TRƯỜNG ĐẠI HỌC XÂY DỰNG HÀ NỘI

KHOA CÔNG NGHỆ THÔNG TIN

BÀI TẬP LỚN

HỌC MÁY

ĐỀ TÀI:

XÂY DỰNG ỨNG DỤNG NHẬN DIỆN BIỂN SỐ XE SỬ DỤNG DEEP LEARNING

Sinh viên thực hiện:

Vũ Xuân Hoàn - 85164

Trần Quốc Thái - 178264

Triệu Việt Hùng – 1523864

Nhóm: 03

Hà Nội 02-2022

TÓM TẮT BÀI TẬP LỚN

Hiện nay, số lượng xe cộ tham gia giao thông trên đường là rất lớn dẫn đến tiêu tốn rất nhiều nhân lực và vật lực cho việc quản lý phương tiện cá nhân trong bãi gửi xe. Nếu không có một công cụ thuận tiện thì việc quản lý phương tiện cá nhân rất mất thời gian, dễ gây nhầm lẫn, thiệt hay cho người sử dụng dịch vụ tại các bãi đỗ xe.

Để giảm tải cho các công việc như thu tiền, bảo hiểm xe, tìm xe cộ trong bãi đỗ xe, trên thế giới đã phát triển công nghệ giám sát tự động đối với các phương tiện giao thông, chính nhờ tính cá nhân của biển số xe mà nó đã trở thành đối tượng chính được sử dụng để nghiên cứu, phát triển trong công nghệ này.

Do đó em muốn chọn đề tài này như bước căn bản trong việc tìm hiểu các công cụ giám sát mạnh hơn như kiểm soát xe lưu thông trên đường hay nhận dạng khuôn mặt ... đang được thế giới rất chú trọng lúc này.

MỤC LỤC

1. GIỚI THIỆU

1.1 . Tổng quan

Nội dung :

- Tìm hiểu về biển số xe và hệ thống nhận dạng biển số xe
- Phát biểu bài toán và hướng giải quyết
- Nghiên cứu một số thuật toán xử lý ảnh và nhận dạng kí tự ứng dụng trong việc nhận dạng biển số xe

1.2 . Nhiệm vụ đề tài.

Từ nội dung nêu trên, đề tài của em sẽ bao gồm các nhiệm vụ sau:

- Tìm hiểu khái quát về xử lý ảnh và bài toán nhận dạng biển số xe.
- Tìm hiểu thông tin về biển số xe và phân loại biển số xe của Việt Nam.
- Tìm hiểu các công đoạn chính của bài toán nhận dạng biển số xe gồm 3 khâu chính:
  - Phát hiện vị trí và tách biển số xe
  - Phân đoạn kí tự trong biển số xe
  - Nhận dạng kí tự
- Cài đặt thử nghiệm.

2. TỔNG QUAN BÀI TOÁN NHẬN DIỆN BIỂN SỐ XE

2.1 . Khái niệm biển số xe

Ở Việt Nam, biển kiểm soát xe cơ giới (hay còn gọi tắt là biển kiểm soát, biển số xe) là tấm biển gắn trên mỗi xe cơ giới, được cơ quan công an cấp (đối với xe quân sự do Bộ Quốc phòng cấp) khi mua xe mới hoặc chuyển nhượng xe. Biển số xe được làm bằng hợp kim nhôm sắt, có dạng hình chữ nhật hoặc hơi vuông, trên đó có in số và chữ (biển xe dân sự không dùng các chữ cái I, J, O, Q, W. Chữ R chỉ dùng cho xe rơ-moóc, sơ-mi rơ-moóc) cho biết: Vùng và địa phương quản lý, các con số cụ thể khi tra trên máy tính còn cho biết danh tính người chủ hay đơn vị đã mua nó, thời gian mua nó phục vụ cho công tác an ninh, đặc biệt trên đó còn có hình Quốc huy Việt Nam dập nổi.

Tiêu chuẩn về kích thước: Ở mỗi nước thường có tiêu chuẩn về kích thước nhất định, còn riêng Việt Nam tỉ lệ kích thước giữa các biển số là gần như giống nhau. Biển số xe có 2 loại, kích thước như sau: Loại biển số dài có chiều cao 110 mm, chiều dài 470 mm; loại biển số ngắn có chiều cao 200 mm, chiều dài 280 mm nên ta sẽ giới hạn tỉ lệ cao/rộng là $3.5 \leq$ cao/rộng ≤ 6.5 (biển một hàng) và $0.8 \leq$ cao/rộng ≤ 1.5 (biển hai hàng).

Số lượng kí tự trong biển số xe nằm trong khoảng [7,9]. Chiều cao của chữ và số: 80mm, chiều rộng của chữ và số: 40mm.

Từ những đặc điểm trên ta có thể thiết lập nhưng thông số, điều khiển để lọc chọn những đối tương phù hợp mà ta cần.

2.2 . Xử lý ảnh và Open CV

Xử lý ảnh là một phân ngành trong xử lý số tín hiệu với tín hiệu xử lý là ảnh. Đây là một phân ngành khoa học mới rất phát triển trong những năm gần đây. Xử lý ảnh gồm 4 lĩnh vực chính: xử lý nâng cao chất lượng ảnh, nhận dạng ảnh, nén ảnh và truy vấn ảnh. Sự phát triển của xử lý ảnh đem lại rất nhiều lợi ích cho cuộc sống của con người. Ngày nay xử lý ảnh đã được áp dụng rất rộng rãi trong đời sống như: photoshop, nén ảnh, nén video, nhận dạng biển số xe, nhận dạng khuôn mặt, nhận dạng chữ viết, xử lý ảnh thiên văn, ảnh y tế,....

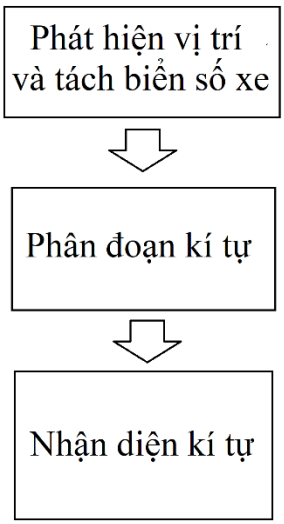
OpenCV (Open Computer Vision) là một thư viện mã nguồn mở hàng đầu cho xử lý về thị giác máy tính, machine learning, xử lý ảnh. OpenCV đươc viết bằng C/C++, vì vậy có tốc độ tính toán rất nhanh, có thể sử dụng với các ứng dụng liên quan đến thời gian thực. Opencv có các interface cho C/C++, Python Java vì vậy hỗ trợ được cho Window, Linux, MacOs lẫn Android, iOS OpenCV có cộng đồng hơn 47 nghìn người dùng và số lượng download vượt quá 6 triệu lần. Opencv có rất nhiều ứng dụng như:

- Nhận dạng ảnh
- Xử lý hình ảnh
- Phục hồi hình ảnh/video
- Thực tế ảo
- Các ứng dụng khác

2.3 . Hướng giải quyết

Hiện nay trên thế giới đã có rất nhiều cách tiếp cận khác nhau với việc nhận dạng biển số xe, tuy nhiên trong phạm vi đồ án này em sẽ giải quyết vấn đề theo 3 bước chính:

1. Phát hiện vị trí và tách biển số xe từ một hình ảnh có sẵn từ đầu vào là camera
2. Phân đoạn các kí tự có trong biển số xe
3. Nhận diện các kí tự đó từ mô hình nhận diện chữ số đã train sẵn.

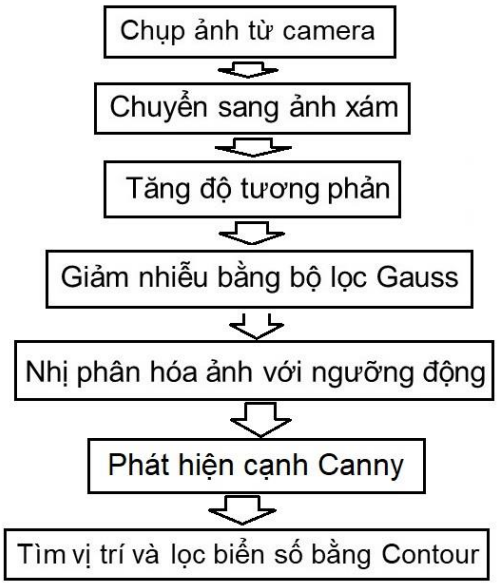


Hình 2.3 - 1 Các bước chính trong nhận dạng biển số xe

3. PHÁT HIỆN VỊ TRÍ VÀ TÁCH BIỂN SỐ XE

3.1 Hướng giải quyết

Sơ đồ dưới đây sẽ tóm gọn các bước để xác định và tách biển số xe từ ảnh:



Hình 3.1 - 1 Xác định và tách biển số xe

Đầu tiên từ ảnh đầu vào xử lý, tách biển số. Ở phạm vi đề tài này, ý tưởng chủ yếu là nhận diện được biển số từ sự thay đổi đột ngột về cường độ ánh sáng giữa biển số và môi trường xung quanh nên ta sẽ loại bỏ các dữ liệu màu sắc RGB bằng cách chuyển sang ảnh xám. Tiếp theo ta tăng độ tương phản với hai phép toán hình thái học Top Hat và Black Hat

để làm nổi bật thêm biển số giữa phông nền, hỗ trợ cho việc xử lý nhị phân sau này. Sau đó, ta giảm nhiễu bằng bộ lọc Gauss để loại bỏ những chi tiết nhiễu có thể gây ảnh hưởng đến quá trình nhận diện, đồng thời làm tăng tốc độ xử lý.

Việc lấy ngưỡng sẽ giúp ta tách được thông tin biển số và thông tin nền, ở đây em chọn lấy ngưỡng động (Adaptive Threshold). Tiếp đó ta sử dụng thuật toán phát hiện cạnh Canny để trích xuất những chi tiết cạnh của biển số. Trong quá trình xử lý máy tính có thể nhầm lẫn biển số với những chi tiết nhiễu, việc lọc lần cuối bằng các tỉ lệ cao/rộng hay diện tích của biển số sẽ giúp xác định được đúng biển số. Cuối cùng, ta sẽ xác định vị trí của biển số trong ảnh bằng cách vẽ Contour bao quanh.

3.2 Chuyển ảnh xám

Ảnh xám (Gray Scale) đơn giản là một hình ảnh trong đó các màu là các sắc thái của màu xám với 256 cấp độ xám biến thiên từ màu đen đến màu trắng, nằm trong giải giá trị từ 0 đến 255, nghĩa là cần 8 bits hay 1 byte để biểu diễn mỗi điểm ảnh này. Lý do cần phải phân biệt giữa ảnh xám và các ảnh khác nằm ở việc ảnh xám cung cấp ít thông tin hơn cho mỗi pixel. Với ảnh thông thường thì mỗi pixel thường được cung cấp 3 trường thông tin trong khi với ảnh xám chỉ có 1 trường thông tin, việc giảm khối lượng thông tin giúp tăng tốc độ xử lý, đơn giản hóa giải thuật nhưng vẫn đảm bảo các tác vụ cần thiết

Ở bài này em sẽ chuyển ảnh xám từ hệ màu HSV thay vì RGB vì với không gian màu HSV ta có ba giá trị chính là: Vùng màu (Hue), độ bão hòa (Saturation), cường độ sáng (Value). Vì lý do đó không gian màu HSV thích nghi tốt hơn đối với sự thay đổi ánh sáng từ môi trường ngoài. Khi chuyển đổi, ảnh xám ta cần là ma trận các giá trị cường độ sáng tách ra từ hệ màu HSV.

3.3 Lọc biển số với contour

3.3.1 Một số phương pháp tìm contour

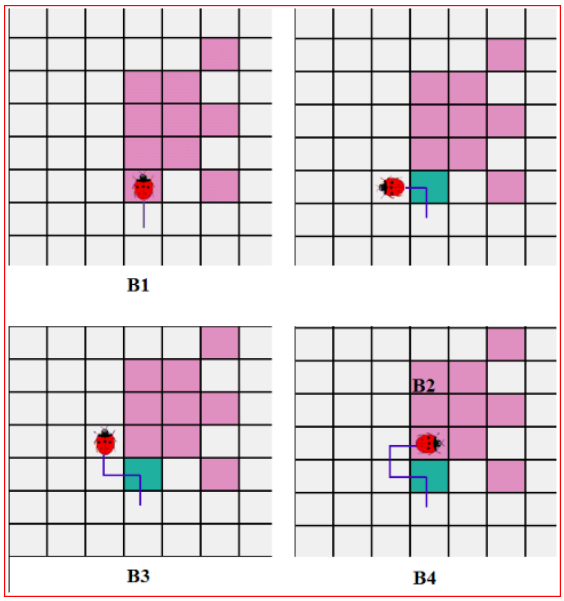
Có thể hiểu Contour là tập hợp các điểm tạo thành đường cong kín bao quanh một đối tượng nào đó. Thường dùng để xác định vị trí, đặc điểm của đối tượng. Có 4 thuật toán Contour Tracing chung nhất. Hai trong số đó có tên là: Square Tracing algorithm và Moore – Neighbor Tracing là dễ để thực hiện và thường xuyên được dùng để dò tìm contour của một mẫu. Với thư viện OpenCV người ta áp dụng thuật toán Suzuki’s Contour tracing. Dưới đây em sẽ trình bày kĩ hơn về 3 phương pháp trên:

a. Thuật toán Square Tracing

Duyệt từ pixel ngoài cùng bên trái phía dưới, đi lên cho tới khi gặp pixel có giá trị bằng 255 (pixel này sẽ được gọi là pixel start) thì bắt đầu di chuyển theo quy tắc sau:

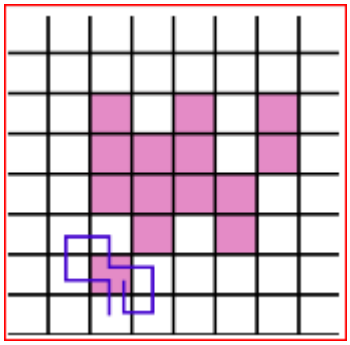
- Nếu gặp Pixel có giá trị bằng 255 thì rẽ trái.
- Nếu gặp Pixel có giá trị bằng 0 thì rẽ phải.
- Di chuyển cho tới khi quay lại pixel start thì dừng lại.

Hình sau mô tả cách hoạt động của thuật toán:

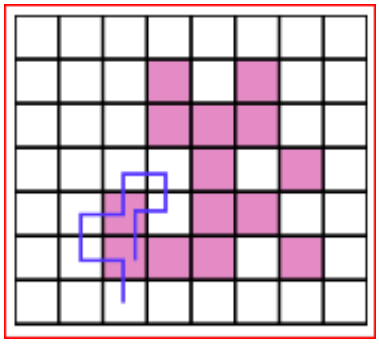


Hình 3.7 - 1 Thuật toán Square Tracing

Thuật toán sẽ kết thúc đúng khi di chuyển vào pixel start lần thứ 2 sau khi đi qua n pixel khác và theo đúng hướng đi vào pixel start lần đầu tiên. Và sai khi di chuyển vào pixel start mà không đúng hướng ban đầu. Vậy thuật toán này chỉ chạy đúng trên đối tượng 4 - connected.



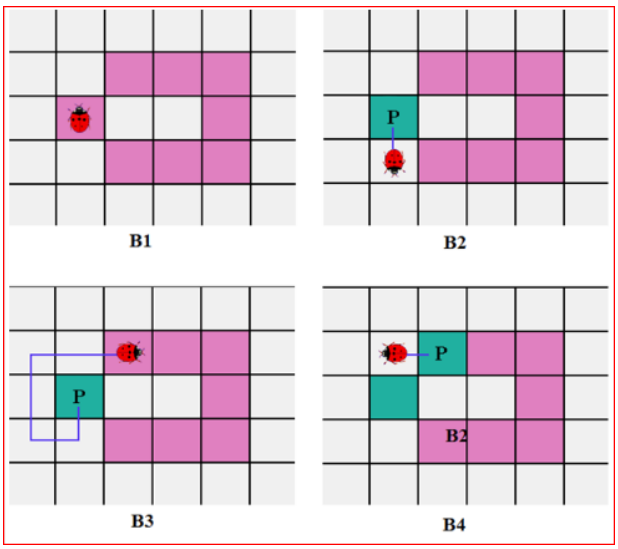
Hình 3.7 - 2 Thuật toán Square Tracing chạy đúng



Hình 3.7 - 3 Thuật toán Square Tracing chạy sai

a. Thuật toán Moore – Neighbor

Thuật toán này có chút khác biệt so với thuật toán Square Tracking, cụ thể: khi gặp pixel có giá trị bằng 255 đầu tiên (pixel start) thì ta sẽ quay lại pixel trước đó, sau đó đi vòng qua các pixel thuộc 8-connected theo chiều kim đồng hồ cho tới khi gặp pixel khác có giá trị bằng 255. Và điệu kiện kết thúc cũng giống như thuật toán Square Tracking.



Hình 3.7 - 4 Thuật toán Moore - Neighbor.

b. Thuật toán Suzuki’s Tracing

Đây là thuật toán được thư viện OpenCV sử dụng, ngoài khả năng xác định được biên của vật thể như hai phương pháp trên. Phương pháp Suzuki’s Tracing còn có khả năng phân biệt được đường biên ngoài (Outer) và đường biên trong (Hole) của vật thể.

Hàm trong OpenCV được biểu diễn như sau:

findContours(InputOutputArray image, OutputArrayOfArrays contours, OutputArray hierarchy, int mode, int method, Point offset=Point())

Các tham số:

image : hình ảnh cần tìm biên, là ảnh nhị phân.
contours : lưu trữ các đường biên tìm được, mỗi đường biên được lưu trữ dưới dạng một vector của các điểm.
hierarchy : chứa thông tin về hình ảnh như số đường viền, xếp hạng các đường viền theo kích thước, trong ngoài,..
mode :
CV_RETR_EXTERNAL : khi sử dựng cờ này nó chỉ lấy ra những đường biên bên ngoài, nhưng biên bên trong của vật thể bị loại bỏ.
CV_RETR_LIST : Khi sử dụng cờ này nó lấy ra tất cả các đường viền tìm được.
CV_RETR_CCOMP : khi sử dụng cờ này nó lấy tất cả những đường biên và chia nó làm 2 level, những đường biên bên ngoài đối tượng, và những đường biên bên trong đối tượng.
CV_RETR_TREE : khi sử dụng cờ này nó lấy tất cả các đường biên và tạo ra một hệ thống phân cấp đầy đủ của những đường lồng nhau.
method :
CV_CHAIN_APPROX_NONE : sử dụng cờ này sẽ lưu trữ tất cả các điểm của đường viền .
CV_CHAIN_APPROX_SIMPLE : Ví dụ : một hình chữ nhật sẽ được mã hoá bằng toạ độ của 4 đỉnh.
CV_CHAIN_APPROX_TC89_L1 or CV_CHAIN_APPROX_TC89_KCOS : Áp dụng thuật toán xấp xỉ Tech-Chin.

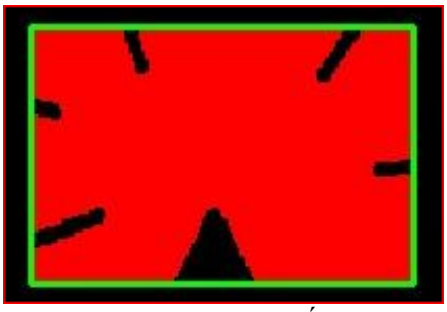
Hình 3.7 - 5 Vẽ Contour với OpenCV

Trong ảnh, những đường màu hồng là đường contour bao quanh vật thể, nhưng vì có quá nhiều đường bao quanh các vật thể không phải biển số nên chúng ta sẽ áp dụng những đặc trưng riêng về tỉ lệ cao/rộng, diện tích trong khung hình cố định như ở mục 2.1 để lọc ra đúng biển số cần tìm.

3.3.2 Lọc biển số

Đầu tiên ta làm xấp xỉ contour thành một hình đa giác và chỉ lấy những đa giác nào chỉ có 4 cạnh. Nghĩa là lúc xấp xỉ contour bộ nhớ chỉ ghi nhớ vị trí các đỉnh của đa giác đó thành một mảng. Số cạnh của đa giác sẽ bằng số đỉnh và bằng chiều dài của mảng đó.

Hình 3.7 - 6 Contour chưa xấp xỉ đa giác



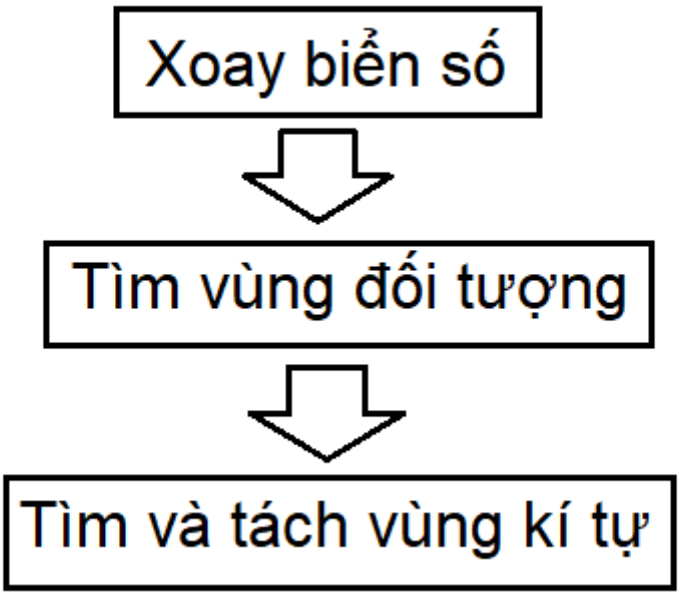
Hình 3.7 - 7 Contour đã xấp xỉ đa giác

Tiếp theo ta tính toán tỉ lệ cao/rộng và diện tích của biển số phù hợp, sau đó ta lưu tất cả những biển số có trong hình dưới dạng tọa độ các đỉnh. Từ đây, ta cắt hình ảnh biển số từ các tọa độ vị trí đã biết để phục vụ cho mục đích tiếp theo "Tách các kí tự trong biển số". Lưu ý ở đây ta cắt từ ảnh nhị phân luôn để máy tính xử lý nhanh hơn, tốn ít thời gian hơn.

4. PHÂN ĐOẠN KÍ TỰ

4.1. Hướng giải quyết

Ở giai đoạn này có những bước chính sau: Xoay biển số để tăng khả năng nhận diện, Tìm tất cả các vùng kín cho là kí tự và lọc ra những kí tự đúng. Tách hình ảnh nhưng kí tự đó ra và đưa vào bộ nhận diện.



Hình 4.1 - 1 Các bước chính trong phân đoạn kí tự

4.2. Xoay biển số

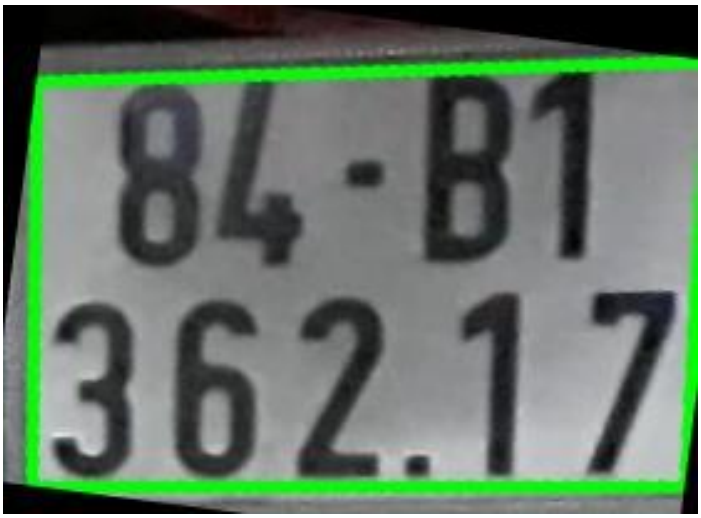
Khi chụp ảnh đầu vào, không phải lúc nào biển số cũng ở chính diện, có thể bị méo sang trái, sang phải, nghiêng góc dẫn đến nếu cứ sử dụng ảnh biển số đã cắt mà không điều chỉnh góc độ dẫn đến ảnh kí tự được cắt ra đưa vào bộ nhận diện rất dễ bị sai. Ví dụ giữa số 1 và số 7, số 2 và chữ Z, chữ B và số 8,...

Hình 4.2 - 1 Ảnh biển số chưa xoay

Phương pháp xoay ảnh em sử dụng ở đây là:

1. Lọc ra tọa độ 2 đỉnh A,B nằm dưới cùng của biển số

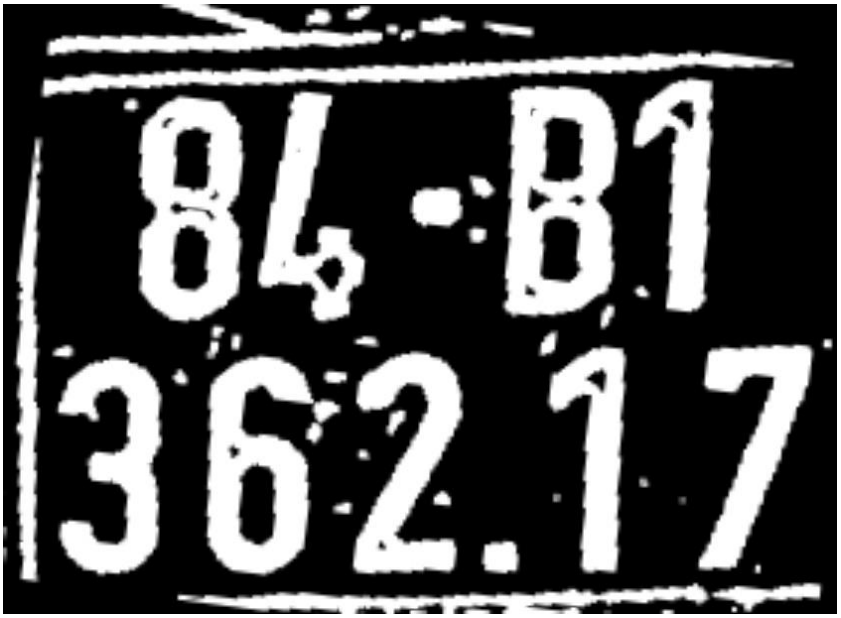
2. Từ 2 đỉnh có tọa độ lần lượt là A(x1, y1) và B(x2,y2) ta có thể tính được cạnh đối và cạnh kề của tam giác ABC.
3. Ta tính được góc quay $\widehat{BAC} = \tan^{-1}\left(\frac{cạnh\ đối}{cạnh\ kề}\right) = \tan^{-1}\left(\frac{BC}{AC}\right)$.
4. Xoay ảnh theo góc quay đã tính. Nếu ngược lại điểm A nằm cao hơn điểm B ta cho góc quay âm.



Hình 4.2 - 2 Ảnh biển số đã xoay

4.3. Tìm vùng đối tượng

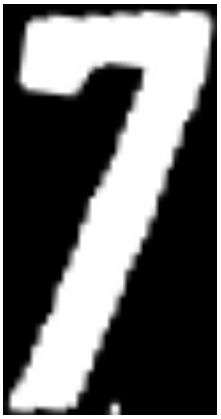
Từ ảnh nhị phân, ta lại tìm contour cho các kí tự (phần màu trắng). Sau đó vẽ những hình chữ nhật bao quanh các kí tự đó. Tuy nhiên việc tìm contour này cũng bị nhiễu dẫn đến việc máy xử lý sai mà tìm ra những hình ảnh không phải kí tự. Ta sẽ áp dụng các đặc điểm về tỉ lệ chiều cao/rộng của kí tự, diện tích của kí tự so với biển số.



Hình 4.3 - 1 Ảnh nhị phân

4.4. Tìm và tách kí tự

Sau khi đã nhận dạng từng kí tự bằng hình chữ nhật và cũng đã có tọa độ vị trí 4 đỉnh của hình đó, ta lúc này có thể cắt hình ảnh kí tự đó ra phục vụ cho giai đoạn sau "Nhận diện kí tự". Lưu ý ở đây ta cắt ảnh nhị phân chứ không cắt từ ảnh gốc.



Hình 4.4 - 1 Ảnh kí tự sau khi cắt

5. NHẬN DIỆN KÍ TỰ

Thực chất quá trình nhận diện kí tự chính là quá trình chuyển đổi từ một hình ảnh là ma trận giá trị các điểm ảnh về một dạng thông tin khác như trong đề tài này là chữ số để có thể giao tiếp với người dùng. Để hiểu hơn về nhận diện, chúng ta cần tìm hiểu về phương pháp nhận diện trong học máy (machine learning).

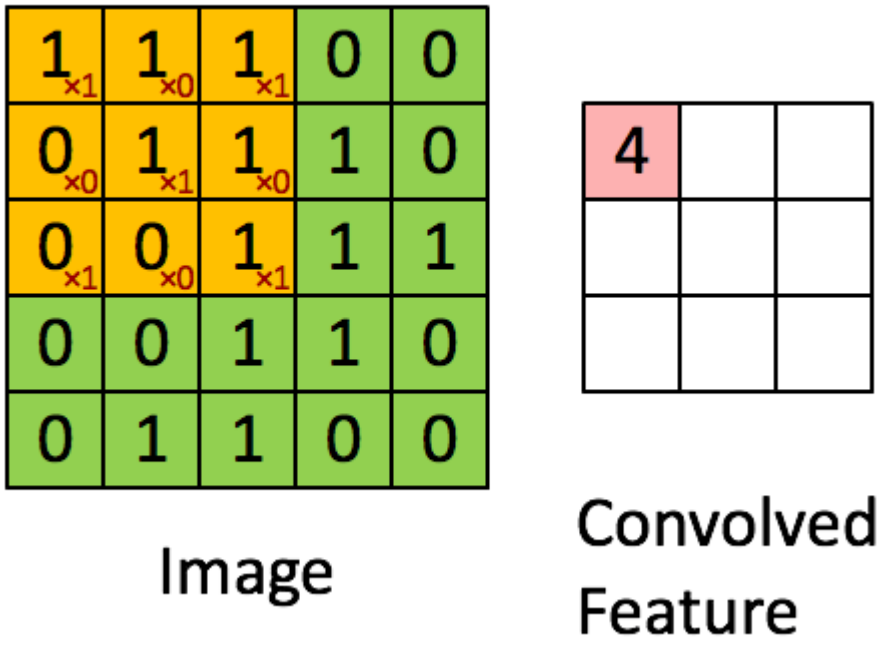
5.1. Sử dụng mô hình CNN

Convolutional Neural Network (CNNs – Mạng nơ-ron tích chập) là một trong những mô hình Deep Learning tiên tiến. Nó giúp cho chúng ta xây dựng được những hệ thống thông minh với độ chính xác cao như hiện nay.

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

Convolutional là gì?

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

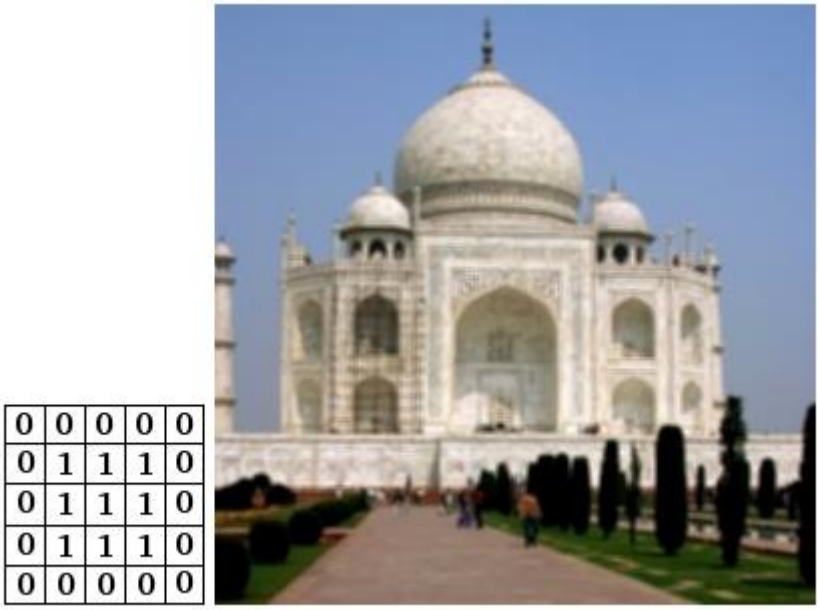


Link ảnh: https://topdev.vn/blog/wp-content/uploads/2019/08/Convolution_schematic.gif

Các convolutional layer có các parameter(kernel) đã được học để tự điều chỉnh lấy ra những thông tin chính xác nhất mà không cần chọn các feature.

Trong hình ảnh ví dụ trên, ma trận bên trái là một hình ảnh trắng đen được số hóa. Ma trận có kích thước 5×5 và mỗi điểm ảnh có giá trị 1 hoặc 0 là giao điểm của dòng và cột. Convolution hay tích chập là nhân từng phần tử trong ma trận 3. Sliding Window hay còn gọi là kernel, filter hoặc feature detect là một ma trận có kích thước nhỏ như trong ví dụ trên là 3×3. Convolution hay tích chập là nhân từng phần tử bên trong ma trận 3×3 với ma trận bên trái. Kết quả được một ma trận gọi là Convoled feature được sinh ra từ việc nhận ma trận Filter với ma trận ảnh 5×5 bên trái.

	0	1	0	
	1	-4	1	
	0	1	0	



0	0	0	0	0
0	1	1	1	0
0	1	1	1	0
0	1	1	1	0
0	0	0	0	0

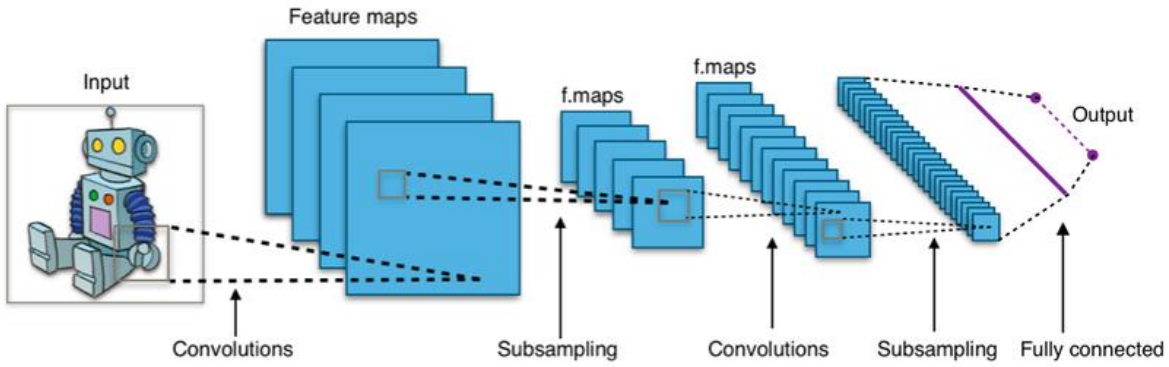
Cấu trúc của mạng CNN

Mạng CNN là một tập hợp các lớp Convolution chồng lên nhau và sử dụng các hàm nonlinear activation như ReLU và tanh để kích hoạt các trọng số trong các node. Mỗi một lớp sau khi thông qua các hàm kích hoạt sẽ tạo ra các thông tin trừu tượng hơn cho các lớp tiếp theo. Mỗi một lớp sau khi thông qua các hàm kích hoạt sẽ tạo ra các thông tin trừu tượng hơn cho các lớp tiếp theo. Trong mô hình mạng truyền ngược (feedforward neural network) thì mỗi neural đầu vào (input node) cho mỗi neural đầu ra trong các lớp tiếp theo.

Mô hình này gọi là mạng kết nối đầy đủ (fully connected layer) hay mạng toàn vẹn (affine layer). Còn trong mô hình CNNs thì ngược lại. Các layer liên kết được với nhau thông qua cơ chế convolution.

Layer tiếp theo là kết quả convolution từ layer trước đó, nhờ vậy mà ta có được các kết nối cục bộ. Như vậy mỗi neuron ở lớp kế tiếp sinh ra từ kết quả của filter áp đặt lên một vùng ảnh cục bộ của neuron trước đó. Mỗi một lớp được sử dụng các filter khác nhau thông thường có hàng trăm hàng nghìn filter như vậy và kết hợp kết quả của chúng lại. Ngoài ra có một số layer khác như pooling/subsampling layer dùng để chắt lọc lại các thông tin hữu ích hơn (loại bỏ các thông tin nhiễu).

Trong quá trình huấn luyện mạng (traning) CNN tự động học các giá trị qua các lớp filter dựa vào cách thức mà bạn thực hiện. Ví dụ trong tác vụ phân lớp ảnh, CNNs sẽ cố gắng tìm ra thông số tối ưu cho các filter tương ứng theo thứ tự raw pixel > edges > shapes > facial > high-level features. Layer cuối cùng được dùng để phân lớp ảnh.



Trong mô hình CNN có 2 khía cạnh cần quan tâm là tính bất biến (Location Invariance) và tính kết hợp (Compositionality). Với cùng một đối tượng, nếu đối tượng này được chiếu theo các gốc độ khác nhau (translation, rotation, scaling) thì độ chính xác của thuật toán sẽ bị ảnh hưởng đáng kể. Pooling layer sẽ cho bạn tính bất biến đối với phép dịch chuyển (translation), phép quay (rotation) và phép co giãn (scaling). Tính kết hợp cục bộ cho ta các cấp độ biểu diễn thông tin từ mức độ thấp đến mức độ cao và trừu tượng hơn thông qua convolution từ các filter. Đó là lý do tại sao CNNs cho ra mô hình với độ chính xác rất cao. Cũng giống như cách con người nhận biết các vật thể trong tự nhiên.

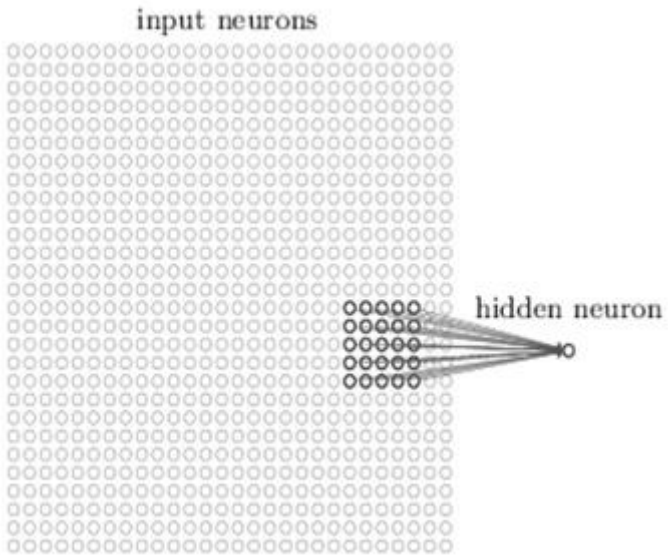
Mạng CNN sử dụng 3 ý tưởng cơ bản:

các trường tiếp nhận cục bộ (local receptive field)

trọng số chia sẻ (shared weights)

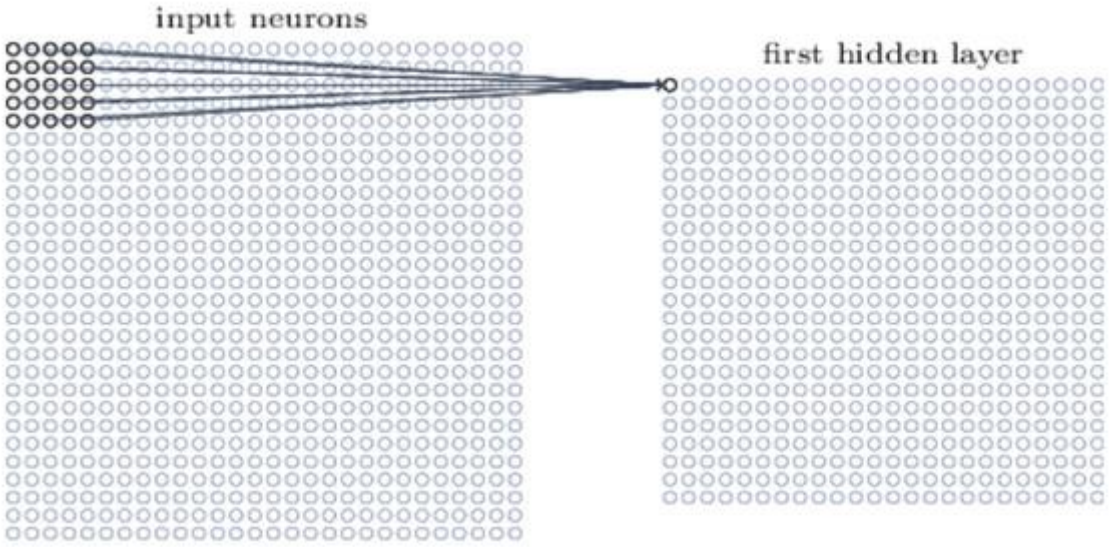
tổng hợp (poolin Trường tiếp nhận cục bộ (local receptive field)

Đầu vào của mạng CNN là một ảnh. Ví dụ như ảnh có kích thước 28×28 thì tương ứng đầu vào là một ma trận có 28×28 và giá trị mỗi điểm ảnh là một ô trong ma trận. Trong mô hình mạng ANN truyền thống thì chúng ta sẽ kết nối các neuron đầu vào vào tầng ảnh. Tuy nhiên trong CNN chúng ta không làm như vậy mà chúng ta chỉ kết nối trong một vùng nhỏ của các neuron đầu vào như một filter có kích thước 5×5 tương ứng (28- 5 + 1) 24 điểm ảnh đầu vào. Mỗi một kết nối sẽ học một trọng số và mỗi neuron ẩn sẽ học một bias. Mỗi một vùng 5×5 đấy gọi là một trường tiếp nhận cục bộ.

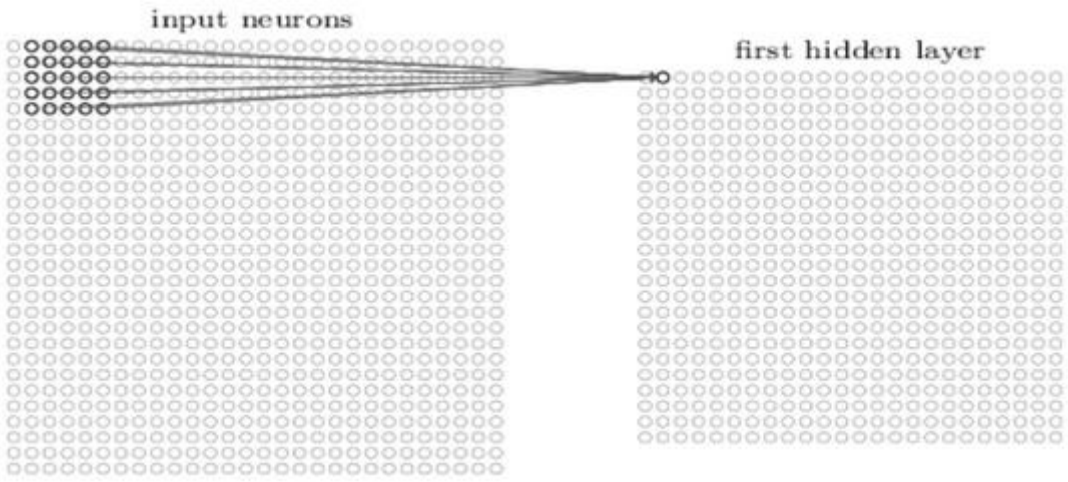


Một cách tổng quan, ta có thể tóm tắt các bước tạo ra 1 hidden layer bằng các cách sau:

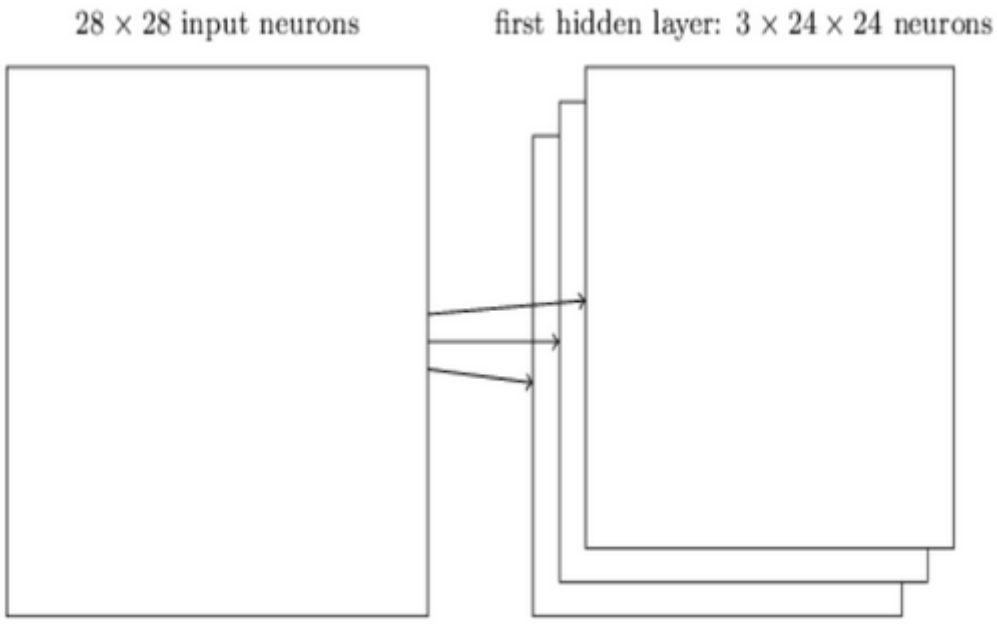
Tạo ra neuron ẩn đầu tiên trong lớp ẩn 1



Dịch filter qua bên phải một cột sẽ tạo được neuron ẩn thứ 2.



Với bài toán nhận dạng ảnh người ta thường gọi ma trận lớp đầu vào là feature map, trọng số xác định các đặc trương là shared weight và độ lệch xác định một feature map là shared bias. Như vậy đơn giản nhất là qua các bước trên chúng ta chỉ có 1 feature map. Tuy nhiên trong nhận dạng ảnh chúng ta cần nhiều hơn một feature map.

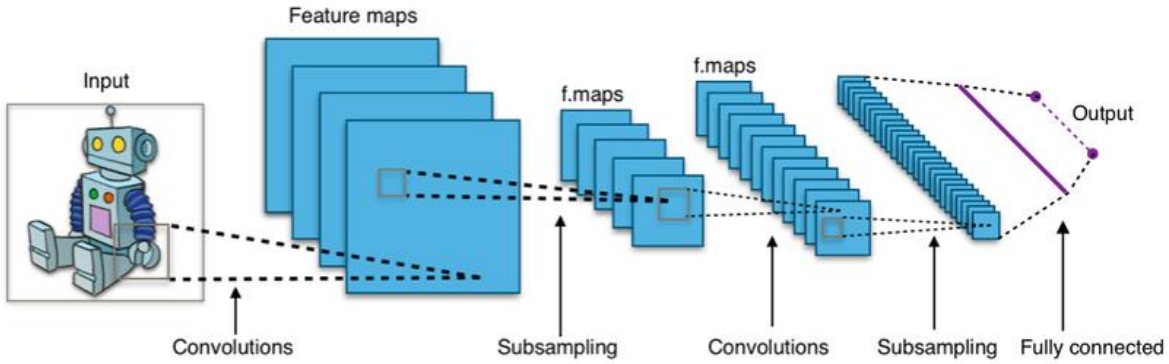


Như vậy, local receptive field thích hợp cho việc phân tách dữ liệu ảnh, giúp chọn ra những vùng ảnh có giá trị nhất cho việc đánh giá phân lớp.

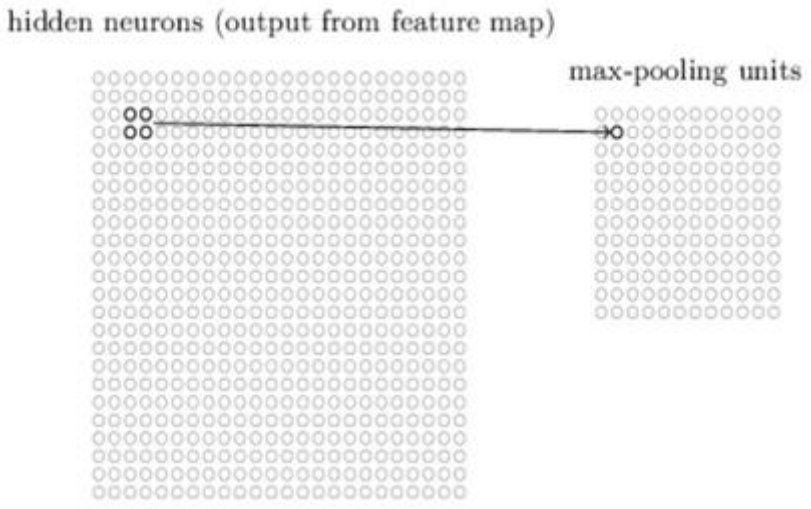
Tóm lại, một convolutional layer bao gồm các feature map khác nhau. Mỗi một feature map giúp detect một vài feature trong bức ảnh. Lợi ích lớn nhất của trọng số chia sẻ là giảm tối đa số lượng tham số trong mạng CNN.

Lớp tổng hợp (pooling layer)

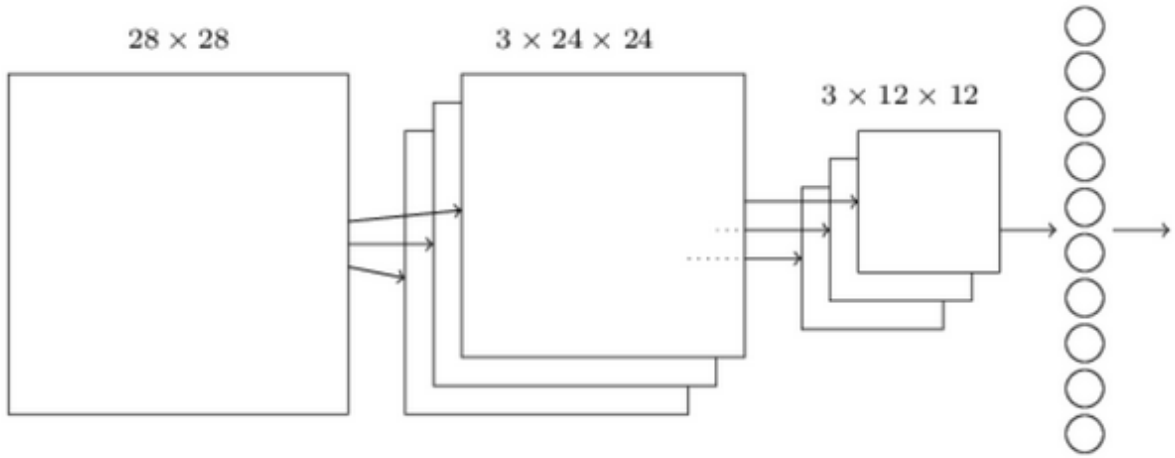
Lớp pooling thường được sử dụng ngay sau lớp convulational để đơn giản hóa thông tin đầu ra để giảm bớt số lượng neuron.



Thủ tục pooling phổ biến là max-pooling, thủ tục này chọn giá trị lớn nhất trong vùng đầu vào 2×2.



Như vậy qua lớp Max Pooling thì số lượng neuron giảm đi phân nửa. Trong một mạng CNN có nhiều Feature Map nên mỗi Feature Map chúng ta sẽ cho mỗi Max Pooling khác nhau. Chúng ta có thể thấy rằng Max Pooling là cách hỏi xem trong các đặc trưng này thì đặc trưng nào là đặc trưng nhất. Ngoài Max Pooling còn có L2 Pooling. Cuối cùng ta đặt tất cả các lớp lại với nhau thành một CNN với đầu ra gồm các neuron với số lượng tùy bài toán.



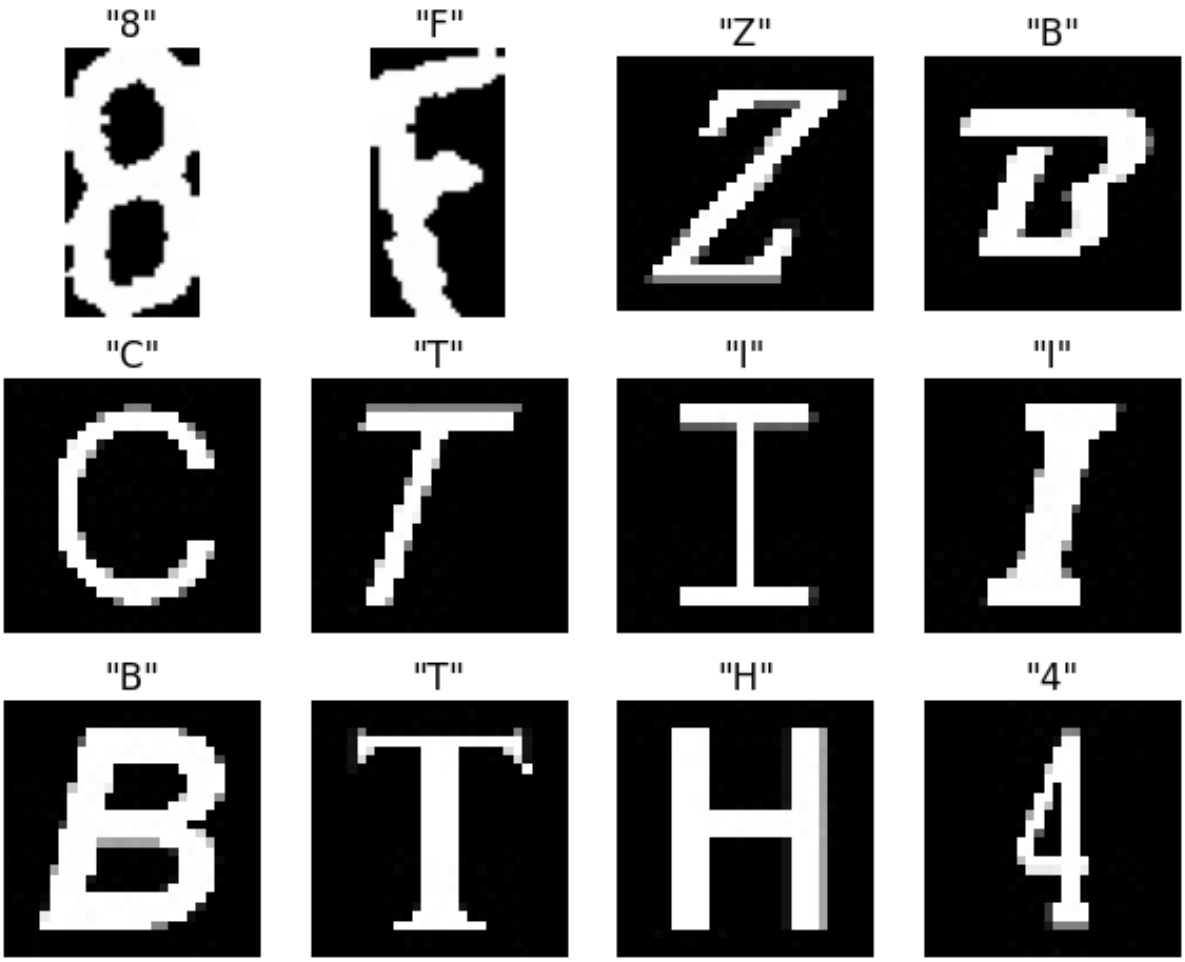
2 lớp cuối cùng của các kết nối trong mạng là một lớp đầy đủ kết nối (fully connected layer) . Lớp này nối mọi nơron từ lớp max pooled tới mọi nơron của tầng ra.

5.2. Hướng giải quyết

Ở giai đoạn cuối này được thực hiện theo những bước sau:

1. Tạo tập dữ liệu để huấn luyện.
2. Huấn luyện mô hình CNN.
3. Đưa hình ảnh từ bước “Phân đoạn kí tự” vào mô hình CNN đã tạo để đưa ra kết quả.
4. In ra kết quả biển số.

Bước 1 và 2 ta sẽ tạo ra mô hình CNN riêng biệt với code chính. Để khi cần nhận diện kí tự ta không cần phải làm lại các bước từ đầu. Đầu tiên em có tập dữ liệu (tập hình ảnh của các chữ số và kí tự) để train từ đường link. Kết quả có dạng như sau:



Hình 5.2 - 1 Tập dữ liệu huấn luyện

Bước 3 và 4. Ta thực hiện đưa ảnh đang xét vào và tính khoảng cách đến tất cả các điểm trong mẫu, kết quả sẽ là mã số biển xe đại điện cho hình ảnh đó. Cuối cùng ta in biển số xe ra terminal. Tuy nhiên ở Việt Nam có hai loại biển số là biển một hàng và biển hai hàng. Về ý tưởng chung để phân biệt hai hàng này ta dựa vào vị trí của hình ảnh kí tự, nếu vị trí nằm thấp 1/3 chiều cao của biển số thì kí tự sẽ được xếp vào hàng một. Ngược lại sẽ được xếp vào hàng hai.

Hình 5.2 - 2 Biển số được phát hiện trên ảnh

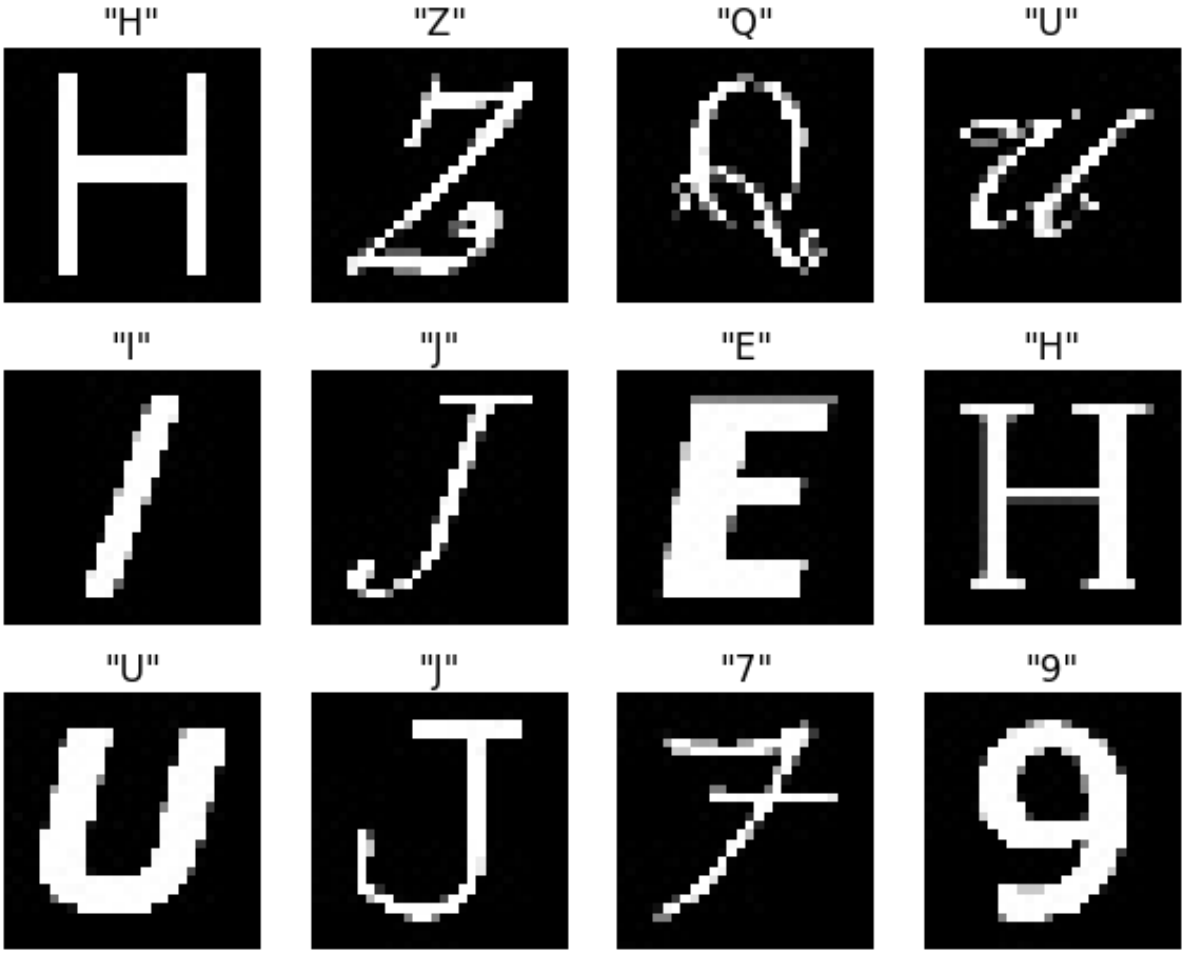
Hình 5.2 – 3. Ảnh biển số được cắt ra để xử lý.

Hình 5.2 - 3 Biển số xe được in ra trên hình gốc

6. KẾT QUẢ THỰC HIỆN

6.1. Cách thức đo đạc, thử nghiệm

Trước tiên em sẽ sử dụng bộ mô hình đã train sẵn có tên Wpod-net để tìm ra vùng ảnh chứa biển số xe[21]. Bộ dữ liệu em sử dụng để train mô hình CNN nhận diện kí tự là các kí tự chữ số từ 0 đến 9, từ A đến Z. Sau đó sẽ nhặt ra bất kì 1 bộ gồm 12 kí tự và in ra màn hình.

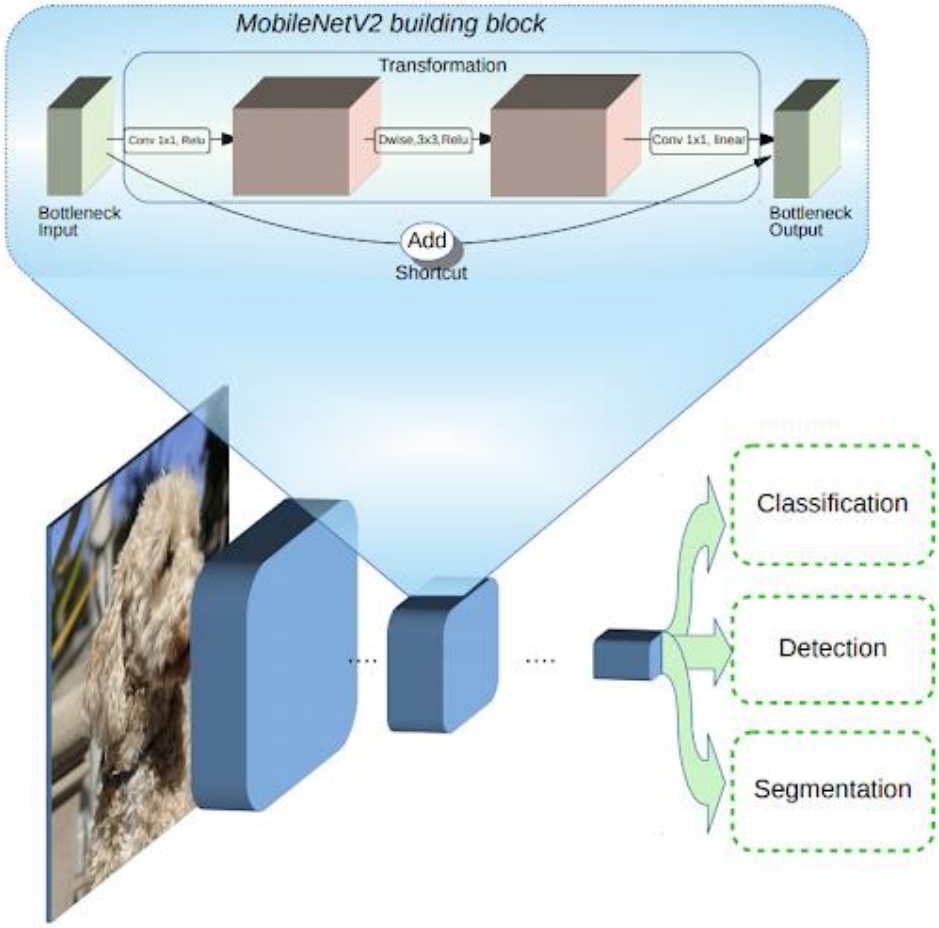


Ở đây ta sẽ có 36 nhãn gán với 36 kí tự chữ số gồm 26 chữ trong bảng chữ cái alphabet và 10 kí tự từ các số từ 0 đến 9. Sau khi các ảnh được đưa vào ta sẽ tiến hành chuyển đổi sang dạng array để xử lý.

```
# Arange input data and corresponding labels
X=[]
labels=[]

for image_path in dataset_paths:
  label = image_path.split(os.path.sep)[-2]
  image=load_img(image_path,target_size=(80,80))
  image=img_to_array(image)

  X.append(image)
  labels.append(label)

X = np.array(X,dtype="float16")
labels = np.array(labels)

print("[INFO] Find {:d} images with {:d} classes".format(len(X),len(set(labels))))


# perform one-hot encoding on the labels
lb = LabelEncoder()
lb.fit(labels)
labels = lb.transform(labels)
y = to_categorical(labels)

# save label file so we can use in another script
np.save('/content/drive/MyDrive/BTLAI2/license_character_classes.npy', lb.classes_)
```

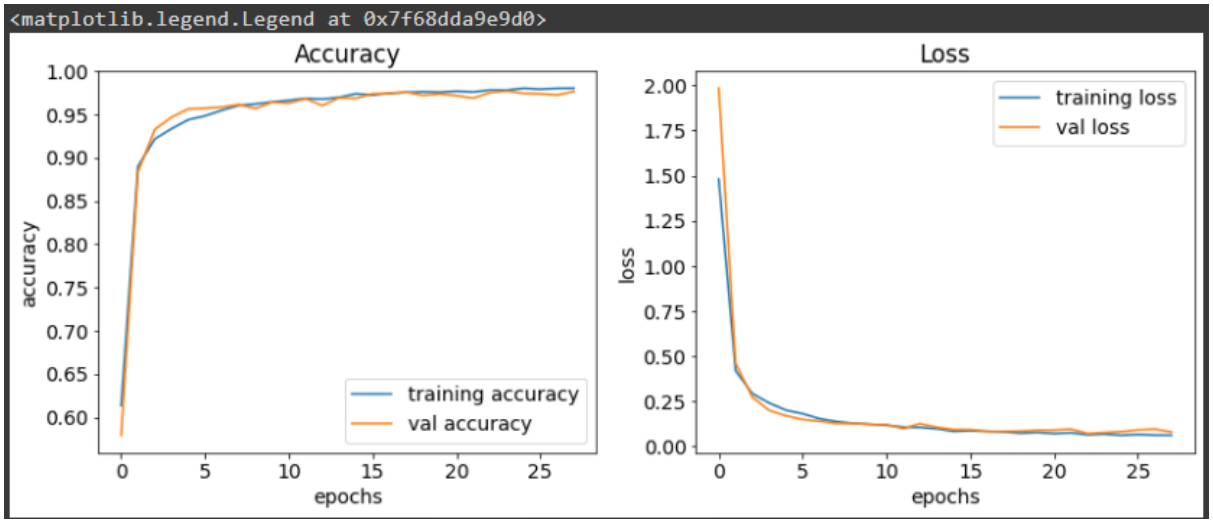
```
[INFO] Find 37623 images with 36 classes
```

Tiếp theo ta sẽ sử dụng mô hình MobileNetV2 (thường được dung trên các thiết bị có cấu hình yếu như điện thoại di động, máy tính Pi, …).



Sau khi train xong ta sẽ save lại mô hình dưới dạng 1 file json và 1 file trọng số .h5 để tiện load lại khi cần. Em sẽ sử dụng bộ dữ liệu gồm 1750 ảnh xe máy ra vào trong hầm và chọn ra 1 ảnh bất kì để đoán nhận biển số.

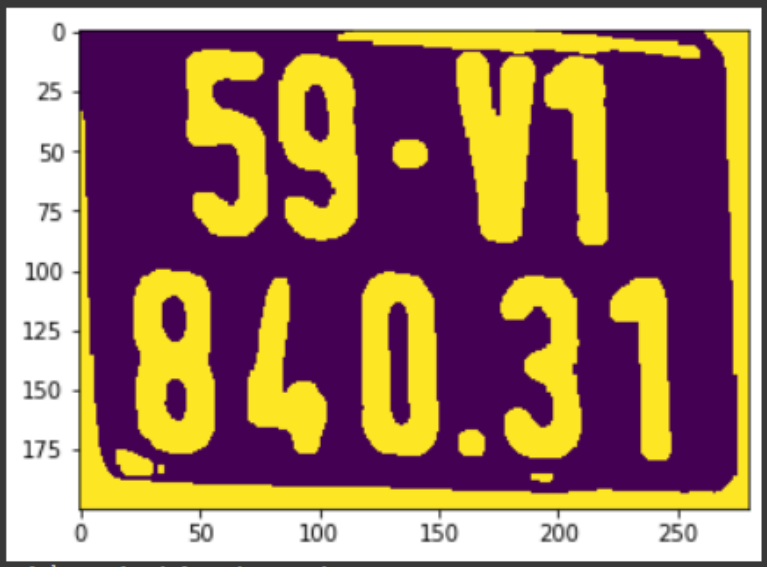
6.2. Kết quả và giải thích



- Ta thấy kết quả sau khi test, Accuracy và Loss cho kết quả khá tốt, khá fit giữa kết quả của tập train và tập val. Khi train qua càng nhiều epochs thì kết quả càng tốt, kết quả tốt nhất ở epoch thứ 25. Nhưng không nên train quá nhiều vì kết quả sẽ không thay đổi đáng kể về sau (<50 epochs).

- Kết quả khi đoán nhận biển số xe như sau:

```
from matplotlib import pyplot as plt
plt.imshow(binary_plate_image, interpolation='nearest')
plt.show()
print("Biển số nhận dạng được: ")
print(string_LP(charater_top, model, labels))
print(string_LP(charater_bot, model, labels))
```




```
Biển số nhận dạng được:
59V1
84031
```

7. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

7.1 . Kết luận

Từ những kết quả thu được ở chương trên em nhận thấy phương pháp nhận diện biển số xe bằng xử lý ảnh và thuật toán CNN có những ưu và khuyết điểm sau:

- Ưu điểm:
  - Dễ cài đặt và sử dụng.
  - Khá nhẹ nếu sử dụng các mô hình như MobileNet,…
  - Phù hợp cho đối tượng sinh viên muốn tìm hiểu căn bản về xử lý ảnh hay trí tuệ nhân tạo.
- Khuyết điểm:
  - Nhận diện kém với sự phản chiếu của biển số, sự di ảnh, chói sáng từ môi trường ngoài, những biển có phần chữ số không rõ ràng, với biển số xe ô tô.

Vì vậy tốt nhất cần đặt camera cố định, với môi trường ánh sáng xung quanh được cài đặt trước. Phông nền cần hạn chế tối da nhưng chi tiết lóe sáng gây nhiễu. Phương pháp này vẫn còn cần sự giám sát của con người nhiều chứ chưa thể hoàn toàn tự động.

7.2 . Hướng phát triển

- Cần thay đổi thuật toán nhận diện CNN sang những thuật toán khác tinh vi và phức tạp hơn như SVM hoặc có thể sử dụng những bộ thư viện đã có sẵn trên thế giới như YOLO, YOLOv3 ....
- Sử dụng camera chuyên dụng cho việc nhận diện biển số xe vì có khả năng chống chịu với sương mù, đêm tối, chói sáng,...
- Sử dụng các thuật toán xử lý ảnh khác để xác định vị trí biển số tốt hơn như phương pháp biến đổi Hough để nhận diện đường thẳng, xác định bằng màu sắc, những thuật toán làm hạn chế sự di ảnh khi xe đang di chuyển.
- Kết hợp với những chương trình khác để quản lý kho bãi, quản lý các phương tiện đang tham gia giao thông, tìm xe thất lạc, theo dấu,...

8. TÀI LIỆU THAM KHẢO

[1] M. J. Ahmed, M. Sarfaz, A. Zidouri, and K. G. AI-Khatib, "License plate recognition system," *Proc. IEEE Int. Conf. Electron. Circuits, Syst.*, vol. 2, no. January, pp. 898–901, 2003, doi: 10.1109/ICECS.2003.1301932.

[2] C. N. E. Anagnostopoulos, "License plate recognition: A brief tutorial," *IEEE Intell. Transp. Syst. Mag.*, vol. 6, no. 1, pp. 59–67, 2014, doi: 10.1109/MITS.2013.2292652.

[3] A. Badr, M. M. Abdel, A. M. Thabet, and A. M. Abdelsadek, "Automatic number plate recognition system," *Ann. Univ. Craiova, Math. Comput. Sci. Ser.*, vol. 38, no. 1, pp. 62–71, 2011, doi: 10.5120/ijca2018917277.

[4] S. L. Chang, L. S. Chen, Y. C. Chung, and S. W. Chen, "Automatic License Plate Recognition," *IEEE Trans. Intell. Transp. Syst.*, vol. 5, no. 1, pp. 42–53, 2004, doi: 10.1109/TITS.2004.825086.

[5] N. D. Linh, N. Van Nhan, and D. Van Dat, "12.pdf," *Tap chi thong tin khoa hoc va cong nghe Quang Binh*, 2018.

[6] D. V. R. Mohan, M. T. Communication, S. Srkr, and E. College, "Number Plate Recognition by using open CV- Python," pp. 4987–4992, 2019.

[7] Nguyễn Vĩnh An, "So sánh một số phương pháp phát hiện biên," *Tạp chí khoa học Trường Đại học Quốc gia Hà Nội*, vol. 31, no. 2, pp. 1–7, 2015.

[8] F. Patel, J. Solanki, V. Rajguru, and A. Saxena, "Recognition of Vehicle Number Plate Using Image Processing Technique," *Adv. Emerg. Med.*, vol. 7, no. 1, pp. 2–8, 2018, doi: 10.18686/aem.v7i1.

[9] L. F. Sanchez, "Automatic Number Plate Recognition System Using Machine Learning Techniques," no. August, pp. 2017–2018, 2018.

[10] K. Sarbjit, "An Efficient Approach for Automatic Number Plate Recognition System under Image Processing," *Int. J. Adv. Res. Comput. Sci.*, vol. 5, no. (6), pp. 43–50, 2014.

[11] N. Simin, F. Choong, and C. Mei, "Automatic Car-plate Detection and Recognition System," pp. 113–114, 2013.

[12] G. D. Yeshwant, S. Maiti, and P. B. Borole, "Automatic Number Plate Recognition System (ANPR System)," *Int. J. Eng. Res.*, vol. 3, no. 7, p. 5, 2014, [Online]. Available: https://www.ijert.org/research/automatic-number-plate-recognition-system-anpr-system-IJERTV3IS071132.pdf.

[13] A. Zelinsky, *Learning OpenCV---Computer Vision with the OpenCV Library (Bradski, G.R. et al.; 2008)[On the Shelf]*, vol. 16, no. 3. 2009.

[14] Chris Dahms (2016), OpenCV 3 License Plate Recognition Python. https://www.youtube.com/watch?v=fJcl6Gw1D8k

[15] OpenCV. Morphological Transformations. https://docs.opencv.org/3.4/d9/d61/tutorial_py_morphological_ops.html

[16] Nhận dạng biển số xe với opencv step by step. https://thorpham.github.io/blog/2018/04/11/regconite-plate-car/

[17] Find and draw Contours - OpenCV 3.4 with python 3. https://www.youtube.com/watch?v=_aTC-Rc4Io0

[18] Contour Features. https://docs.opencv.org/trunk/dd/d49/tutorial_py_contour_features.html

[19] Tìm hiểu về Contour, moments trong xử lý ảnh. https://congdongopencv.blogspot.com/2017/11/tim-hieu-ve-contour-moments-trong-xu-ly.html

[20] Suzuki’s contour tracing algorithm OpenCV - Python. https://theailearner.com/tag/suzuki-contour-algorithm-opencv/

[21] Wpod-net: https://github.com/sergiomsilva/alpr-unconstrained