TRƯỜNG ĐẠI HỌC NGOẠI NGỮ - TIN HỌC
THÀNH PHỐ HỒ CHÍ MINH
KHOA CÔNG NGHỆ THÔNG TIN

BÀI BÁO CÁO ĐỒ ÁN

TRIỂN KHAI MẠNG NEURAL NETWORK VÀ CONVOLUTIONAL NEURAL NETWORK VÀ THỰC NGHIỆM PHÂN LỚP MNIST

Môn Học: Trí Tuệ Nhân Tạo

Giáo viên hướng dẫn: PGS.TS Dương Tuấn Anh

Sinh viên thực hiện:
Trần Minh Hải 19DH110032
Phạm Võ Nhật Đăng: 19DH110144

MỤC LỤC

Danh mục hình ảnh

Danh mục hình bảng

Danh mục từ viết tắt

ANN	Neural Network
CNN	Convolutional Neural Network

1. GIỚI THIỆU

Ngày nay, hễ nhắc tới các thiết bị điện tử hay bất cứ thiết bị nào người ta đều nhắc đến trí tuệ nhân tạo được tích hợp trên thiết bị đó. Vậy trí tuệ nhân tạo là gì và được ứng dụng như thế nào trong cuộc sống?

Trí tuệ nhân tạo hay trí thông minh nhân tạo (Artificial intelligence – viết tắt là AI) là một ngành thuộc lĩnh vực khoa học máy tính (Computer science). Là trí tuệ do con người lập trình tạo nên với mục tiêu giúp máy tính có thể tự động hóa các hành vi thông minh như con người.

Trí tuệ nhân tạo khác với việc lập trình logic trong các ngôn ngữ lập trình là ở việc ứng dụng các hệ thống học máy (machine learning) để mô phỏng trí tuệ của con người trong các xử lý mà con người làm tốt hơn máy tính.

Cụ thể, trí tuệ nhân tạo giúp máy tính có được những trí tuệ của con người như: biết suy nghĩ và lập luận để giải quyết vấn đề, biết giao tiếp do hiểu ngôn ngữ, tiếng nói, biết học và tự thích nghi,…

1.1.Machine learning là gì?

Machine Learning là một thuật ngữ rộng để chỉ hành động bạn dạy máy tính cải thiện một nhiệm vụ mà nó đang thực hiện. Cụ thể hơn, machine learning đề cập tới bất kỳ hệ thống mà hiệu suất của máy tính khi thực hiện một nhiệm vụ sẽ trở nên tốt hơn sau khi hoàn thành nhiệm vụ đó nhiều lần. Hay nói cách khác, khả năng cơ bản nhất của machine learning là sử dụng thuật toán để phân tích những thông tin có sẵn, học hỏi từ nó rồi đưa ra quyết định hoặc dự đoán về một thứ gì đó có liên quan. Thay vì tạo ra một phần mềm với những hành động, hướng dẫn chi tiết để thực hiện một nhiệm vụ cụ thể, máy tính được "huấn luyện" bằng cách sử dụng lượng dữ liệu và các thuật toán để học cách thực hiện nhiệm vụ.

1.2. Deep learning là gì?

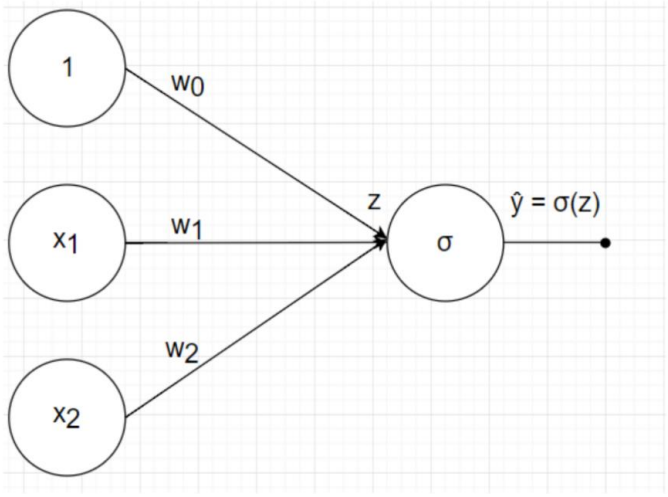
Deep learning là loại machine learning mà trong đó máy tự đào tạo chính nó. Deep learning đòi hỏi rất nhiều dữ liệu đầu vào và sức mạnh tính toán hơn là machine learning, nhưng nó đã bắt đầu được triển khai bởi các công ty công nghệ lớn như Facebook, Amazon. Trong đó, một trong những cái tên nổi tiếng nhất về machine learning là AlphaGo, một máy tính có thể chơi cờ vây với chính bản thân nó cho đến khi nó có thể dự đoán những đường đi nước bước chính xác nhất đủ để đánh bại nhiều nhà vô địch trên thế giới.

2. GIỚI THỆU MẠNG NEURAL NETWORK VÀ CONVOLUTIONAL NEURAL NETWORK

2.1. Mạng Neural Network

Một mạng nơ ron nhân tạo (artificial neural network – ANN hay neural network) được cấu tạo từ các đơn vị xử lý được gọi là perceptron, được xây dựng theo nhiều kiểu khác nhau để hình thành một cấu trúc mạng.

Một ANN bao gồm nhiều perceptron. Mỗi perceptron nhận dữ liệu đầu vào, xử lý đầu vào và cung cấp một giá trị đầu ra



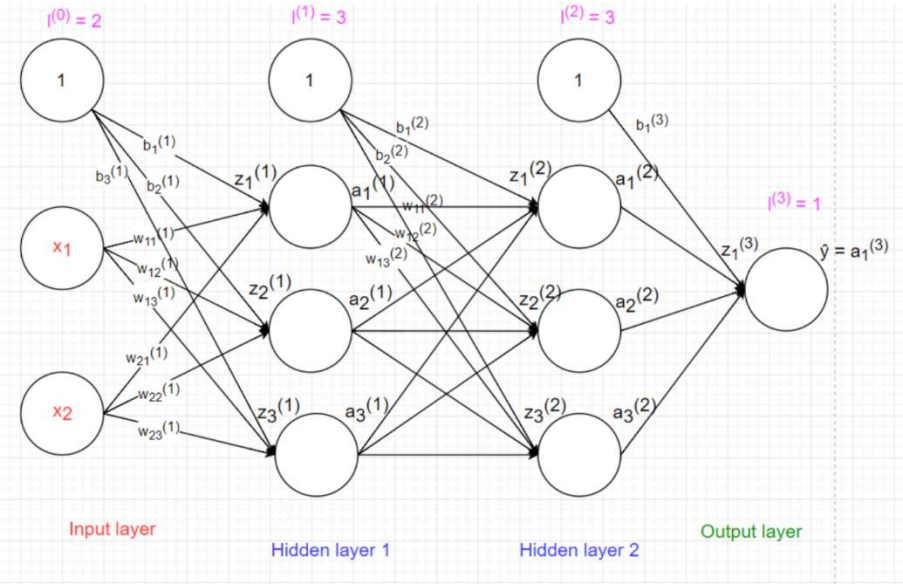
***Hình 1**. Một perceptron*

Trong hình 1 có:

- Tính tổng linear : $z = w_0 + w_1 * x_1 + w_2 * x_2$
- Áp dụng sigmoid function: $\hat{y} = \sigma(z)$

Đầu vào có thể là dữ liệu đầu vào thô hoặc là đầu ra của các perceptron khác. Đầu ra có thể là kết quả cuối cùng (thí dụ 1 có nghĩa là yes, 0 có nghĩa là no) hoặc là đầu vào cho những perceptron khác

Mỗi mạng nơ ron ANN bao gồm nhiều perceptrons được tổ chức thành các tầng (layer). Một cấu trúc tiêu biểu của ANN được minh họa ở Hình 2.



***Hình 2**. Một mạng Nơ-ron*

Mô hình neural network trên gồm 3 layer. Input layer có 2 node ($l^0 = 2$), hidden layer 1 có 3 node, hidden layer 2 có 3 node và output layer có 1 node.

Perceptron đơn lẻ chỉ có thể biểu diễn những mặt cong quyết định tuyến tính (linear decision surfaces). Ngược lại, mạng nơ ron nhiều tầng huấn luyện bằng giải thuật lan truyền ngược (backpropagation algorithm) có thể biểu diễn được những mặt cong quyết định phi tuyến (nonlinear decision surfaces).

Huấn luyện mạng nơ-ron:

- Khởi đầu quá trình huấn luyện, vector trọng số w được gán những giá trị ban đầu một cách ngẫu nhiên.
- Cho tập dữ liệu huấn luyện, giải thuật huấn luyện sẽ lấy từng mẫu huấn luyện nạp vào mạng nơ ron để nhận được giá trị đầu ra của mạng nơ ron và dùng độ sai biệt các giá trị đầu ra mà mạng xấp xỉ và giá trị đầu ra mong muốn để điều chỉnh lại giá trị của vector trọng số w. Công việc này tiếp tục với mọi mẫu trong tập huấn luyện.
- Khi giải thuật huấn luyện duyệt qua hết mọi mẫu trong tập huấn luyện, ta bảo giải thuật này đã hoàn tất một epoch (lượt lặp). Nhưng giải thuật phải lặp lại nhiều epoch. Số epoch này do người dùng ấn định. Thông thường điều kiện dừng của giải thuật huấn luyện mạng nơ ron là khi giải thuật đã lặp đủ số lượng epoch qui định trước. Thí dụ số epoch là 500 hay 1000.

2.2. Mạng Convolutional neural network

Mạng nơ ron tích chập (convolutional neural network –CNN) là một họ mạng nơ ron đa tầng được thiết kế chuyên cho dữ liệu hai chiều như hình ảnh và video.

Mạng nơ ron CNN tận dụng cấu trúc không gian của hình ảnh.

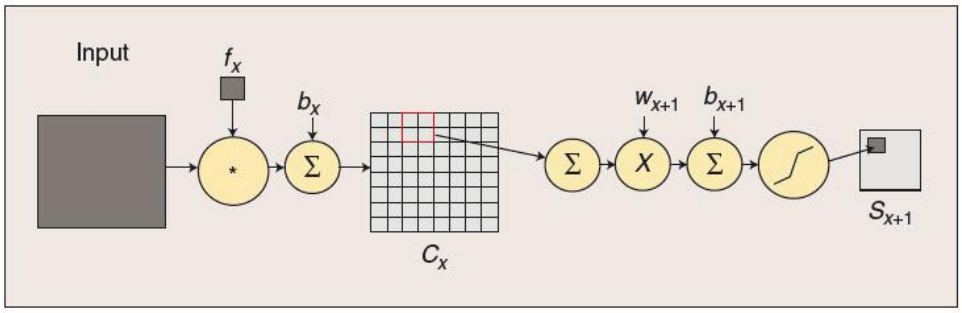
Trong mạng nơ ron CNN, những bộ phận của hình ảnh (được gọi là các trường tiếp nhận cục bộ - local receptive field) được coi như là những dữ liệu đầu vào của tầng thấp nhất trong cấu trúc phân cấp của mạng

Thông tin được lan truyền qua các tầng khác nhau của mạng mà tại mỗi tầng sự sàng lọc số (digital filtering) được áp dụng để rút trích những đặc trưng tiềm ẩn từ dữ liệu quan sát.

Có hai loại tầng sau đây làm nên mạng nơ ron CNN:

- Tầng tích chập (Convolution layer)
- Tầng lấy mẫu giảm (Subsampling layer hay pooling layer)

Mạng CNN vận dụng các kỹ thuật từ các lãnh vực như mạng nơ ron (neural network), mô hình đồ thị (graphical modeling), toán tối ưu (optimization,) , nhận dạng mẫu (pattern recognition) và xử lý tín hiệu (signal processing).



***Hình 3**. Minh họa mạng nơ-ron tích chập*

Quá trình tích chập p bao gồm việc tính tích chập dữ liệu đầu vào (hình ảnh) với một *bộ lọc* (filter) khả huấn luyện f_x, rồi thêm một *độ lệch* (bias) khả huấn luyện b_x để sinh ra tầng tích chập C_x.

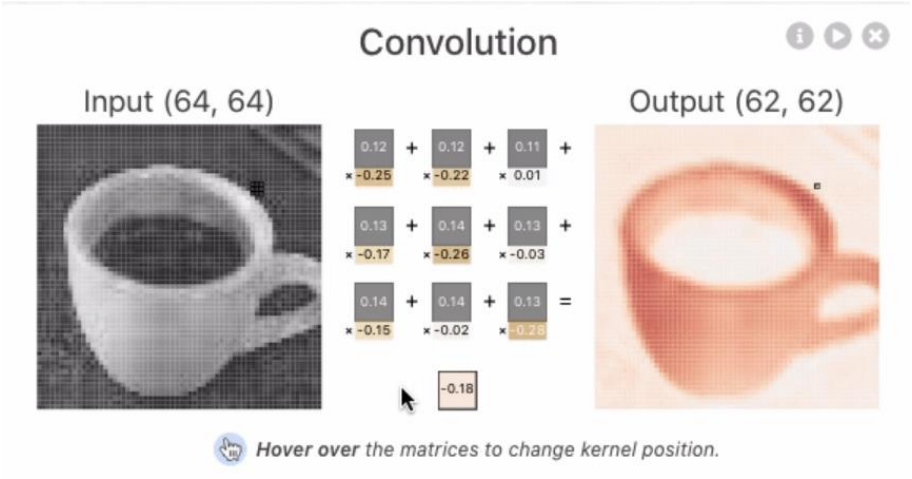
Quá trình lấy mẫu giảm bao gồm lấy tổng một vùng lân cận, *đánh trọng số* bằng trị vô hướng w_{x+1}, thêm độ lệch khả huấn luyện b_{x+1} và truyền sang một hàm sigmoid để tạo ra một *bản đồ đặc trưng* nhỏ hơn S_{x+1}.

2.2.1. Convolutional layer

Đây là lớp quan trọng nhất của CNN, lớp này có nhiệm vụ thực hiện mọi tính toán. Những yếu tố quan trọng của một convolutional layer là: stride, padding, filter map, feature map.

- CNN sử dụng các filter để áp dụng vào vùng của hình ảnh. Những filter map này được gọi là ma trận 3 chiều, mà bên trong nó là các con số và chúng là parameter. Mỗi một kết nối sẽ học một trọng số và mỗi neuron ẩn sẽ học một bias. Mỗi một vùng đấy gọi là một trường tiếp nhận cục bộ.
- Stride có nghĩa là khi bạn dịch chuyển filter map theo pixel dựa vào giá trị trừ trái sang phải. Và sự chuyển dịch này chính là Stride.
- Padding: Là các giá trị 0 được thêm vào với lớp input.

- Feature map: Nó thể hiện kết quả của mỗi lần filter map quét qua input. Sau mỗi lần quét sẽ xảy ra quá trình tính toán.



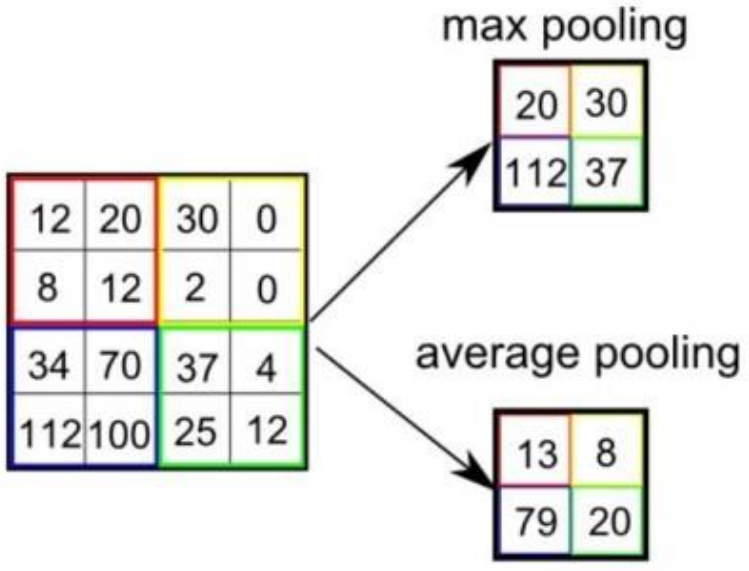
Hình 4. *Green channel trong convolutional layer*

Hình 5 minh họa green channel khi đi qua một convolutional layer với filter(kernel) để tạo ra feature map

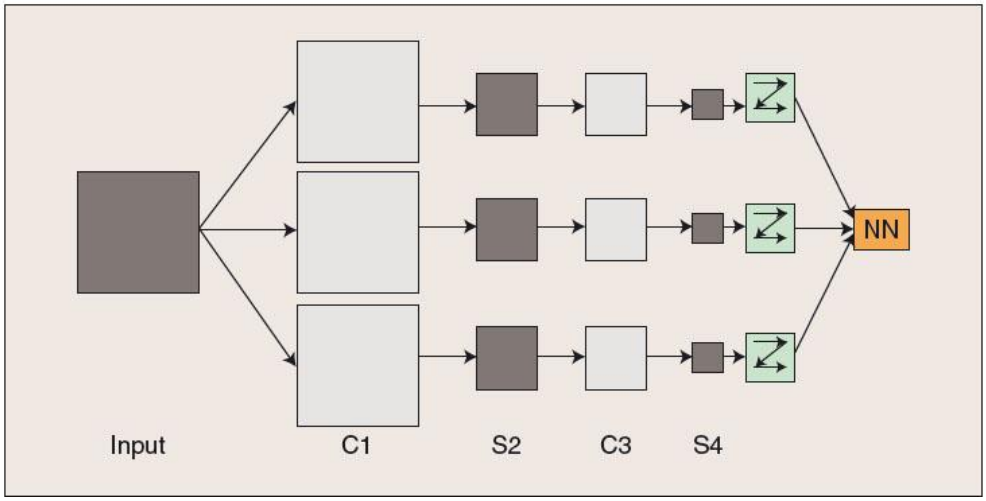
2.2.2. Pooling layer

Pooling layer thường được dùng giữa các convolutional layer, để giảm kích thước dữ liệu nhưng vẫn giữ được các thuộc tính quan trọng. Kích thước dữ liệu giảm giúp giảm việc tính toán trong model.

Có 2 loại pooling layer phổ biến là: max pooling và average pooling.



Hình 5. *Ví dụ về hai loại pooling layer*



Hình 6. *Ví dụ mạng CNN*

Trong hình 6:

- Hình ảnh nhập vào được tích chập với 3 bộ lọc (filter) và các độ lệch (bias) để tạp ra ba bản đồ đặc trưng (feature map) ở tầng C1.
- Mỗi nhóm 4 pixel trong bản đồ đặc trưng được cộng, đánh trọng số và phối hợp với một độ lệch, rồi được truyền sang một hàm sigmoid để sinh ra ba bản đồ đặc trưng khác ở tầng S2.
- Ba bản đồ đặc trưng ở tầng S2 lại được lọc để tạo ra tầng C3. Kế đến, cả hệ phân cấp tạo ra tầng S4 với cùng một cách như khi tạo ra tầng S2.
- Sau cùng các giá trị pixel này được rasterize và biểu diễn thành một vector đơn để đưa sang một mạng nơ ron truyền thống để tạo đầu ra.

2.2.3. Dense Layer (Tầng dày đặc)

Tại giai đoạn cuối cùng của quá trình, các trị đầu ra được truyền thẳng sang một mạng nơ ron truyền thẳng để tạo ra trị đầu ra cuối cùng của hệ thống (tầng này còn được gọi là tầng dày đặc).

Mối liên hệ mật thiết giữa các tầng và thông tin dạng không gian trong mạng CNN khiến cho những mạng này rất thích hợp với ứng dụng xử lý hình ảnh.

Tóm lại:

- Các tầng lấy mẫu giảm (Pooling layer) có thể lấy mẫu hình ảnh theo kiểu nhỏ bớt (down-sample) và giảm thiểu chi tiết.
- Các tầng tích chập (Convolution layer) có thể phát hiện các đặc trưng tại bất kỳ phần nào của tấm ảnh.

2.3. Activation function

Hàm kích hoạt (activation function) mô phỏng tỷ lệ truyền xung qua node của một neuron thần kinh. Trong một mạng nơ-ron nhân tạo, hàm kích hoạt đóng vai trò là thành phần phi tuyến tại output của các nơ-ron. Trong bài viết này, chúng ta sẽ cùng tìm hiểu các hàm kích hoạt phổ biến nhất và các ưu, nhược điểm của chúng.

Trong hình 1 sử dụng hàm kích hoạt tuyến tính, cũng là mô hình neural network đơn giản nhất hay còn được gọi là Logistic regression

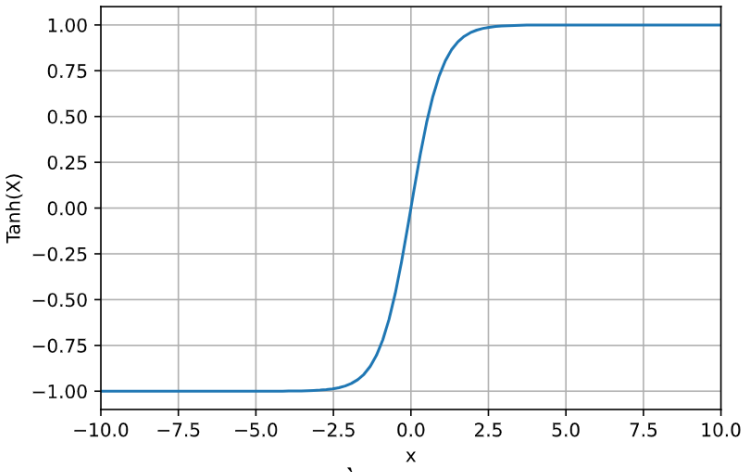
Thông thường các hàm kích hoạt là các hàm phi tuyến, nếu mạng nơ-ron của chúng ta dù cho nhiều lớp mà không sử dụng hàm phi tuyến thì kết quả của mạng đó như một lớp tuyến tính mà thôi

2.3.1. Hàm sigmoid

Công thức:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

Phân tích:



Hình 7. *Đồ thị hàm sigmoid*

Giống như perceptron, đơn vị sigmoid trước tiên tính tổ hợp tuyến tính của các giá trị đầu vào và áp dụng một hàm ngưỡng lên kết quả đó. Tuy nhiên trong trường hợp của đơn vị sigmoid, đầu ra của hàm ngưỡng là một hàm liên tục theo giá trị đầu vào của nó.

Đầu ra của hàm này trong tầm từ 0 đến 1, tăng dần một cách đơn điệu theo giá trị đầu vào.

Ta có thể áp dụng nguyên tắc suy giảm độ dốc để huấn luyện Một đơn vị sigmoid

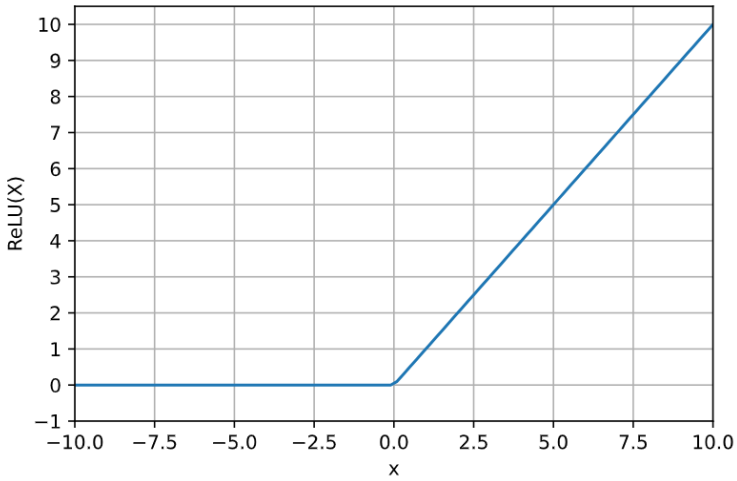
Mạng nhiều tầng kết nối nhiều đơn vị sigmoid $\Rightarrow$ giải thuật Lan Truyền Ngược (Backpropagation)

2.3.2. Hàm relu

Công thức:

$$f(x) = max(x)$$

Phân tích:



Hình 8. *Đồ thị hàm relu*

Hàm ReLU đang được sử dụng khá nhiều trong những năm gần đây khi huấn luyện các mạng neuron. ReLU đơn giản lọc các giá trị < 0. Nhìn vào công thức chúng ta dễ dàng hiểu được cách hoạt động của nó. Một số ưu điểm khá vượt trội của nó so với Sigmoi

Tốc độ hội tụ nhanh hơn

Tính toán nhanh hơn

2.3.3. Hàm Softmax

Softmax là một hàm toán học chuyển đổi một vectơ số thành một vectơ xác suất, trong đó xác suất của mỗi giá trị tỷ lệ với tỷ lệ tương đối của mỗi giá trị trong vectơ.

Việc sử dụng phổ biến nhất của hàm softmax trong học máy ứng dụng là nó được sử dụng như một hàm kích hoạt trong mô hình mạng nơ-ron. Cụ thể, mạng được định cấu hình để xuất ra N giá trị, một giá trị cho mỗi lớp trong nhiệm vụ phân loại và hàm softmax được sử dụng để chuẩn hóa kết quả đầu ra, chuyển đổi chúng từ giá trị tổng có trọng số thành xác suất tổng bằng một

Công thức:

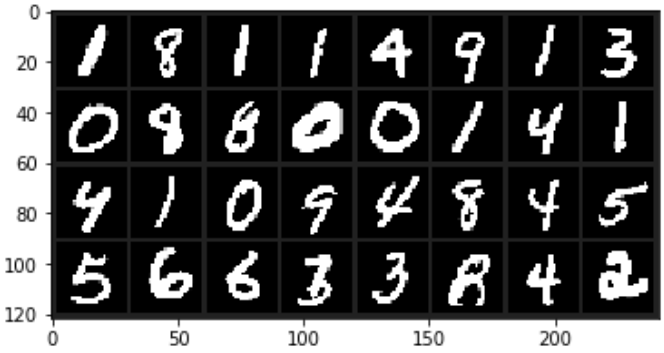
$$Softmax(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$$

X đại diện cho các giá trị từ các nơ-ron của lớp đầu ra. Hàm mũ hoạt động như một hàm phi tuyến tính. Sau đó, các giá trị này được chia cho tổng các giá trị theo cấp số nhân để chuẩn hóa và sau đó chuyển chúng thành xác suất.

3. TRIỂN KHAI MẠNG NEURAL NETWORK VÀ CONVOLUTONAL NEURAL NETWORK PHÂN LỚP MNIST DATASETS

3.1.Định nghĩa bài toán

MNIST được giới thiệu năm 1998 bởi Yann Lecun và cộng sự nhằm đánh giá các mô hình phân lớp. MNIST là tập dữ liệu chữ viết từ 0 đến 9.



Hình 9. *Mnist dataset*

Trong đó, mỗi hình là một ảnh đen trắng chứa một số được viết tay có kích thước là 28x28. Bộ dataset vô cùng đồ sộ với khoảng 60.000 data training và 10.000 data test và được sử dụng phổ biến trong các thuật toán nhận dạng ảnh.

Nhiệm vụ: Sử dụng ANN và CNN để phân lớp bộ dataset này

Công cụ thực hiện: Phần mềm Keras

Keras là một open source cho Neural Network được viết bởi ngôn ngữ Python. Nó là một library được phát triển vào năm 205 bởi Francois Chollet, là một kỹ sư nghiên cứu Deep Learning. Keras có thể sử dụng chung với các thư viện nổi tiếng như Tensorflow, CNTK, Theano. Một số ưu điểm của Keras như:

- Dễ sử dụng, dùng đơn giản hơn Tensor, xây dựng model nhanh.
- Run được trên cả CPU và GPU.

3.2. Tiền sử lý dữ liệu

3.2.1. Chia dữ liệu thành training, validation, test

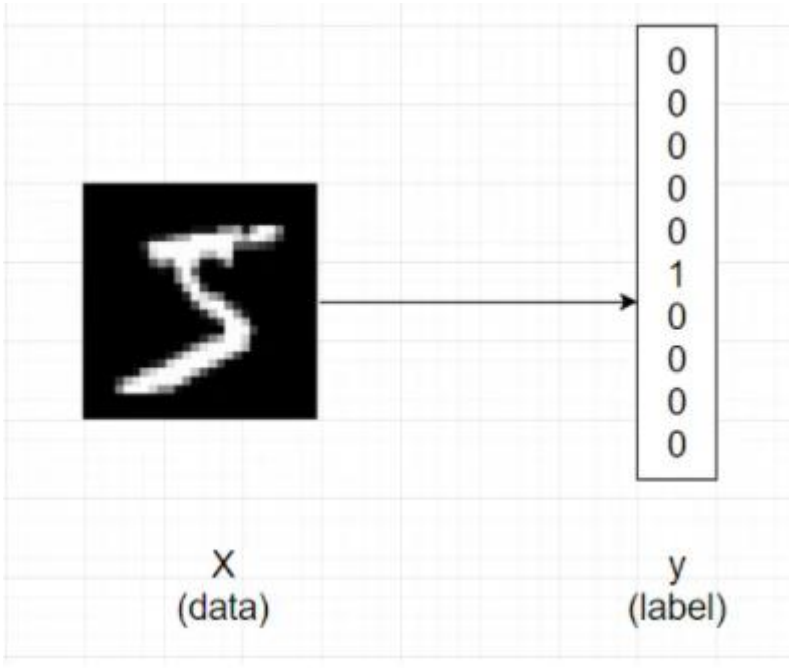
MNIST dataset có 60.000 tập trainning và 10.000 tập test. Keras có hỗ trợ bộ dataset này nên ta không cần tải về.

Chia tập tranining thành 2 phần:

- Training: gồm 50.000 ảnh
- Validation: gồm 10.000 ảnh

3.2.2. Chuyển đổi one-hot encoding

One-hot encoding là quá trình biến đổi từng giá trị thành các đặc trưng nhị phân chỉ chứa giá trị **1** hoặc **0**. Mỗi mẫu trong đặc trưng phân loại sẽ được biến đổi thành một vector có kích thước **m** chỉ với một trong các giá trị là **1** (là nhãn(label) của lớp đó)



Hình 10. *Ví dụ one-hot encoding*

Hình 10 là ví dụ về one-hot encoding, biểu diễn hình số 5 dưới dạng vector có 10 giá trị và giá trị thứ 5 được biểu diễn bằng 1

3.2.3. Chuẩn hóa dữ liệu(data normalization)

Trong xử lý ảnh, chuẩn hóa là một quá trình thay đổi phạm vi giá trị cường độ pixel.

Ảnh xám được biểu diễn bằng một giá trị nguyên trong khoảng từ [0,255].

Giá trị 0 là màu đen, 255 là màu trắng và giá trị pixel càng gần 0 thì càng tối và càng gần 255 thì càng sáng.

Vậy để chuẩn hóa dữ liệu(normalization), ta lấy dữ liệu chia cho 255 để đưa về miền giá trị [0-1]

Normalization sẽ giúp mô hình học nhanh hơn và hội tụ nhanh hơn, tính toán ít hơn.

3.3. Loss Function cho mạng Neural Network và Convolutional Neural Network

Loss function hay còn gọi là hàm mất mát, thể hiện một mối quan hệ giữa $\hat{y}$ (là kết quả dự đoán của model) và y (là giá trị thực tế)

Loss function mục đích là để tối ưu mô hình của mình sao cho tốt nhất và dùng đánh giá độ tốt của model, $\hat{y}$ càng gần y thì càng tốt. Tức là khi ta dựa vào loss function chúng ta có thể tính gradient descent để tối ưu loss function càng về gần 0 càng tốt

Categorical_crossentropy là hàm lỗi cho mô hình phân loại nhiều lớp trong đó có hai hoặc nhiều nhãn đầu ra. Nhãn đầu ra được biết đổi thành một vector one-hot encoding trước khi áp dụng hàm này

Công thức Categorical_crossentropy cho bài toán phân loại Mnist dataset

$$Loss = -\sum_{i=1}^{10} y_i * \log(\hat{y}_i)$$

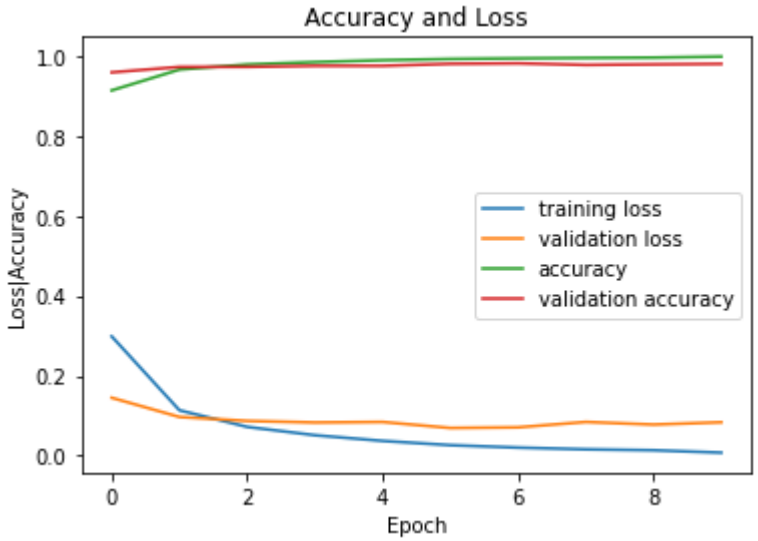
Optimizer(thuật toán tối ưu) là cơ sở để xây dựng mô hình ANN và CNN, với mục đích học được các features của dữ liệu đầu vào để tìm các tham số phù hợp. Thuật toán Adam được sử dụng phổ biến trong các mô hình với ưu điểm hội tụ nhanh và độ chính xác cao.

3.4. Triển khai mạng Neural Network

Xây dựng mạng Neurl Network cho Mnist dataset gồm có 3 lớp:

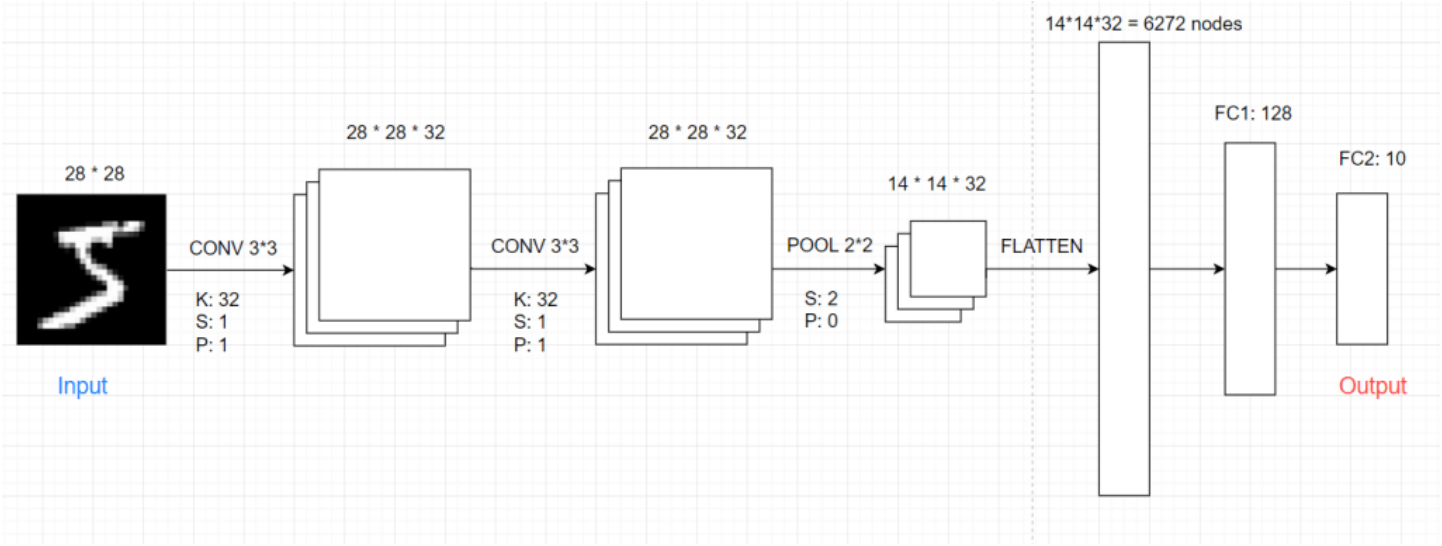
- **Lớp đầu vào** gồm có 128 node. Với đầu vào là một vector – Input_shape(784,) và activation function là Relu
- **Lớp ẩn** gồm có 32 node với activation function là Relu
- **Lớp đầu ra** gồm có 10 node với activation function là Softmax. Tương ứng với 10 lớp của tập dataset

Sử dụng **Loss function** là Categorical_crossentropy và **Optimizer** Adam



Hình 11. *Accuracy và loss trên tập train và tập validation của mạng ANN*

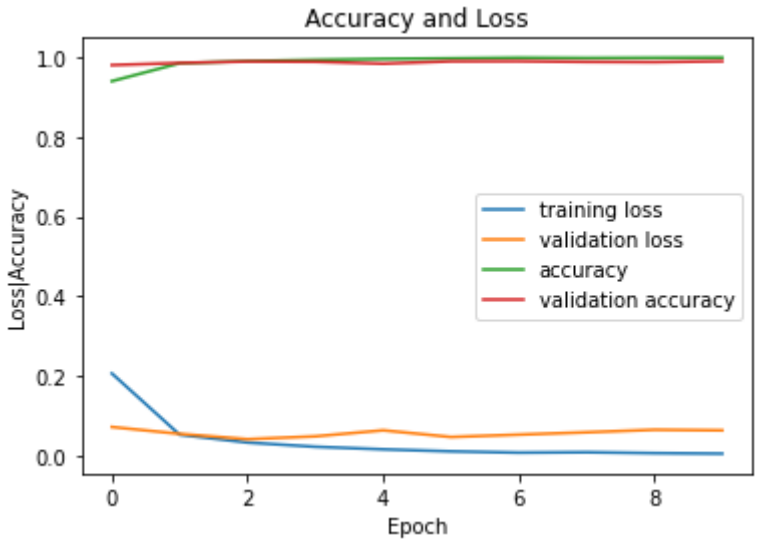
3.5.Triển khai mạng Convolutional Neural Network



Hình 12. *Mạng Convolutional Neural Network cho Mnist dataset*

Trong hình 14:

- Convolutional layer đầu tiên ta cần một đầu vào là kích thước của một ảnh – Input_shape(28,28,1). Hình ảnh được tích chập với 32 bộ lọc với size là (3,3) cho ra các features map.
- Convolutional layer tiếp theo được tích chập với 32 bộ lộc với size là (3,3) cho ra các features map tiếp theo
- Lớp tiếp theo là Pooling cho ra các features map có size là (14,14)
- Tiếp theo sử dụng Flatten layer để chuyển sang vector đưa vào Dense layer gồm có 128 node
- Lớp đầu ra gồm có 10 node với activation function là Softmax. Tương ứng với 10 lớp của tập dataset

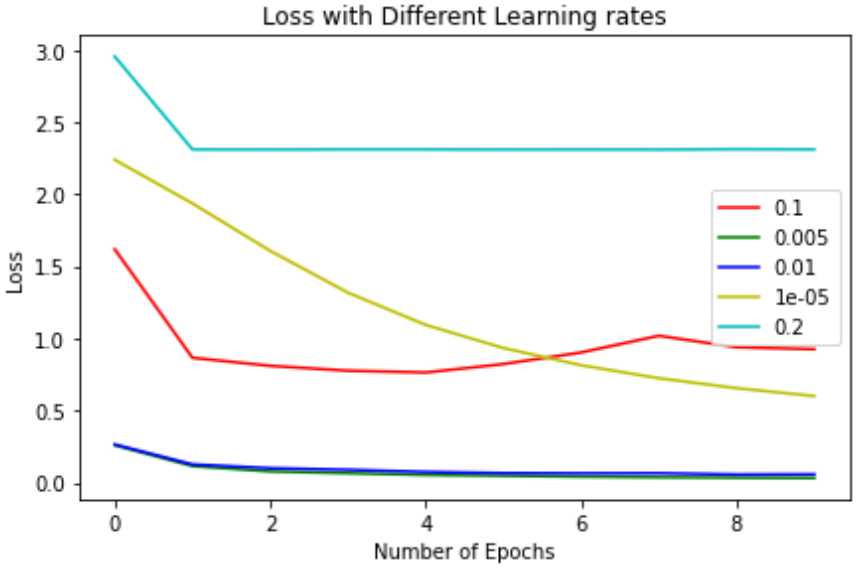


Hình 13. *Accuracy và loss trên tập train và tập validation của mạng CNN*

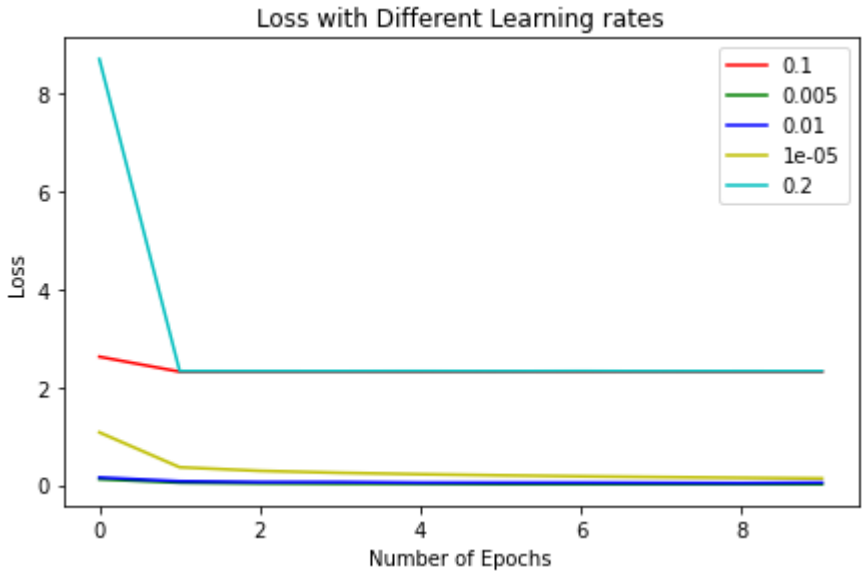
3.6.Thực nghiệm tinh chỉnh mô hình Neural Network và Convolutional Neural Network

3.6.1. Thực nghiệm thay đổi các learning_rate

Learning_rate (tốc độ học tập) là một siêu tham số kiểm soát mức độ thay đổi mô hình để đáp ứng với sai số ước tính mỗi khi trọng số của mô hình được cập nhật. Việc chọn tốc độ học là một thách thức vì giá trị quá nhỏ có thể dẫn đến quá trình luyện tập lâu dài có thể gặp khó khăn, trong khi giá trị quá lớn có thể dẫn đến việc học một bộ trọng lượng dưới tối ưu quá nhanh hoặc quá trình luyện tập không ổn định.



Hình 14. *Loss với các learning_rate khác nhau của mạng ANN*



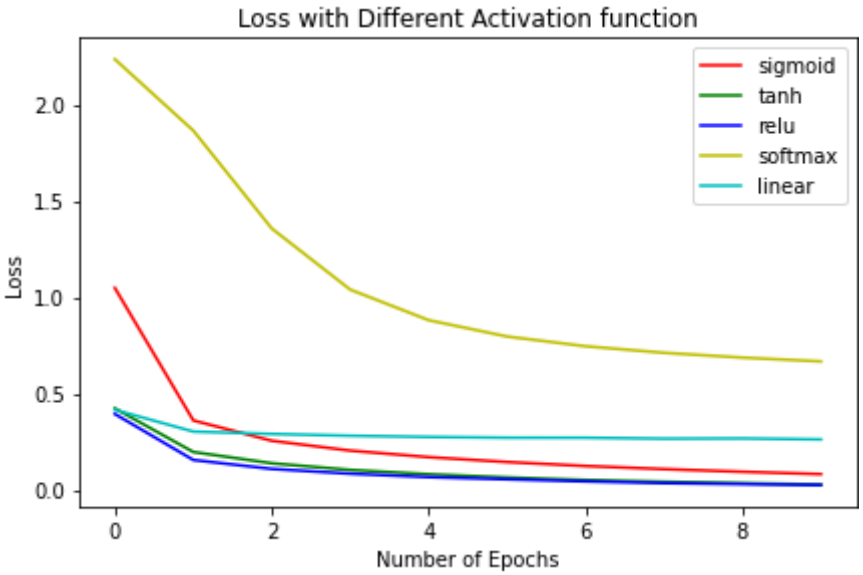
Hình 15. *Loss với các learning_rate khác nhau của mạng CNN*

Nhận xét:

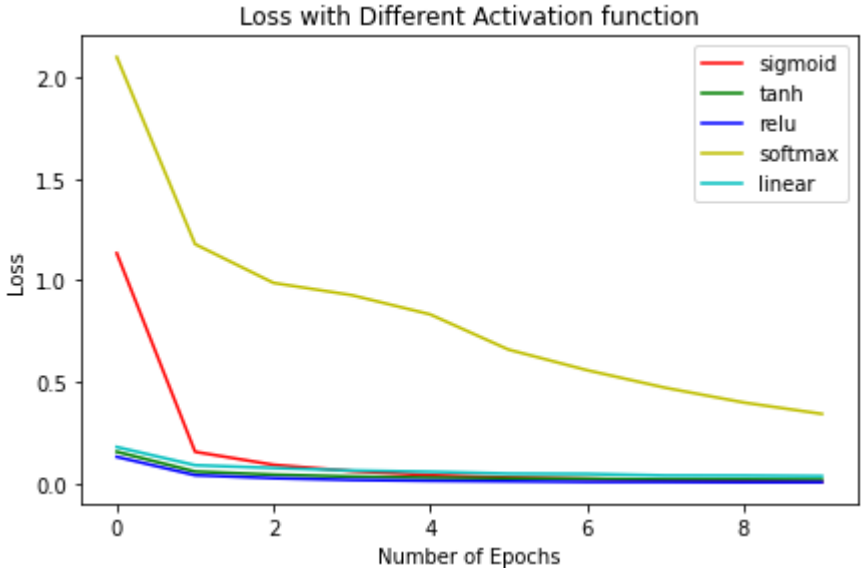
- Với learning_rate bằng 0.01 và 0.005 thì cả hai mạng hội tụ từ epoch đầu tiên cho Loss gần bằng 0
- Với leanring_rate bằng 0.1 và 0.2 thì cả mạng không cho Loss tốt (Loss cao) ➔ mô hình mạng sẽ không tốt
- Với learning_rate bằng 1e-05 thì mạng Neural Network hội tụ khá chậm. Còn mô hình Convolutional Neural Network hội tụ khá là tốt ở những epoch cuối cùng

Kết luận: qua các lần thực nghiệm với learning_rate khác nhau cho thấy tầm quan trọng của việc chọn learning_rate phù hợp cho mô hình để cho các kết quả tốt nhất

3.6.2. Thực nghiệm thay đổi Activation function



Hình 16. *Loss với các Activation function khác nhau của mạng ANN*



Hình 17. *Loss với các Activation function khác nhau của mạng CNN*

Nhận xét:

- Đối với hàm Relu, Tanh, Sigmoid đều cho Loss nhỏ, riêng hàm Relu và hàm Tanh cho Loss nhỏ từ những epoch đầu tiên

- Đối với mạng Neural Network với hàm Softmax và Linear cho kết quả của Loss sẽ không thay đổi sau một vài epochs đầu tiên của. Bởi vì hàm Softmax sẽ trả về xác suất còn hàm Linear là một đường thẳng. Nhưng đối với Convolutional Neural Network vẫn cho Loss gần bằng 0

4. KẾT QUẢ THỰC NGHIỆM

4.1.Metric đánh giá mô hình

Sau khi hoàn tất quá trình thực nghiệm, hiệu suất của hai mạng Neural Network và Convolutional Neural Network được đánh giá trên các chỉ số hiệu suất sau đây: Precision, Recall, F1-Score.

Precision: Mức độ dự báo chính xác trong những trường hợp được dự báo là Positive.

$$Precision = \frac{true\ positive}{true\ positive + false\ positive}$$

Recall: Mức độ dự báo chuẩn xác những trường hợp là Positive trong những trường hợp thực tế là Positive.

$$Recall\ = \frac{true\ positive}{true\ positive + false\ nagative}$$

F1-Score: Trung bình điều hòa giữa Precision và Recall. Đây là chỉ số thay thế lý tưởng cho accuracy khi mô hình có tỷ lệ mất cân bằng mẫu cao.

$$F1 - Score\ = \frac{2 * Precision * Recall}{Precision + Recall}$$

4.2. Kết quả

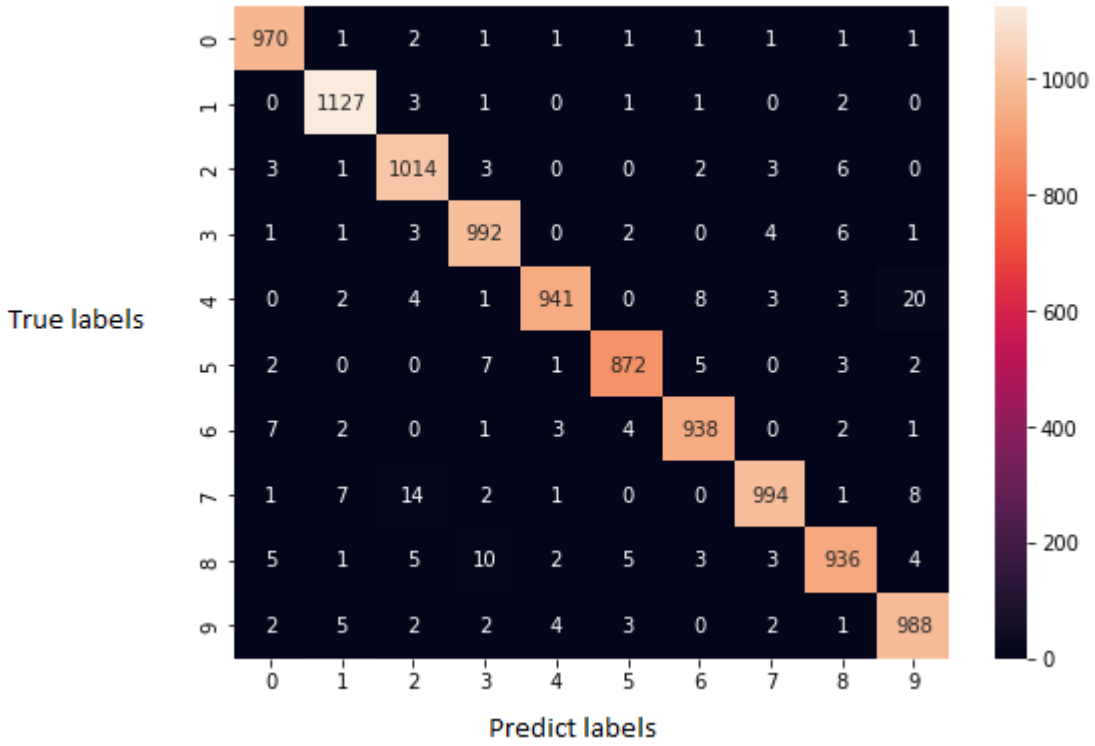
Sau khi chọn được các tham số như số node, learning_rate, activation function ta có kết quả tốt nhất của thực nghiệm cho các mô hình:

Bảng 1. *Loss và Accuracy của hai mô hình sau khi thực nghiệm*

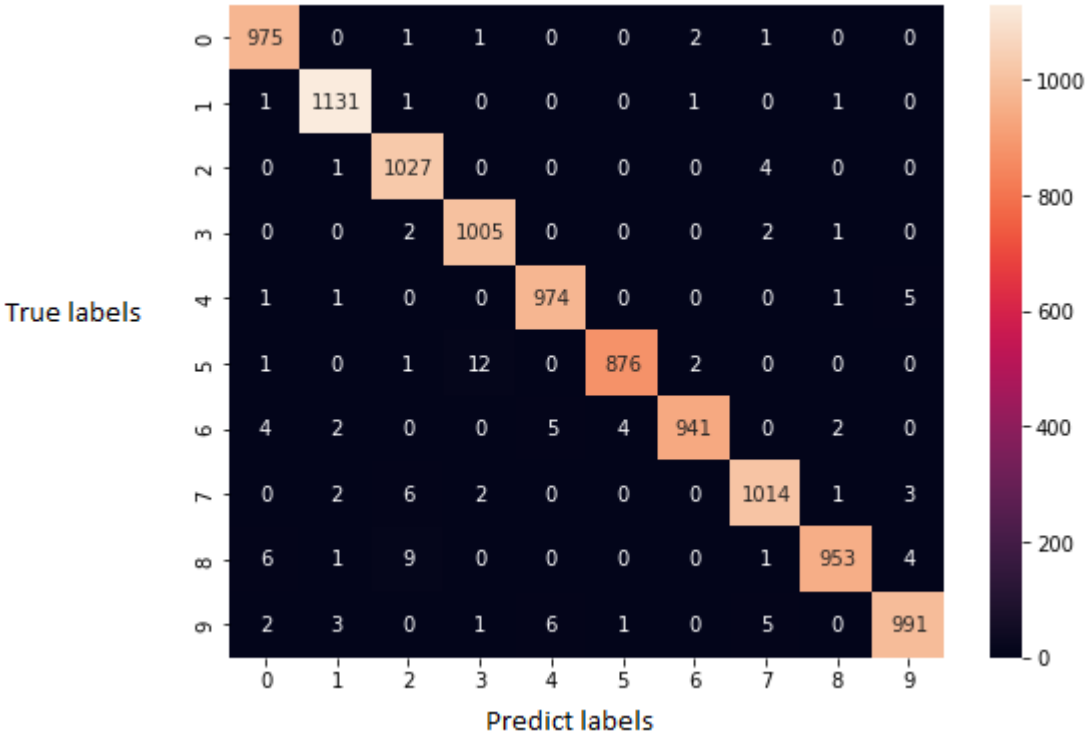
	Training Loss	**Training Accuracy**	**Validation Loss**	**Validation Accuracy**
Neural Network	0.0255	99.29%	0.0923	97.55%
Convolutional Neural Network	0.0053	99.81%	0.0553	98.90%

Bảng 2. *Đánh giá trên tập test của hai mô hình*

	Neural Network	**Convolutional Neural Network**
Loss	0.0779	0.045
Accuracy	97.7%	98.86%
Precision	97.7%	98.9%
Recall	97.7%	98.8%
F1-Score	97.7%	98.9%



Hình 18. *Confusion matrix của mạng Neural Network*



Hình 19. *Confusion matrix của mạng Convolutional Neural Network*

5. TÀI LIỆU THAM KHẢO

[1] PGS.TS Dương Tuấn Anh, Chương 6 – Classification – Trí tuệ nhân tạo, Trường Đại học Ngoại Ngữ Tin Học TPHCM

[2] PGS.TS Dương Tuấn Anh, Chương 8 – Deep Learning – Trí tuệ nhân tạo, Trường Đại học Ngoại Ngữ Tin Học TPHCM

[3] Sách Deep Learning cơ bản – Nguyễn Thanh Tuấn – Tải bản lần thứ 2

[4] Convolutional Neural Network với Keras

[5] Optimizer – Hiểu các thuật toán tối ưu

[6] Activation function trong mạng Neural Network