Hematologic Cell Identification Using Neural Network

Grylls LIANG

Section 1: Project Background

Hematology plays a vital role in diagnosing and monitoring various medical conditions, including anemia, leukemia, infections and clotting disorders. Hematologic cell identification is a critical task in the field of medical diagnostics and treatment. Traditionally, this task has been performed manually by medical professionals, which could be time-consuming and error-prone due to the intricated morphological features of different cell types and the inherent variability within microscopic images. Fortunately, with the development of machine learning, massive image identification tasks could be completed much easier, which have substantially reduced the workload of manual process with a remarkable precision.

The main task of this project is to develop a classification system capable of distinguishing among five classes of white blood cells: basophil, eosinophil, lymphocyte, monocyte and neutrophil. The project dataset come with labelled images by senior pathologists, detail description of datasets are as below:

Raabin-WBC Dataset (WBC):

This dataset comprises microscopic images of different types of white blood cells, including basophils, eosinophils, lymphocytes, monocytes, and neutrophils. Each image contains only one or two stained white blood cells and is associated with a specific cell type, forming the basis for the project. For the downstream classification tasks, this dataset comprises 301 basophil, 1066 eosinophil, 3461 lymphocyte, 795 monocyte, and 8891 neutrophil images.

Papillary Renal Cell Carcinoma Dataset (pRCC):

The pRCC dataset contains a diverse range of medical images, they are selected and cropped by pathologists from the TCGA-KIRP dataset. This dataset comprises 870 type 1 ROIs and 547 type 2 ROIs, with each image meeting the M scale dataset criteria. While not directly related to hematologic cells, the features learned from this dataset can potentially aid in recognizing patterns and improving generalization.

Camelyon16 Dataset (CAM16):

This WSI dataset is derived from the Cancer Metastases in Lymph Nodes challenge. In each WSI, we select 5 to 10 ROIs with dimensions of 8000*8000 (2560*2560 under the CPIA standard) to meet the average L scale dataset criteria. The dataset comprises 540 tumor and 541 normal images. Similarly, the Camelyon16 dataset, although not inherently tied to hematologic cell classification, contributes to the pre-training process by exposing the model to additional medical imaging data. The insights gained from this dataset can play a pivotal role in enhancing the model's ability to capture intricate details.

Section 2: Method Introduction and Analysis

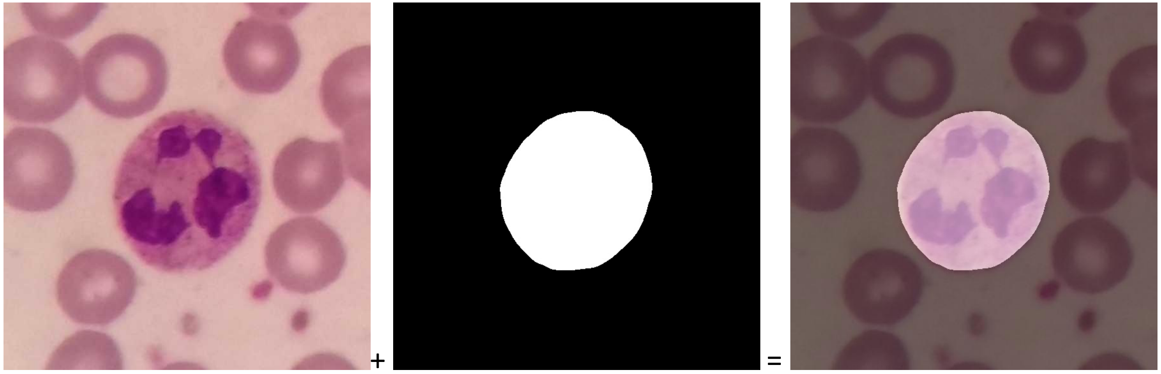
To achieve the project goal, I firstly developed a simple model based on convolutional Neural Network(CNN), using all Raabin-WBC(WBC) datasets as training model. Then, I made some attempts to improve my basic model:

1. Fine-tuning parameters, refining model structure and adding dropout;
2. Applying masks to corresponding images, treating them as new training dataset;
3. Replace those images who have mask with masked image, plus other raw training images as training datasets;
4. Pre train my model with CAM16 based on ResNet50 (w/ and w/o imagenet dataset);

Here are specific steps of the task process:
Step1: Data Processing

For all datasets, I loaded them as ImageDataGenerator, in this step, I reshaped all datasets to a consistent size of 224*224*3 to have a standard input in the model. To reduce overfitting and improve the model accuracy, I applied data augmentation, using rotation, horizontal and vertical flips, shearing.

I found the original dataset has provided us with mask, firstly I do not quite understand how to use it, I thought the mask should be applied to all images, and have no idea how to implement it. After I realized the mask and image are one-one matched, I applied the matched ones with their own mask, here is one example of masked image vs. raw image:



Raw_image + mask = Masked_image

Step2: Build the Basic Model without Additional Information and Fine tuning

For this part, I used CNN as my base model, after some times attempts, I finally get a model with several CNN layers, max pool layers which could be seen in my code. At the beginning, I found that the model seems overfitting, as the accuracy can reach around 0.8, while the validation accuracy is only 0.6128. I tried to increase the epoch to see whether it could reach a better validation accuracy, and found when the epoch is larger, the validation accuracy did have some change. My idea is it got stuck in the local minimum, or the learning rate is too small that results in slow convergence. I changed my optimizer to Adam, giving a higher learning rate. After the change, the validation accuracy changed faster, but still not very high as the accuracy. It is obvious that I encountered with overfitting, what I did is to add some dropout, and when the model went deeper, I gradually added the dropout rate. Finally the validation result (Shown in Table1, Appendix) can reach over 0.9 when using sufficient training datasests.

Step3: Using Additional Information

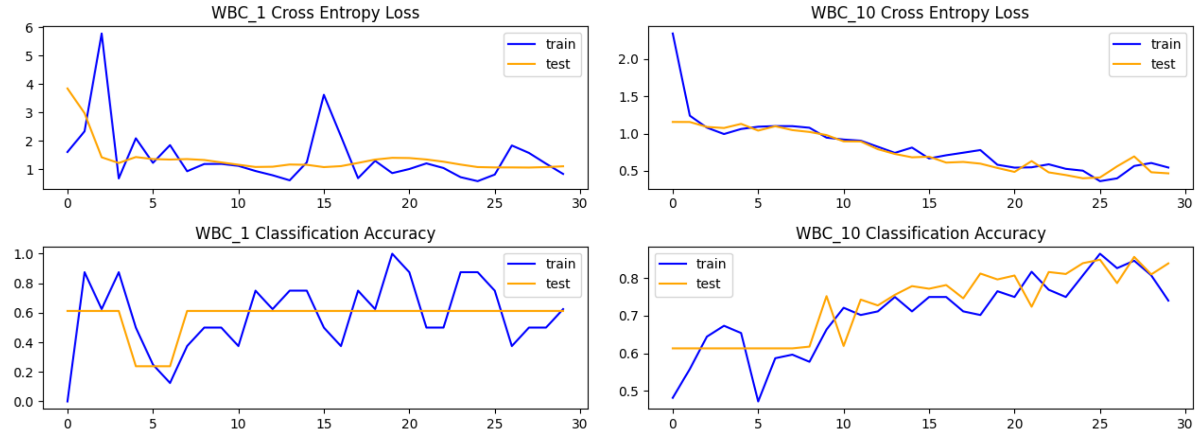
The first additional information I used is the image mask. After mask process, I used those masked images as training dataset, and run my basic model. I thought it may be better, but unfortunately, the accuracy dropped (Shown in Table 1, Appendix). Since mask did help us to extract the object and the feature of the target cell, this is due to the size of the dataset had dropped dramatically by 90%. To approve my idea, I merged masked images and raw images, training all the data, and get a similar result as my basic model. What should be noticed is that, looking at the trend of accuracy. Due to limited time, I did not try longer epoch and Hypothesis Testing, which may help to identify if this way works.

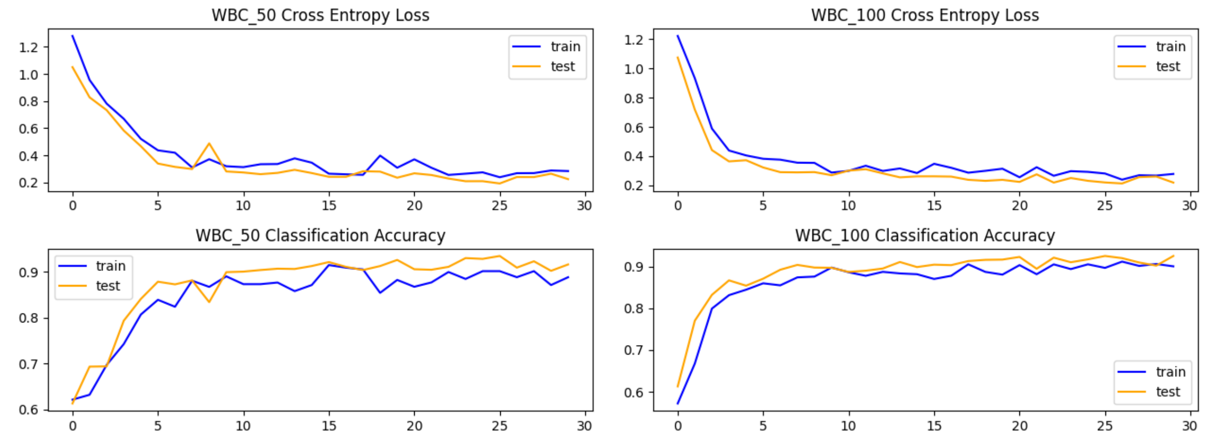
The second attempt I made is to use pre-trained model. I chose ResNet50 as my base model, and using CAM16 to pre train the model. Unfortunately, during this process, my validation accuracy was fixed at 0.5 all the way, while the training accuracy could reach 1, which I still do not figure out the reason. It seems the final model take a strategy that always output one of the two classes(e.g. tumor), but since the metric is accuracy instead of loss, and the training accuracy is not 0.5(the CAM16 dataset is equally divided into 2 classes), this cannot be true. When I am writing the last version of the report, the pre-trained model finally get an accuracy of 0.8519 after 300 epochs, but I have no time to run the pre-trained model to see the validation result using WBC training dataset. I will submit the pre-trained model.

Section 3: Results Output

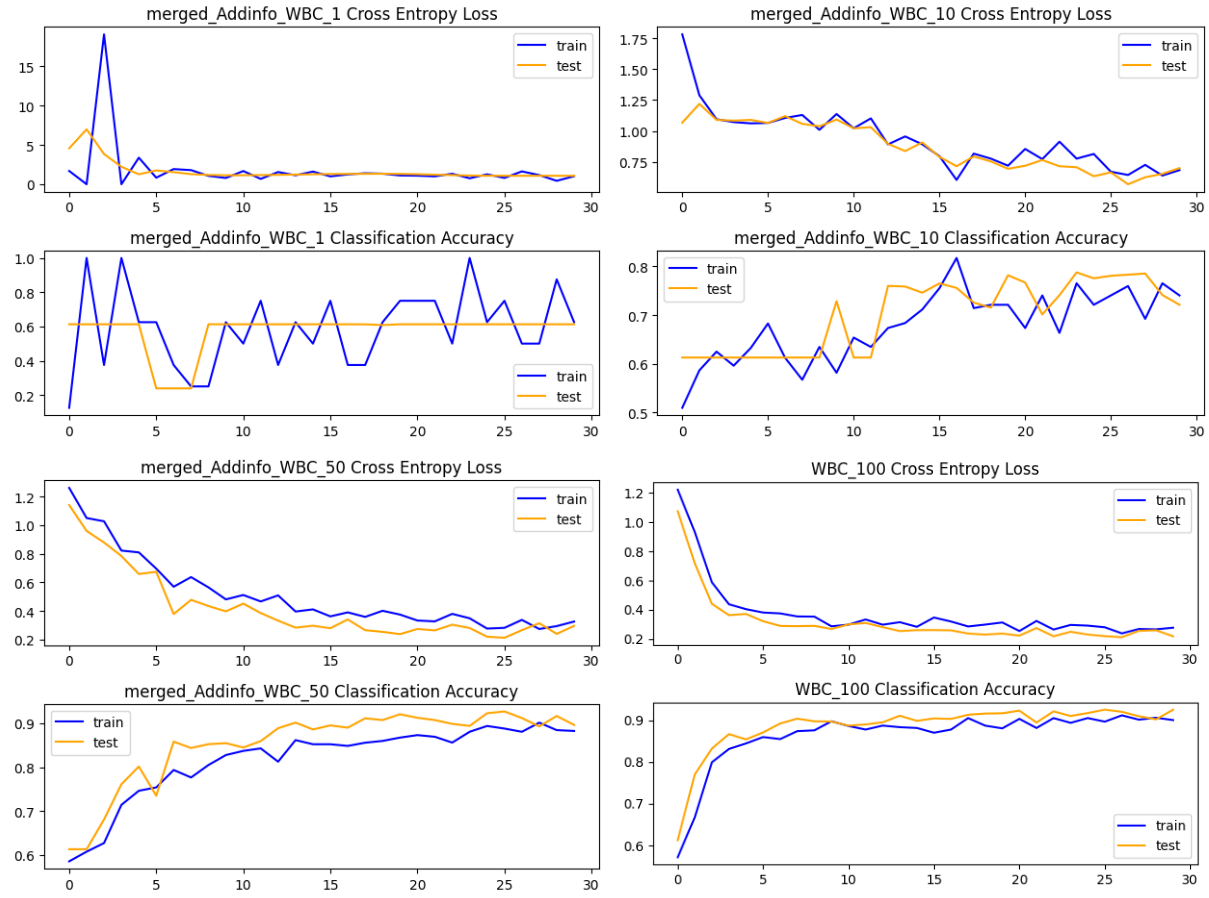
Training result without additional information:

WBC_1:





Training with additional information:



Section 4: Conclusion

From above, the whole process only takes several hours to train the model, and once the model is trained, it could have a quite high accuracy to identify massive cell types even it is just a simple label. With additional knowledge and complex model structure, as well as more data and better parameter, it could help a lot in Hematology identification.

Appendix

	Best Accuracy	Mean Accuracy	Last Epoch Accuracy
WBC_1(w/o add info)	0.6128	0.5754	0.6128
WBC_10(w/o add info)	0.8570	0.7316	0.8397
WBC_50(w/o add info)	0.9346	0.8756	0.9161
WBC_100(w/o add info)	0.9253	0.8861	0.9253
WBC_1(w/ only masked dataset)	0.6128	0.6002	0.6128
WBC_10(w/ only masked dataset)	0.6128	0.5797	0.6128
WBC_50(w/ only masked dataset)	0.6128	0.5879	0.6128
WBC_100(w/ only masked dataset)	0.6163	0.5881	0.6128
WBC_1(merged dataset)	0.6128	0.5752	0.6128
WBC_10(merged dataset)	0.7876	0.7020	0.7211
WBC_50(merged dataset)	0.9271	0.8533	0.8964
WBC_100(merged dataset)	0.9259	0.8805	0.9259

Table 1 Accuracy