Initial Blog Post

Team Name: 42
Team Member: Cangcheng Tang, Guanzhong Chen, Shiyu Liu
Github Repo: https://github.com/shiyuliu1/Kaggle-42
EDA & Baseline Kaggle Entry:
https://github.com/shiyuliu1/Kaggle-42/blob/master/src/EDA_Baseline_Kaggle_Entry.ipynb

Data

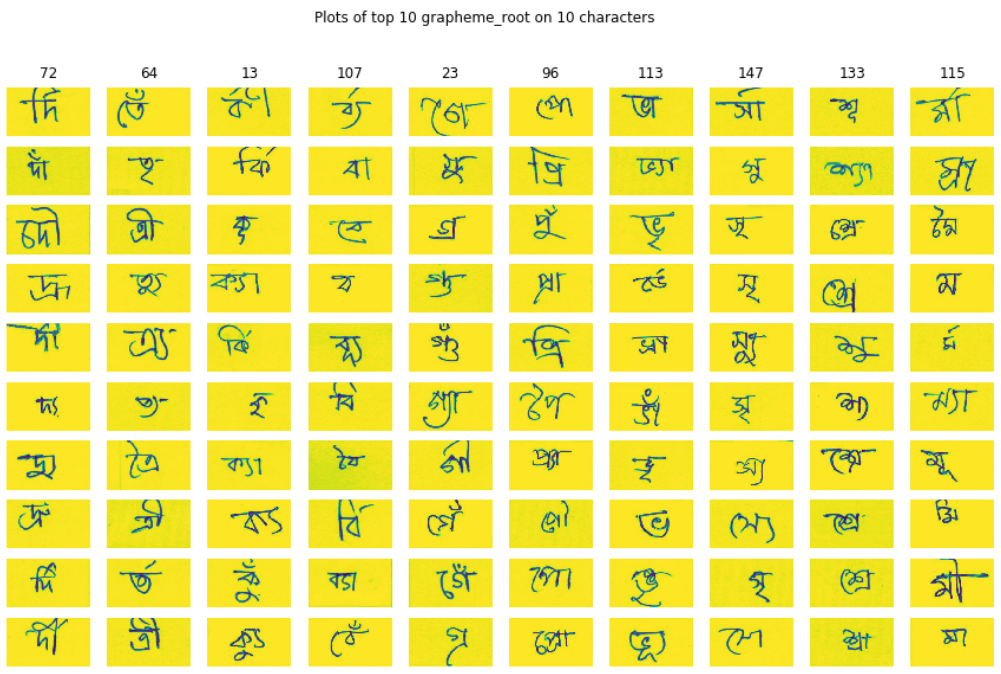
In this dataset, the feature variables are 137x236 grayscale images of Bengali characters. 3 targets for each character: grapheme root, vowel diacritics, and consonant diacritics. The training set consists of 20,0840 data points. The test set's size is the same, but can not be fully downloaded, only a few rows are provided to ensure consistency after submission.

EDA

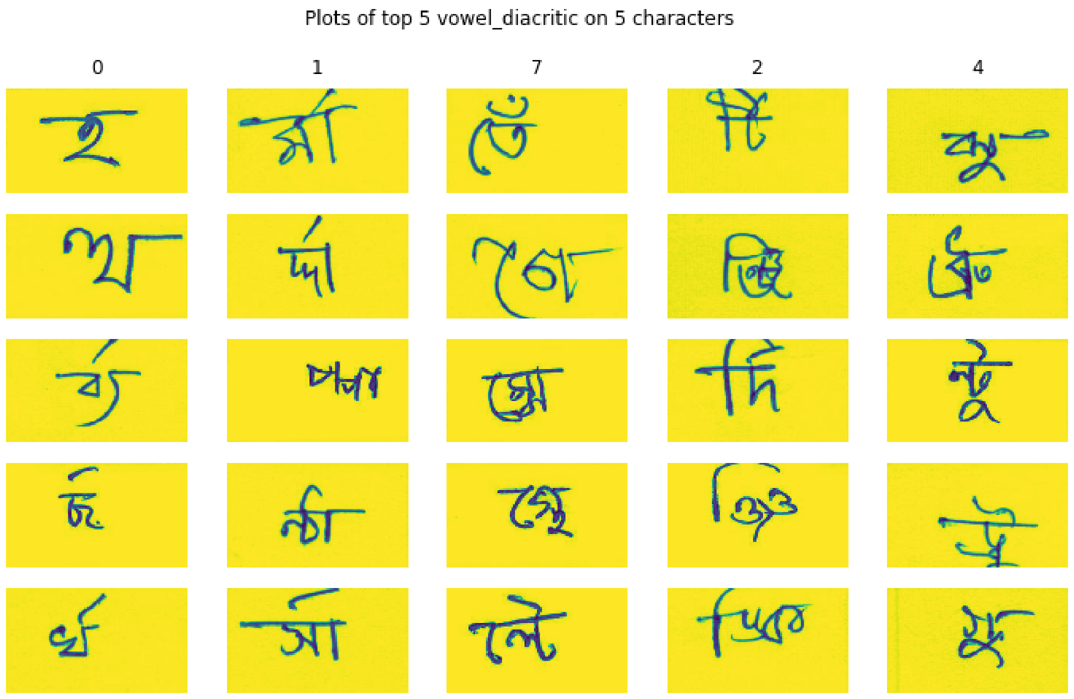
EDA has been focused on exploring the target variables, displaying handwritten Bengali images and developing the cropping strategy based on images' characteristics.

There are 3 targets for each character: grapheme root, vowel diacritics, and consonant diacritics. There are 168 types of consonant diacritic roots, 11 types of vowel diacritics, and 7 types of consonant diacritics.

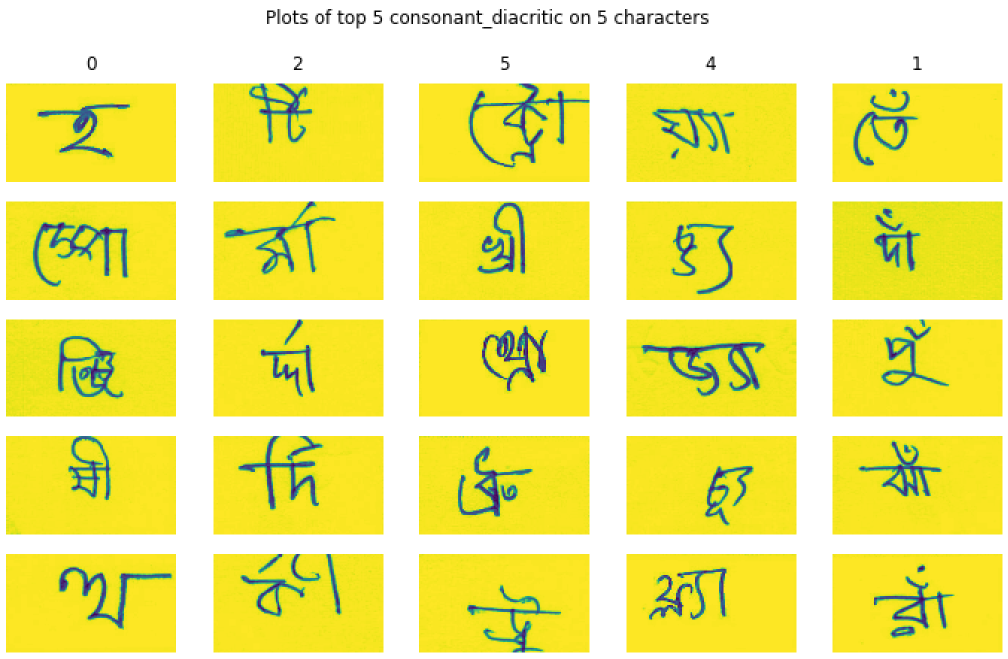
Among the consonant diacritic roots, we chose the top ten popular roots and plotted them out;



Among the vowel diacritics, we chose the top five popular vowel diacritics and plotted them out;



Among the consonant diacritics, we chose the top five consonant diacritics and plotted them out.



In addition, our customized functions enable us to find out the images with particular consonant diacritic roots, vowel diacritics, and consonant vowels and plot them out. Such exploratory works have given us a comprehensive view of how the data looks like as well as how we are supposed to prepare the data for fitting the models.

Feature Engineering

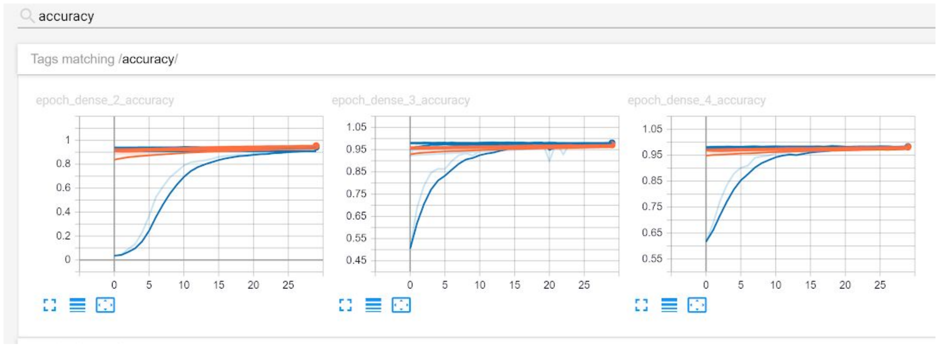
To reduce the number of parameters and increase training speed, the initial size of the images, 136*236, has been reduced to 64*64. Specifically, such transformation crop the blank parts of the image, then squeeze the height and width of each image to 64, making it easy to pass into Keras without padding a bunch of zeros at the edges. Images can lose much important information due to resizing. We used inter-area interpolation after cropping to resize images. (credit: this kernel)

Baseline Model Architecture and Training

Our baseline model training aims to ascertain that our data structure and initial model are compatible. convolutional neural network was chosen as the baseline model for this training, as mentioned in our first journal. By investigating public online notebooks, we first built a CNN consisting of 20 convolutional layers with batch normalization and dropouts after max-pooling layers. This neural network takes a 64x64 image as input after cropping and resizing. The output layer is designed to be three parallel layers with 168 nodes, 11 nodes, and 7 nodes, respectively.

Our first dataset was successfully fitted into the CNN model. The training results were relatively promising: within 30 epochs training on only one training dataset, the training accuracy and validating accuracy are all above .9. The accuracy of classifying both vowel diacritics and consonant diacritics has reached .97. We suppose there is a potential improvement in predicting the grapheme roots due to its relatively low accuracy of .93. We then trained all four data sets, resulting in an overall increase in all of the three accuracy scores.

Finally, we built a tensorboard for visualizing the training history, as shown in figure 1. The accuracy plot shows that both training and validation accuracy generally keep increasing across epochs, and the loss plot shows that the loss function generally decreases, indicating there is nearly no overfitting problem for the current model.



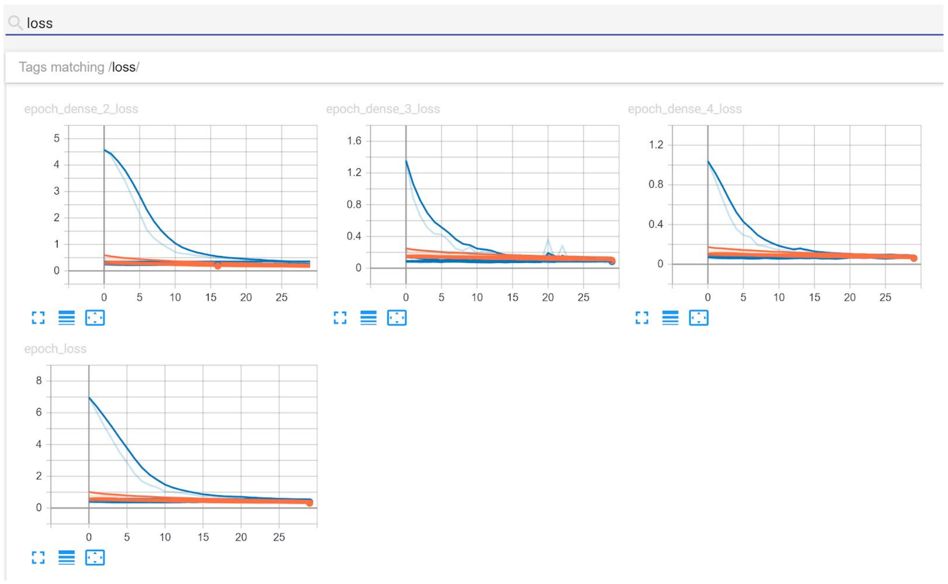


Figure1. The tensorboard for visualizing training history. It can be observed nearly no overfitting problem for the current model.

Next Steps

Our next step will first focus on improving the general performance of CNN. Specifically, adding early stopping, modifying learning rate and improving model architectures will be implemented to see if there will be an increase in the overall classification accuracy. Then our concentration will fall in modifying the CNN architectures by specifically improving the performance in classifying grapheme roots. This is because grapheme roots have much more classes than the other two target variables. We plan to build a deeper CNN for classifying this grapheme roots.

Work Cited

Bengali Graphemes: Starter EDA+ Multi Output CNN.
https://www.kaggle.com/kaushal2896/bengali-graphemes-starter-eda-multi-output-cnn/data#Exploratory-Data-Analysis