Capstone Proposal for Machine Learning Advanced Nanodegree

Venkatesh Prasad V M

ASL Alphabets Recognition

11th July 2018

DOMAIN BACKGROUND

We have seen people with speaking disability use sign language like ASL (American Sign Language). The problem is, it is not understood by common people which isolates them from the society. Now a days computers are really good at understanding human communications, even better than other humans with Deep Learning. In this project, I created an application with python which creates a CNN (Convolutional Neural Network) to train a model which can predict the alphabet of the given ASL alphabet image as input to the model.

PROBLEM STATEMENT

To determine whether a hand gesture image is an American Sign Language alphabet and find the appropriate class label (or alphabet) from the image.

DATASETS AND INPUTS

The dataset is taken from Kaggle. The dataset contains two zips, asdfsadf.zip as training data and afdasdf.zip as test data. The training data set contains 87,000 images, from which 0.1 fraction is used for validation data. The shape of every image is (200, 200, 3). There are 29 classes, of which 26 are for the letters A-Z and 3 classes for SPACE, DELETE and NOTHING. These 3 classes are very helpful in real time applications, and classification. The test data set contains 29 images to encourage the use of real world test images. Distribution of data is same since every class has 3000 images.

```
Label: A   Shape: (200, 200, 3)
```

SOLUTION STATEMENT

The Convolutional Neural Network is used to solve this problem. The CNN will learn the important features of the image, in this case number of opened fingers, angle of a finger, finger grooves, etc., and classify the image to the corresponding class. The loss function used is categorical cross entropy function in keras, which is

$$-\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C}\mathbf{1}_{y_i \in C_c}\log p_{model}[y_i \in C_c]$$

whose number is N, and the categories c, whose number is C. The term 1_{y_i \in C_c} is the indicator function of the i^{th} observation belonging to the c^{th} category. The p_{model}[y_i \in C_c] is the probability predicted by the model for the i^{th} observation to belong to the c^{th} category (Referred from here). Adam optimizer is used to update weights and bias.

BENCHMARK MODEL

I found a model by a kaggler, that predicts the appropriate alphabet with an accuracy of 91% which runs on 10 epochs, uses both adam and rmsprop optimizer with learning rate 0.0001.

EVALUATION METRICS

The evaluation metric function is similar to the loss function (which is categorical loss function in this case), except that the results from evaluating a metric are not used when training the model. Hence the evaluation metrics is categorical accuracy of the label of a predicted image with the actual label of the image.

PROJECT DESIGN

The steps taken in this project are as follows,

- ➔ Download the datasets from Kaggle and extract them.
- ➔ All the 200 x 200 images are converted to 128 x 128 and 0.1 fraction is taken as validation set and the remaining is the training set.
- ➔ For the every image rotation range and shift range is given to create a better model.
- ➔ The CNN architecture here is,
    - ◆ Layer 1 : inputs = 64, kernel_size=(4,4), strides=1
    - ◆ inputs = 64, kernel_size=(4,4), strides=2

- Activation : relu
- Dropout: probability = 0.5
- Layer 2 : inputs = 128, kernel_size=(4,4), strides=1
- inputs = 128, kernel_size=(4,4), strides=2
- Activation : relu
- Dropout: probability = 0.5
- Layer 3 : inputs = 256, kernel_size=(4,4), strides=1
- inputs = 256, kernel_size=(4,4), strides=2
- Activation : relu
- Dropout: probability = 0.5
- Flatten
- Dense : inputs = 512
- Output layer with softmax activation function

➔ The model is then trained with the training set and validation set with appropriate number of batch number and epochs..

➔ A random image from the test set is taken and processed with the model for testing purposes.

➔ Google Colab is used to utilise the cloud GPU for better efficiency of the model.