CS 583 Course Project: Kaggle competition on Kannada MNIST Data

Yangyang Yu
Xuesong Liu
Ashish Negi
Stevens Insitutes of Technology
New Jersey, United States

December 2, 2019

1 Summary

We participate an activate competition of classifying hand written Kannada digits. The final model we choose is ensemble CNN with data augmentation, a deep convolutional neural network architecture, which takes the 28*28 pixels images as input and outputs the class labels. We implement the convolutional neural network using Keras and run the code on Google Colab with Intel(R) Xeon(R) CPU@2.20GHz and 12GB memory and NVIDIA Tesla T4. Performance is evaluated on the classification accuracy. In the public leaderboard, our score is 0.98700; we rank 202 among the 961 teams.

2 Description

1)Problem:

The problem is to classify hand written Kannada digits on images. This is a multi-class classification and image recognition problem. The competition is at https://www.kaggle.com/c/Kannad MNIST. The dataset is an extended version for the hand written Arabic digits image dataset named MNIST. The MNIST dataset was one of the benchmark classification problem, although it become a trivial problem now as more and more novel machine learning techniques emerge. As an extension for MNIST dataset, the goal of Kannada dataset is to correctly identify the digits written in one of the Indian languages.

2)Data:

The raw data are the csv files stores the digit image pixels. The files are containing 784 columns for each pixel value of one digit image. For a image, its width and height are 28. The number of training samples is n=60,000. The number of classes is 10.

While doing the task, we have 60 samples, we use 48000 for training, 12000 for validation. And the test set is 5000 samples in total.

3)Challenges:
First, without the data augmentation and ensemble algorithm, our training accuracy is already above 99%. It will be very hard if we would like to increase the accuracy and public score with the original dataset. This inspires us to try the data augmentation.
But it still be challenging to do the tiny improvement on accuracy when the accuracy is closer to 1.

Second, the ensemble method that we implemented in this task help us increase the public score and ranking. But we still suffering from the overfitting problem.
That is predictable as the model structure is more complex than the single CNN. And that can been seen the figure of training and validation accuracy and it also reflects in the result of identifying digits in test set.

Figure 1

3 Compared Methods

Advanced Methods:
1. Data Augmentation:
Based on the feature of this task, we implemented the data augmentation methods and further adjust the related parameter in order to make more data samples for us to train the model.
The parameters that we tuned are:
1) Degree range for random rotations (rotation range = 10)

2) Shifting the image in two directions: Width and Height shifting (both 0.1).

By doing this, we get more samples in a reasonable way. In order to avoid generating images which are hard to tell aligning with multiple similar images, we choose not use the flip, zooming and shearing option. Moreover, we found that by controlling the degree of shifting and rotation within a certain degree, we can get a better score.
For Example, using shifting as 0.2 instead of 0.1 and rotation as 20 instead of 10, we get a lower score.

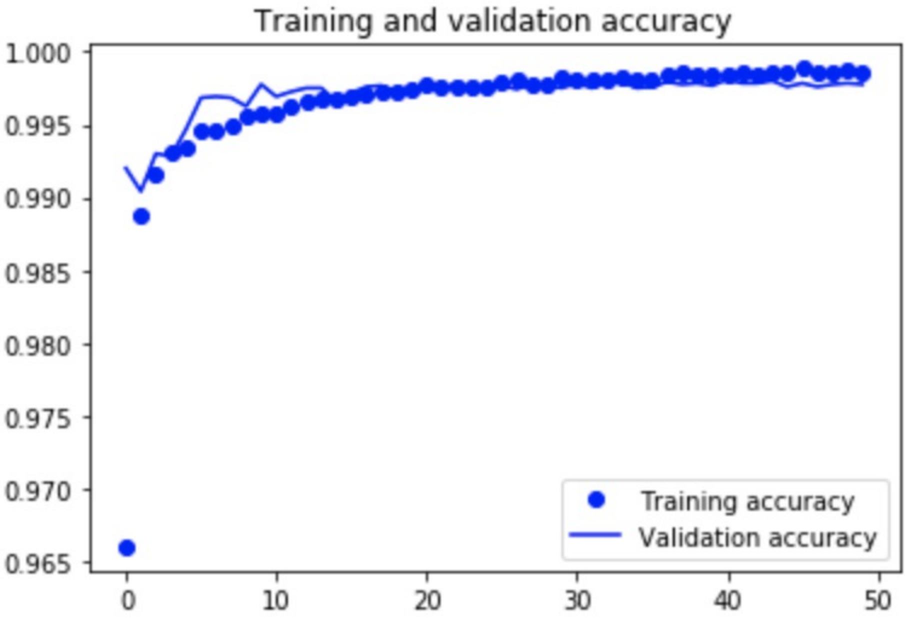
2. Ensemble Methods:
In out competition, we eventually ensemble 15 layers of CNNs working as a pipeline to achieve to drive the public score from 0.9742 to 0.9870.

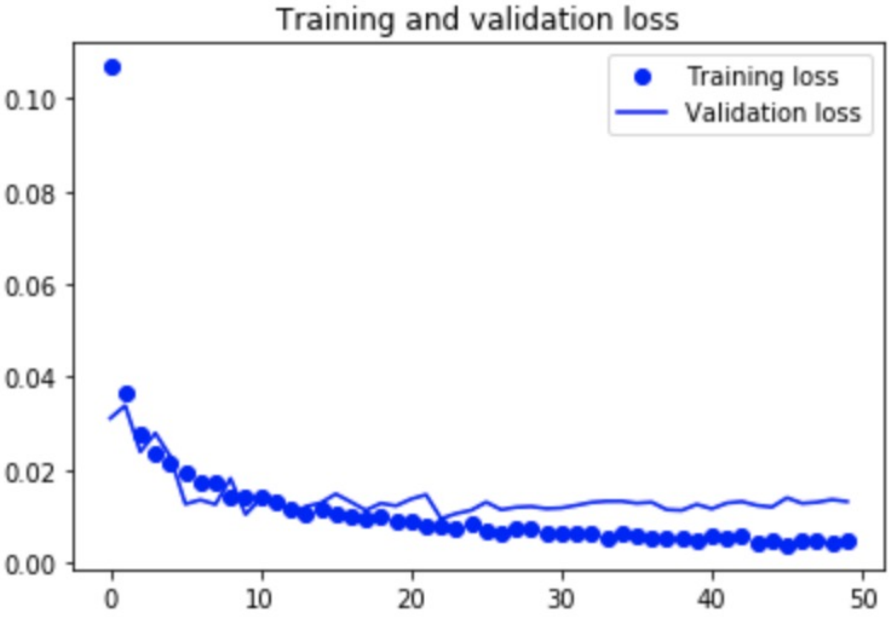
4 Outcomes

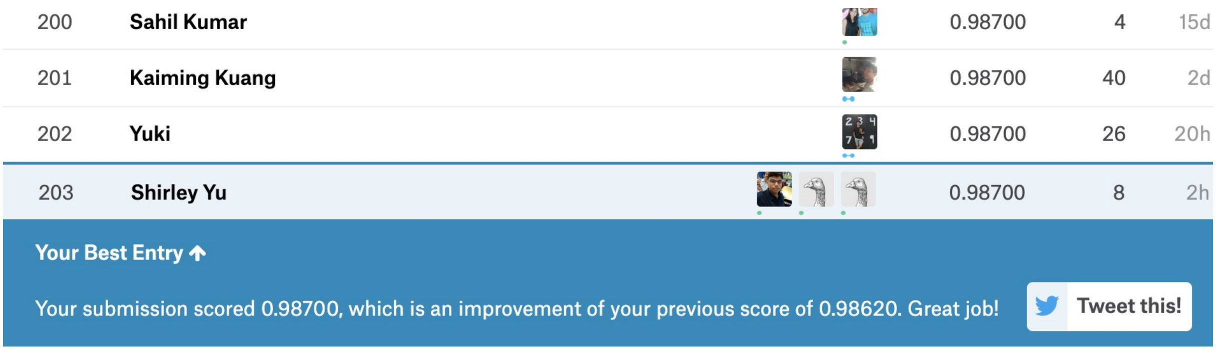
We participated in an active competition.
Using this method, our training accuracy is 99.77% and validation accuracy is 99.74%.

For test data, our score is 0.987 in the public leaderboard. We rank 203/961 in the public leaderboard, which is the approximately the top 21 percentile in the full ranking. Our ranking screenshot is shown in Figure below.

Figure 2. Team Ranking on Public Leaderboard


Training and validation accuracy
1.000
0.995
0.990
0.985
0.980
0.975
0.970
0.965
0
10
20
30
40
50
● Training accuracy
— Validation accuracy


Training and validation loss
0.10
0.08
0.06
0.04
0.02
0.00
0
10
20
30
40
50
● Training loss
— Validation loss



200	**Sahil Kumar**	0.98700	4	15d
201	**Kaiming Kuang**	0.98700	40	2d
202	**Yuki**	0.98700	26	20h
203	**Shirley Yu**	0.98700	8	2h

Your Best Entry ↑

Your submission scored 0.98700, which is an improvement of your previous score of 0.98620. Great job!