**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**ĐỒ ÁN TỐT NGHIỆP**

**Xây dựng mô hình AI nhận diện biển số xe vào ra bãi gửi xe máy**

**Lê Thế Tài**

Tai.lt174180@sis.hust.edu.vn

**Ngành Công Nghệ Thông Tin**

**Chuyên ngành Khoa học máy tính**

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:** | TS. Nguyễn Bá Ngọc  Chữ ký của GVHD |
| **Bộ môn:** | Hệ Thống Thông Tin |
| **Viện:** | Công nghệ thông tin và truyền thông |
| **HÀ NỘI, 6/2018** | |

**ĐỀ TÀI TỐT NGHIỆP**

Giáo viên hướng dẫn

Ký và ghi rõ họ tên

**Lời cảm ơn**

**Tóm tắt nội dung đồ án**

Hiện nay số lượng các tòa nhà cao tầng, tòa chung cư, trung tâm thương mại,... cùng với số lượng phương tiện cá nhân ngày càng tăng dẫn đến các phương pháp quản lý phương tiện gửi đỗ xe bằng vé ngày càng tỏ ra kém hiệu quả, thường dẫn đến tình trạng ùn ứ ở cửa ra vào bãi đỗ xe trong khoảng thời gian cao điểm, giờ tan tầm,… đặc biệt là trong các trường học, trường đại học khi mà số lượng xe ra vào lớn. Ở những phương pháp truyền thống, việc xử lý xác nhận cho phép xe vào ra hoàn toàn bằng con người nên dẫn đến thời gian xử lý lâu, tốn kém nhiều nhân lực và có thể dẫn đến sai sót. Ngoài ra còn cần thêm nhân lực để thống kê, kiểm soát các phương tiện cá nhân ra vào mỗi ngày.

Sinh viên thực hiện

Ký và ghi rõ họ tên

Vấn đề đặt ra là tìm một giải pháp thích hợp để đưa máy móc, công nghệ vào việc quản lý xe ra vào một cách tự động và chính xác. Chính vì thế mà một hệ thống tự động nhận diện biển số xe được đề xuất qua đó có thể rút ngắn rất nhiều thời gian cũng như chi phí vận hành bãi gửi xe, đáp ứng nhu cầu gửi, lấy xe một cách nhanh chóng và hiệu quả.

Trong đồ án này, em sẽ xây dựng một mô hình tự động nhận diện biển số xe và xây dựng một hệ thống quản lý bãi đỗ xe để có thể tích hợp mô hình nhận diện thông minh vào hệ thống. Trang we quản lý sẽ hiển thị hình ảnh các xe vào ra, thống kê số lượng xe trong bãi, đăng ký vé xe theo tháng,… Các công nghệ được sử dụng để xây dựng web quản lý và mô hình nhận diện là FastApi và các mô hình Deep Learning

**MỤC LỤC**

[CHƯƠNG 1. TỔNG QUAN ĐỀ TÀI 1](#_Toc107626225)

[1.1 Giới thiệu đề tài 1](#_Toc107626226)

[1.2 Đối tượng sử dụng 1](#_Toc107626227)

[1.3 Mục tiêu đề tài 1](#_Toc107626228)

[1.4 Cấu trúc báo cáo tốt nghiệp 1](#_Toc107626229)

[CHƯƠNG 2. Kiến Trúc Hệ Thống 2](#_Toc107626230)

[2.1 Mô hình kiến trúc toàn bộ hệ thống 2](#_Toc107626231)

[2.2 Mô hình hệ thống thông minh nhận diện biển số xe 2](#_Toc107626232)

[CHƯƠNG 3. CƠ SỞ LÝ THUYẾT VÀ THUẬT TOÁN 3](#_Toc107626233)

[3.1 Mô Hình YOLO v3 3](#_Toc107626234)

[3.1.1 Mô hình YOLO 3](#_Toc107626235)

[3.1.2 Mô hình YOLO v3 6](#_Toc107626236)

[3.2 Mô Hình CNN 7](#_Toc107626237)

[3.2.1 Convolution là gì 8](#_Toc107626238)

[3.2.2 Cấu trúc của mạng CNN: 10](#_Toc107626239)

[3.2.3 Trường tiếp nhận cục bộ (local receptive field): 11](#_Toc107626240)

[3.2.4 Trọng số chia sẻ (shared weight and bias): 12](#_Toc107626241)

[3.2.5 Lớp tổng hợp (pooling layer): 12](#_Toc107626242)

[3.3 Các thuật toán tối ưu trong mô hình 13](#_Toc107626243)

[3.3.1 Loss function 13](#_Toc107626244)

[3.3.2 Thuật toán tối ưu Adam 14](#_Toc107626245)

[3.3.3 BatchNormalization 14](#_Toc107626246)

[3.3.4 Drop out 15](#_Toc107626247)

[3.3.5 Data Agumentation 15](#_Toc107626248)

[3.4 Các Độ Đo Đánh giá mô hình 16](#_Toc107626249)

[3.4.1 IOU (Intersection over union) 16](#_Toc107626250)

[3.4.2 Precision và Recall 17](#_Toc107626251)

[3.4.3 AP và mAP 17](#_Toc107626252)

[3.5 Các Thuật Toán Segment 18](#_Toc107626253)

[3.5.1 Làm mượt ảnh bằng Gaussian blur 18](#_Toc107626254)

[3.5.2 Lọc trung vị 19](#_Toc107626255)

[3.5.3 Adaptive Threshole 19](#_Toc107626256)

[3.5.4 Connected component labeling 19](#_Toc107626257)

[CHƯƠNG 4. Xây Dựng Hệ Thống 20](#_Toc107626258)

[4.1 Hệ Thống Nhận Diện Biển Số 20](#_Toc107626259)

[4.1.1 Thu Thập Dữ Liệu 20](#_Toc107626260)

[4.1.2 Gán nhãn dữ liệu nhận dạng vùng chứa biển số 21](#_Toc107626261)

[4.1.3 Huấn Luyện YoloV3 tiny nhận dạng vùng chứa biển số 21](#_Toc107626262)

[4.1.4 Sử dụng thuật toán Segment tách từng ký tự trên biển số 24](#_Toc107626263)

[4.1.5 Gán nhãn và chuẩn bị dữ liệu cho mô hình CNN 26](#_Toc107626264)

[4.1.6 Nhận diện ký tự với mạng CNN 27](#_Toc107626265)

[CHƯƠNG 5. KẾT QUẢ ĐẠT ĐƯỢC 34](#_Toc107626266)

[5.1 Kết luận 34](#_Toc107626267)

[5.2 Hướng phát triển của đồ án trong tương lai 35](#_Toc107626268)

[TÀI LIỆU THAM KHẢO 36](#_Toc107626269)

[PHỤ LỤC 37](#_Toc107626270)

**DANH MỤC HÌNH VẼ**

[Hình 2.1 Đồ thị kiểu bánh 20](#_Toc20580104)

[Hình 2.2 Đồ thị kiểu thanh ngang 20](#_Toc20580105)

[Hình 2.3 Đồ thị kiểu cột đứng 21](#_Toc20580106)

[Hình 2.4 Đồ thị kiểu đường 21](#_Toc20580107)

[Hình 2.5 Đồ thị kiểu diện tích 22](#_Toc20580108)

**DANH MỤC HÌNH VẼ**

[Bảng 1.1 Thống kê các thiết bị và giá thành 8](#_Toc20580109)

# TỔNG QUAN ĐỀ TÀI

## Giới thiệu đề tài

Trong thời đại công nghệ mới, máy móc thay thế con người dần trở thành điều tất yếu để nâng cao năng suất lao động, tăng cường độ chính xác, giảm thiểu rủi ro từ các yếu tố chủ quan của con người và giảm chi phí vận hành, sự thay thế này diễn ra ngay cả trong những hoạt động bình thường nhất. Trong các hoạt động quản lý phương tiện thì việc quản lý trông coi ở các bãi đỗ xe là hết sức quan trọng để bảo vệ tài sản cá nhân của khách hàng nhất là trong giai đoạn các phương tiện công cộng ở thành phố hiện nay chưa thể đáp ứng được nhu cầu đi lại của người dân, phương tiện cá nhân vẫn là thứ không thể thiếu trong cuộc sống sinh hoạt hằng ngày nên nhu cầu số lượng phương tiện cần bãi gửi là rất lớn đặc biệt là ở các trường học, trường đại học và công sở và thường bị ách tắc, chen lấn rất lâu trong các bãi xe để lấy và gửi xe.

Dựa vào những yếu tố đó mà cần tích hợp một hệ thống thông minh tự động nhận diện và lưu lại biển số xe giúp cải thiện tất cả những điều trên. Trong đồ án tốt nghiêp này em sẽ tự tìm hiểu và phát triển một hệ thống như vậy, đi liền với thực tế và giải quyết những vấn đề cấp bách đấy.

## Đối tượng sử dụng

Mô hình được phát triển hướng tới cả những người quản lý bãi gửi xe và cả khách hàng gửi xe tại bãi.

* Đối với khách hàng:
  + Bảo vệ tài sản phưng tiện: Quy trình khép kín, hệ thống tự động lưu trữ hình ảnh xác thực biển số xe vào ra
  + Lưu lại thông tin gửi xe một cách rõ ràng
  + Giảm thời gian chờ đợi xác thực để cho xe vào ra
  + Bảo mật thông tin gửi xe
* Đối với ban quản lý:
  + Quản lý dễ dàng hơn khi số lượng xe lớn
  + Giảm số lượng người vận hành từ đó giảm chi phí
  + Quản lý doanh thu chặt chẽ, chi tiết và chính xác
  + Truy xuất dữ liệu nhanh chóng: Thông in xe, biển số xe
  + Có được những thống kê chính xác: Số lượng xe hiện có trong bãi, số lượng xe ra vào theo ngày, tháng, …

## Mục tiêu đề tài

Xây dựng được một hệ thống hoàn hỉnh nhất theo khả năng để có thể áp dụng vào thực tế, góp phần phát triển và giải quyết các vấn đề, bài toán trong thực tế

## Cấu trúc báo cáo tốt nghiệp

Báo cáo

**Chương 1: Tổng quan**

**Chương 2: Kiến Trúc Hệ Thống**

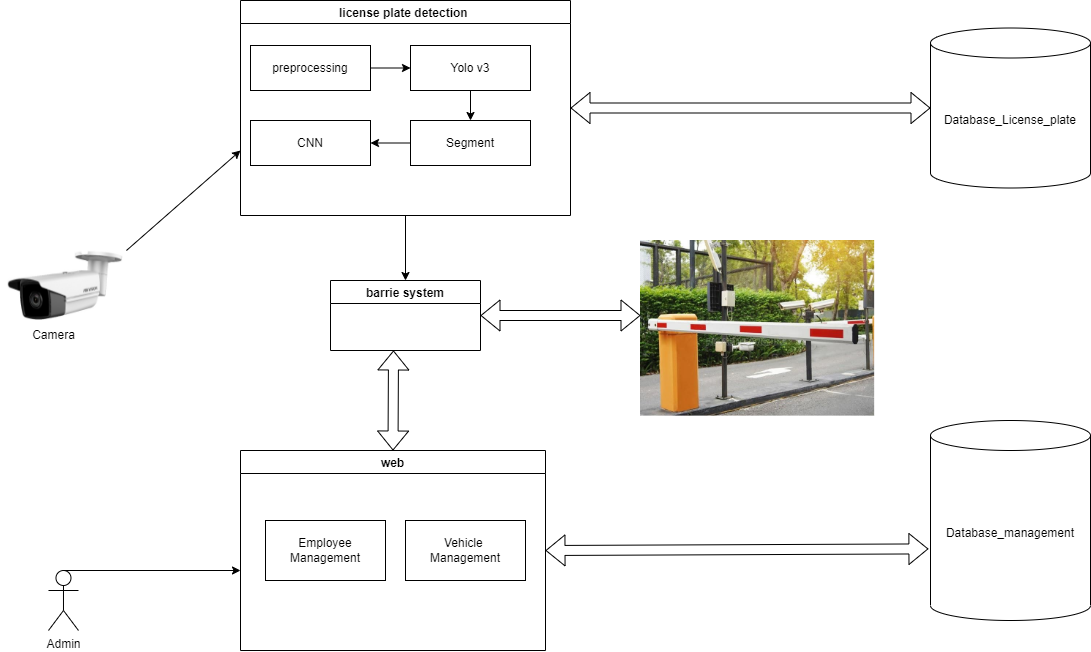
**Chương 3: Cơ Sở Lý Thuyết Và Trình Bày Thuật Toán**

**Chương 4: Xây Dựng Hệ Thống**

**Chương 5: Kết Quả Đạt Được**

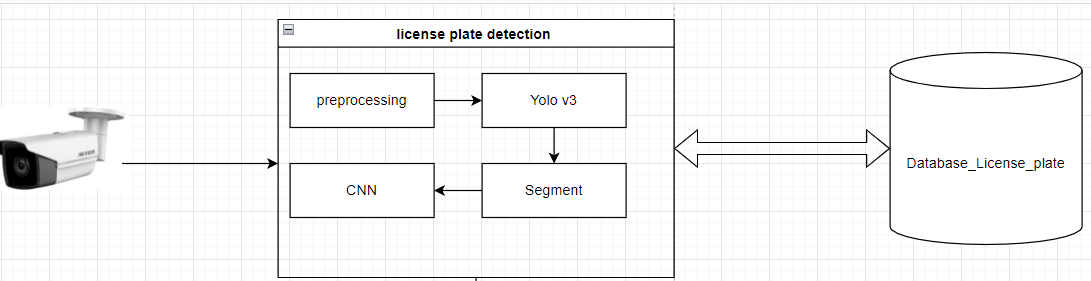
# Kiến Trúc Hệ Thống

## Mô hình kiến trúc toàn bộ hệ thống



* Hệ thống bao gồm:
  + Hệ thống tự động nhận diện biển số xe
  + Hệ thống barrie quản lý vào ra
  + Hệ thống web quản lý
  + Cơ sở dữ liệu lưu trữ bao gồm: Database License plate và Database Management

## Mô hình hệ thống thông minh nhận diện biển số xe



* Hình ảnh biển số xe sẽ được trích xuất từ camera sau đó được đưa vào hệ thống nhận diện
* Hệ thống nhận diện: tiền xử lý ảnh đầu vào sau đó đưa qua mô hình Yolo để xác định vùng chứa biển số xe tiếp đến áp dụng các thuật toán Segment để cắt ra ký tự của biển và cuối cùng đưa qua mô hình Deep learning để phân loại ký tự.
* Kết quả biển số xe thu được sẽ lưu vào database để hệ thống web xử lý.

# CƠ SỞ LÝ THUYẾT VÀ THUẬT TOÁN

## Mô Hình YOLO v3

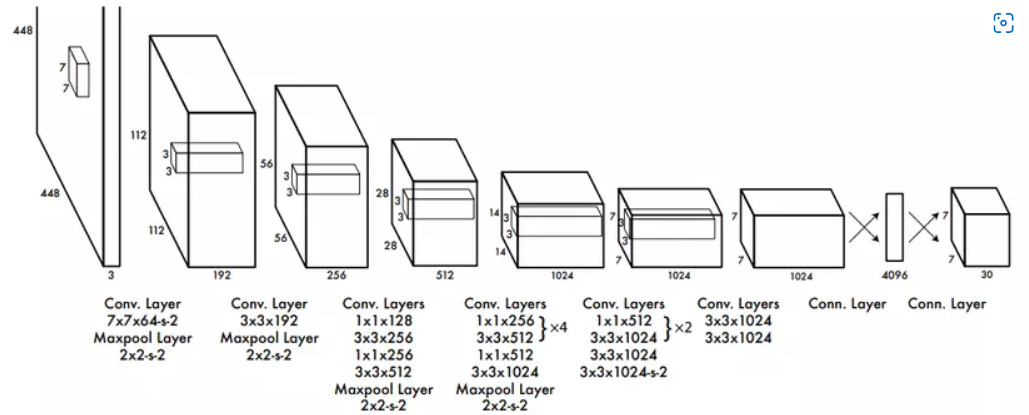
### Mô hình YOLO

Trong vài năm trở lại đây, object detection là một trong những đề tài rất hot của deeplearning bởi khả năng sử dụng cao, dữ liệu dễ chuẩn bị và kết quả ứng dụng thì cực kỳ nhiều. các thuật toán mới của object detection như YOLO có tốc độ khá nhanh và độ chính xác cao nên có thể thực hiện các tác vụ dường như là real time (trong thời gian thực) và các mô hình cũng trở nên nhẹ hơn nên có thể hoạt động trên các thiết bị IoT để tạo nên các thiết bị thông minh.

Mạng YOLO viết tắt của “you only look one”, về độ chính xác thì YOLO không phải là thuật toán tốt nhất nhưng nó là thuật toán nhanh nhất trong lớp các mô hình object detection.

#### Kiến trúc chung của mạng YOLO

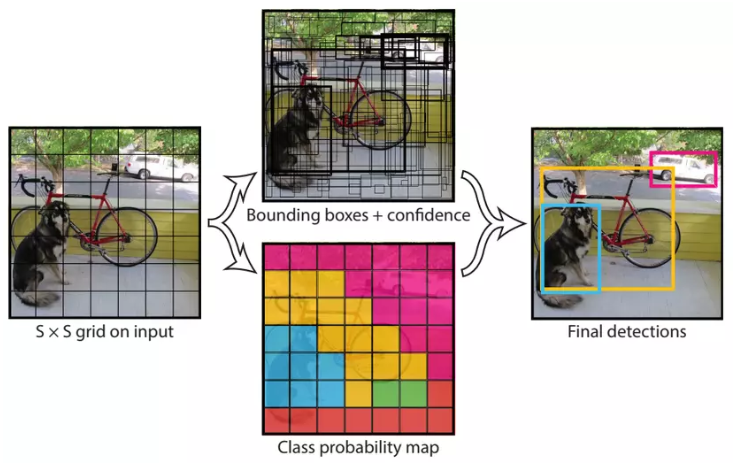
Kiến trúc mạng YOLO bao gồm: base network là các mạng tích chập làm nhiệm vụ trích xuất đặc trưng .Phần phía sau là fully connected layers sẽ dự đoán ra xác suất và tọa độ đối tượng.



ở hình trên thì ảnh đầu vào có kích thước (448\*448\*3) sẽ đi qua 5 lớp tích chập ở phía sau có tác dụng trích xuất đặc trưng. Đầu ra của base network là một feature map có kích thước (7\*7\*1024) sẽ được sử dụng làm đầu vào cho fully connected layer có tác dụng dự đoán nhãn và tọa độ bounding box của vật thể

#### Cách YOLO hoạt động

Đầu vào của mô hình là một ảnh, mô hình sẽ nhận dạng ảnh đó có đối tượng nào hay không, sau đó sẽ xác định tọa độ của đối tượng trong bức ảnh. Ảnh đầu vào được chia thành thành *S*×*S* ô thường thì sẽ là 3×3, 7×7, 9×9, … Tùy theo từng phiên bản YOLO



Với đầu vào là một ảnh, đầu ra của mô hình sẽ là một ma trận 3 chiều có kích thước S×S× (5×N + M) với số lượng tham số mỗi ô là (5×N + M) với N và M lần lượt là số lượng Box và Class mà mỗi ô cần dự đoán. Ví dụ với hình ảnh trên chia thành 7×7 ô, mỗi ô cần dự đoán 2 bounding box và 3 object: con chó, ô tô, xe đạp thì output là 7×7×13, mỗi ô sẽ có 13 tham số, kết quả trả về 7×7×2 = 98 bounding box.

Dự đoán mỗi bounding box gồm 5 thành phần: (x, y, w, h, prediction) với (x, y) là tọa độ tâm của bounding box, (w, h) lần lượt là chiều rộng và chiều cao của bounding box, prediction được định nghĩa bằng Pr(Object) \* IOU(pred, truth).

#### Hàm tính IOU

IOU (Intersection over union) là hàm đánh giá độ chính xác của object detection trên tập dữ liệu cụ thể. IOU được tính bằng:

IOU =

Area of Overlap: là diện tích phần giao nhau giữa predicted bounding box với grounth-truth bounding box

Area of Union: là diện tích phần hợp giữa predicted bounding box và grounth-truth bounding box.

#### Hàm Loss

YOLO sử dụng hàm độ lỗi bình phương giữa dự đoán và nhãn để tính độ lỗi cho mô hình. Cụ thể, độ lỗi tổng sẽ bằng tổng của 3 độ lỗi con:

* Độ lỗi của việc dự đoán nhãn của Object – Classifycation Loss

Mô hình chỉ tính Classifycation Loss cho những ô được gán nhãn là có Object:

*Lclassification*

Trong đó:

= 1 nếu ô vuống đang xét có object và ngược lại

: là xác suất có điều kiện của lớp c tại ô vuông tương ứng mà mố hình dự đoán

* Độ lỗi của việc dự đoán tọa độ cũng như chiều dài, rộng của bounding box đó – Localization loss

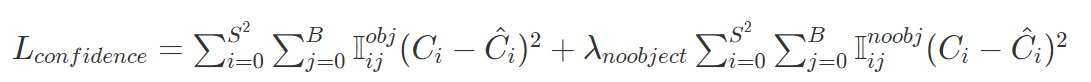
Localization loss dùng để tính giá trị lỗi cho boundary box được dự đoán bao gồm offset x, y và chiều dài, rộng so với nhãn chính xác. Ta không tính toán trực tiếp giá trị lỗi này trên kích thước của ảnh mà cần chuẩn dưới kính thước ảnh về đoạn [0-1] đối với tọa độ điểm tâm, và không dữ đoán trực tiếp điểm tâm mà phải dự đoán giá trị lệch offset x, y so với ô vuông tương ứng. Việc chuẩn hóa kích thước ảnh và dự đoán offset làm cho mô hình nhanh hội tụ hơn so với việc dự đoán giá trị mặc định.

Độ lỗi localization loss được tính bằng tổng đỗ lỗi bình phương của offsetx, offsety và chiều dài, rộng trên tất cả các ô vuông có chứa object. Tại mỗi ô vuông đúng, ta chọn 1 boundary box có IOU (Intersect over union) tốt nhất, rồi sau đó tính độ lỗi theo các boundary box này.

Localization loss là độ lỗi quan trọng nhất trong 3 loại độ lỗi trên. Do đó, ta cần đặt trọng số cao hơn cho độ lỗi này.

* Độ lỗi của ô vuông có chưa object nào hay không – Confidence Loss

Confidence loss thể hiện độ lỗi giữa dự đoán boundary box đó chứa object so với nhãn thực tế tại ô vuông đó. Độ lỗi này tính nên cả những ô vuông chứa object và không chứa object.

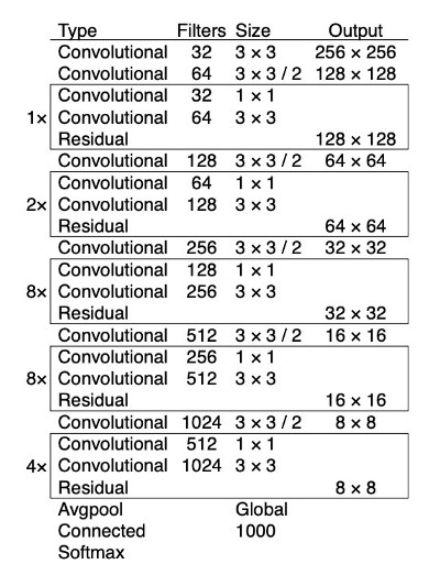


Độ lỗi này là độ lỗi bình phường của dự đoán boundary đó chứa object với nhãn thực tế của ô vuông tại vị trí tương ứng, độ lỗi tại ô vuông mà nhãn chứa object quan trọng hơn là độ lỗi tại ô vuông không chứa object, do đó cần sử dụng hệ số lambda để cân bằng điều này.

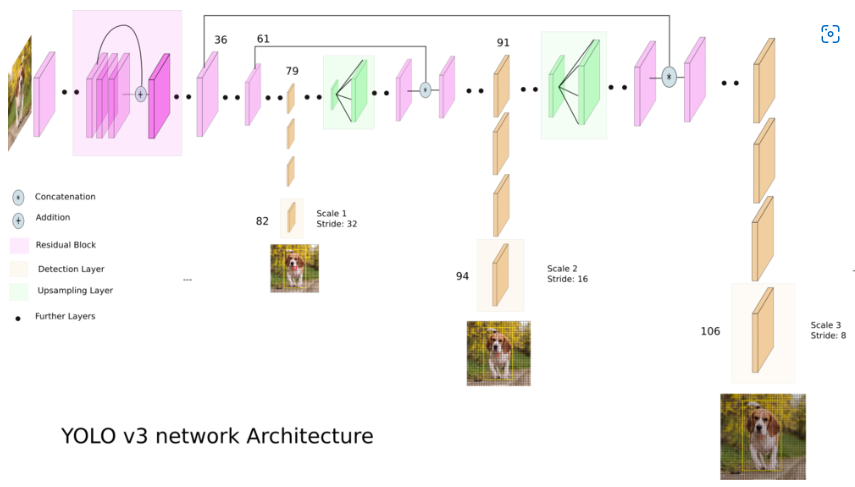
Tổng lỗi của chúng ta sẽ bằng tổng của 3 loại độ lỗi trên:

### Mô hình YOLO v3

Mô hình YOLOv3 sử dụng darknet53 làm backbone (có 53 lớp tích chập)



Bản thân YOLOv3 được thiết kế thêm 53 lớp chập nữa, tổng cộng có 106 lớp chập và mô hình đã được đưa vào các lớp như: residual block, upsampling, skip connection.



* Để nhận dạng được những đối tượng có kích thước nhỏ trong ảnh, YOLOv3 nhận dạng 3 lần trên một bức ảnh với kích thước khác nhau.

Giả sử ta có bức ảnh đầu vào có kích thước: 416×416:

* Tại lớp chập thứ 82: ảnh được chia thành các ô lưới có kích thước 13×13. Tại đây, các ô lưới sẽ có trách nhiệm tìm các vật thể có kích thước lớn trong hình.
* Tại lớp chập thứ 94: Bức ảnh được chia thành các ô lưới với kích thước 26×26. Và có trách nhiệm tìm các vật thể có kích thước trung bình.
* Tương tự, tại lớp chập 106, bức ảnh được chia thành các ô lưới với kích thước 52×52 và có trách nhiệm tìm các vật thể có kích thước bé.

Ảnh sau khi được chia thành các ô lưới, thì được gọi là Downsample images (ảnh giảm mẫu)

* Ở YOLO các phiên bản trước, kết quả cuối cùng của model là S các tensor còn sang YOLOv3 thì kết quả đầu ra lại được thay đổi thành S×S×255, trong đó:
* S lần lượt được gán các giá trị 13, 26, 52
* Giá trị 255 = B×(5+C) với B = 3 là số lượng Bounding box, và C = 80 là số lớp vật thể.
* Mỗi bounding box chứa 5 giá trị bao gồm: x, y, w, h và p (xác suất ô lưới có chứa vật thể)
* Để tính ra kết quả cuối cùng thì mỗi bounding box sẽ nhân giá trị P lần lượt với P1, P2, …Pc (là xác suất ô lưới đó chứa các vật thể cụ thể) và lựa chọn ra kết quả cao nhất.
* Với mỗi bức ảnh được nhận dạng, thì YOLOv3 sẽ sinh ra (13×13 + 26×26 + 52×52) ×3 = 10,647 bounding box. Từ đó sẽ tính xác suất và xuất ra kết quả là những vật thể có xác suất cao nhất.
* Anchor Box:

Anchor box thực ra là các bounding box nhưng được tạo sẵn ( chứ không phải kết quả của quá trình nhận dạng – prediction).

Với mỗi ô sẽ có 3 Anchor box khác nhau với các kích thước:

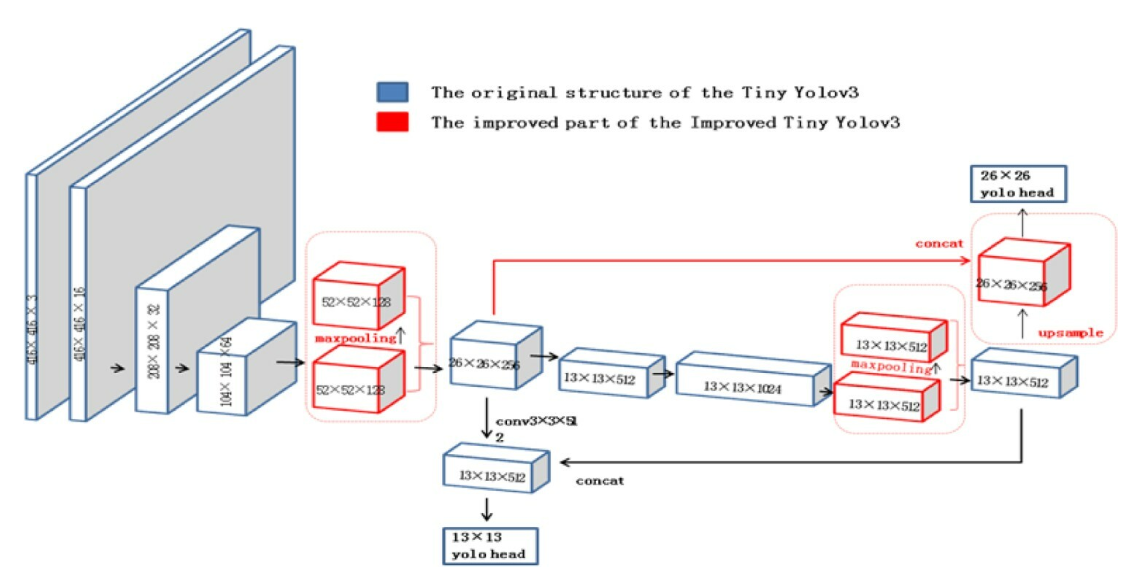
* Kích thước 13×13: (116×90), (156×198), (373×326)
* Kích thước 26×26: (30×61), (62×45), (59×119)
* Kích thước 52×52: (10×13), (16×30), (33×23)

Trong quá trình đào tạo sẽ kết hợp với thuật toán Kmean cluster (thuật toán phân cụm). Cùng với ground truth để tính ra sai sót giữa ground truth và anchor box bằng cách điều chỉnh các giá trị x, y, w, h, từ đó học được các đặc điểm của vật thể.

### Mô hình YOLOv3 Tiny

YOLOv3 Tiny là một phiên bản rút gọn hơn rất nhiều của YOLOv3, tuy được rút gọn về kiến trúc mô hình nhưng nó vẫn được áp dụng các kỹ thuật tương tự như YOLOv3 với các lớp upsampling giúp mô hình có thể nhận diện được các vật thể có kích thước nhỏ và khác nhau.

Do có kích thước nhỏ hơn để phục vụ cho việc hoạt động realtime cho nên YOLOv3 tiny có độ chính xác thấp hơn YOLOv3 rất nhiều do vậy nó phù hợp với các bài toán nhận diện vật thể tương đối rõ ràng. Cụ thể là trong bài toán nhận diện biển số xe này, YOLOv3 tiny hoạt động rất tốt với độ chính xác cao



Kiến trúc mô hình YOLOv3 tiny

Phần màu xanh là kiến trúc gốc của YOLOv3 tiny, phần màu đỏ là các tầng thêm để cải tiến để cải thiện độ chính xác

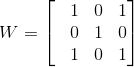
## Mô Hình CNN

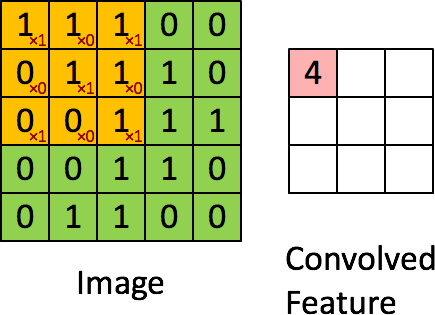
Convolutional Neural Network (CNNs – Mạng nơ-ron tích chập) là một trong những mô hình Deep Learning tiên tiến. Nó giúp cho chúng ta xây dựng được những hệ thống thông minh với độ chính xác cao như hiện nay. Như hệ thống xử lý ảnh lớn như Facebook, Google hay Amazon đã đưa vào sản phẩm của mình những chức năng thông minh như nhận diện khuôn mặt người dùng, phát triển xe hơi tự lái hay drone giao hàng tự động.

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), em sẽ khái quát về thuật toán này.

### Convolution là gì

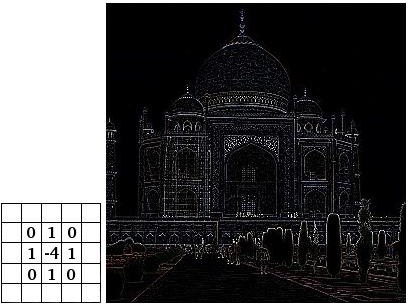
Ta định nghĩa kernel là một ma trận vuông kích thước k\*k trong đó k là số lẻ. k có thể bằng 1, 3, 5, 7, 9,… Ví dụ kernel kích thước 3\*3.

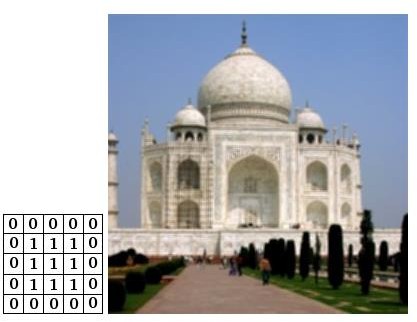


Kí hiệu phép tính convolution (⊗), kí hiệu Y = X ⊗ W.

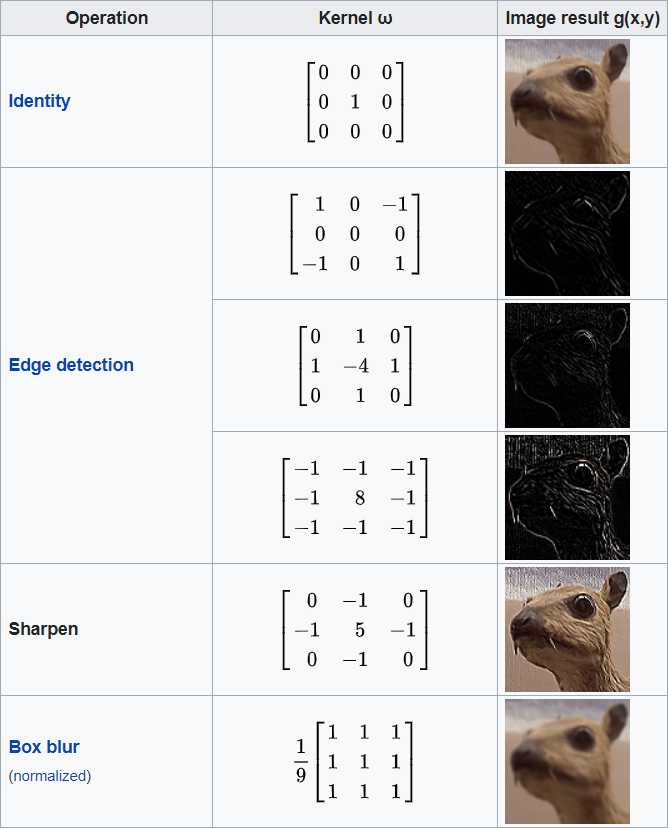
Trong hình ảnh ví dụ trên, ma trận bên trái là một hình ảnh trắng đen được số hóa. Ma trận có kích thước 5×5 và mỗi điểm ảnh có giá trị 1 hoặc 0 là giao điểm của dòng và cột.

Convolution hay tích chập là nhân từng phần tử 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.





**Ý nghĩa của phép tính convolution:** Mục đích của phép tính convolution trên ảnh là làm mờ, làm nét ảnh, xác định các đường… Mỗi kernel khác nhau thì sẽ phép tính convolution sẽ có ý nghĩa khác nhau. Ví dụ:



### Cấu trúc của mạng CNN:

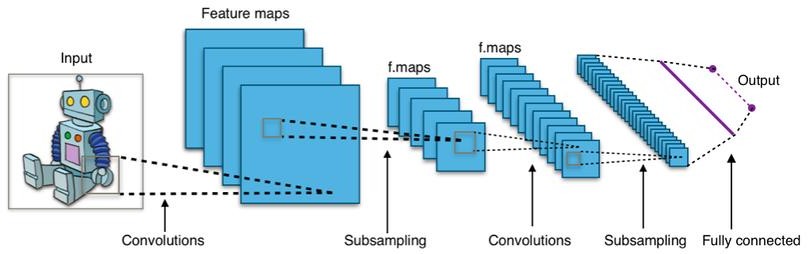
Mạng CNN là một tập hợp các lớp Convolution chồng lên nhau và sử dụng các hàm nonlinear activation như ReLU hoặc tanh để kích hoạt các trọng số trong các node. Mỗi một lớp sau khi thông qua các hàm kích hoạt sẽ tạo ra các thông tin trừu tượng hơn cho các lớp tiếp theo.

Sau những lớp Convolution sẽ là 1 mạng nơ-ron nhận đầu vào là vector (vector này thu được nhờ việc duỗi thẳng 1 ma trận tính toán được sau những phép tích chập).

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 (Fully Connected Layer) 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).

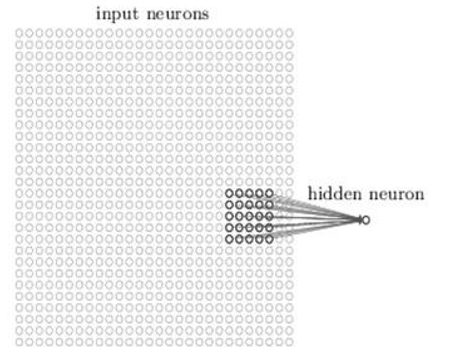
+ Trọng số chia sẻ (shared weights).

+ Tổng hợp (pooling).

### 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.



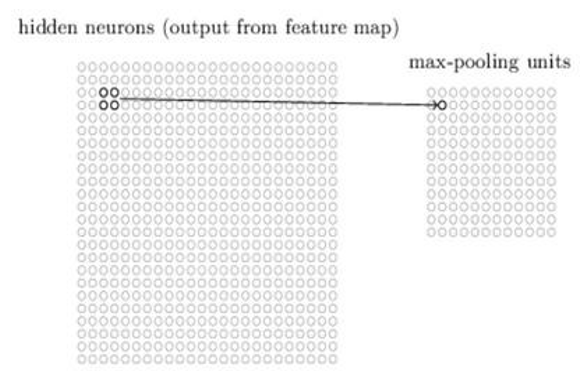
Local receptive field thích hợp cho việc phân tách dữ liệu ảnh, giúp chọn ra những vùng ảnh có giá trị nhất cho việc đánh giá phân lớp.

### Trọng số chia sẻ (shared weight and bias):

Đầu tiên, các trọng số cho mỗi filter (kernel) phải giống nhau. Tất cả các nơ- ron trong lớp ẩn đầu sẽ phát hiện chính xác feature tương tự chỉ ở các vị trí khác nhau trong hình ảnh đầu vào. Chúng ta gọi việc map từ input layer sang hidden layer là một feature map. 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.

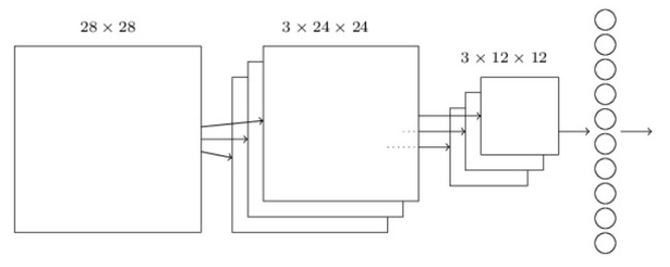
### Lớp tổng hợp (pooling layer):

Lớp pooling thường được sử dụng ngay sau lớp convulation 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.



Max Pooling là cách hỏi xem trong các đặc trưng này thì đặc trưng nào là đặc trưng nhất.

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.



Và đây chính là đầu vào cho 1 mạng Nơ-ron. Mục địch thêm lớp tích chập phía trước là để mô hình học được thêm nhiều đặc trưng của ảnh hơn trước khi kết nối với mạng nơ-ron nhằm tăng độ chính xác của mô hình.

## Các thuật toán tối ưu trong mô hình

### Loss function

Loss function trả về một số thực không âm thể hiện sự chênh lệch giữa hai đại lượng: ŷ: label được dự đoán và y: label thực tế. Loss function giống như một hình thức để bắt model đóng phạt mỗi lần nó dự đoán sai, và số mức phạt tỉ lệ thuận với độ trầm trọng của sai sót. Trong mọi bài toán supervised learning, mục tiêu của ta luôn bao gồm giảm thiểu tổng mức phạt phải đóng. Trong trường hợp lý tưởng ŷ =y, loss function sẽ trả về giá trị cực tiểu là 0.

Trong bài toán phân loại ký tự này, em sử dụng hàm loss là category CrossEntropy:

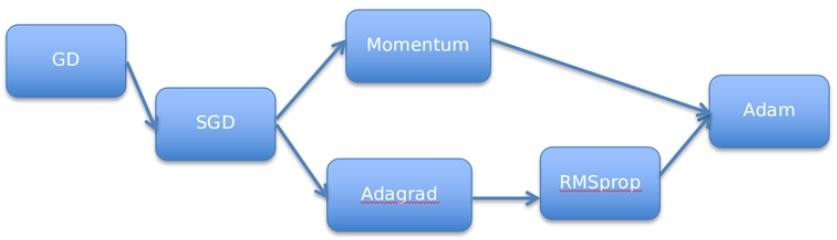
Loss = -

Trong đó: là giá trị thật của nhãn

là giá trị dự đoán của model

Hàm loss cross entropy để đánh giá sự sai lệnh giữa hai phân phối cụ thể ở đây là nhãn lớp thật và nhãn lớp dự đoán của model.

### Thuật toán tối ưu Adam



Hầu hết những mô hình học máy nói chung và học sâu nói riêng đều cần 1 thuật toán để tối ưu nhằm cực tiểu hóa hàm mục tiêu, nhằm tìm ra được bộ tham số thích hợp và tối ưu nhất cho mô hình.

Một số thuật toán tối ưu phổ biến như Gradient Descent, Momentum,…Trong số những thuật toán tối ưu thì Adam là 1 trong những thuật toán hay được sử dụng nhất vì nó khắc phục được nhược điểm của những thuật toán tối ưu RMSprop, AdaGrad. Nếu giải thích theo hiện tượng vật lí thì Momentum giống như 1 quả cầu lao xuống dốc, còn Adam như 1 quả cầu rất nặng có ma sát, vì vậy nó dễ dàng vượt qua local minimum tới global minimum và khi tới global minimum nó không mất nhiều thời gian dao động qua lại quanh đích vì nó có ma sát nên dễ dừng lại hơn.

Không giống như các thuật toán trước đó thì learning rate hầu như giống nhau trong quá trình training (learning rate là hằng số), Adagrad coi learning rate là 1 tham số. Tức là Adagrad sẽ cho learning rate biến thiên sau mỗi thời điểm t.

### BatchNormalization

Batch Normalization là một phương pháp hiệu quả khi training một mô hình mạng nơ ron. Mục tiêu của phương pháp này chính là việc muốn chuẩn hóa các feature (đầu ra của mỗi layer sau khi đi qua các activation) về trạng thái zero-mean với độ lệch chuẩn 1. non-zero mean có ảnh hưởng xấu đến việc training mô hình:

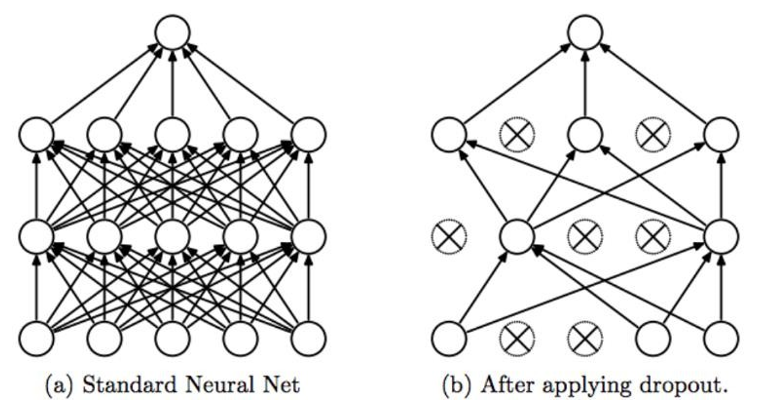
* Thứ nhát có thể hiểu rằng Non zero mean là hiện tượng dữ liệu không phân bố quanh giá trị 0, mà dữ liệu có phần nhiều giá trị lớn hơn không, hoặc nhỏ hơn không. Kết hợp với vấn đề high variance khiến dữ liệu trở nên có nhiều thành phần rất lớn hoặc rất nhỏ. Vấn đề này rất phổ biến khi training các mạng nơ ron với số layer sâu. Việc feature không phân phối trong những khoảng ổn định (giá trị to nhỏ thất thường) sẽ có ảnh hưởng đến quá trình tối ưu của mạng. Việc tối ưu một mạng nơ ron sẽ cần phải sử dụng đến tính toán đạo hàm, giá trị trọng số x ảnh hưởng trực tiếp đến giá trị của đạo hàm. Do đó nếu x mang các giá trị thay đổi không ổn định dẫn đến đạo hàm sẽ có thể bị quá lớn, hoặc quá nhỏ dẫn đến việc learning model không được ổn định. Và điều đó cũng đồng nghĩa với việc chúng ta có thể sử dụng các learning rate cao hơn trong quá trình training khi sử dụng Batch Normalization.
* Batch normalization có thể giúp chúng ta tránh được hiện tượng giá trị của x rơi vào khoảng bão hòa sau khi đi qua các hàm kích hoạt phi tuyển. Vậy nên nó đảm bảo rằng không có sự kích hoạt nào bị vượt quá cao hoặc quá thấp. Điều này giúp cho các weights mà khi không dùng BN có thể sẽ không bao giờ được học thì nay lại được học bình thường. Điều này giúp chúng ta làm giảm đi sự phụ thuộc vào giá trị khởi tạo của các tham số.
* Batch Normalization còn có vai trò như một dạng của **regularization** giúp cho việc giảm thiểu overfiting. Sử dụng batch normalization sẽ không cần phải sử dụng quá nhiều dropout và điều này rất có ý nghĩa vì sẽ không bị mất quá nhiều thông tin khi dropout weigths của mạng. Tuy nhiên vẫn nên sử dụng kết hợp cả hai kĩ thuật này

### Drop out

Trong mạng neural network, kỹ thuật dropout là việc chúng ta sẽ bỏ qua một vài unit trong suốt quá trình train trong mô hình, những unit bị bỏ qua được lựa chọn ngẫu nhiên, tức là unit đó sẽ không tham gia và đóng góp vào quá trình huấn luyện (lan truyền tiến và lan truyền ngược).

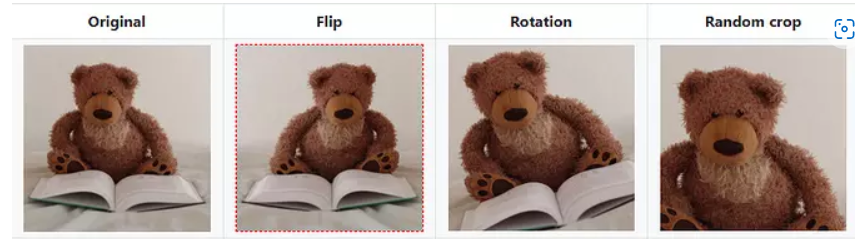
Tại mỗi giai đoạn huấn luyện, mỗi node có xác suất bị bỏ qua là 1-p và xác suất được chọn là p.

Dropout có thể chống được over-fitting, Dropout là một kỹ thuật khác với L1 và L2, là một cách tiếp cận khác để regularization trong mạng neural networks.



### Data Agumentation

Có một kĩ thuật nữa giải quyết vấn đề có ít dữ liệu cho việc training model, đó là data augmentation. Augmentation là kĩ thuật tạo ra dữ liệu training từ dữ liệu mà ta đang có. Có một số kĩ thuật phổ biến như Flip, Rotation, Scale, Crop, Translation, Color shift, Noise addition, Information loss, Constrast change, …



## Các Độ Đo Đánh giá mô hình

Đối với mỗi model Object detection sau khi đào tạo, cần có những thang điểm để đánh giá sự chính xác của nó. Ta đánh giá một model dựa trên: loss function, IOU avg, mAP, đánh giá trực quan, và nhiều bài báo cũng như các trang web lớn thường sử dụng mAP như là thước đo chính.

### IOU (Intersection over union)

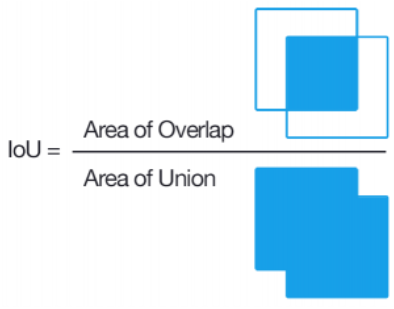
Intersection over Union là chỉ số đánh giá được sử dụng để đo độ chính xác của phát hiện đối tượng trên tập dữ liệu cụ thể. Chỉ số này thường được gặp trong các Object Detection Challenge. IOU thường được đánh giá hiệu năng của các bộ phát hiện đối tượng như HOG + Linear SVM và mạng nơ ron tích chập (R-CNN, FastR-CNN, YOLO, …).

Để áp dụng được IoU để đánh giá cần:

– Đường bao thực (ground-truth bounding box): là đường bao mà chúng ta gán cho vật thể bằng LabelImage Tool

– Đường bao dự đoán (predicted bouding box): là đường bao chúng ta sử dụng file Weights sau khi đào tạo để nhận dạng

Tỷ lệ IoU (Intersection over union) là tỉ lệ giữa đo lường mức độ giao nhau giữa hai đường bao (thường là đường bao dự đoán và đường bao thực) để nhằm xác định hai khung hình có bị đè chồng lên nhau không.  Tỷ lệ này được tính dựa trên phần diện tích giao nhau gữa 2 đường bao với phần tổng diện tích giao nhau và không giao nhau giữa chúng.



Các tiêu chí được dùng để đánh giá:

– Đối tượng được nhận dạng đúng với tỉ lệ IOU> 0.5 (True positive: TP)

– Đối tượng được nhận dạng sai với tỉ lệ IOU < 0.5 (False positive: FP)

– Đối tượng không được nhận dạng (False negative: FN)

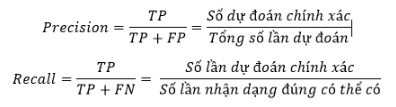
### Precision và Recall

#### Precision

Precision sẽ cho biết rằng trong những cái model dự đoán là Positive thì có bao nhiêu % là Positive thật. Ví dụ: 10 ca bệnh được đưa vào khám, trong đó có 8 ca bệnh được model dự đoán là ung thư. Cơ mà thực tế chỉ có 5 ông là ung thư thật còn lại 3 ông là model dự đoán sai. Precision đại diện cho độ tin cậy

#### Recall

Recall sẽ cho biết model có thể tóm đúng được bao nhiêu Positive trong dữ liệu được cho. Ví dụ: 10 ca bệnh vào khám, trong đó có 5 ca bệnh là ung thư thật. Giả sử model nhận diện được 3 ông là ung thư, 7 ông còn lại là bình thường.



### AP và mAP

Từ precision và recall đã được định nghĩa ở trên chúng ta cũng có thể đánh giá mô hình dựa trên việc thay đổi một ngưỡng và quan sát giá trị của Precision và Recall. Khái niệm Area Under the Curve (AUC) cũng được định nghĩa tương tự. Với Precision-Recall Curve, AUC còn có một tên khác là Average precision (AP).

Giả sử có 𝑁 ngưỡng để tính precision và recall, với mỗi ngưỡng cho một cặp giá trị precision, recall là 𝑃𝑛 =1, 2, …, 𝑁. Precision-Recall curve được vẽ bằng cách vẽ từng điểm có toạ độ (𝑃𝑛) trên trục toạ độ và nối chúng với nhau. AP được xác định bằng: 𝐴𝑃=∑(𝑅𝑛−𝑅𝑛−1) × 𝑃𝑛.

Như vậy với mỗi class trong bài toán ta sẽ có một giá trị AP. mAP là Mean Average Precision là trung bình cộng giá trị AP của các class khác nhau

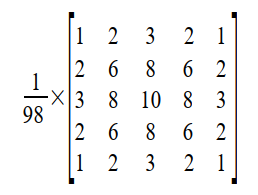
## Các Thuật Toán Segment

### Làm mượt ảnh bằng Gaussian blur

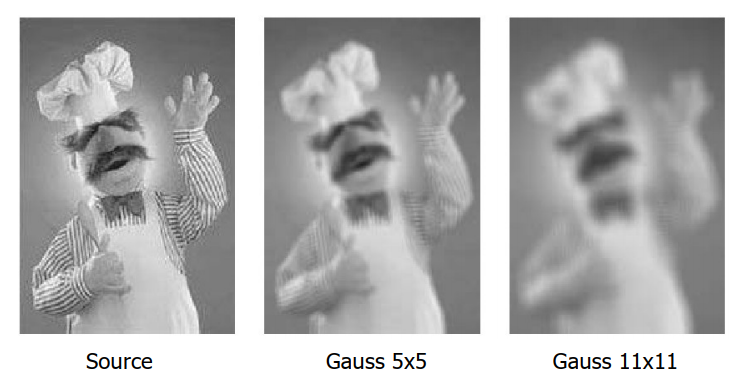
Nguyên tắc chung của các phương pháp lọc là cho ma trận ảnh nhân với một ma trận lọc (Kernel). Ma trận lọc lọc (Kernel) còn có thể được gọi là cửa số chập (trong phép nhân chập), cửa sổ lọc, mặt nạ, …

Việc nhân ảnh với ma trận lọc giống như việc trượt ma trận lọc theo hàng trên ảnh và nhân với từng vùng của ảnh, cộng các kết quả lại tạo thành kết quả của điểm ảnh trung tâm.

Đối với làm mượt ảnh bằng Gaussian Blur, ta sử dụng ma trận lọc có các giá trị tuân theo phân phối chuẩn



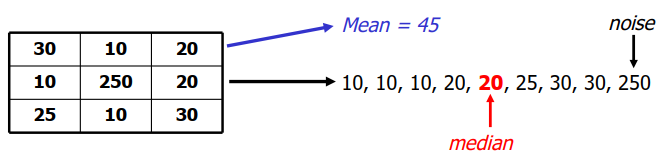
Bộ lọc gauss làm trơn tốt hơn và kết quả ít nhiễu hơn so với bộ lọc trung bình



### Lọc trung vị

Bộ lọc trung vị loại bỏ nhiễu tốt hơn so với bộ lọc trung bình hay gauss đặc biệt là nhiễu có dạng muối tiêu. Là bộ lọc phi tuyến, không thể được cài đặt bởi phép nhân chập.

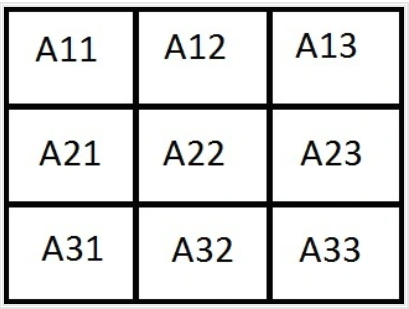
Bộ lọc thực hiện thay giá trị của điểm ảnh bởi giá trị trung vị của các điểm ảnh lân cận N×N



### Adaptive Threshole

Thuật toán simple thresholding hoạt động khá tốt. Tuy nhiên có 1 nhược điểm là giá trị ngưỡng bị/được gán toàn cục. Thực tế khi chụp, hình ảnh nhận được thường bị ảnh hưởng của nhiễu, ví dụ như là bị phơi sáng, bị đèn flask, … Một trong những cách được sử dụng để giải quyết vấn đề trên là chia nhỏ bức ảnh thành những vùng nhỏ (region), và đặt giá trị ngưỡng khác nhau trên những vùng nhỏ đó, thuật toán adaptive thresholding ra đời.

Ta sẽ lấy ngưỡng nhị phân trên một vùng cục bộ là một cửa sổ có kích thước *mn*. Ngưỡng cần xác định đối với tâm của cửa sổ đó sẽ là trung bình cộng các giá trị của các lân cận của nó hoặc nhân giá trị xung quanh điểm cần xét với trọng số gauss rồi tính trung bình của nó.



Giả sử ta lấy ngưỡng động cho điểm A22 và kích thước cửa sổ la 3x3. Khí đó giá trị T sẽ là T = 1/8(A11 + A12+ A13 + A21 + A23 + A31 + A32 + A33). Nếu A22 > T, đặt A22 về 255, nếu A22 < T, đặt A22 về 0.

### Connected component labeling

Thuật toán Connected component labeling hay còn gọi là thuật toán gán nhãn các thành phần liên thông trong ảnh.

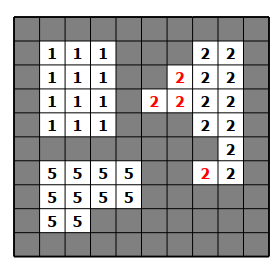
Thành phần liên thông là tập hợp các điểm lân cận thuộc vào một vùng

Thuật toán sẽ duyệt lần lượt từng pixel trong ảnh để gán nhãn cho từng vùng, tất cả các pixel thuộc một vùng thì phải có cùng nhãn.

Hai pixel có cùng nhãn khi chúng là hàng xóm ở trong cận trên + trái hoặc cận dưới + phải trong các lần duyệt. Đôi khi chúng ta cần nhiều lần lặp để hoàn thành việc gán nhãn, khi đó ta lặp đến khi không có sự thay đổi nào.







# Xây Dựng Hệ Thống

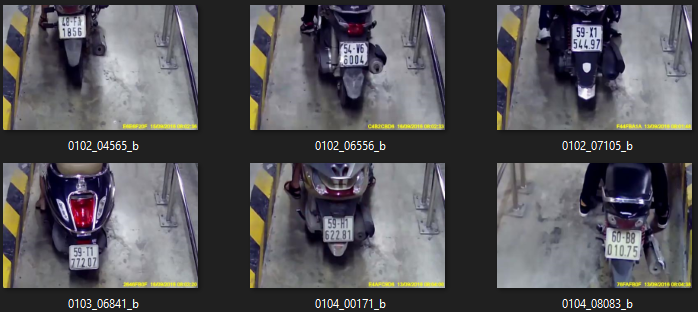
## Hệ Thống Nhận Diện Biển Số

### Thu Thập Dữ Liệu

Với mục tiêu xây dựng một hệ thống nhận diện biển số xe của cả ô tô và xe máy trong bãi gửi xe trong môi trường gần với thực tế nhất tại Việt Nam, tuy nhiên do tập dữ liệu công khai còn hạn chế nên em chỉ thu thập được tập dữ liệu hình ảnh gửi xe của phương tiện xe máy. Chính vì vậy em quyết định làm mô hình nhận diện biển số xe máy mà chủ yếu dạng hình vuông.

Tập dữ liệu được thu thập từ nguồn công khai trên internet của bãi gửi xe GreenParking gồm 1900 ảnh biển số xe vào bãi.

Nguồn dataset: “[Bien\_so\_xe\_full (mediafire.com)](https://www.mediafire.com/file/qkt7cv0itaeh5xz/Bien_so_xe_full.zip/file)”



### Gán nhãn dữ liệu nhận dạng vùng chứa biển số

Dữ liệu thu thập được dưới dạng hình ảnh file .jpg chưa được gán nhãn vùng chứa biển số xe. Để có thể đưa dữ liệu vào huấn luyện thì cần gán nhãn cho các dữ liệu này.

Cụ thể đối với mỗi hình ảnh trích xuất từ camera cần gãn nhãn vùng chứa biển số xe bao gồm 4 tọa độ của vùng đó. Để gán nhãn cho dữ liệu em dùng một cụ có tên là “Labellimage”. Sau khi gán nhãn công cụ sẽ trả về file text bao gồm các thông tin nhãn và tọa độ bốn góc để làm đầu vào huấn luyện mô hình Yolo



File text nhãn input để đưa vào huấn luyện mô hình yolo có dạng:

<Object class> <x> <y> <width> <height>

<Object class>: số nguyên từ 0 đại diện cho lass

<x>: tọa độ x của trung tâm bounding box

<y>: tọa độ y của trung tâm bounding box

<width>: chiều rộng của bounding box

<height>: chiều cao của bounding box

VD: “0 0.509534 0.466997 0.150424 0.207921”

### Huấn Luyện YoloV3 tiny nhận dạng vùng chứa biển số

#### Lựa chọn mô hình

Với các bài toán nhận diện vật thể thì Yolo là mô hình phổ biến và chủ yếu được dùng hiện nay. Qua thời gian đã có một số phiên bản yolo được cải tiến và phát triển giúp cải thiện độ chính xác cũng như thời gian xử lý của mô hình, từ mô hình đầu tiên YoloV1 đến YoloV5 đã có rất nhiều sự khác biệt và mô hình sau đạt được kết quả xử lý hơn mô hình trước rất nhiều. Tuy nhiên dựa vào từng bài toán mà chúng ta sẽ chon ra mô hình phù hợp nhất.

Đối với bài toán nhận diện biển số xe trích xuất từ hình ảnh camera sẽ không cần một mô hình có tốc độ xử lý quá cao trong thời gian thực đồng thời vùng chứa biển số xe trong ảnh tương đối rõ ràng và không quá phức tạp đối với nền xung quanh. Chính vì vậy sau khi tham khảo thì mô hình **YoloV3 tiny** là phù hợp nhất với bài toán nhận diện vùng chứa biển số xe này.

Do YoloV3 tiny là mô hình rút gọn của YoloV3 chính vì thế nó có tốc độ xử lý nhanh hơn rất nhiều.

#### Huấn luyện mô hình

Dữ liệu đã gán nhãn gồm 1760 hình ảnh biển số xe và file text nhãn em tiến hành huấn luyện mô hình.

Bộ dữ liệu được phân chia thành tập **train** và **val** với tỉ lệ 8:2

Sau khi điều chỉnh các thông số cần thiết trong file config của mô hình để phù hợp với tập dữ liệu:

Batch = 1 (số lượng data mỗi lần load)

Subdivision = 1 (số lượng mini batch)

LearningRate = 0.001

LearningRate decay = 0.0005

Momentum = 0.9

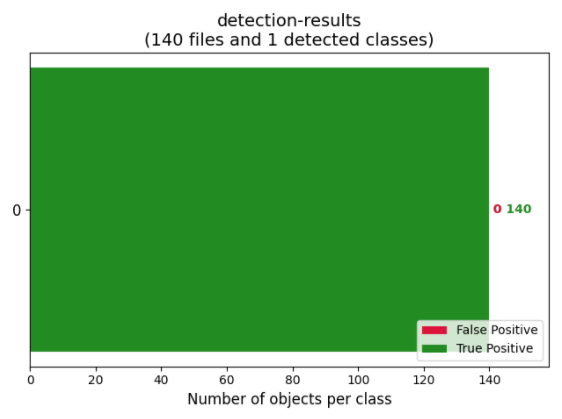
Tiến hành huấn luyện dữ liệu trên Google Colab sau 8000 vòng thu được loss model = 0.4 và khá ổn định nên em dừng quá trình huấn luyện và thu được file weight

#### Đánh giá mô hình

Đối với mỗi model sau khi huấn luyện , cần có những thang điểm để đánh giá sự chính xác của nó. Đối với các mô hình object detection, sự đánh giá thường dựa trên loss function, IOU Avg, mAP

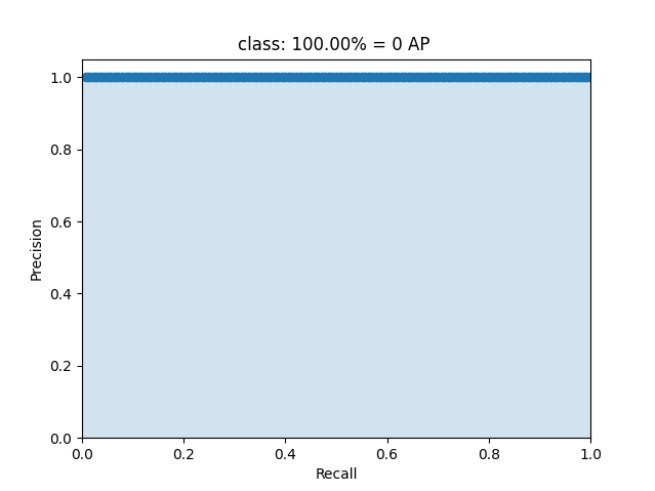
Sau khi thu được file weight tiến hành kiểm tra với tập test gồm 140 ảnh đã được gán nhãn và đánh giá

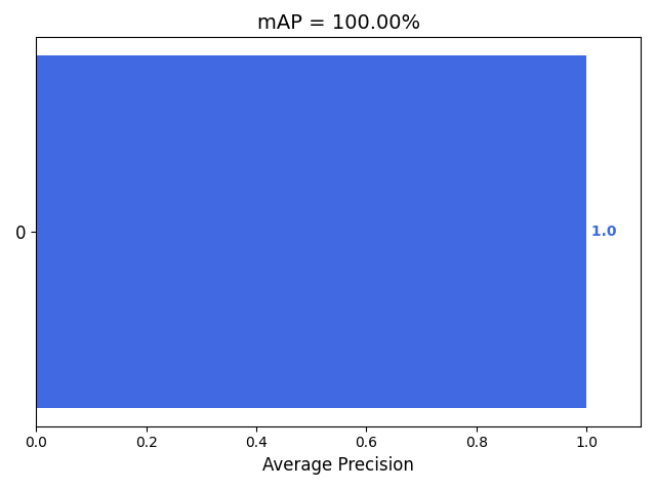
* Tỉ lệ đánh giá đúng và sai (True Possitive : False positive)



Với ngưỡng IOU bằng 0.5, đối tượng nhận dạng đúng với IOU > 0.5 (TP) và đối tượng nhận dạng sai với ngưỡng IOU < 0.5 (FP) mô hình cho kết quả rất tốt với độ chính xác 100% TP.

* Tính AP và mAP





Có thể thấy mô hình đánh giá rất chính xác với chỉ một lớp duy nhất và dữ liệu rõ ràng, biển số xe trong ảnh có màu trắng nổi bật với nền xung quanh ko phức tạp thì mô hình đơn giản và nhanh như YoloV3 Tiny là phù hợp với bài toán

### Sử dụng thuật toán Segment tách từng ký tự trên biển số

#### Lấy ngưỡng đưa ảnh về nhị phân

Đầu tiên ta sẽ chuyển ảnh từ không gian màu BGR sang HSV. Không gian màu HSV có ba giá trị H(Hue): vùng chứa màu sắc, S(Saturation): độ bão hòa và V(Value): độ sáng. Vì vùng chứa biển số xe có các vùng sáng tối rất rõ ràng là khung nền màu trắng và các ký tự có màu đen nên em lấy theo giá trị V để có thể lấy ngưỡng cho kết quả tốt nhất.

Tiếp theo resize ảnh lên kích thước lớn hơn sau đó làm mượt ảnh bằng GaussianBlur. Bước này sẽ có thể làm các vùng sát nhau được phân định rõ ràng hơn khi tiến hành lấy ngưỡng.

Cuối cùng lấy ngưỡng động bằng hàm **AdaptiveThreshold( )** So với lấy ngưỡng cục bộ thì lấy ngưỡng động cho kết quả tốt hơn với ảnh có độ sáng kém và không đều ở từng vùng.

Sau khi lấy ngưỡng thì các ký tự vẫn còn hiện tượng nhiễu muối tiêu nên em tiếp tục áp dụng lọc trung vị và phép đóng để ảnh trông sắc nét rõ ràng và mượt hơn

Kết quả sau khi lấy ngưỡng:



#### Tách ký tự ra khỏi ảnh

Để tách từng ký tự ra khỏi biển số xe sử dụng thuật toán **Connected components analysis**. Sau khi áp dụng thuật toán ta sẽ thu đc các vùng ký tự, nhiễu (khung biển, dấu gạch, dấu chấm, …) và nền đã được gán nhãn.

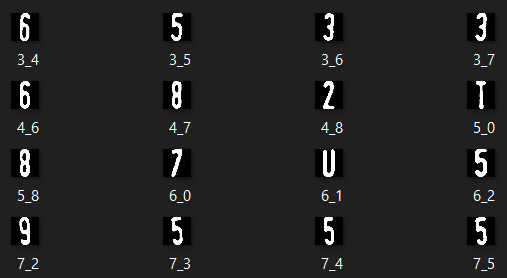
Sau khi thu được các vùng ta loại bỏ nền với nhãn bằng 0 và áp dụng hàm **findContour( )** để định hình các ký tự và nhiễu. Tiếp đến tạo một bao đóng hình chữ nhật xung quanh contour để tính toán và loại bỏ tối đa nhiễu.

Với mỗi contour bao quanh ký tự và nhiễu đều có đặc điểm nhất định nên ta sẽ dựa vào để xét các tỉ lệ ngưỡng loại bỏ nhiễu. Trong bước này em sử dụng bốn đại lượng là:

* AspectRatio: tỉ lệ giữa chiều dài và chiều rộng của contour
* Solidity: tỉ lệ giữa diện tích contour và diện tích hình chữ nhật bao quanh contour
* heightRatio: tỉ lệ giữa chiều cao của contour và biển số xe
* widthRatio: tỉ lệ giữa chiều rộng contour và biển số xe

Ngưỡng của các đại lượng được lấy qua thực nghiệm và quan sát ví dụ như chiều dài của một ký tự thường không vượt quá 40% và không bé hơn 20% chiều dài của biển số xe máy hình vuông nên sẽ lấy ngưỡng 0.2 < heightRatio < 0.4

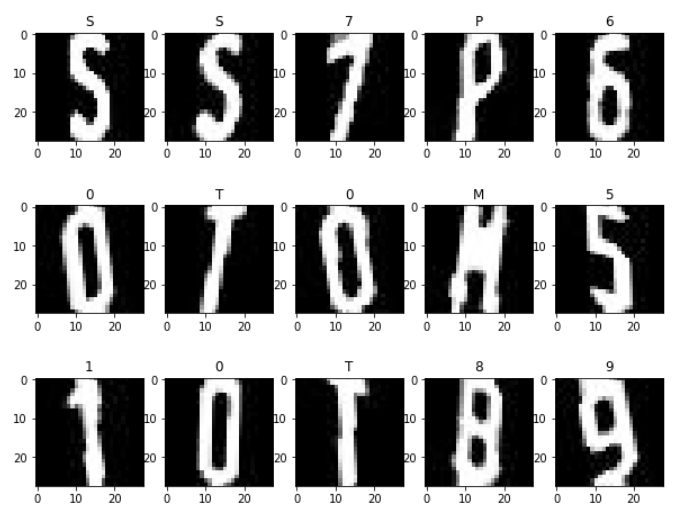
Kết quả thu được sau khi thực hiện thuật toán:



### Gán nhãn và chuẩn bị dữ liệu cho mô hình CNN

Sau khi Segment em thu được tập dữ liệu gồm 15000 ký tự chữ và số từ biển số xe chưa được gán nhãn.

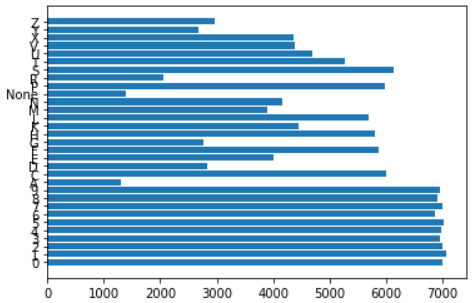
Do để phù hợp với dữ liệu bài toán và cho kết quả tốt nhất thì em thực hiện gán nhãn thủ công tập dữ liệu trên với 31 nhãn gồm các số từ 0 đến 9, các chữ cái alphabet xuất hiện trên biển số xe và một class nhiễu.



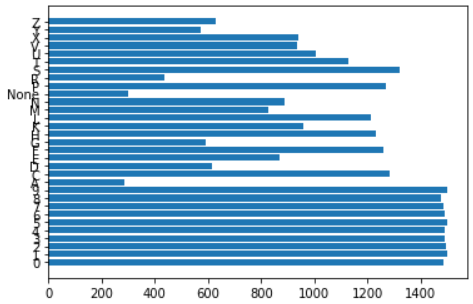
Sau khi hoàn thành việc gán nhãn, nhận thấy việc dữ liệu bị imbalance do các ký tự chữ cái trên biển số xe là rất ít so với chữ số, có những chữ cái sau khi hoàn thành việc gán nhãn chỉ xuất hiện < 50 lần nên em sử dụng các phương pháp Data Augumentation để làm giàu dữ liệu. Phụ thuộc vào số lượng dữ liệu ở mỗi lớp mà em sinh ngẫu nhiên ra các bức ảnh với phương pháp Rotation (xoay ảnh) và shear range (làm méo ảnh) mô phỏng việc biển số xe bị nghiêng.

Dữ liệu sau khi được làm giàu gồm 220.000 ảnh thuộc 32 class được chia theo tỉ lệ Train : Val : Test = 70:15:15

Tập Train: gồm 156441 ảnh

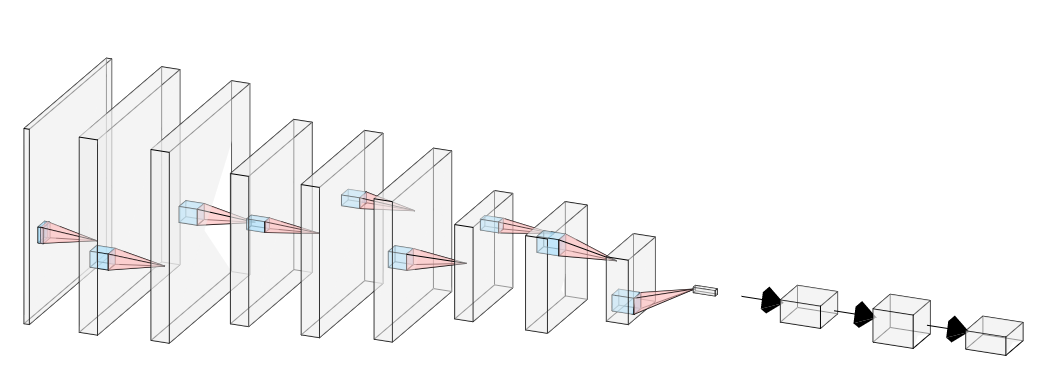


Tập Test: gồm 33191 ảnh

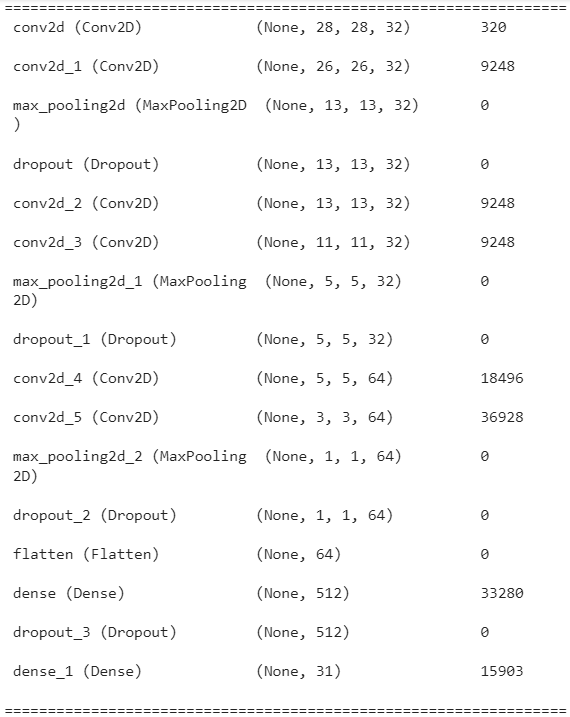


### Nhận diện ký tự với mạng CNN

#### Kiến trúc mô hình



* Đầu vào của mô hình là ảnh nhị phân kích thước (28×28×1)
* Tiếp theo là convolution layer với 6 lớp tích chập có mặt nạ kích thước (3×3) xen kẽ giữa 2 lớp tích chập là dropout và 1 lớp maxpooling có nhiệm vụ giảm số chiều



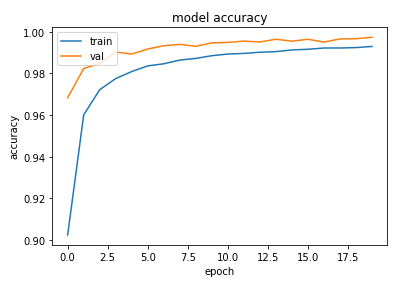
* (0) Tầng tích chập đầu tiên có mặt nạ kích thước (3×3×32), stride = 1, hàm kích hoạt Relu, cài đặt padding = “same” nên ta thu được feature map kích thước (28×28×32). Số lượng tham số = 3\*3\*32 + 32 = 320
* (1) Tầng tích chập thứ 1 cũng có mặt nạ kích thước (3×3×32), stride = 1, hàm kích hoạt Relu, padding = “valid” nên thu được feature map kích thước (26×26×32). Số lượng tham số = 3\*3\*32\*32 + 32 = 9248
* Tầng theo là lớp max pooling và drop out với mặt nạ kích thước (2×2), stride = 1, padding = 0, thu được feature map kích thước (13×13×32). Tầng này không có tham số
* (2) Tầng tích chập thứ 2 có mặt nạ kích thước (3×3×32), stride = 1, hàm kích hoạt Relu, cài đặt padding = “same” nên ta thu được feature map kích thước (13×13×32). Số lượng tham số = 3\*3\*32\*32 + 32 = 9248
* (3) Tầng tích chập thứ 3 có mặt nạ kích thước (3×3×32), stride = 1, hàm kích hoạt Relu, cài đặt padding = “valid” nên ta thu được feature map kích thước (11×11×32). Số lượng tham số = 3\*3\*32\*32 + 32 = 9248
* Tầng theo là lớp max pooling drop out với mặt nạ kích thước (2×2), stride = 1, padding = 0, thu được feature map kích thước (5×5×32). Tầng này không có tham số
* (4) Tầng tích chập thứ 4 có mặt nạ kích thước (3×3×64), stride = 1, hàm kích hoạt Relu, cài đặt padding = “same” nên ta thu được feature map kích thước (5×5×64). Số lượng tham số = 3\*3\*64\*32 + 64 = 18496
* (5) Tầng tích chập thứ 5 có mặt nạ kích thước (3×3×64), stride = 1, hàm kích hoạt Relu, cài đặt padding = “valid” nên ta thu được feature map kích thước (1×1×64). Số lượng tham số = 3\*3\*64\*64 + 64 = 36928
* Tầng theo là lớp max pooling drop out với mặt nạ kích thước (2×2), stride = 1, padding = 0, thu được feature map kích thước (1×1×64). Tầng này không có tham số
* Tiếp theo là lớp Flatten có
* Tần fully connected đầu tiên có kích thước 512, số lượng tham số = 512\*1\*1\*64 + 512 = 33280
* Tầng fully connected cuối cùng có kích thước bằng số class = 31. Số lượng tham số = 512\*31 + 31 = 15903

Mô hình tổng cộng có 132.671 tham số huấn luyện

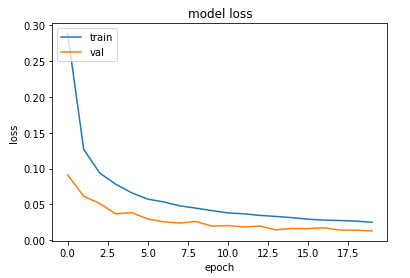
#### Huấn luyện mô hình

* Tập dữ liệu Train bao gồm 156441 ảnh nhị phân kích thước (28×28×1)
* Tập dữ liệu Val bao gồm 33191 ảnh nhị phân kích thước (28×28×1)
* Hàm loss “categorical\_crossentropy” của thư viện keras: hàm này yêu cầu nhãn label được mã hóa one hot
* Optimizer Adam với learing rate khởi tạo = 0.001
* Learning rate decay: sử dụng hàm ReduceLROnplateau( ) với đối số là monitor = “val\_accuracy”, factor = 0.2 (), patience = 5 (sau 5 epoch mà val\_accuracy không cải thiện thì sẽ giảm learning rate)
* Batch size bằng 256, huấn luyện với 50 epoch

#### Kết quả sau khi huấn luyện



Accuracy



Loss

Mô hình CNN cho kết quả rất tốt với accuracy trên cả tập Train và tập Val đều rất cao:

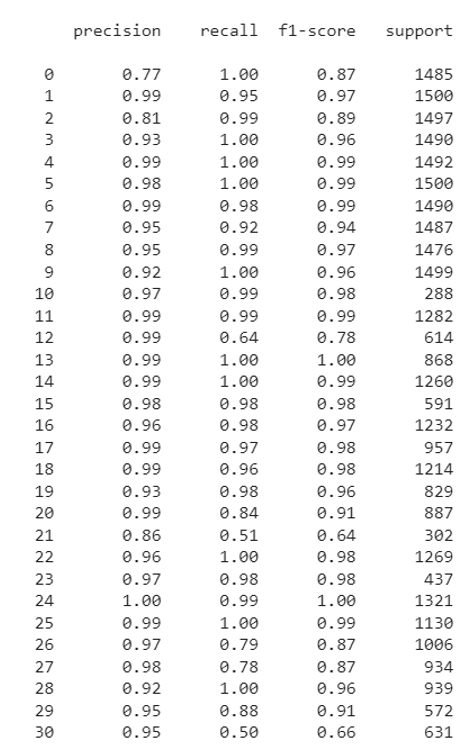
* loss: 0.0250 - accuracy: 0.9929
* val\_loss: 0.0130 - val\_accuracy: 0.9973
* Kiểm tra trên tập test:

test\_loss: 0.0085

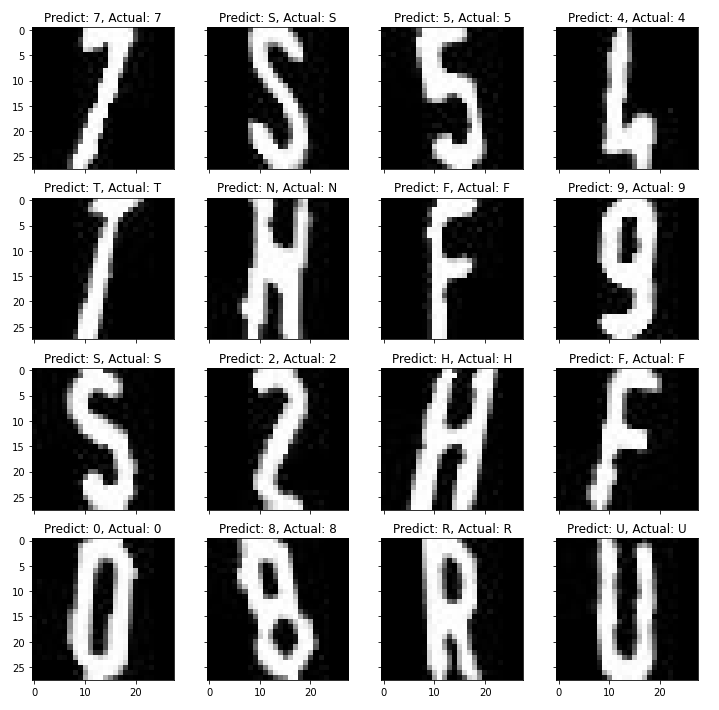
test\_acuracy: 0.9936

từ biểu đồ acuracy và loss giữa tập Train và Valiadation cùng với kết quả đánh giá trên tập test ta có thể thấy mô hình dự đoán rất tốt và không gặp phải hiện tượng Overfitting

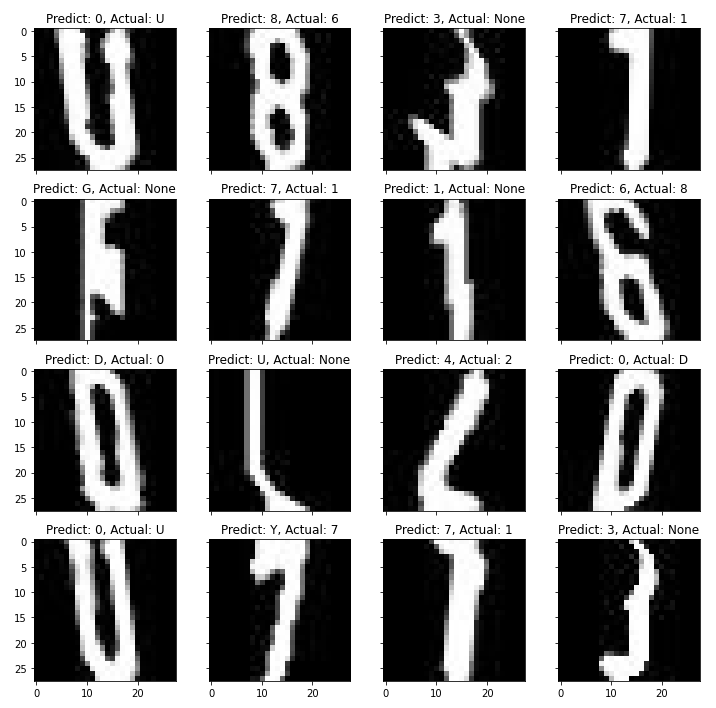
* Precision và Recall trên từng lớp



* Ta có thể thấy giữa hai ký tự dễ nhầm lẫn là 0 và D thì F1 score của hai lớp này thấp hơn các lớp khác.
* Lớp ký tự số 2 và Z cũng có hiện tượng tương tự như vậy thêm vào đó lớp 21 là nhiễu cũng dễ nhầm lẫn với các lớp khác nên cũng có Precision và Recall tương đối thấp



Các ký tự mô hình dự đoán đúng



Các ký tự mô hình dự đoán sai

### Các công cụ và IDE để phát triển mô hình

Trong Đề tài này, để giải quyết bài toán, thử nghiệm và phát triển hai mô hình là YOLOv3 nhận diện biển số và CNN nhận diện ký tự thì em thực hiện bằng python trên Google Colab.



# KẾT QUẢ ĐẠT ĐƯỢC

## Kết luận

Nội dung phần kết luận này tùy thuộc vào từng đồ án. Lưu ý trong phần kết luận không nên có bất cứ phương trình, biểu đồ hay bảng biểu nào. Cần trình bày rõ nội dung đồ án tốt nghiệp đã đáp ứng đầy đủ các yêu cầu của đề bài hay chưa. Trình bày về ý nghĩa của các kết quả thu được, các đánh giá nhận xét về tính khả thi, tính chính xác của kết quả, tính thực tế của đồ án…Cần lưu ý hạn chế sử dụng các tính từ, trạng từ mạnh trong khi miêu tả kết quả đạt được, cần đảm bảo tính trung thực của các kết luận.

Trình bày các kiến thức mà sinh viên đã đạt được sau khi thực hiện đồ án tốt nghiệp. Đồng thời trình bày về các kỹ năng đã học được (kỹ năng tự tìm kiếm tài liệu, tổng hợp thông tin, kỹ năng chế bản, kỹ năng trình bày, viết báo….).

## Hướng phát triển của đồ án trong tương lai

Nêu tóm tắt hướng mở rộng của đề tài trong tương lai nếu có. Đây là mục tùy chọn vì phụ thuộc vào loại đề tài.

# TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | Trần Bách, Lưới điện và hệ thống điện, Nhà xuất bản Khoa học Kỹ thuật, 2004. |
| [2] | Abe Masayuki, “A Practical Approach to Accurate Fault Location on Extra High Voltage Teed Feeders,” *IEEE Transaction on Power Delivery,* pp. 159-168, 1995. |
| [3] | Microsoft, "Add citations in a Word document," 2017. |

# PHỤ LỤC

1. **Chi tiết số liệu thí nghiệm**

Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có).

1. **Chi tiết các bước tính toán**

Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có).

1. **Chi tiết sơ đồ mô phỏng**

Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trình phụ lục tại đây (nếu có). Trìn