Exam Report

Yongchao Qiao

In this 7-day competition, I got a good evaluation score on the test set, but I think the best model I got is still not enough to meet the demands for industrial purpose since the test loss is not that low. However, now I will introduce everything I did to get the existing best model for me and other possible directions of improvement if given more time. Generally, there are three main directions I followed to get this best model, they are data augmentation, ensemble models, matched evaluation methods, weights initialization and learning rate scheduler. Then I will explain them in detail but follow an order in processing the competition.

I. Fundamental preparation

This part is about initial processing for the original data, data augmentation and predict file that will be submitted. Actually, it contains my work on day1. First, I downloaded the train.zip to the cloud with the given link and read images as well as category names. Then I saved images and targets into different tuples group by the file suffix, like png or txt. Next, I wrote a loop to check whether the images and targets are matched correctly. There are total 929 images, 844 of whose size are (1200, 1600, 3) with 85 belonging to (1383, 1944, 3). Since the cells in these images are not that big, I resized them into one same size as (120, 160, 3) to avoid losing much information, fortunately this size was verified as the most suitable one for my model and the memory of cloud CPU and GPU which helped me train the model precisely. Then I planed to split the original data into train and test set directly, but the strategy of the split function confused me, since without a reasonable strategy, the distribution of train set, test set and original whole set might be different, which would disturb the prediction on the true test set. Therefore, I classified the original dataset into different categories by the combination of cell names shown in the txt file (Here I did not take the order of the names in the combination into consideration). In this way, I found the original dataset could be classified into 40 categories, where some categories only contained one observation. So I decided to augment the original dataset to 9290 observations first. That is, each image would generate 9 other new images by six reasonable methods like rotation, horizontal flip, vertical flip, shear, feature-wise_center and feature-wise_std_normalization. After getting the augmented dataset, I got the x_total with size (images, 120, 160, 3) and replaced the name strings with corresponding one-hot-encoded category numbers with following the order: ["red blood cell", "difficult", "gametocyte", "trophozoite", "ring", "schizont", "leukocyte"] to get the target_total. For example, if an image contains red blood cells and trophozoite cells, the target will be [1, 0, 0, 1, 0, 0, 0]. Also, I gave each image a label (from 0 to 39) based on the frequency of its category in a descending order to get the target_category_coding which will be used as the strategy to split the augmented dataset. Then the augmented dataset could be split successfully into train set and test set, with strategy as the category label. In this way, both train set and test set contain all 40 categories. Next, I built the CNN network, trained the model and check my predict file. In the process of tuning the parameter, each time I adjusted the value of only one parameter to detect the best value of this parameter.

II. Evaluation methods

After reading the introduction of this competition, I found that the final evaluation method was the Binary Cross-Entropy loss. So I used the same evaluation method to test my model after each training. This really helped me narrow the distance between the score I got on my own validation set and that of the blackboard. However, there are still some modifications regarding output form for my evaluation methods

during this 7-day competition. On day1, I used probability output to calculate both train loss and test loss to get some good models and used the binary output to submit. On day2, I used probability output to calculate train loss and binary output to calculate test loss to get some good models and used binary output to submit. On day3-day7, I used probability output to calculate both train loss and test loss to get some good models and used the probability output to submit.

III. Feedback analysis and adjustments

This is the main part that I used to find my best model. During the competition, I focused on my evaluation score for each submission on the blackboard, since it could tell me the direction of improvement regarding my future models. On day1-day3, I used one same CNN network, which gave me a large distance between test loss on my own test set and the true test set on the blackboard, though I added the shuffle and early stopping operation for each epoch and tried many values of batch size, learning rate and dropout. So I decided to rebuild the CNN network and change the random seed. Of course, I tried many kinds of CNN networks under the memory limit of GPU, like less layers with more kernels and more layers with less kernels. Finally, I found a good one which would be used on day4-day7. For this CNN network, I increased the number kernels in each convolution layer which is shown as below.

```
class CNN(nn.Module):
    def __init__(self):
        super(CNN, self).__init__()
        self.conv1 = nn.Conv2d(3, 8, (5, 5))  # output (n_examples, 8, 116, 156)
        self.convnorm1 = nn.BatchNorm2d(8)
        self.pool1 = nn.MaxPool2d((2, 2))  # output (n_examples, 8, 58, 78)
        self.conv2 = nn.Conv2d(8, 16, (3, 3))  # output (n_examples, 16, 56, 76)
        self.convnorm2 = nn.BatchNorm2d(16)
        self.pool2 = nn.MaxPool2d((2, 2))  # output (n_examples, 16, 28, 38)
        self.conv3 = nn.Conv2d(16, 32, (3, 3))  # output (n_examples, 32, 26, 36)
        self.convnorm3 = nn.BatchNorm2d(32)
        self.pool3 = nn.MaxPool2d((2, 2))  # output (n_examples, 32, 13, 18)
        self.conv4 = nn.Conv2d(32, 48, (3, 3))  # output (n_examples, 48, 11, 16)
        self.relectpad = nn.ReflectionPad2d((0, 0, 1, 0))  # 13 18
        self.convnorm4 = nn.BatchNorm2d(48)
        self.pool4 = nn.MaxPool2d((2, 2))  # output (n_examples, 48, 6, 9)
        self.linear1 = nn.Linear(48 * 6 * 8, 128)  # input will be flattened to (n_example
        self.linear1_bn = nn.BatchNorm1d(128)
        self.drop1 = nn.Dropout(DROPOUT1)
        self.linear2 = nn.Linear(128, 64)
        self.linear2_bn = nn.BatchNorm1d(64)
        self.drop2 = nn.Dropout(DROPOUT2)
        self.linear3 = nn.Linear(64, 7)
        self.act = nn.ReLU(inplace=True)
```

Then I tried different methods to improve the score. For example, I adjusted the batch_size, learning rate, the weight initialization methods, parameters in different optimizers and learning rate scheduler. Finally, I found the combination of Adam, xavier_normal weight initialization method and CosineAnnealingLR performed well in my model which also helped me get my best single models on each day. However, I did not add extra weight to the loss while training, since I thought this given imbalanced data might be the true

distribution of the real data.

IV. Ensemble models

The ensemble model is an important tool for me to improve my evaluation score on the blackboard. Actually, I used this tool in the submission of day5-day7. In these days, I combined the best single models of previous days up to each present day. For example, on day7 I combined the best single models from day4, day5, day6 and day7 to find a best ensemble model. Finally, I got my best model in this competition with combining one best single model from day 4 and day6 respectively and three good single models from day7. Also the evaluation score for my best model in this competition is 0.204833.

V. Other ideas to make possible improvement

The first idea is that assuming there are only 40 categories in the total dataset, we can transform this Multi-Label classification problem into multiple classification with 40 categories as the target. I think this idea might give me some new taste with much more observations like 92900 images. The second ideas is that we can try more ensemble methods for ensemble models, since I only used averaged method to get my best ensemble model.