TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHTN
ĐHQG-HCM

fit@hcmus

Báo cáo

NHẬP MÔN HỌC MÁY

Đồ án cuối kì
Handwritten Digit Recognition

Giảng viên hướng dẫn

Giảng viên lí thuyết	Bùi Tiến Lên
Giảng viên thực hành	Nguyễn Ngọc Đức

Thông tin sinh viên

MSSV	Họ và Tên
19127292	Nguyễn Thanh Tình
19127303	Hình Ích Trình
19127378	Nguyễn Việt Thanh Duy
19127496	Trương Quang Minh Nhật
19127501	Trần Phạm Minh Nhựt

19KHMT1

Nhóm 7

TP. Hồ Chí Minh, Tháng 12 2021

THÔNG TIN NHÓM

MSSV	Họ và Tên	Công việc	Đánh giá
19127292	Nguyễn Thanh Tình	- Thiết kế slide và báo cáo. - Dựng video. - Lồng tiếng Phần 0. Introduction và Phần 1. Giới thiệu.	100%
19127303	Hình Ích Trình	- Biên soạn nội dung và kịch bản. - Lồng tiếng Phần 2. Các kĩ thuật liên quan.	100%
19127378	Nguyễn Việt Thanh Duy	- Biên soạn nội dung và kịch bản. - Tham gia deploy ứng dụng. - Lồng tiếng Phần 3. Phương pháp nghiên cứu.	100%
19127496	Trương Quang Minh Nhật	- Tester, người Demo. - Lồng tiếng Phần 5. Demo phần mềm nhận dạng chữ số viết tay.	100%
19127501	Trần Phạm Minh Nhựt	- Tham gia deploy ứng dụng. - Lồng tiếng Phần 4. Kết quả và Thảo luận.	100%

LỜI CẢM ƠN

Chân thành cảm ơn thầy **TS. Bùi Tiến Lên** – giảng viên phụ trách phần lí thuyết Nhập môn học máy – đã cung cấp cho cả lớp nói chung và nhóm chúng em nói riêng những kiến thức thật bổ ích trên từng tiết học, để nhóm chúng em có được nền tảng thực hiện tốt đồ án này.

Thành phố Hồ Chí Minh, ngày 28 tháng 12 năm 2021

Tập thể nhóm 7 – 19KHMT1

TỔ CHỨC HOẠT ĐỘNG NHÓM

Chủ đề và paper

- Chủ đề: Handwritten Digit Recognition
- Link paper: https://paperswithcode.com/paper/effective-handwritten-digit-recognition-using

Công cụ liên lạc

- Facebook Messenger.
- Zoom.

Quản lí file

- Google Drive.

Quản lí code

- GitHub.
  Link repository: https://github.com/tranphamminhnhut2203/ml-hcmus

Triển khai

- Streamlit.
  Link: https://share.streamlit.io/tranphamminhnhut2203/ml-hcmus/main.py

MỤC LỤC

0. ĐẶT VẤN ĐỀ

- Qua các ví dụ sau, hãy tìm ra vấn đề chung của những người sau đây:
  - Anh A là nhân viên văn thư tại một cơ quan nhà nước, hằng ngày nhận rất nhiều loại văn bản cần phải nhập lại vào phần mềm.
  - Chị B làm việc tại một đơn vị nghiên cứu thị trường nên phải tiếp nhận rất nhiều phiếu điều tra khảo sát ý kiến khách hàng.
  - Anh C làm việc tại một trường học, anh thường xuyên nhận phiếu dự thi từ thí sinh.
  - Chị D là nhân viên ngân hàng, chị phải nhập thông tin của khách hàng vào hệ thống phần mềm một cách thủ công để quản lý, bên cạnh đó còn có nhiều biểu mẫu thông tin khách hàng cần phải xử lý.
  - Chị E là sinh viên đại học, chị thường xuyên phải ghi chép lại các bài giảng trên lớp và phải lưu trữ lại để học tập.

- Qua các ví dụ trên, ta có thể thấy nhu cầu lưu trữ lại dữ liệu từ các văn bản viết tay là vô cùng lớn. Thế nhưng, việc nhập lại các trường thông tin 1 cách thủ công khiến con người tốn rất nhiều thời gian và dễ dẫn đến sai sót.

- Để giải quyết vấn đề nói trên, chúng ta cần phải xây dựng 1 phần mềm có khả năng nhận dạng các chữ viết tay. Phần mềm sẽ nhận vào hình ảnh của các văn bản viết tay từ đó nhận dạng và trích xuất ra được dữ liệu dưới dạng văn bản để lưu trữ.

- Đây là một bài toán khó bởi vì chữ viết của con người là vô cùng đa dạng, mỗi người sẽ có một nét chữ khác nhau. Muốn giải quyết được bài toán này chúng ta cần phải hiểu và vận dụng được một số kĩ thuật trong ML.

- Chúng ta sẽ cùng nhau tìm hiểu và xây dựng một phần mềm đơn giản hơn có khả năng nhận dạng các chữ số viết tay. Có thể xem đây là 1 bài toán kinh điển và rất phù hợp cho những người bắt đầu tiếp cận với công nghệ Machine Learning.

- Nội dung chính sẽ bao gồm các phần sau:
  - Giới thiệu.
  - Các kĩ thuật liên quan.
  - Phương pháp nghiên cứu.
  - Kết quả và thảo luận.
  - Demo phần mềm nhận dạng chữ số viết tay..

Nào chúng ta hãy cùng bắt đầu!

1. GIỚI THIỆU

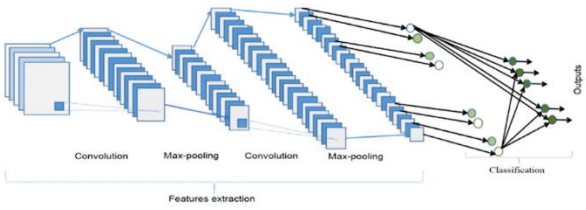
- Để nhận dạng chữ số viết tay, có nhiều phương pháp và kỹ thuật khác nhau như: logic mờ, giải thuật di truyền, mô hình xác suất thống kê, v.v… Từ lâu đã có nhiều công trình nghiên cứu về nhận dạng chữ số viết tay, tuy nhiên các ứng dụng vẫn chưa đáp ứng được hoàn toàn yêu cầu của người dùng.

- Hiện nay với sự phát triển không ngừng của máy tính, phương pháp học sâu ra đời đã đáp ứng cơ bản trong việc nhận dạng và xử lý ảnh.

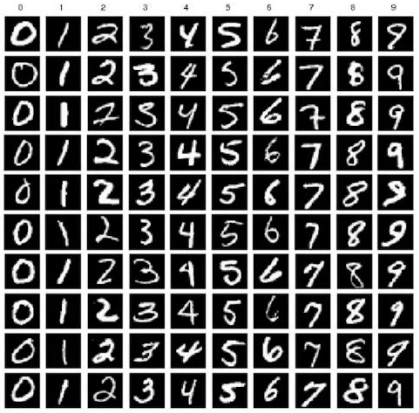
- Học sâu là một thuật toán dựa trên một số ý tưởng từ não bộ tới việc tiếp thu nhiều tầng biểu đạt, cả cụ thể lẫn trừu tượng. qua đó làm rõ nghĩa của các loại dữ liệu.

- Trong các bài toán nhận dạng thông qua hình ảnh, học sâu cho ta hiệu năng cũng như độ chính xác tương đối vượt trội so với các phương pháp phân lớp truyền thống.

- Nên trong video này, nhóm sẽ huấn luyện mô hình mạng CNN (một thuật toán học sâu) trên tập dữ liệu MNIST để xây dựng nên một mô hình có khả năng nhận dạng được các chữ số viết tay từ input dưới dạng hình ảnh.



Convolutional Neural Network (CNN)
(một thuật toán Deep Learning)



Tập dữ liệu MNIST

2. CÁC KỸ THUẬT LIÊN QUAN

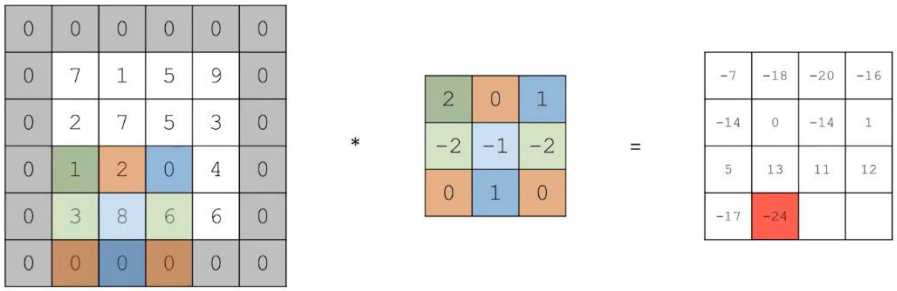
a. Giới thiệu CNN

- CNN là tên viết tắt của từ Convolutional Neural Network (hay còn gọi là mạng nơ ron tích chập). Đây là một trong những mô hình học sâu vô cùng tiên tiến. CNN sẽ cho phép ta xây dựng các hệ thống thông minh với độ chính xác vô cùng cao. Hiện nay, CNN được ứng dụng rất nhiều trong những bài toán nhận dạng object trong ảnh.
- Convolution - tích chập - được sử dụng đầu tiên trong xử lý tín hiệu số(signal processing). Nhờ vào nguyên lý biến đổi thông tin, các nhà khoa học đã áp dụng kĩ thuật này vào xử lý ảnh và video số. Để dễ hình dung, chúng ta có thể xem tích chập như một cửa sổ trượt (sliding window) áp đặt lên một ma trận.
- Mô hình mạng CNN:
  - Mô hình CNN chỉ đơn giản gồm một vài layer của convolution kết hợp với các hàm kích hoạt phi tuyến như ReLU hay tanh để tạo ra thông tin ở mức trừu tượng hơn cho các layer tiếp theo.
  - Trong mô hình mạng nơ-ron truyền thẳng (feedforward neural network), các layer kết nối trực tiếp với nhau thông qua vector trọng số w(weighted vector). Các layer này còn được gọi là có kết nối đầy đủ(fully connected layer) hay affine layer.
  - Trong mô hình CNN thì ngược lại. 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 nơron ở layer tiếp theo sinh ra từ filter áp đặt lên một vùng ảnh cục bộ của nơron layer trước đó. Mỗi layer như vậy được áp đặt các filter khác nhau, thông thường có vài trăm đến vài nghìn filter như vậy. Một số layer khác như pooling/subsampling layer dùng để chắt lọc lại các thông tin hữu ích hơn (loại bỏ các thông tin nhiễu).Layer cuối cùng được dùng để phân lớp ảnh
  - Trong suốt quá trình huấn luyện, CNN sẽ tự động học được các thông số cho các filter. Ví dụ,trong nhiệm vụ phân lớp ảnh, CNN sẽ cố gắng tìm ra thông số tối ưu cho các filter tương ứng theo thứ tự raw pixel > edges > shapes > facial > high-level features..
  - CNN có tính bất biến và tính kết hợp cục bộ(Location Invariance and Compositionality). Với cùng một đối tượng, nếu đối tượng này được chiếu theo các góc độ khác nhau thì độ chính xác của thuật toán sẽ bị ảnh hưởng đáng kể. Pooling layer sẽ cho tính bất biến đối với phép dịch chuyển (translation), phép quay (rotation) và phép co giãn (scaling). Tính kết hợp cục bộ cho ta các cấp độ biểu diễn thông tin từ mức độ thấp đến mức độ cao và trừu tượng hơn thông qua convolution từ các filter. Đó là lý do tại sao CNN cho ra mô hình với độ chính xác rất cao

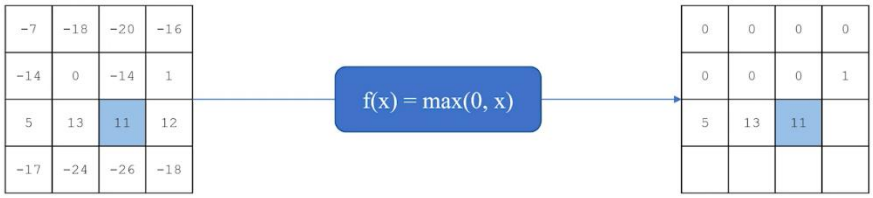
Tiếp theo, nhóm sẽ trình bày chi tiết các lớp trong mô hình:

Mô hình bắt đầu với Convolutional Layer. ReLU Layer thường được cài đặt ngay sau Convolutional Layer hoặc thậm chí kết hợp cả hai layer này thành một layer. Các layer tiếp theo có thể là Convolutional hay Pooling Layer tùy theo kiến trúc mà ta muốn xây dựng. Cuối cùng sẽ là Fully-Connected Layer để tiến hành phân lớp

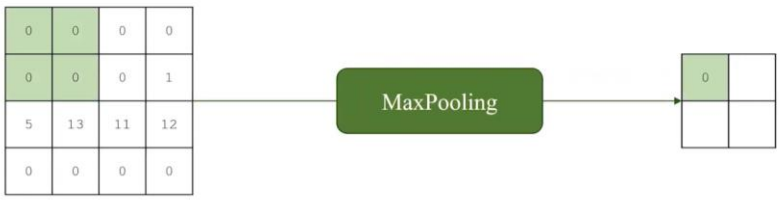
- Convolutional Layer:
  - Layer này chính là nơi thể hiện tư tưởng ban đầu của CNN. Thay vì kết nối toàn bộ điểm ảnh, layer này sẽ sử dụng một tập các bộ lọc (filters) có kích thước nhỏ so với ảnh để áp vào một vùng trong ảnh và tiến hành tính tích chập giữa bộ lọc và giá trị điểm ảnh trong vùng cục bộ đó. Bộ lọc sẽ lần lượt được dịch chuyển theo một giá trị bước trượt (stride) chạy dọc theo ảnh và quét toàn bộ ảnh.
  - Như vậy, với một bức ảnh 4x4 và một filter 3×3, ta sẽ có kết quả là một tấm ảnh mới có kích thước 4x4 là kết quả tích chập của filter và ảnh (với điều kiện đã thêm padding vào ảnh gốc để tính tích chập cho các trường hợp filter quét ra các biên cạnh). Với bao nhiêu filter trong lớp này thì ta sẽ có bấy nhiêu ảnh tương ứng mà lớp này trả ra và được truyền vào lớp tiếp theo. Các trọng số của filter ban đầu sẽ được khởi tạo ngẫu nhiên và sẽ được học dần trong quá trình huấn luyện mô hình.



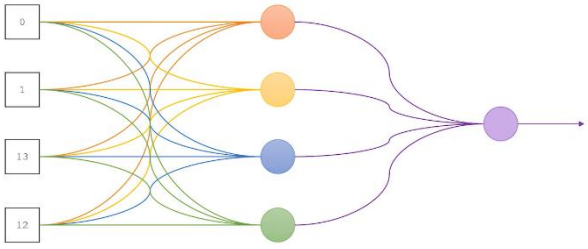
- Rectified Linear Unit (ReLU) Layer: Layer này thường được cài đặt ngay sau layer Convolution. Layer này sử dụng hàm kích hoạt f(x) = max(0, x). Nói một cách đơn giản, layer này có nhiệm vụ chuyển toàn bộ giá trị âm trong kết quả lấy từ lớp Convolution thành giá trị 0. Ý nghĩa của cách cài đặt này chính là tạo nên tính phi tuyến cho mô hình. Tương tự như trong mạng truyền thẳng, việc xây dựng dựa trên các phép biến đổi tuyến tính sẽ khiến việc xây dựng đa tầng đa lớp trở nên vô nghĩa. Có rất nhiều cách để khiến mô hình trở nên phi tuyến như sử dụng các hàm kích hoạt sigmoid, tanh,... nhưng hàm f(x) = max(0, x) dễ cài đặt, tính toán nhanh mà vẫn hiệu quả.



- Pooling Layer:
  - Layer này sử dụng một cửa sổ trượt quét qua toàn bộ ảnh dữ liệu, mỗi lần trượt theo một bước trượt (stride) cho trước. Khác với layer Convolution, layer Pooling không tính tích chập mà tiến hành lấy mẫu (subsampling). Khi cửa sổ trượt trên ảnh, chỉ có một giá trị được xem là giá trị đại diện cho thông tin ảnh tại vùng đó (giá trị mẫu) được giữ lại. Các phương thức lấy phổ biến trong layer Pooling là MaxPooling (lấy giá trị lớn nhất), MinPooling (lấy giá trị nhỏ nhất) và AveragePooling (lấy giá trị trung bình)
  - Với ảnh có kích thước 4×4 và layer Pooling sử dụng bộ lọc có kích thước 2×2 có bước trượt stride là 2, phương pháp sử dụng là MaxPooling. Mỗi lần trượt chỉ có giá trị lớn nhất trong 4 giá trị nằm trong vùng cửa sổ 2×2 của bộ lọc được giữ lại và đưa vào ma trận đầu ra. Như vậy,sau khi qua layer Pooling, ảnh sẽ giảm kích thước xuống còn 2×2 (kích thước mỗi chiều giảm 2 lần).
  - Pooling Layer có vai trò giảm kích thước dữ liệu. Với một bức ảnh kích thước lớn qua nhiều Pooling Layer sẽ được thu nhỏ lại tuy nhiên vẫn giữ được những đặc trưng cần cho việc nhận dạng (thông qua cách lấy mẫu). Việc giảm kích thước dữ liệu sẽ làm giảm lượng tham số, tăng hiệu quả tính toán và góp phần kiểm soát hiện tượng quá khớp (overfitting).



- Fully Connected (FC) Layer: Layer này tương tự với layer trong mạng nơ-ron truyền thẳng, các giá trị ảnh được liên kết đầy đủ vào các nơ-ron trong layer tiếp theo. Sau khi ảnh được xử lý và rút trích đặc trưng từ các layer trước đó, dữ liệu ảnh sẽ không còn quá lớn so với mô hình truyền thẳng nên ta có thể sử dụng mô hình truyền thẳng để tiến hành nhận dạng.



b. Giới thiệu về thư viện Keras:

Thư viện nhóm sử dụng đó chính là Keras.

- Keras là một thư viện được phát triển vào năm 2015 bởi François Chollet, là một kỹ sư nghiên cứu deep learning tại google. Nó là một open source cho neural network được viết bởi ngôn ngữ python. keras là một API bậc cao có thể sử dụng chung với các thư viện deep learning nổi tiếng như Tensorflow(được phát triển bởi gg), CNTK(được phát triển bởi microsoft),Theano(người phát triển chính Yoshua Bengio). Keras có một số ưu điểm như :
  - Dễ hiểu, dễ sử dụng
  - Xây dựng model nhanh.
  - Có thể chạy trên cả cpu và gpu
  - Hỗ trợ xây dựng CNN , RNN và có thể kết hợp cả 2.
- Có thể nói, Keras là một trong những lựa chọn tối ưu cho các ứng dụng học sâu. Vì thế nhóm quyết định sử dụng thư viện Keras vào trong phần cài đặt chương trình của nhóm.

c. Giới thiệu về tập dữ liệu MNIST

- Trong phần thực nghiệm, nhóm sử dụng tập dữ liệu MNIST (Yann LeCun, Corinna Cortes và Christopher, 1989). Đây là tập dữ liệu thường dùng để đánh giá hiệu quả của các mô hình nhận dạng ký tự số viết tay. Tập dữ liệu MNIST có nguồn gốc từ tập NIST do tổ chức National Institute of Standards and Technology (NIST) cung cấp, sau đó được LeCun cập nhật và chia thành 2 tập riêng biệt:
  - Tập dữ liệu huấn luyện gồm có 60.000 ảnh kích thước 28×28 của chữ số viết tay được dùng cho việc huấn luyện mô hình học máy. Tất cả các ảnh trong tập dữ liệu đều được căn chỉnh và biến đổi thành dữ liệu dạng điểm gồm 60.000 mẫu có 784 chiều là giá trị mức xám của các điểm ảnh thuộc về 10 lớp (giá trị từ 0 đến 9).
  - Tập dữ liệu kiểm tra gồm có 10.000 ảnh của chữ số viết tay được dùng cho việc kiểm thử.Các ảnh trong tập dữ liệu kiểm tra cũng được biến đổi và căn chỉnh tương tự tập dữ liệu huấn luyện.

3. PHƯƠNG PHÁP NGHIÊN CỨU

a. Ý tưởng

- Mô hình của nhóm sẽ là sự kết hợp giữa các layer của CNN với MLP. Trong đó,các layer của CNN được dùng để chắt lọc thông tin trong ảnh nhằm xây dựng vectơ đặc trưng dùng để phân loại ảnh. MLP đóng vai trò như một bộ phân loại, nhận đầu vào là vectơ đặc trưng xây dựng bởi các layer của CNN và đầu ra là các kết quả phân loại.
- Các layer của mạng cùng với chức năng của mỗi layer này:
  - Layer đầu tiên của mạng là một Input Layer, layer này chứa các ảnh cần phân loại. Mỗi ảnh là một ma trận xám có kích thước n n. Ví dụ,kích thước của ảnh là 28×28, khi đó mỗi ảnh có 784 phần tử, mỗi phần tử là một giá trị mức xám.
  - Layer tiếp theo của mạng là một Convolutional Layer được gọi là Conv2D. Layer này nhận đầu vào là các ảnh từ lớp Input. Trong Convolutional Layer, nhóm sử dụng nhiều filter với kích thước bằng nhau để quét trên ảnh đầu vào (từ Input layer) và tạo ra các ánh xạ đặc trưng cho ảnh.Sau Convolutional Layer, nhóm cũng sử dụng hàm kích hoạt ReLU.
  - Tiếp theo, nhóm định nghĩa một Pooling Layer có giá trị tối đa được gọi là MaxPooling2D. Layer này nhận đầu vào là kết quả của Convolutional Layer ở trên và nó thực hiện chắt lọc lại thông tin, loại bỏ thông tin nhiễu trước khi truyền cho layer tiếp theo của mạng. Trong phần thực nghiệm, nhóm sử dụng cửa sổ với kích thước pool size là 2×2 để lấy giá trị lớn nhất trong 4 giá trị mà cửa sổ này quét qua trên ma trận đầu ra của Convolutional Layer.
  - Sau Pooling Layer, nhóm sử dụng một Dropout Layer với giá trị Dropout được thiết lập là 0,25. Nó được cấu hình để loại trừ ngẫu nhiên 25% tổng số các nơ-ron trong layer để giảm vấn đề overfitting.
  - Để chắt lọc được nhiều thông tin hữu ích từ ảnh, nhóm xây dựng mạng CNN sâu hơn bằng cách bổ sung thêm một số layer của mạng, chúng bao gồm các layer sau: Convolutional, Pooling, Dropout. Trong đó,Convolutional Layer sẽ sử dụng nhiều filter với kích thước bằng nhau, Pooling Layer sử dụng cửa sổ với kích thước pool size là 2×2, Dropout Layer với giá trị Dropout là 0,25
  - Tiếp theo,nhóm sử dụng một layer chuyển đổi dữ liệu ma trận 2D thành một véc-tơ gọi là Flatten. Kết quả nhóm thu được một vectơ các giá trị đặc trưng của ảnh, véc-tơ này phù hợp với định dạng đầu vào của một MLP.
  - Sau khi thu được vectơ đặc trưng của ảnh qua các layer của CNN, nhóm sử dụng MLP làm bộ phân loại để phân loại ảnh. MLP của nhóm sử dụng nhiều layer ẩn với số nơron được cấu hình trong quá trình thực nghiệm. Do đây là nhiệm vụ phân loại đa lớp (10 lớp) nên đầu ra của MLP nhóm sử dụng 10 nơron và một hàm kích hoạt softmax để đưa ra các dự đoán là các giá trị xác suất cho mỗi lớp

b. Cài đặt

- Chuẩn bị dữ liệu cho huấn luyện mô hình:
  - Để tải tập dữ liệu MNIST về máy tính, nhóm sử dụng thư viện học sâu của Keras. Tập dữ liệu được tải xuống tự động lần đầu tiên được gọi và lưu trữ trong thư mục chính của người dùng trong ~/.keras/datasets/mnist.pkl.gz dưới dạng một tệp tin. Điều này rất thuận tiện cho việc thực nghiệm các mô hình học máy.

```
import numpy
from tensorflow import keras
from keras.datasets import mnist
import matplotlib.pyplot as plt
(X_train, y_train), (X_test, y_test) = mnist.load_data()
```

- Xây dựng và huấn luyện mô hình:
  - Sau khi dữ liệu MNIST đã được tải về máy tính, nhóm sẽ huấn luyện một mô hình DCNN trên tập dữ liệu này. Trong phần này, nhóm sẽ xây dựng một DCNN bằng cách kết hợp các lớp của CNN với MLP
  - Trong các thực nghiệm, nhóm xây dựng mô hình với việc áp dụng một số thuật toán tối ưu hóa và thiết lập các giá trị mức học (learning rate) khác nhau. nhóm cũng thiết lập kích thước cho các filter lần lượt là 5×5 và 3×3. Dưới đây là mô hình huấn luyện với việc sử dụng hàm lỗi logarithmic, thuật toán tối ưu hóa adam và giá trị learning_rate được thiết lập là 0,01

```
def DCNN1_model():
    model = keras.Sequential()
    model.add(keras.layers.Conv2D(64, (5, 5), input_shape=(28,28,1), activation = 'relu'))
    model.add(keras.layers.MaxPooling2D(pool_size=(2, 2)))
    model.add(keras.layers.Dropout(0.25))
    model.add(keras.layers.Conv2D(32, (3, 3), activation= 'relu'))
    model.add(keras.layers.MaxPooling2D(pool_size=(2, 2)))
    model.add(keras.layers.Dropout(0.25))
    model.add(keras.layers.Flatten())
    # MLP with 3 hidden layer
    model.add(keras.layers.Dense(375, activation='relu'))
    model.add(keras.layers.Dropout(0.25))
    model.add(keras.layers.Dense(225, activation='relu'))
    model.add(keras.layers.Dropout(0.25))
    model.add(keras.layers.Dense(135, activation='relu'))
    model.add(keras.layers.Dropout(0.25))
    model.add(keras.layers.Dense(10, activation= 'softmax'))
    model.compile(loss = 'sparse_categorical_crossentropy',optimizer = keras.optimizers.Adam(learning_rate=0.01), metrics=['accuracy'])
    return model
```

- Đánh giá mô hình
  - Để đánh giá hiệu suất của mô hình đã xây dựng, nhóm sử dụng tập dữ liệu kiểm tra MNIST. nhóm đánh giá mô hình với các giá trị epoch là 10 và batch size là 256.

```
model = DCNN1_model()
model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=10, batch_size=256, verbose=2)
scores = model.evaluate(X_test, y_test, verbose=0)
print("DCNN1 Error: %.2f%%" % (100-scores[1]*100))
```

4. KẾT QUẢ VÀ THẢO LUẬN

a. Đây là kết quả sau khi chạy chương trình

```
✓ 7m 22.3s
Epoch 1/10
235/235 - 44s - loss: 2.8847 - accuracy: 0.3707 - val_loss: 0.4989 - val_accuracy: 0.8557 - 44s/epoch - 186ms/step
Epoch 2/10
235/235 - 42s - loss: 0.5287 - accuracy: 0.8523 - val_loss: 0.1620 - val_accuracy: 0.9591 - 42s/epoch - 179ms/step
Epoch 3/10
235/235 - 44s - loss: 0.3470 - accuracy: 0.9101 - val_loss: 0.1448 - val_accuracy: 0.9620 - 44s/epoch - 188ms/step
Epoch 4/10
235/235 - 43s - loss: 0.3167 - accuracy: 0.9204 - val_loss: 0.1084 - val_accuracy: 0.9693 - 43s/epoch - 184ms/step
Epoch 5/10
235/235 - 45s - loss: 0.2951 - accuracy: 0.9262 - val_loss: 0.1181 - val_accuracy: 0.9727 - 45s/epoch - 192ms/step
Epoch 6/10
235/235 - 42s - loss: 0.3022 - accuracy: 0.9269 - val_loss: 0.1041 - val_accuracy: 0.9731 - 42s/epoch - 180ms/step
Epoch 7/10
235/235 - 42s - loss: 0.3063 - accuracy: 0.9283 - val_loss: 0.1297 - val_accuracy: 0.9716 - 42s/epoch - 181ms/step
Epoch 8/10
235/235 - 45s - loss: 0.3097 - accuracy: 0.9273 - val_loss: 0.1202 - val_accuracy: 0.9729 - 45s/epoch - 190ms/step
Epoch 9/10
235/235 - 45s - loss: 0.3075 - accuracy: 0.9291 - val_loss: 0.1039 - val_accuracy: 0.9738 - 45s/epoch - 193ms/step
Epoch 10/10
235/235 - 46s - loss: 0.2922 - accuracy: 0.9323 - val_loss: 0.0940 - val_accuracy: 0.9776 - 46s/epoch - 198ms/step
DCNN1 Error: 2.24%
```

b. Thảo luận

- Có thể thấy sau 10 lần chạy, nhóm đã huấn luyện được mô hình CNN với độ lỗi chỉ có 2.24%.
- Tổng thời gian huấn luyện là 7 phút 22 giây

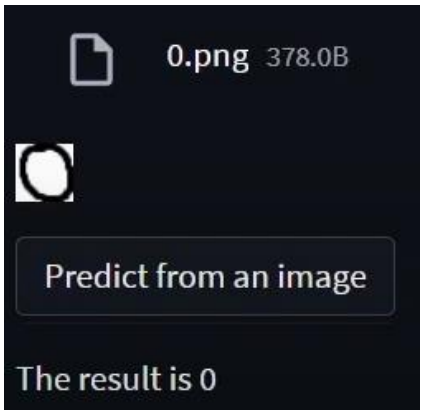
Sau đó nhóm đã thử huấn luyện lại mô hình với các cấu hình mạng khác nhau và thu được bảng tóm tắt độ chính xác phân loại, tỷ lệ lỗi và thời gian trung bình khi thực thi như sau:

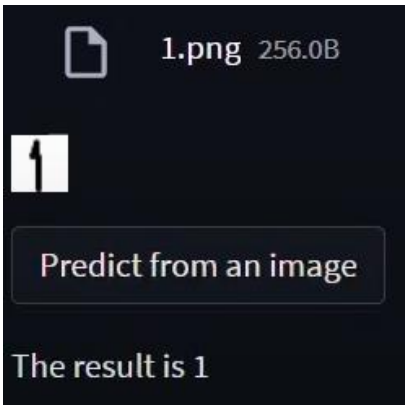
Cấu hình mạng			Kết quả phân loại		Thời gian trung bình thực hiện 1 epoch (giây)
Số filter trong mỗi Convolutional Layer	Hàm tối ưu hóa	Mức học	Độ chính xác cao nhất (%)	Tỷ lệ lỗi (%)	
32; 16	sgd	0,01	96,85	3,13	402,1
32; 16	adam	0,01	99,20	0,80	367,8
32; 16	adam	0,05	99,15	0,85	386,0
32; 16	adam	0,001	99,06	0,94	397,2
32; 16	adam	0,005	99,09	0,91	463,0
64; 32	adam	0,01	99,34	0,74	844,5

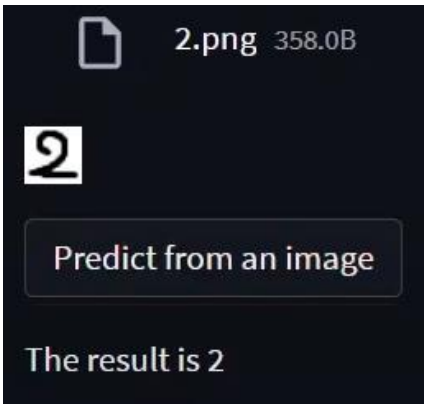
- Từ các kết quả thực nghiệm cho thấy, mô hình nhóm xây dựng đạt kết quả phân loại cao nhất với độ chính xác là 99,34% và tỷ lệ lỗi thấp nhất là 0,74% khi sử dụng các filter với số lượng tương ứng là 64 và 32, hàm tối ưu hóa adam, mức học là 0,01.
- Trong các thực nghiệm nhóm cũng thấy rằng thời gian để huấn luyện mô hình tăng đáng kể khi tăng số lượng các filter trong mỗi Convolutional Layer. Cụ thể khi sử dụng các filter với số lượng là 32 và 16 trong các Convolutional Layer, mô hình sẽ mất khoảng 367,8 giây cho mỗi epoch, trong khi tăng số filter lên 64, 32 thì thời gian cho mỗi epoch là khoảng 844,5 giây. Như vậy,để đạt được kết quả cao thì đòi hỏi phải mất nhiều thời gian để huấn luyện mô hình

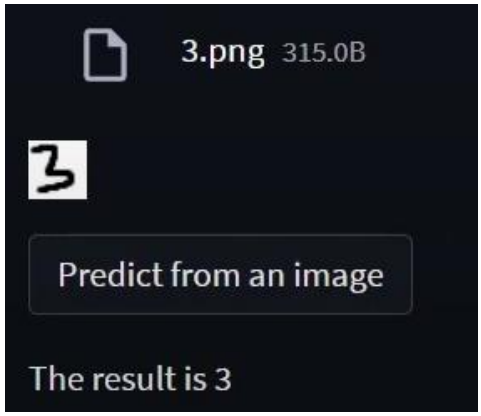
5. DEMO

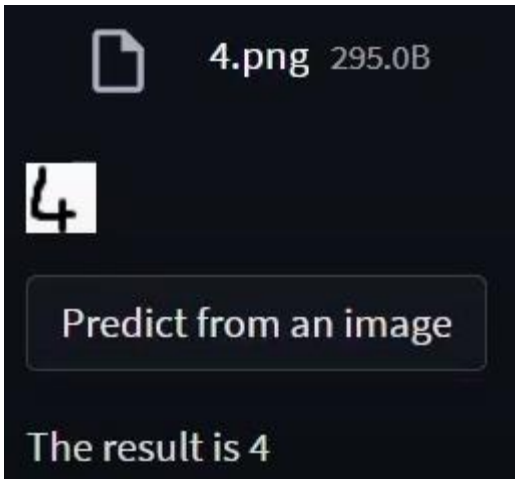
Ngoài việc test những dữ liệu ở trong MNIST Test, nhóm còn sủ dụng những dữ liệu tự tạo để kiểm định hoạt động của mô hình. Nhóm đã cố vẽ những trường hợp khó nhận dạng, và kết quả vẫn thu về được sự chính xác của mô hình.

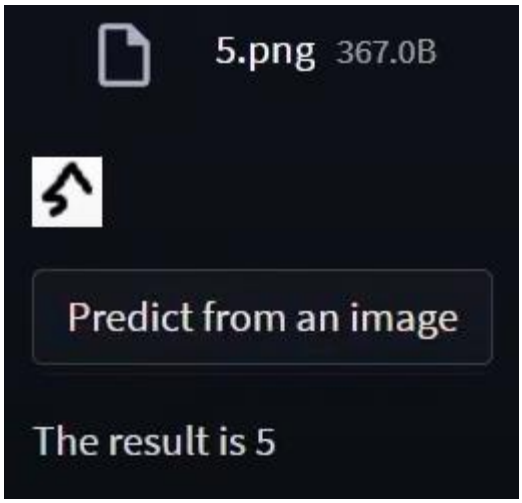




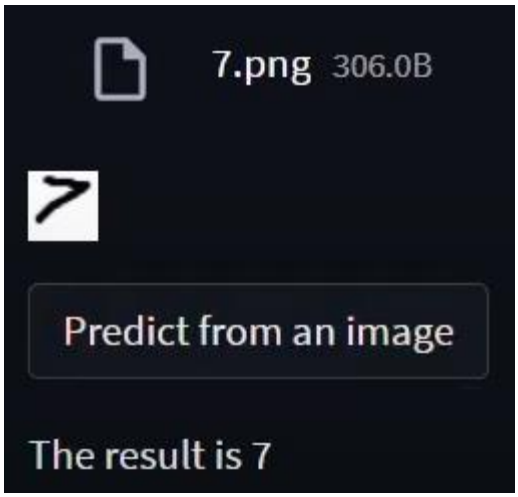


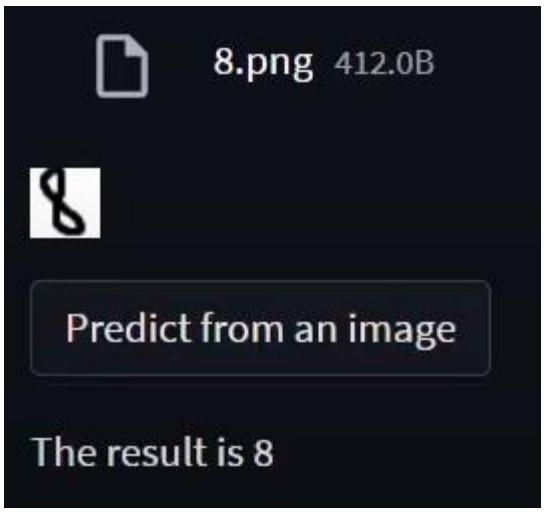


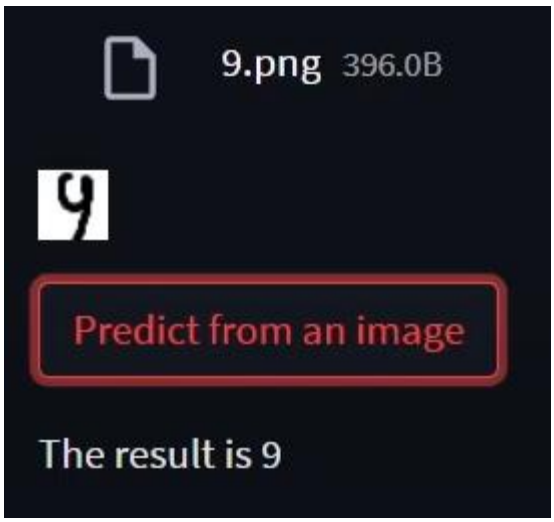

4.png 295.0B
Predict from an image
The result is 4


5.png 367.0B
Predict from an image
The result is 5


6.png 371.0B
Predict from an image
The result is 6


7.png 306.0B
Predict from an image
The result is 7


8.png 412.0B
Predict from an image
The result is 8


9.png 396.0B
Predict from an image
The result is 9

TỔNG KẾT

Vậy là nhóm đã giới thiệu cũng như đề xuất phương pháp nhằm giải quyết bài toán nhận dạng chữ số viết tay một cách chi tiết, đầy đủ và rõ ràng. Nhóm cũng đã xây dựng mô hình học sâu với việc kết hợp các lớp của CNN với MLP và huấn luyện trên tập dữ liệu chữ số viết tay thông dụng MNIST. Các kết quả khi huấn luyện và kiểm tra mô hình đều mang lại tính đúng đắn và sự hiệu quả. Từ thành công bước đầu với đề tài có ý nghĩa này, đây sẽ là bước đệm để có thể dễ dàng tiếp cận với những chủ đề phức tạp hơn, có thể đó là nhận dạng được các ký tự viết tay của con người.

ĐỒ ÁN CUỐI KÌ

Nhập môn học máy
19KHMT1

Giảng viên hướng dẫn

TS. Bùi Tiến Lên

NHÓM 7

19127292	Nguyễn Thanh Tình
19127303	Hình Ích Trình
19127378	Nguyễn Việt Thanh Duy
19127496	Trương Quang Minh Nhật
19127501	Trần Phạm Minh Nhựt

Paper tham khảo

Handwritten Digit Recognition

https://paperswithcode.com/paper/effective-handwritten-digit-recognition-using

fit@hcmus

TÀI LIỆU THAM KHẢO

[1] Paper Handwritten Digit Recognition
https://paperswithcode.com/paper/effective-handwritten-digit-recognition-using

[2] CNN Explainer
https://poloclub.github.io/cnn-explainer/

----------Hết----------