Histopathological Cancer Detection

XIU ZHEN HUANG
13.09.2025

Overview

- Brief Description of Dataset
- Exploratory Data Analysis (EDA)
- Models Detail
  - Logistic Regression
  - Random Forest
  - CNN
  - Transfer Learning (ResNet50)
  - Transfer Learning (DenseNet121)
- Results and Analysis
- Conclusion

The full project locates at the git repository.

Brief Description of Dataset

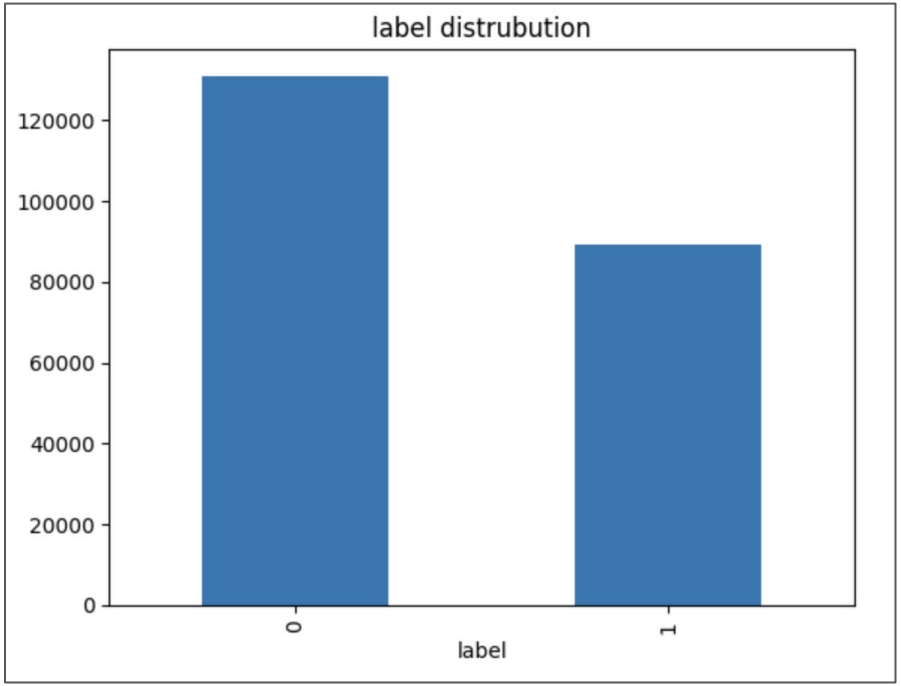
- The "Histopathologic Cancer Detection" dataset available on Kaggle.
- The training dataset includes 220025 histopathologic images labeled 1 or 0, which means the image either containing cancer or not.
- The size of each image is 96x96 pixels with 3-channel (RGB) format.

```
Data shape:(96, 96, 3)
There are 220025 training images and 57458 test images
```

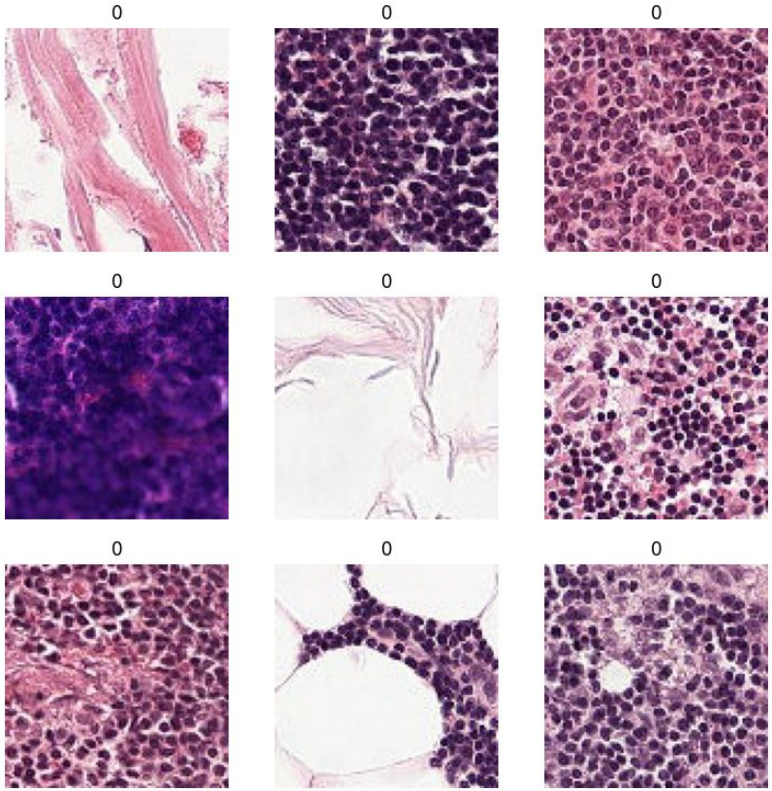
	id	label
0	f38a6374c348f90b587e046aac6079959adf3835	0
1	c18f2d887b7ae4f6742ee445113fa1aef383ed77	1
2	755db6279dae599ebb4d39a9123cce439965282d	0
3	bc3f0c64fb968ff4a8bd33af6971ecae77c75e08	0
4	068aba587a4950175d04c680d38943fd488d6a9d	0

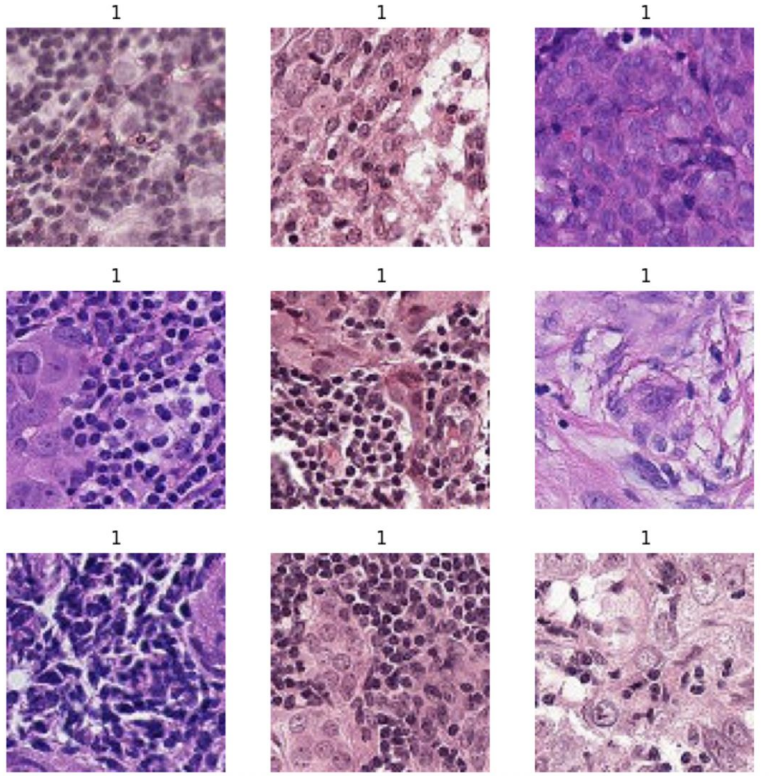
EDA - Label Distribution

There are 89117 positive training images and 130908 negative training images. The training dataset is slightly skewed with more negative samples.



EDA - Sample Data

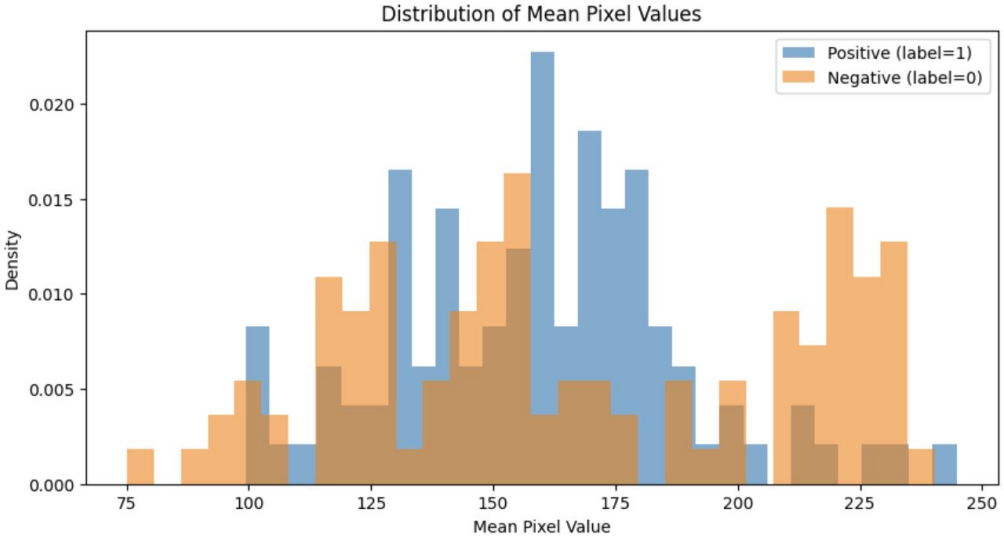




EDA - Image Pixel Distribution

Calculate pixel count with 100 samples in Postive/Negative

- T-test: statistic=-1.4245, p=0.1561
  It means the average value is not different in positive and negative samples.
- Mann-Whitney U: statistic=4700.0000, p=0.4643
  It means the median value is not different in positive and negative samples.
- KS test: statistic=0.2600, p=0.00222
  It means the distrubition shape is really different in positive and negative samples.



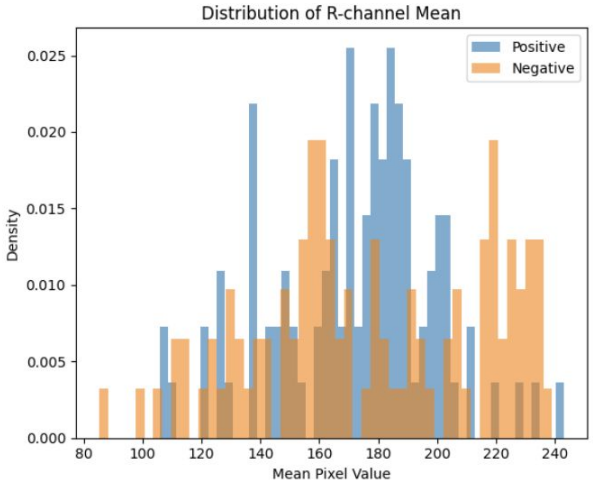
EDA - Color Channel Distrubution

Calculate pixel count with 100 samples in Postive/Negative

T-test: statistic=-0.8953, p=0.3719

Mann-Whitney U: statistic=4756.0000, p=0.5519

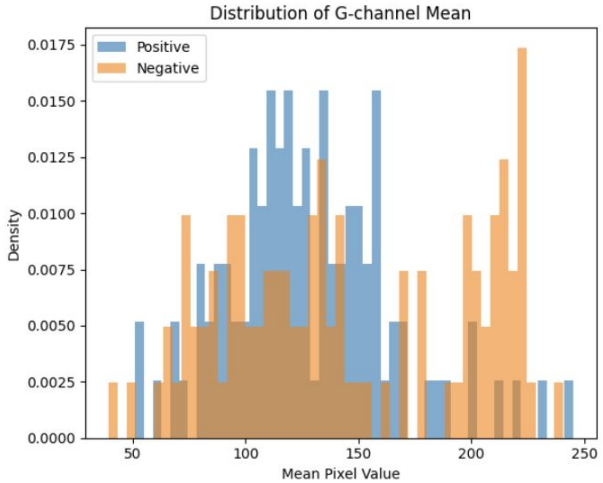
KS test: statistic=0.2600, p=0.00222



T-test: statistic=-2.5207, p=0.01261

Mann-Whitney U: statistic=4278.0000, p=0.07792

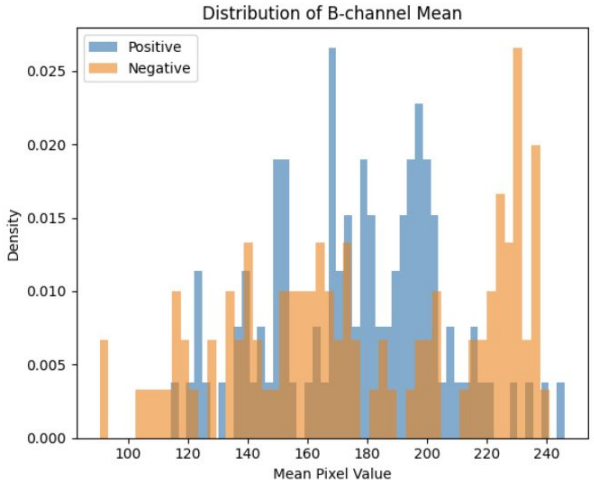
KS test: statistic=0.2800, p=0.0007377222



T-test: statistic=-0.3635, p=0.7167

Mann-Whitney U: statistic=4847.0000, p=0.7094

KS test: statistic=0.2600, p=0.00222



Design Detail

I employed two different approaches for histopathological cancer detection.

First, I implemented a traditional machine learning with handcrafted features, where I used GLCM and LBP for feature extraction. These features were then used to train Logistic Regression and Random Forest models.

Second, I adopted a deep learning approach using Convolutional Neural Networks (CNNs). I compared a baseline CNN with transfer learning models based on ResNet50 and DenseNet121.

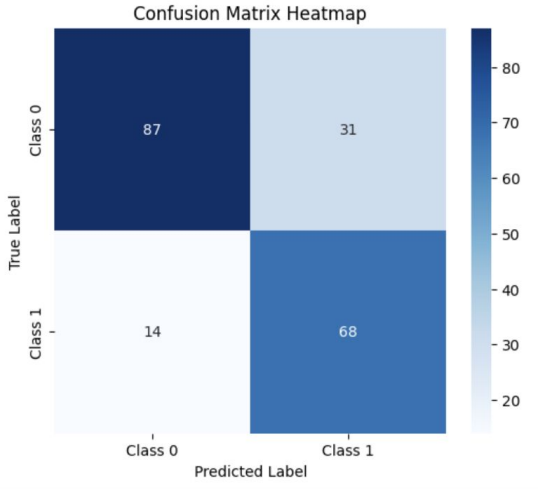
Logistic Regression (GLCM)

- Gray-Level Co-occurrence Matrix (GLCM) was used for feature extraction
- Six texture features — contrast, dissimilarity, homogeneity, energy, correlation, and ASM — were computed from the GLCM using graycoprops
- The model optimized the log-loss.

```
ACC: 0.775
AUC: 0.8662670525010335
[[87 31]
 [14 68]]
              precision    recall  f1-score   support

           0     0.8614    0.7373    0.7945       118
           1     0.6869    0.8293    0.7514        82

    accuracy                         0.7750       200
   macro avg     0.7741    0.7833    0.7730       200
weighted avg     0.7898    0.7750    0.7768       200
```



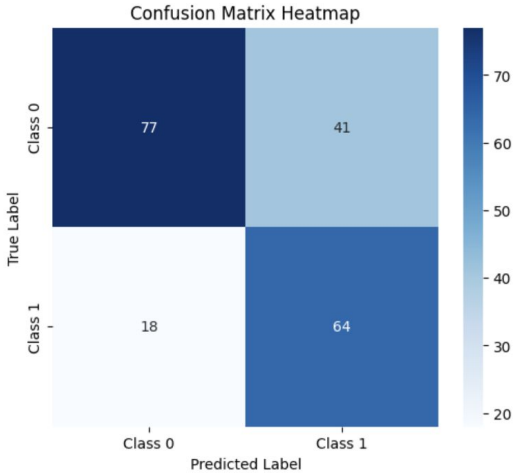
Logistic Regression (LBP)

- LBP was used for feature extraction
- The model optimized the binary cross-entropy loss.

```
ACC: 0.705
AUC: 0.8052914427449358
[[77 41]
 [18 64]]
              precision    recall  f1-score   support

           0     0.8105    0.6525    0.7230       118
           1     0.6095    0.7805    0.6845        82

    accuracy                         0.7050       200
   macro avg     0.7100    0.7165    0.7037       200
weighted avg     0.7281    0.7050    0.7072       200
```



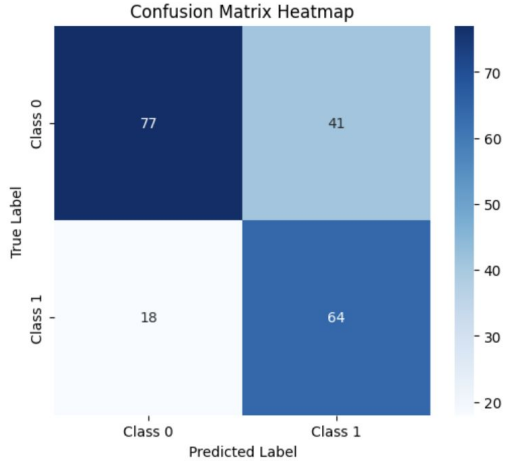
Random Forest (GLCM)

- Gray-Level Co-occurrence Matrix (GLCM) was used for feature extraction
- Six texture features — contrast, dissimilarity, homogeneity, energy, correlation, and ASM — were computed from the GLCM using graycoprops
- The ensemble consisted of 400 decision trees. Each tree was grown to full depth without pruning, and nodes were split if they contained at least 2 samples.

```
ACC: 0.785
AUC: 0.8612546506821
[[77 41]
 [18 64]]
              precision    recall  f1-score   support

           0     0.8151    0.8220    0.8186       118
           1     0.7407    0.7317    0.7362        82

    accuracy                         0.7850       200
   macro avg     0.7779    0.7769    0.7774       200
weighted avg     0.7846    0.7850    0.7848       200
```



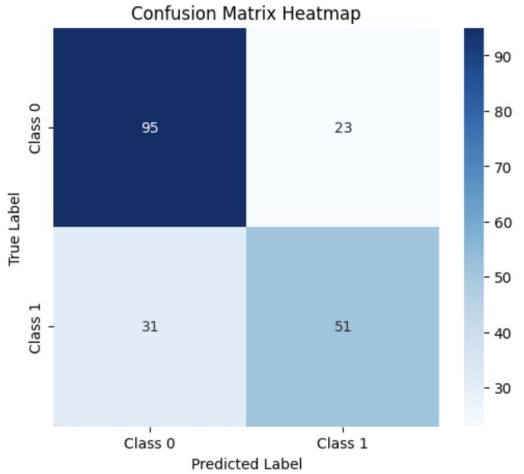
Random Forest (LBP)

- LBP was used for feature extraction
- The model optimized the binary cross-entropy loss.
- The ensemble consisted of 400 decision trees. Each tree was grown to full depth without pruning, and nodes were split if they contained at least 2 samples.

```
ACC: 0.73
AUC: 0.801829268292683
[[95 23]
 [31 51]]
              precision    recall  f1-score   support

           0     0.7540    0.8051    0.7787       118
           1     0.6892    0.6220    0.6538        82

    accuracy                         0.7300       200
   macro avg     0.7216    0.7135    0.7163       200
weighted avg     0.7274    0.7300    0.7275       200
```



CNN

- As the number of convolutional kernels increases, the network captures low-level features (color, edges) progressing to high-level features (shapes, structures).
- Dropout set to 0.5 to avoid overfitting.
- Use ‘relu’ as activation function.

Layer (type)	Output Shape	Param #
conv2d_3 (Conv2D)	(None, 94, 94, 32)	896
max_pooling2d_3 (MaxPooling2D)	(None, 47, 47, 32)	0
conv2d_4 (Conv2D)	(None, 45, 45, 64)	18,496
max_pooling2d_4 (MaxPooling2D)	(None, 22, 22, 64)	0
conv2d_5 (Conv2D)	(None, 20, 20, 128)	73,856
max_pooling2d_5 (MaxPooling2D)	(None, 10, 10, 128)	0
flatten_1 (Flatten)	(None, 12800)	0
dense_2 (Dense)	(None, 256)	3,277,056
dropout_1 (Dropout)	(None, 256)	0
dense_3 (Dense)	(None, 1)	257

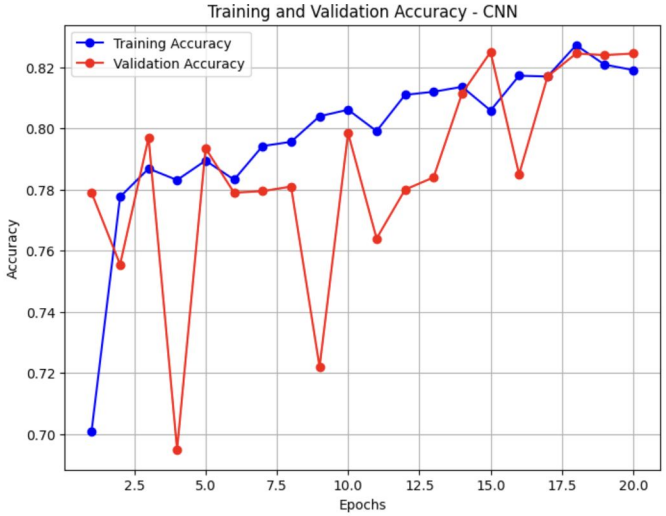
Total params: 3,370,561 (12.86 MB)

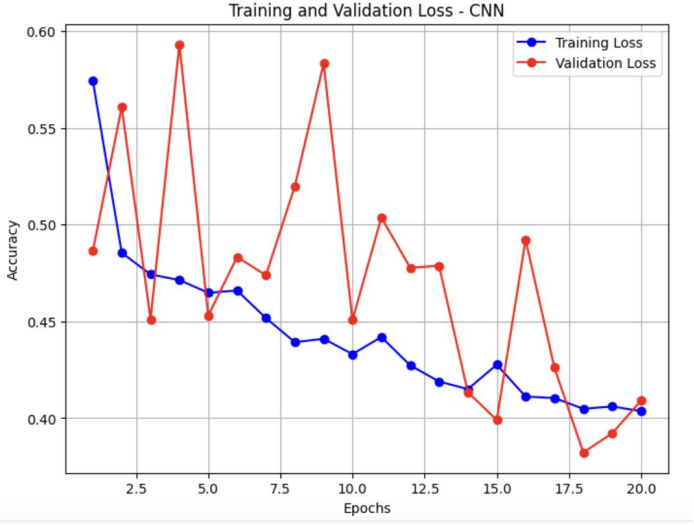
Trainable params: 3,370,561 (12.86 MB)

Non-trainable params: 0 (0.00 B)

CNN

- I only used 10000 samples to train the model because the training process is time cost. Additionally, I trained the network for 20 epochs.
- We can see accuracy is about 0.8144 at 20 epoch. And loss rate also decreased to only 0.41135.





Transfer Learning (ResNet50)

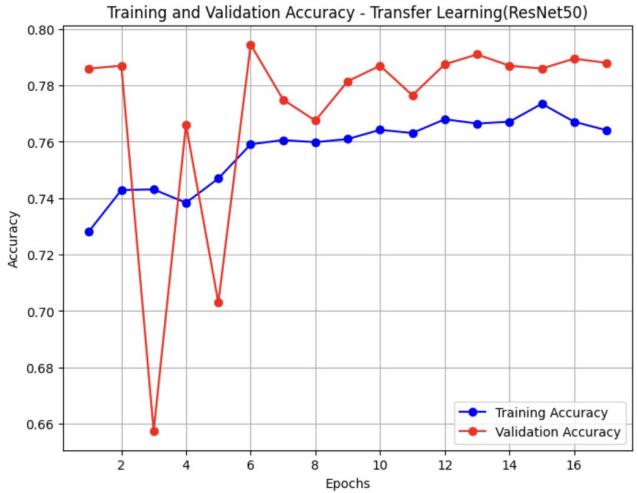
- Use ResNet50 pre-training model as feature extraction.
- Concatenate GAP and GMP to stabiliz the feature representation and get the significant features.
- Dropout 4096 to 1024 to avoid overfitting.
- Use BatchNorm to do normalization.

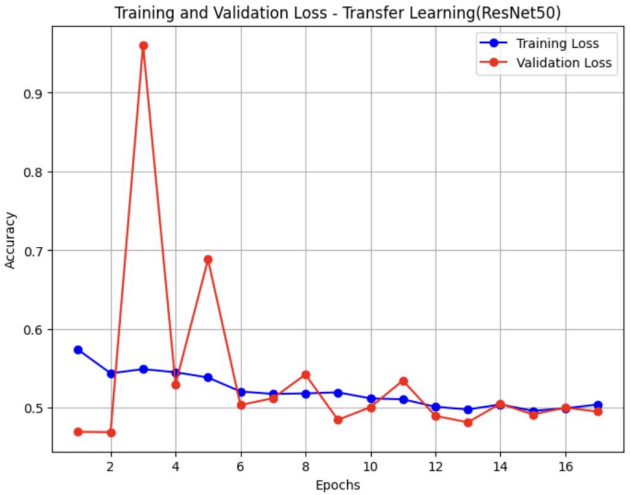
Layer (type)	Output Shape	Param #	Connected to
input_layer_5 (InputLayer)	(None, 96, 96, 3)	0	-
resnet50 (Functional)	(None, 3, 3, 2048)	23,587,712	input_layer_5[0]…
global_average_poo… (GlobalAveragePool…	(None, 2048)	0	resnet50[0][0]
global_max_pooling… (GlobalMaxPooling2…	(None, 2048)	0	resnet50[0][0]
concatenate_1 (Concatenate)	(None, 4096)	0	global_average_p… global_max_pooli…
batch_normalizatio… (BatchNormalizatio…	(None, 4096)	16,384	concatenate_1[0]…
dropout_4 (Dropout)	(None, 4096)	0	batch_normalizat…
dense_6 (Dense)	(None, 1024)	4,195,328	dropout_4[0][0]
batch_normalizatio… (BatchNormalizatio…	(None, 1024)	4,096	dense_6[0][0]
dropout_5 (Dropout)	(None, 1024)	0	batch_normalizat…
dense_7 (Dense)	(None, 1)	1,025	dropout_5[0][0]

Total params: 27,804,545 (106.07 MB)
Trainable params: 19,184,641 (73.18 MB)
Non-trainable params: 8,619,904 (32.88 MB)

Transfer Learning (ResNet50)

- I only used 10000 samples to train the model because the training process is time cost. Additionally, I trained the network for 20 epochs.
- We can see accuracy is about 0.7736 at 20 epoch. And loss rate also decreased to only 0.4923.





Transfer Learning (DenseNet121)

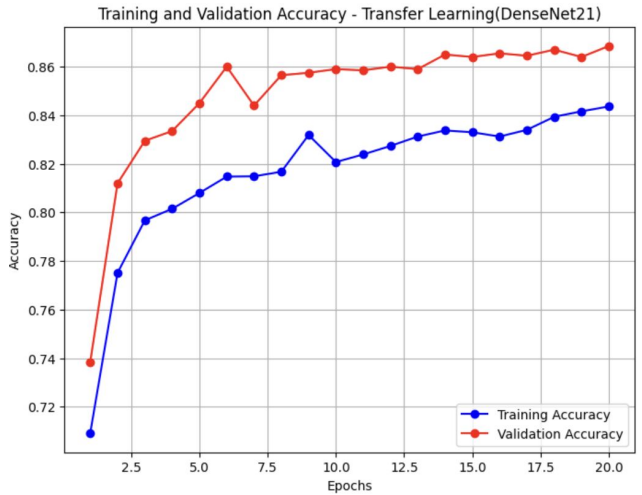
- Use DenseNet121 pre-training model as feature extraction.
- Concatenate GAP and GMP to stabilize the feature representation and get the significant features.
- A fully connected layer with 4096 units was reduced to 1024 units with Dropout to mitigate overfitting.
- Use BatchNorm to do normalization.

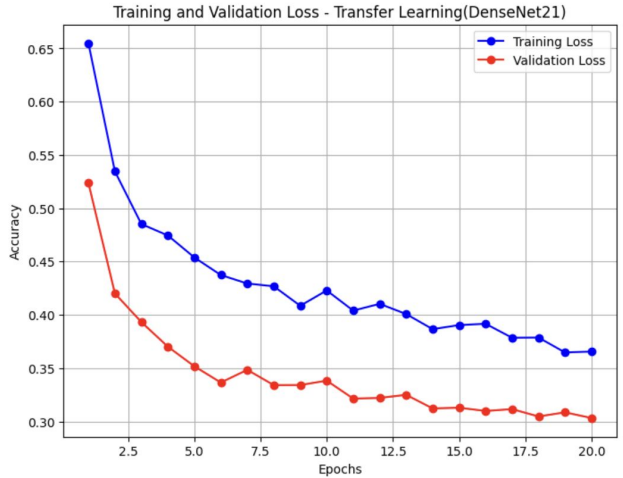
Layer (type)	Output Shape	Param #	Connected to
input_layer_11 (InputLayer)	(None, 96, 96, 3)	0	–
densenet121 (Functional)	(None, 3, 3, 1024)	7,037,504	input_layer_11[0…
global_average_poo… (GlobalAveragePool…	(None, 1024)	0	densenet121[0][0]
global_max_pooling… (GlobalMaxPooling2…	(None, 1024)	0	densenet121[0][0]
concatenate_4 (Concatenate)	(None, 2048)	0	global_average_p… global_max_pooli…
batch_normalizatio… (BatchNormalizatio…	(None, 2048)	8,192	concatenate_4[0]…
dropout_10 (Dropout)	(None, 2048)	0	batch_normalizat…
dense_12 (Dense)	(None, 256)	524,544	dropout_10[0][0]
batch_normalizatio… (BatchNormalizatio…	(None, 256)	1,024	dense_12[0][0]
dropout_11 (Dropout)	(None, 256)	0	batch_normalizat…
dense_13 (Dense)	(None, 1)	257	dropout_11[0][0]

Total params: 7,571,521 (28.88 MB)
Trainable params: 529,409 (2.02 MB)
Non-trainable params: 7,042,112 (26.86 MB)

Transfer Learning (DenseNet121)

- I only used 10000 samples to train the model because the training process is time cost. Additionally, I trained the network for 20 epochs.
- We can see accuracy is about 0.8478 at 20 epoch. And loss rate also decreased to only 0.3543.





Results and Analysis

The results show that traditional machine learning models using handcrafted features achieve moderate performance, with Random Forest slightly outperforming Logistic Regression.

Model	Accuracy
Logistic Regression (GLCM features)	~77.5%
Logistic Regression (LBP features)	~70.5%
Random Forest (GLCM features)	~78.5%
Random Forest (LBP features)	~73%
CNN	~81.44%
Transfer Learning (ResNet50)	~77.36%
Transfer Learning (DenseNet121)	~84.78%

The baseline CNN model provides a noticeable improvement over classical approaches, indicating that deep feature extraction is more effective than handcrafted features for histopathological image classification.

Among all models, Transfer Learning with DenseNet121 achieves the highest accuracy (~84.78%), showing the advantage of leveraging pre-trained deep architectures for this task.

However, it should be emphasized that these results are based on a reduced training set (10,000 samples) rather than the full dataset, and therefore they should be interpreted as indicative rather than conclusive.

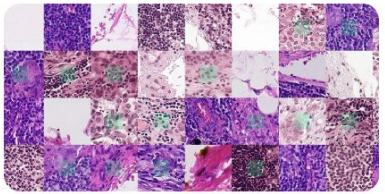
Conclusion

- Traditional ML models with handcrafted features achieved moderate accuracy.
- The baseline CNN outperformed them by learning richer image representations.
- Transfer Learning, especially with DenseNet121, provided the best performance (~84.78%), highlighting the effectiveness of pre-trained deep networks for histopathological cancer detection.
- Future work will include training on the full dataset and exploring advanced augmentation and regularization techniques to further improve performance.

Kaggle Dashboard

Histopathologic Cancer Detection

Identify metastatic tissue in histopathologic scans of lymph node sections



Overview Data Code Models Discussion Leaderboard Rules Team Submissions

Leaderboard

Raw Data Refresh

YOUR RECENT SUBMISSION

sample_submission.csv
Submitted by HUANGXIUZHEN · Submitted 3 days ago

Score: 0.8173
Public score: 0.8410

Jump to your leaderboard position

Reference

- Medical images texture analysis: A review
- Setting the learning rate of your neural network
- Classification of Breast Cancer Histopathological Images Using Transfer Learning with DenseNet121