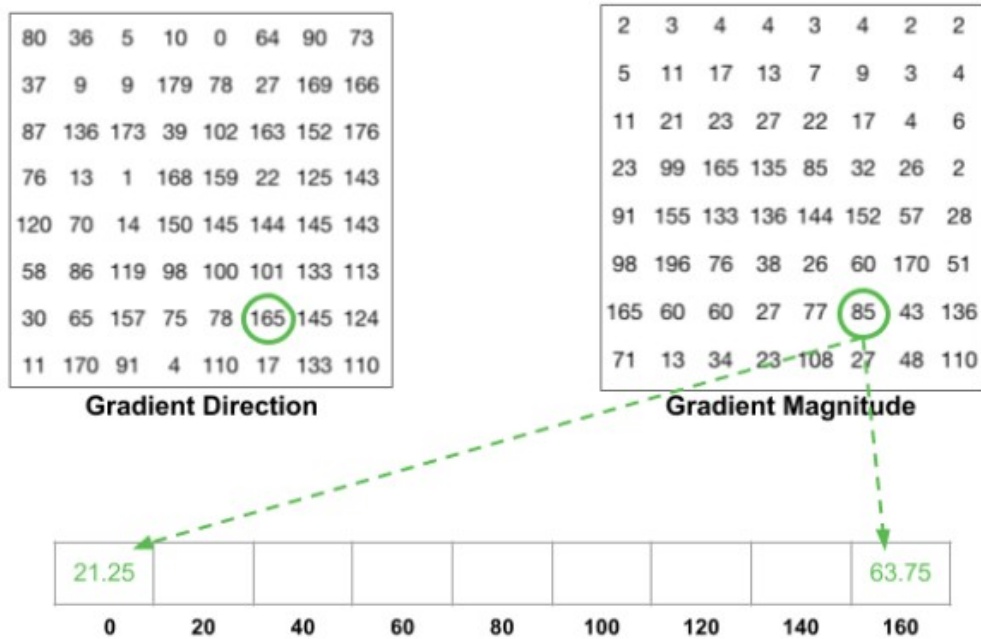
Tại mỗi đầu mút là các giá trị chia hết cho độ rộng của một bin(chẳng hạn 0, 20, 40,... là những đầu mút bin). Trong trường hợp độ lớn phương gradients không rơi vào các đầu mút, ta sẽ sử dụng *linear interpolation* để phân chia độ lớn gradient về 2 bins liên kế mà giá trị phương gradient rơi vào. Ví dụ: Giá trị phương gradients rơi vào khoảng giữa bin thứ  $(l - 1)$  và bin thứ  $l$ . Khi đó tại 2 bins  $(l - 1)$  và  $l$  được điền vào giá trị cường độ theo công thức *interpolation*.

- Giá trị tại bins  $l - 1$  :

$$x_{l-1} = \frac{(x_1 - x)}{x_1 - x_0} * y$$

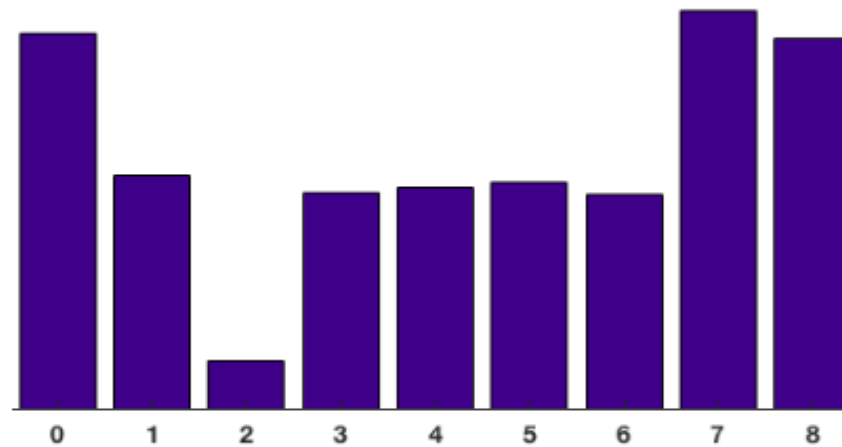
- Giá trị tại bins  $l$ :

$$x_l = \frac{(x - x_0)}{x_1 - x_0} * y$$



Hình 4.9 Ví dụ về việc ánh xạ các giá trị cường độ Gradient tương ứng.

Sau khi tính tổng tất cả các độ lớn gradient thuộc cùng 1 *bins* của vector ta thu được biểu đồ *Histogram of Gradient* như bên dưới:



Hình 4.10 Biểu đồ *Histogram of Gradient* gồm 9 bins tương ứng với mỗi ô vuông trong lưới.

- **Chuẩn hóa vector histogram theo block 16x16:** Vector histogram sẽ bị phụ thuộc vào cường độ của các *pixels* của một bức ảnh. Với những ảnh có cùng nội dung nhưng bức ảnh biến thể tối hơn được tạo thành từ ma trận ảnh gốc nhân 1/2. Khi giá trị vector histogram của ảnh gốc cũng sẽ gấp đôi vector histogram của ảnh biến thể. Chính vì vậy, chúng ta cần chuẩn hóa các vector histogram để những bức ảnh của cùng đối tượng được đồng nhất.

Chuẩn hóa norm bậc 2:

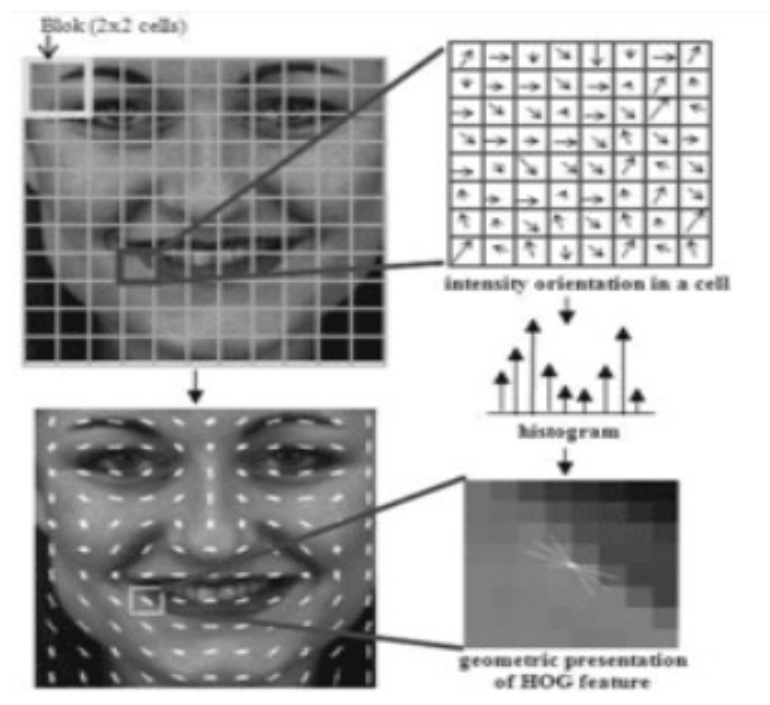
$$\|x\|_2 = \sqrt{\left(\sum_i x_i^2\right)} = \sqrt{x_1^2 + x_2^2 + \dots + x_i^2}$$

Trong đó  $x_i$  là các vector histogram tương ứng của các gradient.

- **Tính toán HOG feature vector:** Sau khi chuẩn hóa các *vector histogram*, chúng ta sẽ ghép nối các vector 1x36 này thành một vector lớn. Đây chính là vector *HOG* đại diện cho toàn bộ ảnh.

Ví dụ: Hình ảnh của chúng ta sẽ được chia thành lưới ô vuông kích thước  $16 \times 8$  (mỗi ô  $8 \times 8$ ). Quá trình tính toán *HOG* sẽ di chuyển 7 lượt theo chiều rộng và 15 lượt theo chiều cao. Như vậy tổng cộng có  $7 \times 15 = 105$  patches, mỗi patches tương ứng với một *vector histogram* 36 chiều. Do đó cuối cùng vector *HOG* sẽ có kích thước là  $105 \times 36 = 3780$  chiều. Đây là một vector có kích thước tương đối lớn nên có thể mô phỏng được đặc trưng của ảnh khá tốt.

Đối với mỗi ô trên lưới ô vuông, chúng ta có thể biểu diễn phân phối *HOG* bao gồm nhóm 9 vector chung gốc chiều dài bằng độ lớn gradient, và góc bằng phương gradient. Khi đó chiều của nhóm các vector sẽ tương đối giống với cấu trúc, đường nét của khuôn mặt như hình bên dưới.



Hình 4.11 Biểu diễn vector histogram trên các lưới ô vuông của hình ảnh gốc.

## 4.2 Thực hiện phân loại khuôn mặt

### 4.2.1 Mô hình VGG

#### 4.2.1.1 Giới thiệu

VGG được giới thiệu bởi nhóm *Visual Geometry* thuộc Đại học Oxford. Kiến trúc này tập trung vào khía cạnh làm tăng chiều sâu của mạng thông qua việc tạo ra các *block Conv* được xếp chồng lên nhau. Năm 2014, VGG được công bố kèm theo một số biến thể của nó: có cấu hình tương tự nhưng khác nhau về kích thước, kích thước được từ 11-19 layers. Nhưng hiệu năng tốt nhất là những phiên bản có 16 layers trở lên. VGG mặc dù có hiệu năng tốt nhất cuộc thi *ILSVRC* năm 2014 tuy nhiên nó không giành được chiến thắng. Nhưng chiến thắng cuộc thi năm đó là *GoogleNet* với thông số top-5 error rate ở mức 6.7% so

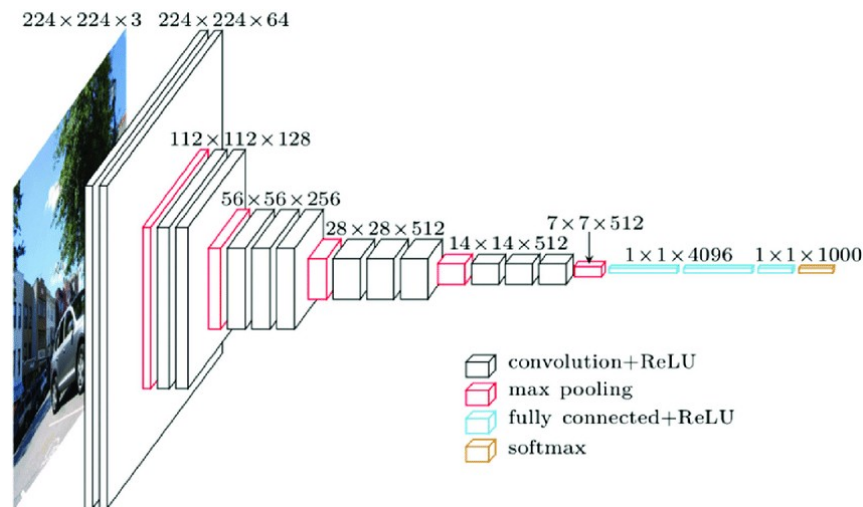
với 7.3% của VGG. Tuy nhiên, VGG vẫn có những điểm nổi bật mà ta không thể phủ nhận, hiện nay nó trở thành chuẩn thiết kế của các kiến trúc được triển khai trong tương lai.

#### 4.2.1.2 Chi tiết về mô hình

Một trong những điểm đặc biệt của VGG không thể không nhắc tới là việc giảm kích thước bộ lọc nhưng gia tăng chiều sâu. Như chúng ta biết việc giảm kích thước bộ lọc là tối cần thiết nếu muốn tăng chiều sâu, bởi lẽ một bộ lọc nhỏ chỉ ghi lại được một vùng nhỏ của ảnh. Có thể hình dung, một đặc tính sau khi trải qua 3 phép tính chập với *kernel*  $3 \times 3$  sẽ ghi nhận một vùng đầu vào có kích thước  $7 \times 7$ . Chú ý rằng sử dụng trực tiếp bộ lọc  $7 \times 7$  lên dữ liệu đầu vào cũng ghi nhận được những thuộc tính thị giá của một vùng đầu vào  $7 \times 7$ . Trong trường hợp thứ nhất, chúng ta sử dụng  $3 \times 3 \times 3 = 27$  tham số, trong khi đó chúng ta sử dụng  $7 \times 7 = 49$  tham số trong phương pháp thứ 2. Do đó, ta cần sử dụng nhiều tài nguyên của máy tính hơn. Không chỉ thế, khi sử dụng 3 tích chập liên tiếp ta thường ghi nhận được những đặc tính phức tạp và thú vị hơn việc sử dụng một lần tích chập  $7 \times 7$ . Việc sử dụng những bộ lọc có kích thước lớn sẽ khó có thể ghi nhận được những hình dạng phức tạp, nhỏ trong vùng ảnh.

Nhìn chung, chiều sâu lớn hơn sẽ hướng tới tính phi tuyến và tính chuẩn hóa cao hơn. Mạng sâu hơn sẽ có tính phi tuyến hơn bởi lẽ nó được tạo thành từ nhiều layers thông qua việc sử dụng hàm kích hoạt *ReLU*. Đồng thời tính chuẩn hóa hơn nằm ở việc tăng chiều sâu sẽ tạo một cấu trúc trên các layer thông qua việc sử dụng lặp đi lặp lại các thành phần tích chập. Như đã đề cập trên, những kiến trúc có độ sâu lớn hơn và giảm kích thước bộ lọc đòi hỏi ít tham số hơn. Sở dĩ có nhận định này bởi lẽ ham số ở mỗi layer có được bởi hình vuông mang kích thước của bộ *kernel*, trong khi số lượng tuyến tính lại phụ thuộc tuyến tính vào độ sâu. Do vậy mà hoàn toàn có thể giảm đáng kể số lượng tham số thông qua việc sử dụng bộ lọc nhỏ hơn, thay vào đó sử dụng những tham số này vào việc tăng độ sâu của mạng sẽ làm tăng năng lực phát hiện các đặc tính phức tạp của mô hình. Vì thế VGG luôn sử dụng *kernel* có kích thước là  $3 \times 3$  và *pooling*  $2 \times 2$ . Tích chập được tính có bước nhảy là 1 sử dụng bước đệm là 1. *Pooling* được tính với bước nhảy bằng 2.

Một điểm nhấn đáng chú ý nữa là VGG luôn tăng số lượng bộ lọc lên 2 lần sau mỗi lần thực hiện phép *Max-Pooling*. Tư tưởng này cũng được kế thừa bởi nhiều mô hình sau này như *Resnet*,...



Hình 4.12 Kiến trúc mô hình VGG16.

Tuy nhiên việc cấu hình mạng theo chiều sâu như vậy không phải là không có vấn đề, một trong số đó là nó sẽ làm tăng tính nhạy cảm của mạng với việc khởi tạo, sự thiếu tính ổn định. Vấn đề này có thể được giải quyết bằng cách sử dụng các mô hình huấn luyện trước, bằng cách này, người ta sử dụng hàng loạt huấn luyện với các kiến trúc nông hơn trước để mới thêm nhiều layers vào sau. Trên thực tế, việc tiền huấn luyện các mô hình không phải được thực hiện trên cơ sở một layer này với một layer khác đơn thuần. Thay vào đó, một tập 11 layer được huấn luyện trước. Những layer đã được huấn luyện này dùng để khởi tạo một tập các vector trong kiến trúc sâu hơn. VGG đạt được *top-5 error rate* ở mức 7.3% trong *ILSVRC*, trong một những kết quả cao tuy nhiên vẫn chưa phải vô địch.

Bên dưới là bảng cấu hình của mạng VGG. Trong số này, kiến trúc cột D chính là kiến trúc giành phần thắng ở *IISVRC*. Theo sau các bước thực hiện tất cả là quá trình *Relu*, *Max-pooling* layer kí hiệu là *M*, phản hồi chuẩn hóa cục bộ là *LRN*. *Softmax* layer kí hiệu là *S*, *FC4906* kí hiệu của layer liên kết toàn bộ với 4096 units.

Các layers đã được kết nối mang những liên kết đơn dày đặc giữa 4096 neuron và 7x7x512 volumns. Chúng ta sẽ nhận thấy rằng hầu hết các tham số của mạng neural được ẩn đi trong các mối liên kết này.



Name:	A	A-LRN	B	C	D	E
# Layers	11	11	13	16	16	19
	C3D64	C3D64	C3D64	C3D64	C3D64	C3D64
		LRN	C3D64	C3D64	C3D64	C3D64
	M	M	M	M	M	M
	C3D128	C3D128	C3D128	C3D128	C3D128	C3D128
			C3D128	C3D128	C3D128	C3D128
	M	M	M	M	M	M
	C3D256	C3D256	C3D256	C3D256	C3D256	C3D256
	C3D256	C3D256	C3D256	C3D256	C3D256	C3D256
				C1D256	C3D256	C3D256
						C3D256
	M	M	M	M	M	M
	C3D512	C3D512	C3D512	C3D512	C3D512	C3D512
	C3D512	C3D512	C3D512	C3D512	C3D512	C3D512
				C1D512	C3D512	C3D512
						C3D512
	M	M	M	M	M	M
	C3D512	C3D512	C3D512	C3D512	C3D512	C3D512
	C3D512	C3D512	C3D512	C3D512	C3D512	C3D512
				C1D512	C3D512	C3D512
						C3D512
	M	M	M	M	M	M
	FC4096	FC4096	FC4096	FC4096	FC4096	FC4096
	FC4096	FC4096	FC4096	FC4096	FC4096	FC4096
	FC1000	FC1000	FC1000	FC1000	FC1000	FC1000
	S	S	S	S	S	S

Hình 4.13 Những cấu hình khác nhau của VGG

Một ứng dụng khá hay được trình bày ở [A. Karpathy, J. Johnson, and L. Fei-Fei. *Stanford University Class CS321n: Convolutional neural networks for visual recognition*, 2016. <http://cs231n.github.io/>] cho thấy một trường hợp khi mà hầu hết các tham số và bộ nhớ của hàm kích hoạt được định vị. Cụ thể, phần lớn bộ nhớ chính cần thiết cho việc lưu trữ của hàm kích hoạt cũng như tính toán gradient trong các pha trước và sau được yêu cầu ở trong giai đoạn đầu của CNN. Đây là một điểm đặc trưng quan trọng bởi lẽ phần bộ nhớ cần cho một mini-batch được ước tính bởi kích cỡ của mini-batch đó. Lấy ví dụ, cần 93MB cho mỗi hình ảnh, với một *mini-batch* kích thước 128, chúng ta sẽ cần bộ nhớ vào khoảng 12GB. Trên thực tế, hầu hết các tham số được yêu cầu bởi layer liên kết toàn bộ ở quá trình cuối. Liên kết 7x7x512 layer với 4096 neuron sẽ cần tới  $7 \times 7 \times 512 \times 4096 = 102\,760\,448$  tham số. Tổng tham số ở tất cả các layer tính ra vào khoảng 138 000 000. Nhận thấy rằng 75% các tham số nằm ở 1 layer kết nối duy nhất. Thêm vào đó, hầu hết các tham số còn lại nằm ở 2 layer liên kết toàn bộ cuối cùng, những liên kết dày đặc chiếm khoảng 90% tham số trong mạng.

#### 4.2.2 Học chuyển tiếp(TrasferLearning)

Học chuyển tiếp là một phương pháp học máy mà trong đó, tái sử dụng mô



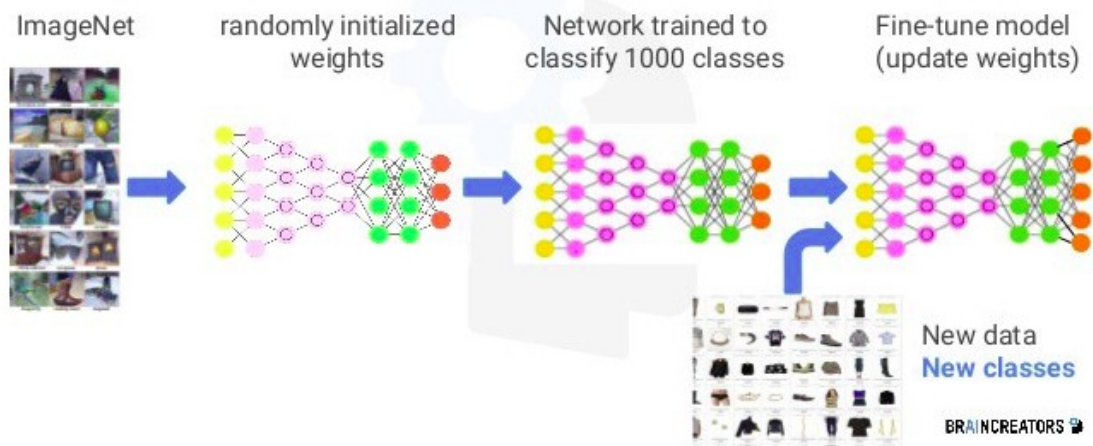
hình đã được triển khai nhằm mục đích xử lý một nhiệm vụ nào đó ở một nhiệm vụ khác. Đây là một phương pháp mang lại cách tiếp cận phổ biến và hiệu quả cao trong *deeplearning*. Nó giải quyết vấn đề thiếu dữ liệu, hệ thống máy tính không đủ mạnh để phục vụ cho cho trình đào tạo một mô hình trên một bộ dữ liệu lớn. Ta có thể tái một số mạng như: *VGG*, *ResNet*, *GoogLeNet*,....

Thông thường có hai cách để sử dụng một *pretrained network*: trích xuất tính năng(*feature extraction*) và tinh chỉnh lại model(*fine turning*).

#### 4.2.2.1 Transfer learning là gì

- Là một kỹ thuật học máy, nơi một mô hình đã được đào tạo về một nhiệm vụ được tái sử dụng trên một nhiệm vụ khác có liên quan.
- Là một tối ưu hóa cho phép giải quyết vấn đề về tốc độ và hiệu suất khi mô hình hóa nhiệm vụ thứ 2
- Liên quan đến các vấn đề xử lý đồng thời nhiều tác vụ trong *deeplearning* và *Concept Drift*(trượt khái niệm).

#### 4.2.2.2 Cách tiếp cận



Hình 4.14 Minh họa kỹ thuật Transfer Learning .

Trong thực xử lý các tác vụ dự đoán, phân loại có thể tiếp cận phương pháp này theo 2 cách:

- **Pre-trained Model:** Một *pre-trained model* được chọn từ các mô hình có sẵn. Hiện nay có nhiều tổ chức lớn *opensource* cho phép chúng ta có thể tiếp cận các mô hình trên các taaph dữ liệu lớn phù hợp với từng loại hình bài toán. Sau khi đã lựa chọn được mô hình phù hợp với bài toán tác vụ ta sử dụng nó như là một điểm(layer) khởi đầu cho mô hình trên nhiệm vụ thứ 2 cần quan tâm. Điều này có thể liên quan đến việc sử dụng tất cả hoặc một phần của mô hình, tùy thuộc vào kỹ thuật mô hình được sử dụng. Mô hình mình có thể cần phải được điều chỉnh hoặc tinh chỉnh trên dữ liệu input-output có sẵn cho nhiệm vụ quan tâm.
- **Develop Model:** Lựa chọn một mô hình có liên quan với sự phong phú của dữ liệu trong đó có một số mối quan hệ trong dữ liệu đầu vào, dữ liệu đầu ra hoặc các khái niệm đã học tổng quát ánh xạ từ dữ liệu đầu vào đến đầu ra. Sau đó

phát triển mô hình theo nhiệm vụ đầu tiên này. Mô hình phải tốt hơn để đảm bảo rằng một số tính năng học tập đã được thực hiện. Sau khi xử lý để mô hình nguồn phù hợp với yêu cầu nhiệm vụ thứ hai thì ta sử dụng nó như là một điểm bắt đầu trên mô hình nhiệm vụ thứ 2. Mô hình có thể cần phải được điều chỉnh hoặc tinh chỉnh trên tập dữ liệu để phù hợp với tác vụ cần quan tâm.

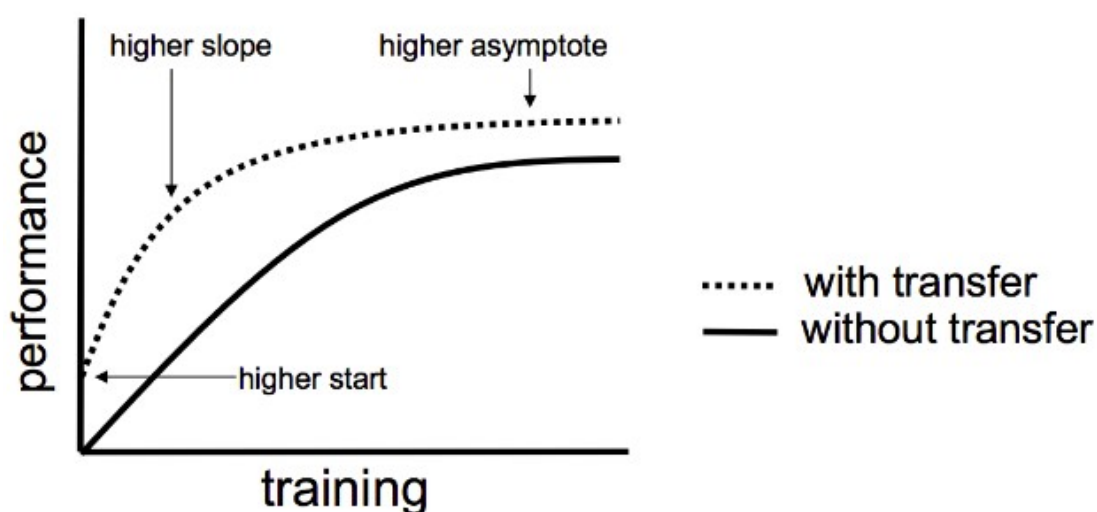
Một số mô hình deep learning có thể tham khảo như:

- **Với xử lý ảnh:** *Oxford VGG Model, Google Inception Model, Microsoft Resnet Model.*
- **Đối với xử lý ngôn ngữ:** *Google's word2vec Model, Stanford's GloVe Model.*

#### 4.2.2.3 Lợi ích và hạn chế

##### a. Lợi ích

- **Thời gian:** Việc sử dụng *Pre-trained Model* bản thân nó không chỉ giúp giảm thời gian và việc tạo ra một model mới để thực hiện một nhiệm vụ đặt ra mà dựa trên một *source tasks* sẵn có, mà còn giảm thời gian đào tạo một mô hình từ đầu vì các trọng số của mô hình nguồn đã có sẵn.
- **Hiệu quả:** *Pre-traine Model* đã cung cấp cho chúng ta một *accuracy* cao ngay từ đầu, do đó khi đào tạo trên nhiệm vụ mục tiêu thì mô hình của bạn sẽ tiếp tục tăng độ *accuracy* thay vì chúng ta phải khởi đầu từ một điểm có *accuracy* thấp hơn.



Hình 2.8 So sánh tương quan hiệu quả của model train từ đầu và khi sử dụng *pre-trained mode*.

##### a. Hạn chế

Đây là một kỹ thuật khó tiếp cận, nếu sai sót trong quá trình sử dụng *pre-trained model* hay thêm/bớt không đúng layer thì khi train, thì chúng ta sẽ không kiểm soát được độ chính xác của mô hình. Khi đó chúng ta phải kiểm tra lại toàn bộ mô hình. Đôi khi nó có thể dẫn chúng ta đến bế tắc.

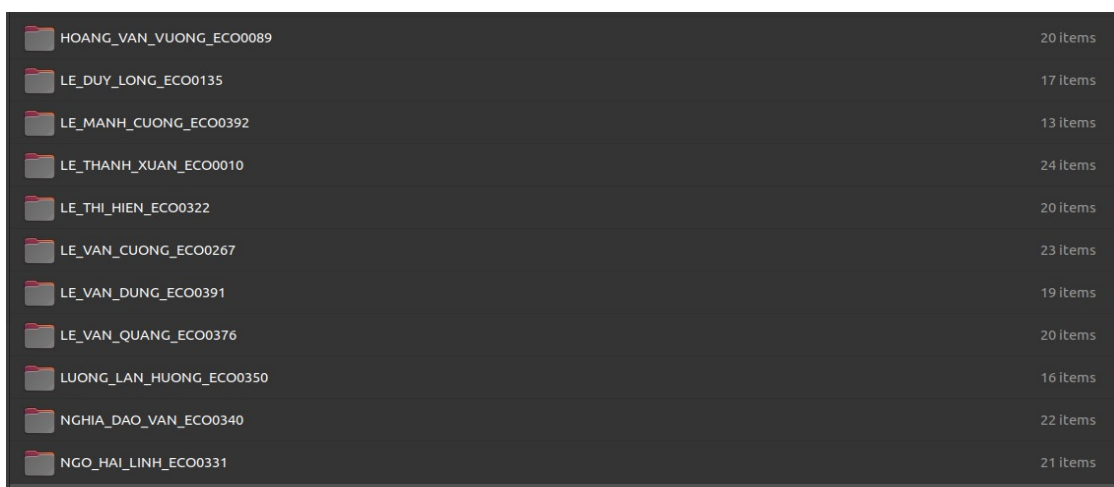
Sẽ gặp vấn đề lớn nếu chúng ta chọn mô hình *pre-trained* không đúng, không liên quan đến tác vụ cần xử lý. Không phải pretrain model nào cũng có thể

dùng để *transfer* và hướng đến nhiệm vụ mong muốn được. Ví dụ không nên dùng *pretrain model* dự đoán chữ viết tay cho tác vụ nhận diện khuôn mặt được.

### 4.3 Ứng dụng và thực nghiệm

#### 4.3.1 Bộ cơ sở dữ liệu

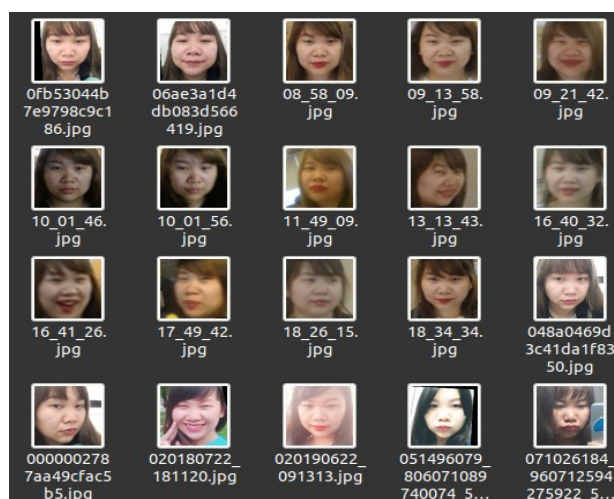
Cơ sở dữ liệu gồm một tập dữ liệu chứa ảnh chân dung của 87 nhân viên được chia thành 87 *thư mục*, mỗi thư mục sẽ chứa ảnh của một nhân viên. Mỗi thư mục chứa 15-25 *ảnh* của nhân viên tương ứng. Bộ dữ liệu này được sử dụng trong quá trình phân loại.



HOANG_VAN_VUONG_ECO0089	20 items
LE_DUY_LONG_ECO0135	17 items
LE_MANH_CUONG_ECO0392	13 items
LE_THANH_XUAN_ECO0010	24 items
LE_THI_HIEN_ECO0322	20 items
LE_VAN_CUONG_ECO0267	23 items
LE_VAN_DUNG_ECO0391	19 items
LE_VAN_QUANG_ECO0376	20 items
LUONG_LAN_HUONG_ECO0350	16 items
NGHIA_DAO_VAN_ECO0340	22 items
NGO_HAI_LINH_ECO0331	21 items

Hình 4.15 Bộ cơ sở dữ liệu 87 nhân viên .

Các ảnh được chụp từ một khoảng cách cố định, và có kích thước là 224x224 thang màu *RGB*, bộ ảnh được thu thập theo nhiều góc cạnh của khuôn mặt, nhiều biểu cảm khác nhau.

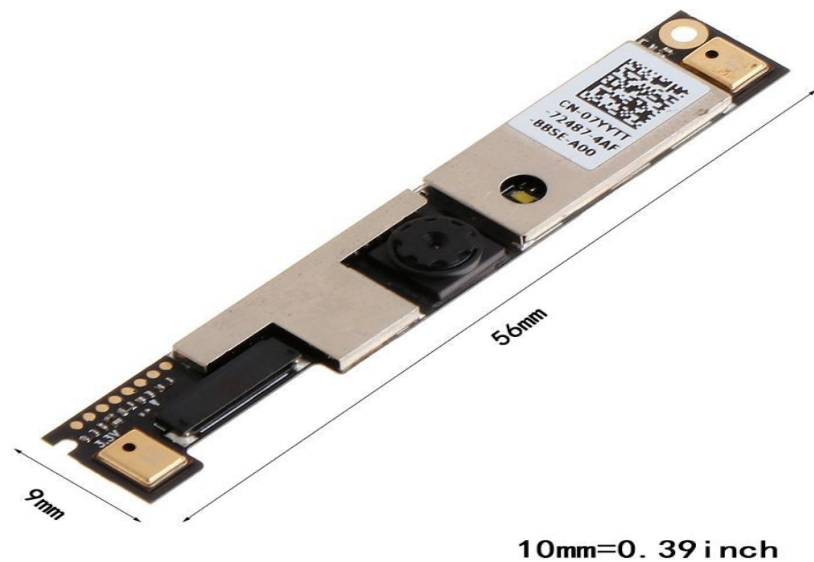


Hình 4.16 Ảnh chân dung khuôn mặt được chụp ở nhiều góc độ, biểu cảm khác nhau.

#### 4.3.2 Quá trình phát hiện khuôn mặt

Chương trình được xây dựng để có thể tương thích trên các thiết bị *camera*

cục bộ. Những kết quả dưới đây được thử nghiệm trên hệ thống *Webcam* của máy tính cá nhân.



Hình 4.17 Webcam laptop Dell E7 sử dụng cho bài toán .

Với việc thực hiện thuật toán *HOG* kết hợp với bộ phân loại *SVM* ta dễ dàng xây dựng một chương trình phát hiện khuôn mặt theo thời gian thực từ các việc ghi lại trực tiếp các thông qua *Webcam*.



Hình 4.18 Kết quả quá trình phát hiện khuôn mặt dựa trên thuật toán *HOG*.

Sau khi đã thực hiện xong công việc phát hiện khuôn mặt trên các frame được truyền vào từ camera cục bộ. Ta thu được ảnh chân dung của của người cần nhận diện.



Hình 4.19 Hình ảnh khuôn mặt thu được sau quá trình phát hiện khuôn mặt.

Những ảnh này có kích thước 224x224. Sau khi thu được ảnh có chứa khuôn mặt ta sẽ cho những ảnh này đi qua hệ thống phân loại.

#### 4.3.3 Hệ thống thống phân loại

Ở hệ thống này em đã sử dụng kiến trúc dựa trên mô hình của VGG16 để phân loại, nhưng do tập dữ liệu nhỏ, nên việc đào tạo mô hình từ nguồn dữ liệu này thì khó có thể đạt được hiệu quả cao. Ngoài ra nó còn có thể dẫn đến một số vấn đề thường gặp như *overfitting*. Để khắc phục điều này, em sử dụng kỹ thuật *transfer learning* để tích hợp vào hệ thống của mình.

- **Pre-trained model :** VGGFace được cung cấp bởi Đại học Oxford vào được xây dựa trên kiến trúc của VGG16.
- **Tập dữ liệu sử dụng của pre-trained model:** pre-trained model sử dụng bộ dữ liệu VGGFace. Đây là bộ dữ liệu vô cùng lớn về mặt người, bộ dữ liệu có sự đa dạng về độ tuổi, các góc độ khuôn mặt, biểu cảm, giới tính.... Bộ dữ liệu chứa 3.31 triệu ảnh với 9131 đối tượng khác nhau. Trung bình sẽ có 362 images trên một đối tượng.
- **Nhiệm vụ của mô hình Pre-trained model:** Nhiệm vụ chính của mô hình ảnh là giải quyết bài toán phân loại trên bộ dữ liệu lớn VGGFace Nhiệm vụ của này phù hợp với công việc chúng ta cần giải quyết.
- **Nhiệm vụ mục tiêu:** Bài toán chúng ta cần giải quyết ở đây là khi chúng ta nhận được ảnh chân dung khuôn mặt được truyền tới. Chúng ta cần xác định xem ảnh chân dung đó là của người nào hay nói cách khác khuôn mặt có trong ảnh là của ai.



Hình 4.20 Quá trình đưa ra đánh giá, dự đoán của .



Model: "vggface\_vgg16"

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 224, 224, 3)	0
conv1_1 (Conv2D)	(None, 224, 224, 64)	1792
conv1_2 (Conv2D)	(None, 224, 224, 64)	36928
pool1 (MaxPooling2D)	(None, 112, 112, 64)	0
conv2_1 (Conv2D)	(None, 112, 112, 128)	73856
conv2_2 (Conv2D)	(None, 112, 112, 128)	147584
pool2 (MaxPooling2D)	(None, 56, 56, 128)	0
conv3_1 (Conv2D)	(None, 56, 56, 256)	295168
conv3_2 (Conv2D)	(None, 56, 56, 256)	590880
conv3_3 (Conv2D)	(None, 56, 56, 256)	590880
pool3 (MaxPooling2D)	(None, 28, 28, 256)	0
conv4_1 (Conv2D)	(None, 28, 28, 512)	1180160
conv4_2 (Conv2D)	(None, 28, 28, 512)	2359808
conv4_3 (Conv2D)	(None, 28, 28, 512)	2359808
pool4 (MaxPooling2D)	(None, 14, 14, 512)	0
conv5_1 (Conv2D)	(None, 14, 14, 512)	2359808
conv5_2 (Conv2D)	(None, 14, 14, 512)	2359808
conv5_3 (Conv2D)	(None, 14, 14, 512)	2359808
pool5 (MaxPooling2D)	(None, 7, 7, 512)	0
global_average_pooling2d_1 (GlobalAveragePooling2D)	(None, 512)	0

Total params: 14,714,688  
Trainable params: 14,714,688  
Non-trainable params: 0

Hình 4.21 Kiến trúc mô hình pre-trained VGGFace.

Bên dưới đây là quá trình sử dụng đào tạo mô hình phân loại. Trong quá trình đào tạo tỷ lệ bộ dữ liệu *train- test* tương ứng là 7 : 3. Dữ liệu kiểm thử trong quá trình đào tạo sẽ lấy ngẫu nhiên từ bộ dữ liệu kiểm thử.

```

Train on 207 samples, validate on 90 samples
Epoch 1/1000
160/207 [=====>.....] - ETA: 0s - loss: 5.0295 - accuracy: 0.0750
Epoch 00001: val_accuracy improved from -inf to 0.18889, saving model to data/models/classifiers/best_weights.h5
207/207 [=====] - 2s 8ms/sample - loss: 4.9223 - accuracy: 0.0918 - val_loss: 3.1962 - val_accuracy: 0.1889
Epoch 2/1000
192/207 [=====>...] - ETA: 0s - loss: 2.3603 - accuracy: 0.4115
Epoch 00002: val_accuracy improved from 0.18889 to 0.56667, saving model to data/models/classifiers/best_weights.h5
207/207 [=====] - 0s 1ms/sample - loss: 2.3306 - accuracy: 0.4106 - val_loss: 1.6971 - val_accuracy: 0.5667
Epoch 3/1000
192/207 [=====>...] - ETA: 0s - loss: 1.0148 - accuracy: 0.7656
Epoch 00003: val_accuracy improved from 0.56667 to 0.74444, saving model to data/models/classifiers/best_weights.h5
207/207 [=====] - 0s 1ms/sample - loss: 0.9780 - accuracy: 0.7778 - val_loss: 1.0712 - val_accuracy: 0.7444
Epoch 4/1000
192/207 [=====>...] - ETA: 0s - loss: 0.4287 - accuracy: 0.9219
Epoch 00004: val_accuracy improved from 0.74444 to 0.85556, saving model to data/models/classifiers/best_weights.h5
207/207 [=====] - 0s 1ms/sample - loss: 0.4168 - accuracy: 0.9275 - val_loss: 0.7411 - val_accuracy: 0.8556
Epoch 5/1000
192/207 [=====>...] - ETA: 0s - loss: 0.1775 - accuracy: 0.9896
Epoch 00005: val_accuracy improved from 0.85556 to 0.91111, saving model to data/models/classifiers/best_weights.h5
207/207 [=====] - 0s 995us/sample - loss: 0.1787 - accuracy: 0.9903 - val_loss: 0.5452 - val_accuracy: 0.9111
Epoch 6/1000
128/207 [=====>.....] - ETA: 0s - loss: 0.1014 - accuracy: 0.9922
Epoch 00006: val_accuracy improved from 0.91111 to 0.92222, saving model to data/models/classifiers/best_weights.h5
207/207 [=====] - 0s 1ms/sample - loss: 0.0955 - accuracy: 0.9952 - val_loss: 0.4661 - val_accuracy: 0.9222

```

Hình 4.22 Quá trình huấn luyện trên tập dữ liệu thực .

Nhờ vào việc sử dụng *pre-trained model* ta đã giảm được tối đa thời gian đào tạo, và đạt được độ chính xác cũng như hiệu quả khá cao. Quá trình đào tạo chỉ mất 16 epoch để đạt được độ chính xác trung bình là 0.922.

Dữ liệu được truyền vào từ đến hệ thống phân loại sau khi qua hệ thống phát hiện khuôn mặt sẽ được xử lý là chuyển đến hệ thống phân loại. Ở đây mô hình sẽ đưa ra ra đánh giá, dự đoán. Kết quả sẽ thu được sẽ được hiển thị như hình bên

dưới.



Hình 4.23 Hình ảnh hiện thị kết quả phân loại.

Bên cạnh đó dữ liệu cũng được xử lý để cho ra thông tin tin liên quan bao gồm: Ngày làm việc, thời gian bắt đầu làm việc , thời gian nghỉ. Những dữ liệu này sau khi được xử lý sẽ được lưu trữ trên một hệ thống cơ sở dữ liệu.

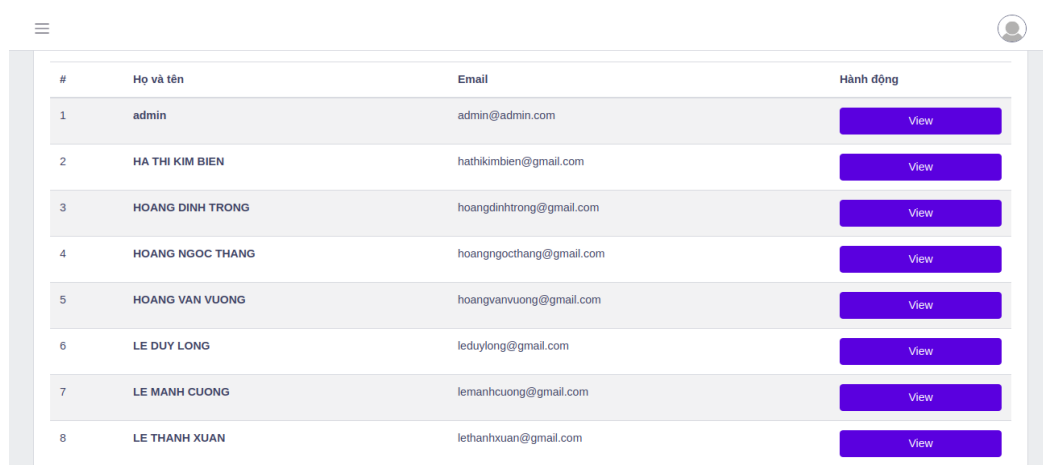
#### 4.3.4 Hệ thống thống Webservice

Kiểm tra, truy xuất thông tin được đơn giản hóa, thân thiện với người dùng hơn dưới dạng một *webserver*. Ở đây mỗi nhân viên và người quản trị hệ thống được cấp một tài khoản có thể sử dụng để kiểm tra thông tin của bản thân liên quan đến lịch sử làm việc của mình. Trong đó người quản trị hệ thống có thể theo dõi được toàn bộ lịch sử làm việc của toàn bộ nhân viên. Dữ liệu này sẽ được cập nhật liên tục.

A screenshot of a web-based login interface. The title 'Login' is at the top in a large, bold, dark font. Below it is the subtitle 'Sign In to your account' in a smaller, lighter font. There are two input fields: the first is labeled 'E-Mail Address' with a person icon to its left, and the second is labeled 'Password' with a lock icon to its left. Below these fields is a blue button with the text 'Login' in white.


Hình 4.24 Giao diện đăng nhập.









Bên dưới là giao diện chứa thông tin làm việc của nhân viên. Lịch sử làm việc được ghi nhận từ ngày 21/12 - 30/12 khi dự án được vận hành thực tế.



#	Họ và tên	Email	Hành động
1	admin	admin@admin.com	<a href="#">View</a>
2	HA THI KIM BIEN	hathikimbien@gmail.com	<a href="#">View</a>
3	HOANG DINH TRONG	hoangdinhtrong@gmail.com	<a href="#">View</a>
4	HOANG NGOC THANG	hoangngocthang@gmail.com	<a href="#">View</a>
5	HOANG VAN VUONG	hoangvanvuong@gmail.com	<a href="#">View</a>
6	LE DUY LONG	leduylong@gmail.com	<a href="#">View</a>
7	LE MANH CUONG	lemanhcuong@gmail.com	<a href="#">View</a>
8	LE THANH XUAN	lethanhxuan@gmail.com	<a href="#">View</a>

Hình 4.25 Giao diện hiển thị danh sách nhân viên .



1	2020-12-21	08:11:25	18:07:36			77.43%	76.82%
2	2020-12-22	08:11:25	17:23:43			93.85%	78.65%
3	2020-12-23	08:09:49	18:46:58			90.26%	93.85%
4	2020-12-24	08:22:34	17:42:56			86.6%	90.26%
5	2020-12-25	08:22:34	18:17:57			86.67%	82.54%
6	2020-12-29	08:07:36	18:46:58			97.24%	94.67%

Hình 4.26 Giao diện hiển thị thông tin về lịch sử làm việc của nhân viên.

#### 4.3.5 Đánh giá tổng quan về hệ thống.

Với kết quả thực nghiệm của hệ thống dựa trên các kết quả đã được ghi nhận, ta thấy rằng ở tác vụ phát hiện khuôn mặt kết quả vẫn chưa được tốt. Vẫn còn phụ thuộc nhiều các yếu tố như ánh sáng, ngoại cảnh của môi trường. Vấn đề này là vấn đề thường thấy trong việc sử dụng HOG trong tác vụ này.

Với việc phân loại thì đôi khi mô hình đã cho những dự đoán sai khi chúng ta đặt ngưỡng dưới dự đoán dưới 75%, và với trong số một số trường hợp phức tạp mô hình chưa đủ thông minh để đưa ra kết quả chính xác nhất. Hiệu quả giảm xuống khi thực hiện 1 số thay đổi: như người dùng đeo thêm kính, khẩu trang, các phụ kiện trang sức,...

Nói chung với các mô hình học sâu, dữ liệu đóng một vai trò quan trọng, ảnh hưởng lớn đến hiệu suất mô hình. Ở tác vụ phân loại, dữ liệu còn hạn chế, nên mô hình chưa đủ thông minh để có thể đạt hiệu suất cao trong những bức ảnh phức tạp có độ trừu tượng cao.



## CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

### 5.1 Kết luận

Đồ án đã trình bày việc sử dụng các phương pháp học sâu, và các kỹ thuật thị giác máy tính trong việc xây dựng hệ thống chấm công. Những nội dung đã đạt được trong quá trình thực hiện như sau:

- Tìm hiểu, nghiên cứu, cài đặt, thử nghiệm các phương pháp kĩ, kỹ thuật trong khuôn khổ đề tài. Để từ đó đã đưa ra được các so sánh, đánh giá, dựa trên những phương pháp đã sử dụng.
- Định hướng bản thân, tầm nhìn mới: Việc nghiên cứu, triển khai đồ án đã giúp em có cái nhìn sâu hơn về các hệ thống trí tuệ nhân tạo bây giờ, và tạo tiền đề cho các nghiên cứu, hướng phát triển mới trong tương lai.

Những khó khăn trong quá trình thực hiện đồ án:

- Việc sử dụng, cài đặt các kĩ thuật về thị giác máy tính còn tương đối khó khăn. Đặc biệt những kĩ thuật học sâu hiện nay thường đòi hỏi việc sử dụng hệ thống máy tính mạnh mẽ.
- Việc thu thập dữ liệu: Do tính chất phụ thuộc vào dữ liệu của các mô hình học sâu. Nên việc làm phong phú bộ dữ liệu để đào tạo trong thời gian ngắn gặp tương đối khó khăn.

### 5.2 Hướng phát triển trong tương lai

Qua những đánh giá dựa trên những kết quả thực nghiệm, chúng ta có thể nhìn thấy được nhiều hướng phát triển mới cho đề tài với mục đích là tăng cường tính hiệu quả:

- Tăng cường việc thu thập dữ liệu, cải thiện độ phong phú của dữ liệu.
- Thử nghiệm các phương pháp hiện đại trong việc phát hiện khuôn mặt.
- Cải tiến mô hình phân loại để đạt được độ chính xác cao hơn trên bộ dữ liệu phức tạp.
- Tinh chỉnh hệ thống để đạt được tốc độ tốt hơn.
- Phát triển thêm những tính năng mới trên hệ thống *webserver*.
- Nghiên cứu về các mô hình sinh dữ liệu. Tăng cường dữ liệu với mục đích làm dữ liệu phong phú hơn.
- Phát triển nhiều tính năng thú vị như: cho phép người dùng có thể checkin bằng smartphone. Và phát triển thêm nhiều ứng dụng xoay quanh việc tăng chất lượng hài lòng của người sử dụng.
- Biến đổi sản phẩm thành sản phẩm ứng dụng thực tiễn, cho các công ty, doanh nghiệp.

Bên cạnh đó, việc giải quyết những khó khăn đã đặt ra sẽ mở ra nhiều hướng giải quyết, hướng đi mới trong tương lai-một hướng đi dễ hơn.

## TÀI LIỆU THAM KHẢO

- [1] Z. Li, J.-i. Imai and M. Kaneko, "Robust face recognition using block-based bag of words.," Pattern Recognition (ICPR), 2010 20th International Conference on. IEEE, pp. 1285-1288, 2010.
- [2] L. Fei-Fei and P. Perona, "A bayesian hierarchical model for learning natural scene categories.," 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. Vol. 2, pp. 524-531, 2005.
- [3] C.-F. Tsai, "Bag-of-words representation in image annotation: A review.," ISRN Artificial Intelligence 2012, 2012.
- [4] L. Fei-Fei, "Stanford University, Computer Vision Lab," 6-9-2012. [Online]. Available: [http://vision.stanford.edu/teaching/cs231a\\_autumn1112/lecture/lecture14\\_intro\\_objrecog\\_bow\\_cs231a.pdf](http://vision.stanford.edu/teaching/cs231a_autumn1112/lecture/lecture14_intro_objrecog_bow_cs231a.pdf).
- [5] A. Martinez and R. Benavente, "The AR Face Database," CVC Technical Report #24, June 1998
- [6] S. Liao, A. K. Jain and S. Z. Li, "Partial face recognition: Alignment-free approach," IEEE Transactions on Pattern Analysis and Machine Intelligence 35.5, pp. 1193-1205, 2013.
- [7] K. Mikolajczyk, A. Zisserman and C. Schmid, "Shape recognition with edge-based features," British Machine Vision Conference (BMVC'03), vol. Vol. 2, pp. 779-788, 2003.
- [8] M. Dantone, J. Gall, G. Fanelli and L. Van Gool, "Real-time facial feature detection using conditional regression forests," Computer Vision and Pattern Recognition (CVPR), pp. 2578-2585, June, 2012.
- [9] S. Chopra, R. Hadsell and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. 1, pp. 539-546, 2005
- [10] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," Neural computation, vol. 1 (4), pp. 541-551, 1989.
- [11] C. Szegedy and e. al, "Going deeper with convolutions," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1-9, 2015.

## **PHỤ LỤC**

Thông tin cấu hình máy tính sử dụng cho quá trình thực hiện đồ án tốt nghiệp:

- Máy tính cá nhân:

Operating System: Ubuntu 20.04.

System Model: Dell Inspiron 3558

Processor: Intel ® Core <sup>™</sup> i5-5300U CPU @ 2.3GHz.

Memory: 8192MB RAM.

## KẾT QUẢ KIỂM TRÙNG TÀI LIỆU

### THÔNG TIN TÀI LIỆU

Email:	hoang.pd155650@sis.hust.edu.vn;hoang.pdang578@gmail.com
Tên file:	20201 datn phan dang hoang 20155650 1.9m.pdf
Thời gian nộp:	03/01/2021 21:12:52
Thời gian trả kết quả:	03/01/2021 21:13:42
Chế độ kiểm tra:	Việt - Việt
Số trang:	50
Số câu:	463
Số câu tương đồng:	4
Mức độ cảnh báo:	THẤP (cao: > 15%; trung bình: 2÷15%; thấp: < 2%)

### KẾT QUẢ KIỂM TRA TRÙNG LẬP

#### Độ tương đồng:

<b>0.86%</b>	<b>0.00%</b>	<b>0.86%</b>	<b>0.65%</b>
Trên tất cả tài liệu	Trên tài liệu nội bộ của trường	Trên tài liệu nội bộ của trường khác	Từ nguồn Internet

#### Nguồn trùng lặp nhiều nhất: 0.648%

Tài liệu hệ thống - nguyenthithuy\_luan\_van\_5909.txt