Udacity Nanodegree Capstone Proposal
Pictionary

Saad Chaouki

March 2021

In the final project of the capstone, I want to create a model that can be used to play Pictionary. The desired outcome is an application where a user can draw. As the user is drawing, a model that is hosted on AWS will be making prediction to try and classify the drawing.

The proposed methodology is a combination of a Neural Network trained and deployed on AWS and a drawing application using Tkinter in Python. The model will be trained on a dataset of drawings and will be accessed through the use of an API, Lambda function, and Endpoint. The input of the model will be an array representing a 28x28 drawing of the user.

Locally, the user will have a drawing application using Tkinter. Whenever the user releases the mouse, the application takes a screenshot of the drawing and converts it to an array of size 784 representing the 28x28 picture. This is then sent to the model to make the prediction. The predictions are then displayed on a different window.

1 Domain Background

The model used will be predicting on images of drawings. Therefore, this is classified as an image recognition project. This is a field where Machine Learning, and more specifically Neural Networks, is very popular. As the model will be predicting multiple types of drawings, this is a multiclass classification problem.

Therefore, the model I will be using is a Neural Network with the input size being 784 and the ouput being the number of classes that will be predicted. The final transformation will be a softmax function.

The reason why I am selecting this project is because I want to be able to use Neural Networks for a more fun project. It will be also very interesting to see the model making prediction as the user is drawing.

Multiple authors have used the same dataset to train models to either predict the drawings of a user or to train models to actually draw. For exmaple, Ha & Eck used a Recurrent Neural Network RNN to create a model that actually draws (Ha & Eck 2017). What most of these authors have in common is that they usually select a Neural Network as the model of choice.

2 Problem Statement

The problem that I am investigating is whether we can use Machine Learning to classify drawings of users. In addition to that, I will test the ability of making real-time predictions as the user is using the application to draw. The drawing application will also be created as part of the project using Tkinter.

3 Datasets and Inputs

The dataset I will use for this project is the Quick Draw[1] dataset by Google. This contains a large of number of drawings with their classifications and is stored as *npz* files. I will select a total of 30 drawings a starting point and will use a total of 75,000 image of each class. One important thing about the images is that their size is 28x28. Therefore, I will ensure that the screenshots of the drawings of users are also transformed to the same format.

Figure 1: Quick Draw Images

The dataset will be directly downloaded from Google Cloud to AWS using the Google Cloud package in Python. Once this is done, the training and testing set will be created and hosted on an S3 Bucket.

4 Solution Statement

The solution to be able to make the predictions on the picture is to create a multiclass Neural Network. The input of the model will be an array representing the image and the output will be the image label (car, pickup truck, house ...). Tkinter will be used to allow the users to draw. The drawings will then be sent to the model to classify.

5 Benchmark Model

No benchmark models were found during my research. However, my goal is to achieve a minimum of 80% accuracy on the testing set.

6 Evaluation Metrics

The main evaluation metric I will be using is the accuracy of the model on all the classes. I will ensure that there is no imbalance in the dataset in terms of images of each class. Additional testing will also be completed by drawing using the application.

7 Project Design

The Neural Network will be created using PyTorch. The model will be trained and deployed on AWS Sagemaker. To access the predictions from the application, I will create a lambda function and API.

[1] https://github.com/googlecreativelab/quickdraw-dataset

From the application perspective, I will create a simple drawing application using Tkinter. This will allow the user to start drawing. Whenever the user releases the mouse, a screenshot will be taken and transformed to a 28x28 image. This will then be transformed to a 768 array before it is sent as text to the model to make the prediction. The received prediction will then be displayed on a separate window.

References

Ha, D. & Eck, D. (2017), 'A neural representation of sketch drawings', *arXiv preprint arXiv:1704.03477* .