ASSIGNMENT 2

Name: 法佑, Student ID: 108062566 05/07/2020

1 Part 1

1.1 Read in csv and print out the field

I use panda package to read csv and print its field. The result is shown below.

```
csv-field: Index(['patient_id', 'breast density', 'left or right breast', 'image view',
       'abnormality id', 'abnormality type', 'calc type', 'calc distribution',
       'assessment', 'pathology', 'subtlety', 'image file path',
       'cropped image file path', 'ROI mask file path'],
      dtype='object')
```

1.2 Print out the image information of both original image and the pre-processed image

The following picture is the information of the pictures before and after the transform. I use crop image (labeled as 'mask' in csv) as data and then use the transform package in pytorch to process the image, including Resize((256, 256)), RandomHorizontalFlip (p=0.5, only in train time), copy the grayscale 1D channel to 2 additional 3D channels and Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225]). In addition, I also balance the data, because the benign is more than the malignant label, I use the up-sampling method to increase the number of malignant labels to the same number as the benign.

```
original image: size ((681, 577)), min (0.0), max (65535.0), mean (42820.609375)
transform image: size ((3, 224, 224)), min (-2.1179039478302), max (2.3138773441314697), mean (0.9053985476493835)
```

2 Part 2 (Try two different classifiers)

2.1 Results of SVM, Kmeans, Random Foreset

I use the package in sklearn to classify images, including SVM, Kmeans, and Random Foreset. It is worth noting that because the input data dimension these algorithms need is two-dimensional, I specifically converted the original four-dimensional image into two dimensions by fixing the first dimension (batch size) and expanding the last three dimensions into one dimension. The results of the three algorithms are shown below (Figure1).

```
---------------------------------- part2 ----------------------------------


----------------- SVM ------------------
accuracy:      61.35 %
f1-score:     0.11267605633802817
recall-score: 0.06201550387596899


---------------- Kmeans ----------------
accuracy:      45.40 %
f1-score:     0.4795321637426901
recall-score: 0.6356589147286822


------------ Random Forest -------------
accuracy:      60.43 %
f1-score:     0.2793296089385475
recall-score: 0.1937984496124031
```

Figure 1: Results of SVM, Kmeans and Random Foreset

2.2 Results of fine-tuning resnet152

I use the pre-trained models provided by pytorch, including alexnet, resnet50, resnet102 and resnet152, and freeze their convolution layers, only update the classifier part, and find that resnet152 has the best effect, and its accuracy rate in test data is close to 72%. The results of the three algorithms are shown below (Figure2). On the other hand, I tried to update the last convolution layer in resnet152, but the effect is not ideal to just update the classifier.
As for the metrics to judge the quality of the model, in addition to accuracy, I also use f1-score and recall, and they are all functions on sklearn.metrics

```
-------------- resnet152 ---------------
load_model: True (71.47239263803681)
Test set: Average loss: 0.0053, Accuracy: 234/326 (71.78%)
accuracy:      71.78 %
f1-score:     0.5779816513761468
recall-score: 0.4883720930232558
```

Figure 2: Results of fine-tuning resnet152

3 Question and Summary

I learned a lot from this assignment, from data processing, such as using transform, dataset and dataloader in panda, pytorch, etc., to building models, including training using sklearn or pytorch packages, and even trying to use transfer learning to train a deep learning network.

4 Used Packages

1. pandas
2. numpy
3. torch
4. pydicom
5. torchvision
6. sklearn
7. argparse