




MAY 13, 2020

DOODLE RECOGNITION SYSTEM

2020 SPRING CSE700

YICHEN WANG

#50161143



Demo

Video + Report: <https://jambako.com/doodle-recognize-system/>

Video Only: <https://jambako.com/wp-content/uploads/2020/05/1080p.mp4>

Code: <https://github.com/yichenwa/DoodleRecognitionSystem>

File Structure

DoodleRecognitionSystem

- |--album
 - |--apple/cat/sun/clock...
- |--dataset
 - |--.npy file from Google Cloud Platform
- |--venv
- |--create_dataset.py (Create dataset)
- |--image.png (The doodle you made last time)
- |--main.py (The main code, you can run this file to run the entire system)
- |--ml_recognize_doodle.py (The file construct CNN model)
- |--recognize_doodle.py (The file connect main.py and ml_recognize_doodle.py)
- |--yic_test_image.npy (The test and train dataset I made from create_dataset.py)
- |--yic_test_labels.npy
- |--yic_train_image.npy
- |--yic_train_labels.npy

Introduction

Nowadays, the memory of phones and computer is larger than 10 years ago, and people like to save a ton of photos in their device, which cause a problem that it will become a little hard to find a specific image. Sometimes we want to share a photo with friends but we need to scroll up and down to find that image.

I am thinking about implementing a system to help people get the photo they want quickly, so I design this doodle recognition system, which supports the user to draw a doodle, and the system will use convolutional neural network algorithm to decide what kinds of images the user is looking for, and return the image in that class.

Background

I used to plan using other image analysis algorithms as average hash or histogram to compare the doodle with all images in the album, then return the most several similar images. However, in the later test, I found the accuracy is too low, as most doodles are very abstract, and lost many important details from the real photo. As I am taking CSE555 this semester, so I think maybe a machine learning algorithm can work.

I select the convolutional neural network because I think it is very close to our human brain, and it has a good performance for large size dataset. So I think it will give better accuracy than other algorithms.

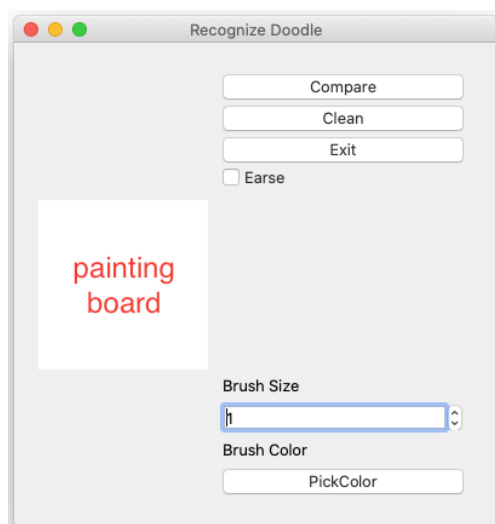
Preparation

From [Google Cloud Platform](#) we can download 50 million doodles, I selected 10 classes from them, which I thought are most common objects appears in people's album: [apple, bicycle, bird, book, car, cat, clock, face, key, sun]. As those images are saved in separate .npy files, so I need to combine them and prepare my database.

I picked 20000 images from each file and combine those 200,000 images as my train dataset, and picked 2000 images from each file as my test dataset. To be honest, I think these are not enough, as doodle is

much more complicated than handwritten digitals. Sadly, limited by my laptop's storage I cannot use a larger dataset. One important part is shuffling the dataset to make the images order randomly, as I picked images from the file by file, so the original dataset put images in the same class close to each other, which will decrease the accuracy.

Design



It has a 140*140 painting board on the left side, the reason use 140*140 is because when we want to predict the image, we need to resize it to 28*28, if the image is too large, it will miss many details after resize, if the painting board is too small, it will be very unfriendly. After some tests, I select 140*140 as my painting board size.

The tool buttons are on the right hand, you can clean the painting board, pick pen colors (mostly that will not affect the result because all the doodles will be converted from RGB to Gray before the prediction). After you finish your artwork, you can click "Compare", then it will return the images it thinks you are looking for.

Accuracy

```
Run: main x
198080/200000 [=====] - ETA: 0s - loss: 0.3160 - accuracy: 0.7249
198208/200000 [=====] - ETA: 0s - loss: 0.3160 - accuracy: 0.7248
198336/200000 [=====] - ETA: 0s - loss: 0.3160 - accuracy: 0.7248
198528/200000 [=====] - ETA: 0s - loss: 0.3159 - accuracy: 0.7247
198656/200000 [=====] - ETA: 0s - loss: 0.3159 - accuracy: 0.7246
198784/200000 [=====] - ETA: 0s - loss: 0.3160 - accuracy: 0.7246
198976/200000 [=====] - ETA: 0s - loss: 0.3159 - accuracy: 0.7245
199168/200000 [=====] - ETA: 0s - loss: 0.3159 - accuracy: 0.7244
199360/200000 [=====] - ETA: 0s - loss: 0.3159 - accuracy: 0.7244
199552/200000 [=====] - ETA: 0s - loss: 0.3159 - accuracy: 0.7243
199744/200000 [=====] - ETA: 0s - loss: 0.3159 - accuracy: 0.7242
199936/200000 [=====] - ETA: 0s - loss: 0.3159 - accuracy: 0.7242
200000/200000 [=====] - 74s 369us/step - loss: 0.3159 - accuracy: 0.7241
Test loss: 0.303
Test accuracy: 0.638
```

I construct a convolutional neural network with 2 hidden layers, and the accuracy for the test dataset is 63.8%. However, the real accuracy for user's doodle should be much lower, there are two reasons: The first one is the user is using a mouse to draw on the painting board, which is much more difficult than on the paper, and all the images in the dataset are people draw on paper, so there will be much more noise on the image made by a mouse.

The second reason is people will draw different artworks for the same item. Just like “There are a thousand Hamlets in a thousand people’s eyes”. This may be the main reason sometimes this system cannot recognize the doodle correctly.

There is a third reason but I fixed it in the developing, as the painting board is white and the default color is black, which is reverse with most database images, so before I use the model to predict the image’s class, I reverse its color.

Improvement

I plan to put this system on a website or a phone application because using a finger to draw on a screen is much more smooth than using a mouse on the laptop. Also, I need to improve its accuracy, one possible way is getting a device with larger storage, then it can go through the entire database from Google Cloud Platform, and it can classify more classes.