**Ứng dụng thuật toán máy   
học phân loại phương tiện giao thông**

**Nhóm 13:**

**Lê Quang Sơn - 15110118**

**Phạm Phú Thịnh - 15110133**

**Đặng Công Đạt - 15110030**

1. **Introduce.**
2. **Lý do chọn đề tài.**

* Trí tuệ nhân tạo (Artifical Intelligent) hay học máy (Machine Learning) là một lĩnh vực đang được nhắc đến khá nhiều trong thời gian gần đây bởi tính ứng dụng của nó trong thực tiễn. Có khá nhiều ứng dụng của công nghệ học máy đã được áp dụng trong cuộc sống hàng ngày như: google dịch, xe ô tô tự lái, hệ thống gợi ý mua hàng, hệ thống nhận diện khuôn mặt...  và nhận diện xe hơi là một phần không thể thiếu trong việc phát triển ô tô tự lái
* 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 vv cũng sử dụng công nghệ nhận dạng hình ảnh.

1. **Ý nghĩa.**

* 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.

1. **Goal.**

* 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 70%.
* Có thể tận dụng vào trong thực tế.

1. **Objective.**
2. **Đặt vấn đề.**

* Phân tích ứng dụng.
* Tìm hiểu thuật toán.
* Hội ý và đưa ra giải pháp.

1. **Chuẩn bị dữ liệu.**

* Thu thập dữ liệu của các phương tiện.
* Phân loại dữ liệu.
* Phân tích dữ liệu.
* Chuyển đổi dữ liệu dạng hình, nhãn thành dạng dữ liệu matrix theo vector.

1. Thuật toán.

* Tìm hiểu các thuật toán cơ bản và phổ biến trong Machine Learing.
* Xác định các thuật toán phù hợp với đề tài.
* Áp dụng thuật toán đã chọn vào đề tài.

1. **Huấn luyện.**

* Viết code.
* Áp dụng thuật toán xử lý dữ liệu đầu vào để tạo ra một bộ model.

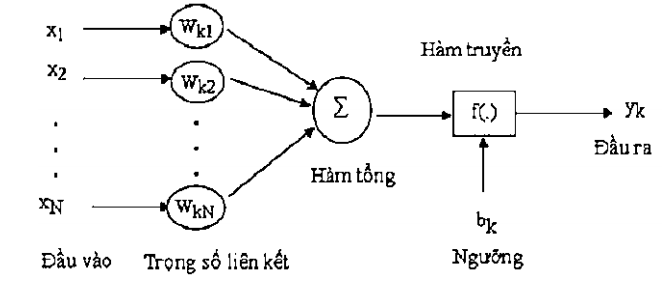
1. **Kiểm thử.**

* Từ dữ liệu đầu vào cho ra kết quả với độ chính xác là bao nhiêu (%) có giống như là mong đợi ban đầu hay chưa.
* Cập nhật lại model nếu như kết quả chưa đạt được như mong muốn ban đầu.

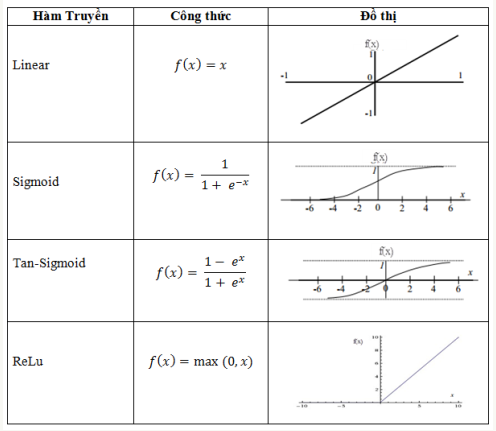
1. **Theory.**
2. **Neural Networks.**
3. ***Tổng quan.***

**-** Mạng neural nhân tạo (Artificial Neural Network: ANN), gọi tắt neural network là mô hình xử lý thông tin mô phỏng hoạt động của các hệ neuron sinh học mà cụ thể hơn ở đây là bộ não con người. Trong đó, thành phần cơ bản của ANN là neural nhân tạo có cách thức hoạt động và xử lý tương tự neuron sinh học. ANN được hình thành từ số lượng lớn các neural được liên kết với nhau theo cấu trúc từng tầng (layer), các neural kết nối với nhau giữa các tầng thông qua trọng số liên kết (weight).

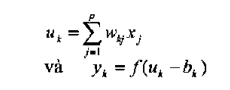
1. ***Cấu trúc.***



* Các thành phần của mạng neural nhân tạo bao gồm:
* Tập các đầu vào: Là các tín hiệu vào (Input signals) của neural, các tính hiệu này thường được đưa vào dưới dạng một vector N chiều.
* Tập các liên kết: Mỗi liên kết được thể hiện bởi một trọng số liên kết - Synaptic weight. Trọng số liên kết giữa các tính hiệu vào thứ j với neural k thường được kí hiệu là wkj. Thông thường, các trọng số này được khởi tạo một cách ngẫu nhiên ở thời điểm khởi tạo mạng và được cập nhật liên tục trong quá trình học mạng.
* Hàm tổng: Thường dùng để tính tổng của tích các đầu vào với trọng số liên kết của nó.
* Ngưỡng: Ngưỡng này thường được đưa vào như một thành phần của hàm truyền.
* Hàm truyền: Hàm này được dùng để giới hạn phạm vi đầu ra của mỗi neural. Nó nhận đầu vào là kết quả của hàm tổng và ngưỡng.Thông thường, phạm vi đầu ra của mỗi neural được giới hạn trong đoạn [0,1] hoặc [-1, 1]. Các hàm truyền rất đa dạng, có thể là các hàm tuyến tính hoặc phi tuyến. Việc lựa chọn hàm truyền nào là tuỳ thuộc vào từng bài toán và kinh nghiệm của người thiết kế mạng. Một số hàm truyền thường sử dụng trong các mô hình mạng nơron được đưa ra trong bảng sau:



* Đầu ra: Là tín hiệu đầu ra của một neural, với mỗi neural sẽ có tối đa một đầu ra.
* Xét về mặt toán học, cấu trúc của một neural k, được mô tả bằng cặp biểu thức:

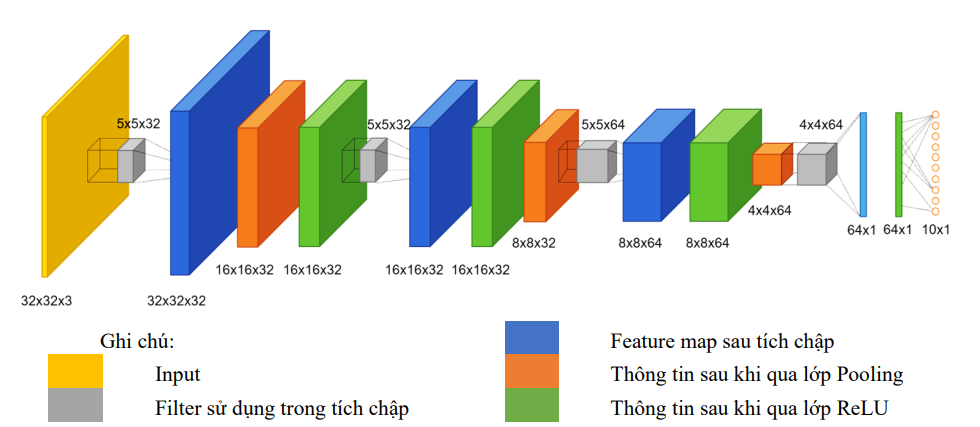
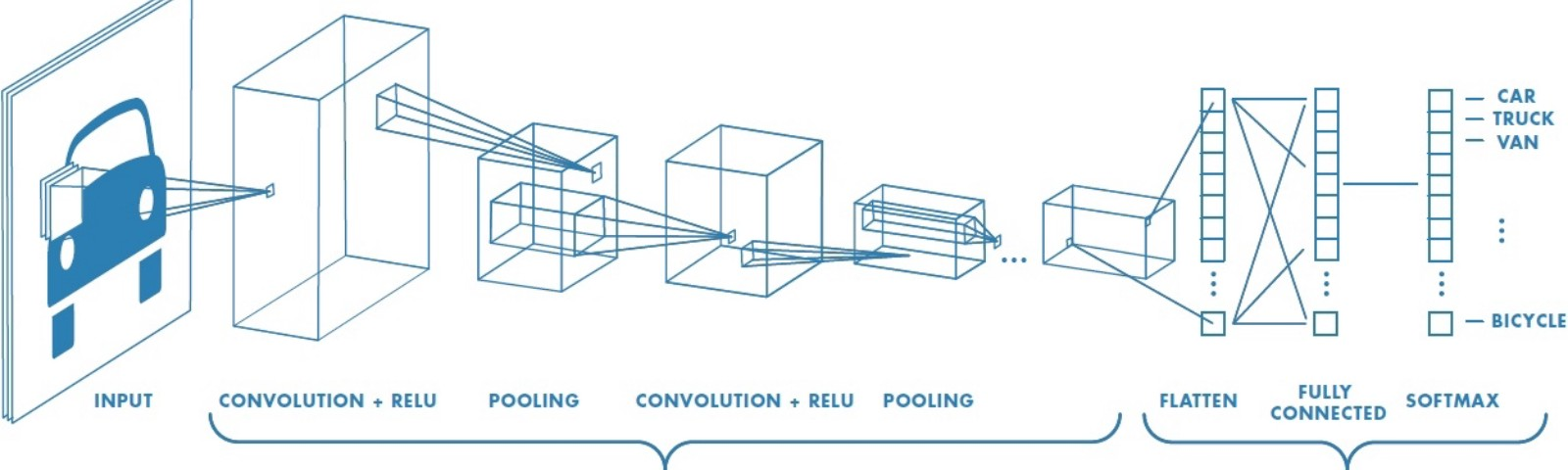


Trong đó:

* **x1, x2, x3...., xp**: là các tín hiệu vào.
* **wk1wk2, wk3…, wkp** là các trọng số liên kết của neural thứ k.
* uk là hàm tổng.
* bk là một ngưỡng.
* f là hàm truyền.
* yk là tín hiệu đầu ra của neural.

**=>** Như vậy neural nhân tạo nhận các tín hiệu đầu vào, xử lý (nhân các tín hiệu này với trọng số liên kết, tính tổng các tích thu được rồi gửi kết quả tới hàm truyền), và cho một tín hiệu đầu ra (là kết quả của hàm truyền).

1. **Cấu trúc của mạng CNN (Convolutional Neural Network).**



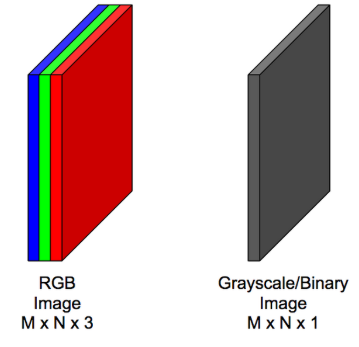
1. ***Input (Dữ liệu Training).***

* Input đầu vào là một bức ảnh được biểu diển bởi ma trận pixel với kích thước: [w x h x d]

+ w: chiều rộng

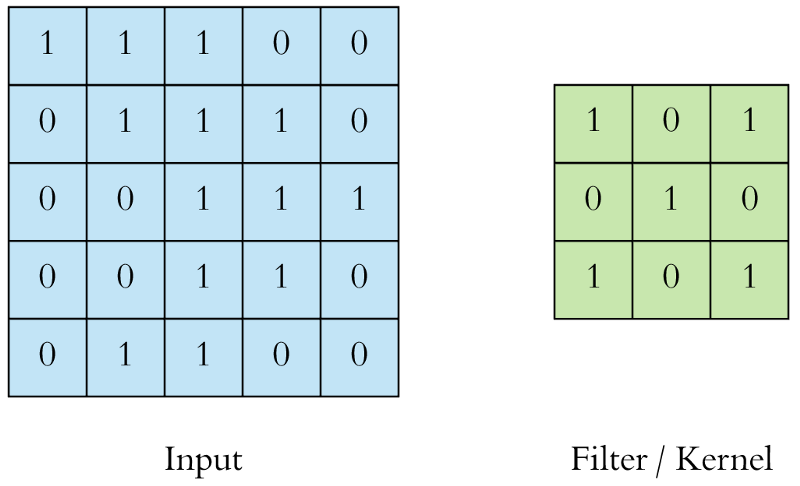
+ h: chiều cao

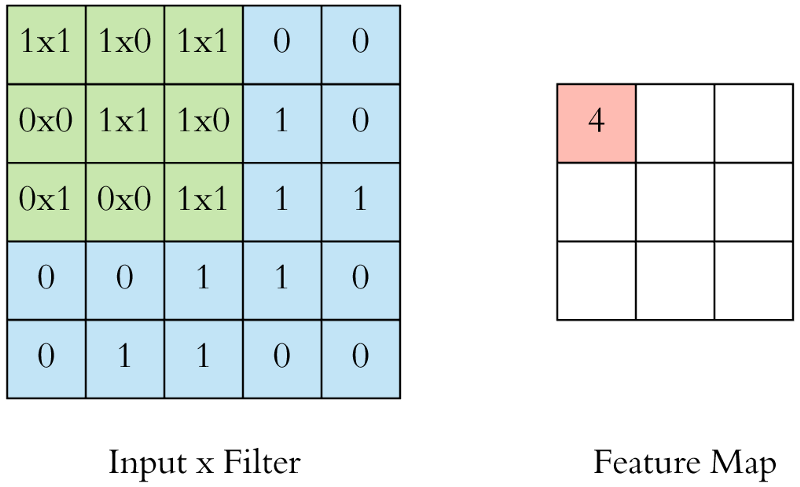
+ d: Là độ sâu, hay dễ hiểu là số lớp màu của ảnh. Ví dụ ảnh RBG sẽ là 3 lớp ảnh Đỏ, Xanh Dương, Xanh.



1. ***Filter (Bộ lọc).***

* Nếu ta có tập dữ liệu huấn luyện lớn và hiệu năng tính toán cao, với những tập ảnh kích thước lớn và nhiều chi tiết phức tạp, các bộ lọc cạnh có thể được huấn luyện tự động từ tập dữ liệu. Nghĩa là các giá trị của ma trận lọc được coi như tham số của một mạng nơ-ron và huấn luyện (sử dụng back-propagation chẳng hạn) để có một tập giá trị tối ưu.
* Đây là một ma trận sẽ quét qua ma trận dữ liệu đầu vào, từ trái qua phải, trên xuống dưới, và nhân tương ứng từng giá trị của ma trận đầu vào mà ma trận kernel rồi cộng tổng lại, đưa qua activation funciton (sigmoid, relu, elu...), kết quả sẽ là một con số cụ thể, tập hợp các con số này lại là 1 ma trận nữa, chính là feature map.

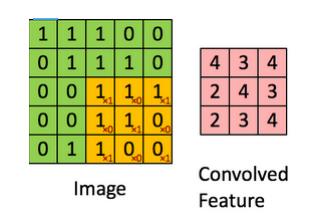




Thực hiện lần lượt cho đến hết.

1. ***Feature Map.***
2. *Convolution.*

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



* Ma trận convolved feature là kết quả nhân từng ma trận có kích thước 3\*3 trong ma trận lớn 5\*5 với một ma trận 3\*3 cho trước.
* Ma trận có kích thước 3\*3 cho trước được gọi là sliding windows hay còn được gọi là kernel, filter hoặc feature detect.
* Lớp tích chập được dùng để phát hiện và trích xuất đặc trưng - chi tiết của ảnh.
* 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.
* Dưới đây là một số các khái niệm cơ bản của phần này:
* **Filter, Kernel hay Feature Detector** đều là cách gọi của ma trận lọc (như mình đã đề cập ở trên). Thông thường, ở các lớp đầu tiên của Conv Layer sẽ có kích thước là [5x5x3]
* **Convolved Feature, Activation Map or Feature Map:** là đầu ra của ảnh khi cho bộ lọc chạy hết bức ảnh với phép tích vô hướng.
* **Receptive field:** là vùng ảnh được chọn để tính tích chập, hay bằng đúng cái kích thước của bộ lọc.
* **Depth:** là số lượng bộ lọc. Lưu ý: ở đây là số lượng bộ lọc (filter) chứ không phải số lượng kênh màu RBG như ở trên.
* **Stride:** được hiểu là khoảng cách dịch chuyển của bộ lọc sau mỗi lần tính. Ví dụ khi stride=2. Tức sau khi tính xong tại 1 vùng ảnh, nó sẽ dịch sang phải 2 pixel. Tương tự với việc dịch xuống dưới
* **Zero-Padding:** là việc thêm các giá trị 0 ở xung quanh biên ảnh, để đảm bảo phép tích chập được thực hiện đủ trên toàn ảnh.
* **Vậy kích thước đầu ra của ảnh với mỗi layer được tính như thế nào?**
* Giả sử ảnh đầu ra là [W2 x H2 x D2]

Thì: W2 = (W1 – F + 2P) / S + 1

H2 = (H1 – F +2P) / S + 1

D2 = K

* Trong đó:

[W1xH1xD1]: Kích thước đầu vào

F: Kích thước bộ lọc Kernel

S: giá trị Stride

P: số lượng zero-padding thêm vào viền ảnh

K: Số lượng bộ lọc (Depth)

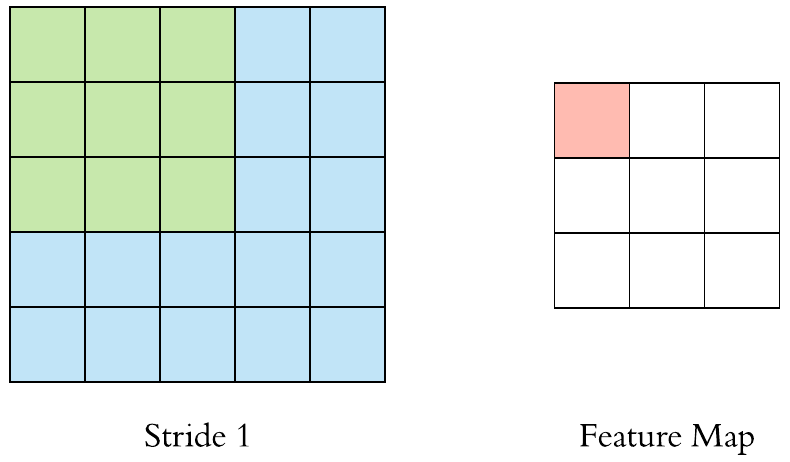
* Ví dụ:
* Kích thước đầu ra của lớp đầu tiên trong kiến trúc mạng học sâu nơ-ron tích chập (Deep Convolutional Neural Networks) đã từng chiến thắng ImageNet Chanllenge 2012 là:
* Đầu vào: [227x227x3], W=227, F=11, S=4, P=0, and K=96.
* Đầu ra: (227 - 11) / 4 + 1 = 55
* Vậy kích thước đầu ra của layer đầu tiên là [55x55x96].

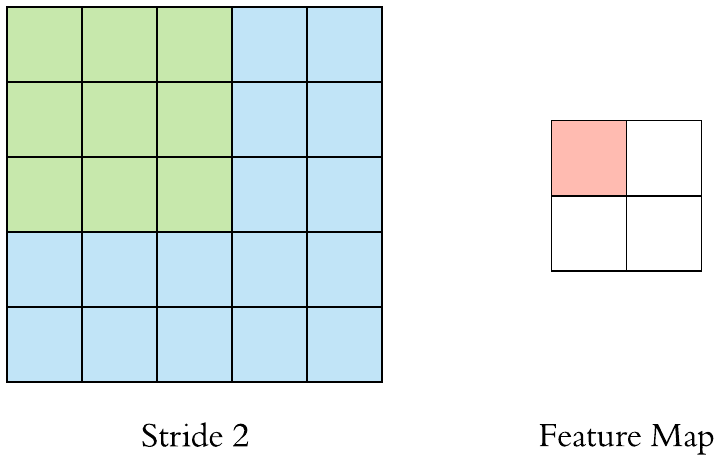
1. *ReLu.*

* ReLu layer áp dụng các kích hoạt (activation function) max(0, x) lên đầu ra của Conv Layer, có tác dụng đưa các giá trị âm về thành 0. Layer này không thay đổi kích thước của ảnh và không có thêm bất kì tham số nào.
* Mục đích của lớp ReLu là đưa ảnh một mức ngưỡng, ở đây là 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.

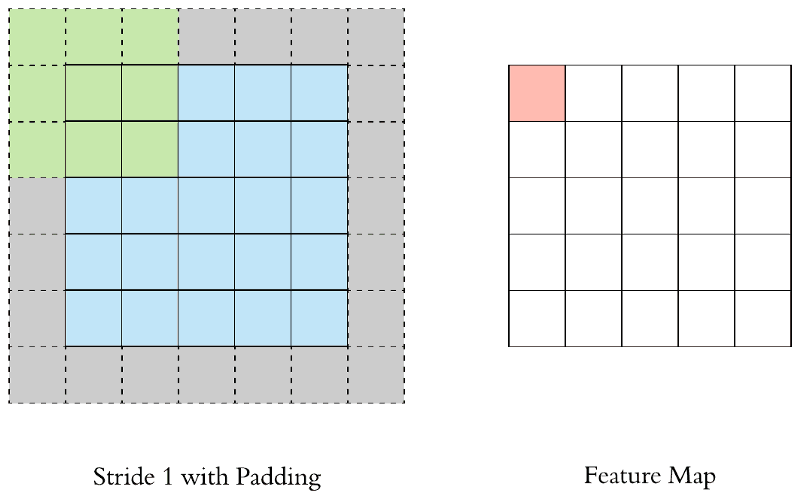
1. *Stride and Padding.*

* 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.





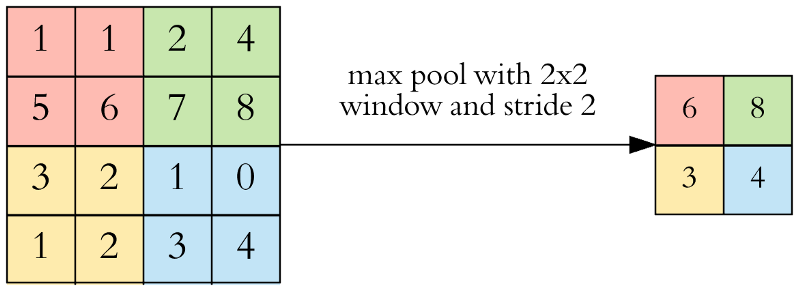
* Padding:
* Khi ta điều chỉnh padding = 1, tức là ta đã thêm 1 ô bọc xung quanh các cạnh của input, muốn phần bọc này càng dày thì ta cần phải tăng padding lên. Hãy nhìn vào ví dụ sau, ta xét padding = 1.



Phần màu xám chính là phần bọc thêm vào input**.**

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:



* 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.

1. *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.

