ECE 4200 Font Recognition Report

Peihan Gao (NetID: pg477)

Submission Date: Dec 13th, 2020

1. Models

<u>1.1 Best model: XGBoost</u>

Among all the attempted algorithms, XGBoost performs best at this font dataset with a private leaderboard score of 94.89**%** (the confusion matrix is available in the Jupyter notebook). Besides the hyperparameter optimization, the data augmentation (rotation, flip, contrast adjustment, and Gaussian filter) is proven as an effective method to improve the prediction performance. The tested combinations of data augmentation method and corresponding accuracies are shown below:

1. No data augmentation: 92.64% (baseline)
2. Rotation + flip: 94.36%
3. Gaussian: 94.20%
4. Contrast adjustment: 93.93%
5. Rotation + flip + contrast adjustment: 94.88% (best combination)
6. Rotation + flip + Gaussian: 94.74%
7. Contrast adjustment + Gaussian: 94.68%
8. Rotation + flip + Gaussian + contrast adjustment : 94.81%

As a result, all these tested methods of data augmentation can independently improve prediction performance. The best combination of data augmentation is “rotation + flip + contrast adjustment”.

With respect to the 7 dimensions of font information, corresponding experiments were done by removing one or several relevant dimensions. The results show that removing the dimension of Unicode value has little effect in the XGBoost model (accuracy decreased by only 0.43%) while the dimensions of original height and width strongly influence the prediction result (accuracy decreased by 12.24%).

<u>1.2 Convolutional Neural Network (CNN)</u>

Both the shallow CNN and a 50-layer Residual network (ResNet) are constructed and tested on the font dataset. The constructed CNN has two inputs, including a

20x20 input into the convolution layers and a 7-dimension input concatenated with the dense layers. The shallow CNN shows an accuracy of about 74% while the ResNet shows an accuracy of approximately 83% with data augmentation. The performance of CNN is unexpectedly worse than XGBoost on this font dataset. It is inferred that although the 7 character attributes were added to the dense layers, their actual effect is limited compared with XGBoost.

1.3 Traditional Machine Learning Method

The traditional ML methods, including logistic regression, naive Bayes, MLP, and PCA+KNN were tested. Their test accuracies are respectively 46.40%, 33.02%, 63.82%, and 64.74%, which are much lower than that of XGBoost or ResNet.

2. Challenges and Bottlenecks

1) The training set size explosively increased to more than a million after data augmentation (rotation, flip, and contrast adjustment). Thus, the program usually crashed on the local environment (RAM: 8GB). The solution is to train the model in the Colab using the high-memory mode (maximum RAM available: 25.51GB). Additionally, no hardware accelerator (GPU/TPU) was used.

2) A failed attempt of combining the XGBoost with the similarity matching was made for the purpose of making full use of the dimension of Unicode value. The program retrieves all the samples with the same value in the dimensions of “Unicode”, “bold”, and “italic”. The XGBoost predicts with the assistance of the retrieved data in the training set by calculating their cross-correlating. In theory, two binarized images of characters with the same values of “Unicode”, “bold”, and “italic” should be exactly the same because the training set is observed to be well cropped. However, the accuracy degraded from 94.88% to 82.92%. Conclusively, the attempt of combining XGBoost with cross-correlating to make full use of the Unicode value finally failed.

3. Discussion about what I learnt

1) CNN is not always the optimal solution for the image dataset.

2) The proper data augmentation may obviously improve the prediction performance but also occupy more RAM and cost more time.