Họ tên: Ngô Đức Tuấn

MSSV: 18520186

Môn học: Nhận dạng

Mã môn: CS338.L22.KHCL

**BÁO CÁO VỀ PHƯƠNG PHÁP SỬ DỤNG CHO BÀI TOÁN NHẬN DẠNG CHỮ SỐ VIẾT TAY (TẬP DATASET MNIST)**

**1. Giới thiệu về dataset**

Dataset sử dụng về phân loại chữ số viết tay (Handwritten digit classification) từ **MNIST**, chứa khoảng 60000 ảnh được dùng để train và 10000 ảnh được dùng để test để phân loại thành 10 loại (tương ứng từ chữ số 0 đến 9). Độ phân giải thấp (28 by 28 pixels).

Để load dataset, ta có thể dùng chính thư viện *keras* đã được tích hợp sẵn. Tuy nhiên, trong bài này em sẽ load data bằng 2 file .csv là ‘train\_new.csv’ và ‘test\_new.csv’ được chia sẻ trên drive của trang submission.

*Mục tiêu*: xác định chữ số trong bức ảnh đầu vào. Trong tập dữ liệu đánh giá (test set), mô hình phải đoán đúng nhãn của input.

*Độ đo*: Độ đo phổ biến trong phân loại và nhận diện ảnh là độ chính xác (accuracy).

**2. Giới thiệu về kiến trúc mạng Lenet**

*2.1 Giới thiệu về mạng Lenet*

Về thuật toán cài đặt em dùng mạng CNN (Convolutional Neural Network), với kiến trúc mạng đó là Lenet. Đây là mạng CNN đầu tiên được giới thiệu bởi Yann Lecun vào năm 1998, cho bài toán nhận dạng chữ số viết tay.

Cấu trúc mạng gồm kích thước ảnh đầu vào là (28,28,3), 2 tầng (Layers) đó là Convolution + MaxPooling lần lượt giảm chiều kích thước xuống (14,14) và (5,5), 2 layer fully connected xen giữa là flatten và output là softmax layers để đưa về dạng phân bố xác suất (probability distribution).

*2.2 Thiết lập mô hình mạng Lenet*

Sau đây là kiến trúc mạng Lenet do em thiết kế. Đầu vào của mạng của kích thước (28,28,1). Mạng có 2 tầng/lớp tích chập Convolution 2D xen kẽ với tầng activation là ReLu và 2 tầng Pooling là Max Pooling và 1 tầng Fully Connected với output là softmax layer.

+ C1 có kernel size=(4, 4), padding=‘same’, number filters=32, tầng activation sử dụng là ‘relu’ => output=(28x28x32).

+ C2 có kernel size=(3, 3), padding= ‘same’, number filters=64, tầng activation sử dụng là ‘relu’ => output=(14,14,64).

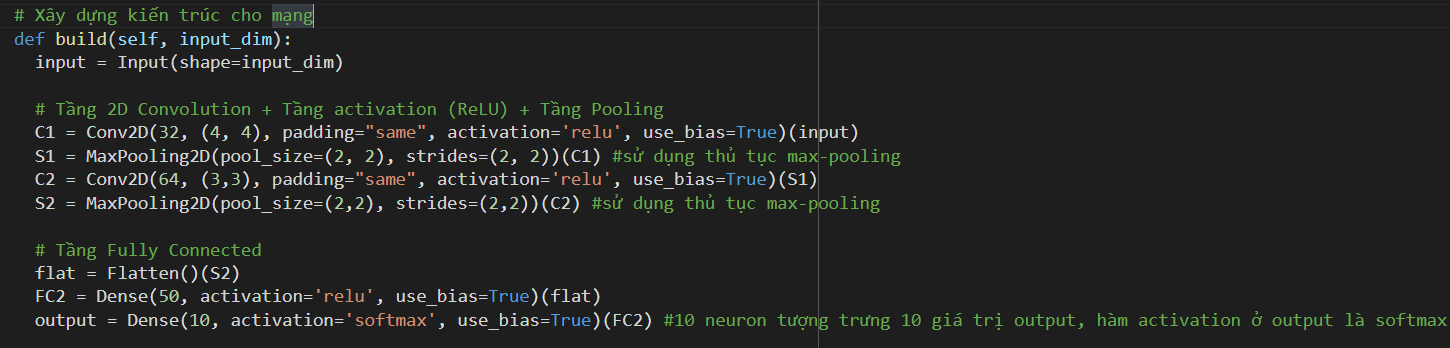
+ S1 (maxpooling layer 1) có pooling size=2x2, stride=2, no padding => output=(14,14,32).

+ S2 (maxpooling layer 2) có pooling size=2x2, stride=2, no padding => output=(7,7,64).

+ Flatten: output = 7x7x64 = 3136.

+ FC1 = Output = 50, activation = ‘relu’.

+ Softmax layer, output = 10 (tương ứng 10 digits).



*Ảnh 1: Thiết lập kiến trúc mạng Lenet*

Hiện tại, sau khi load dataset file .csv lên, em thấy kích thước của tập train là (29400, 785) tương ứng số lượng dữ liệu là 29400, số lượng cột là 785 cột (784 cột ứng với 28\*28 = 784 pixels nhận được trong khoảng từ 0-255 và 1 cột label chứa thông tin nhãn chữ số) và tương tự với tập test là (12600, 784). Để có thể đưa vào mô hình Lenet để huấn luyện, ta cần phải reshape về dạng kích thước (28,28). Về cách thực hiện chi tiết, em có trình bày trong phần notebook.

*2.3 Biên dịch mô hình*

Trước khi mô hình có thể được huấn luyện, chúng ta cần thêm các hàm toán học giúp cho mô hình đi “đúng hướng”. Đây được gọi là quá trình biên dịch mô hình (compile).

+ *Hàm mất mát* – dùng để tính toán “độ sai sót” của mô hình trong quá trình huấn luyện và dự đoán. Ta cần phải làm cho nghiệm hàm này nhỏ nhất để kiểm soát mô hình đi đúng mục tiêu, mất mát càng ít, chính xác càng cao. Hàm mất mát (loss function) em sử dụng là `crossentropy`.

+ *Hàm tối ưu* – là “công cụ” để làm hàm mất mát có nghiệm nhỏ nhất, qua việc cập nhật và chỉnh sửa các parameters có trong hàm mất mát. Hàm tối ưu (optimizer function) em sử dụng là `Stochastic Gradient Descent (SGD)`

+ *Thang đo* – dùng để đo đạc độ chính xác qua mỗi quá trình huấn luyện và kiểm tra. Em sử dụng thang đo (metrics) là `accuracy`, vì đây là thang đo phổ biến nhất trong các bài toán về phân loại và nhận dạng ảnh.

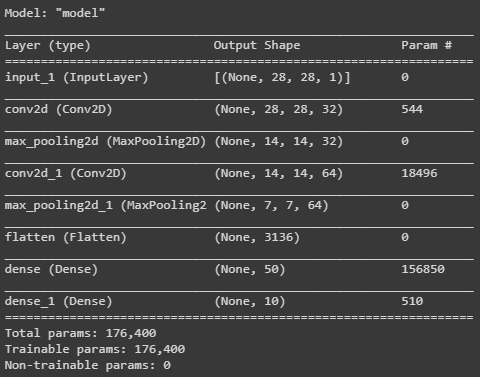
Ngoài ra về các bước khai phá dữ liệu (data mining), tiền xử lý dữ liệu (preprocessing data), tách dữ liệu train-test (train-test-split), em sẽ trình bày cụ thể trong phần notebook.

**3. Train mô hình và đánh giá mô hình**

Em train mô hình với thông số như sau, batch\_size = 128, epochs = 30, verbose = 1.

Sau khi thực hiện epochs 30 lần, em được kết quả fit trên tập train có độ chính xác (accuracy) là 98.23% và đánh giá trên tập test là 97.53%.

Đây là kiến trúc mô hình của em (các tầng cụ thể với output shape) đã được em trình bày ở phần 2.2. Tổng số params tham gia là 176400.



*Ảnh 2: Kiến trúc mô hình*

**4. Submit trên hệ thống submission** <http://submission.aiclub.cs.uit.edu.vn/>

Sau khi submit trên hệ thống, kết quả em nhận được là 0.97587 và xếp hạng 27/156.

Graphical user interface, application

Description automatically generated

*Ảnh 3: Kết quả submit lên hệ thống submission*