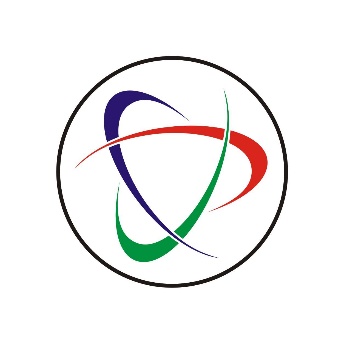
**ĐẠI HỌC ĐÀ NẴNG**

**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**KHOA ĐIỆN TỬ VIỄN THÔNG**



**BÁO CÁO**

**ĐỒ ÁN CHUYÊN NGÀNH**

**NGÀNH KỸ THUẬT MÁY TÍNH**

**ĐỀ TÀI: XÂY DỰNG MÔ HÌNH NHẬN DIỆN BIỂN BÁO GIAO THÔNG**

**GVHD : ThS. TRẦN THỊ MINH HẠNH**

**SINH VIÊN : NGUYỄN BÁ NHÂN**

**MÃ SINH VIÊN : 106170114**

**LỚP : 17DT2**

**Đà Nẵng, 08/2021**

**LỜI CẢM ƠN**

Lời đầu tiên, em xin chân thành cảm ơn ThS. Trần Thị Minh Hạnh. Trong quá trình tìm hiểu và thực hiện đồ án, em đã nhận được sự quan tâm giúp đỡ, hướng dẫn rất tận tình, tâm huyết của cô.

Em cũng xin chân thành cảm ơn các thầy cô giáo trong trường ĐH Bách Khoa Đà Nẵng, đặc biệt là các thầy cô giáo của Khoa Điện tử-Viễn thông đã dạy dỗ cho em các kiến thức từ đại cương cho đến chuyên ngành, giúp em có được cơ sở lý thuyết cũng như kỹ năng cần thiết cho hành trang của mình.

**Mục lục**

[CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI NHẬN Dạng BIỂN BÁO GIAO THÔNG 7](#_Toc80777976)

[1.1 Giới thiệu chương 7](#_Toc80777977)

[1.2 Giới thiệu về thị giác máy tính 7](#_Toc80777978)

[1.3 Khái quát về nhận dạng 8](#_Toc80777979)

[1.3.1 Giới thiệu về nhận dạng 8](#_Toc80777980)

[1.3.2 Bản chất của quá trình nhận dạng 8](#_Toc80777981)

[1.4 Các phương pháp nhận dạng 9](#_Toc80777982)

[1.5 Khái quát về đề tài nhận dạng biển báo giao thông 9](#_Toc80777983)

[1.5.1 Đặt vấn đề 9](#_Toc80777984)

[1.5.2 Tóm tắt quá trình thực hiện đề tài 10](#_Toc80777985)

[1.6 Tổng kết chương 10](#_Toc80777986)

[CHƯƠNG 2: CỞ SỞ LÝ THUYẾT 11](#_Toc80777987)

[2.1 Giới thiệu chương 11](#_Toc80777988)

[2.2 Deep learning 11](#_Toc80777989)

[2.3 Neural Network (1) 11](#_Toc80777990)

[2.3.1 Perceptron 11](#_Toc80777991)

[2.3.2 Bias 13](#_Toc80777992)

[2.3.3 Sigmoid Neurons 13](#_Toc80777993)

[2.3.4 Kiến trúc mạng Neural Network 14](#_Toc80777994)

[2.4 Convolutional Neural Network (2) 15](#_Toc80777995)

[2.4.1 Tổng quát 15](#_Toc80777996)

[2.4.2 Lớp tích chập - Convolutional Layer 16](#_Toc80777997)

[2.4.3 Lớp lấy mẫu - Pooling layer 16](#_Toc80777998)

[2.4.4 Lớp kết nối đầy đủ - Fully Connected Layer 17](#_Toc80777999)

[2.4.5 Softmax 18](#_Toc80778000)

[2.4.6 Batch Normalization 18](#_Toc80778001)

[2.5 Retinanet 19](#_Toc80778002)

[2.5.1 Focal Loss (1) 19](#_Toc80778003)

[2.5.2 Retina net (2) 23](#_Toc80778004)

[2.6 Tổng kết chương 24](#_Toc80778005)

[CHƯƠNG 3: THỰC NGHIỆM VÀ ĐÁNH GIÁ KẾT QUẢ 25](#_Toc80778006)

[3.1 Giới thiệu chương 25](#_Toc80778007)

[3.2 Thu thập dữ liệu 25](#_Toc80778008)

[3.3 Tiền xử lý dữ liệu 26](#_Toc80778009)

[3.3.1 Tiền xử lý và gán nhãn cho ảnh 26](#_Toc80778010)

[3.3.2 Chia dữ liệu thành tập train và validation 28](#_Toc80778011)

[3.4 Chuẩn bị huấn luyện mô hình 29](#_Toc80778012)

[3.5 Huấn luyện mô hình 29](#_Toc80778013)

[3.6 Kết quả 29](#_Toc80778014)

[3.6.1 Kết quả huấn luyện 29](#_Toc80778015)

[3.6.2 Kết quả nhận dạng trên ảnh thực 31](#_Toc80778016)

[3.7 Đánh giá kết quả 32](#_Toc80778017)

[3.7.1 Ưu điểm 32](#_Toc80778018)

[3.7.2 Hạn chế 32](#_Toc80778019)

[3.8 Hướng phát triển 33](#_Toc80778020)

**Danh mục bảng số liệu**

[Bảng 3.1 Bộ dữ liệu Zalo\_traffic\_2020 26](#_Toc80649871)

[Bảng 3.2 Dữ liệu sau khi phân chia 28](#_Toc80649872)

[Bảng 4.1 Kết quả 20 epoch đầu tiên 30](#_Toc80649873)

**DANH MỤC HÌNH ẢNH**

[Hình 2.1 nơ-ron sinh học 12](#_Toc80649874)

[Hình 2.2 Perceptron 12](#_Toc80649875)

[Hình 2.3 Neural Network 14](#_Toc80649876)

[Hình 2.4 Neural Network 2 hidden layer 15](#_Toc80649877)

[Hình 2.5 Cấu trúc CNN 16](#_Toc80649878)

[Hình 2.6 Max pooling kích thước 2x2 17](#_Toc80649879)

[Hình 2.7 Phép flate Phép flatten đưa tensor về thành 1 vector 17](#_Toc80649880)

[Hình 2.8 Phép biến đổi Batch normalization 19](#_Toc80649881)

[Hình 2.9 Ví dụ cho focal loss 21](#_Toc80649882)

[Hình 2.10 Kiến trúc của RetinaNet 23](#_Toc80649883)

[Hình 3.1 Ảnh thu thập thông qua chụp trên đường phố 25](#_Toc80649884)

[Hình 4.1 Kết quả train ở epoch 74 30](#_Toc80649885)

[Hình 4.2 Kết quả nhận dạng trên ảnh thực 31](#_Toc80649886)

[Hình 4.3 Kết quả nhận dạng trên ảnh thực 31](#_Toc80649887)

[Hình 4.4 Kết quả nhận dạng trên ảnh thực 32](#_Toc80649888)

**DANH MỤC BIỂU ĐỒ**

[Biểu đồ 2.1 Đồ thị hàm Sigmoid 14](#_Toc80649889)

[Biểu đồ 2.2 Đồ thị của focal loss 22](#_Toc80649890)

[Biểu đồ 3.1 Biểu đồ nhãn dữ liệu 26](#_Toc80649891)

[Biểu đồ 3.2 Mật độ các đối tượng xuất hiện trong tập train 28](#_Toc80649892)

[Biểu đồ 4.1 Kết quả 20 epoch đầu tiên 29](#_Toc80649893)

**DANH MỤC TỪ VIẾT TẮT**

|  |  |
| --- | --- |
| **NN** | **Neural Network** |
| **CNN** | **Convolutional Neural Network** |
| **FM** | **Feature Map** |
| **FPN** | **Feature Pyramid Network** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# TỔNG QUAN ĐỀ TÀI NHẬN Dạng BIỂN BÁO GIAO THÔNG

## Giới thiệu chương

Chương này sẽ cung cấp những thông tin cơ bản về thị giác máy tính xử lý ảnh và các ứng dụng thực tế về xử lý ảnh trong đời sống xã hội hiện nay. Từ đó khái quát về đề tài nhận dạng biển báo giao thông và lý do chọn đề tài.

## Giới thiệu về thị giác máy tính

Thị giác máy tính (computer vision) là một lĩnh vực bao gồm các phương pháp thu nhận, xử lý ảnh kĩ thuật số, phân tích và nhận dạng các hình ảnh và, nói chung là dữ liệu đa chiều từ thế giới thực để cho ra các thông tin số hoặc biểu tượng, ví dụ trong các dạng quyết định. Việc phát triển lĩnh vực này có bối cảnh từ việc sao chép các khả năng thị giác con người bởi sự nhận diện và hiểu biết một hình ảnh mang tính điện tử.  Sự nhận diện hình ảnh có thể xem là việc giải quyết vấn đề của các biểu tượng thông tin từ dữ liệu hình ảnh qua cách dùng các mô hình được xây dựng với sự giúp đỡ của các ngành lý thuyết học, thống kê ,vật lý và hình học. Thị giác máy tính cũng được mô tả là sự tổng thể của một dải rộng các quá trình tự động và tích hợp và các thể hiện cho các nhận thức thị giác.

Thị giác máy tính là một môn học khoa học liên quan đến lý thuyết đằng sau các hệ thống nhân tạo có trích xuất các thông tin từ các hình ảnh. Dữ liệu hình ảnh có thể nhiều dạng, chẳng hạn như chuỗi video, các cảnh từ đa camera, hay dữ liệu đa chiều từ máy quét y học. Thị giác máy tính còn là một môn học kỹ thuật, trong đó tìm kiếm việc áp dụng các mô hình và các lý thuyết cho việc xây dựng các hệ thống thị giác máy tính.

Thị giác nhận tạo quan tâm việc mô phỏng lại thị giác của con người, dùng mô hình toán học để biểu diễn và hiểu các hình ảnh từ thực tế.

* Ứng dụng:
* Hệ thống an ninh, bảo mật thông tin.
* Xe tự hành.
* Y khoa (X-quang, ảnh tia gamma).
* Thiên văn học.
* Phát hiện lỗi sản phẩm ứng dụng trong công nghiệp.
* Tìm kiếm thông tin bằng hình ảnh trên internet.

## Khái quát về nhận dạng

### Giới thiệu về nhận dạng

Nhận dạng là quá trình nhận diện được và phân loại các đối tượng được biển diễn theo mô hình nào đó và gán cho chúng một nhãn theo những tính toán quy luật và các mẫu có sẵn. Quá trình đó dựa vào các mẫu mà máy đã được học và biết trước. Trong trường hợp khác gọi là không được học.

* Trong lý thuyết nhận dạng ảnh có ba cách tiếp cận khác nhau:
  + Nhận dạng theo cấu trúc.
  + Nhận dạng dựa vào phân hoạch không gian
  + Nhận dạng dựa vào kĩ thuật mạng nơron
* Hai cách tiếp cận đầu các đối tượng quan sát và thu được phải trải qua tiền xử lý nhằm tăng chất lượng ảnh, làm nổi các chi tiết, trích chọn và biển diễn các đặc trưng và cuối cùng là nhận dạng.
* Cách tiếp cận thứ 3 hoàn toàn khác, dựa vào cơ chế đoán nhận, lưu trữ và phân biệt đối tượng mô phỏng theo hoạt động của hệ thần kinh con người. Do cơ chế đặc biệt, các đối tượng thu nhận bởi thị giác người không cần qua giai đoạn cải thiện mà chuyển ngay sang giai đoạn tổng hợp, đối sánh với các mẫu đã lưu trữ để nhận dạng.

### Bản chất của quá trình nhận dạng

Quá trình nhận dạng gồm 3 giai đoạn chính:

* Lựa chọn mô hình biểu diễn đối tượng.
* Lựa chọn luật ra quyết định (phương pháp nhận dạng) và suy diễn quá trình.
* Học nhận dạng.

Khi mô hình biểu diễn đối tượng đã được xác định, quá trình nhận dạng chuyển sang giai đoạn học. Học là giai đoạn rất quan trọng. Thao tác học nhằm cải thiện, điều chỉnh việc phân hoạch tập đối tượng thành các lớp.

Việc nhận dạng chính là tìm ra quy luật và các thuật toán để có thể gán đối tượng vào một lớp.

## Các phương pháp nhận dạng

* *Nhận dạng dựa trên phân hoạch không gian.*

Trong kỹ thuật này, các đối tượng nhận dạng là các đối tượng định lượng. Một đối tượng được biểu diễn bởi một vector nhiều chiều.

* *Nhận dạng theo cấu trúc.*

Ngoài cách biểu diễn theo định lượng như đã mô tả ở trên, tồn tại nhiều kiểu đối tượng mang tính định tính. Trong cách biểu diễn này, người ta quan tâm đến các dạng và mối quan hệ giữa chúng.

* *Nhận dạng theo mạng nơron.*

Mạng noron nhân tạo (Artificial Neural Network) bao gồm các nút được nối với nhau bởi các liên kết noron. Mỗi liên kết kèm theo một trọng số nào đó. Nhiệm vụ của quá trình huấn luyện (học) mạng là cập nhật các trọng số khi có thêm các thông tin về các mẫu học, hay nói cách khác, các trọng số được điều chỉnh sao cho dáng điệu vào ra của nó mô phỏng hoàn toàn phù hợp môi trường đang xem xét.

Trong mạng, một số noron được nối với môi trường bên ngoài như các đầu ra, đầu vào.

## Khái quát về đề tài nhận dạng biển báo giao thông

### Đặt vấn đề

Trong vài năm trở lại đây, Artificial Intelligence (trí tuệ nhân tạo), Computer vision (thị giác máy tính)… đang phát triển rất nhanh chóng và mạnh mẽ. Ứng dụng của AI rất đa dạng và ảnh hưởng tích cực đến đời sống như: Ô tô tự lái, hệ thống dịch tự động, chatbox tự động trả lời… Hiện nay, có rất nhiều nghiên cứu về bài toán nhận diện biển báo giao thông, các hệ thống từ đơn giản đến phức tạp, sử dụng nhiều thuật toán khác nhau.

Đối với bài toán phân loại biển báo giao thông, tức là nhận diện bức ảnh này thuộc loại biển báo gì. Với mỗi ngõ vào hình ảnh, ta cần phát hiện được biển báo giao thông trong ảnh sau đó phân loại biển báo đó thuộc loại gì. Đây là bài toán phân lớp (Classification). Có nhiều thuật toán để phân lớp dữ liệu, từ cổ điển đến hiện đại. Nhưng mạng Nơ- ron tích chập được đánh giá là thuật toán hiệu quả, tránh được tác động từ môi trường, nhiễu, sự thay đổi khoảng cách ảnh đến camera. Trong đề tài này, mục tiêu tôi đặt ra là xây dựng mô hình nhận diện biển báo giao thông trong video. Từ đó so sánh hiệu quả giữa các mô hình khác nhau.

Có rất nhiều mô hình để phân loại đối tượng có thể sử dụng trong bài toán này với độ hiệu quả đặc trưng khác nhau như: R-CNN, Fast R-CNN, Faster R-CNN, Retinanet, YOLO… Trong đồ án này, tôi hướng tới độ chính xác cao và khả năng nhận dạng tốt những đối tượng nhỏ. Đáp ứng yêu cầu đó, Retinanet là mô hình phù hợp khi mà có độ chính xác được đánh giá là cao hơn so với các mô hình khác, đồng thời khả năng phát hiện đối tượng cũng là rất mạnh.

### Tóm tắt quá trình thực hiện đề tài

Quá trình thực hiện đề tài được chia làm nhiều giai đoạn:

* Tìm hiểu về các thuật toán và mô hình phát hiện, phân loại đối tượng để chọn mô hình phù hợp.
* Nghiên cứu xây dựng một mô hình CNN, Retinanet.
* Thu thập và tiền xử lý dữ liệu, phân chia dữ liệu.
* Train mô hình.
* Thử nghiệm và đánh giá độ chính xác.
* Chỉnh sửa và hoàn thiện.

## Tổng kết chương

Chương một đã khái quát một số thông tin, khái niệm cơ bản về thị giác máy tính và bài toán nhận dạng. Qua đó giới thiệu sơ bộ về đề tài nhận dạng biển báo giao thông. Trong chuong tiếp theo ta sẽ tìm hiểu một số lý thuyết về học máy, mạng CNN và mô hình Retinanet được sử dụng trong đồ án này.

# CỞ SỞ LÝ THUYẾT

## Giới thiệu chương

Chương này đưa ra lý thuyết về Deep Learning, mạng Neural Network và Retinanet. Mục 2.2 giới thiệu giản lượt về Deep learning, mục 2.3 trình bày về mạng Neural Network từ đó đưa ra cơ sở để trình bày về thuật toán Convolutional Neural Network trong mục 2.4, và mục 2.5 trình bày về Retinanet.

## Deep learning

Deep Learning (Học sâu) là một nhánh nhỏ của Machine Learning (Máy học), bắt nguồn từ thuật toán Neural Network. Tuy nhiên, gần đây Deep Learning mang lại nhiều thành tựu kĩ thuật nhờ vào sự phát triển của Big Data (Dữ liệu lớn), với tập dữ liệu càng lớn giúp phát triển các mạng Neural Network sâu hơn với khả năng dự đoán chính xác cao hơn. Bắt nguồn từ các loại hình kinh doanh internet truyền thống như tìm kiếm web và quảng cáo. Deep learning ngày nay đã tạo ra nhiều lợi ích hơn trong cuộc sống hằng ngày. Ví dụ như trong lĩnh vực y tế, việc phân tích các hình ảnh X-quang được ứng dụng vào Deep Learning để nghiên cứu và dự đoán các khả năng mang bệnh của người khám. Và đặc biệt Deep Learning còn được ứng dụng trong các xe tự hành để dự đoán các tình huống giao thông.

## Neural Network (1)

Mạng nơ-ron nhân tạo (*Neural Network* - **NN**) là một mô hình lập trình rất đẹp lấy cảm hứng từ mạng nơ-ron thần kinh. Kết hợp với các kĩ thuật học sâu (*Deep Learning*), NN đang trở thành một công cụ rất mạnh mẽ mang lại hiệu quả tốt nhất cho nhiều bài toán khó như nhận dạng ảnh, giọng nói hay xử lý ngôn ngữ tự nhiên.

### Perceptron

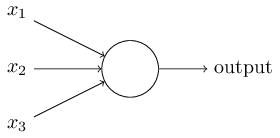
Một mạng nơ-ron được cấu thành bởi các nơ-ron đơn lẻ được gọi là các *perceptron*. Nên trước tiên ta tìm hiểu xem perceptron là gì đã rồi tiến tới mô hình của mạng nơ-ron sau. Nơ-ron nhân tạo được lấy cảm hứng từ nơ-ron sinh học như hình mô tả bên dưới:



Hình 2.1 nơ-ron sinh học

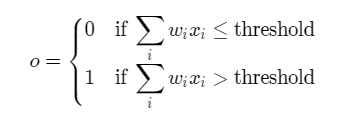
Như hình trên, ta có thể thấy một nơ-ron có thể nhận nhiều đầu vào và cho ra một kết quả duy nhất.

Mô hình của perceptron cũng tương tự như vậy:

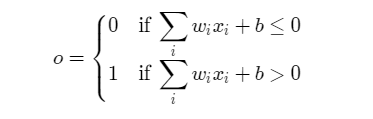


Hình 2.2 Perceptron

Một perceptron sẽ nhận một hoặc nhiều đầu **x** vào dạng nhị phân và cho ra một kết quả output 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 b nào đó:



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

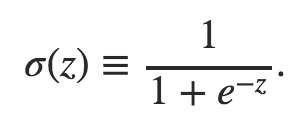


### Bias

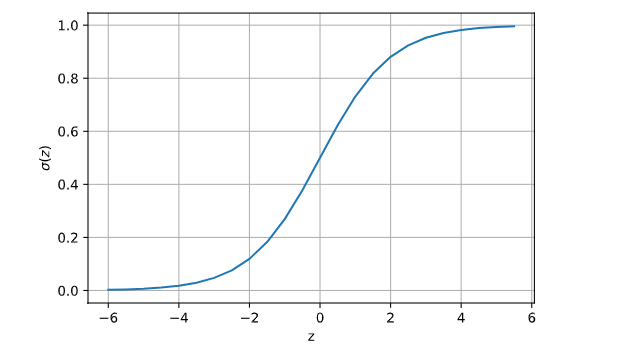
Giá trị ngưỡng b ở trên còn được gọi là bias, tuy nhiên thông thường ta sẽ không so sánh giá trị output với bias, mà sẽ xem giá trị bias giống như là một input bổ sung. Nghĩa là thay vì ở trên chúng ta có 3 giá trị input, thì bây giờ chúng ta sẽ có giá trị input thứ tư chính là giá trị b, và trọng số của nó luôn luôn là 1. Điều này giúp cho việc so sánh giá trị output của chúng ta sẽ thuận tiện hơn vì chỉ cần so sánh output xem nó lớn hơn 0 hay nhỏ hơn 0 là được. Lớn hơn 0 thì xem là 1, còn nhỏ hơn 0 thì xem như là 0.

### Sigmoid Neurons

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 có công thức:



Đồ thị hàm sigmoid:



Biểu đồ 2.1 Đồ thị hàm Sigmoid

### Kiến trúc mạng Neural Network

Mạng NN 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 2.3 Neural Network

Một mạng NN sẽ có 3 kiểu tầng:

* **Tầng vào** (*input layer*): Là tầng bên trái cùng của mạng thể hiện cho các đầu vào của mạng.
* **Tầng ra** (*output layer*): Là tầng bên phải cùng của mạng thể hiện cho các đầu ra của mạng.
* **Tầng ẩn** (*hidden layer*): Là tầng nằm giữa tầng vào và tầng ra thể hiện cho việc suy luận logic của mạng

Một NN chỉ có một tầng vào và một tầng ra nhưng có thể có nhiều tầng ẩn.



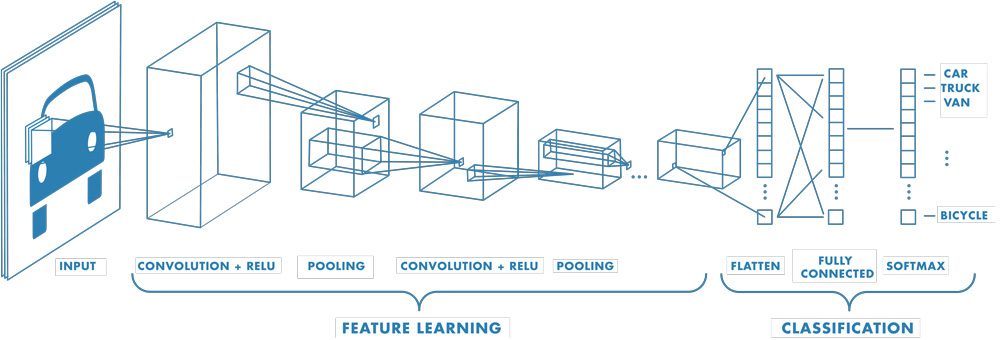
Hình 2.4 Neural Network 2 hidden layer

## Convolutional Neural Network (2)

### Tổng quát

Convolutional Neural Network (CNN) hay còn được gọi là mạng Nơ-ron tích chập là một trong những mô hình của Deep Learning. Tác dụng của thuật toán này chính là tạo ra những hệ thống thông minh, có sự phản ứng với độ chính xác cao. Ví dụ như Facebook, Google,.. đã đưa vào sản phẩm của mình chức năng nhận diện khuôn mặt,… Ứng dụng cơ bản nhất của thuật toán này là phân lớp, tức là phân biệt hoặc là cái này hoặc là cái kia, tức là khi đưa hình ảnh vào máy tính, nó sẽ là các điểm ảnh hai chiều và điều CNN thực hiện đó là khi các điểm ảnh thanh đổi thì máy tính vẫn biết được đó là hình ảnh gì. CNNs được chia thành 3 chiều : rộng, cao, sâu. Các Nơ- ron trong mạng không liên kết hoàn toàn với toàn bộ Nơ-ron kế đến mà chỉ liên kết tới một vùng nhỏ. Cuối cùng, một tầng đầu ra được tối giản thành vec-tơ của giá trị xác suất. CNNs gồm 2 thành phần:

* Phần tầng ẩn hay phần rút trích đặc trưng: Trong phần này, mạng sẽ tiến hành tính toán hàng loạt phép tích chập (Convolutional layer) và phép hợp nhất (pooling) để phát hiện các đặc trưng.
* Phần phân lớp: Tại phần này, một số lớp các liên kết đầy đủ (Fully Connected) sẽ đóng vai trò như một bộ phân lớp các đặc trưng đã rút trích trước đó.Tầng này sẽ đưa ra xác suất của một đối tượng trong hình.



Hình 2.5 Cấu trúc CNN

### Lớp tích chập - Convolutional Layer

Đây là phép tính quan trọng trong CNN. Ta thực hiện phép tích chập bằng cách trượt kernel/filter theo kiểu dữ liệu đầu vào. Tại mỗi vị trí, ta tiến hành phép nhân ma trận và tính tổng để đưa vào feature map. Feature map này chính là một ma trận 3 chiều, trong đó bao gồm những con số (parameter).

Ở lớp đầu tiên, khối tích chập dùng để phát hiện cạnh thẳng đứng và cạnh nằm ngang, cạnh nghiêng. Ở các lớp tiếp theo, khối tích chập sẽ nhận diện các đường cong, đường gấp khúc.Càng sử dụng nhiều lớp tích chập thì càng phát hiện được những đặc trưng phức tạp hơn, có thể là các vật thể như con mèo, bông hoa, bánh xe.

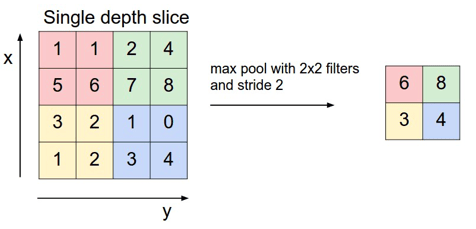
Tại khối tích chập, các thông số được sử dụng bao gồm:

* Stride (S) : Số pixel bạn dịch chuyển ở mỗi lần trượt.
* Padding (P): Tăng thêm một số hàng và cột ( thường có giá trị bằng 0) để tránh việc giảm kích thước ở output cũng như để lấy thông tin ở các pixel ngoài cùng. Có 2 loại padding là “valid” tức là padding =0 và “same” tức là padding để kích thước ngõ ra bằng kích thước ngõ vào.

### Lớp lấy mẫu - Pooling layer

Mục đích là làm giảm số parameters mà ta cần phải tính toán, từ đó giảm thời gian tính toán mà vẫn giữ được các đặc trưng quan trọng. Ngoài ra nó còn tránh overfitting.

Có 2 loại pooling: max pooling lấy giá trị lớn nhất trong một pooling window và average pooling lấy giá trị trung bình trong một pooling window.

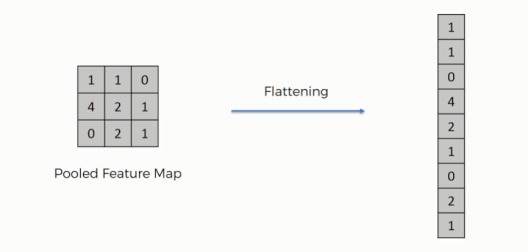


Hình 2.6 Max pooling kích thước 2x2

Pooling hoạt động gần giống convolution, ở max pooling cửa sổ trượt qua từng giá trị của ma trận dữ liệu đầu vào, chọn ra giá trị lớn nhất nằm trong cửa sổ trượt.

### Lớp kết nối đầy đủ - Fully Connected Layer

Sau khi ảnh được truyền qua nhiều lớp tích chập và lớp tổng hợp thì model đã học được tương đối các đặc điểm của ảnh (ví dụ bánh xe, khung xe,…) thì tensor của output cuối cùng, kích thước H\*W\*D sẽ được chuyển về 1 vector kích thước (H\*W\*D).

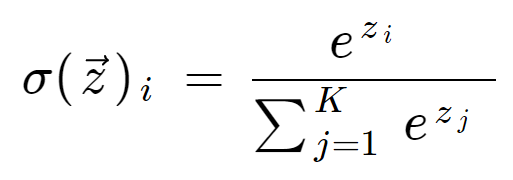


Hình 2.7 Phép flate Phép flatten đưa tensor về thành 1 vector

Sau đó, mỗi điểm của vector sẽ được liên kết với toàn bộ output của mode giống như một lớp của mạng Neural Network. Lớp kết nối đầy đủ cuối của mạng có nhiệm vụ phân loại theo yêu cầu của bài toán.

### Softmax

Sau khi mạng CNNs học qua các lớp phía trước, ta thu được vector đặc trưng. Khi đó, cần dựa vào vector đặc trưng này để phân loại vào các lớp theo yêu cầu bài toán. Hàm softmax là một mô hình xác suất để với mỗi input x, tính được output *σ*i thể hiện xác suất để input đó rơi vào lớp thứ i.



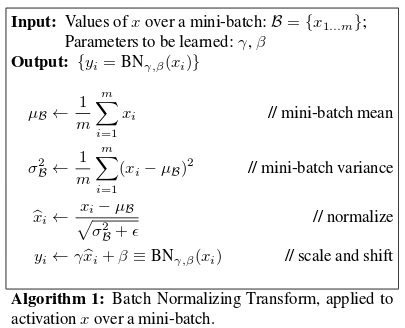
### Batch Normalization

Trong quá trình training,có thể xảy ra hiện tượng gradients không ổn định do gradients trước và sau khi di chuyển qua 1 layers không giống nhau. Đây là vấn đề liên quan đến phân phối của inputs của các layers.

Batch Normalization học những giá trị trung bình và phương sai của đầu vào sau đó chuẩn hóa đầu vào để có giá trị trung bình là 0 và phương sai là 1. Việc chuẩn hóa dữ liệu đầu vào giúp cho quá trình huấn luyện nhanh hơn (hàm chi phí hội tụ nhanh hơn) so với việc không chuẩn hóa.

Một lợi ích được nhận định của batch normalization là huấn luyện nhanh hơn. Trong đề tài đồ án này, chúng tôi kiểm tra hiệu quả của batch normalization thông qua việc so sánh mô hình baseline và mô hình baseline có sử dụng batch normalization.

Thuật toán chuẩn hóa cụ thể là:



Hình 2.8 Phép biến đổi Batch normalization

## Retinanet

### Focal Loss (1)

#### Focal Loss Function

Trong bài báo được trình bày vào tháng 1, 2018 tựa đề Focal Loss for Dense Object Detection, nhóm tác giả Tsung-Yi Lin, Priya Goyal, ... của FAIR (Facebook AI research) đã công bố một hàm loss function mới mang tính đột phá trong việc cải thiện hiệu xuất của lớp mô hình one-stage detector trong object detection.

Dựa trên nhận định rằng mất cân bằng dữ liệu giữa các nhóm foreground-background là nguyên nhân chính dẫn tới sự kém hiệu quả, xin trích dẫn:

*“We discover that the extreme foreground-background class imbalance encountered during training of dense detectors is the central cause”* (1)

Nhóm tác giả đã đề xuất một sự điều chỉnh trong hàm cross entropy loss để giải quyết triệt để ảnh hưởng của mất cân bằng dữ liệu.

#### Cross Entropy Loss

giá trị cross entropy đối với phân phối xác suất thực tế p và phân phối xác suất dự báo q:



Tại các nhãn bằng 0 giá trị đóng góp vào loss function bằng 0. Do đó cross entropy có thể viết lại thành:



Trong cross entropy ta thấy rằng vai trò đóng góp vào loss function của các class cùng bằng -log(pi). Khi xảy ra hiện tượng mất cân bằng, chúng ta muốn rằng mô hình sẽ dự báo chuẩn hơn đối với những class thiểu số. Do đó cần một hàm loss function hiệu quả hơn, có thể điều chỉnh được giá trị phạt lớn hơn nếu dự báo sai đối với nhóm thiểu số. Mục đích là để hạn chế dự báo sai nhóm thiểu số vì nếu dự báo sai nhóm thiểu số thì hàm loss function sẽ trở nên lớn hớn.

#### Hàm balanced cross entropy

Cách tự nhiên nhất là áp dụng trọng số bằng nghịch đảo tần suất nhãn vào cross entropy. Hàm loss function mới được gọi là *balanced cross entropy*:

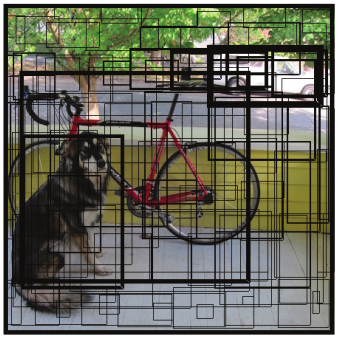


Trong đó  , fi là tần suất của class i. Chúng ta cộng thêm ϵ *dương rất nhỏ* để tránh mẫu bằng 0. Với hàm loss function này, các classes xuất hiện ít hơn thì có giá trị tác động tới loss function lớn hơn.

Hàm *balanced cross entropy* là hàm số cân bằng được tỷ lệ phân phối của mẫu. Nhưng nó chưa thực sự thay đổi được gradient descent của loss function. Trong khi mô hình được huấn luyện trên mẫu mất cân bằng trầm trọng có giá trị gradient descent chịu ảnh hưởng phần lớn bởi class chiếm đa số. Do đó chúng ta cần một sự điều chỉnh triệt để hơn giúp gia tăng ảnh hưởng của nhóm thiểu số lên gradient descent. Đó chính là hàm *focal loss*, một hàm số tiếp tục kế thừa *balanced cross entropy* và điều chỉnh được *gradient descent*.

#### Sự ra đời của focal loss

Focal loss là hàm loss function lần đầu được giới thiệu trong RetinaNet. Hàm loss function này đã chứng minh được tính hiệu quả trong các bài toán object detection. Đây là lớp bài toán có sự mất cân bằng nghiêm trọng giữa hai class positive (các bounding box có chứa object) và negative (các bounding box không chứa object). Thường thì *negative* có số lượng lớn hơn *positive* rất nhiều. Lấy ví dụ như hình bên dưới:



Hình 2.9 Ví dụ cho focal loss

Chỉ có 4 bounding box thuộc positive (đường viền in đậm), các trường hợp còn lại thuộc nhóm negative.

Do đó nếu áp dụng loss function là hàm cross entropy sẽ giảm độ chính xác khi dự báo các bounding box có chứa object. Trong bài báo Focal Loss for Dense Object Detection tác giả đã giới thiệu hàm Focal Loss dựa trên hai tham số là α,γ như sau:



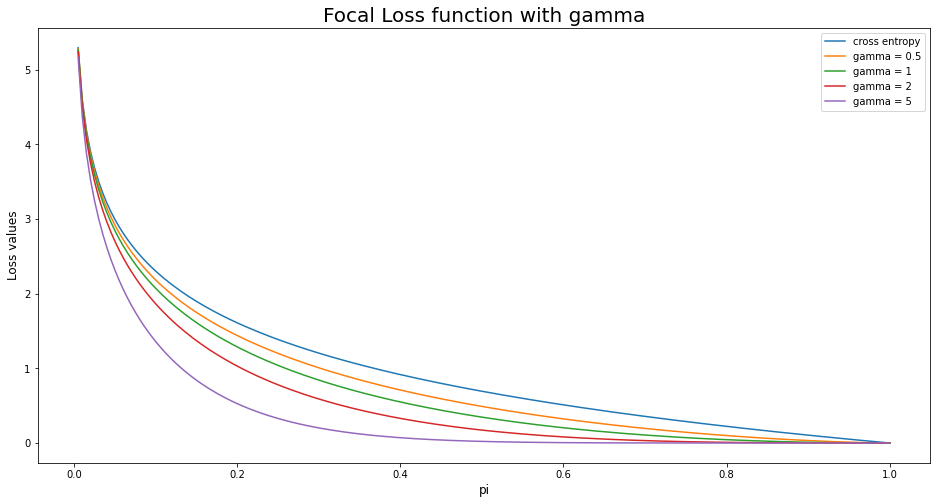
Chúng ta thường chọn giá trị γ ∈ [0,5].

Ta thấy hàm *focal loss* chỉ thêm nhân tử (1−qi) γ so với công thức của *balanced cross entropy*. Tuy nhiên nhân tử này lại có tác dụng rất lớn trong việc điều chỉnh *ảnh hưởng của nhãn* lên đồng thời *loss function* và *gradient descent*. Thật vậy, xét hai trường hợp dễ dự báo và khó dự báo:

* Dễ dự báo: Chúng ta thấy rằng mô hình huấn luyện trên mẫu mất cân bằng dễ dàng dự báo chính xác các mẫu đa số. Những trường hợp này được gọi là dễ dự báo. Xác suất qi của của các trường hợp dễ dự báo có xu hướng cao hơn. Do đó (1−qt) γ có xu hướng rất nhỏ và dường như không tác động lên loss function đáng kể.
* Khó dự báo: Trường hợp khó dự báo thì qi là một giá trị nhỏ hơn. Do đó độ lớn tác động của nó lên loss function là (1−qi) ­γ sẽ gần bằng 1. Mức độ tác động này lớn hơn nhiều lần so với trường hợp dễ dự báo. Cụ thể hơn, nếu trường hợp dễ dự báo có qi=0.9 và khó dự báo có qi=0.1 thì tỷ lệ chênh lệch của đóng góp vào loss function khi γ=2 sẽ là:



Tỷ lệ này sẽ còn lớn hơn nữa nếu tăng γ hoặc giá trị của qi đối với trường hợp dễ dự báo càng gần 1 và khó dự báo càng gần 0.



Biểu đồ 2.2 Đồ thị của focal loss

Trục hoành thể hiện xác suất là background của bounding box. Các giá trị xác suất pi quanh 1 là xác suất cao. Do chúng là nhóm chiếm đa số nên là nhóm dễ dự báo. Các giá trị xác suất pi quanh 0 là các trường hợp không phải background và chúng là những nhóm chiếm thiểu số và khó dự báo hơn.

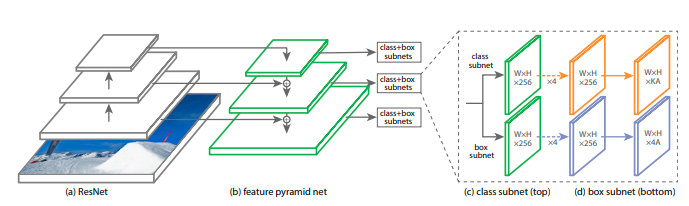
Khi đường cong của focal loss càng trũng xuống thì tỷ số về loss function giữa các trường hợp khó dự báo so với dễ dự báo càng lớn. Do đó chúng ta có xu hướng phạt nặng những trường hợp khó dự báo nếu chúng dự báo sai.

Ta nhận thấy rằng cross entropy là trường hợp đường cong kém cong nhất. Do đó nó sẽ không tốt bằng focal loss trong trường hợp mẫu mất cân bằng.

### Retina net (2)

Retina Net là một mô hình giải quyết được vấn đề mất cân bằng trong phân phối giữa foreground và background trong các bài toán one-stage detection bằng cách sử dụng hàm *focal loss* thay cho *cross entropy*.

Cấu trúc của Retinanet:



Hình 2.10 Kiến trúc của RetinaNet

Từ ảnh đầu vào sẽ đưa qua một mạng CNN đóng vai trò trích xuất đặc trưng. Tuy nhiên ở đây kết hợp thêm một mạng FPN (Feature Pyramid Network) để tạo ra một loạt các output dạng kim tự tháp như trên ảnh Mục đích của việc này là “…for detecting objects at different scales” nghĩa là nhận diện đối tượng ở các tỷ lệ scale khác nhau.

FPN sẽ gồm một nhánh đi lên (a) và nhánh đi xuống (b). Nhánh đi lên đóng vai trò trích xuất đặc trưng, sẽ giảm về kích thước nhưng tăng về ý nghĩa (giảm 2 lần sau mỗi lần downsampling). Một số mạng sau vài layer Conv thì dùng feature map (FM) cuối cùng để detection nên dễ bỏ qua các object nhỏ.

Do đó, ở đây FPN bổ sung thêm nhánh bên phải đi xuống để up sampling từ (FM) nhỏ nhất. Mỗi lần upsampling kích thước sẽ tăng lên gấp đôi. Và lúc này tuy là layer được tạo ra vẫn giữ ý nghĩa nhưng vị trí của object đã bị mất qua các bước down sampling và up sampling. Tác giả khắc phục bằng cách nối với một layer ở nhánh đi lên (layer cùng level) bằng phép cộng element-wise additional với hi vọng giữ lại được thông tin đó.

Sau khi cộng vào thì sẽ sinh ra một loạt các FM mới và Retinanet dùng các FM đó để đưa vào một 2 mạng phụ (subnet ) là class subnet và box subnet để predict các thông tin chúng ta cần là:

* Class subnet thì predict probality của các class trên từng điểm ảnh trên ảnh. Output của mạng này như trên hình là WxHxKA. Trong đó W, H là width và height của ảnh. KA là độ sâu = K\*A, K là số class và A là số anchorbox.
* Box subnet thì predict ra các box của vật thể tại từng điểm ảnh và output có độ sâu là 4xA.

## Tổng kết chương

Sau khi đi hết chương hai thì ta đã lược khảo qua một vài kiến thức cơ bản machine learning cũng như tìm hiểu sơ bộ về mô hình Retinanet. Trong những chương tiếp theo sẽ là quá trình từ chuẩn bị cho đến huấn luyện mô hình và kết quả.

# THỰC NGHIỆM VÀ ĐÁNH GIÁ KẾT QUẢ

## Giới thiệu chương

Trong chương này tôi sẽ nói về quá trình, phương pháp thu thập, xử lý dữ liệu và huấn luyện mô hình. Sau đó đánh giá kết quả đạt được.

## Thu thập dữ liệu

Đây là công việc tốn thời gian và chi phí nhất trong bất kì một dự án nhận dạng nào. Và trong bài nhận dạng biển báo giao thông này cũng không ngoại lệ.

Dữ liệu biển báo giao thông được thu thập thông qua nhiều cách và nhiều nguồn khác nhau như tìm kiếm qua chụp trực tiếp trên đường giao thông, tìm kiếm và chọn lọc từ các diễn đàn, các nguồn chia sẽ data trên Internet như Github, Google, các diễn đàn AI, facebook, các blog cá nhân của những tiền bối đi trước.



Hình 3.1 Ảnh thu thập thông qua chụp trên đường phố

Trong bài này, tôi lựa chọn sử dụng những bộ dữ liệu:

* Zalo\_traffic\_2020 từ blog Mì AI. (4)
* TrafficSign\_Dataset-master
* Dataset\_GTSDB\_format
* Chụp và tổng hợp từ một số con đường tại thành phố Đà Nẵng

## Tiền xử lý dữ liệu

Là quá trình quan trọng ảnh hưởng trực tiếp đến kết quả mô hình nhận diện. Mục đích nâng cao chất lượng ảnh đầu vào cũng như nâng cao chất lượng dataset để mô hình “học” được chính xác hơn.

### Tiền xử lý và gán nhãn cho ảnh

Ảnh sau khi được thu thập, chọn lọc sẽ được tiến hành xử lý để nâng cao chất lượng, sau đó sẽ được gán nhãn các đối tượng.

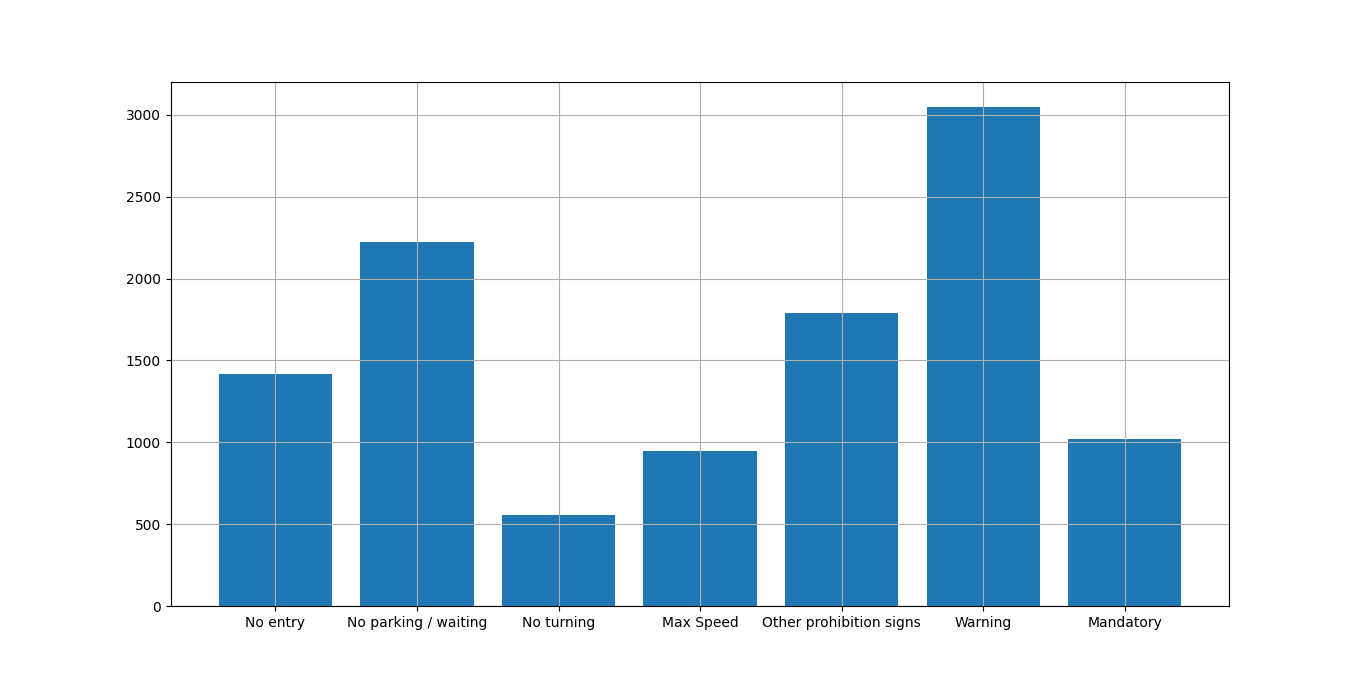
Quá trình gán nhãn sẽ được thực hiện bằng phần mềm LabelImg.

Tuy nhiên, đa số những dữ liệu thu thập được bằng cách sưu tầm đều được dán nhãn sẵn nên ta có thể dùng code để review và thay đổi nhãn lại theo định dạng mong muốn. Đồng thời, lưu lại với định dạng csv để có thể sử dụng với model retinanet.

Ví dụ: Bộ dữ liệu Zalo\_traffic\_2020 đã được dán nhãn sẵn và lưu ở dạng file json

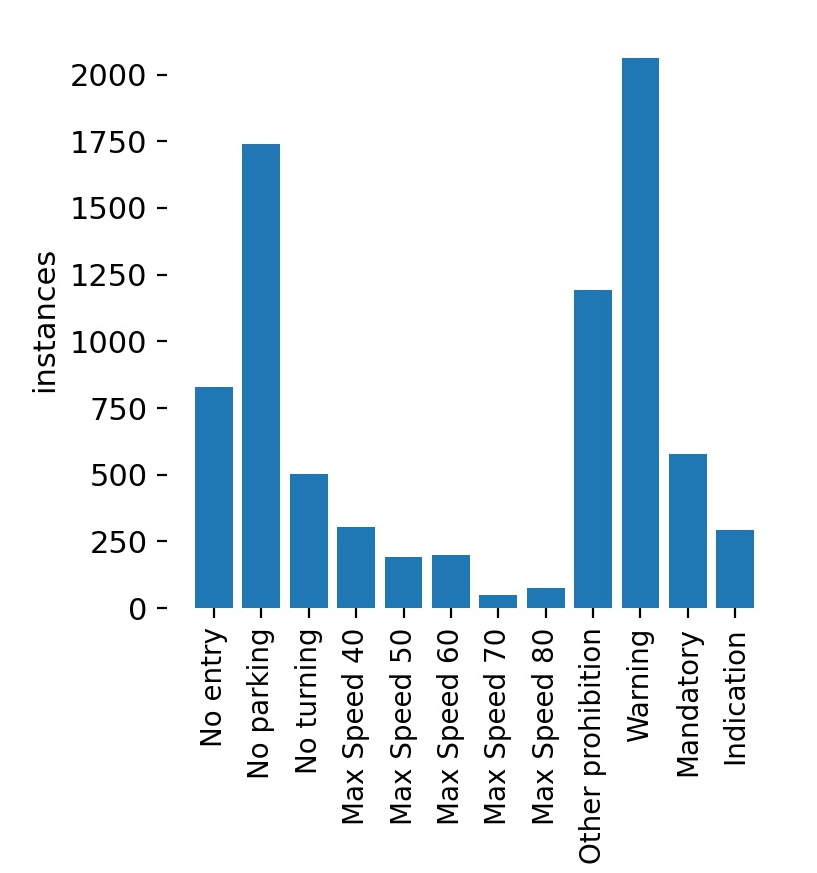
|  |  |
| --- | --- |
| No entry | 1416 |
| No parking / waiting | 2221 |
| No turning | 556 |
| Max Speed | 949 |
| Other prohibition signs | 1787 |
| Warning | 3049 |
| Mandatory | 1022 |

Bảng 3.1 Bộ dữ liệu Zalo\_traffic\_2020



Biểu đồ 3.1 Biểu đồ nhãn dữ liệu

* Sau khi tiến hành chọn lọc loại bỏ những đối tượng không rõ ràng, không đạt tiêu chuẩn và những đối tượng sai, không phù hợp với cấu hình biển báo của Việt Nam thì thu được tập 4920 ảnh bao gồm cả ảnh chụp và ảnh từ những bộ dữ liệu sưu tầm được.
* Tiến hành dán dãn những ảnh chụp và những ảnh từ những bộ chưa được dán nhãn sẵn. Kết quả thu được 10047 dãn gồm 12 đối tượng khác nhau:
  + 0:"No entry",
  + 1:"No parking",
  + 2:"No turning",
  + 3:"Max Speed 40",
  + 4:"Max Speed 50",
  + 5:"Max Speed 60",
  + 6:"Max Speed 70",
  + 7:"Max Speed 80",
  + 8:"Other prohibition",
  + 9:"Warning",
  + 10:"Mandatory",
  + 11:"Indication".
* Lưu tên 12 đối tượng này vào một file csv mới (classes.csv) để định nghĩa các đối tượng cho mô hình huấn luyện.



Biểu đồ 3.2 Mật độ các đối tượng xuất hiện trong tập train

### Chia dữ liệu thành tập train và validation

Sau khi xử lý và dán nhãn thì bộ dữ liệu đã có thể được sử dụng để huấn luyện mô hình, tuy nhiên ta cần chia ra thành các tập train, val để phục vụ cho quá trình huấn luyện, kiểm tra và đánh giá.

Ở đây tôi chia dữ liệu theo tỉ lệ 8-2. Tức là 80% cho tập train, 20% cho tập validation.

Sau khi chia ta sẽ thu được 2 file csv: train, validation lưu nhãn và annotation của các ảnh trong dữ liệu.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Số lượng ảnh | Số lượng dối tượng |  |
| Train set | 4010 | 8402 |
| Validation set | 910 | 1645 |

Bảng 3.2 Dữ liệu sau khi phân chia

## Chuẩn bị huấn luyện mô hình

Retinanet có rất nhiều source code được công khai phổ biến. Trong đồ án này tôi xin sử dụng source code dựng sẵn được đăng tải trên Github: <https://github.com/fizyr/keras-retinanet>

* Tiến hành tải source code về máy.
* Cài đặt các thư viện và package cần thiết:
  + Tensorflow 2.3.0
  + Keras 2.4.3
  + Keras-retinanet
* Biên dịch Cython code bằng lệnh: python setup.py build\_ext –inplace

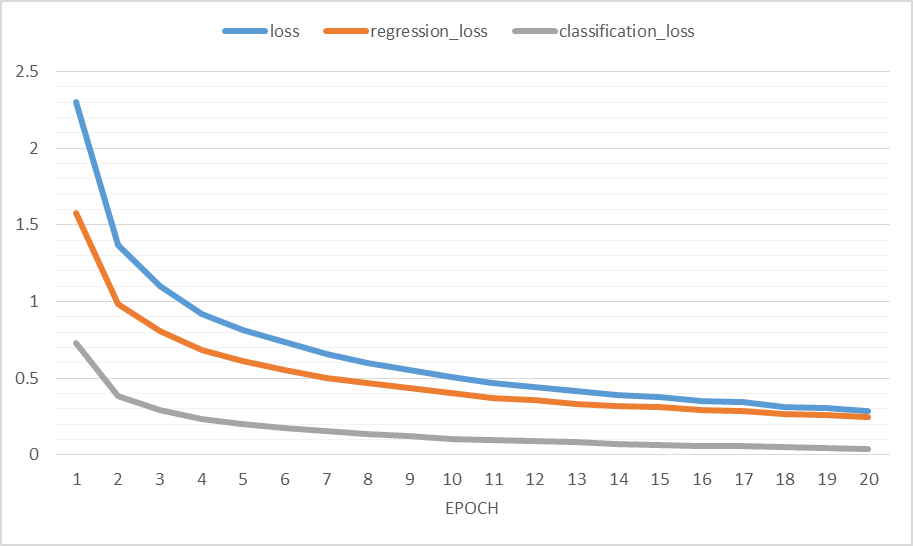
Hoàn tất quá trình chuẩn bị, ta có thể bắt đầu tiến hành huấn luyện mô hình.

## Huấn luyện mô hình

Chạy file train.py để bắt đầu huấn luyện với dữ liệu đầu vào là hai file csv chứa hai tập dữ liệu train và validation đã chuẩn bị ở chương thu thập dữ liệu.

## Kết quả

### Kết quả huấn luyện

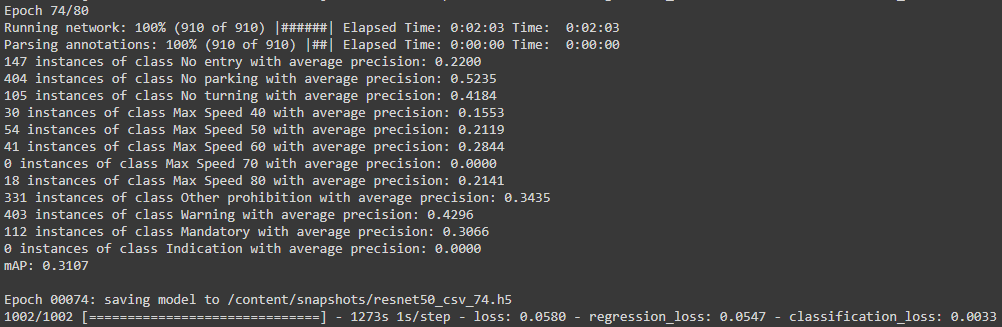


Biểu đồ 4.1 Kết quả 20 epoch đầu tiên

|  |  |  |  |
| --- | --- | --- | --- |
| epoch | loss | regression\_loss | classification\_loss |
| 1 | 2.3037 | 1.5745 | 0.7292 |
| 2 | 1.3684 | 0.9868 | 0.3816 |
| 3 | 1.0979 | 0.8087 | 0.2892 |
| 4 | 0.9199 | 0.6861 | 0.2338 |
| 5 | 0.8121 | 0.613 | 0.1991 |
| 6 | 0.7333 | 0.5561 | 0.1772 |
| 7 | 0.6574 | 0.5041 | 0.1533 |
| 8 | 0.5998 | 0.4657 | 0.1341 |
| 9 | 0.5562 | 0.4327 | 0.1235 |
| 10 | 0.508 | 0.4047 | 0.1033 |
| 11 | 0.4678 | 0.3728 | 0.095 |
| 12 | 0.4421 | 0.3547 | 0.0874 |
| 13 | 0.4172 | 0.3322 | 0.085 |
| 14 | 0.389 | 0.3193 | 0.0697 |
| 15 | 0.3763 | 0.3116 | 0.0647 |
| 16 | 0.3505 | 0.2951 | 0.0554 |
| 17 | 0.3437 | 0.2852 | 0.0585 |
| 18 | 0.3142 | 0.2664 | 0.0478 |
| 19 | 0.3021 | 0.2587 | 0.0434 |
| 20 | 0.2846 | 0.2468 | 0.0378 |

Bảng 4.1 Kết quả 20 epoch đầu tiên

Kết quả epoch 74:



Hình 4.1 Kết quả train ở epoch 74

Sau 74 epoch thì mAP chỉ giao động quanh mức 0.31 và loss gần giảm rất ít ở mỗi epoch (khoảng 0.005) nên tôi quyết định dừng lại và chạy thử, đánh giá mô hình.

### Kết quả nhận dạng trên ảnh thực



Hình 4.2 Kết quả nhận dạng trên ảnh thực



Hình 4.3 Kết quả nhận dạng trên ảnh thực



Hình 4.4 Kết quả nhận dạng trên ảnh thực

## Đánh giá kết quả

### Ưu điểm

* Khả năng phát hiện đối tượng và nhận dạng khá tốt, độ chính xác cao.
* Dù tập dữ liệu rất không cân bằng nhưng mô hình vẫn phân lớp đối tượng khá chính xác, đây là ưu điểm có được nhờ sử dụng hàm focal loss. Nhờ vào đặc điểm này, chi phí thu thập và xử lý dữ liệu sẽ giảm đáng kể.
* Khả năng ứng dụng thực tiễn cao, tiềm năng phát triển lớn. Có thể tích hợp vào nhiều hệ thống hỗ trợ lái xe như hỗ trợ lái tự động, cảnh báo vượt giới hạn tốc độ cho người lái xe,…

### Hạn chế

* Tốc độ xử lý chậm, cần có GPU để tối ưu tốc độ xử lý
* Do tập dữ liệu quá mất cân bằng (Class ít mẫu nhất chỉ có 59 mẫu trong tập train) dẫn đến khả năng phát hiện tương đối kém với một số class có số lượng mẫu thấp.
* Số lượng loại biển báo nhận dạng được còn hạn chế, hiện tại chỉ tập trung vào các loại biển báo cấm.

## Hướng phát triển

* Nâng cao độ chính xác
* Bổ sung thêm các loại biển báo khác vào mô hình, tăng lượng dữ liệu vào để nâng cao khả năng nhận dạng của mô hình.
* Tăng khả năng nhận dạng trong điều kiện ánh sáng yếu
* Tối ưu mô hình để tăng tốc độ xử lý, có thể ứng dụng được vào các hệ thống thực tế như camera hành trình, hệ thống tự động lái trên xe ô tô …

**TÀI LIỆU THAM KHẢO**

1. **Hải, Đỗ Minh.** Mạng nơ-ron nhân tạo - Neural Networks. *Hai's Blog.* [Trực tuyến] https://dominhhai.github.io/vi/2018/04/nn-intro/.

2. **Toàn, Nguyễn Thế.** Thuật toán CNN – Convolutional Neural Network. *Viblo.* [Trực tuyến] https://viblo.asia/.

3. *Focal Loss for Dense Object Detection.* **Tsung-Yi Lin; Priya Goyal; Ross Girshick; Kaiming He; Piotr Dollar.** 2018.

4. **Khánh, Phạm Đình.** Focal Loss trong RetinaNet. *phamdinhkhanh blog.* [Trực tuyến] 23 08 2020. https://phamdinhkhanh.github.io/2020/08/23/FocalLoss.html.

5. **Thắng, Nguyễn Chiến.** Xây dựng hệ thống nhận diện biển báo giao thông bằng Retinanet. *Mì AI.* [Trực tuyến] 1 4 2021. https://www.miai.vn/2021/04/01/xay-dung-he-thong-nhan-dien-bien-bao-giao-thong-bang-retinanet/.

6. **Humbarwadi, Srihari.** Object Detection with RetinaNet. *Keras.* [Trực tuyến] 24 07 2020. https://keras.io/examples/vision/retinanet/.