Hand-drawn Sketch recognition

Ashwathi. R, Arjun Porwal, Arvind
Indian Institute of Technology, Jodhpur

Abstract

Feature selection has become of interest to many research areas which deal with machine learning and data mining, because it provides the classifiers to be fast, cost-effective, and more accurate. In this paper the effect of feature selection on the accuracy of Neural network multilayer perceptron k-nearest neighbors and random forest classifiers is presented.

Introduction

The goal of Sketch-based image retrieval is to allow non-artist users to draw visual content (usually objects) and then find matching examples in an image collection. Sketch-based image retrieval is an alternative or a complement to widely used language-based image querying (e.g. Google image search). In the computer graphics community, sketch-based image retrieval has been used to drive image synthesis approaches.

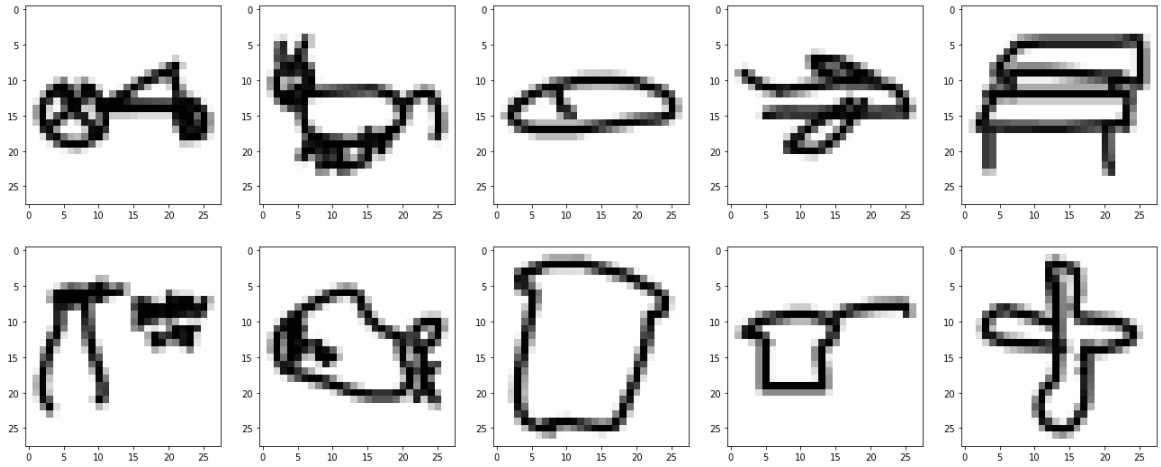
Sketch recognition mainly focuses on matching the sketches of the objects with the image database present. Based on the result obtained we classify them and retrieve the appropriate images. In a system that performs image searches, it is desirable for both to find close image matches and to classify an input image. But extraction of distinct features from input query is quite challenging in object sketch recognition systems. More effort is needed to determine optimal features from a certain set, based on attributes of the object that needs to be recognized and classifiers to be used. However, due to the ambiguity and limitations the procedure of object classifications has been recently over- powered by multiple approaches on neural networks such as deep learning.

Data

Our total data size is about 1GB with 60,000 drawings in 10 label classes with 6000 images of each class. Each drawing comes with specific variables:
cat, bicycle, bear, airplane, ant, banana, bench, book, bottlecap and bread.

Approach

We began by understanding the structure of arrays that make up a sketch and preprocessing the data. We took an input of 60000 images and used cross validation to divide it into train images of size 48000 images and test images of size 12000 such that 4:1.We then visualize the images by making a plot of the first 10 images . The data looks like as follows



From there, we tackled CNN architectures and used the rest of the models for their predictions and accuracy.

Data Preprocessing

1) The images were given as an array of 784 integers ranging from 0 to 255 as grayscale value.
2) We normalize the data by dividing it by 255 and bringing it by range in 0 to 1 .
3) For the CNN classifier, we change the image size to a 2-D matrix of size 28 x 28.

Most people draw doodles in similar ways. For example, if I asked you to draw a sun, you would begin with a single circle, and then dash lines radiating from the center in clockwise order. To capture this information, we used greyscale/color-coded processing to take advantage of the RGB channel while building the CNN so the model could recognize differences between each stroke. We assigned a color to each chronological stroke of a doodle, thus allowing the model to gain information on individual strokes instead of only the whole image.

Building Models

After completing all the data collection and preprocessing steps, it was time to get started on the most interesting part of the project — model building!

Before we dove into CNN, we tried some basic classifiers to compare different machine learning algorithms and familiarize ourselves with the data. We pulled numpy files of the data from Google Cloud Storage. This data has already been preprocessed and rendered into a 28x28 grayscale bitmap in numpy .npy format. Since the entire dataset included over 345 categories, we eventually went with a small subset that only contains the following 10 classes: airplane, ant, banana, bench, book and bread.

Random Forest (RF)

We first began with a Random Forest classifier. We utilized GridSearchCV to cross-validate the model and optimize parameters. We found that accuracy tends to plateau after 150 trees, so we used n_estimators = 150 as our final model, returning an accuracy of 0.769.

K-Nearest Neighbours (KNN)

Secondly, we tried the k-Nearest Neighbor (kNN) classifier, arguably the simplest and easiest model to understand. Its algorithm classifies unknown data points by finding the most common class among the k-closest examples. We cross validated the n_neighbors and found that the best model given is k = 5, which returns 0.707 accuracy.

Multi-Layer Perceptron (MLP)

Lastly, we tried the Multi-Layer Perceptron (MLP) from scikit-learn. We cross validated over different hidden layer sizes and learning rates, deciding on a hidden layer size of (784,784) and learning rate alpha = 0.001, which gave accuracy of 0.759.

Convolutional Neural Network (CNN)

Convolutional neural networks (CNN) have emerged as a powerful framework for feature representation and recognition. Convolutional neural networks are a type of neurobiologically inspired feed-forward artificial neural network which consist of multiple layers of neurons, and the neurons in each layer are then collected into sets. At the input layer, where the data gets introduced to the CNN, these neuron sets map to small regions of the input image. Deep neural networks (DNN), especially CNNs, can automatically learn features instead of manually extracting features and its multi-layer learning can get more effective expression.

Firstly, a larger batch size would help in addressing the noise due to mislabeled training data. The size parameter denotes the image size/resolution and has a significant impact on accuracy. For example, a comparison of 32x32 and 128x128 shows us that a size of 32x32 is too pixelated to achieve an accurate model.

The first model used two convolutional layers and one dense layer, each with a depth of 64 and 128 respectively. Such an increase in image size, however, required either a larger receptive field conv layer or an additional conv layer. Accordingly, we included one more layer when training with larger image size. The accuracy comes out to maximum at 0.8717.

Selecting an Optimizer

After trying a number of them for increasing the model complexity and maintaining the accuracy, we decided to compare the performance of the Adam optimizer.

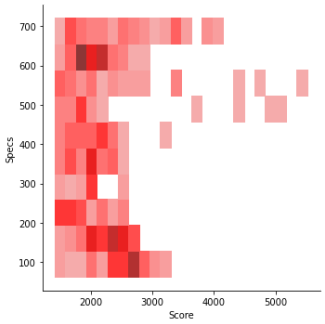
Adam is an adaptive learning rate optimization algorithm which incorporates both moving averages of moments to achieve per-parameter scaling of updates. In particular, it increases the step size of variables with slow moving updates and decreases the step size of fast moving updates. It is similar to the RMSProp optimization, but also incorporates momentum updates. Given learning rate α, decay parameters β_1, β_2, and numerical stability constant ε, the update step is

$$m := \beta_1 m + (1 - \beta_1) \nabla x$$
$$v := \beta_2 v + (1 - \beta_2)(\nabla x)^2$$
$$x := x - \alpha m / (\sqrt{v} + \varepsilon)$$

Feature Selection

(For decreasing memory node and increasing accuracy)

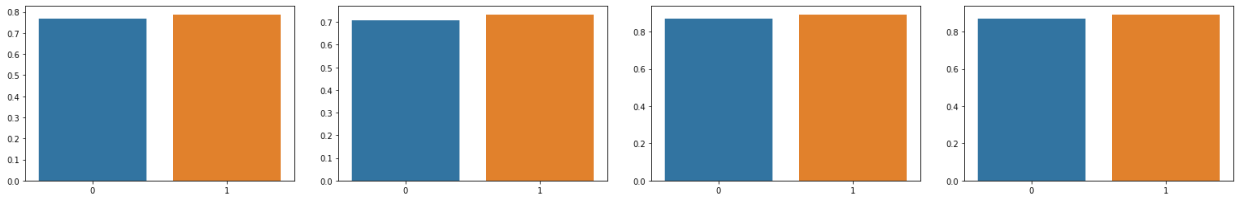
Feature Selection is the process where you automatically or manually select those features which contribute most to your prediction variable or output in which you are interested in. Having irrelevant features in your data can decrease the accuracy of the models and make your model learn based on irrelevant features.



S..No	Feature Index	Feature Score
1	529	5518.50
2	501	5166.24
3	506	4995.40
4...	557	4758.47
400	356	1422.60

To keep it as a 2-D image matrix the only option for the number of features are perfect squares. So, we tried with values 225,324,400 and 625 and found best results with n=400 compared to complexity. The final accuracy after feature selection compare to whole data are as follows -

	RF		**KNN**		**MLP**		**CNN**	
	Running time	Test Accuracy	Running time	Test Accuracy	Running time	Test Accuracy	Running time	Test Accuracy
Complete Dataset	3min 40s	0.7693	25min	0.7076	23min	0.7592	5min	0.8717
n=400	1min 30s	0.7881	16min	0.7337	17min	0.7880	3min 45s	0.8924



Conclusion

The project, Sketch Recognition for Image Classification and Retrieval, is developed with the objective of recognizing hand drawn sketches and retrieving related images of that hand drawn object or sketch. At the back end of the project various python codes are run in order to extract the low-level features of the query image as well as the training images which are maintained in the system. The two main algorithms employed for classification of images are MLP and CNN. The performance evaluation conducted after the completion of the project, for the respective algorithms, show that the CNN algorithm works best for classification. It has an accuracy rate of 90% for most of the images among all the dataset images. This helps in concluding that the project employs two methods or algorithms to serve the purpose of image classification, although it works best when the CNN algorithm is chosen.

With the feature selection method we were successfully able to decrease complexity, reduce time consumption and increase the accuracy in all the classifiers. The change was observed clearly in Multilayer Perceptron.

Acknowledgement

We would like to express our special thanks to our instructor Dr. Richa Singh for her efforts to make us this really wonderful project which helps in bordering our views and exploring the things.We would also like to thank all the TA's for helping in doubt related to the project.

References

- Aishwarya M S , Arvindh K , Dr. Vimuktha Evangeleen Salis, 2019, Sketch Recognition for Image Classification and Retrieval, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 08, Issue 07 (July 2019),
- Esra Mahsereci Karabuluta , Selma Ayşe Ozelb , Turgay Ibrikcib, A comparative study on the effect of feature selection on classification accuracy,Electrical-Electronics Engineering Department, Balcalı/Adana, 01330, Turkey.
- Kirti Pande, Doodling with Deep Learning!, Our Journey with Sketch recognition, Dec 15, 2018, https://towardsdatascience.com/doodling-with-deep-learning-1b0e11b858aa.
- ANTOL, S., ZITNICK, C. L., AND PARIKH, D. 2014. Zero-Shot. Learning via Visual Abstraction. In ECCV.
- @article{ sketchy2016, author = {Patsorn Sangkloy and Nathan Burnell and Cusuh Ham and James Hays}, title = {The Sketchy Database: Learning to Retrieve Badly Drawn Bunnies}, journal = {ACM Transactions on Graphics (proceedings of SIGGRAPH)}, year = {2016},}

Members' Contribution

Arjun Porwal (B19EE014) - Model Analysis and Report Making
Arvind (B19CSE016) - Data Processing, validation and Model Making
Ashwathi (B19CSE017) - Background research, evaluation graphs and Performance analysis