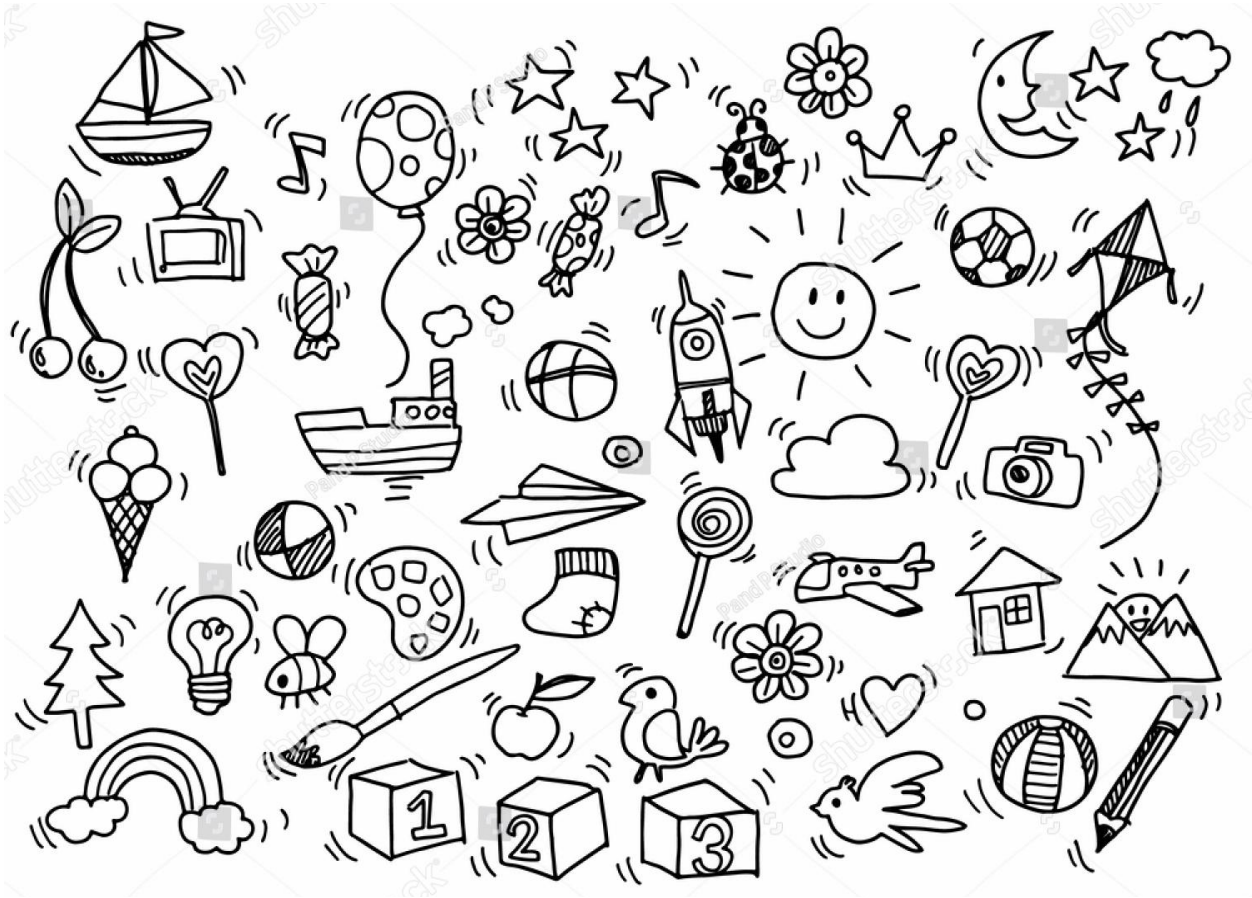


Doodle Classifier

15.12.2021

—

Samin Islam

CSE428: Image Processing

Computer Science and Engineering Department,

Brac University.

code

1. Overview

This project is a doodle classifier that will try to predict doodles with the help of image processing techniques based on neural networks architecture.

2. Problem Statements

- To preprocess the meta data, collecting the relevant data for the classification task
- To build 3D RGB bit map of images from 2D strokes from the dataset
- Limited RAM
- Unrecognized images
- Possible best accuracy for different doodles
- To detect doodle as per the class label

3. Motivation

A tutorial by Coding train on youtube, "Doodle classifier" has been my inspiration to choose this fun topic for the course project. Doodling is a by product of a subconscious state of our mind while doing or thinking about something else. Though the dataset that I have chosen is collected from a pool of conscious but quick drawing, credit goes to the giant tech company Google.

4. Dataset

4.1. Background

Quickdraw is an online game site where the user has to draw as asked on the site within 20 seconds. The intriguing part of this site is Google has injected a built-in Neural network model that decides whether the drawing is recognisable enough as asked by the website, and if not the player loses. This is a part of Google's research works, and all these doodles drawn by people across the world are stored as a dataset, that is the quickdraw dataset that I have used in this project. This dataset contains 345 categories of doodles each having hundreds and thousands of image information. There is a kaggle challenge for doodle recognition and a simplified dataset can be found there having a bit less data than the raw one.

There has not been much research with this rich and diverse dataset, but a team of researchers from Robotics Lab of Universidad Carlos III de Madrid did a statistical analysis extracting three parameters from this dataset with the help of RNN model. They took all the samples of only three

categories depending on the complexity of sketching those; mountain, book and whale. The authors of this paper considered mountain the low complexity sketch, whale the high complexity while book is moderately complex to sketch. These are the three parameters, among which the first one classification score demands an LSTM based Neural Network architecture. These two tables are the statistical analysis of their work.

Table 1: Arithmetic mean and the standard deviation of the distribution of sketches as a function of the 3 parameters

	Classification score		Number of strokes		Sketch length	
Category	μ	σ	μ	σ	μ	σ
mountain	5.16	2.13	2.09	2.15	18.04	15.08
book	2.96	2.67	7.21	3.78	42.50	17.17
whale	4.44	2.49	5.55	3.45	45.38	17.68

Table 2 : Z-score, giving an impression of the number of outliers in the dataset.

4.2. Description

As the dataset is huge, more than 50 million of samples related to doodle images which can be an issue with storage. One interesting challenge about this dataset is it does not contain any bitmap matrices of pixels to compose images. It contains the strokes of each image with data points. Due to storage limitations 15 categories has been takes as labels, under which 150 samples of information has been chosen from each randomly. In the preprocessing stage at first the given information from the dataset that is the arrays of strokes' points needed to be converted to bitmap matrices of pixels, only the recognised labeled data has been considered in the process.

Kaggle Link - https://www.kaggle.com/competitions/quickdraw-doodle-recognition/data

The raw dataset inside the json file followed the following format:

Tabele 3: Raw dataset table contents

Key	***Type***	***Description***
key_id	64-bit unsigned integer	A unique identifier across all drawings
word	string	Category the player was prompted to draw
recognized	boolean	Whether the word was recognized by the game
timestamp	datetime	When the drawing was created
countrycode	string	A two letter country code(locatoin of the player)
drawing	string	A JSON array representing the vector drawing

5. Preprocessing Data

The strokes are 2D data and VGG16 would require RGB structure that is a 3D array. For that a a ***draw_it()*** function has been built that will return 3D numpy arrays. The arrays will be normalized composing of pixels, stored in ***X-array***.

```
Function draw_it(strokes)->
    take an image
    for each stroke in strokes->
        for i from 0 till (len(stroke[0])-1)->
                Draw lines as per the index
    resize image
    normalization
    3D numpy arrays after normalization
    return normalized 3D numpy array
```

For the prediction, the labels are required that are the names of each category. This task has been done by extracting each file name and stored it in a ***Y-array***. Each categorical data from the ***Y-array*** has been converted to a vector using ***one hot encoder***.

6. Architecture

The proposed architecture is an ensemble one, where the base model of the **VGG16** has been implemented along with 3 extended fully connected layers. **VGG16** is a CNN based neural network architecture, the winner model of **ILSVR(Imagenet)** competition in 2014. The architecture has 16 deep layers, and later on another clone with deeper structure has been presented(**VGG19**). The **ImageNet** database contains a pretrained version of the network that has been trained on more than a million images. The pretrained network can categorize photos into 1000 different object categories, including several animals, a keyboard, a mouse, and a pencil..

The base model was not trained on the doodle dataset thus no changes of its weight happened. The prediction layer has been discarded of the base model which had the labels for the imageNet dataset. The altered model represented in this project is extended with a flatten layer, 3 dense layers with the **reLu** activation function. For avoiding outfitting after each dense layer a dropout layer has been adjoined. A softmax function at the output layer has been joined with 15 labels for prediction. The evaluation has been done on categorical value. The base model has not been tweaked to any of the weights but the extended 3 fully connected layers were trained on the dataset and changed its weights to minimize the loss.

7. Methodology

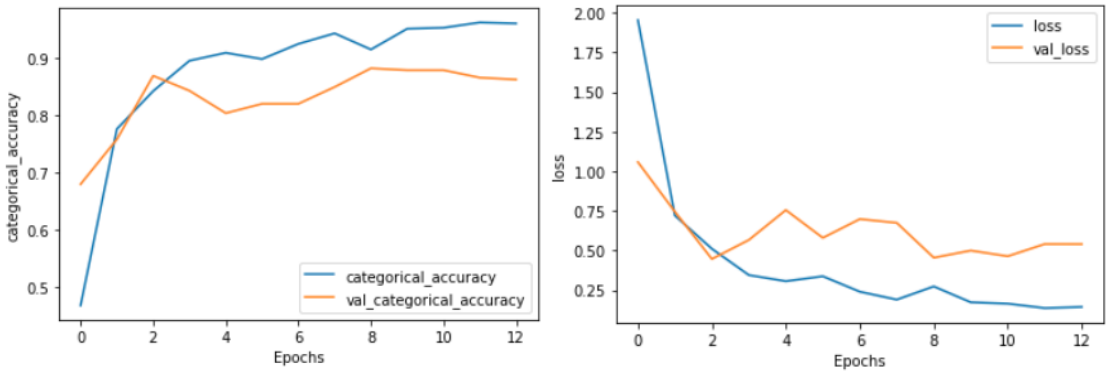
Keras, one of the popular Deep Learning and Nueral Network framework is used as the fundamental building block of the architecture. Important libraries(numpy, pandas, matplotlib, PIL etc) has been imported for preprocessing tasks as well. For maxplooing, Global Average Pooling is also used, along with Adam optimizer for optimizing the task. For avoiding outfitting after each dense layer there is a dropout layer with 0.2. Categorical value as the evaluation is done. The extended three fully connected layers were trained on the dataset and had their weights modified to reduce loss, unlike the base model, which did not change any of the weights.The dataset was divided into 85% and 15% for training and testing, or validating, the classifier, while fitting the model, using a batch size of 10.Number of classes is 15 and from each class 150 samples have been taken for implementing the task. Dimension of the images is 224 and images will be on RGB space. Initially while importing the base model the output layer is set to false as the model's output layers labels are irrelevant to this project's dataset. Moreover, additional layers have been added and an output layer for this project's task requires only 15 classes from the quickdraw dataset. The summary of the proposed model given below.

The summary of the model

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 vgg16 (Functional)           (None, 7, 7, 512)         14714688
 flatten_1 (Flatten)          (None, 25088)             0
 dense_4 (Dense)              (None, 1024)              25691136
 dropout_3 (Dropout)          (None, 1024)              0
 dense_5 (Dense)              (None, 512)               524800
 dropout_4 (Dropout)          (None, 512)               0
 dense_6 (Dense)              (None, 256)               131328
 dropout_5 (Dropout)          (None, 256)               0
 dense_7 (Dense)              (None, 15)                3855
=================================================================
Total params: 41,065,807
Trainable params: 26,351,119
Non-trainable params: 14,714,688
```

In the model in each layer-I is used relu activation, and the number of neurons is the multiples of 2, and added a dropout layer where 20% of the neurons will be deactivated before providing inputs for the next layer. It has been done to avoid overfitting. And here sequentially added all the layers starting from the

base vgg16 to my customized output layer with 15 labels and softmax function. 15 epochs has been run and results did not improve after the 3rd epoch.



Tthe result of the lassification task is depicted above, which seems moderate, neither too good nor too bad. We can see the accuracy curves of the model and the validation set, and then the loss curve. The curve for the validation set that is for unknown data seems a moderate result for both the accuracy and the loss. I think less categories with more samples from each one might provide a better result.

8. Limitations

The biggest limitation for this task was the limited RAM of colaboratory which led to other issues. Sessions have been crashed multiple times while adding more data. Some images are absolutely unrecognisable due to careless sketching of the players, though I discarded those while storing the input but those images also wasted some storage. More data needed to train the classifier as VGG16 is a massive pre-trained model trained on imageNet dataset, and small amounts of data will not provide good results.

9. Future Work

For the future work a small number of classes with more samples from each can be considered for this task. If more storage can be managed, training the whole dataset might give better results with diversity as the dataset consists of more than 50 million. Another interesting work can be done, generating new doodles by implementing GAN can be a thrilling task. By this we can create our own dataset of doodles which can be a great contribution as today's world requires more data in research.

10. Reference

1. Fernandez-Fernandez, R., Victores, J. G., Estevez, D., & Balaguer, C. (2019). Quick, Stat!: A Statistical Analysis of the Quick, Draw! Dataset. *arXiv preprint arXiv:1907.06417*.
2. Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.