**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

**KHOA CÔNG NGHỆ THÔNG TIN**

**BỘ MÔN**: KHOA HỌC MÁY TÍNH

**AI SYSTEM**

**HANDWRITTING RECOGNITION USING TENSORFLOW**

**Giáo viên hướng dẫn:**

Nguyễn Ngọc Thảo

**Thông tin sinh viên:**

**Tên:** Bùi Trọng Xuyến

**Mssv:** 1612835

**Email:** btxuyenhcmus@gmail.com

**Tên:** Lương Công Văn

**Mssv:** 1612804

**Email:** congvan2498@gmail.com

**Tên:** Phạm Hoàng Tín

**Mssv:** 1612696

**Email:** Phamhoangtin1998@gmail.com

**tp. HCM, 26/04/2019**

1. **GIỚI THIỆU**

* Trong hệ thống nghiên cứu AI lần này, chúng tôi sẽ thiết lập và huấn luyện một tensorflow-AI để phân loại “NIST Special database 19”. Hệ thống này là một mạng thần kinh liên hợp (CNN). Nó phân biệt giữa 47 lớp: tất cả các chữ cái viết hoa, tất cả các số và một vài chữ cái viết thường.
* Giới thiệu về bộ NIST Special database 19:
* Chứa toàn bộ tài liệu đào tạo của NIST cho tài liệu in dấu tay và nhận dạng ký tự. Nó xuất bản với mẫu dấu tay từ 3600 nhà văn, 810,000 hình ảnh ký tự được phân lập từ biểu mẫu của họ.
* Version của bộ dữ liệu trong hệ thống lần này được lấy từ nguồn sau:

[*https://s3.amazonaws.com/nist-srd/SD19/by\_merge.zip*](https://s3.amazonaws.com/nist-srd/SD19/by_merge.zip)

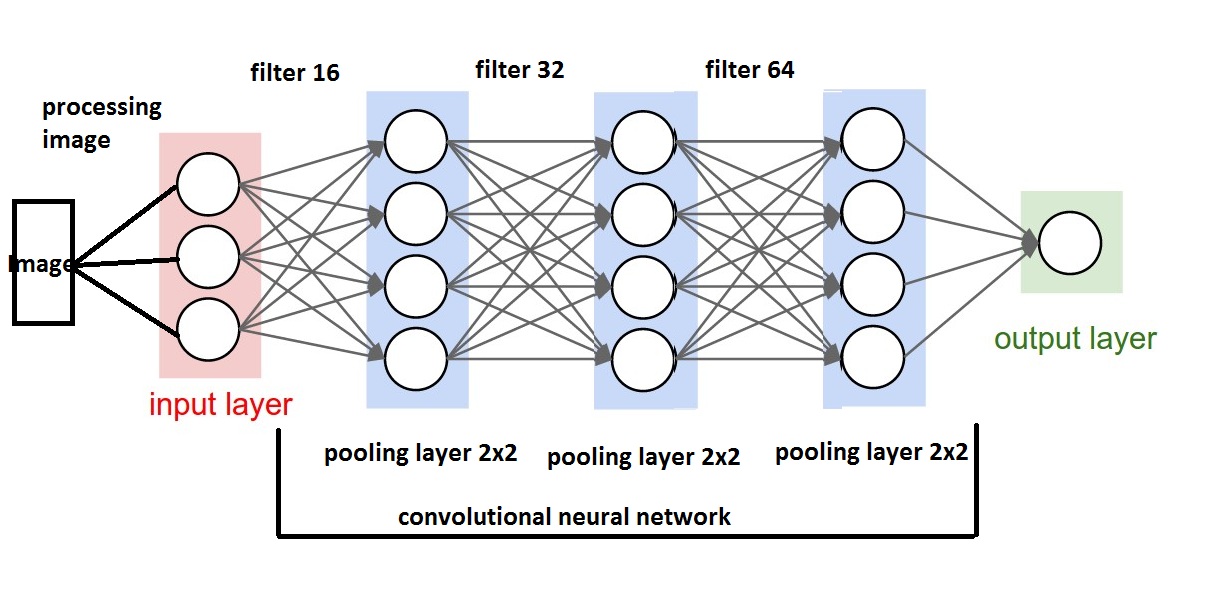
“Dung lượng 530.013 KB”

* Convolutional neural network (CNN):
* Giống với mạng neural thông thường, chúng được tạo từ các neural(*Node*) có trọng lượng (*w*). Mỗi neural sẽ nhận được một giá trị đầu vào và thực hiện “function activation”. Toàn bộ mạng theo một sơ đồ duy nhất, từ các pixel thô ban đầu sẽ qua các “hidden layer” với các trọng số khác nhau, số lượng “hidden layer” càng nhiều hay độ phức tạp của mạng càng nhiều thì độ chính xác càng cao, sau đó sẽ được phân lớp và cho ra output đúng.
* Mạng neural nhận được một đầu vào và biến đổi nó thông qua một loạt các “hidden layer”. Một “hidden layer” được tạo ra từ một tập hợp các neural, trong đó một neural được kết nối hoàn toàn (hoặc một số) neural ở lớp trước, mỗi neural trong một lớp hoạt động độc lập và không chia sẻ tài nguyên cho bất kì neural nào trong lớp đó.



***Hình 1.*** *Neural nerwork*

* Sơ đồ khối hệ thống xây dựng lần này



***Hình 2:*** *sơ đồ khối hệ thống*

1. **PREPROCESSING THE DATA**

* Tải xuống cơ sỡ dữ liệu “by\_merge.zip” sau đó sử dụng một tập lệnh nhỏ để đặt chúng vào cùng một thư mục và đổi tên chúng theo định dạng “class\_[class]\_index\_[index].png” (ví dụ như class\_47 \_index\_001.png). Việc đặt tên hình ảnh như vậy để đảm cấu trúc xử lý ảnh.
* Nguồn dữ liệu của database đã được cung cấp ở phần giới thiệu. Vì kích thước qua lớn nên không thể đính kèm theo tài liệu này.
* Việc đổi tên dữ liệu có thể thực hiện bằng tay hoặc mã code được cung cấp cấp trong tệp “rename\_images.py”

*copyfile(os.path.join(subdir, file),*

*os.path.join(output\_dir, "class\_{}\_index\_{}.png".format(class\_index, index)))*

* Sử dụng “opencv” để chuyển đổi những hình ảnh đã theo cấu trúc quy định này thành 2 mảng. Một mảng chứa các hình ảnh, được định kích cỡ lại thành 32x32 và thang độ xám với dạng (num\_images, 32, 32, 1). Mảng khác chứa label của các hình ảnh đó có dạng (num\_labels, 47). Tập lệnh để chuyển đổi hình ảnh trong tệp “data\_handler.py” được kèm theo tài liệu này.
* Trong thao tác này ta có sử dụng thêm thư viện “numy” của python thì thư viện này là một thư viện cực kì mạnh về các thao tác mảng, ma trận, vector,...
* Opencv là 1 framework được cung cấp để xử lý hình ảnh liên quan đến lĩnh vực computer vision.

Môi trường window cài đặt opencv như sau:

*C:\Users\Administrator>pip install opencv-python*

Hoặc

*C:\Users\Administrator>pip install opencv-python==”version”*

Sẽ có 2 trường hợp xảy ra là máy bạn đã tồn tại opencv hoặc là nó sẽ tải các gói site-packages hỗ trợ opencv về và bạn có thể sử dụng nó ngay với câu lệnh

>>*import cv2 as cv*

* Chúng ta tạo mảng thứ nhất

*def get\_2d\_array(im\_path):*

*"""Return image scaled with factor 0.5 as numpy array."""*

*im\_color = cv2.imread(im\_path) # Read the image as a numpy array.*

*im\_color = im\_color[32:64 + 32, 32:64 + 32]*

*# Shape = (64,64,3) (x\_pixels,y\_pixels, color\_channels)*

*im\_color = cv2.resize(im\_color, (32,32)) # Rescale the image*

*# Shape = (32,32,3) (x\_pixels\*scale, y\_pixels\*scale, color\_channels)*

*im = np.zeros(shape=(32,32,1)) # Create an empty array with the final shape (32,32,1)*

*for i, x in enumerate(im\_color): # Fill the array*

*for n, y in enumerate(x): # Note: We cannot use cv2.cvtColor(im\_color, cv2.COLOR\_BGR2GRAY), because*

*im[i][n][0] = (y[0] + y[1] + y[2]) / 3 # that will return an array with the shape (32,32), but we need*

*return im*

* Tạo mảng thứ 2

*def get\_label(name):*

*"""Returns label number for file name*

*>>> get\_label("class\_10\_Index\_3454.jpeg")*

*10*

*"""*

*return int(name.split("\_")[1])*

* Để thực thi được 2 hàm này thì chúng ta phải cung cấp đối số cho 2 hàm này chính là đường dẫn thư mục ảnh mà chúng ta đã đối tên lại và cho chung vào một thư mục như đã yêu cầu ở trên. Và cách đọc là kỹ thuật của ngôn ngữ python.
* Như vậy là ta có thể kết thúc bước tiền xử lý dữ liệu và 2 mảng đã tạo đó sẽ trực tiếp tham gia vào CNN.

Tổng quan lại như sau

* Tải database về và giải nén.
* Chạy tệp “rename\_images.py” để tổng hợp tất cả các hình ảnh về một thư mục với cấu trúc tên quy định.
* Chạy tệp “data\_handler.py” để tạo 2 mảng tham gia vào phần sau.

1. **TRAINNING DATA**

* Xác định biểu đồ tensorflow:

1. Convolution layer: filter\_size 4x4, filters: 16
2. Pooling layer 2x2
3. Convolution layer: filter\_size 4x4, filters: 32
4. Pooling layer 2x2
5. Convolution layer: filter\_size 4x4, filters: 64
6. Pooling layer 2x2
7. Densely connected layer: neurons: 512
8. Dropout
9. Output layer: neurons: 47

* Chúng tôi sử dụng hàm trợ giúp để hoàn thành biểu đồ này. Và kĩ thuật được thể hiện trong tệp “training\_32x32.py”.
* Import

*import numpy as np*

*import tensorflow as tf*

*import math*

*import sys*

*import time*

*import datetime*

*import os*

Chúng ta cần “numpy” để làm việc với dữ liệu mảng, “tensorflow” sẽ là thành phần chính vì chúng ta đang sử dụng nó để tạo ra AI, “math” để tính toán các công thức gọn gàng, “sys” để đọc các đối số-sử dụng nó để chỉ định số lượng chu kỳ đào tạo thông qua dòng lệnh, “datetime” được sử dụng để ghi nhật kí.

* Tạo một lớp helper để làm việc với tập train và test:

*class NISTHelper():*

*def \_\_init\_\_(self, train\_img, train\_label, test\_img, test\_label):*

*self.i = 0*

*self.test\_i = 0*

*self.training\_images = train\_img*

*self.training\_labels = train\_label*

*self.test\_images = test\_img*

*self.test\_labels = test\_label*

i và test\_i dùng để theo dõi chỉ số hiện tại trong mảng train và mảng test.

training\_images và test\_images là mảng với cấu trúc [?, 32, 32, 1], ? là số lượng hình ảnh, 32, 32 tương ứng với độ phân giải của hình ảnh và 1 tương đương với số kênh màu thì ở đây được xử lý ở mức xám nên số kênh là 1.

trainning\_labels và test\_labels là mảng với cấu trúc [?, 47], ngày từ đầu chúng ta đã cho 47 nhãn dán khác nhau.

*def next\_batch(self, batch\_size):*

*x = self.training\_images[self.i:self.i + batch\_size]*

*y = self.training\_labels[self.i:self.i + batch\_size]*

*self.i = (self.i + batch\_size) % len(self.training\_images)*

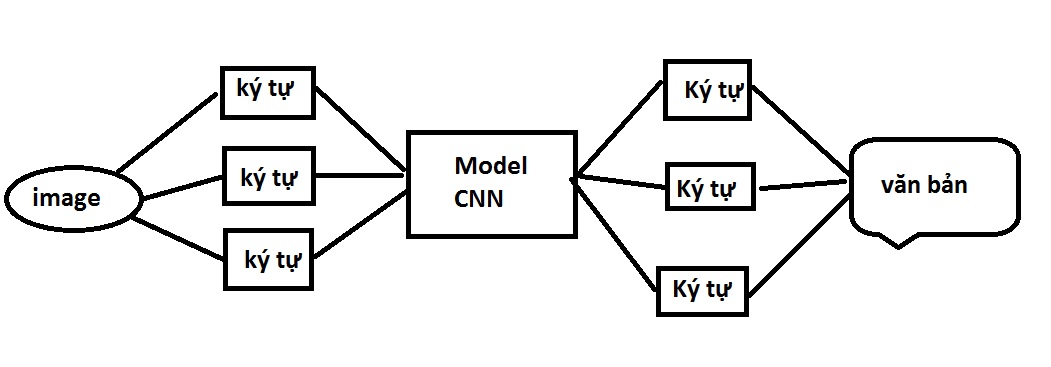
*return x, y*

hàm này trả về hình ảnh batch\_size tiếp theo.

* Hiện tại mức độ làm việc và cáo chỉ mới ở mức tạo lớp. Các hàm kỹ thuật để tạo ra biểu đồ tensorflow sẽ được tiếp tục cho đến hết đồ án.

1. **PREDICTION**

* Truyền vào một ảnh có kích thước lớn, sau đó cắt nhỏ các chữ cái có trong hình ra.
* Đưa vào rename\_images.py các hình ảnh ký tự đã được cắt để có cấu trúc như tập train.
* Sau đó ta sẽ có được mảng các ký tự của hình ảnh lớn.
* Đưa vào mô hình đã xây dựng cho ra kết quả và tập hợp tất cả các ký tự đó lại thành một văn bản hoàn chỉnh từ bức ảnh đầu vào.



*Hình 3: quy trình nhận dạng văn bản*

1. **PROCESS**

Get Data

Prediction

Test

Training

Preprocessing data

100%

0%

* Bước lấy dữ liệu và tiền xử lý đã hoàn thành
* Ở bước này khó khăn chính là cài python3 trên môi trường window
* Cài opencv cho python trên môi trường window. Thường xảy ra lỗi không tương thích version opencv và không tương thích bàn 32bit hay 64bit.
* Traning với ở bước đầu là tạo đối tượng thông tin phù hợp với mạng CNN. Còn đang trong giai đoạn nghiên cứu thuật toán.
* Khó khăn trong việc hiểu thuật toán vì này là kiến thức mới khó hiểu tiếng anh và khó hiểu công thức toán.
* Mới hiểu định nghĩa được thế nào là mạng neural network.
* Nhóm cần thời gian nghiên cứu nhiều hơn.

1. **REFERENCE**

[1]. [*https://www.nist.gov/srd/nist-special-database-19*](https://www.nist.gov/srd/nist-special-database-19)

[2]. [*http://cs231n.github.io/convolutional-networks/*](http://cs231n.github.io/convolutional-networks/)

[3].[*https://github.com/frereit/TensorflowHandwritingRecognition*](https://github.com/frereit/TensorflowHandwritingRecognition)