KAGGLE COMPETITION SPRING 2020 BENGALI.AI HANDWRITTEN GRAPHEME CLASSIFICATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Bengali.AI Handwritten Grapheme Classification Competition on Kaggle is a task to classify the 5th most spoken language in the world, Bengali's Handwritten Letters. The letter consists 3 parts, grapheme roots, consonant diacritics, vowel diacritics, which combine the meaning and the pronunciation of the letter. To classify it, we build up Convolutional Neural Network and Multi-classifier implemented by Tensorflow Keras. To improve the accuracy, we add more layers and applied dataset cleaning. And we eventually get 85.07 percent accuracy, which is ranked at 1654 in 2059 teams.

1 PROBLEM UNDERSTANDING

The Bengali.AI Handwritten Grapheme Classification is hold by Kaggle Competition, `https://www.kaggle.com/c/bengaliai-cv19/overview`. The task is to accurately classify the test cases of the Bengali Handwritten Letter, classify which grapheme roots, consonant diacritics, vowel diacritics make the letter by learning the labeled handwritten letter cases. Bengali has 49 letters, where 11 vowels and 38 consonants in its alphabet, there are also 18 potential diacritics, or accents, which makes the classification harder.

The data offers us 200840 handwritten training cases and the composition of grapheme roots, consonant diacritics, vowel diacritics. The evaluation is based on the accuracy of output in the test letter case component prediction.

2 PROBLEM SOLVING

2.1 DATA LOADING

In the competition, the data is saved as .parquet type of file, which is not efficient to load. As a writer of a public notebook posted and uploaded on the Kaggle Community r[1] , we decided to use the feather filetype to load the huge data efficiently.

2.2 DATA CLEANING

By implementing exploratory data analysis to the letter image, we found not all the dataset are valid. Some image data of the handwritten letter only include half of the letter, what we called "cut-off" images, which might have the risk that the images didn't cover the whole letter and lead to a bad accuracy in testing. And some images have some random dots, which should be common in the handwritten letter, but we believe the dots would affect the accuracy. Thus, we removed all the "bad images" – the "cut-off" images and images with random interference. After that, we have only 148340 cases to train with.

[1]Bangali.AI super fast data loading with feather `https://www.kaggle.com/corochann/bangali-ai-super-fast-data-loading-with-feather?fbclid=IwAR3fqh26X2MgnsoZg4sMqPxT7xe28qD0lX_Q-hmk5RPVGonHocKARkkr_04`

2.3 MODEL SELECTING

Convolutional Neural Network is a Deep Learning class of neural network, which basically share weights across space to the hidden layers between input and output. CNN is a great tool to classify images due to its moving kernel filters to extract important content in an image and apply non-linear function, like ReLU, LeakyReLU to manipulate data and apply pooling to find the important information in the images.

Combine Convolutional Neural Network with Multi-classifier would help us to classify the three components in a handwritten letter's image.

2.4 MODEL BUILDING

The credit to the code goes to public Kaggle notebook r[2] .

2.4.1 ADD LAYERS

Implementing Tensorflow Keras, we added more than 10 layers in the model using Convolutional Neural Network applying filter, activation, normalization, dropout and maxpooling. Then dense it to the number of uniqueness of the three components to define outputs.

2.4.2 DEFINE FUNCTIONS

By defining the compile function, we set up the loss function, optimize method and evaluation way. With defining the callback functions, we defined the stop points and monitor. And we also manipulate the ImgaeDataGenerator function to change it from Single-classifier to Multi-classifiers. Moreover, for the input, we need to scale the image pixel data from 255 into values between 0-1, as well as applying resize and reshape function to transfer flatten arrays into Rank-4 matrix.

2.4.3 TRAIN MODELS

After applying datasets split, we split the datasets into random training data, testing data to train the model and try to reach the optimized weights.

2.4.4 PREDICT MODELS

After generate the optimized weights, we can use the model to predict the test case and successfully receive 85.07 percent accuracy.

A APPENDIX

Code is attached as another file.

[2]Bengali.ai complete beginner tutorial 95acc `https://www.kaggle.com/deshwalmahesh/bengali-ai-complete-beginner-tutorial-95-acc?fbclid=IwAR1W79oJYX0Z0bIZy0Emx76pXkryEIxVl0y7C3jOH97JmU_Jzkc7Bcb66OA`