**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

**KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

**----🙣🕮🙡----**



**MACHINE LEARNING**

**ĐỀ TÀI**

**NHẬN DIỆN CÁC LOẠI PHƯƠNG TIỆN GIAO THÔNG**



**GVHD : TS. Vũ Quang Huy**

**SVTH : Lê Quang Sơn 15110118**

**Phạm Phú Thịnh 15110133**

**Đặng Công Đạt 15110030**

**Tp. Hồ Chí Minh, tháng 12 năm 2018**

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

Giáo viên hướng dẫn

LỜI CẢM ƠN

Kính gửi thầy Vũ Quang Huy !

Đây đã là năm thứ tư của thời sinh viên. Chúng em cảm nhận được sự khó khăn thời điểm này nhưng cũng rất hạnh phúc khi có Thầy hỗ trợ chúng em.

Em viết những dòng này để cảm ơn Thầy vì những nổ lực trong suốt học kỳ vừa qua. Những lời gợi ý, quan tâm không những giúp bài báo cáo này được hoàn thiện, mà còn hoàn thiện cả chính con người của chúng em. Là sinh viên, chúng em đã trưởng thành hơn rất nhiều để sẵn sàng cho những thách thức khi ra trường. Những bài học của Thầy luôn là hành trang vô giá cho con đường phía trước.

Một lần nữa, chúng em cảm ơn Thầy!

Sinh viên

Đặng Công Đạt

Lê Quang Sơn

Phạm Phú Thịnh

**MỤC LỤC**

[LỜI CẢM ƠN 2](#_Toc532852923)

[Chương I. Giới thiệu. 3](#_Toc532852924)

[1. Machine Learning là gì? 3](#_Toc532852925)

[2. Lí do chọn đề tài. 5](#_Toc532852926)

[3. Ý nghĩa đề tài. 5](#_Toc532852927)

[Chương II. Mục tiêu. 7](#_Toc532852928)

[Chương III. Quá trình thực hiện. 8](#_Toc532852929)

[1. Đặt vấn đề. 8](#_Toc532852930)

[2. Chuẩn bị dữ liệu. 8](#_Toc532852931)

[3. Chọn thuật toán. 8](#_Toc532852932)

[4. Huấn luyện. 8](#_Toc532852933)

[5. Kiểm thử. 8](#_Toc532852934)

[Chương IV. Cơ sở lý thuyết. 10](#_Toc532852935)

[1. Xử lý ảnh. 10](#_Toc532852936)

[2. Neural Networks. 10](#_Toc532852937)

[3. Convolutional Neural Networks – CNN: (Mạng nơ-ron tích chập). 12](#_Toc532852938)

[4. Thuật toán train. 20](#_Toc532852939)

[Chương V. Mô hình Block Diagram. 22](#_Toc532852940)

[Chương VI. Thống kê, kế hoạch, phân công công việc. 24](#_Toc532852941)

[Chương VII. Ứng dụng 24](#_Toc532852942)

[1. Thư viện keras 24](#_Toc532852943)

[a. Giới thiệu 24](#_Toc532852944)

[b. Một số chức năng lưu ý (áp dụng khi code) 24](#_Toc532852945)

[2. Áp dụng thuật toán vào ứng dụng 26](#_Toc532852946)

[3. Kết quả và đánh giá 34](#_Toc532852947)

[Chương VIII. Tổng kết 39](#_Toc532852948)

[TÀI LIỆU THAM KHẢO 40](#_Toc532852949)

**DANH MỤC HÌNH VẼ**

[**Hình 1:** Tổng quan về Machine Learning 5](#_Toc532855582)

[**Hình 2**: Kiến trúc của mạng neural networks. 11](#_Toc532855583)

[**Hình 3**:Quá trình xử lý thông tin của ANN 11](#_Toc532855584)

[**Hình 4:** Convolutional 13](#_Toc532855585)

[**Hình 5**: Input Neurons 15](#_Toc532855586)

[**Hình 6:**Neuron layer đầu tiên 15](#_Toc532855587)

[**Hình 7**:Neuron layer thứ 2 16](#_Toc532855588)

[**Hình 8**: Tạo ra các f.maps thông qua quá trinh Max Pooling 17](#_Toc532855589)

[**Hình 9**: Đưa Input đầu vào thành vector 1 chiều 17](#_Toc532855590)

[**Hình 10**: Stride = 1 18](#_Toc532855591)

[**Hình 11**: Stride = 2 18](#_Toc532855592)

[**Hình 12**: Padding = 1 và Stride = 1 19](#_Toc532855593)

[**Hình 13:** Max Pooling với Stride 2 19](#_Toc532855594)

[**Hình 14**: Fully Connected Layer (FC) 20](#_Toc532855595)

[**Hình 15**: Block Diagram 22](#_Toc532855596)

[**Hình 16**: Planning 24](#_Toc532855597)

[**Hình 17**: Model accuracy 32](#_Toc532855598)

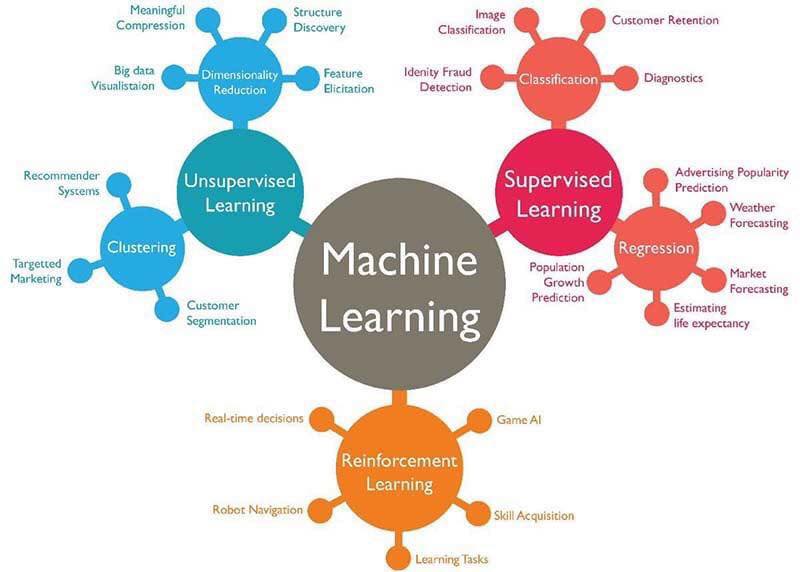
[**Hình 18**: Model loss 33](#_Toc532855599)

# Chương I. Giới thiệu.

1. **Machine Learning là gì?**

Máy học là một lĩnh vực của trí tuệ nhân tạo liên quan đến việc nghiên cứu và xây dựng các kĩ thuật cho phép các hệ thống "học" tự động từ dữ liệu để giải quyết những vấn đề cụ thể.

Ví dụ, nếu bạn cung cấp cho chương trình máy học các hình ảnh chụp X-quang cùng những triệu chứng tương ứng, nó sẽ có thể học từ đó và đưa ra các giải pháp hỗ trợ hoặc tự động phân tích những hình ảnh X-quang tương tự trong tương lai.



**Hình 1:** Tổng quan về Machine Learning

Về bản chất, máy học sẽ dựa trên các dữ liệu có sẵn để đưa ra phân tích, dự đoán trong tương lai.

Máy học có thể được ứng dụng để so sánh tất cả các hình ảnh khác nhau và tìm ra những hình ảnh có triệu chứng tương tự. Khi bạn cung cấp thêm một hình ảnh mới, nó sẽ so sánh nội dung này với các mẫu đã thu thập và cho bạn biết về những hình ảnh có khả năng chứa triệu chứng đã nghiên cứu trước đó. Đây được gọi là “Supervised Learning”, loại máy học này cần có thuật toán đào tạo để đưa ra dữ liệu.

Một loại máy học khác được gọi là “Unsupervised Learning”, chúng không cần thuật toán có sẵn và có thể tự "học" để tìm ra kết quả.

Loại máy học phổ biến thứ ba là "Reinforcement Learning (học củng cố)", dựa trên việc cung cấp cho máy học thuật toán cùng một bộ quy tắc với các ràng buộc cụ thể, sau đó nó sẽ tự tìm cách đạt được mục tiêu tốt nhất theo các yêu cầu.

Máy học củng cố thường liên quan đến việc tìm một mục tiêu, phần thưởng như ghi điểm trong một trò chơi hoặc giảm lượng tiêu thụ điện cho một cơ sở. Thuật toán sẽ tự động điều chỉnh để tối đa hóa mục tiêu đạt được theo các ràng buộc được cung cấp. Học củng cố được biết nhiều thông qua việc dạy AI chơi các trò chơi máy tính như CS-Go, StarCraft, Poker, Dota.

Về bản chất, máy học sẽ nghiên cứu dữ liệu để phân loại thông tin hoặc dự đoán các trường hợp có thể xảy ra trong tương lai.

Trí tuệ nhân tạo mang một phạm vi ý nghĩa rất rộng. “Cho đến gần đây, chúng tôi tin rằng trí tuệ nhân tạo là cách ứng dụng khoa học và kỹ thuật làm cho máy tính có thể hoạt động theo cách giống như trí thông minh của con người”, Andrew Moore, Trưởng khoa Khoa học Máy tính tại Đại học Carnegie Mellon nhận định.

1. **Lí do chọn đề tài.**

Công nghệ nhận dạng hình ảnh có tiềm năng lớn trong việc áp dụng rộng rãi trong các ngành công nghiệp khác nhau. Các công ty lớn như Tesla, Google, Uber, Adobe Systems…, cũng sử dụng công nghệ nhận dạng hình ảnh.

Số lượng phương tiện giao thông ngày càng tăng cao áp dụng công nghệ nhận dạng bằng hình ảnh vào nhận diện xe hơi giúp con người sử dụng hệ thống có thể phân loại các phương tiện một cách dễ dàng.

Ngoài nhận diện các phương tiện giao thông chúng ta có thể áp dụng công nghệ nhận dạng hình ảnh vào nhiều lĩnh vực khác nhau…

1. **Ý nghĩa đề tài.**

Số lượng người dùng xe hơi ngày càng tăng hệ thống có thể phân loại được từng loại xe bên cạnh đó hệ thống phát hiện phân loại các đối tượng vi phạm, phạm tội chẳng hạn như xe chạy quá tốc độ, xe vượt đèn đỏ....

Cùng với sự ra đời và phát triển của rất nhiều phần mềm, ứng dụng hiện đại, vấn đề về nghiên cứu, thiết kế và đưa **Xe tự lái** vào trong cuộc sống đang là một ý tưởng đầy triển vọng và tương đối khả thi trong tương lai gần.

Áp dụng Machine Learning & AI vào trong phân loại các phương tiện giao thông giúp cho việc quản lý các phương tiện dễ dàng hơn. Điều này sẽ tiết kiệm công sức, thời gian và chi phí rất nhiều.

Có thể áp dụng việc phân loại phương tiện quan sát tình hình giao thông và phát hiện phương tiện vi phạm.

# Chương II. Mục tiêu.

Đáp ứng được yêu cầu đề ra của giáo viên và người hướng dẫn.

Các thành viên trong nhóm hiểu được thuật toán mà nhóm dự định áp dụng.

Nhận biết được các phương tiện giao thông thông qua hình ảnh.

Hiểu thêm về những kiến thức và công nghệ mới.

Kết quả đầu ra có độ chính xác trên 80%.

Có thể tận dụng vào trong thực tế.

# Chương III. Quá trình thực hiện.

1. **Đặt vấn đề.**

Các thành viên trong nhóm tìm hiểu các khái niệm cơ bản, dựa vào thực trạng và nhu cầu sử dụng cũng như ý nghĩa của đề tài mà nhóm đã chọn.

Sau khi hoàn thành việc tìm hiểu, các thành viên trong nhóm đã phân tích các vấn đề thực tế có thể áp dụng cho đề tài nhóm đã chọn. Từ đó, hội ý và đưa ra giải pháp cuối cùng để thực hiện đề tài để có thể đạt được kết quả tốt nhất.

1. **Chuẩn bị dữ liệu.**

Tìm nguồn cung cấp dữ liệu (Internet, …).

Phân loại dữ liệu (hai loại).

* Loại dữ liệu gồm những hình ảnh có chứa các phương tiện giao thông.
* Loại dữ liệu gồm những hình ảnh không chứa các loại phương tiện giao thông.

Phân tích dữ liệu. Xem dữ liệu đó thuộc loại dữ liệu nào, có phương tiện giao thông nào trong đó hay là không.

Chuyển đổi dữ liệu dạng hình, nhãn thành dạng dữ liệu matrix theo dạng vector.

1. **Chọn thuật toán.**

Nghiên cứu và tìm hiểu các thuật toán cơ bản và phổ biến trong Machine Learing.

Xác định thuật toán phù hợp với đề tài mà nhóm đã chọn.

Áp dụng thuật toán đã chọn vào đề tài mà nhóm đã chọn.

1. **Huấn luyện.**

Từ dữ liệu đã xử lí trước, thông qua việc áp dụng thuật toán để tạo ra 1 bộ model để từ đó so sánh với dữ liệu test và cho ra kết quả.

1. **Kiểm thử.**

Input hình cần kiểm tra sau khi dự đoán sẽ cho ra kết quả với số liệu về độ chính xác (theo %).

Tiếp tục cập nhật và hoàn thiện model.

Kết quả mong muốn trong quá trình kiểm thử là output cho ra kết quả với độ chính xác > 80%.

# Chương IV. Cơ sở lý thuyết.

1. **Xử lý ảnh.**
2. Đặc trưng của ảnh (Feature)

Đặc trưng ảnh là những chi tiết xuất hiện trong ảnh, từ đơn giản như cạnh, hình khối, chữ viết tới phức tạp như mắt, mặt, chó, mèo, bàn, ghế, xe, đèn giao thông...

1. Bộ lọc ảnh:

Là bộ lọc giúp phát hiện và trích xuất các đặc trưng của ảnh, có thể là bộ lọc góc, cạnh, đường chéo, hình tròn, hình vuông, v.v.

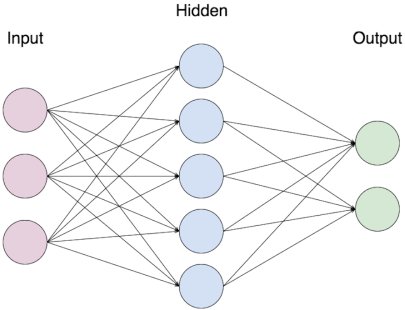
1. Xử lý ảnh:

Trong bài toán này thì các bức ảnh là hình ảnh của các loại phương tiện giao thông được chụp. Mỗi hình ảnh có kích thước khác nhau nên muốn xử lý những hình ảnh đó cần trải qua các bước tách các đặt trưng của ảnh để tiện cho việc xử lý. Bước này được gọi là bước tiền xử lý.

1. **Neural Networks.**
2. **Tổng quan.**

Mạng nơron nhân tạo, Artificial Neural Network (ANN) là một mô hình xử lý thông tin phỏng theo cách thức xử lý thông tin của các hệ nơron sinh học. Nó đƣợc tạo nên từ một số lƣợng lớn các phần tử (nơron) kết nối với nhau thông qua các liên kết (trọng số liên kết) làm việc nhƣ một thể thống nhất để giải quyết một vấn đề cụ thể nào đó. Một mạng nơron nhân tạo đƣợc cấu hình cho một ứng dụng cụ thể (nhận dạng mẫu, phân loại dữ liệu,...) thông qua một quá trình học từ tập các mẫu huấn luyện. Về bản chất học chính là quá trình hiệu chỉnh trọng số liên kết giữa các nơron.

1. **Kiến trúc của neural networks.**

****

**Hình 2**: Kiến trúc của mạng neural networks.

Kiến trúc chung của một neural networks gồm 3 thành phần đó là Input Layer, Hidden Layer và Output Layer.

Hidden Layer gồm các Neuron, nhận dữ liệu input từ các Nueron ở lớp (Layer) trước đó và chuyển đổi các input này cho các lớp xử lý tiếp theo. Trong một ANN có thể có nhiều Hidden Layer.

1. **Quá trình xử lý thông tin.**



**Hình 3**:Quá trình xử lý thông tin của ANN

Inputs: Mỗi Input tương ứng với 1 thuộc tính (attribute) của dữ liệu (patterns). Ví dụ như trong ứng dụng của ngân hàng xem xét có chấp nhận cho khách hàng vay tiền hay không thì mỗi Input là một thuộc tính của khách hàng như thu nhập, nghề nghiệp, tuổi, số con,…

Output: Kết quả của một ANN là một giải pháp cho một vấn đề, ví dụ như với bài toán xem xét chấp nhận cho khách hàng vay tiền hay không thì output là yes (cho vay) hoặc no (không cho vay).

Connection Weights (Trọng số liên kết): Đây là thành phần rất quan trọng của một ANN, nó thể hiện mức độ quan trọng (độ mạnh) của dữ liệu đầu vào đối với quá trình xử lý thông tin (quá trình chuyển đổi dữ liệu từ Layer này sang layer khác). Quá trình học (Learning Processing) của ANN thực ra là quá trình điều chỉnh các trọng số (Weight) của các input data để có được kết quả mong muốn.

Bộ tổng (Summing function): Thường dùng để tính tổng của tích các đầu vào với trọng số liên kết của nó**.**

Ngưỡng (còn gọi là một độ lệch - bias): Ngưỡng này thường được đưa vào như một thành phần của hàm truyền.

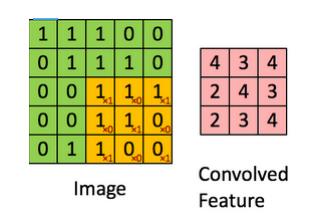
Hàm truyền (Transfer function): Hàm này được dùng để giới hạn phạm vi đầu ra của mỗi nơron. Nó nhận đầu vào là kết quả của hàm tổng và ngưỡng.

1. **Convolutional Neural Networks – CNN: (Mạng nơ-ron tích chập).**
2. **Tổng quan.**

CNN là mạng nơ-ron phổ biến nhất được dùng cho dữ liệu ảnh. Bên cạnh các lớp liên kết đầy đủ (FC layers), CNN còn đi cùng với các lớp ẩn đặc biệc giúp phát hiện và trích xuất những đặc trưng - chi tiết (patterns) xuất hiện trong ảnh gọi là Lớp Tích chập (Convolutional Layers). Chính những lớp tích chập này làm CNN trở nên khác biệt so với mạng nơ-ron truyền thống và hoạt động cực kỳ hiệu quả trong bài toán phân tích ảnh.

1. **Convolutional là gì?**

Là một quá trình mà một cửa sổ trượt (Sliding Windows) trên một ma trận.



**Hình 4:** Convolutional

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.

1. **Các thành phần cơ bản của mạng CNN.**

Convolution Layer là lớp quan trọng nhất trong cấu trúc mạng CNN.

Conv dựa trên lý thuyết xử lý tín hiệu số, việc lấy tích chập sẽ giúp trích xuất được những thông tin quan trọng từ dữ liệu.

Giống như các lớp ẩn khác, lớp tích chập lấy dữ liệu đầu vào, thực hiện các phép chuyển đổi để tạo ra dữ liệu đầu vào cho lớp kế tiếp (đầu ra của lớp này là đầu vào của lớp sau). Phép biến đổi được sử dụng là phép tính tích chập. Mỗi lớp tích chập chứa một hoặc nhiều bộ lọc - bộ phát hiện đặc trưng (filter - feature detector) cho phép phát hiện và trích xuất những đặc trưng khác nhau của ảnh.

Đặc trưng của ảnh là gì? Đặc trưng ảnh là những chi tiết xuất hiện trong ảnh, từ đơn giản như cạnh, hình khối, tới phức tạp như mắt, mặt, chó, mèo, bàn, ghế, xe, v.v.. Bộ lọc phát hiện đặc trưng là bộ lọc giúp phát hiện và trích xuất các đặc trừng của ảnh, có thể là bộ lọc góc, cạnh, đường chéo, hình tròn, hình vuông, v.v.

1. **Cấu trúc mạng CNN và cách hoạt động.**

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.

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. Nghĩa là mỗi nơ-ron ở layer tiếp theo sinh ra từ filter áp đặt lên một vùng ảnh cục bộ của nơ-ron layer trước đó.

Trong mô hình CNN có 2 khía cạnh cần quan tâm là tính bất biến (Location Invariance) và tính kết hợp (Compositionality):

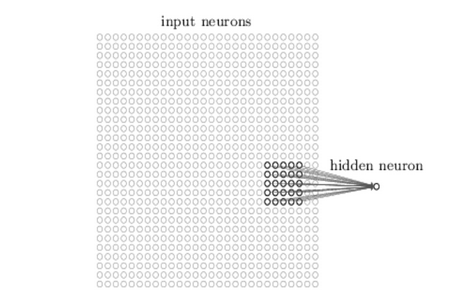
* Tính bất biến: Với cùng một đối tượng, nếu đối tượng này được chiếu theo các gốc độ khác nhau (translation, rotation, scaling) thì độ chính xác của thuật toán sẽ bị ảnh hưởng đáng kể. Pooling layer sẽ cho bạn tính bất biến đối với phép dịch chuyển (translation), phép quay (rotation) và phép co giãn (scaling).
* Tính kết hợp: 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.

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).

Các trường tiếp nhận cục bộ (local receptive field)

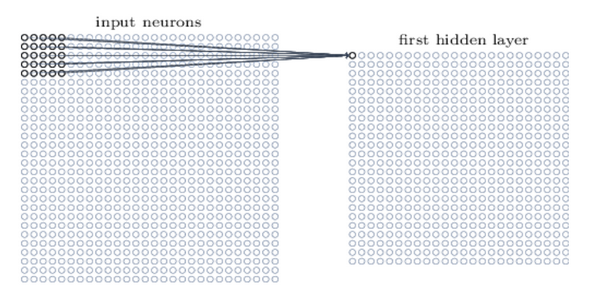
* Đầu vào của mạng CNN là một ảnh. Ví dụ như ảnh có kích thước 28x28 thì tương ứng đầu vào là một ma trận có 28x28 và giá trị mỗi điểm ảnh là một ô trong ma trận. Trong mô hình mạng ANN truyền thống thì chúng ta sẽ kết nối các neuron đầu vào vào tầng ảnh. Tuy nhiên trong CNN chúng ta không làm như vậy mà chúng ta chỉ kết nối trong một vùng nhỏ của các neuron đầu vào như một filter có kích thước 5x5 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 5x5 đấy gọi là một trường tiếp nhận cục bộ.



**Hình 5**: Input Neurons

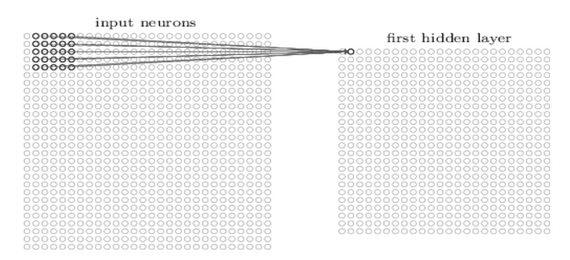
Các bước tạo ra hidden layer.

* **Tạo ra neuron ẩn đầu tiên trong lớp ẩn 1.**



**Hình 6:**Neuron layer đầu tiên

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



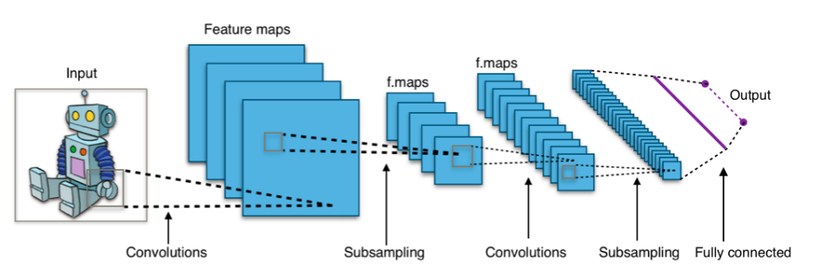
**Hình 7**:Neuron layer thứ 2

Trọng số chia sẽ (shared weights)

* 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. Vậy mối quan hệ giữa số lượng Feature map với số lượng tham số là gì?
* Chúng ta thấy mỗi fearture map cần 25 = 5x5 shared weight và 1 shared bias. Như vậy mỗi feature map cần 5x5+1 = 26 tham số. Như vậy nếu có 10 feature map thì có 10x26 = 260 tham số. Chúng ta xét lại nếu layer đầu tiên có kết nối đầy đủ nghĩa là chúng ta có 28x28=784 neuron đầu vào như vậy ta chỉ có 30 neuron ẩn. Như vậy ta cần 28x28x30 shared weight và 30 shared bias. Tổng số tham số là 28x28x30+30 tham số lớn hơn nhiều so với CNN.
* 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.

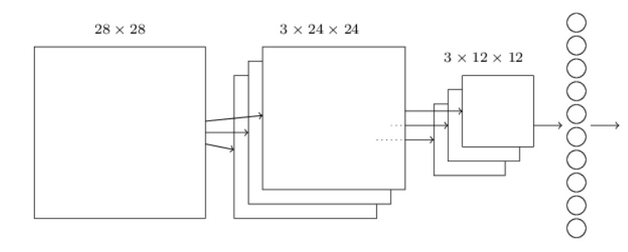
Tổng hợp (pooling)

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



**Hình 8**: Tạo ra các f.maps thông qua quá trinh Max Pooling

* 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.



**Hình 9**: Đưa Input đầu vào thành vector 1 chiều

* Hai 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.

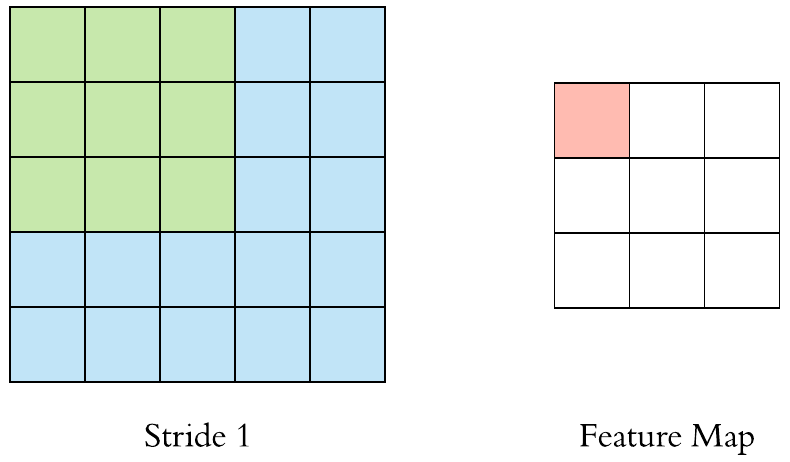
1. **Một số đặc điểm trong CNN.**

**Sliding Window:**

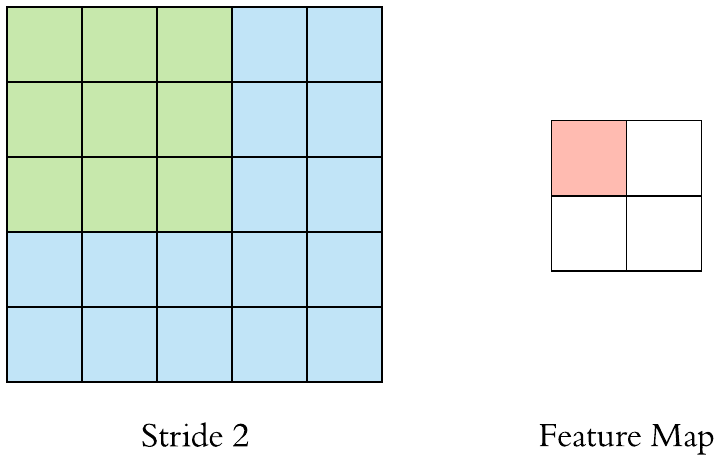
Một ma trận có kích thước nhỏ thường có kích thước là 3x3 hoặc 5x5.

**Stride:**

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



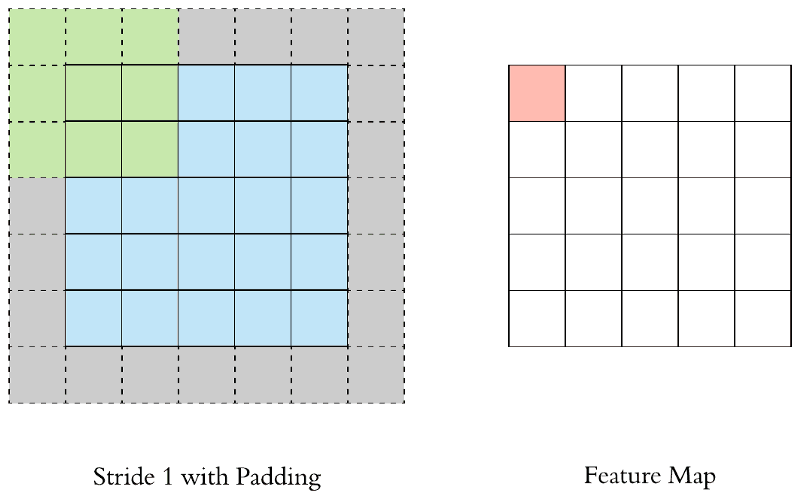
**Hình 10**: Stride = 1



**Hình 11**: Stride = 2

**Padding:**

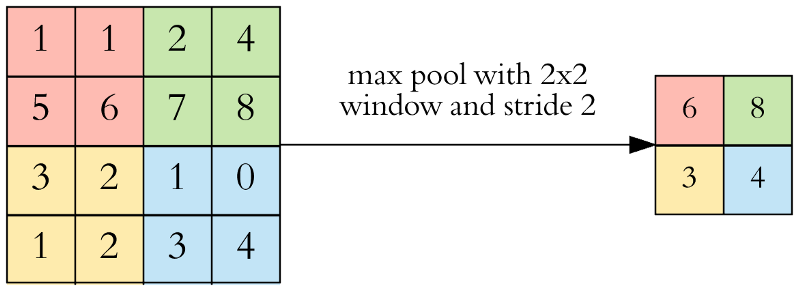
Khi ta điều chỉnh padding = 1, tức là ta đã thêm 1 ô bọc xung quanh các cạnh của input, muốn phần bọc này càng dày thì ta cần phải tăng padding lên. Hãy nhìn vào ví dụ sau, ta xét padding = 1.



**Hình 12**: Padding = 1 và Stride = 1

**Pooling:**

Mục đích của pooling rất đơn giản, nó làm giảm số hyperparameter mà ta cần phải tính toán, từ đó giảm thời gian tính toán, tránh overfitting. Loại pooling ta thường gặp nhất là max pooling, lấy giá trị lớn nhất trong một pooling window. Pooling hoạt động gần giống với convolution, nó cũng có 1 cửa sổ trượt gọi là pooling window, cửa sổ này trượt qua từng giá trị của ma trận dữ liệu đầu vào (thường là các feature map trong convolutional layer), chọn ra một giá trị từ các gía trị nằm trong cửa sổ trượt (với max pooling ta sẽ lấy giá trị lớn nhất). Chọn pooling window có kích thước là 2 \* 2, stride = 2 để đảm bảo không trùng nhau, và áp dụng max pooling:

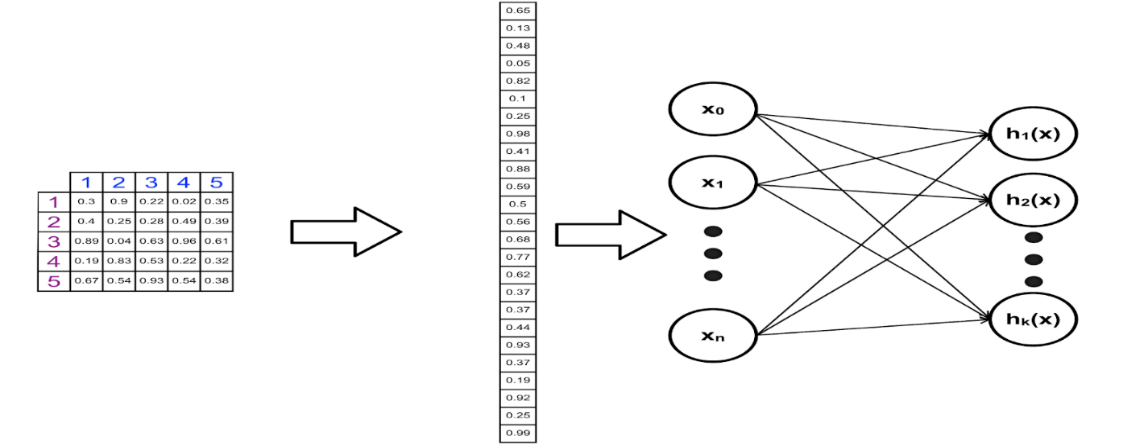


**Hình 13:** Max Pooling với Stride 2

Thông thường max pooling có kích thước là 2 và stride=2. Nếu lấy giá trị quá lớn, thay vì giảm tính toán nó lại làm phá vỡ cấu trúc ảnh và mất mát thông tin nghiêm trọng. Vì vậy mà một số chuyên gia không thích sử dụng layer này mà thay vào đó sử dụng thêm các lớp Conv Layer và tăng số stride lên mỗi lần.

Activation function: Đây là hàm nhận vector đầu vào, sau đó biến đổi để trả về vector đầu ra. Activation function cho phép ghi nhận được kết quả của dạng linear và nonlinear functions.

*Fully Connected Layer (FC).*



**Hình 14**: Fully Connected Layer (FC)

Tên tiếng viết là Mạng liên kết đầy đủ. Tại lớp mạng này, mỗi một nơ-ron của layer này sẽ liên kết tới mọi nơ-ron của lớp khác. Để đưa ảnh từ các layer trước vào mạng này, buộc phải dàn phẳng bức ảnh ra thành 1 vector thay vì là mảng nhiều chiều như trước. Tại layer cuối cùng sẽ sử dụng 1 hàm kinh điển trong học máy mà bất kì ai cũng từng sử dụng đó là softmax để phân loại đối tượng dựa vào vector đặc trưng đã được tính toán của các lớp trước đó.

Layer này cũng chính là 1 fully connected ANN. Thường thì sau các lớp Conv+Pooling thì sẽ là 2 lớp Fully connected, 1 layer để tập hợp các feature layer mà ta đã tìm ra, chuyển đổi dữ liệu từ 3D, hoặc 2D thành 1D, tức chỉ còn là 1 vector. Còn 1 layer nữa là output, số neuron của layer này phụ thuộc vào số output mà ta muốn tìm ra. Giả sử với tập dữ liêu MNIST chẳng hạn, ta có tập các số viết tay từ 0 -> 9. Vậy output sẽ có số neuron là 10.

1. **Thuật toán train.**
2. **Vấn đề.**

Sau khi có một tập các ảnh dữ liệu training là ảnh có chứa các phương tiện giao thông hoặc không có phương tiện giao thông, nhóm phải áp dụng thuật toán mà nhóm đã chọn (CNN) để tạo ra một Model cho bài toán từ tập dữ liệu vector để biết được ảnh nào (bộ vector nào) là phương tiện giao thông nào (xe đạp, xe máy, xe hơi…), và ảnh nào (bộ vector nào) là không có chứa loại phương tiên nào trong đó.

1. **Hướng giải quyết.**

Sau khi chuyển tất cả các ảnh về dạng vector ảnh số, ta phải tìm ra quy luật phân biệt của các vector ảnh, sau đó dùng thuật toán để phân chia các ảnh về 2 phía, dựa theo label ta sẽ biết được ảnh nào là có phương tiện giao thông và ảnh nào là không có chứa phương tiện giao thông.

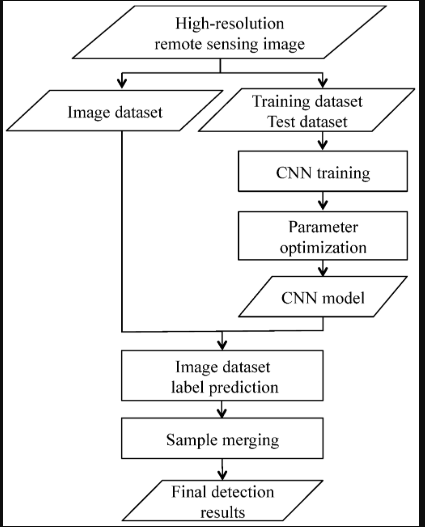
Cách giải quyết là phải tìm được một siêu phẳng để phân chia 2 phần có phương tiện giao thông và không có phương tiện giao thông sao cho công bằng nhất.

Giả sử ban đầu ta có bộ dữ liệu training với label có phương tiện giao thông và không có phương tiện giao thông có dạng sau: , ).

Trong đó:

* : Là là vector ảnh số được kéo ra từ ảnh tìm kiếm được.
* : Là label với 2 giá trị là có phương tiện giao thông / không có phương tiện giao thông. (1/-1: Tùy theo cách đặt nhãn)
* : sẽ là mặt phẳng phân chia giữa 2 lớp có phương tiện giao thông và không có phương tiện giao thông.
* Vậy với cặp dữ liệu thì khoảng cách từ điểm đó đến mặt phẳng phân chia là: .
* Với trong đó d là số chiều không gian trên vector ảnh số.
* Nếu ta tìm được min(|(1)|) thì ta sẽ tìm được khoảng gần nhất từ một điểm trong 2 lớp đến siêu phẳng (margin).
* Để tối ưu hóa bài toán này ta phải tìm được **w** và b sao cho margin là lớn nhất.

# Chương V. Mô hình Block Diagram.

****

**Hình 15**: Block Diagram

Lưu đồ của phương pháp được đề xuất được hiển thị trong hình bên trên.

Chúng ta đã sử dụng một số mẫu train được thu thập được từ các nguồn khác nhau (internet …) để huấn luyện chúng qua thuật toán CNN (convolutional neural network hay là mạng nơ ron tích chập) và tính toán độ chính xác phân loại dựa trên một số mẫu thử nghiệm được thu thập độc lập với mẫu training. Dựa trên các tham số chính của CNN.

Ví dụ: số lượng các node trong lớp chập thứ nhất, số lượng các node trong lớp chập thứ hai và số lượng đơn vị ẩn trong lớp được kết nối đầy đủ (fully connected)) được điều chỉnh liên tục cho đến khi tìm thấy sự kết hợp tốt nhất của các thông số trong đó độ chính xác tổng thể là cao nhất trên các bộ dữ liệu dùng để training. Bằng cách điều chỉnh các tham số, chúng tôi đã đạt được mô hình CNN tốt nhất và lưu nó để sử dụng tiếp có thể áp dụng model đó để test dữ liệu thực tế.

Bộ dữ liệu hình ảnh để phát hiện có phương tiện giao thông trong bức ảnh hay không được thu thập thông qua kỹ thuật cửa sổ trượt (kích thước cửa sổ là 17 × 17 và bước trượt là 3 pixel).

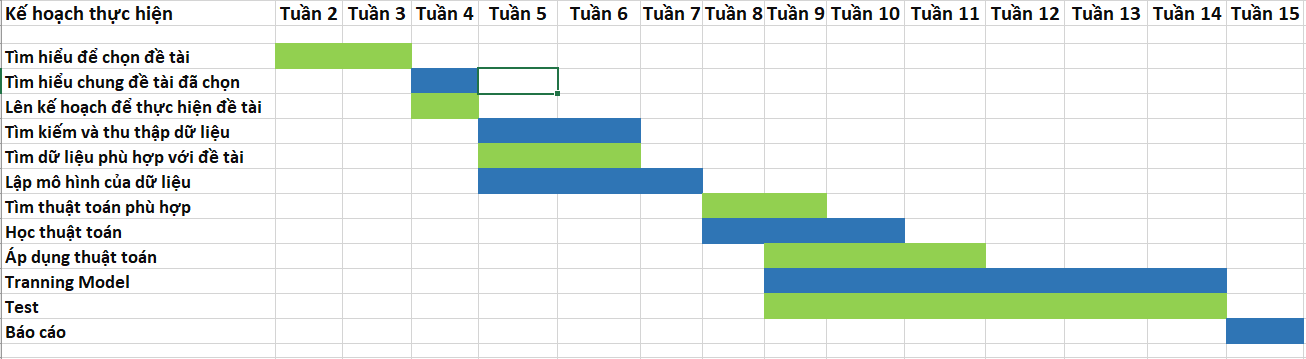
Đã sử dụng mô hình CNN tốt nhất thu được trước đó để dự đoán nhãn cho từng mẫu trong bộ dữ liệu hình ảnh.

Đối với tất cả các mẫu được dự đoán là lớp có phương tiện giao thông, chúng tôi đã hợp nhất các tọa độ tương ứng với cùng một mẫu có phương tiện giao thông trong đó (khoảng cách không gian nhỏ hơn một ngưỡng nhất định) vào một tọa độ và thu được kết quả phát hiện phương tiện giao thông cuối cùng.

Thực hiện các bước sau:

* Bước 1: Từ hình ảnh gốc ban đầu tìm kiếm được (ảnh chứa các phương tiện giao thông, không chứa phương tiện giao thông).
* Bước 2: Tạo ra một dataset chứa các hình ảnh tìm kiếm được.
* Bước 3: Dataset sẽ chia ra làm hai bộ dữ liệu. Một bộ dữ liệu dùng cho việc training và bộ dữ liệu còn lại dùng để test.
* Bước 4: Tiếp theo chúng ta sẽ sử dụng bộ dữ liệu training cho bộ dữ liệu này training qua thuật toán CNN (Convolutional Neural Network).
* Bước 5: Kế tiếp chúng ta sẽ tối ưu hóa các tham số sử dụng trong thuật toán.
* Bước 6: Sau khi training qua thuật toán CNN xong chúng ta sẽ nhận được một model.
* Bước 7: Tiếp theo chúng ta sẽ đưa ra các dự đoán cho các nhãn.
* Bước 8: Kết hợp các dữ kiện lại với nhau.
* Bước 9: Cuối cùng cho ra kết quả (xem xét kết quả có đạt được mục tiêu mà nhóm đề ra không nếu không thì quay lại B4).

# Chương VI. Thống kê, kế hoạch, phân công công việc.



**Hình 16**: Planning

# Chương VII. Ứng dụng

## **Thư viện keras**

### **Giới thiệu**

Keras là một neural networks API cấp cao, được viết bằng Python và có khả năng chạy trên TensorFlow, CNTK hoặc Theano. Nó được phát triển với trọng tâm là cho phép thử nghiệm nhanh. Có thể đi từ ý tưởng đến kết quả với sự chậm trễ ít nhất có thể là chìa khóa để thực hiện nghiên cứu tốt.

Trang chủ: <https://keras.io/>

* 1. **Một số chức năng lưu ý (áp dụng khi code)**

**Sequential model methods**

**Compile** : cấu hình model để training

compile(optimizer, loss=**None**, metrics=**None** )

Optimizer :   đều được xây dựng dựa trên thuật toán Gradient Descent (vd: SGD, RMSprop, Adam, Adamax).

* Loss: hàm loss function để áp dụng cho quá trình training, mean squared error với bài toán regression, binary crossentropy cho bài toán classification, categorical crossentropy với bài toán multiclass-classification và sparse categorical crossentropy cũng cho bài toán multiclass-classification nhưng với label chưa được đưa về dạng one-hot.
* Metrics: Danh sách các số liệu được mô hình đánh giá trong quá trình training và kiểm tra, thường dùng metrics=['accuracy'].

**Fit:** training model với số lần epoch (các lần lặp trên tâp dữ liệu)

fit(x=**None**, y=**None**, batch\_size=**None**, epochs=1, verbose=1, validation\_data=**None**)

* X: Mảng dữ liệu training
* Y: Mảng label của dữ liệu training
* Batch\_size: Số lượng samples cho mỗi cập nhật gradient
* Epochs: Số lượng epochs để training model. Epochs là một lần lặp trên toàn bộ dữ liệu x và y được cung cấp
* Verbose: là một số nguyên gồm 0, 1, hoặc 2. Verbosity mode. 0 = silent, 1 = progress bar, 2 = one line per epoch
* Validation\_data: đánh giá model ở giai đoạn cuối của mỗi epochs.

Samples được đánh giá phải khác samples training (x\_valuation, y\_valuation)

**Evaluate:** Trả về giá trị mất mát (loss value) và giá trị số liệu cho mô hình ở giai đoạn đánh giá model

evaluate(x=**None**, y=**None**)

* X: Mảng dữ liệu để đánh giá (khác dữ liệu training)
* Y: Mảng label dữ liệu để đánh giá

**Predict :** Tạo dự đoán đầu ra cho các input samples.

predict(x)

* X: Dữ liệu để predict

**Convolutional Layers:**

Conv2D: 2D convolution layer

keras.layers.Conv2D(filters, kernel\_size, data\_format=**None**, activation=**None**, use\_bias=**True**)

Lớp này tạo ra a kernel convolution được kết hợp với the layer input trên một kích thước không gian duy nhất để tạo ra một tensor đầu ra.

* Filters: kích thước của không gian đầu ra
* Kernel\_size; chỉ định chiều cao và chiều rộng của 2D convolution window
* Data\_format: kích thước trong các đầu vào
* Activation: Activation function để sử dụng

**Available activations (Một số activation dùng trong projects)**

**Softmax:**

|  |
| --- |
| keras.activations.softmax(x, axis=-1) |

* **x**: Input tensor.
* **axis**: là số nguyên, trục dọc theo đó chuẩn hóa softmax được áp dụng

**relu:**

|  |
| --- |
| keras.activations.relu(x, alpha=0.0, max\_value=None, threshold=0.0) |

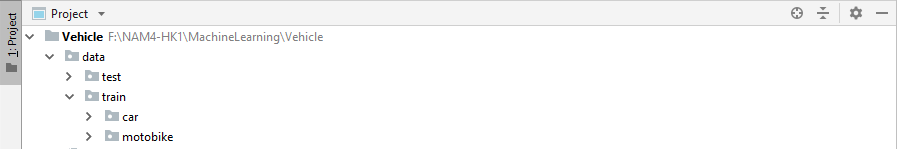
* **x**: Input tensor.
* **alpha**: là kiểu float. Độ dốc của phần tiêu cực. Mặc định là 0
* **max\_value**: là kiểu float. Ngưỡng bão hòa
* **threshold**: float. Threshold value for thresholded activation.

1. **Áp dụng thuật toán vào ứng dụng**

**Data:**

Chuẩn bị 2 tập dữ liệu gồm train và test.

Trong tập dữ liệu có 2 lable: car và motobike.



**Trainning model:**

Import các dữ liệu cần thiết:

|  |
| --- |
| *# Import libraries* **from** keras.models **import** Sequential **from** keras.layers **import** Conv2D **from** keras.layers **import** MaxPooling2D **from** keras.layers **import** GlobalAveragePooling2D **from** keras.layers **import** Dense  **from** keras.layers **import** Dropout **from** keras.layers **import** Flatten  **import** matplotlib.pyplot **as** plt |

Chúng ta khai báo các layers cần thiết để dùng cho thuật toán CNN:

Khởi tạo model bằng Sequential sau đó dùng method add để thêm các layer.

Conv2D: (Convolutional Layers: chứa các layer trong mạng nơ ron tích chập) là convolution layer dùng để lấy feature từ image.

MaxPool2D: (Pooling Layers : Chứa các layer dùng trong mạng CNN.) dùng để lấy feature nổi bật(dùng max) và giúp giảm parameter khi training.

Dense: layer này sử dụng như một layer neural network bình thường.

Flatten: dùng để lát phẳng layer để fully connection.

Dropout: dùng để hạn chế overfiting.

Khởi tạo Model:

Chúng ta khởi tạo model bằng Sequential sau đó dùng method add để thêm các layer.

|  |
| --- |
| # Initalize CNN  classifier = Sequential() |

Dùng convolution layer (Conv2D) dùng để lấy feature từ image:

|  |
| --- |
| *# Add 2 more convolution layers*  classifier.add(Conv2D(filters=32, kernel\_size=(3,3), input\_shape=(64, 64, 3), activation=**'relu'**)) |

Trong đó:

* *filters*: số filter của convolution layer
* *kernel\_size*: size Sliding window trượt trên image.
* *relu: max(0,x)* :dùng trong các layer CNN để giảm chi phí tính toán.

Có tác dụng đưa các giá trị âm về thành 0. Để loại bỏ các giá trị âm không cần thiết mà có thể sẽ ảnh hưởng cho việc tính toán ở các layer sau đó.

* *input\_shape:* chính là kích thước của dữ liệu đầu vào.

Layer đầu tiên là layer input nên có input\_shape.

Dùng pooling layer:

|  |
| --- |
| # Add max pooling layer  classifier.add(MaxPooling2D(pool\_size=(2,2))) |

Trong đó:

* *MaxPooling2D*: lấy những đặc điểm nổi bật nhất và resize lại ảnh.

*pool\_size*: size pooling, thường có size ma trận 2x2, đối với ảnh lớn thì 4x4.

Dùng global average pooling layer:

|  |
| --- |
| *# Add global average pooling layer*  classifier.add(GlobalAveragePooling2D()) |

Dùng Dropout layer:

|  |
| --- |
| # Add Dropout layer  classifier.add(Dropout(0.2)) |

* Hạn chế overfiting với tỉ lệ Dropout = 0.2

Dùng Flatten layer:

|  |
| --- |
| # Add Flatten layer  classifier.add(Flatten()) |

Trong đó:

* *Flatten*: dùng để lát phẳng layer để fully connection chuyển thành ma trận cột.

Dùng Dense layer:

|  |
| --- |
| *# Add full connection*  classifier.add(Dense(units=2, activation=**'softmax'**)) |

* *Dense:* layer này sử dụng như một layer neural network bình thường.
* *activation*: softmax dùng trong multi classifier

Sau khi build model xong thì compile nó có tác dụng biên tập lại toàn bộ model của chúng ta đã build. Ở đây chúng ta có thể chọn các tham số để training model như :

|  |
| --- |
| # Compiling the ANN  classifier.compile(optimizer='adam', loss='categorical\_crossentropy', metrics=['accuracy']) |

Trong đó:

* Thuật toán training thông qua tham số *optimizer* (ta chọn Adam để tối ưu với learning\_rate = 0.0001),
* Function *loss* của model chúng ta có thể sử dụng mặc định hoặc tự build thông qua tham số loss, (categorical\_crossentropy dùng trong classifier nhiều class)
* Chọn *metrics* hiện thị khi model được training.

Trong mô hình Keras với API chức năng, cần gọi fit\_generator để huấn luyện dữ liệu hình ảnh được tăng cường bằng cách sử dụng ImageDataGenerator.

|  |
| --- |
| # Fit CNN to images  from keras.preprocessing.image import ImageDataGenerator  train\_datagen = ImageDataGenerator(  rescale=1./255,  shear\_range=0.2,  zoom\_range=0.2,  horizontal\_flip=True) |

Import data:

|  |
| --- |
| train\_set = train\_datagen.flow\_from\_directory(  'data/train',  target\_size=(64, 64),  batch\_size=32,  class\_mode='categorical')  test\_set = test\_datagen.flow\_from\_directory(  'data/test',  target\_size=(64, 64),  batch\_size=32,  class\_mode='categorical') |

Training dữ liệu ảnh:

|  |
| --- |
| history = classifier.fit\_generator(  train\_set,  samples\_per\_epoch=540,  epochs=10,  validation\_data=test\_set,  validation\_steps=201) |

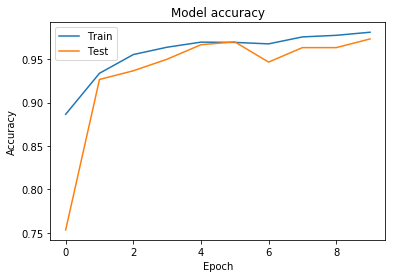
Sau khi trainning nó sẽ lưu lại và tạo ra một model với tên là “**model2.h5**”

|  |
| --- |
| # Save the model  classifier.save("model2.h5") |

Tiếp theo nó sẽ tạ ra Plot training & validation accuracy values:

|  |
| --- |
| plt.plot(history.history['acc'])  plt.plot(history.history['val\_acc'])  plt.title('Model accuracy')  plt.ylabel('Accuracy')  plt.xlabel('Epoch')  plt.legend(['Train', 'Test'], loc='upper left')  plt.show() |

Kết quả sau khi show giá trị accuracy

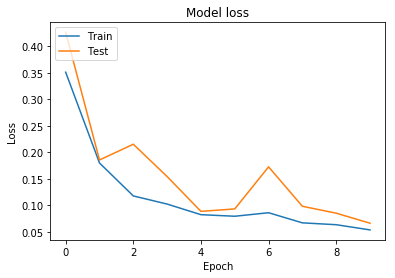


**Hình 17**: Model accuracy

Plot training & validation loss values:

|  |
| --- |
| plt.plot(history.history['loss'])  plt.plot(history.history['val\_loss'])  plt.title('Model loss')  plt.ylabel('Loss')  plt.xlabel('Epoch')  plt.legend(['Train', 'Test'], loc='upper left')  plt.show() |

Kết quả sau khi show giá trị loss:



**Hình 18**: Model loss

**Test model sau khi train:**

Import thư viện:

|  |
| --- |
| from keras.models import load\_model  import matplotlib.pyplot as plt  import numpy as np #Dùng cho đại số tuyến tính và các phép toán đại số  from PIL import Image |

Load mode:

|  |
| --- |
| # Load model  model = load\_model('model2.h5') |

Sau khi load model thành công, chúng ta sử dụng một hình ảnh để test model:

|  |
| --- |
| # Data tự chuẩn bị để test  img = Image.open('car\_test.jpg').convert('RGB')  img = img.resize((64,64))  imgArr = np.array(img)  print(imgArr.shape)  # shape that CNN expects is a 4D array (batch, height, width, channels) <batch: số samples>  imgArr = imgArr.reshape(1,imgArr.shape[0],imgArr.shape[1],3)  plt.imshow(img)  plt.show() |

Kết quả hình dạng của ảnh được đưa vào để test model:

|  |
| --- |
| (64, 64, 3) |

Tiếp theo, show ra dự đóa kết quả và dự đoán lable:

|  |
| --- |
| # Predicting the Test set results  y\_pred = model.predict(imgArr)  print("Predict results: \n", y\_pred[0])  print("Predict lable: \n", np.argmax(y\_pred[0])) |

Kết quả:

|  |
| --- |
| Predict results:  [1. 0.]  Predict lable :  0 (car) |

Predict results: Dự đoán với lable tương ứng với xác xuất cao nhất.

Predict results: dự đoán hình ảnh đưa vào test thuộc lable 1

1. **Kết quả và đánh giá**

Train lần thứ nhất với data gồm: 540 hình ảnh dùng để train và 200 hình ảnh dùng để test.

|  |
| --- |
| **Kết quả:**  Epoch 1/10  16/16 [==============================] - 112s 7s/step - loss: 0.6898 - acc: 0.5234 - val\_loss: 0.6464 - val\_acc: 0.6450  Epoch 2/10  16/16 [==============================] - 95s 6s/step - loss: 0.6699 - acc: 0.5679 - val\_loss: 0.5531 - val\_acc: 0.6447  Epoch 3/10  16/16 [==============================] - 93s 6s/step - loss: 0.6636 - acc: 0.5856 - val\_loss: 0.5051 - val\_acc: 0.6953  Epoch 4/10  16/16 [==============================] - 94s 6s/step - loss: 0.5518 - acc: 0.7622 - val\_loss: 0.4866 - val\_acc: 0.7747  Epoch 5/10  16/16 [==============================] - 93s 6s/step - loss: 0.4049 - acc: 0.8331 - val\_loss: 0.5434 - val\_acc: 0.7559  Epoch 6/10  16/16 [==============================] - 92s 6s/step - loss: 0.3823 - acc: 0.8164 - val\_loss: 0.5711 - val\_acc: 0.7596  Epoch 7/10  16/16 [==============================] - 94s 6s/step - loss: 0.3137 - acc: 0.8739 - val\_loss: 0.3834 - val\_acc: 0.8201  Epoch 8/10  16/16 [==============================] - 93s 6s/step - loss: 0.2833 - acc: 0.8837 - val\_loss: 0.4464 - val\_acc: 0.8151  Epoch 9/10  16/16 [==============================] - 93s 6s/step - loss: 0.2848 - acc: 0.8778 - val\_loss: 0.7139 - val\_acc: 0.7507  Epoch 10/10  16/16 [==============================] - 93s 6s/step - loss: 0.3323 - acc: 0.8739 - val\_loss: 0.5052 - val\_acc: 0.7939 |

|  |  |
| --- | --- |
|  |  |

Train lần thứ 2 với data gồm: 3427 hình ảnh dùng để train và 700 hình ảnh dùng để test.

|  |
| --- |
| **Kết quả:**  Epoch 1/10  107/107 [==============================] - 532s 5s/step - loss: 0.4885 - acc: 0.8175 - val\_loss: 0.6002 - val\_acc: 0.7145  Epoch 2/10  107/107 [==============================] - 507s 5s/step - loss: 0.3272 - acc: 0.8646 - val\_loss: 0.2921 - val\_acc: 0.8813  Epoch 3/10  107/107 [==============================] - 509s 5s/step - loss: 0.1801 - acc: 0.9314 - val\_loss: 0.1902 - val\_acc: 0.9300  Epoch 4/10  107/107 [==============================] - 504s 5s/step - loss: 0.1548 - acc: 0.9410 - val\_loss: 0.1272 - val\_acc: 0.9487  Epoch 5/10  107/107 [==============================] - 503s 5s/step - loss: 0.1358 - acc: 0.9463 - val\_loss: 0.4486 - val\_acc: 0.8614  Epoch 6/10  107/107 [==============================] - 505s 5s/step - loss: 0.1276 - acc: 0.9515 - val\_loss: 0.2410 - val\_acc: 0.9116  Epoch 7/10  107/107 [==============================] - 506s 5s/step - loss: 0.1233 - acc: 0.9544 - val\_loss: 0.7674 - val\_acc: 0.7099  Epoch 8/10  107/107 [==============================] - 504s 5s/step - loss: 0.1132 - acc: 0.9550 - val\_loss: 0.1502 - val\_acc: 0.9475  Epoch 9/10  107/107 [==============================] - 505s 5s/step - loss: 0.1219 - acc: 0.9550 - val\_loss: 0.2206 - val\_acc: 0.9210  Epoch 10/10  107/107 [==============================] - 505s 5s/step - loss: 0.1056 - acc: 0.9576 - val\_loss: 0.1397 - val\_acc: 0.9474 |

|  |  |
| --- | --- |
|  |  |

Train lần thứ 3 với data gồm: 6527 hình ảnh dùng để train và 300 hình ảnh dùng để test.

|  |
| --- |
| **Kết quả:**  Epoch 1/10  203/203 [==============================] - 444s 2s/step - loss: 0.3511 - acc: 0.8865 - val\_loss: 0.4265 - val\_acc: 0.7533  Epoch 2/10  203/203 [==============================] - 374s 2s/step - loss: 0.1800 - acc: 0.9338 - val\_loss: 0.1856 - val\_acc: 0.9267  Epoch 3/10  203/203 [==============================] - 372s 2s/step - loss: 0.1179 - acc: 0.9553 - val\_loss: 0.2153 - val\_acc: 0.9367  Epoch 4/10  203/203 [==============================] - 372s 2s/step - loss: 0.1025 - acc: 0.9638 - val\_loss: 0.1543 - val\_acc: 0.9500  Epoch 5/10  203/203 [==============================] - 374s 2s/step - loss: 0.0825 - acc: 0.9697 - val\_loss: 0.0888 - val\_acc: 0.9667  Epoch 6/10  203/203 [==============================] - 373s 2s/step - loss: 0.0795 - acc: 0.9694 - val\_loss: 0.0933 - val\_acc: 0.9700  Epoch 7/10  203/203 [==============================] - 372s 2s/step - loss: 0.0861 - acc: 0.9677 - val\_loss: 0.1727 - val\_acc: 0.9467  Epoch 8/10  203/203 [==============================] - 375s 2s/step - loss: 0.0671 - acc: 0.9757 - val\_loss: 0.0983 - val\_acc: 0.9633  Epoch 9/10  203/203 [==============================] - 374s 2s/step - loss: 0.0636 - acc: 0.9775 - val\_loss: 0.0853 - val\_acc: 0.9633  Epoch 10/10  203/203 [==============================] - 373s 2s/step - loss: 0.0537 - acc: 0.9811 - val\_loss: 0.0664 - val\_acc: 0.9733 |

|  |  |
| --- | --- |
|  |  |

Đánh giá:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Số lượng hình ảnh train | Số lượng hình ảnh test | Độ mất mát | Độ chính xác |
| 1 | 540 | 200 | 33.23% | 87.39% |
| 2 | 3427 | 700 | 10.56% | 94.74% |
| 3 | 6527 | 300 | 5,37% | 97.33% |

Với bảng kết quả, ta có thể thấy việc thay đổi và tăng số lượng hình ảnh training lên cho thuật toán CNN là rất cần thiết. Ngược lại, số lượng hình ảnh training thấp thì sẽ cho kết quả có độ chính xác không cao.

Sau nhiều thay đổi tỷ lệ hình ảnh train/test, nhóm đã quyết định sử dụng MODEL\_3 để cho ra kết quả tốt nhất với kết quả là 97.33 %.

# Chương VIII. Tổng kết

Sau 15 tuần tìm hiểu về đề tài và các thuật toán có thể áp dụng cho đề tài của nhóm. Nhóm chúng em đạt được một số điểm như sau:

Hiểu được các thuật toán cơ bản của machine learning cơ bản và áp dụng chúng vào các ví dụ đơn giản trong thực tế.

Biết được tầm quan trọng của machine learning trong thời đại 4.0.

Hiểu được thuật toán mà nhóm sử dụng để áp dụng vào đề tài mà nhóm đã chọn (nhận diện các loại phương tiện giao thông).

Nắm được các đặc điểm cơ bản về CNN (Convolutional Neural Networks: mạng nơ-ron tích chập) và áp dụng thành công thuật toán vào đề tài mà nhóm đã chọn.

Nhìn thấy được tầm quan trọng của machine learning trong đời sống hằng ngày là vô cùng to lớn.

Machine learning là một lĩnh vực rộng lớn có nhiều ứng dụng thực tế trong cuộc sống hằng ngày và ngày càng phát triển đặc biệt là trong thời đại 4.0 và qua môn học này nhóm em cũng nhìn thấy tiềm năng của lĩnh vực này là rất lớn trong tương lai.

Tóm lại trong thời đại công nghệ đang bùng nổ mạnh mẽ như hiện nay việc áp dụng machine learning vào trong đời sống là một việc làm rất thiết thực, nó có thể giúp ích rất nhiều cho con người. Giúp chúng ta tiết kiệm thời gian, tiết kiệm chi phí, có thể tạo ra các robot làm việc trong môi trường nguy hiểm thay thế cho con người…

# TÀI LIỆU THAM KHẢO

[1] Deep Learning Based Oil Palm Tree Detection and Counting for High-Resolution Remote Sensing Images.

<https://www.mdpi.com/2072-4292/9/1/22/htm>

[2] Nguyễn Phúc Lương - Ứng dụng Convolutional Neural Network trong bài toán phân loại ảnh

<https://viblo.asia/p/ung-dung-convolutional-neural-network-trong-bai-toan-phan-loai-anh-4dbZNg8ylYM>

[3] Nguyen The Toan - Sử dụng CNN trong bài toán nhận dạng mặt người (Phần 1)

<https://viblo.asia/p/su-dung-cnn-trong-bai-toan-nhan-dang-mat-nguoi-phan-1-eW65GoOP5DO>

[4] Code từ đầu – Machine learning - Neural Network và Deep Learning là gì?

<https://codetudau.com/neural-network-va-deep-learning-la-gi/index.html>

[5] Tổng quan về Mạng Neuron (Neural Network)

<http://bis.net.vn/forums/p/482/898.aspx>