Trường Đại học Bách Khoa Hà Nội

Viện Công nghệ Thông Tin và Truyền Thông

Đồ án Tốt nghiệp Đại học

Phát hiện lỗi của camera thông qua phân tích ảnh mẫu bằng phương pháp học máy

Sinh viên thực hiện Phan Xuân Đức

Người hướng dẫn TS. Nguyễn Thị Thu Hương

Hà Nội, 12/2018

Lời cam kết

Họ và tên sinh viên: PHAN XUÂN ĐỨC

Điện thoại liên lạc : 0964.386.500. Email: phanxuanduc1996@gmail.com

Lớp: CN – CNTT 01 – Khóa : 59. Hệ đào tạo: Cử nhân công nghệ.

Tôi – *Phan Xuân Đức* – cam kết Đồ án Tốt nghiệp (ĐATN) là công trình nghiên cứu của bản thân tôi dưới sự hướng dẫn của *TS. Nguyễn Thị Thu Hương*. Các kết quả nêu trong đồ án là trung thực, là thành quả của riêng tôi, không sao chép theo bất kỳ công trình nào khác. Tất cả những tham khảo trong đồ án – bao gồm hình ảnh, bảng biểu, số liệu, và các câu từ trích dẫn – đều được ghi rõ ràng và đầy đủ nguồn gốc trong danh mục tài liệu tham khảo. Tôi xin hoàn toàn chịu trách nhiệm với dù chỉ một sao chép vi phạm quy chế của nhà trường.

Hà Nội, ngày 08 tháng 12 năm 2018

Tác giả ĐATN

Phan Xuân Đức

Lời cảm ơn

"Khi ta ở, chỉ là nơi đất ở,

Khi ta đi, đất đã hoá tâm hồn!"

Xin được dùng hai câu thơ trong bài "Tiếng hát con tàu" của nhà thơ Chế Lan Viên để mở đầu cho những dòng cảm xúc này.

Rong ruổi hết một đời sinh viên, còn lại trong mỗi chúng ta là một chút lưu luyến, một chút nuối tiếc. Tiếc nuối, không hẳn vì ta chưa thực sự cố gắng, mà ta tiếc nuối cho những gì đã qua, là dòng cảm xúc pha trộn của những bước đi cuối chặng đường. Những năm tháng trôi qua, điều quan trọng nhất ta học được ở giảng đường đại học không phải là kiến thức, mà là những lần đối đầu với khó khăn thử thách, hay cả những lần chúng ta đứng dậy sau vấp ngã. Những điều đó đã tôi luyện cho chúng ta một tinh thần thép, một sức mạnh phi thường để ta có thể đứng vững khi bước vào cuộc đời.

Bách Khoa trong tôi, là nơi có thầy cô tận tình bên trang giáo án, có những sinh viên cặm cụi trên thư viện, là nơi chứa những cảm giác trái ngược trong mỗi mùa thi. Hơn bốn năm trời sống giữa Bách Khoa, tôi cũng kịp dành cho mình những cảm xúc đặc biệt về Bách Khoa. Đó là những lần dành học bổng, những lần trượt môn, những lúc thâu đêm để hoàn thành bài tập hay cả những buổi chiều thư viện cùng đám bạn. Sau cùng, tất cả mọi thứ đủ vừa vặn để tạo nên một Bách Khoa đẹp nhất.

Tôi và các bạn, những thế hệ tương lai của đất nước, chúng ta không thể đi suốt quãng đường đại học mà không có những người đưa đò chèo lái. Suốt những năm tháng Bách Khoa ấy, tôi thực sự rất may mắn đã gặp được những người lái đò đầy tâm huyết nhất. Vậy nên tôi muốn được gửi lời cảm ơn sâu sắc đến tất cả thầy cô Trường Đại học Bách Khoa Hà Nội, thầy cô trong Viện Công nghệ thông tin & Truyền thông.

Đặc biệt nhất, tôi xin gửi những lời cảm ơn chân thành nhất tới cô giáo TS.Nguyễn Thị Thu Hương, giảng viên Bộ môn Khoa học máy tính. Nhờ được sự giúp đỡ tận tình của cô trong suốt những năm tháng Bách Khoa, tôi đã xin được những học bổng nhờ những bức thư giới thiệu của cô và quan trọng hơn cả, cô đã giúp tôi hoàn thành đồ án tốt nghiệp "Phát hiện lỗi của camera thông qua phân tích ảnh mẫu bằng phương pháp học máy" một cách tốt nhất. Xin kính chúc cô thêm nhiều sức khỏe để có thể tiếp tục truyền những điều tốt nhất cho những thế hệ sinh viên Bách Khoa trong tương lai.

Sinh viên

Phan Xuân Đức

Lời nói đầu

Trong cuộc sống hiện nay, smartphone là một thiết bị gần như không thể thiếu đối với mỗi người với sự tiện ích của nó. Trong đó, việc chụp ảnh để lưu lại những khoảnh khắc trong đời là một điều thiết yếu đối với mỗi người sử dụng. Để đáp ứng đầu ra tốt đối với camera, các nhà máy sản xuất điện thoại luôn xem công đoạn kiểm tra chất lượng của camera đạt tiêu chuẩn hay không là một vấn đề rất quan trọng. Trước đây, những việc kiểm tra được nhân viên sản xuất thực hiện thủ công : chụp ảnh, so sánh và đánh giá chất lượng bằng mắt. Việc kiểm tra thủ công như vậy rất phụ thuộc vào con người, môi trường làm việc. Công việc thủ công lặp này đi lặp lại như vậy gây sự nhàm chán và ảnh hưởng rất lớn đến kết quả đầu ra của sản phẩm. Để phát triển hơn trong thời đại công nghiệp 4.0, các nhà máy cần hướng đến việc kiểm thử tự động, nó vừa tiết kiệm nhân lực và mang lại tính hiệu quả cao.

OpenCV (Open Computer Vision) là một thư viện mã nguồn mở hàng đầu cho bài toán xử lý về thị giác máy tính, học máy, xử lý ảnh và các tính năng tăng tốc GPU trong hoạt động thời gian thực. OpenCV được viết bằng tối ưu hóa C/C++, thư viện có thể tận dụng lợi thế của xử lý đa lõi. Trong đề tài này, chúng tôi sử dụng OpenCV interface C++ để trích xuất các đặc trưng ảnh rồi sử dụng các phương pháp học máy. Thư viện OpenCV cung cấp cho chúng ta những hàm để cải thiện chất lượng ảnh, tìm biên ảnh và hỗ trợ trong việc tìm các đặc trưng của ảnh. Với đầu vào là tập ảnh đồ hình chụp từ camera điện thoại, thư viện OpenCV hỗ trợ trong đồ án cải thiện chất lượng ảnh và thực hiện tìm biên, trích xuất đặc trưng. Đầu ra chúng ta thu được là tập các file dữ liệu csv lưu trữ các đặc trưng ảnh, với mỗi hàng là một ảnh trong bộ dữ liệu tương ứng.

Phương pháp học máy sử dụng là bộ phân loại máy hỗ trợ vector (Support Vector Machine – SVM) được quan tâm và sử dụng trong các bài toán nhận dạng và phân loại. Từ các công trình khoa học, phương pháp SVM được khẳng định là phương

pháp có khả năng vượt trội về tính hiệu quả, độ chính xác và khả năng xử lý bộ dữ liệu một cách linh hoạt. Việc sử dụng máy hỗ trợ vector SVM đã và đang là một trong những lựa chọn tối ưu trong việc giải quyết các bài toán phân loại / dự báo trong một số nghành khoa học. Trong phạm vi đồ án, chúng tôi thực hiện một số thuật toán học máy cổ điển, mô hình mạng nơ-ron bằng phương pháp Convolutional Neural Network và so sánh kết quả với mô hình học máy SVM. Kết quả thu được sẽ giúp chúng ta đánh giá được chất lượng của từng mô hình qua độ chính xác của thuật toán.

Bố cục của quyển đồ án tốt nghiệp như sau :

Chương 1, Đặt vấn đề. Chương này sẽ trình bày về vấn đề cần giải quyết và những hướng giải pháp cho vấn đề này trong khuôn khổ đồ án.

Chương 2, Một số kiến thức liên quan. Trong chương này sẽ giới thiệu về các kỹ thuật xử lý ảnh sử dụng trong bài toán. Phần sau của chương sẽ trình bày về các thuật toán học máy sử dụng trong đồ án này.

Chương 3, Tìm lỗi của camera theo ảnh mẫu. Nội dung chương 3 sẽ trình bày về tiền xử lý ảnh đầu vào và áp dụng các thuật toán SVM và CNN vào bài toán. Phần sau của chương sẽ trình bày kết quả phân loại ảnh của hai mô hình SVM và CNN, sau đó so sánh với các thuật toán phân loại khác.

Chương 4, Kết luận và hướng phát triển. Nội dung chương 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

Mục lục

Danh mục hình vẽ

Danh mục bảng

Danh mục các từ viết tắt

OpenCV	Open Source Computer Vision Library Thư viện mã nguồn mở thị giác máy tính
HSV	Hue Staturation Value Color Space Không gian màu bão hòa giá trị màu sắc
SIFT	Scale-Invariant Feature Transform Chuyển đổi tỉ lệ bất biến đặc trưng
SVM	Support Vector Machine Máy hỗ trợ vector
CNN	Convolutional Neural Network Mạng nơ-ron tích chập
CNTT	Công nghệ thông tin
ĐATN	Đồ án tốt nghiệp
SV	Sinh viên
Scan	Quét
Load	Tải lên (dữ liệu)

Danh mục thuật ngữ

Test case	Trường hợp kiểm tra
Case Blur	Trường hợp ảnh bị mờ không
Case Flare	Trường hợp tia sáng đều không
Case Flash	Trường hợp đèn flash được bật không
Case White	Trường hợp chụp nền trắng
Smartphone	Điện thoại thông minh
Smart Factory	Nhà máy thông minh
Camera	Máy ảnh
Vector	Véc tơ
Neural Network	Mạng nơ-ron nhân tạo
Deep Learning	Học sâu
Machine Learning	Học máy

Chương 1 Đặt vấn đề

1.1 Bài toán.

Trong quá trình chuyển hóa thành các *"Smart Factory"* của các nhà máy Samsung tại Việt Nam thì việc thay thế các giai đoạn thủ công thành tự động là rất cần thiết. Một trong những bước đi cho sự chuyển hóa là thực hiện công đoạn kiểm tra chất lượng camera của smartphone bằng cách sử dụng các thuật toán học máy. Việc kiểm tra chất lượng camera là một trong những quy trình chuẩn đầu ra quan trọng cho một chiếc điện thoại. Hiện nay, việc thực hiện kiểm tra được nhân viên sản xuất dưới nhà máy thực hiện một cách thủ công: chụp ảnh, so sánh và đánh giá bằng mắt. Việc kiểm tra thủ công như vậy rất phụ thuộc vào con người, môi trường làm việc. Công việc thủ công lặp này đi lặp lại như vậy gây sự nhàm chán và ảnh hưởng rất lớn đến kết quả đầu ra của sản phẩm. Vì vậy, vấn đề mà chúng ta cần giải quyết trong bài toán này là cần phân loại các điện thoại có camera lỗi và cảnh báo cho người quản lý chuỗi sản xuất của thiết bị điện thoại đó để họ có biện pháp xử lý nghiệp vụ đối với từng trường hợp lỗi của camera đó. Vì là đặc thù công việc sản xuất nên chúng ta không thể quyết định loại bỏ thiết bị hay không mà chúng ta chỉ đưa ra cảnh báo.

1.2 Phạm vi đề tài.

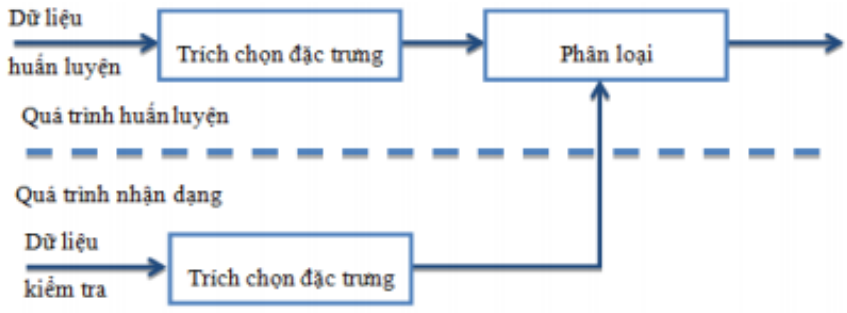
Cơ sở dữ liệu của đồ án là các ảnh chụp các mẫu đồ hình từ các camera điện thoại cần phải kiểm tra. Những loại mẫu đồ hình đã được quy định bởi nhà máy sản xuất. Tập ảnh đã được phân loại đạt và không đạt được cung cấp từ nhà máy Samsung. Do mang tính đặc thù nên đồ án chỉ thực hiện trên bốn trường hợp lỗi có thể xảy ra: kiểm tra ảnh bị mờ hay không (case blur), kiểm tra tia sáng phát ra có đều hay không (case flare), đèn flash có được bật hay không (case flash), ảnh trên nền trắng có nhiễu không (case white). Nếu một phần ảnh của toàn bộ ảnh không thỏa mãn điều kiện thì camera

sẽ bị gán là không thỏa mãn và sẽ được cảnh báo với người công nhân để họ kiểm tra riêng đối với camera đó.

Phương pháp học máy sẽ được áp dụng để kiểm tra các ảnh được chụp với từng trường hợp kiểm tra bằng cách phân loại là ảnh đó có thỏa mãn hay không. Nếu có một trường hợp tương ứng lỗi thì camera đó không đạt yêu cầu.

1.3 Định hướng giải pháp.

Trong đồ án này, chúng tôi sử dụng thư viện OpenCV để thực hiện trích xuất các đặc trưng cho từng trường hợp ảnh cần phân loại. Sau khi hoàn thành, dữ liệu thu được sẽ đưa vào hệ thống phân loại và thực hiện: huấn luyện và nhận dạng. Trong quá trình huấn luyện, các đặc trưng sẽ được trích chọn cho từng trường hợp và bộ phân loại sẽ được huấn luyện để phân chia không gian đặc trưng. Quá trình nhận dạng, bộ phân loại đã được huấn luyện gán một mẫu đầu vào vào một trong các lớp dựa trên các đặc trưng đó. Một mô hình sẽ được xây dựng dựa trên các dữ liệu huấn luyện và mô hình này sẽ được sử dụng để phân loại một dữ liệu mới vào các lớp.



Hình 1.1 *Quá trình huấn luyện và nhận dạng của hệ thống*

Ảnh sẽ được chia thành hai lớp với nhãn là 0 và 1.

Nhãn 0 ứng với ảnh không thỏa mãn, tức là camera bị lỗi. Nhãn 1 ứng với trường hợp ảnh thỏa mãn, tức là camera không bị lỗi.

Phương pháp học máy SVM được sử dụng để phân loại ảnh vào các trường hợp lỗi hoặc không lỗi. SVM sử dụng một không gian đa chiều để phân hoạch các mẫu. Các mẫu nhận dạng được biểu diễn dưới dạng vector. Ban đầu, thuật toán dựa trên tập

mẫu để phân hoạch không gian vector thành các không gian con. Mỗi không gian con tương ứng với một loại mẫu. Dựa vào không gian con này, phương pháp SVM có thể nhận dạng được các mẫu. Phương pháp này sẽ được trình bày chi tiết ở những chương sau.

Đồ án đã thực hiện được việc phân loại ảnh vào các trường hợp 0 và 1. Từ đó có thể phân loại được rằng camera tương ứng có đang bị lỗi hay không. Ngoài ra, chúng tôi còn xây dựng một vài thuật toán học máy khác để so sánh với thuật toán SVM. Trong đó, hệ thống mạng neural network cũng được xây dựng để thực hiện phân loại ảnh. Neural network là mạng có ứng dụng rất mạnh ở hiện tại, đặc biệt là trong Deep Learning với các kết quả thực tiễn đã được chứng minh như Alpha Go của Google,…

Chương 2 Một số kiến thức liên quan

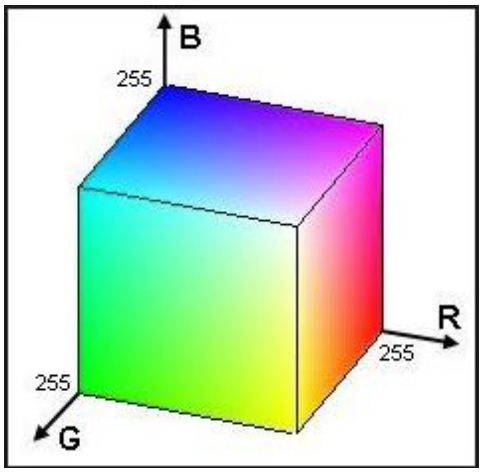
2.1 Một số kỹ thuật xử lý ảnh.

2.1.1 Thu nhận và biểu diễn ảnh.

2.1.1.1 Không gian màu RGB (Red – Green - Blue).

Mô hình màu RGB sử dụng mô hình bổ sung trong đó ánh sáng đỏ, xanh lục và xanh lam được tổ hợp với nhau theo nhiều phương thức khác nhau để tạo thành các màu khác. Từ viết tắt RGB trong tiếng Anh có nghĩa là đỏ (Red), xanh lục (Green) và xanh lam (Blue), là ba màu gốc trong các mô hình ánh sáng bổ sung.

Nếu như một ảnh số được mã hóa bằng 24 bit, nghĩa là 8 bit cho kênh R, 8 bit cho kênh G, 8 bit cho kênh B, thì mỗi kênh này màu này sẽ nhận giá trị từ 0 - 255. Với mỗi giá trị khác nhau của các kênh màu kết hợp với nhau ta sẽ được một màu khác nhau, như vậy ta sẽ có tổng cộng 255 x 255 x 255 = 1.66 triệu màu sắc. Ví dụ: màu đen là sự kết hợp của các kênh màu (R, G, B) với giá trị tương ứng (0, 0, 0), màu trắng có giá trị (255, 255, 255), màu vàng có giá trị (255, 255, 0).



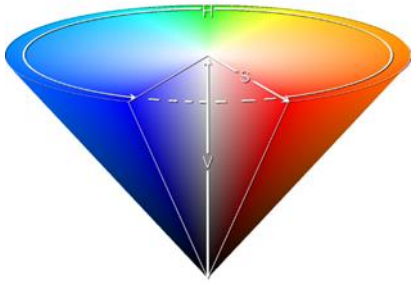
Hình 2.1 *Không gian màu RGB.*

2.1.1.2 Không gian màu HSV (Hue – Saturation – Value).

Thực chất của không gian này là sự biến đổi của không gian RGB. Không gian HSV được mô tả bằng lệnh lập phương RGB quay trên đỉnh Black. H (Hue) là góc quay theo trục V (value) qua 2 đỉnh Black và White (xem Hình 2.2).

Các giá trị biến thiên của H, S, V như sau :

- H (Hue): Sắc thái màu có giá trị từ 0^o (màu đỏ) - 360^o.
- S (Saturation): Độ bão hoà.
- V (Value): Giá trị cường độ sáng có giá trị từ 0 (đỉnh nón, màu đen) – 1 (đáy nón, độ sáng tối đa).



Hình 2.2 *Không gian màu HSV.*

2.1.1.3 Chuyển đổi không gian RGB sang không gian HSV và ngược lại.

- *Chuyển đổi từ không gian RGB sang HSV:*

Giả sử ta có một điểm màu có giá trị trong hệ RGB là (R, G, B). Ta chuyển sang không gian HSV như sau:

Đặt M = Max(R, G, B), m = Min(R, G, B) và C = M - m.

Nếu M = R, H' = (G - B) / C mod 6.

Nếu M = G, H' = (B - R) / C + 2.

Nếu M = B, H' = (R - G) / C + 4.

Và H = H' x 60.

Trong trường hợp C = 0, H = 0^o.

V = M.

S = C / V.

- *Chuyển đổi từ không gian HSV sang RGB:*

Giả sử ta có không gian màu HSV với H = [0, 360], S = [0, 1], V = [0, 1]. Khi đó, ta tính:

C = V x S.

H’ = H / 60.

X = C(1 - |(H' mod 2) - 1|).

m = V – C.

Ta biểu diễn hệ (R, G, B) như sau:

$$(R_1, G_1, B_1) = \begin{cases} (0,0,0)\ nếu\ H\ không\ xác\ định \\ (C,X,0)\ nếu\ 0 \le H' < 1 \\ (X,C,0)\ nếu\ 1 \le H' < 2 \\ (0,C,X)\ nếu\ 2 \le H' < 3 \\ (0,X,C)\ nếu\ 3 \le H' < 4 \\ (X,0,C)\ nếu\ 4 \le H' < 5 \\ (C,0,X)\ nếu\ 5 \le H' < 6 \end{cases}$$

$$(R, G, B) = (R_1 + m, G_1 + m, B_1 + m)$$

2.1.2 Xử lý và nâng cao chất lượng hình ảnh.

Nâng cao chất lượng ảnh là một bước quan trọng, tạo tiền đề cho xử lý ảnh. Mục đích chính là nhằm làm nổi bật một số đặc tính của ảnh như thay đổi độ tương phản, lọc nhiễu, nổi biên, làm trơn biên ảnh, khuếch đại ảnh,…0

2.1.2.1 Một số kỹ thuật không phụ thuộc không gian.

Các phép toán không phụ thuộc không gian là các phép toán không phụ thuộc vị trí của điểm ảnh.

Ví dụ: Phép tăng giảm độ sáng, phép thông kê tần suất, biến đổi tần suất,... Một trong những khái niệm quan trọng trong xử lý ảnh là biểu đổ tần suất (Histogram).

- **Chuẩn hóa histogram:**

Chuẩn hóa histogram là kỹ thuật phân bố lại histogram của ảnh. Những khu vực có độ tương phản thấp sẽ đạt được độ tương phản cao hơn mà không làm ảnh hưởng đến đặc điểm của ảnh.

Khái niệm histogram: Biểu đồ tần suất của mức xám g của ảnh I là số điểm ảnh có giá trị g của ảnh I. Ký hiệu là h(g). 0

Ví dụ:

$$I = \begin{bmatrix} 1 & 2 & 0 & 4 \\ 1 & 0 & 0 & 7 \\ 2 & 2 & 1 & 0 \\ 4 & 1 & 2 & 1 \\ 2 & 0 & 1 & 1 \end{bmatrix}$$

g	0	1	2	4	7
h(g)	5	7	5	2	1

Xét một ảnh một ảnh xám được định nghĩa bởi một hàm f(x,y). Cho I là tổng số mức xám trong ảnh (ví dụ I = 256). Ta sử dụng biểu đồ tần suất histogram để xác định tần suất của mỗi mức xám i, …, I – 1.

$$H(i) = |\{[x,y]\}|\ 0 \le x < w\ \wedge 0 \le y < h\ \wedge f(x,y) = i\}|$$

Trong đó:

- x, y : Tọa độ điểm ảnh.
- w : Độ rộng của ảnh.
- h : Chiều cao của ảnh.

Giá trị lớn nhất, nhỏ nhất và trung bình trong histogram được định nghĩa như sau:

$$H_{min} = \min_{\substack{0 \le x < w \\ 0 \le y < h}} \{f(x,y)\}$$

$$H_{max} = \max_{\substack{0 \le x < w \\ 0 \le y < h}} \{f(x,y)\}$$

$$H_{avg} = \frac{1}{w * h} \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} f(x,y)$$

Trong đó:

$$0 \le H_{min} \le H_{avg} \le H_{max} \le I - 1$$

Mục đích của việc chuẩn hóa histogram là để đạt được một histogram chuẩn mà $H_{min} = 0, H_{max} = I - 1, H_{avg} = \frac{I}{2}$. Để đạt được mục đích này, ta sẽ xây dựng một hàm biến đổi *g(i).*

Cho $[x_1, y_1] = [H_{min}, 0]$, $[x_2, y_2] = [H_{avg}, \frac{I}{2}]$ và $[x_3, y_3] = [H_{max}, I - 1]$ thì hàm g(i) được định nghĩa như sau:

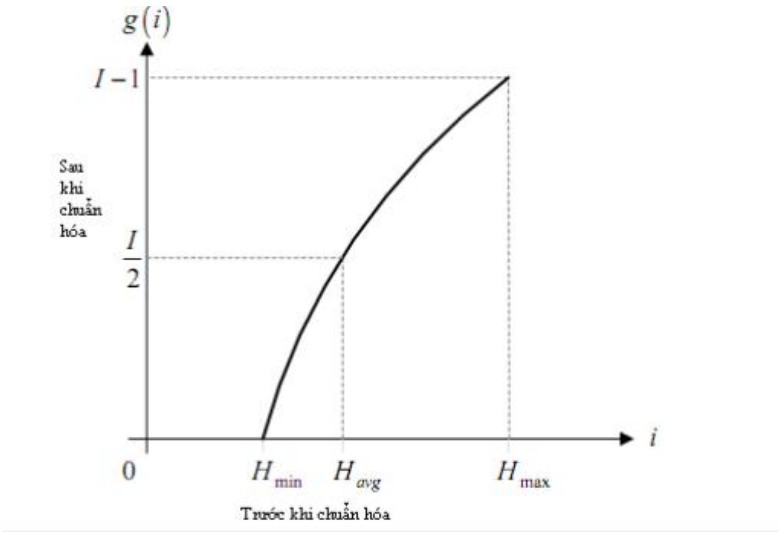
$$g(i) = \sum_{j=1}^{3} \left(y_j \prod_{\substack{k=1 \\ k \ne j}}^{3} \frac{i - x_k}{x_j - x_k} \right)$$

Có thể triển khai cụ thể như sau:

$$g(i) = y_1 . \frac{i - x_2}{x_1 - x_2} . \frac{i - x_3}{x_1 - x_3} + y_2 . \frac{i - x_1}{x_2 - x_1} . \frac{i - x_3}{x_2 - x_3} + y_3 . \frac{i - x_1}{x_3 - x_1} . \frac{i - x_2}{x_3 - x_2}$$

Thay lần lượt các điểm đã cho ở trên vào, với số mức xám 256 ta có:

$$g(i) = 128 . \frac{i - H_{min}}{H_{avg} - H_{min}} . \frac{i - H_{max}}{H_{avg} - H_{max}} + 255 . \frac{i - H_{min}}{H_{max} - H_{min}} . \frac{i - H_{avg}}{H_{max} - H_{avg}}$$



Hình 2.3 *Chuẩn hóa histogram.*

- **Tách ngưỡng:**

Với kĩ thuật tách ngưỡng đơn giản: Nếu pixel có giá trị lớn hơn giá trị ngưỡng thì nó được gán một giá trị (thường là 1), ngược lại nhỏ hơn giá trị ngưỡng thì nó được gán một giá trị khác (thường là 0).

Giả sử ta có ảnh I ~ kích thước m x n, hai số Min, Max và ngưỡng θ. Khi đó, kĩ thuật tách ngưỡng được thể hiện như sau:

```
for(i = 0; i < m; i++) {
    for(j = 0; j < n; j++){
        I[i,j] = I[i,j] > = θ ? Max : Min;
    }
}
```

- **Kỹ thuật tách ngưỡng tự động:**

Kỹ thuật tìm tách ngưỡng tự động nhằm tìm ra ngưỡng một cách tự động dựa vào histogram theo nguyên lý trong vật lý là vật thể tách làm hai phần nếu tổng độ lệch trong từng phần là tối thiểu.

Ngưỡng θ trong kỹ thuật tách ngưỡng thường được cho bởi người sử dụng. Kỹ thuật tách ngưỡng tự động nhằm tìm ra ngưỡng θ một cách tự động dựa vào histogram. 0

Giả sử , ta có:

- Ảnh I ~ Kích thước m x n.
- G ~ Số là số mức xám của ảnh kể cả khuyết thiếu.
- t(g) ~ Số điểm ảnh có mức xám ≤ g.

$\mathrm{m(g)} = \frac{1}{t(g)} \sum_{i=0}^{g} i.h(i)$ ~ mô men quán tính TB có mức xám ≤ g.

Hàm f: g → f(g):

$$f(g) = \frac{t(g)}{mxn - t(g)} [m(g) - m(G-1)]^2$$

Tìm θ sao cho:

$$f(\theta) = \max_{0 \le g < G-1} \{f(g)\}$$

2.1.2.2 Thuật toán tách ngưỡng tự động Otsu.

Otsu là tên một nhà nghiên cứu người Nhật đã nghĩ ra ý tưởng cho việc tính ngưỡng *T* một cách tự động (adaptive) dựa vào giá trị điểm ảnh của ảnh đầu vào nhằm thay thế cho việc sử dụng ngưỡng cố định (fixed hay const). Phương pháp này cho kết quả là mỗi ảnh khác nhau có một ngưỡng tương ứng khác nhau bằng các bước xử lý như sau:

- *Chọn một giá trị khởi tạo cho T:*

Ta nên chọn giá trị mang tính công thức, ví dụ, $T = \frac{\min + max}{2}$. Chúng ta tránh dùng các giá trị mang tính định lượng thiết lập cứng).

- *Phân hoạch ảnh sử dụng T:*

Kết quả của bước này sẽ tạo ra hai nhóm điểm ảnh: G_1 chứa tất cả các điểm ảnh với giá trị (intensity) > T và G_2 chứa các điểm ảnh với giá trị (intensity) ≤ T.

- *Tính trung bình (Average hay Mean):*

Ta cần tính giá trị m_1 và m_2 của các điểm ảnh thuộc $G_1 v$à G_2.

- *Tính lại T dựa vào m_1 và m_2:*

$$T = \frac{m_1 + m_2}{2}$$

- Lặp lại bước 2 đến 4 cho tới khi nào giá trị chênh lệch giữa *T cũ* và *T mới* là không đáng kể (nhỏ hơn một giá trị cho trước *deltaT*). *deltaT* thường được sử dụng là sai số từ các phép tính toán trong quá trình xử lý. Trong trường hợp này T sẽ có phép sai số là 1 / 2*(giá trị đơn vị của điểm ảnh).

2.1.2.3 Nhân chập ảnh với phép lọc số ảnh.

Để thực hiện một phép lọc số ảnh, ta cần phải tiến ảnh nhân chập ảnh đầu vào với một ma trận lọc hay cửa sổ nhân chập (Kernel). Toàn bộ các điểm ảnh (Pixel) trên ảnh sẽ được tiến hành nhân chập với ma trận lọc, tâm của ma trận lọc sẽ được đặt trùng vào vị trí của điểm ảnh (Pixel) đang được tính nhân chập làm thay đổi các giá trị của pixel ban đầu.

Công thức tính nhân chập:

$$I_{dst}(x,y) = I_{src}(x,y) * M(u,v) = \sum_{u=-n}^{n} \sum_{v=-n}^{n} I_{src}(x+u, y+v) * M(u,v)$$

Với n = (Kích thước ma trận lọc - 1) / 2 và lấy tâm làm của ma trận lọc làm điểm gốc.

2.1.2.4 Một số kỹ thuật lọc nhiễu.

- **Lọc trung bình:**

Với lọc trung bình, mỗi điểm ảnh (Pixel) được thay thế bằng trung bình trọng số của các điểm trong vùng lân cận. Ta giả sử rằng, chúng ta có một ma trận lọc (Kernel) (3x3) quét qua từng điểm ảnh của ảnh đầu vào I_{src}. Tại vị trí mỗi điểm

ảnh lấy giá trị của các điểm ảnh tương ứng trong vùng (3x3) của ảnh gốc đặt vào ma trận lọc (Kernel). Giá trị điểm ảnh của ảnh đầu ra I_{dst} là giá trị trung bình của tất cả các điểm trong ảnh trong ma trận lọc (Kernel).

Một ảnh đầu vào với I(x,y) là giá trị điểm ảnh tại một điểm (x,y) và một ngưỡng θ.

- *Bước 1*: Tính tổng các thành phần trong ma trận lọc (Kernel).
- *Bước 2:* Chia lấy trung bình của tổng các thành phần trong ma trận được tính ở trên với số lượng các phần tử của cửa sổ lọc ra một giá trị $I_{tb}(x, y)$.
- *Bước 3*: Hiệu chỉnh:
  Nếu $I(x, y) - I_{tb}(x, y) > \theta$ thì $I(x, y) = I_{tb}(x, y)$.
  Nếu $I(x, y) - I_{tb}(x, y) <= \theta$ thì $I(x, y)=I(x, y)$.

θ là một giá trị cho trước và có thể có hoặc không tùy thuộc vào mục đích.

Tác dụng:

Trong lọc trung bình, thường người ta ưu tiên cho các hướng để bảo vệ biên của ảnh khỏi bị mờ khi làm trơn ảnh. Các kiểu ma trận lọc (Kernel) được sử dụng tùy theo các trường hợp khác nhau. Các bộ lọc trên là bộ lọc tuyến tính theo nghĩa là điểm ảnh ở tâm cửa sổ sẽ được thay bởi tổ hợp các điểm lân cận chập với ma trận lọc (Kernel).

- **Lọc trung vị:**

Lọc trung vị là lọc phi tuyến. Một phép lọc phi tuyến là một kết quả không thể thu được từ một tổng trọng số của các điểm ảnh (Pixel) lân cận. Sau khi đã định nghĩa kích thước vùng lân cận, giá trị điểm ảnh (Pixel) trung tâm được thay bằng trung vị tức là giá trị chính giữa của tất cả các giá trị của các điểm trong vùng lân cận.

Cho một dãy số $X_1, X_2, X_3, ..., X_n$ được sắp xếp theo thứ tự tăng dần hoặc giảm dần. Khi đó X_{tv} được tính bởi công thức:

Nếu *n* lẻ: $X_{tv} = X(\frac{n}{2} + 1)$

Nếu *n* chẵn: $X_{tv} = \frac{X\left(\frac{n}{2}\right)+X\left(\frac{n}{2}+1\right)}{2}$

Kích thước của ma trận lọc (Kernel) thường dùng có kích thước 3x3, 5x5, 7x7.

Thuật toán:

- Bước 1: Xác định điểm ảnh (Pixel) I(x,y).
- Bước 2: Sắp xếp các điểm ảnh (Pixel) lân cận của theo một thứ tự nhất định (tăng dần hoặc giảm dần).
- Bước 3: Xác định I_{tv} (Chính là X_{tv} trong nội dung phía trên).
- Bước 4: Hiệu chỉnh:
  Nếu $I(x, y) - I_{tv}(x, y) > \theta$ thì $I(x, y) = I_{tv}(x, y)$.
  Nếu $I(x, y) - I_{tv}(x, y) \leq \theta$ thì $I(x, y) = I(x, y)$.

2.1.2.5 Các phép toán hình thái học.

Hình thái học toán học (Mathematical morphology) là một lý thuyết và kỹ thuật để phân tích và xử lý cấu trúc hình học, dựa trên lý thuyết tập hợp, lý thuyết lưới, cấu trúc liên kết và chức năng ngẫu nhiên. Hình thái học toán học phổ biến nhất được áp dụng cho hình ảnh kỹ thuật số. Ngoài ra hình thái học toán học, nó có thể được sử dụng là tốt trên đồ thị, bề mặt mắt lưới, chất rắn, và nhiều các cấu trúc không gian khác.

Hình thái học toán học đã được phát triển cho hình ảnh nhị phân, và sau đó được mở rộng cho ảnh đa mức xám (Image Grayscale),... Đây là một trong những kỹ thuật được áp dụng trong giai đoạn tiền xử lý. Hai phép toán thường dùng là *phép giãn nở (Dilation)* và *phép co (Erosion)*. Từ hai phép toán cơ bản này người ta phát triển thành một số phép toán như *phép đóng (Closing)* và *phép mở (Opening)*.

- **Phép co ảnh:**

Với hai tập A và B là các tập trong Z^2, phép co đối với A theo B, ký hiệu là $A \ominus B$, là một tập hợp gồm tất cả các điểm z sao cho $(B)_z$ bị chứa trong A.

Công thức: $A \ominus B = \{ z \mid (B)_z \subseteq A \}$

Một điểm ảnh trong ảnh gốc (hoặc 1 hoặc 0) sẽ chỉ được coi là 1 nếu tất cả các điểm ảnh dưới hạt nhân là 1, nếu không nó sẽ là 0.

Hình 2.4 *Phép co với nhân 5x5 với toàn bộ là giá trị 1.*

- **Phép dãn ảnh:**

Với hai tập A và B là các tập trong Z^2, phép giãn đối với A theo B, ký hiệu A⊕B, là một tập hợp tất cả các điểm z sao cho $(\hat{B})_z$ có ít nhất một phần tử chung với A.

Công thức: $A \oplus B = \{ z \mid \left(\hat{B}\right)_z \cap A \neq \emptyset \}$

Phần tử pixel là '1' nếu ít nhất một pixel dưới hạt nhân là '1'. Vì vậy, nó làm tăng vùng màu trắng trong hình ảnh hoặc kích thước của đối tượng tiền cảnh lên.

Hình 2.5 *Phép giãn cho ảnh.*

- **Phép mở:**

Phép mở của một tập A bởi phần tử cấu trúc B, kí hiệu là A∘B, được định nghĩa bởi: $A \circ B = (A \ominus B) \oplus B$

Phép mở là phép toán thực hiện phép co sau đó là phép giãn. Nó rất hữu ích trong việc loại bỏ nhiễu.

Hình 2.6 *Phép mở*

- **Phép đóng:**

Phép đóng của một tập A bởi phần tử cấu trúc B, kí hiệu là A•B, được định nghĩa bởi: $A \bullet B = (A \oplus B) \ominus B$

Phép đóng là phép toán thực hiện phép giãn sau đó là phép co. Nó rất hữu ích trong việc đóng các lỗ nhỏ bên trong các đối tượng nền, hoặc các điểm nhỏ màu đen trên đối tượng.

Hình 2.7 *Phép đóng*

2.1.3 Biên và các phương pháp phát hiện biên.

Biên là vấn đề quan trọng trong trích chọn đặc điểm nhằm tiến tới hiểu ảnh. Cho đến nay chưa có định nghĩa chính xác về biên, trong mỗi ứng dụng người ta đưa ra các độ đo khác nhau về biên, một trong các độ đo đó là độ đo về sự thay đổi đột ngột về cấp xám. Tập hợp các điểm biên tạo nên biên hay đường bao của đối tượng. Xuất phát từ cơ sở này người ta thường sử dụng hai phương pháp phát hiện biên cơ bản:

Phát hiện biên trực tiếp: Phương pháp này làm nổi biên dựa vào sự biến thiên mức xám của ảnh. Kỹ thuật chủ yếu dùng để phát hiện biên ở đây là kỹ thuật lấy đạo hàm. Nêu lấy đạo hàm bậc nhất của ảnh ta có các kỹ thuật Gradient, nếu lấy đạo hàm bậc hai của ảnh ta có kỹ thuật Laplace. Ngoài ra còn có một số các tiếp cận khác.

Phát hiện biên gián tiếp: Nếu bằng cách nào đó ta phân được ảnh thành các vùng thì ranh giới giữa các vùng đó gọi là biên. Kỹ thuật dò biên và phân vùng ảnh là hai bài toán đôi ngẫu nhau vì dò biên để thực hiện phân lớp đối tượng mà khi đã phân lớp xong nghĩa là đã phân vùng được ảnh và ngược lại, khi đã phân vùng ảnh đã được phân lớp thành các đối tượng, do đó có thể phát hiện được biên.

Phương pháp phát hiện biên trực tiếp tỏ ra khá hiệu quả và ít chịu ảnh hưởng của nhiễu, song nếu sự biến thiên độ sáng không đột ngột, phương pháp tỏ ra kém hiệu quả, phương pháp phát hiện biên gián tiếp tuy khó cài đặt, song lại áp dụng khá tốt trong trường hợp này. [2]

2.1.3.1 Kỹ thuật phát hiện biên Gradient.

Theo định nghĩa, gradient là một véctơ có các thành phần biểu thị tốc độ thay đổi giá trị của điểm ảnh [2], ta có:

$$\begin{cases} \dfrac{\partial f(x,y)}{\partial x} = f'x = \dfrac{f(x+dx,y) - f(x,y)}{dx} \\ \dfrac{\partial f(x,y)}{\partial y} = f'_y = \dfrac{f(x,y+dy) - f(x,y)}{dy} \end{cases}$$

Trong đó, *dx, dy* là khoảng cách (tính bằng số điểm) theo hướng *x* và *y*.

Tuy ta nói là lấy đạo hàm nhưng thực chất chỉ là mô phỏng và xấp xỉ đạo hàm bằng các kỹ thuật nhân chập vì ảnh sô là tín hiệu rời rạc nên đạo hàm không tồn tại.

Với dx = dy = 1, ta có mặt nạ nhân chập theo hướng x là A = (-1 1), theo hướng y là B = $\begin{pmatrix} -1 \\ 1 \end{pmatrix}$

2.1.3.2 Kỹ thuật phát hiện biên Laplace.

Các phương pháp đánh giá gradient ở trên làm việc khá tốt khi mà độ sáng thay đổi rõ nét. Khi mức xám thay đổi chậm, miền chuyển tiếp trải rộng, phương pháp cho hiệu quả hơn đó là phương pháp sử dụng đạo hàm bậc hai Laplace. [2]

Toán tử Laplace được định nghĩa như sau:

Ta có:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

$$\frac{\partial^2 f}{\partial x^2} = \frac{\partial}{\partial x}\left(\frac{\partial f}{\partial x}\right) \approx \frac{\partial}{\partial x}\big(f(x+1,y) - f(x,y)\big)$$

$$\approx [f(x+1,y) - f(x,y)] - [f(x,y) - f(x-1,y)]$$

$$\approx f(x+1,y) - 2f(x,y) + f(x-1,y)$$

Tương tự:

$$\frac{\partial^2 f}{\partial y^2} = \frac{\partial}{\partial x}\left(\frac{\partial f}{\partial x}\right) \approx \frac{\partial}{\partial x}\big(f(x+1,y) - f(x,y)\big)$$

$$\approx [f(x,y+1) - f(x,y)] - [f(x,y) - f(x,y-1)]$$

$$\approx f(x,y+1) - 2f(x,y) + f(x,y-1)$$

Vậy:

$$\nabla^2 f = f(x+1,y) + f(x,y+1) - 4f(x,y) + f(x-1,y) + f(x,y-1)$$

Dẫn tới:

$$H = \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

Trong thực tế, người ta thường dùng nhiều kiểu mặt nạ khác nhau để xấp xỉ rời rạc đạo hàm bậc hai Laplace.

2.2 Vấn đề trích xuất đặc trưng của hình ảnh.

2.2.1 Một số đặc điểm cơ bản.

Các đặc điểm của đối tượng được trích chọn tuỳ theo mục đích nhận dạng trong quá trình xử lý ảnh. Có thể nêu ra một số đặc điểm của ảnh sau đây:

Đặc điểm không gian: Phân bố mức xám, phân bố xác suất, biên độ, điểm uốn,...

Đặc điểm biến đổi: Các đặc điểm loại này được trích chọn bằng việc thực hiện lọc vùng (zonal filtering). Các bộ vùng được gọi là *"mặt nạ đặc điểm"* (feature mask) thường là các khe hẹp với hình dạng khác nhau (chữ nhật, tam giác, cung tròn,...).

Đặc điểm biên và đường biên: Đặc trưng đường biên của đối tượng rất hữu ích trong việc trích chọn các thuộc tính bất biến và được dùng khi nhận dạng đối tượng. Các đặc điểm này có thể được trích chọn nhờ toán tử gradient, toán tử la bàn, toán tử Laplace, toán tử "chéo không" (zero crossing),...

Việc trích chọn hiệu quả các đặc điểm giúp cho việc nhận dạng các đối tượng ảnh chính xác, với tốc độ tính toán cao và dung lượng nhớ lưu trữ giảm xuống.

2.2.2 Đặc trưng hình dạng.

Màu sắc và kết cấu là những thuộc tính có khái niệm toàn cục trong một ảnh. Trong khi đó, hình dạng không phải là một thuộc tính của ảnh. Do đó, hình dạng thường được mô tả sau khi các ảnh được phân đoạn thành các vùng hoặc các đối tượng. Hay hình dạng chỉ là biên của đối tượng nào đó trong ảnh. Một biểu diễn đặc trưng tốt cho một đối tượng phải bất biến với dịch chuyển, quay và tỉ lệ.

2.2.3 Mômen hình ảnh và các bất biến mômen.

Trong xử lý ảnh, thị giác máy tính và những miền liên quan, một mômen hình ảnh là một trọng số trung bình đặc trưng của cường độ sáng các điểm ảnh, hoặc là một hàm của mô men thường được chọn để có các thuộc tính hoặc sự thể hiện hấp dẫn.

Mômen hình ảnh thường được dùng để mô tả đối tượng sau khi phân đoạn. Các thuộc tính đơn giản của hình ảnh được tìm thấy thông qua mô men hình ảnh bao gồm diện tích (hoặc cường độ tổng thể), trọng tâm và thông tin về hướng của nó.

2.2.3.1 Mômen thô.

Đối với hàm 2D liên tục f (x , y) mô men (đôi khi được gọi là "mô-men thô") của bậc (p + q) được định nghĩa là:

$$M_{pq} = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} x^p y^q f(x,y) dxdy$$

Với p, q = 0, 1, 2,…

Thích ứng với hình ảnh vô hướng (thang độ xám) với cường độ điểm ảnh I(x, y), khoảnh khắc hình ảnh thô M_{ij} được tính bằng:

$$M_{ij} = \sum_x \sum_y x^i y^j I(x,y)$$

Trong một số trường hợp, điều này có thể được tính toán bằng cách xem xét hình ảnh dưới dạng hàm mật độ xác suất, tức là bằng cách chia tỷ lệ trên cho:

$$\sum_x \sum_y I(x,y)$$

Một định lý duy nhất (Hu [1962]) nói rằng nếu f(x , y) liên tục lặp lại và chỉ có giá trị *nonzero* trong một phần hữu hạn của mặt phẳng *xy*, mômen của tất cả các thứ tự đều tồn tại, và chuỗi mô men (M_{pq}) là xác định duy nhất bởi f(x , y). Ngược lại, (M_{pq}) xác

định duy nhất f(x , y). Trong thực tế, hình ảnh được tóm tắt với các chức năng của một vài mômen bậc thấp hơn.

Các thuộc tính hình ảnh đơn giản có nguồn gốc thông qua các mô men thô bao gồm:

Diện tích (đối với hình ảnh nhị phân) hoặc tổng mức xám (đối với hình ảnh màu xám): M_{00}

Trọng tâm: $\{\bar{x}, \bar{y}\} = \left\{\frac{M_{10}}{M_{00}}, \frac{M_{01}}{M_{00}}\right\}$

2.2.3.2 Mômen trung tâm.

Những mômen trung tâm được định nghĩa là:

$$\mu_{pq} = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} (x - \bar{x})^p (y - \bar{y})^q f(x, y)$$

Những mômen trung tâm đến bậc 2 là:

$$\mu_{00} = M_{00}$$

$$\mu_{01} = 0$$

$$\mu_{10} = 0$$

$$\mu_{11} = M_{11} - \bar{x}M_{01} = M_{11} - \bar{y}M_{10}$$

$$\mu_{20} = M_{20} - \bar{x}M_{10}$$

$$\mu_{02} = M_{02} - \bar{y}M_{01}$$

2.2.3.3 Mômen bất biến.

- **Bất biến tịnh tiến:**

  Những mômen trung tâm μ_{ij} của bất cứ bậc nào, bất biến đối với tịnh tiến.

- **Bất biến tỉ lệ:**

Các bất biến η đối với cả tịnh tiến và tỉ lệ có thể được xây dựng từ các mômen trung tâm bằng cách chia thông qua một mômen trung tâm thứ 0 được chia tỷ lệ đúng:

$$n_{ij} = \frac{\mu_{ij}}{\mu_{00}^{\left(1+\frac{i+j}{2}\right)}}$$

Trong đó: $i + j \geq 2$

- **Bất biến xoay:**

Như thể hiện trong công trình của Hu , bất biến liên quan đến tịnh tiến, tỉ lệ và quay có thể được xây dựng:

$$I_1 = \eta_{20} + \eta_{02}$$

$$I_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$$

$$I_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$$

$$I_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$

$$I_5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$

$$I_6 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})$$

$$I_7 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] - (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$

Chúng được gọi là *mômen bất biến Hu.*

Công thức đầu tiên, I_1, tương tự với mômen quán tính xung quanh tâm của hình ảnh, nơi cường độ của điểm ảnh tương tự với mật độ vật lý. Công thức cuối cùng, I_7, là biến thể bất biến, cho phép nó phân biệt hình ảnh phản chiếu của những hình ảnh giống hệt nhau.

2.2.4 Đặc trưng cục bộ bất biến SIFT.

Các đặc trưng này bất biến với việc thay đổi tỉ lệ ảnh, quay ảnh, đôi khi là thay đổi điểm nhìn và thêm nhiễu ảnh hay thay đổi cường độ chiếu sáng của ảnh.

Phương pháp được lựa chọn có tên là *Scale-Invariant Feature Transform (SIFT)* và đặc trưng trích rút được gọi là đặc trưng SIFT (SIFT Feature). Các đặc trưng SIFT này được trích rút ra từ các điểm hấp dẫn cục bộ (Local Interest Point).

Điểm hấp dẫn (Interest Point (Keypoint)): Là vị trí (điểm ảnh) "hấp dẫn" trên ảnh. "Hấp dẫn" ở đây có nghĩa là điểm đó có thể có các đặc trưng bất biến với việc quay ảnh, co giãn ảnh hay thay đổi cường độ chiếu sáng của ảnh.

Phương pháp trích rút các đặc trưng bất biến SIFT được tiếp cận theo phương pháp thác lọc, theo đó phương pháp được thực hiện lần lượt theo các bước sau:

- **Phát hiện các điểm cực trị Scale-Space** (Scale-Space Extrema Detection):

  Bước đầu tiên này tiến hành tìm kiếm các điểm hấp dẫn trên tất cả các tỉ lệ và vị trí của ảnh. Nó sử dụng hàm Different-of-Gaussian để xác định tất cả các điểm hấp dẫn tiềm năng mà bất biến với quy mô và hướng của ảnh.

- **Định vị các điểm hấp dẫn** (Keypoint Localization):

  Một hàm kiểm tra sẽ được đưa ra để quyết định xem các điểm hấp dẫn tiềm năng có được lựa chọn hay không?

- **Xác định hướng cho các điểm hấp dẫn** (Orientation assignment):

  Xác định hướng cho các điểm hấp dẫn được chọn.

- **Mô tả các điểm hấp dẫn** (Keypoint descriptor):

Các điểm hấp dẫn sau khi được xác định hướng sẽ được mô tả dưới dạng các vector đặc trưng nhiều chiều.

2.3 Support Vector Machine (SVM).

2.3.1 Khái quát về SVM.

Support Vector Manchines (SVM) là kỹ thuật mới đối với việc phân lớp dữ liệu, là phương pháp học sử dụng không gian giả thuyết các hàm tuyến tính trên không gian đặc trưng nhiều chiều, dựa trên lý thuyết tối ưu và lý thuyết thống kê. Sử dụng phương pháp SVM không phải chịu những hạn chế của chiều dữ liệu và hạn chế về các mẫu.

SVM lần đầu tiên được đề xuất bởi Vapnik và kể từ đó đã thu hút được một lượng cao mức độ quan tâm trong nghiên cứu học máy cộng đồng.

SVM đã được sử dụng trong một phạm vi rộng của thế giới thực, vấn đề như phân loại văn bản, nhận dạng âm thanh, phân loại hình ảnh và phát hiện đối tượng, mảng phân tích dữ liệu biểu hiện gen, phân loại dữ liệu. Chứng minh rằng SVM luôn vượt trội hơn các phương pháp học máy có giám sát khác. Tuy nhiên, một số bộ dữ liệu, hiệu suất của SVM là rất nhạy cảm với cách các tham số chi phí và thông số hạt nhân được thiết lập. Kết quả là người sử dụng bình thường cần phải tiến hành kiểm chứng chéo rộng để tìm ra các thiết lập thông số tối ưu. Qúa trình này thường được gọi là mô hình lựa chọn. Một vấn đề thực tế với mô hình lựa chọn là quá trình này rất tốn thời gian.

SVM được thiết lập của học giám sát liên quan, phương pháp được sử dụng để phân loại và hồi quy. Chúng thuộc về một gia đình của phân loại tuyến tính tổng quát. Một tài sản đặc biệt của SVM là SVM đồng thời hạn chế tối đa các thực nghiệm lỗi phân loại và tối đa hóa lề hình học. Vì vậy SVM gọi lề tối đa phân loại. SVM dựa trên kết cấu giảm thiểu rủi ro (SRM). Đồ thị SVM vecto đầu vào cho một không gian chiều cao hơn, nơi tối đa một siêu phẳng tách được xây dựng. Hai siêu phẳng song song được xây dựng trên mỗi bên của siêu phẳng mà tách dữ liệu. Các phân cách siêu phẳng là siêu phẳng tối đa hóa các khoảng cách giữa hai siêu phẳng song song. Một giả thiết được đưa ra rằng biên độ lớn hơn hoặc khoảng cách giữa các siêu phẳng song song tốt hơn các lỗi tổng quát của phân loại.

2.3.2 Cơ sở của SVM tuyến tính.

Đây là trường hợp đơn giản nhất. Trong trường hợp này, tập mẫu là một tập có thể phân chia tuyến tính bằng một siêu phẳng. Và SVM đi tìm siêu phẳng này.

2.3.2.1 Giai đoạn huấn luyện SVM.

Xét tập n mẫu huấn luyện: $\{(x_1, y_1), (x_2, y_2), (x_3, y_3), \ldots, (x_n, y_n)\}$.

Trong đó x_i là một vector đầu vào được biểu diễn trong không gian $X \subseteq R^n$ với y_i là một nhãn lớp $y_i \in \{1, -1\}$.

Cần xác định được một siêu phẳng mà có thể tách biệt được hai lớp trên. Có thể có nhiều siêu phẳng như vậy và vấn đề là cần tìm ra siêu phẳng nào làm cho khoảng cách Euclid giữa hai lớp trên là lớn nhất (Hình 2.8). Lúc đó các vector có khoảng cách gần siêu phẳng nhất được gọi là *Support Vector*.

Giả sử phương trình siêu phẳng cần tìm là $w.x + b = 0$ trong đó w là vector pháp tuyến của siêu phẳng, $w \in Rn$. Ta có hai bất phương trình sau:

$w.x_i + b \leq -1$ với $y_i = -1$

$w.x_i + b \geq +1$ với $y_i = +1$

Kết hợp hai bất phương trình trên:

$y_i (w.x_i + b) - 1 \geq 0$ **(CT.2-1)**

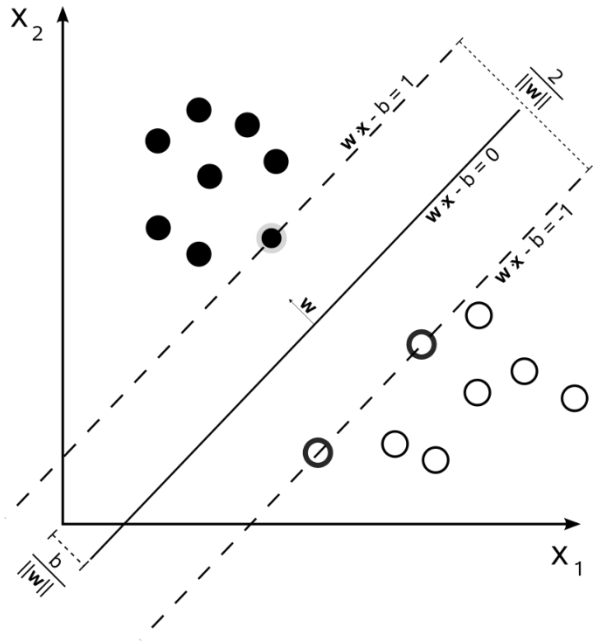
Khi đó những support vector xi thỏa mãn phương trình $w.x_i + b = -1$ nằm trên siêu phẳng H_1, phương trình $w.x_i + b = 1$ nằm trên siêu phẳng H_2.

Khoảng cách có dấu d_1 từ gốc tọa độ đến H_1 là: $d_1 = (1 - b) / \|w\|$

Khoảng cách có dấu d_2 từ gốc tọa độ đến H_2 là: $d_2 = (-1 - b) / \|w\|$

Suy ra khoảng cách phân hoạch d giữa H_1 và H_2 là: $d = |d_1 - d_2| = 2 / \|w\|$

Do đó để có d lớn nhất thì $\|w\|$ phải nhỏ nhất hay nói cách khác phải đi tìm cực tiểu của $\frac{1}{2}\|w\|$ (được biến đổi một chút để sau này dễ tìm w).



Hình 2.8 *Ví dụ siêu phẳng với lề cực đại trong không gian* R^2

Tại sao lại phải cần tìm siêu phẳng ứng với khoảng cách phân hoạch lớn nhất. Đơn giản để khi có thêm mẫu mới thì khả năng phân tách hai lớp là lớn nhất.

- **Bài toán:**

Tìm cực tiểu của $\frac{1}{2}\|w\|$ *theo w và b.*

Với điều kiện $y_i(w.x_i + b) - 1 \geq 0$ *trong đó* $i = 1, 2, ..., l$.

Đây là loại bài toán tối ưu có ràng buộc, trong đó hàm mục tiêu là một hàm lồi và miền ràng buộc cũng là một tập lồi.

Do có tính lồi nên để giải bài toán trên ta có thể chuyển qua giải bài toán đối ngẫu tương ứng.

- **Bài toán đối ngẫu:**

Tìm cực đại của : $\theta(u)$ *với* $u \in R^l$, $u \geq 0$.

Trong đó: $\theta(u) = inf\left\{\frac{1}{2}||w||^2 - \sum_{i=1}^{l}[y_i(w.x_i + b) - 1] : w \in R^n,\ b \in R\right\}$

Để giải bài toán đối ngẫu trước tiên ta phải tìm cực tiểu của:

$L(w,b) = \frac{1}{2}||w||^2 - \sum_{i=1}^{l}[y_i(w.x_i + b) - 1] = 0$ theo *w* và *b*.

Vì L là hàm hai biến *w*, *b* bậc hai không ràng buộc nên theo điều kiện Fermat, cực tiểu của L xảy ra tại *w* và *b* sao cho:

$\frac{\partial L(w,b)}{\partial w} = w - \sum_{i=1}^{l} u_i x_i y_i = 0 \Rightarrow w = \sum_{i=1}^{l} u_i x_i y_i$ **(CT.2-2)**

$\frac{\partial L(w,b)}{\partial b} = \sum_{i=1}^{l} u_i y_i = 0$ **(CT.2-3)**

Lúc đó giá trị của L là:

$$L_0(w,b) = \frac{1}{2}||w||^2 - \sum_{i=1}^{l}[y_i(w.x_i + b) - 1]$$

$$= \sum_{i=1}^{l} u_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} u_i u_j y_i y_j x_i x_j$$

Như vậy bài toán đối ngẫu được viết lại thành:

Tìm cực đại của: $F(u) = \sum_{i=1}^{l} u_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} u_i u_j y_i y_j x_i x_j$

Dựa trên: $u_i \geq 0 \ với \ i = 1, 2, \dots, l.$ $\sum_{i=0}^{l} u_i y_i = 0$

Giải bài toán này ta tìm được *u* và từ đó tính được *w* theo công thức **(**CT.2-2).

Để tính *b*, vận dụng điều kiện *Karush – Kuhn – Tucker (KKT)* được phát biểu bởi Fletcher năm 1987:

$$u_i[y_i(w.x_i + b) - 1] = 0 \quad i = 1, 2, \dots, l$$

Do đó đối với một *i* thì có hai trường hợp:

- **u_i = 0:**

  Trong trường hợp này $y_i(w.x_i + b) - 1 > 0$ suy ra x_i không nằm trên siêu phẳng H_1 hay H_2. Vì $u_i = 0$ nên x_i không tham gia vào việc cấu trúc *w* theo công thức

(CT.2-2). Những x_i này là không cần thiết và có thể được bỏ đi mà không ảnh hưởng đến w.

- **$u_i > 0$:**

Lúc này $y_i(w.x_i + b) - 1 = 0$ suy ra x_i nằm trên siêu phẳng H_1 hay H_2. x_i được gọi là *support vector* và tham gia vào việc tính w. Thường thì số lượng *support vector* nhỏ hơn nhiều so với số lượng mẫu.

Do đó để tính b chỉ cần chọn một i mà có $u_i > 0$, lúc đó:

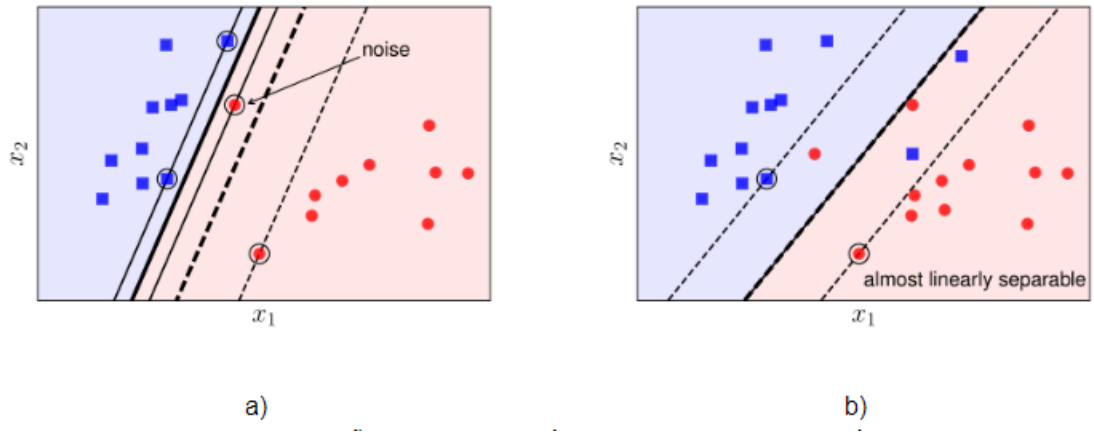
$$y_i(w.x_i + b) - 1 = 0 \; nên \; b = \frac{1}{y_i} - wx_i = y_i - wx_i \qquad \textbf{(CT.2-4)}$$

2.3.2.2 Giai đoạn nhận dạng.

Bây giờ giả sử có một mẫu dữ liệu cần nhận dạng x^* nào đó, thì phân lớp y^* của x^* (-1 hay 1) được xác định thông qua công thức:

$$y^* = f(x^*) = sign(wx^* + b) = sign\left(\sum_{i=1}^{l} u_i x_i y_i x^* + b\right) \qquad \textbf{(CT.2-5)}$$

2.3.3 SVM biên mềm.



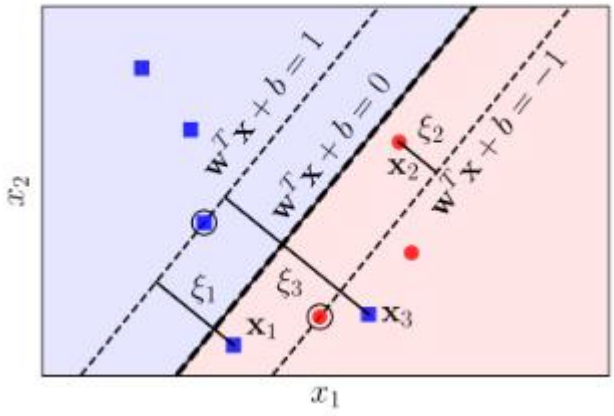
Hình 2.9 *Soft margin SVM. a) Khi có nhiễu. b) Khi dữ liệu gần phân biệt tuyến tính.*

Có hai trường hợp dễ nhận thấy SVM làm việc không hiệu quả hoặc thậm chí không làm việc:

Trường hợp 1: Dữ liệu vẫn phân biệt tuyến tính như hình 2.9a nhưng có một điểm nhiễu của lớp tròn đỏ ở quá gần so với lớp vuông xanh. Trong trường hợp này, nếu ta sử dụng SVM thuần thì sẽ tạo ra một lề (margin) rất nhỏ. Ngoài ra, đường phân lớp nằm quá gần lớp vuông xanh và xa lớp tròn đỏ. Trong khi đó, nếu ta hy sinh điểm nhiễu này thì ta được một biên tốt hơn rất nhiều được mô tả bởi các đường nét đứt. SVM thuần vì vậy còn được coi là nhạy cảm với nhiễu (sensitive to noise).

Trường hợp 2: Dữ liệu không phân biệt tuyến tính nhưng gần phân biệt tuyến tính như hình 2.9b. Trong trường hợp này, nếu ta sử dụng SVM thuần thì rõ ràng bài toán tối ưu là không tìm được, tức tập nghiệm khả thi *(feasible set)* là một tập rỗng, vì vậy bài toán tối ưu SVM trở nên vô nghiệm. Tuy nhiên, nếu ta lại chịu hy sinh một chút những điểm ở gần biên giữa hai lớp, ta vẫn có thể tạo được một đường phân chia khá tốt như đường nét đứt đậm. Các đường support đường nét đứt mảnh vẫn giúp tạo được một margin lớn cho bộ phân lớp này. Với mỗi điểm nằm lần sang phía bên kia của các đường hỗ trợ (hay đường margin, hoặc đường biên) tương ứng, ta gọi điểm đó rơi vào vùng không an toàn. Chú ý rằng vùng an toàn của hai lớp là khác nhau, giao nhau ở phần nằm giữa hai đường hỗ trợ.

Trong cả hai trường hợp trên, margin tạo bởi đường phân chia và đường nét đứt mảnh còn được gọi là *Soft margin* (biên mềm). Cũng theo cách gọi này, SVM thuần còn được gọi là *Hard margin SVM* (SVM biên cứng).



Hình 2.10 *Giới thiệu các biến slack ξ*

Với mỗi điểm x_n trong tập toàn bộ dữ liệu huấn luyện, ta giới thiệu thêm một biến đo sự hy sinh ξ_n tương ứng. Biến này còn được gọi là *slack variable.* Với những điểm x_n nằm trong *vùng an toàn*, ξ_n = 0. Với những điểm nằm trong *vùng không an toàn* như $x_1\ hay\ x_2\ hay\ x_3$, ta có ξ_n > 0. Nhận thấy rằng nếu $y_i = \pm 1$ là nhãn của x_i trong vùng không an toàn thì $\xi_i = |w^T x_i + b - y_i|$.

Với Soft Margin SVM, hàm mục tiêu sẽ có thêm một số hạng nữa giúp tối thiểu sự hy sinh. Từ đó ta có hàm mục tiêu:

$$\frac{1}{2}||w||_2^2 + C\sum_{n=1}^{N}\xi_n$$

Hằng số *C* được dùng để điều chỉnh tầm quan trọng giữa biên và sự hy sinh.

Ta sẽ có bài toán tối ưu ở dạng chuẩn cho *Soft-margin SVM*:

$$(w, b, \xi) = \underset{w,b,\xi}{\operatorname{argmin}}\left(\frac{1}{2}||w||_2^2 + C\sum_{n=1}^{N}\xi_n\right)$$

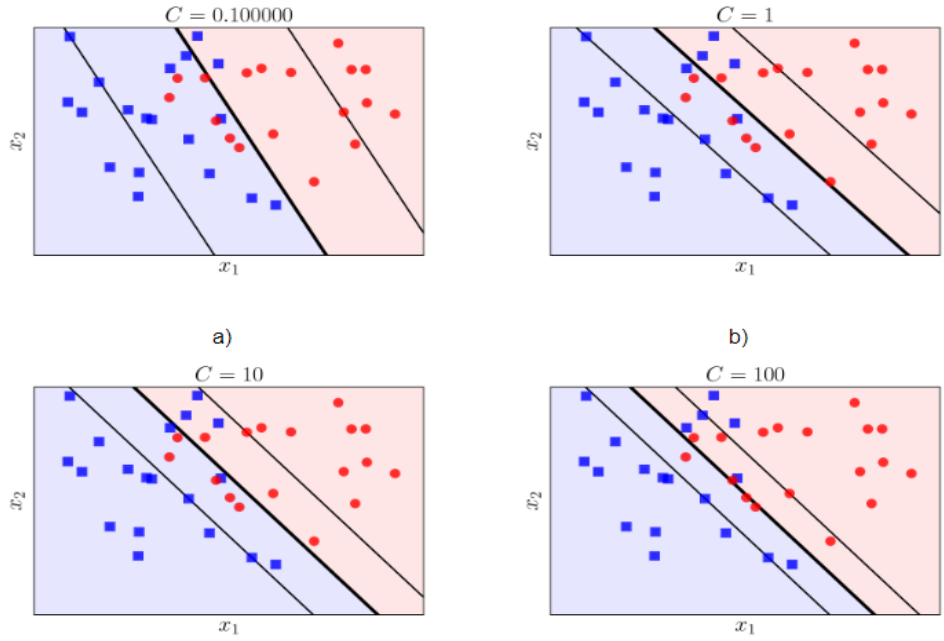
Với ràng buộc: $1 - \xi_n - y_i(w^T x_i + b) \leq 0,\ \forall n = 1, 2, \dots, N$

$$-\xi_n \leq 0, \forall n = 1, 2, \dots, N$$

Nếu C nhỏ, việc sự hy sinh cao hay thấp không gây ảnh hưởng nhiều tới giá trị của hàm mục tiêu, thuật toán sẽ điều chỉnh sao cho $||w||_2^2$ là nhỏ nhất, tức biên là lớn nhất, điều này sẽ dẫn tới $\sum_{n=1}^{N} \xi_n$ sẽ lớn theo.

Ngược lại, nếu C quá lớn, để hàm mục tiêu đạt giá trị nhỏ nhất, thuật toán sẽ tập trung vào làm giảm $\sum_{n=1}^{N} \xi_n$.

Trong trường hợp C rất rất lớn và hai classes là phân biệt tuyến tính, ta sẽ thu được $\sum_{n=1}^{N} \xi_n = 0$. Chú ý rằng giá trị này không thể nhỏ hơn 0. Điều này đồng nghĩa với việc không có điểm nào phải hy sinh, tức ta thu được nghiệm cho *Hard-margin SVM*. Nói cách khác, Hard-margin SVM chính là một trường hợp đặc biệt của Soft-margin SVM.

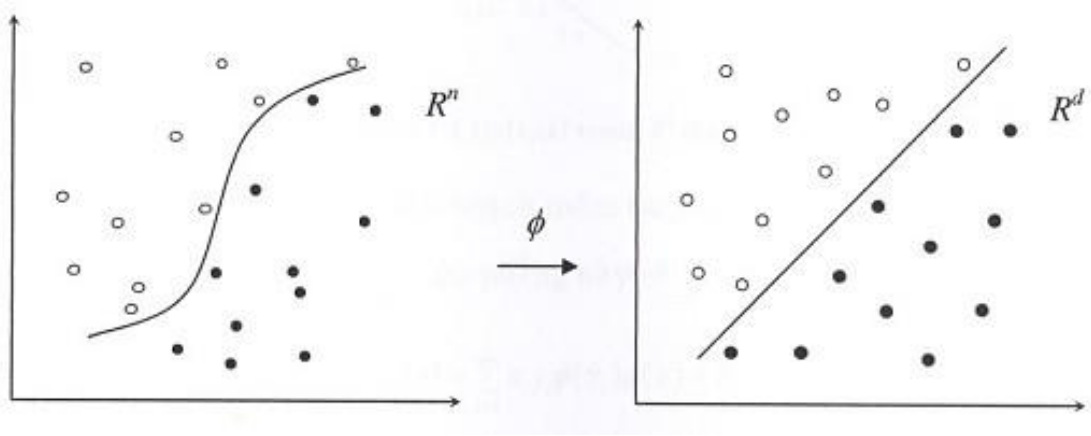


Hình 2.11 *Ảnh hưởng của C lên nghiệm.*

2.3.4 SVM phi tuyến.

Trong trường hợp tổng quát, thực tế mặt phân hoạch có thể là một mặt phi tuyến bất kỳ (Hình 2.11). Giả sử các mẫu xi thuộc không gian R^n, không gian này được gọi là không gian giả thiết (Hypothesis Space). Để tìm mặt phi tuyến trong không gian này, có thể áp dụng một thủ thuật là ánh xạ các vector mẫu x_i từ R^n vào một không gian

R^d có số chiều lớn hơn (d > n, d có thể bằng ∞). R^d được gọi là không gian đặc trưng (feature space). Sau đó áp dụng SVM tuyến tính để tìm ra một siêu phẳng phân hoạch trong không gian đặc trưng R^d. Siêu phẳng này sẽ ứng với mặt phi tuyến trong không gian R^n.



Hình 2.12 *Một mặt phân chia phi tuyến có thể trở thành một siêu phẳng trong không gian lớn hơn.*

Ánh xạ từ không gian R^n vào không gian R^d:

Gọi ánh xạ được áp dụng là ϕ, như vậy:

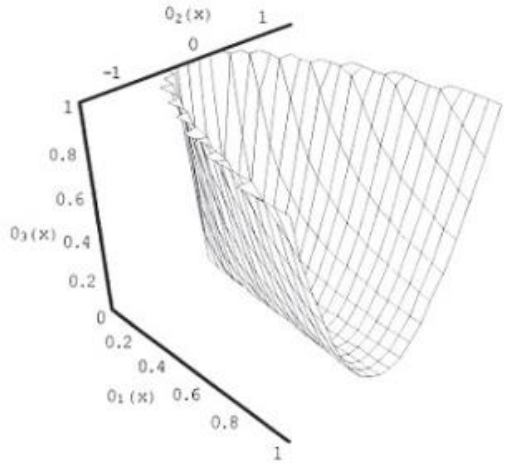
$$\phi : R^n \rightarrow R^d$$

$$x \longrightarrow \phi(x)$$

Ví dụ: x = (x_1, x_2) ∈ R^2. Định nghĩa hàm ánh xạ : $R^2 \rightarrow R^3$ như sau:

$$\phi(x) = \left({x_1}^2, \sqrt{2}x_1x_2, {x_2}^2\right)$$

Với ánh xạ trên thì mặt hình vuông [-1, 1] x [-1, 1] trong không gian R^2 sẽ trở thành một mặt cong trong không gian R^3 như trong hình 2.12. Bây giờ dùng một mặt phẳng trong không gian R^3 này thì có thể chia mặt cong trên thành hai phần (mà trong không gian R^2 thì phải dùng một đường cong mới được kết quả phân chia tương ứng).



Hình 2.13 *Mặt phẳng [-1, 1] x [-1, 1] trong R^2 thành mặt cong trong R^3*

Áp dụng siêu phẳng phân hoạch mềm (soft- margin hyperplane) trong không gian R^d cho các mẫu $\phi(x_i)$ thì siêu phẳng này sẽ là:

$$f\big(\phi(x)\big) = \sum_{i=1}^{l} u_i y_i \phi(x_i)\phi(x) + b = 0$$

Từ đó cũng có hàm phân hoạch trong không gian R^n là hàm hợp $f_0(\phi)$. Đây là hàm phi tuyến.

Cách tiếp cận này gặp phải vấn đề là hàm $\phi(x)$ có thể có số chiều rất lớn (nếu không gian R^d có *d* lớn). Và do đó tiêu tốn thời gian nhiều trong các phép tính. Tuy nhiên, có thể nhận xét rằng trong các phép tính $\phi(x)$ chỉ xuất hiện dưới dạng tích vô hướng tức là dạng $\phi(x)\phi(y)$ mà không xuất hiện đơn lẻ. Đây là một nhận xét quan trọng trong việc tìm ra quy tắc sau:

Thay vì sử dụng dạng tường minh của $\phi(x)$ thì chỉ cần sử dụng hàm biểu diễn giá trị vô hướng $\phi(x)\phi(y)$.

Đặt $K(x, y) = \phi(x)\phi(y)$ được gọi là *hàm hạt nhân (kernel function).*

Như vậy là chỉ cần biết dạng của hàm hạt nhân $K(x, y)$ mà không cần biết cụ thể ánh xạ $\phi(x)$. Lúc đó hàm nhận dạng trở thành:

$$f \circ \phi(x) = \sum_{i=1}^{l} u_i y_i K(x_i, x) + b$$

2.3.5 Lựa chọn nhân SVM.

Có rất nhiều hàm nhân trong SVM, vậy làm thế nào để chọn một hàm nhân tốt cũng là một vấn đề. Mặc dù, mục đích chung, có một số hàm nhân phổ biến:

- Nhân Linear: $K(x_i, x_j) = {x_i}^T x_j$
- Nhân Polynomial: $K(x_i, x_j) = (\gamma {x_i}^T x_j + r)^d,\ \gamma > 0$
- Nhân RBF (Radial Baisis Function): $K(x_i, x_j) = \exp\left(-\gamma \|x_i - x_j\|^2\right), \gamma > 0$
- Nhân Sigmoid: $K(x_i, x_j) = \tanh(\gamma {x_i}^T x_j + r)$

Ở đây γ, r, d là tham số hạt nhân, trong những hàm nhân phổ biến, *RBF* là hàm nhân chính vì những lý do sau:

- Biểu đồ mẫu của nhân RBF trong một không gian nhiều chiều hơn không giống như nhân Linear (nhân tuyến tính).
- Nhân RBF có ít siêu tham số hơn so với nhân Polynomial (nhân đa thức).
- Nhân RBF có ít khó khăn hơn trong việc xử lý và giải quyết bài toán hàm mất mát.

2.3.6 Các ưu điểm chính của SVM.

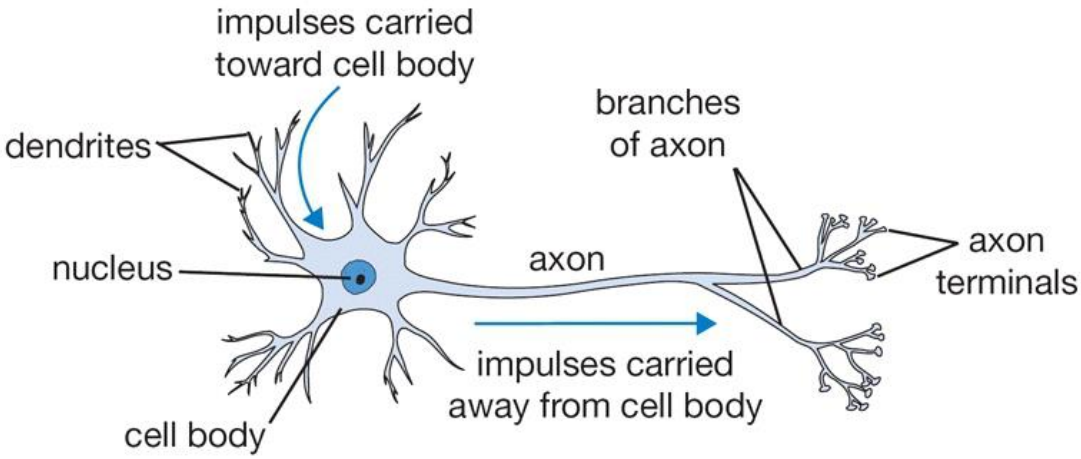
- SVM được sử dụng trong các tình huống của mẫu dữ liệu hữu hạn. Nó nhằm mục đích để được tối ưu giải pháp dựa trên các thông tin hiện tại chứ không phải là giá trị tối ưu khi số mẫu có xu hướng là vô hạn.
- Các thuật toán cuối cùng được chuyển thành tối ưu hóa các chương trình bậc hai. Về mặt lý thuyết, nó sẽ nhận được một giá trị tối ưu hóa toàn cầu, mà không thể tránh khỏi giải quyết vấn đề tối ưu hóa địa phương trong khi sử dụng mạng thần kinh.
- Các thuật toán thực hiện một ánh xạ phi tuyến từ không gian dữ liệu ban đầu vào một số không gian chiều cao đặc trưng. Trong đó nó xây dựng một biểu

thức tuyến tính chức năng để thay thế các hàm phi tuyến trong không gian dữ liệu ban đầu. Điều này đảm bảo rằng SVM có khả năng khái quát hóa tốt. Đồng thời, nó giải quyết các vấn đề của thảm họa kích thước vì sự phức tạp tính toán của nó là độc lập với kích thước mẫu.

2.4 Mô hình CNN (Convolutional Neural Network).

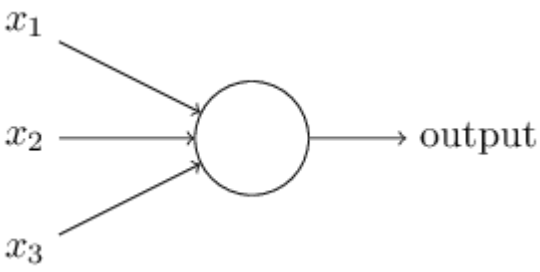
2.4.1 Perceptron cơ bản.

Một mạng nơ-ron được cấu thành bởi các nơ-ron đơn lẻ được gọi là các perceptron. Nên trước tiên ta tìm hiểu xem perceptron là gì đã rồi tiến tới mô hình của mạng nơ-ron sau. Nơ-ron nhân tạo được lấy cảm hứng từ nơ-ron sinh học như hình mô tả bên dưới:



Hình 2.14 *Mạng nơ-ron nhân tạo. Nguồn: https://cs231n.github.io/*

Như hình trên, ta có thể thấy một nơ-ron có thể nhận nhiều đầu vào và cho ra một kết quả duy nhất. Mô hình của perceptron cũng tương tự như vậy:



Hình 2.15 *Perceptron.*

Một perceptron sẽ nhận một hoặc nhiều đầu **x** vào dạng nhị phân và cho ra một kết quả *o* dạng nhị phân duy nhất. Các đầu vào được điều phối tầm ảnh hưởng bởi các tham số trọng lượng tương ứng **w** của nó, còn kết quả đầu ra được quyết định dựa vào một ngưỡng quyết định (threshold) *b* nào đó:

$$o = \begin{cases} 0 & if \ \sum_i w_i x_i \ \leq threshold \\ 1 & if \ \sum_i w_i x_i > threshold \end{cases}$$

Đặt $b = -threshold$, ta có thể viết lại thành:

$$o = \begin{cases} 0 & if \ \sum_i w_i x_i + b \leq 0 \\ 1 & if \ \sum_i w_i x_i + b > 0 \end{cases}$$

Để dễ hình dung, ta lấy ví dụ việc đi đá bóng hay không phụ thuộc vào 4 yếu tố sau:

(1) Trời có nắng hay không? (2) Có hẹn trước hay không?

(3) Bố mẹ có vui hay không? (4) Bạn đá bóng có đi đủ không?

Ta coi bốn yếu tố đầu vào $x_1, \ x_2, \ x_3, \ x_4$ và nếu $o = 0$ thì ta không đi đá bóng, còn nếu o = 1 thì ta đi đá bóng. Giả sử mức độ quan trọng của bốn yếu tố trên lần lượt là $w_1 = 0.05, \ w_2 = 0.5, \ w_3 = 0.2, \ w_4 = 0.25$ và chọn ngưỡng b = -0.5 thì ta có thể thấy rằng việc trời nắng có ảnh hưởng chỉ đến 5% tới quyết định đi đá bóng và việc có hẹn từ trước quyết định tới 50% việc đi đá bóng hay không.

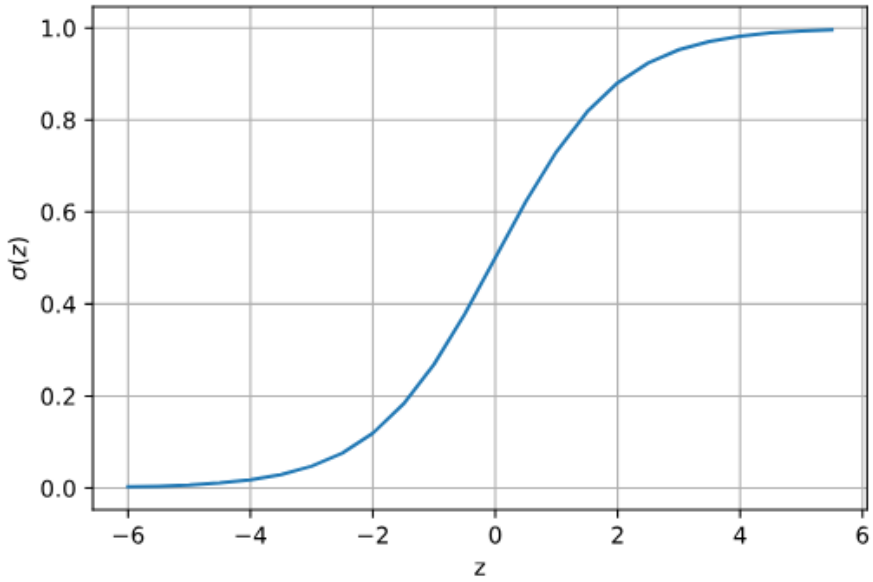
2.4.2 Sigmoid neurals.

Với đầu vào và đầu ra dạng nhị phân, ta rất khó có thể điều chỉnh một lượng nhỏ đầu vào để đầu ra thay đổi chút ít, nên để linh động, ta có thể mở rộng chúng ra cả khoảng [0, 1]. Lúc này đầu ra được quyết định bởi một hàm $sigmoid \ \sigma(\boldsymbol{w^T x})$.

Công thức hàm sigmoid:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Đồ thị của hàm này cũng cân xứng rất đẹp thể hiện được mức độ công bằng của các tham số:



Hình 2.16 *Đồ thị hàm sigmoid.*

Đặt $\boldsymbol{z = w^T x}$ thì công thức perceptron sẽ có dạng:

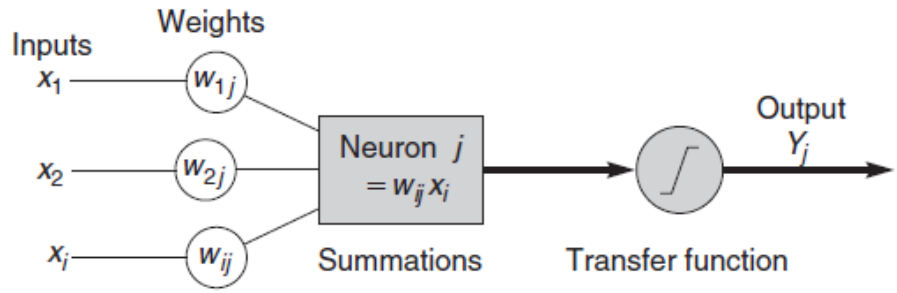
$$o = \sigma(z) = \frac{1}{1 + \exp(-\boldsymbol{w^T x})}$$

Ta có thể thấy rằng mỗi sigmoid neuron cũng tương tự như một bộ phân loại tuyến tính *(logistic regression)* bởi xác suất: $P(y_i = 1 \,|x_i; \boldsymbol{w}) = o(\boldsymbol{w^T x})$

Thực ra thì ngoài hàm sigmoid ra, ta còn có thể một số hàm khác như *tanh, ReLU* để thay thế hàm sigmoid bởi dạng đồ thị của nó cũng tương tự như sigmoid. Một cách tổng quát, hàm perceptron được biểu diễn qua một *hàm kích hoạt* (activation function) *f(z)* như sau:

$$o = f(z) = f(\boldsymbol{w^T x})$$

Bằng cách biểu diễn như vậy, ta có thể coi nơ-ron sinh học được thể hiện như sau:

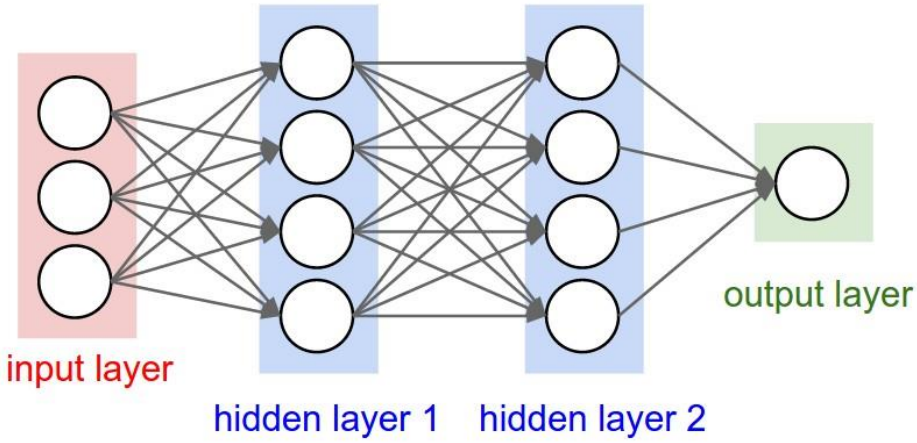


Hình 2.17 *Mô hình nơ-ron*

Các hàm kích hoạt buộc phải là hàm phi tuyến. Vì nếu nó là tuyến tính thì khi kết hợp với phép toán tuyến tính $\boldsymbol{w^T x}$ thì kết quả thu được cũng sẽ là một thao tác tuyến tính dẫn tới việc nó trở nên vô nghĩa.

2.4.3 Kiến trúc mạng nơ-ron.

Neural network là sự kết hợp của của các tầng perceptron hay còn được gọi là perceptron đa tầng (multilayer perceptron) như hình vẽ bên dưới:



Hình 2.18 *Neural Network – 2 Lớp ẩn*

Một mạng nơ-ron sẽ có 3 kiểu tầng:

- **Lớp đầu vào** *(Input layer)*: Là tầng bên trái cùng của mạng thể hiện cho các đầu vào của mạng.

- **Lớp ra** *(Output layer)*: Là tầng bên phải cùng của mạng thể hiện cho các đầu ra của mạng.
- **Lớp ẩn** *(Hidden layer)*: Là tầng nằm giữa tầng vào và tầng ra thể hiện cho việc suy luận logic của mạng.

Một mô hình mạng nơ-ron chỉ có 1 lớp đầu vào, 1 lớp đầu ra nhưng có thể có rất nhiều lớp ẩn.

Trong mạng nơ-ron, mỗi nút mạng là một sigmoid nơ-ron nhưng hàm kích hoạt của chúng có thể khác nhau. Tuy nhiên trong thực tế người ta thường để chúng cùng dạng với nhau để tính toán cho thuận lợi.

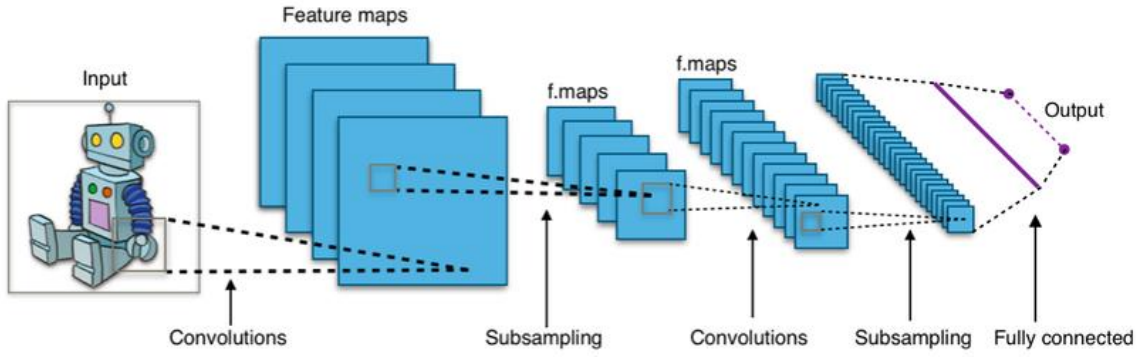
Ở mỗi tầng, số lượng các nút mạng (nơ-ron) có thể khác nhau tuỳ thuộc vào bài toán và cách giải quyết. Nhưng thường khi làm việc người ta để các tầng ẩn có số lượng nơ-ron bằng nhau. Ngoài ra, các nơ-ron ở các tầng thường được liên kết đôi một với nhau tạo thành mạng kết nối đầy đủ (full-connected network). Khi đó ta có thể tính được kích cỡ của mạng dựa vào số tầng và số nơ-ron. Ví dụ ở hình trên ta có:

4 tầng mạng, trong đó có 22 tầng ẩn.

3 + 4*2 + 1 = 12 nút mạng.

(3*4 + 4*4 + 4*1) + (4 + 4 + 1) = 41 tham số.

2.4.4 Giới thiệu về mô hình CNN.



Hình 2.19 *Mô hình CNN.*

- **Feature (Đặc trưng):**

CNN so sánh hình ảnh theo từng mảnh, các mảnh mà nó tìm thấy được gọi là các *feature* (đặc trưng). Bằng cách tìm ở mức thô các feature khớp nhau ở cùng vị trí trong hai hình ảnh, CNN sẽ nhận ra sự tương đồng giữa hai ảnh sẽ tốt hơn so với việc khớp toàn bộ bức ảnh.

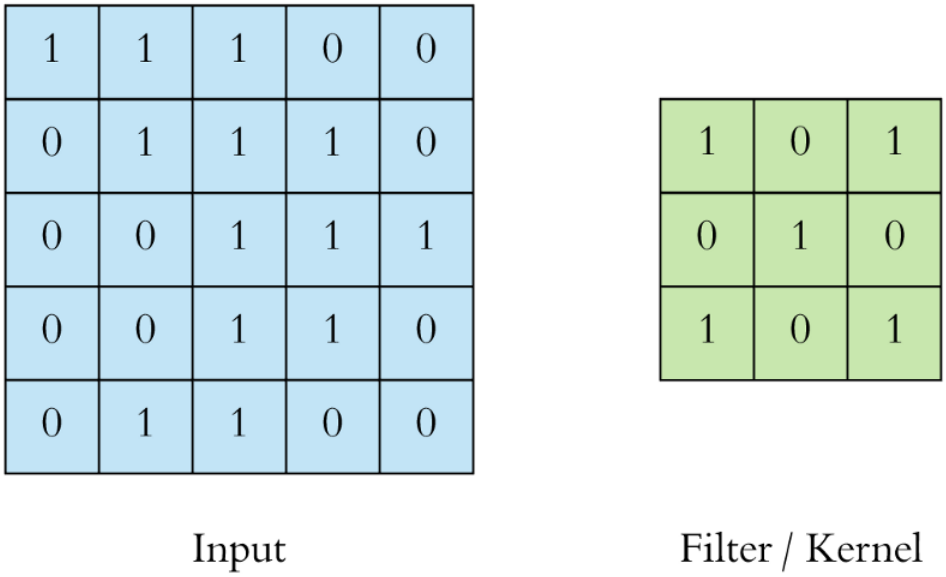
Mỗi feature giống như những hình ảnh mini – mảng hai chiều nhỏ. Các feature khớp với các khía cạnh chung của bức ảnh.

- **Convolution (Tích chập):**

Khi kiểm thử một ảnh mới, mô hình CNN sẽ không thể biết được chính xác nơi các feature này sẽ khớp nên nó sẽ thử chúng khắp mọi nơi, mọi vị trí mà nó có thể xét đến. Khi tính toán sự khớp nhau của một feature trên toàn bộ ảnh chúng ta làm thành một filter (bộ lọc).

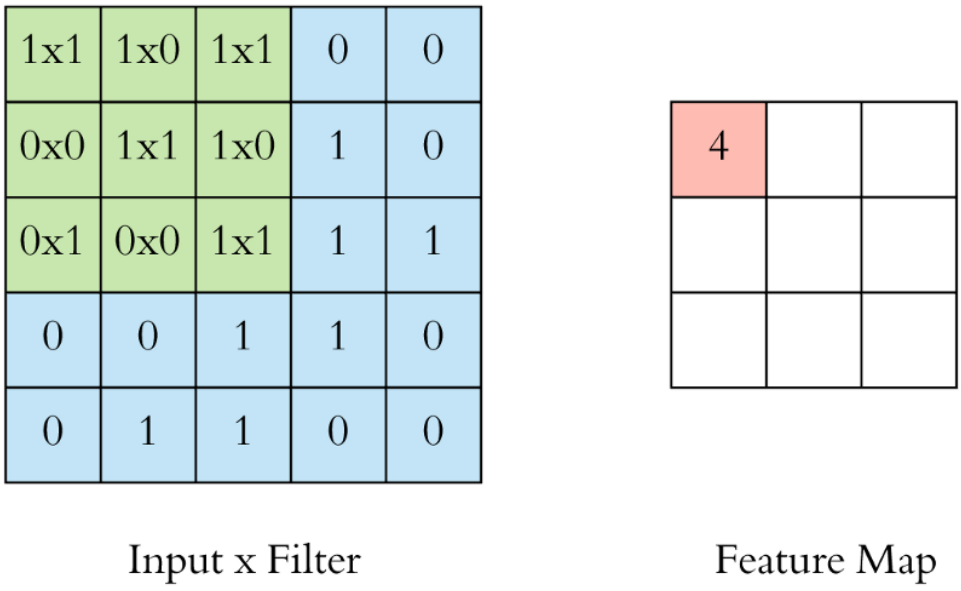
Convolutional gồm hai khái niệm khác là Convolution Filter và Convolutional Layer. Trong mạng neural network thông thường, từ input, rồi đi qua các hidden layer rồi ra được output. Với CNN, Convolutional Layer cũng chính là hidden layer, khác ở chỗ, Convolutional Layer là một tập các *feature map* và mỗi *feature map* này là một bản scan (quét) của input ban đầu, nhưng được trích xuất ra các đặc trưng cụ thể. Quét như thế nào thì lại dựa vào Convolution Filter hay kernel (nhân). Đây là một ma trận sẽ quét qua ma trận dữ liệu đầu vào, từ trái qua phải, trên xuống dưới, và nhân tương ứng từng giá trị của ma trận đầu vào mà ma trận kernel rồi cộng tổng lại, đưa qua hàm kích hoạt (*activation funciton*) (sigmoid, relu, elu, ...), kết quả sẽ là một con số cụ thể, tập hợp các con số này lại là một ma trận nữa, chính là *feature map*.

Hãy nhìn vào ví dụ sau cho dễ hiểu: Chúng ta có một ma trận đầu vào input và một kernel:



Hình 2.20 *Ma trận Input và Kernel.*

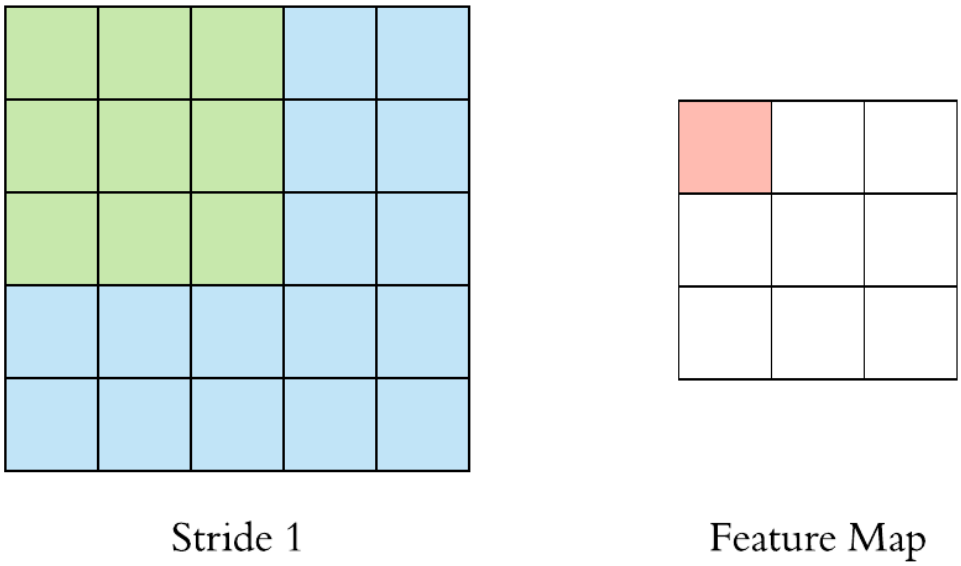
Ta sẽ quét kernel qua từng phần tử của input. Và tính toán như trên: nhân tương ứng, rồi cộng tổng kết quả, đưa qua hàm kích hoạt, kết quả thu được là một giá trị tại *feature map*.



Hình 2.21 *Ma trận Input x Filter và Feature Map.*

- **Stride and Padding:**

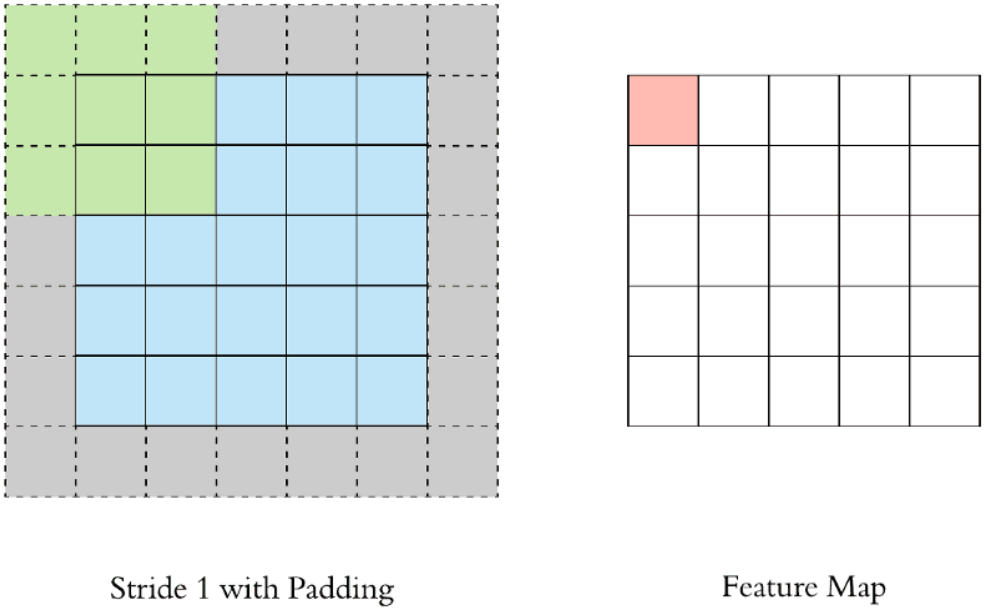
Stride là khoảng cách giữa hai kernel khi quét. Với stride = 1, kernel sẽ quét hai ô ngay cạnh nhau, nhưng với stride = 2, kernel sẽ quét ô số 1 và ô số 3. Bỏ qua ô ở giữa. Điều này nhằm tránh việc lặp lại giá trị ở các ô bị quét. *Stride = 1.*



Hình 2.22 *Ví dụ về stride = 1.*

Nếu chọn *stride* và *size* của *kernel* càng lớn thì *size* của *feature map* càng nhỏ, một phần lý do đó là bởi kernel phải nằm hoàn toàn trong input. Có một cách để giữ nguyên kích cỡ của feature map so với ban đầu. Đấy là *Padding*. Khi điều chỉnh padding = 1, tức là đã thêm một ô bọc xung quanh các cạnh của input, muốn phần bọc này càng dày thì cần phải tăng padding lên.

Hãy nhìn vào ví dụ sau, ta xét padding = 1:

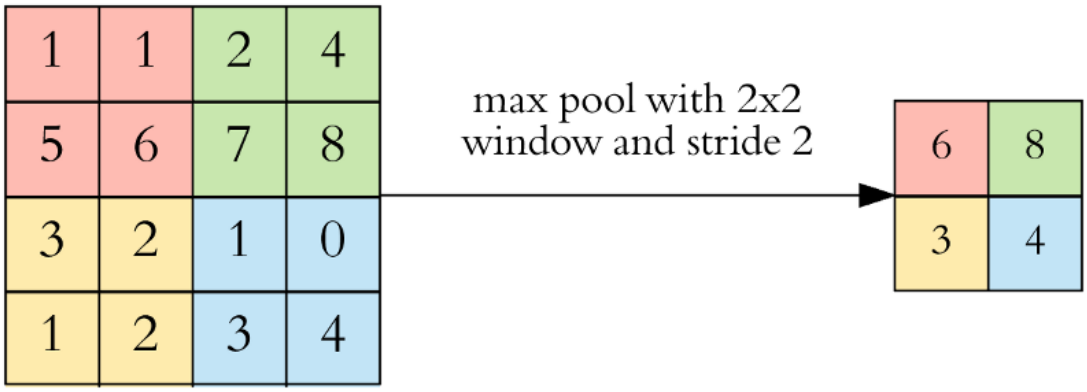


Hình 2.23 *Ví dụ về stride 1 với padding.*

Phần màu xám chính là phần bọc thêm vào input. Với stride=1 và padding=0, từ bức ảnh input ban đầu, mô hình sẽ quét kernel qua và tạo thành các ô như sau để *map* thành *feature map.*

- **Pooling:**

Pooling là một cách lấy những hình ảnh lớn và làm co chúng lại trong khi vẫn giữ được những thông tin quan trọng nhất. Mục đích của *pooling* rất đơn giản, nó làm giảm số *hyperparameter* (siêu tham số) mà ta cần phải tính toán, từ đó giảm thời gian tính toán, tránh *overfitting*. Loại *pooling* thường gặp nhất là *max pooling*, lấy giá trị lớn nhất trong một cửa sổ pool. 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 giá trị nằm trong cửa sổ trượt (với max pooling ta sẽ lấy giá trị lớn nhất). Hãy cùng nhìn vào ví dụ sau, ta chọn *pooling window* có kích thước là 2 * 2, *stride* = 2 để đảm bảo không trùng nhau, và áp dụng max pooling:

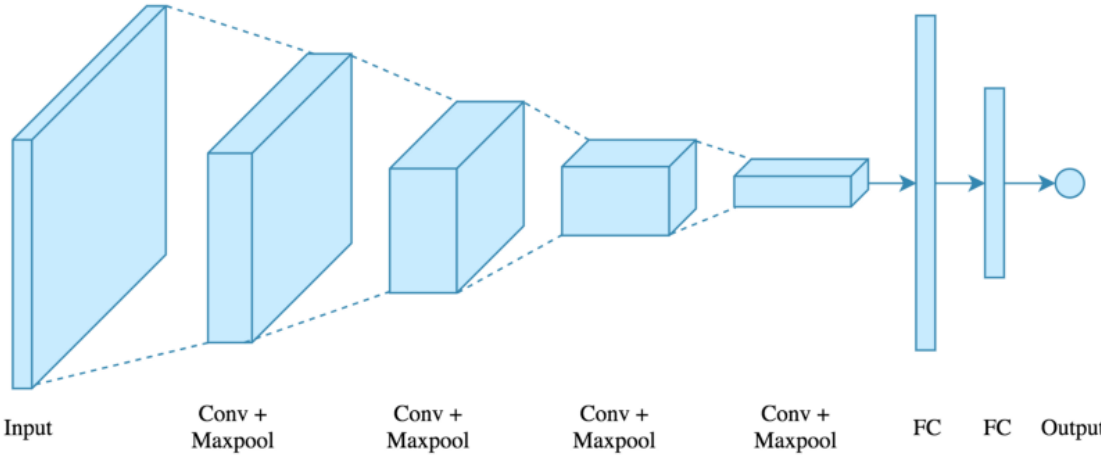


Hình 2.24 *Ví dụ về Pooling.*

- **Fully Connected:**

Trong Neural Network thì là một mạng có đầy đủ các kết nối với nhau. Layer này cũng chính là một fully connected Artificial Neural Network. Thường thì sau các lớp Convolutional + Pooling thì sẽ là hai lớp Fully connected, một layer để tập hợp các *feature layer* mà đã được tìm ra, chuyển đổi dữ liệu từ 3D, hoặc 2D thành 1D, tức chỉ còn là một vector. Còn một layer nữa là output, số nơ-ron của tầng này phụ thuộc vào số output mà ta muốn tìm ra. Giả sử với tập dữ liệu MNIST chẳng hạn, ta có tập các số viết tay từ 0 → 9. Vậy output sẽ có số neuron là 10.

- **CNN Structure:**



Hình 2.25 *Cấu trúc của CNN.*

Hình trên đã biểu diễn rất rõ ràng kiến trúc của một mạng CNN. Lớp đầu tiên là lớp ảnh đầu vào, qua các lớp Convolutional và Maxpool, đầu ra sẽ tiếp tục được

đưa vào mạng liên kết đầy đủ (fully-connected layer). Lớp Maxpool thường nằm sau mỗi quá trình của lớp Convolutional.

- **Lan truyền ngược (Backpropagation):**

Bài toán sử dụng mô hình mạng nơ-ron chúng ta đang đề cập là việc tính toán theo dạng forward (tiến theo hướng phía trước). Tuy nhiên nếu chỉ sử dụng như vậy thì mô hình sẽ chưa thể hoàn chỉnh, vì vậy chúng ta cần phải có một phương pháp để cập nhật các tham số như trọng số để mô hình có thể đạt kết quả tốt hơn cho những bộ kiểm thử khác. Trong chương trình học máy, tồn tại một phương pháp giúp ta có thể giải quyết được bài toán này, đó là phương pháp lan truyền ngược (backpropagation).

Để có thể sử dụng lan truyền ngược, chúng ta cần có một tập dữ liệu đã được gán nhãn bằng phương pháp thủ công hoặc tự động nào đó. Ban đầu chúng ta sử dụng chúng với mô hình CNN chưa được huấn luyện, tức là các tham số khởi tạo ban đầu đều được chọn ngẫu nhiên. Tuy nhiên việc chọn ngẫu nhiên này cũng có ảnh hưởng nhiều đến tốc độ giải quyết bài toán. Nếu giá trị khởi tạo ban đầu gần với giá trị thực cần tìm thì mô hình sẽ được học rất nhanh, và ngược lại nếu khởi tạo ban đầu khá chênh lệch thì mô hình sẽ phải học lâu hơn. Chúng ta sẽ lựa chọn một số lượng ảnh đầu vào từng đợt train (batch_size) để cho mô hình tự tối ưu. Mỗi lần huấn luyện chúng ta sẽ thu được một đầu ra trọng số và kết quả dự đoán của mô hình. Hàm mất mát được hình thành để đánh giá độ sai lệch giữa kết quả đầu ra và kết quả thực tế của dữ liệu vừa được huấn luyện. Chúng ta sẽ cần phải tối ưu hóa sự mất mát này bằng một phương pháp nào đó, ví dụ như AdamOptimizer, Gradient Descent,… Hàm mất mát thường được sử dụng trong những mô hình học máy là cross-entropy và hàm sai số bình phương trung bình. Sau khi tính được đạo hàm theo các tham số, chúng ta sẽ cập nhật các trọng số mới theo tốc độ học trong hàm tối ưu mất mát.

2.5 Tiêu chí đánh giá các mô hình.

Có rất nhiều phương pháp phân lớp đã được sử dụng trong những năm qua:

- Phương pháp phân lớp Logistic.
- Phương pháp phân lớp Probit.
- Phân lớp cây quyết định (Decision tree classification).
- Mô hình phân lớp SVM (Support Vector Machine).
- Mạng nơ-ron (Neural Network).
- Phân tích thống kê (Naïve Bayes).
- Các thuật toán di truyền.
- Mô hình phân lớp K - hàng xóm gần nhất (K- Nearest Neightbor Classifier) (KNN).
- Phương pháp tập thô (Rough set Approach).

Trong từng ứng dụng cụ thể cần lựa chọn mô hình phân loại phù hợp. Việc lựa chọn đó căn cứ vào sự so sánh các mô hình phân loại với nhau, dựa vào các tiêu chí sau:

- ***Độ chính xác dự đoán*** *(Predictive Accuracy):*

Độ chính xác là khả năng của mô hình để dự đoán chính xác nhãn lớp của dữ liệu mới hay dữ liệu chưa biết.

- ***Tốc độ*** *(Speed):*

Tốc độ là những chi phí tính toán liên quan đến quá trình tạo ra và sử dụng mô hình.

- ***Sức mạnh*** *(Robustness):*

Sức mạnh là khả năng mô hình tạo ra các dự đoán đúng với các dữ liệu nhiễu và dữ liệu với giá trị không đầy đủ.

- ***Khả năng mở rộng*** *(Scalability):*

Khả năng mở rộng là khả năng xây dựng mô hình một cách hiệu quả với các 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 liên quan đến kích thước của cây quyết định hay độ cô đọng của các luật.

Chương 3 Tìm lỗi của camera theo ảnh mẫu

3.1 Dữ liệu cho bài toán.

Đồ án này thực hiện việc phân loại với bốn trường hợp: Case Blur, Case Flare, Case Flash, Case White. Cơ sở dữ liệu là các ảnh chụp (đã được phân loại đạt và không đạt) được cung cấp từ nhà máy nên mang tính đặc thù. Mỗi ảnh của các trường hợp được các nhân viên sản xuất của nhà máy thực hiện bằng việc dùng điện thoại và chụp các khung đồ hình có những loại kích thước nhất định, theo mỗi độ cao tùy vào từng nhân viên nhưng phải chụp bao quát được hết đồ hình và ảnh đồ hình gần khít với khung ảnh chụp của camera. Góc chụp các đồ hình là theo phương thẳng đứng, điện thoại được đặt trên khung giá đỡ. Vì thực hiện thủ công nên có những trường hợp nhân viên sản xuất đặt lệch điện thoại so với chuẩn quy định. Vì vậy có thể ảnh hưởng đến dữ liệu ảnh đầu vào. Tuy nhiên vì nhân viên sản xuất đã được đào tạo đặc thù công việc nên phần lớn các ảnh đều được chụp đúng chuẩn quy định.

Kích thước ảnh đầu vào gồm các ảnh có kích thước như sau:

3264 x 2448 ; 4032 x 3024 ; 4608 x 3456 ; 5665 x 4248 pixels

Sau khi xử lý cắt các khu vực được xét là đặc trưng của hai trường hợp Blur và Flare, kích thước bộ tập ảnh dùng cho việc huấn luyện là:

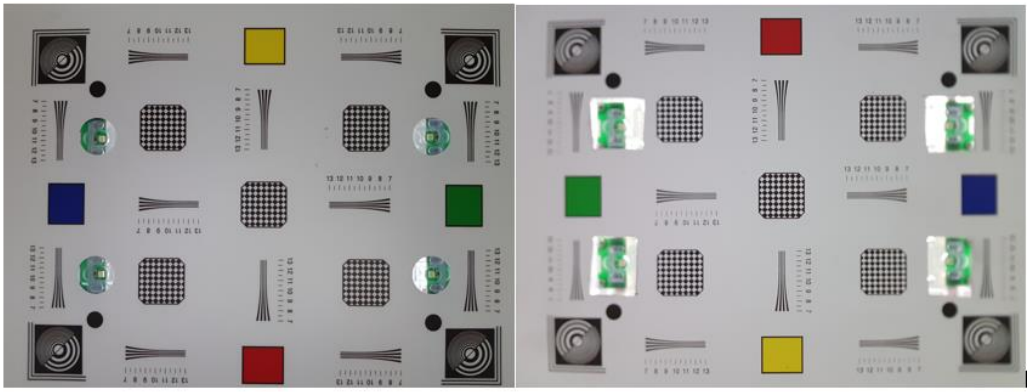
Blur: 80 x 300 ; Flare: 300 x 300.

Trường hợp	Tập huấn luyện		Tập kiểm thử	
	Ảnh NG	Ảnh OK	Ảnh NG	Ảnh OK
Blur	708	732	50	50
Flare	819	1721	50	50
Flash	78	225	50	53

White	117	101	61	50

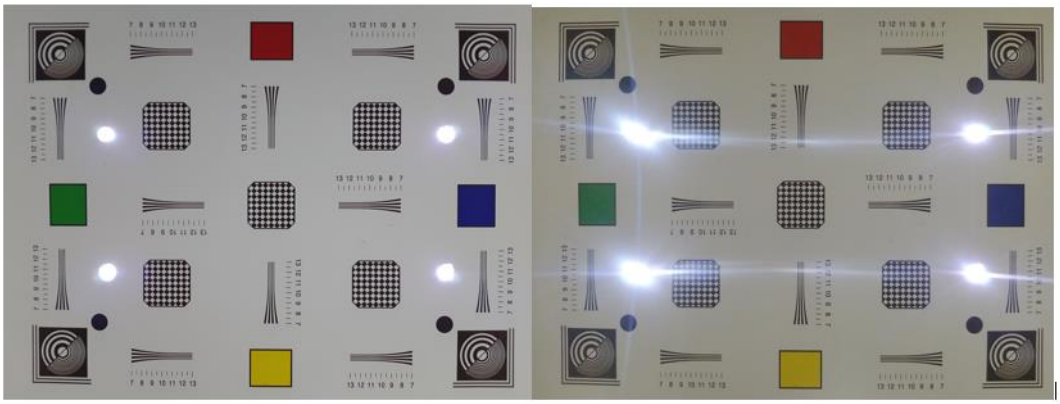
Bảng 3.1 *Dữ liệu ảnh của bài toán.*

- **Case Blur:** Ảnh chụp nền đồ hình điều kiện bình thường.



Hình 3.1 *Ví dụ về case Blur.*

- **Case Flare:** Ảnh chụp nền đồ hình với đèn led được bật.



Hình 3.2 *Ví dụ về case Flare.*

- **Case Flash:** Ảnh chụp nền đồ hình kiểm tra đèn flash được bật hay không.

Hình 3.3 *Ví dụ về case Flash.*

- **Case White:** Kiểm tra camera có bị lỗi khi chụp nền đồ hình màu trắng.

Hình 3.4 *Ví dụ về Case White.*

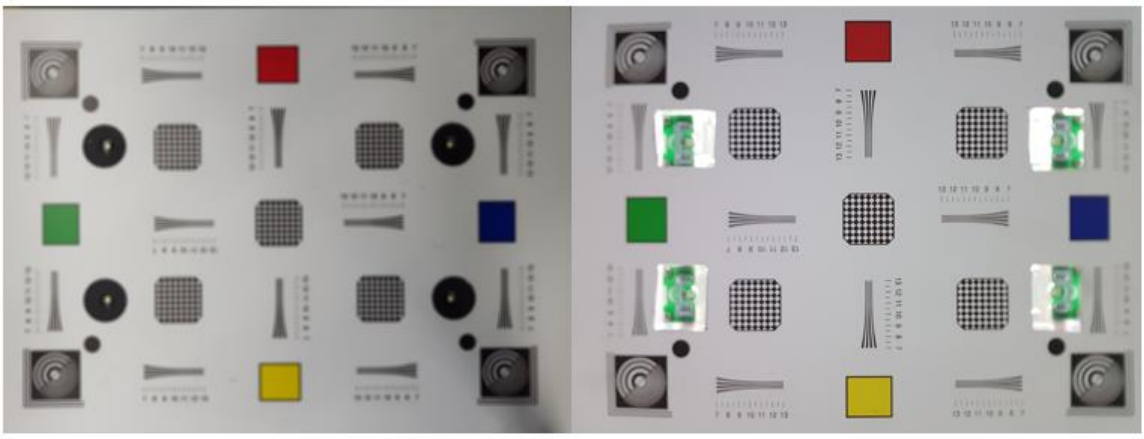
3.2 Tiền xử lý dữ liệu và trích xuất đặc trưng.

3.2.1 Case Blur.

Ảnh thỏa mãn là ảnh nhận diện được các ô màu sắc, có thể nhận diện được các số, hình biểu đồ không bị lẫn vào nhau. Ngược lại, ảnh sẽ bị phân là không thỏa mãn điều kiện.

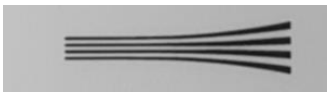
Tập ảnh lỗi có 2 trường hợp có thể xảy ra:

- Ảnh bị mờ toàn bộ.
- Ảnh bị mờ một phần.



Hình 3.5 *Ví dụ về hai trường hợp mờ của ảnh.*

Dựa trên thiết kế của đồ hình, việc xác định độ mờ của ảnh sẽ được chia nhỏ thành xác định độ mờ ở 12 khu vực như hình 3.5.

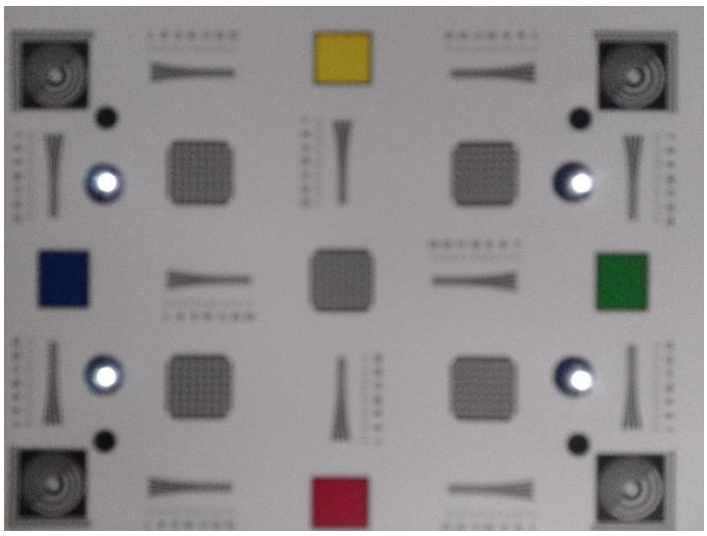


Hình 3.6 *Khu vực xác định đặc trưng mờ.*

3.2.1.1 Tiền xử lý – cắt ra các ảnh nhỏ.

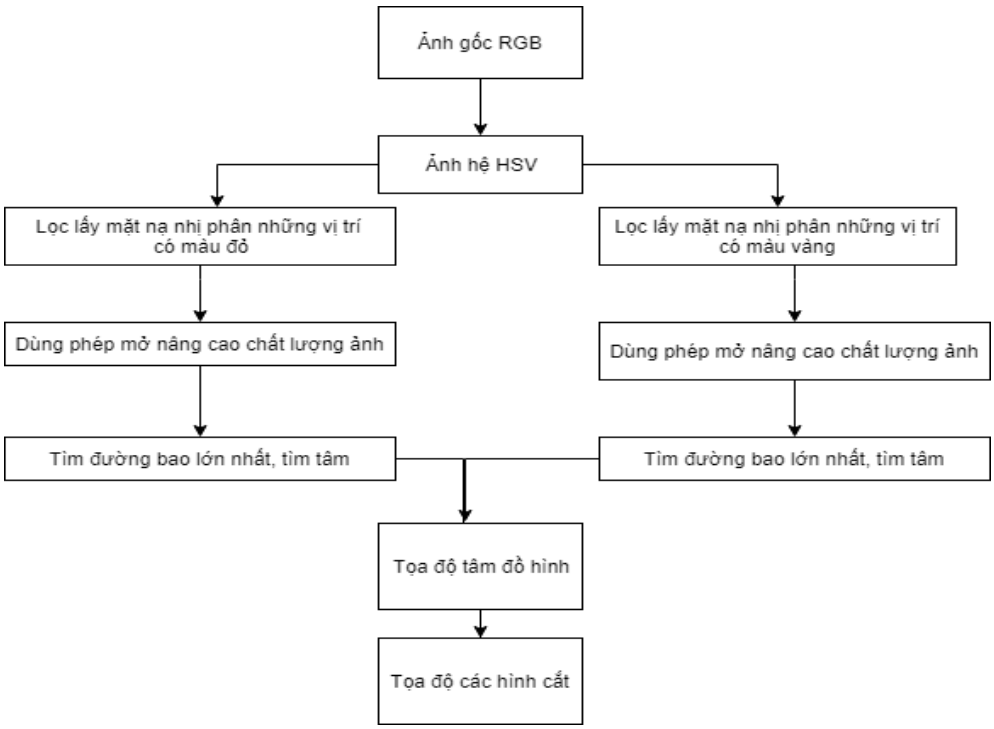
Để cắt ra 12 ảnh con thì bước đầu tiên ta cần xác định tâm của đồ hình trong ảnh.

Tập ảnh bao gồm kích thước 3264x2448 và 4032x3024 và được chụp thẳng đứng, tuy nhiên sẽ có những bức ảnh tâm đồ hình không đạt được chính giữa bức ảnh mà bị chụp lệch đi như Hình 3.7.

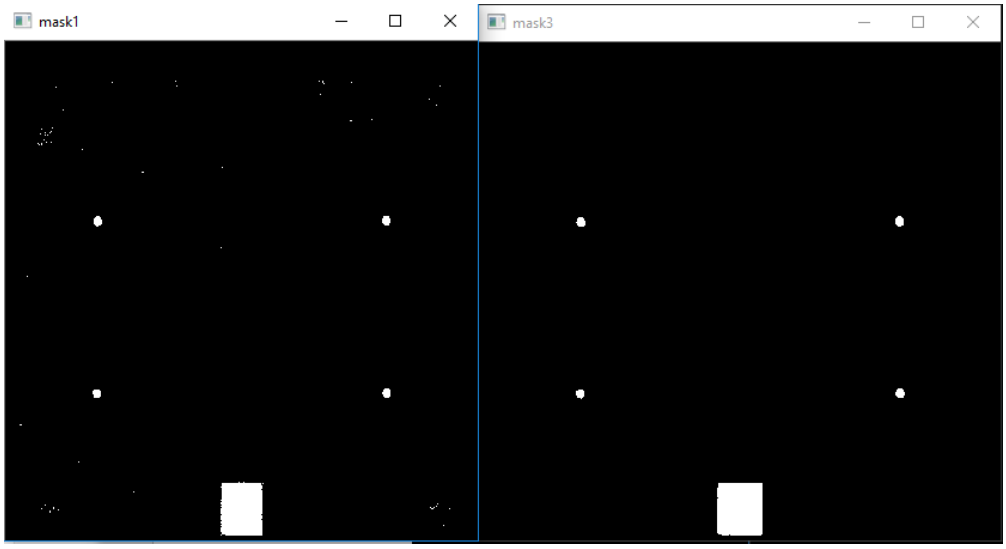


Hình 3.7 *Một trường hợp ảnh chụp không đạt chính giữa tâm đồ hình.*

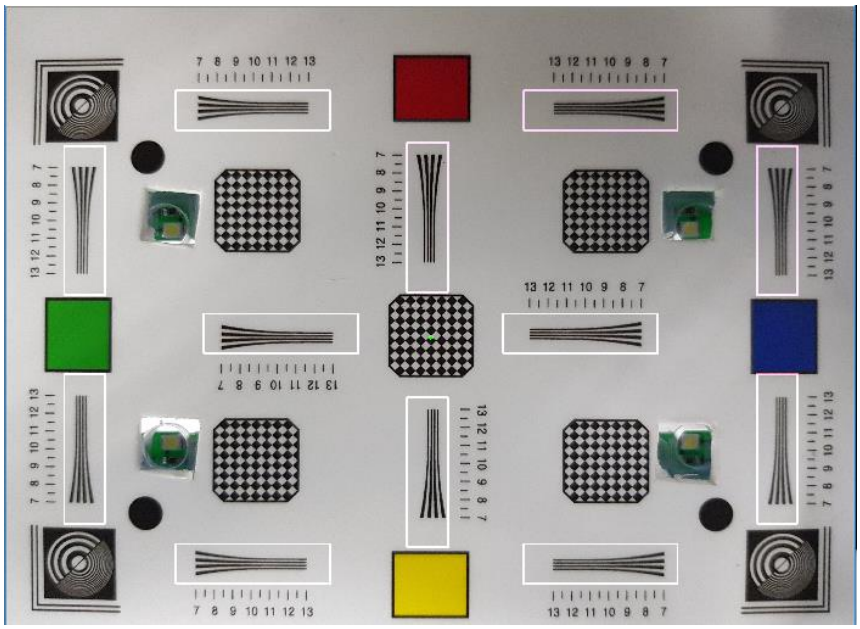
Đồ án đề xuất tìm tâm thông qua việc xác định tâm là trung điểm của tâm hai hình màu vàng và đỏ. Ảnh sẽ đi qua một mặt nạ chuyển qua hệ màu HSV sau đó lọc với màu vàng, sử dụng hàm findContour của OpenCV và xác định tâm ô màu vàng. Tương tự với ô vuông đỏ. Bước tiếp theo, ta chỉ cần cắt ra các ảnh nhỏ được dựa vào tỉ lệ so với tâm đồ hình vừa tìm được.



Hình 3.8 *Quá trình thực hiện cắt ảnh cho case blur.*



Hình 3.9 *Kết quả sau khi sử dụng phép mở giảm nhiễu.*



Hình 3.10 *Tọa độ tâm và khu vực các hình cắt.*

3.2.1.2 Trích xuất đặc trưng.

Để dễ dàng hơn cho việc xử lý, ảnh nhỏ được chuyển thành ảnh xám. Việc chuyển đổi này không làm mất đi tính chất và các đặc trưng của ảnh. Đặc điểm của các đối tượng ảnh vẫn được giữ nguyên sau khi chuyển đổi.

Dựa theo tài liệu [7], việc tính toán giá trị biểu thị độ mờ của một ảnh được thực hiện như sau: nhân chập ảnh xám với ma trận lọc Laplace 3x3:

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

sau đó tính phương sai của ảnh kết quả.

- **Nhân chập với ma trận lọc bằng phương pháp Laplace:**

Lý do phương pháp này được sử dụng chính là do định nghĩa của toán tử Laplace. Toán tử Laplace được sử dụng để đo lường đạo hàm bậc hai của một ảnh. Nó làm nổi bật các khu vực của một hình ảnh chứa các thay đổi cường độ nhanh, giống như các toán tử Sobel và Scharr và thường được sử dụng trong việc xác định biên.

- **Tìm phương sai:**

Phương sai là giá trị kỳ vọng của bình phương của độ lệch của X so với giá trị trung bình của nó.

Ảnh thu được nếu có phương sai cao chứng tỏ điểm là biên và điểm không biên có sự phản hồi mạnh, đây là một ảnh bình thường, có nét. Ngược lại, ảnh thu được có phương sai thấp chứng tỏ chỉ có một chút phản ứng nhỏ của các điểm ảnh, cho thấy có rất ít biên trong trong ảnh, đây là ảnh bị mờ.

Công thức:

$\mu = E(X)$: Giá trị trung bình (mean) của biến ngẫu nhiên X.

$var(X) = E((X - \mu)^2)$: Phương sai của biến ngẫu nhiên X.

Khi tính phương sai, để thuận tiện ta dùng công thức:

$$var(X) = E(X^2 - 2XE(X) + \big(E(X)\big)^2) = E(X^2) - 2\big(E(X)\big)^2 + \big(E(X)\big)^2$$
$$= E(X^2) - \big(E(X)\big)^2$$

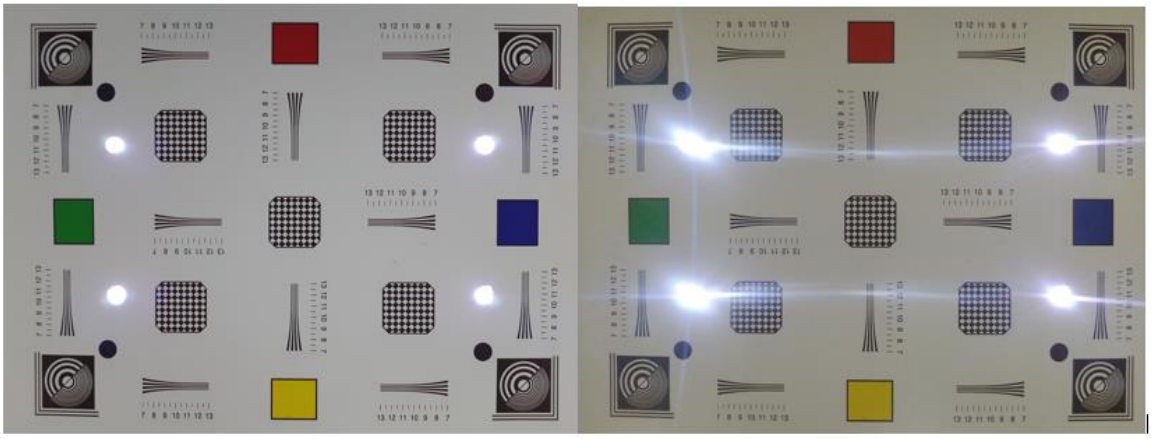
Áp dụng với X là giá trị mức xám của điểm ảnh.

Case Blur được trích rút với 3 đặc trưng:

- Phương sai sau khi nhân chập với mặt nạ Laplace.
- Giá trị mức xám trung bình của bức ảnh.
- Giá trị mức xám có tần số cao nhất (giá trị mức xám tại nền).

3.2.2 Case Flare.

Tập ảnh là ảnh chụp nền đồ hình với bốn đèn led được bật. Ảnh thỏa mãn có các vùng lân cận đèn led sáng không xuất hiện không xuất hiện các tia sáng flare ra thành các vệt. Các vệt sáng lỗi được quy định là có độ dài lớn hơn hai lần bán kính khu vực đèn led sáng. Bất kỳ góc nào trong bốn góc đèn led sáng có tia sáng flare thì ảnh đó không thỏa mãn.



Hình 3.11 *Ví dụ về case flare.*

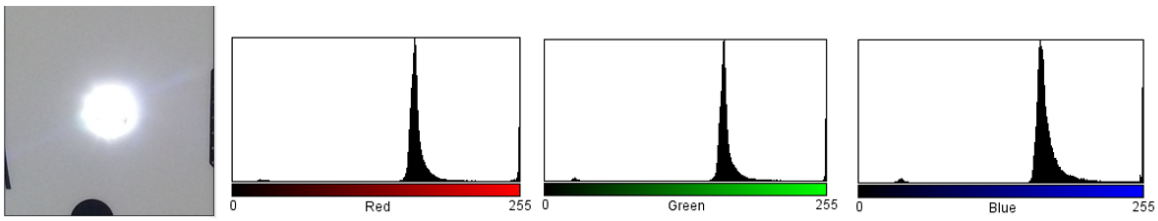
3.2.2.1 Tiền xử lý – Cắt ra các ảnh nhỏ.

Ảnh nhỏ được cắt thông qua việc xác định bốn tâm sáng đèn led dựa trên tách ngưỡng với ngưỡng > 250, cắt hình vuông ảnh bao quanh tâm led.

Hình 3.12 *Khu vực tìm đặc trưng case flare.*

3.2.2.2 Trích xuất đặc trưng.

Case flare được xác định dựa trên sự phân tích ánh sáng quanh đèn led, nên trước hết ta cần xác định khu vực sáng so với nền xung quanh và những chi tiết màu đen không liên quan có thể lẫn vào.

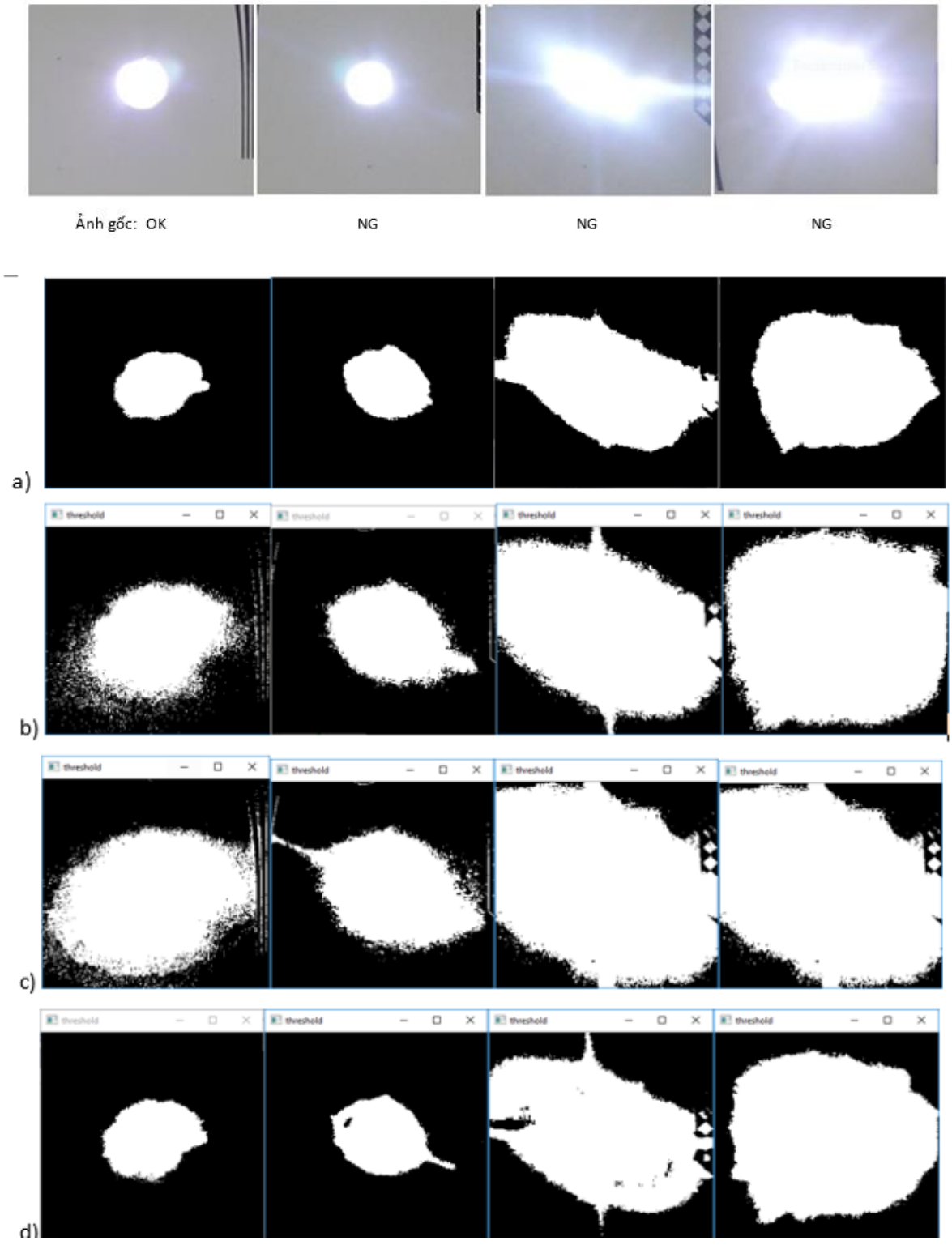


Hình 3.13 *Lược đồ histogram cho 3 kênh màu.*

Xét lược đồ histogram cho một ảnh đèn led, ta thấy ánh sáng flash có giá trị ở kênh màu blue nhạy hơn các kênh khác. Đồ thị chia thành hai đỉnh rõ rệt nên thích hợp cho việc tách ngưỡng.

Có hai phương pháp tách ngưỡng phổ biến là tách ngưỡng cố định và tách ngưỡng thích ứng.

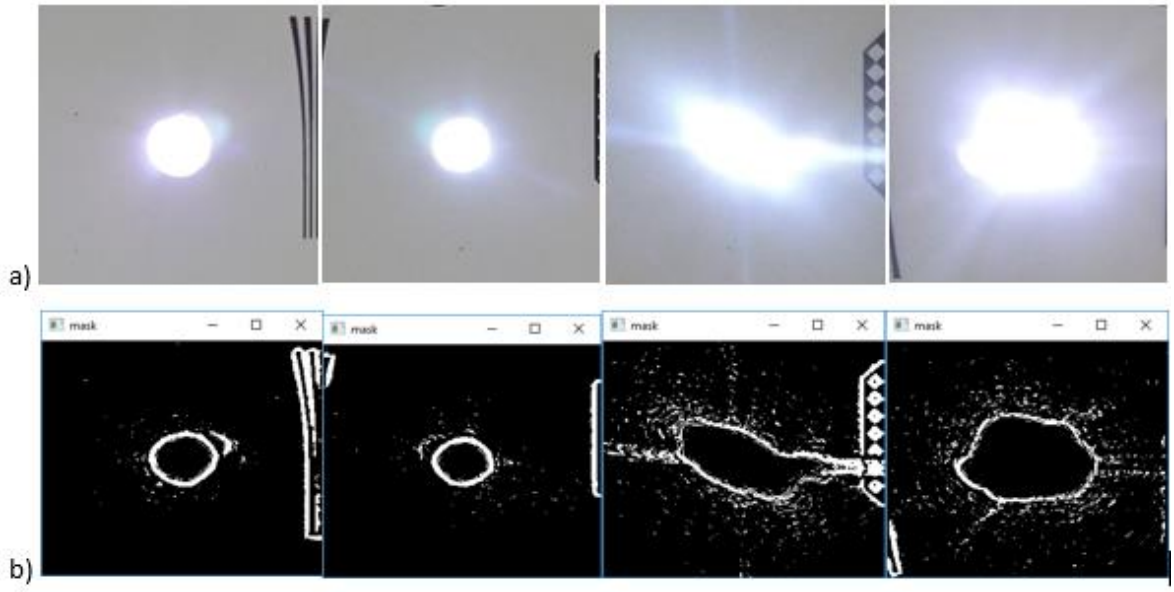
- **Tách ngưỡng cố định:**
  - Tách ngưỡng cố định sử dụng ngưỡng từ tách ngưỡng tự động (thuật toán Otsu) (Hình 3.14a).
  - Tách ngưỡng cố định từ ảnh xám (Hình 3.14b với ngưỡng bằng 170).
  - Tách ngưỡng cố định từ kênh màu blue (Hình 3.14c với ngưỡng bằng 170).
  - Tách ngưỡng cố định từ ảnh HSV theo kênh khoảng độ sáng và độ bão hòa. (Hình 3.14d với $0 \leq S \leq 51$; $190 \leq V \leq 255$).



Hình 3.14 *Kết quả thử nghiệm tách ngưỡng cố định.*

- **Tách ngưỡng thích ứng trên từng block ảnh**:

Chọn ngưỡng T(x, y) là một tổng trọng số (tương quan chéo với một cửa sổ Gaussian) của khối blockSize × blockSize của (x, y) thì ta thu được:



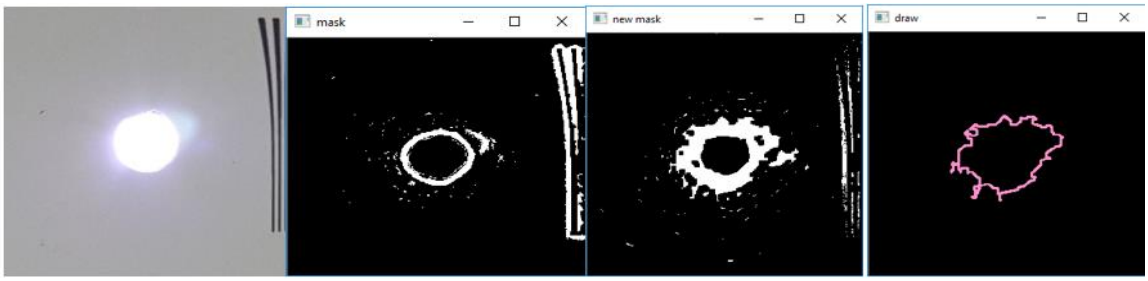
Hình 3.15 *Kết quả thử nghiệm tách ngưỡng thích ứng.*

Tách ngưỡng thích ứng đem lại hình thái chung của khu vực tia sáng tốt nhất tuy nhiên những tia sáng nhỏ và mờ không được thể hiện rõ, dễ bị mất hẳn khi lọc nhiễu, mặt khác những chi tiết thừa cũng lọt vào.

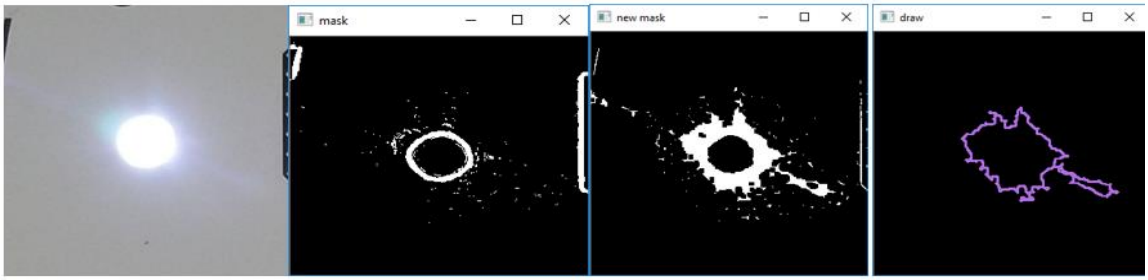
Tách ngưỡng cố định có mặt hạn chế là có những ảnh rất sáng (nền ảnh sáng do môi trường) thì một ngưỡng cố định sẽ tốt trong một số trường hợp ảnh nhất định. Xét ba trường hợp áp dụng tách ngưỡng cố định thì việc tách từ kênh màu blue có khả năng bao quát được các tia sáng mờ tốt nhất.

Do đó, đồ án đề xuất giải pháp tạo mặt nạ nhị phân là kết quả từ phép **"&"** giữa mặt nạ sau tách ngưỡng thích ứng và mặt nạ sau tách ngưỡng cố định từ kênh màu blue.

Kết quả thu được đi qua các bước giảm nhiễu và làm trơn ảnh. Lúc này hình ảnh thu được đã có sự khác biệt giữa ảnh đạt và ảnh có tia sáng flare rất mờ. Đường bao xung quanh vùng sáng được tìm bằng hàm findContour (sử dụng thuật toán từ [8]) của OpenCV.



Ảnh OK

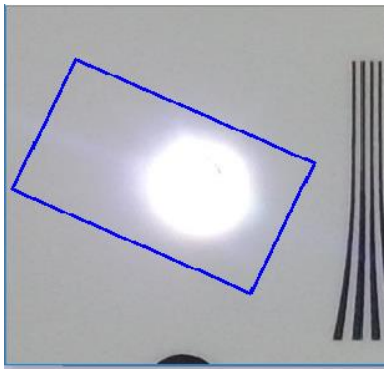


Ảnh NG

Hình 3.16 *Kết quả sau khi kết hợp hai loại tách ngưỡng.*

OpenCV cung cấp nhiều đặc trưng về contour vừa tìm được, những đặc trưng được đồ án sử dụng là:

- Diện tích: Tìm diện tích hình tạo bởi đường viền vừa tìm được.
- Chiều dài: Chiều dài của hình chữ nhật xoay quanh bao quanh đường viền.
- Chiều rộng: Chiều rộng của hình chữ nhật xoay quanh bao quanh đường viền.

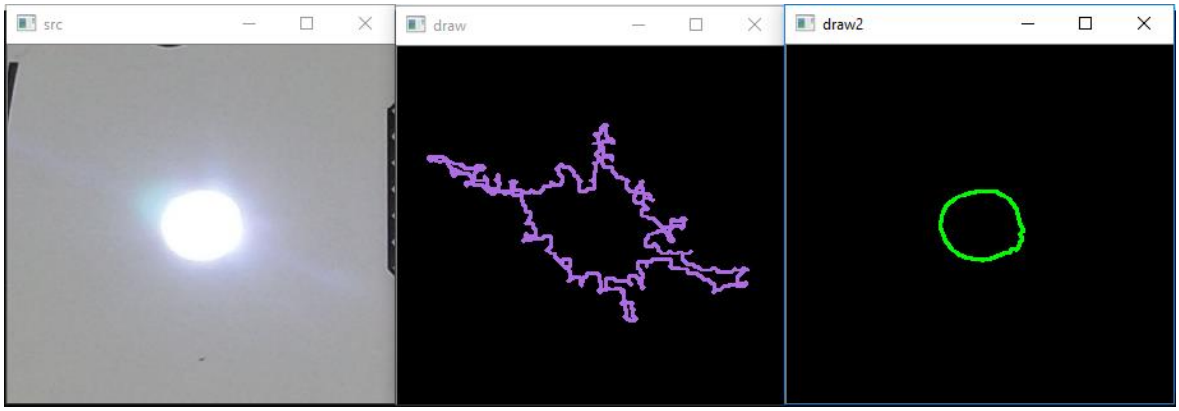


Hình 3.17 *Hình chữ nhật xoay bao quanh khu vực sáng.*

- 7 giá trị bất biến Humoment trình bày ở phần trước.

Đặc trưng SIFT được đồ án xét đến nhưng do các điểm hấp dẫn (keypoint) trên mỗi ảnh rất ít, không phải là một sự lựa chọn tốt cho đặc trưng.

Bên cạnh đó, đèn và kích thước đèn có thể ảnh hưởng đến những đặc trưng được xác định nên đồ án trích xuất đặc trưng trên ánh sáng đèn led với ngưỡng 250 để xác định những vùng sáng thật sự phục vụ cho việc huấn luyện tốt hơn. Những đặc trưng trích xuất là tương tự.



Hình 3.18 *Hai đường viền từ hai bộ lọc ngưỡng.*

Từ đó ta có, case Flare đã được trích xuất 20 đặc trưng:

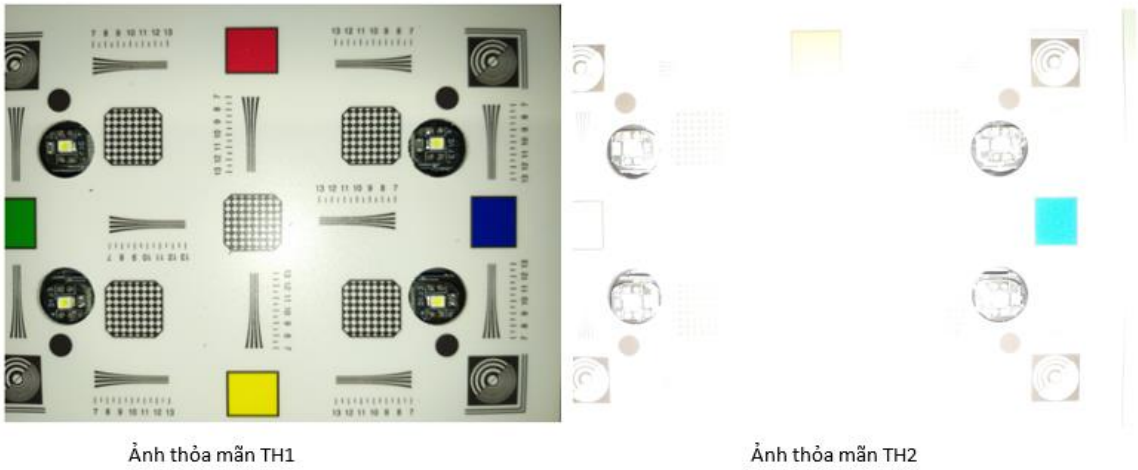
- Diện tích đường bao, chiều dài, chiều rộng, 7 bất biến Humoment từ ánh sáng led và tia sáng flare phát hiện được.
- Diện tích đường bao, chiều dài, chiều rộng, 7 bất biến Humoment từ ánh sáng led dễ thấy, có cường độ cao.

3.2.3 Case Flash.

Với các ảnh đèn flash được bật, điều kiện thỏa mãn là:

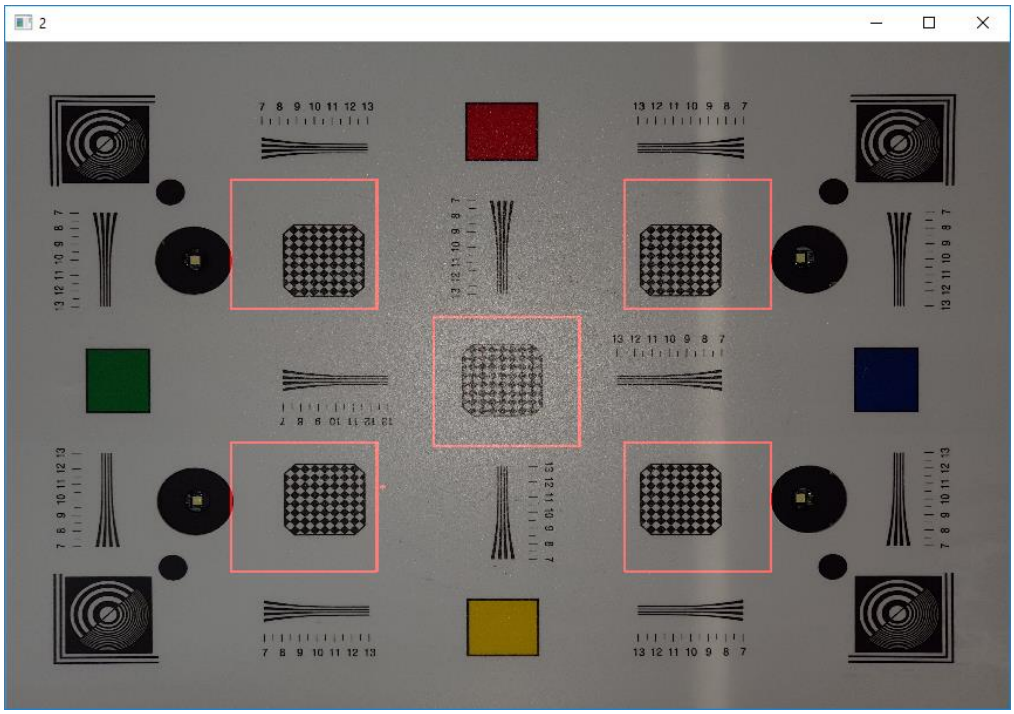
Khu vực trung tâm của ảnh có vùng sáng lóa (do bật flash) có độ sáng của khu vực trung tâm sáng hơn trên 20 đơn vị so với các khu vực biên ảnh. Đối với mẫu điện thoại thấp, tốc độ cửa chập chậm dẫn tới cháy sáng ảnh thì khu vực trung tâm ảnh và lân cận có độ sáng lớn hơn trên 220 đơn vị.

Ảnh flash không bật thì các vùng không có sự chênh lệch nhiều về độ sáng.



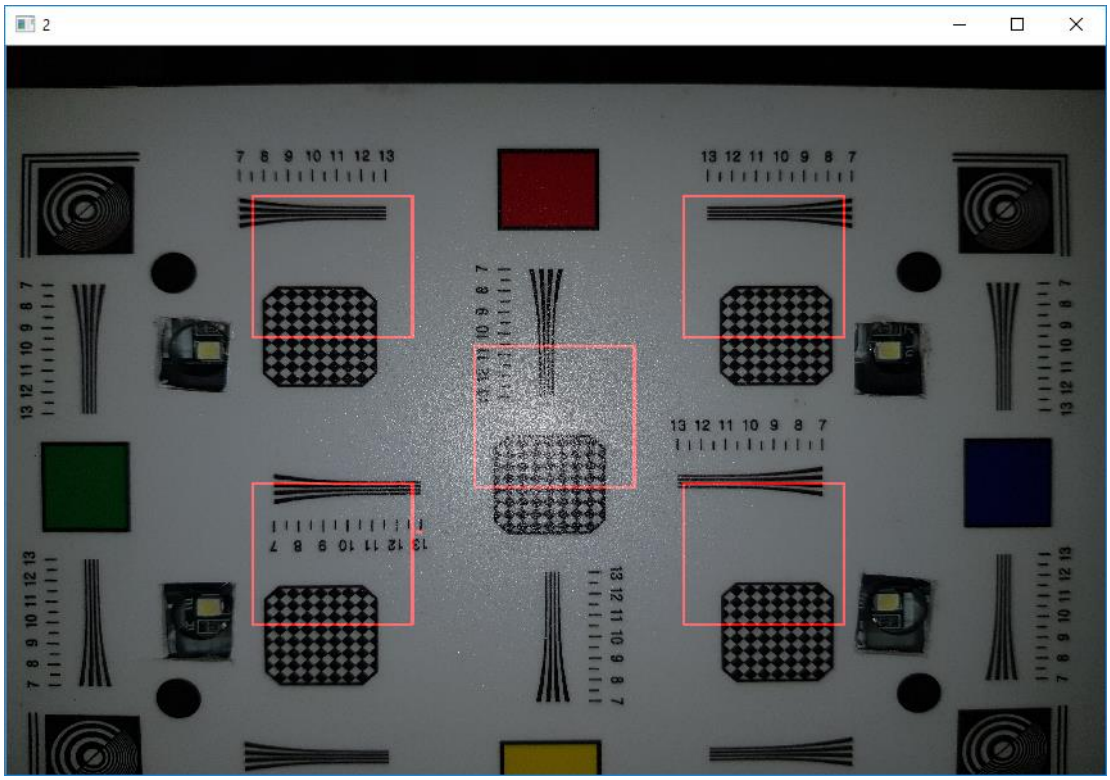
Hình 3.19 *Hai trường hợp thỏa mãn của case flash.*

Việc xác định đèn flash có được bật hay không được xác định bằng cách so sánh cường độ sáng giữa năm vùng: vùng trung tâm bức ảnh nơi thu nhận ánh đèn hắt vào và 4 vùng chéo so với vùng trung tâm. Năm vùng được xét được thể hiện qua Hình 3.20.



Hình 3.20 *Ảnh các vùng được xét trong case flash.*

Việc so sánh độ sáng giữa năm vùng rất dễ, chỉ cần xem xét giá trị trung bình của điểm ảnh trong từng ô. Tuy nhiên phần hình màu đen có thể làm ảnh hưởng đến giá trị trung bình do một số ảnh chụp lệch như Hình 3.21.



Hình 3.21 *Trường hợp ảnh đặt lệch trong case flash.*

Đồ án đưa ra hướng giải quyết là phân biệt phần hình màu đen ra so với nền bằng tách ngưỡng tự động (thuật toán Otsu) do trong hình nhỏ có sự khác biệt rõ giữa đồ hình đen và nền. Nếu pixel nhỏ hơn ngưỡng (phần hình màu đen) sẽ được gán bằng 0, lớn hơn ngưỡng (nền) giữ nguyên giá trị. Giá trị trung bình của ảnh sẽ chỉ tính bởi những pixel có giá trị khác 0.

Như vậy, với mỗi vùng, đồ án chọn đặc trưng là giá trị ngưỡng tự động tại vùng đó và giá trị trung bình mức xám của những pixel nền tại kênh blue của ảnh (ánh sáng flash có kênh màu blue cao hơn các kênh còn lại).

Tổng số đặc trưng cho case Flash là 10 đặc trưng:

- Giá trị ngưỡng tự động tại vùng trung tâm và 4 vùng chéo xung quanh.

- Giá trị trung bình mức xám của những pixel nền tại vùng trung tâm và 4 vùng chéo xung quanh.

3.2.4 Case White.

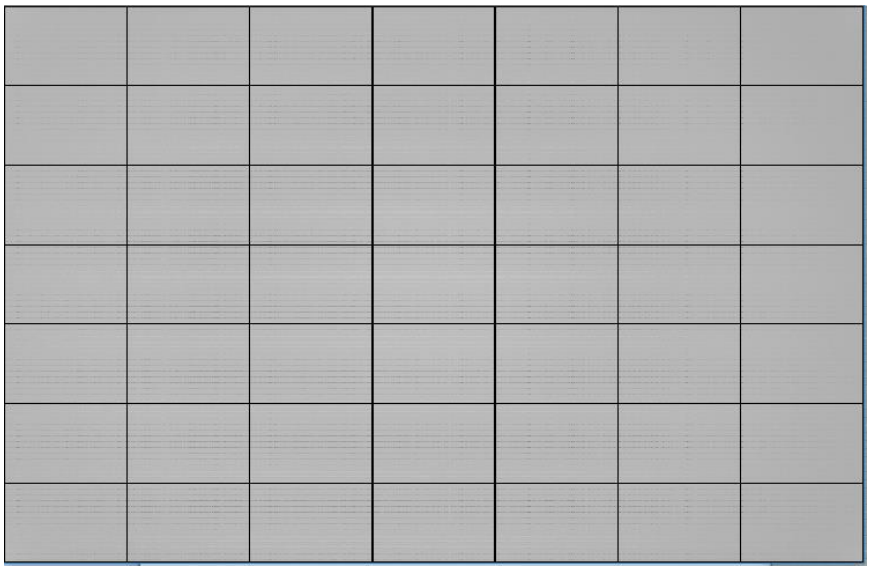
Đối với ảnh nền trắng, hình ảnh đạt phải thỏa mãn:

- Độ sáng trung bình của ảnh phải > 120 nhận biết chụp trên nền trắng.
- Vì trên nền trắng nên giá trị của các kênh màu xấp xỉ nhau nếu có kênh màu nào lớn hơn các kênh khác 20 thì ảnh coi như bị ám màu.
- Nền trắng trong suốt không có các vết bẩn, các vệt sáng.

Việc sử dụng học máy sẽ không xét đến những ngưỡng cụ thể mà trích xuất các giá trị nguyên thủy cho máy tự huấn luyện.

Từ những tiêu chí cần đạt, việc trích đặc trưng cần qua các bước (đã chuyển ảnh thành ảnh xám):

- **Kiểm tra ám màu:** Chia hình ảnh thành 9 khối, trích ra giá trị trung bình của mỗi kênh màu trên mỗi khối.
- **Xác định ảnh có đều màu hay không**: chia nhỏ hình ảnh thành 49 khối bằng nhau, ta lấy ra giá trị trung bình mức xám của khối, giá trị 4 mức xám cao nhất, giá trị 4 mức xám thấp nhất.



Hình 3.22 *Ảnh được chia thành 49 khối.*

- **Xác định có đốm hoặc vết:** dùng tách ngưỡng thích ứng để làm nổi đốm hoặc vết và tìm biên của các đốm, vết nếu có. Diện tích của đốm sẽ được trích xuất.

Hình 3.23 *Ảnh sau tách ngưỡng của case white.*

Tổng có 469 đặc trưng được trích xuất ở case White:

- Giá trị trung bình của mỗi kênh màu trên mỗi khối (ảnh chia 9 khối).
- Giá trị trung bình mức xám của khối, giá trị 4 mức xám cao nhất, giá trị 4 mức xám thấp nhất (ảnh chia 49 khối).
- Diện tích đốm sau tách ngưỡng.

3.3 Thực hiện phân loại ảnh.

3.3.1 Chuẩn hóa dữ liệu.

Đồ án sử dụng phương pháp Rescaling, đưa tất cả các giá trị đặc trưng về [0,1] theo công thức:

$$x' = \frac{x - \min(x)}{max(x) - min(x)}$$

Trong đó *x* là giá trị ban đầu, *x'* là giá trị sau khi chuẩn hóa, *min(x)* và *max(x)* được tính trên toàn bộ dữ liệu huấn luyện với cùng một thành phần.

Các đặc trưng trên một ảnh được gán thêm nhãn 1 (ảnh OK) và 0 (ảnh NG) làm đầu vào cho nhãn các ảnh tập huấn luyện và nhãn thực của tập test.

3.3.2 Sử dụng mô hình SVM.

Phương pháp SVM yêu cầu dữ liệu được diễn tả như các vectơ của các số thực. Như vậy nếu đầu vào chưa phải là số thì ta cần phải tìm cách chuyển chúng về dạng số của SVM.

- **Tiền xử lý dữ liệu:**

  Thực hiện biến đổi dữ liệu phù hợp cho quá trình tính toán, tránh các số quá lớn mô tả các thuộc tính. Thường nên co giãn *(scaling)* dữ liệu để chuyển về đoạn [-1, 1] hoặc [0, 1].

- **Chọn hàm hạt nhân:**

  Lựa chọn hàm hạt nhân phù hợp tương ứng cho từng bài toán cụ thể để đạt được độ chính xác cao trong quá trình phân lớp.

- **Lựa chọn tham số:**

  Các tham số được lựa chọn cho việc huấn luyện với tập mẫu. Trong quá trình huấn luyện sẽ sử dụng thuật toán tối ưu hóa khoảng cách giữa các siêu phẳng trong quá trình phân lớp, xác định hàm phân lớp trong không gian đặc trưng nhờ việc ánh xạ dữ liệu vào không gian đặc trưng bằng cách mô tả hạt nhân, giải quyết cho cả hai trường hợp dữ liệu là phân tách và không phân tách tuyến tính trong không gian đặc trưng.

- **Kiểm thử tập test:**

  Sử dụng mô hình và các tham số vừa nhận được để đưa vào tập test để kiểm tra độ chính xác của mô hình đó.

3.3.3 Sử dụng mô hình mạng CNN.

Khởi đầu chương trình chúng ta sẽ gọi các thư viện cần thiết. Trong bài toán này, chúng tôi sử dụng mô hình CNN xây dựng trên nền tảng thư viện Tensorflow, OpenCV và Numpy chạy trên máy tính có GPU. Thông số cấu hình của máy tính sử dụng cho bài toán phân loại này được nêu trong phần *Phụ lục*. Tiếp theo, mô hình sẽ đưa dữ liệu vào chương trình để huấn luyện, đối với bài toán trên là ta có một tập các ảnh đã được gán nhãn và phân loại vào các thư mục *"ng", "ok"* với mỗi trường hợp bộ ảnh Blur, Flare, Flash, White. Mục tiêu của chúng ta là xây dựng một mô hình CNN đủ sâu để có thể giải quyết tốt được bài toán. Đầu tiên, vì điều kiện máy tính và nhận thấy ảnh đầu vào có kích thước 3 x 4 nên chúng tôi đã thực hiện co giãn ảnh về kích thước 180 x 240, và chúng ta chuyển về thành một vector thay vì đặt là một ma trận điểm ảnh.

Trong bài toán này, đồ án sử dụng mô hình CNN với hai lớp ẩn, một lớp *fully-connected* và một lớp đầu ra cho bài toán. Kích thước batch_size được sử dụng là 32. Cách hoạt động của mô hình đã được nêu trong phần giới thiệu về mô hình CNN. Ở lớp Convolutional, mạng được xây dựng bởi conv2D, tức là đầu vào dữ liệu là bốn chiều. Ở từng lớp convolutional, hàm kích hoạt ReLU sẽ được sử dụng. Sau đó lớp đầu ra sẽ được đưa vào chạy bằng maxpool2D. Kết quả đầu ra sau một lần conv + pool được đưa vào hàm Softmax. Công thức hàm ReLU và hàm Softmax:

Hàm ReLU:

$$f(x) = \max(0, x)$$

Tác dụng của hàm ReLU là giảm thiểu sự tính toán phức tạp, vì đạo hàm của hàm ReLU là một số thực và việc đưa tất cả điểm dữ liệu nhỏ hơn 0 về 0 để tránh sai số gây ảnh hưởng trong việc tính toán ở phần maxpooling.

Hàm Softmax:

$$a_i = \frac{exp(z_i)}{\sum_i^C exp(z_i)}$$

Trong đó C là số classes đầu ra.

Hàm softmax được dùng vì đầu ra của C classes ứng với mỗi class được phân loại sẽ có tổng là 1. Mặt khắc, ta thấy với công thức như trên, a_i luôn luôn khác không. Từ việc này chúng ta có thể sử dụng tính toán hàm mất mát bằng cross-entropy qua việc đầu ra được lưu dưới dạng one-hot coding. Hàm cross-entropy nhận giá trị rất cao nếu như kết quả đầu ra khác xa so với kết quả thực tế. Trong khi đó thì sự chênh lệch giữa các loss (mất mát) ở gần hay xa trong hàm bình phương khoảng cách là không đáng kể. Về mặt tối ưu, hàm cross-entropy sẽ cho nghiệm gần với kết quả thực hơn vì những nghiệm ở xa sẽ bị trừng phạt giá trị loss rất nặng.

Khi chúng ta thu được *hàm mất mát (loss function)*, việc tối ưu hàm mất mát trong bài toán này được thực hiện bởi AdamOptimizer với tốc độ học tự cập nhật và lựa chọn ra tốc độ học phù hợp nhất đối với bài toán. Việc này cũng tương tự như Gradient Descent, chúng ta tối ưu các trọng số của mô hình mạng và sẽ được cập nhật bằng phương pháp *backpropagation* như đã đề cập phía trên. Mô hình sẽ tối ưu để cho cả dữ liệu training và dữ liệu validation (thu được từ việc lấy ngẫu nhiên 20% dữ liệu trong bộ huấn luyện ban đầu) đều có hàm mất mất là nhỏ nhất có thể. Sau số lượng vòng lặp thì mô hình sẽ dừng lại. Ở đây chúng tôi chưa sử dụng *early stopping* cho bài toán, tức là khi có dấu hiệu hàm loss của tập validation có xu hướng tăng lên thì chúng ta sẽ dừng lại việc huấn luyện cho dù chưa huấn luyện đủ số vòng lặp quy định. Mô hình thu được sau quá trình train sẽ được lưu vào trong chương trình để lúc kiểm thử thì chúng ta chỉ cần gọi mô hình tránh việc huấn luyện lại dữ liệu từ đầu.

Dữ liệu sau khi được huấn luyện sẽ được lưu vào một model để khi cần kiểm thử bộ ảnh nào đó thì chỉ cần gọi ra mà không cần phải thực hiện lại việc huấn luyện. Với kết quả nhận được, mỗi model thu được của từng trường hợp đều có kích thước lớn hơn 1.5GB. Chúng ta có thể thấy rằng, mô hình CNN với một vài lớp đơn giản nhưng phải thực hiện khối lượng công việc tính toán rất lớn.

3.4 Đánh giá chất lượng camera.

3.4.1 Kết quả của mô hình SVM.

- **Lựa chọn tham số mô hình SVM:**

Mô hình SVM được sử dụng với loại có tham số C, sử dụng nhân RBF vì những ưu điểm đã được nêu trong chương 2.

Kết quả tối ưu tham số thu được cho từng trường hợp là:

Blur : gamma = 0.50625 ; C = 62.5

Flare: gamma = 0. 50625 ; C = 12.5

Flash : gamma = 0. 50625 ; C = 0.5

White : gamma = 0. 50625 ; C = 2.1

- **Kết quả kiểm nghiệm của mô hình SVM:**

Trường hợp	Độ chính xác	Tỉ lệ đúng: Ảnh NG	Tỉ lệ đúng: Ảnh OK	Thời gian chạy
Blur	97.00%	48/50 (96.00%)	49/50 (98.00%)	40 ms
Flare	95.00%	45/50 (90.00%)	50/50 (100.00%)	116 ms
Flash	97.12%	50/50 (100.00%)	51/54 (94.44%)	34 ms
White	97.30%	58/61(95.30%)	50/50 (100.00%)	803 ms

Bảng 3.2 *Kết quả thuật toán SVM.*

3.4.2 Kết quả của mô hình CNN.

- **Dữ liệu ảnh trong mô hình CNN:**

Mô hình CNN được xây dựng với bộ dữ liệu là các ảnh gốc được điều chỉnh kích thước để phù hợp với cấu hình của máy tính dùng cho việc huấn luyện. Ở hai trường hợp Blur và Flare, chúng ta sẽ xét hai bộ dữ liệu ảnh, bộ thứ nhất chính là ảnh được cắt các khu vực đặc trưng như ở phần SVM đã sử dụng. Bộ ảnh thứ hai

chính là bộ ảnh căn chỉnh kích thước về 180 x 240 của cả bốn trường hợp Blur, Flare, Flash, White:

Trường hợp	Tập huấn luyện		Tập kiểm thử	
	Ảnh NG	Ảnh OK	Ảnh NG	Ảnh OK
Blur	134	151	50	50
Flare	333	554	50	50
Flash	78	225	50	54
White	117	101	61	50

Bảng 3.3 *Kích thước dữ liệu ảnh huấn luyện CNN được co giãn.*

- **Kết quả kiểm nghiệm mô hình CNN:**

Kết quả của bộ ảnh 4 trường hợp đều có kích thước là 180 x 240:

Trường hợp	Độ chính xác	Tỉ lệ đúng: Ảnh NG	Tỉ lệ đúng: Ảnh OK
Blur	99.00%	49/50 (98.00%)	50/50 (100.00%)
Flare	95.00%	48/50 (96.00%)	47/50 (94.00%)
Flash	71.15%	41/50 (82.00%)	33/54 (61.11%)
White	90.99%	51/61(83.61%)	50/50 (100.00%)

Bảng 3.4 *Kết quả thuật toán CNN.*

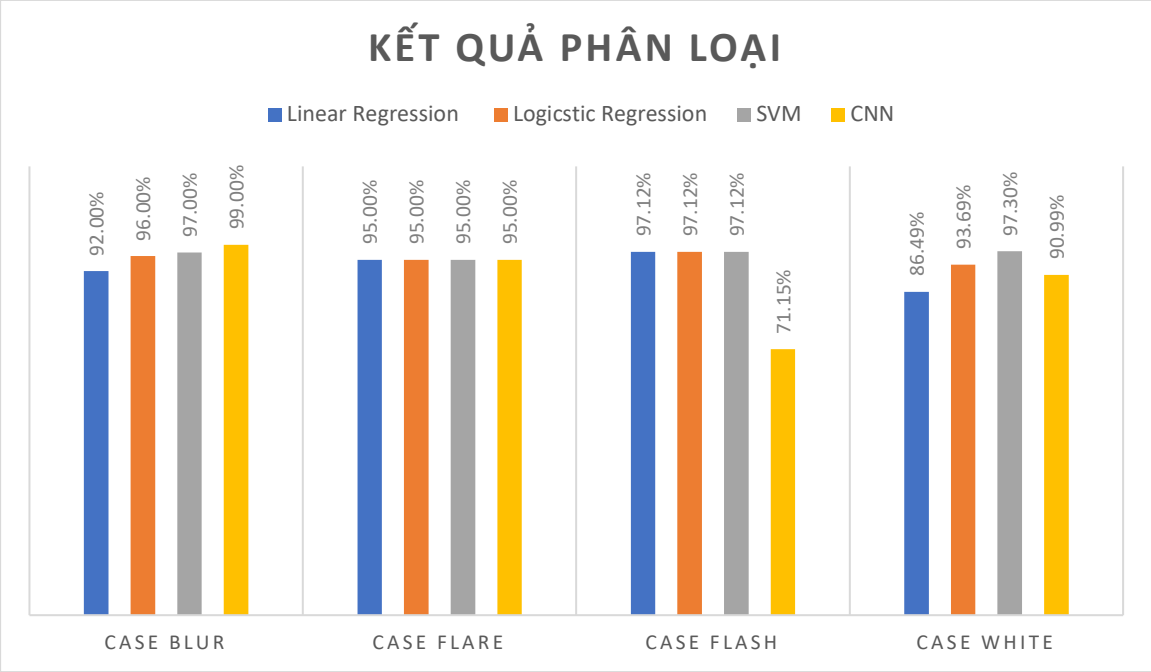
Ở trường hợp của Blur và Flare, chúng ta xét thêm việc sử dụng bộ dữ liệu đầu vào là ảnh cắt từ những khu vực đặc trưng của ảnh gốc. Kết quả thu được của hai trường hợp này có kết quả bằng với kết quả của 2 trường hợp co giãn ảnh.

3.4.3 So sánh với thuật toán Linear Regression và Logistic Regression

	Linear Regression	Logistic Regression	SVM	CNN
Case Blur	92.00 %	96.00 %	97.00 %	99.00%
Case Flare	95.00 %	95.00 %	95.00 %	95.00%
Case Flash	97.12 %	97.12 %	97.12 %	71.15%

Case White	86.49 %	93.69 %	97.30 %	90.99%

Bảng 3.5 *So sánh độ chính xác giữa các thuật toán.*



Hình 3.24 *Kết quả phân loại chung của 3 thuật toán.*

Linear Regression và Logistic Regression là hai thuật toán hoạt động tốt với dữ liệu có sự phân biệt tuyến tính và xây dựng những mô hình đơn giản.

Case Flash và Case Flare có kết quả đồng đều giữa 3 loại học máy chứng tỏ đặc trưng trích xuất là gần phân biệt tuyến tính.

Các case còn lại phương pháp SVM đã chứng minh việc tạo mô hình của mình tốt hơn đặc biệt với case white số lượng đặc trưng lớn.

Mô hình CNN với việc co giãn kích thước ảnh đã một phần làm mất thông tin của ảnh. Đặc biệt trong trường hợp Flash, mô hình đã khó phân biệt được khu vực ở giữa có sáng hơn các khu vực xung quanh hay không. Vì vậy kết quả trong trường hợp này đạt khá thấp. Tuy nhiên chúng tôi vẫn muốn sử dụng mô hình này để thử nghiệm xem kết quả và ứng dụng thực tế của CNN. Qua kết quả trên chúng ta thấy rằng mô hình CNN cần một cấu hình máy huấn luyện có cấu trúc lớn đủ để thực hiện khối lượng tính toán phức tạp.

3.4.4 Đánh giá kết quả phân loại.

Nhìn chung kết quả phân loại là khả quan, đặc trưng trích xuất khá tốt để chạy học máy. Những trường hợp còn phân loại sai:

- Case Blur: Ảnh có nền rất tối.
- Case Flare: Những ảnh có tia flare nhẹ vẫn chưa được nhận dạng tốt.
- Case Flash: Bộ test ảnh NG có các ảnh chụp của case flare không nhận dạng đúng.
- Case White: Phân loại sai các ảnh bị ám màu rất nhẹ.

Chương 4 Kết luận và hướng phát triển

4.1 Kết luận.

Đồ án trình bày phương pháp áp dụng mô hình học máy SVM vào bài toán phân loại ảnh trong quá trình kiểm tra chất lượng camera, cụ thể nội dung công việc đạt được bao gồm:

- Nêu một số đặc trưng của ảnh và thực hiện cắt các khu vực ảnh chứa đặc trưng được đề cập.
- Tiền xử lý dữ liệu ảnh đầu vào và sự áp dụng của xử lý ảnh vào việc tách đặc trưng.
- Thực hiện giải quyết bài toán bằng phương pháp Support Vector Machine.
- Thực hiện co giãn ảnh đầu vào, huấn luyện trên hệ thống server, lưu thành model và áp dụng vào việc giải quyết bài toán.
- Thực hiện giải quyết bài toán bằng một số thuật toán khác như Logistic Regression, Linear Regression. So sánh đánh giá cùng với một số phương pháp phân loại khác phía trên.

SVM là phương pháp phân loại mạnh, là thuật toán xử lý đầy đủ các vấn đề của quá trình phân loại: giải quyết vấn đề overfitting rất tốt, giải quyết các bài toán dữ liệu có số chiều lớn,...

Ứng dụng phương pháp máy véc-tơ hỗ trợ để phân loại ảnh trong quá trình kiểm tra chất lượng camera của smartphone có kết quả phân loại tốt, các case đều lớn hơn 95%.

Với sự hạn chế về thời gian cùng với kiến thức và khả năng lập trình của chúng tôi thì chương trình không tránh khỏi những thiếu sót. Cụ thể hạn chế trong đồ án mà chúng tôi nhận thấy của chương trình là:

- Đặc trưng trích chọn chưa thực sự tối ưu.
- Bộ huấn luyện và kiểm thử còn khá nhỏ.
- Việc phân loại hai nhãn còn đơn giản.
- Máy dùng để huấn luyện chưa đủ mạnh để có thể dùng bộ ảnh gốc trong việc huấn luyện.

4.2 Hướng phát triển.

Với kết quả trên, hướng nghiên cứu phát triển tiếp theo cho đồ án là:

- Thêm các đặc trưng tốt hơn, tìm hiểu các đặc trưng cấp cao để tối ưu giản lược các đặc trưng cần trích xuất.
- Mở rộng tập huấn luyện và tập test. Thay vì chỉ có hai nhãn, tập ảnh mới phân loại tăng số lượng nhãn cho các trường hợp chi tiết hơn.
- Tối ưu thuật toán hiện tại và áp dụng thêm một số mô hình học máy tiên tiến khác.
- Mở rộng điều kiện nhận dạng ra các ảnh có bị chụp nghiêng.
- Tiến tới thực hiện các dự án học máy khác áp dụng cho nhà máy như: kiểm tra điện thoại đã được dán tem khi xuất xưởng, kiểm tra công nhân làm việc đúng quy trình qua video,…

Tài liệu tham khảo

Tài liệu tiếng Việt:

[1] Lương Mạnh Bá, Nguyễn Thanh Thủy, Nhập môn xử lý ảnh số, Đại học Bách Khoa Hà Nội.

[2] TS. Đỗ Năng Toàn, TS. Phạm Việt Bình, Giáo trình xử lý ảnh, Đại học Thái Nguyên, Viện Công nghệ thông tin, 11/2007.

[3] Slide xử lý ảnh số, Học viện Công nghệ bưu chính viễn thông.

[4] Trần Minh Văn, Luận văn thạc sĩ: Tìm hiểu phương pháp SVM và ứng dụng trong phương pháp nhận dạng chữ viết tay trực tuyến, Đại học quốc gia Thành phố Hồ Chí Minh, trường Đại học Khoa học tự nhiên, 2004.

Tài liệu Tiếng Anh:

[5] C.-W. Hsu and C.J.Lin, A comparison of methods for multi-class support vector machines. IEEE Transactions on Neural Networks, 13(2):415-425, 2002.

[6] M. K. Hu, "Visual Pattern Recognition by Moment Invariants," IRE Trans. Info. Theory, vol. IT-8, 1962, p. pp.179–187.

[7] J.L Pech-Pacheco & G. Cristobal, Diatom autofocusing in brightfield microscopy: a comparative study.

[8] Suzuki, S. and Abe, K., "Topological Structural Analysis of Digitized Binary Images by Border Following," pp. CVGIP 30 1, pp 32-46 (1985).

[9] T.Joachims (1999), Transductive Inference for Text Classification using Support Vector Machines. International Conference on Machine Learning (ICML), 1999.

[10] T.Joachims (2003), Transductive learning via spectral graph partitioning. Proceeding of The Twentieth International Conference on Machine Learning (ICML2003): 290-297.

[11] V. Vapnik, The Nature of Statistical Learning Theory, NY: Springer-Verlag, 1995.

[12] V.Sindhwani, S.S.Keerthi (2007), Newton Methods for Fast Solution of Ser-supervised Linear SVMs. Large Scale Kernel Machines, MIT Press, 2005.

[13] Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin, "A Practical Guide to Support Vector Classification". Deptt of Computer Sci. National Taiwan Uni, Taipei, 106, Taiwan http://www.csie.ntu.edu.tw/~cjlin, 2007.

[14] Lowe David, Distinctive image features from scale-invariant keypoints., International Journal of Computer Vision 2004.

[15] Yushi Jing, PageRank for images products search.

[16] Kamarul Hawari Ghazali, Feature Extraction technique using SIFT, The International Conference on Electrical and Engineering.

[17] Boser, B, E, I. Guyon, and V. Vapnik (1992), A training algorithm for optimal margin classifiers. In Proceedings of the Fifth Annual Workshop on Computational Learning Theory, pages. 144 – 152. ACM Press, 1992.

[18] Chang, C.-C. and C.J.Lin (2001), LIBSVM: a library for support vector machines. http://www.csie.ntu.edu.tw/~cjlin/libsvm

Website:

[19] https://sites.google.com/a/wru.vn/cse445spring2016/

[20] http://driving.stanford.edu/

[21] "Machine Leaning cơ bản," [Online]. Available: machinelearningcoban.com

[22] Arden Dertat, https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2 , 11/2017.

Phụ lục

Thông tin cấu hình máy tính sử dụng trong quá trình thực hiện đồ án:

- **Máy tính cá nhân:**

Operating System: Windows 7 Professional 64-bit (6.1, Build 7601).

System Manufacturer: SAMSUNG ELECTRONICS CO.

System Model: 370E5L/371B5L.

Processor: Intel(R) Core(TM) i7-6700HQ CPU @ 2.60GHz (8 CPUs), ~2.6GHz

Memory: 8192MB RAM.

GPU: GeForce 920MX.

- **Máy tính Server:**

Operating System: Windows 10 Pro 64-bit (10.0, Build 17134).

System Manufacturer: ASUSTeK COMPUTER INC.

System Model: Z10PA-D8 Series.

Processor: Intel(R) Xeon(R) CPU E5-2683 v3 @ 2.00GHz (56 CPUs), ~2.0GHz

Memory: 65536MB RAM.

GPU: NVIDIA GeForce GTX 1080.