DIAGNOSIS OF BREAST CANCER USING DEEP LEARNING

SUMMER INTERNSHIP PROJECT REPORT

by

RASHI ANUBHI SRIVASTAVA

Department of Electrical Engineering

National Institute of Technology, Andhra Pradesh

May-August, 2020

Intern under

Dr. Sri Phani Krishna Karri

Date of Completion: 2nd August, 2020

Declaration by Author

This is to declare that this report has been written by me. No part of the report is plagiarized from other sources. All information included from other sources have been duly acknowledged. I aver that if any part of the report is found to be plagiarized, I shall take full responsibility for it.

Rashi Anubhi Srivastava

ABSTRACT

Breast cancer affects one out of eight females worldwide. It is diagnosed by detecting the malignancy of the cells of breast tissue. Modern medical image processing techniques work on histopathology images captured by a microscope, and then analyze them by using different algorithms and methods. Machine learning algorithms are now being used for processing medical imagery and pathological tools. Manual detection of a cancerous cell maybe a tiresome task and involves human error, and hence computer-aided mechanisms are applied to get better results as compared with manual pathological detection systems. In deep learning, this is often done by extracting features through a convolutional neural network (CNN) and then classifying employing a fully connected network. Deep learning is extensively utilized within the medical imaging field, as it does not require prior expertise in a related field. In this project, I have trained a convolution network and obtained prediction accuracy.

NOMENCLATURE

1. EDA – Exploratory Data Analysis
2. IDC – Invasive Ductal Carcinoma
3. H&E Stain – Hematoxylin & Eosin Stain
4. RGB – Red Green Blue light
5. CNN – Convolutional Neural Network
6. Conv - Convolution

TABLE OF CONTENTS

INTRODUCTION AND PROJECT DESCRIPTION

1.1 Problem Statement

Breast cancer is the most common form of cancer in women, and invasive ductal carcinoma (IDC) is the most common form of breast cancer. Accurately identifying and categorizing breast cancer subtypes is an important clinical task, and automated methods can be used to save time and reduce error. To analyze the cellular structures in the breast histology images we were instead leveraging basic computer vision and image processing algorithms, but combining them in a novel way. These algorithms worked really well — but also required quite a bit of work to put together.

1.2 Problem Description

As deep learning researchers, practitioners, and engineers it's important for us to gain hands-on experience applying deep learning to medical and computer vision problems — this experience can help us develop deep learning algorithms to better aid pathologists in predicting cancer.

The goal of this script is to identify IDC when it is present in otherwise unlabeled histopathology images. The dataset consists of 277,524 50x50 pixel RGB digital image patches that were derived from 162 H&E-stained breast histopathology samples. These images are small patches that were extracted from digital images of breast tissue samples. The breast tissue contains many cells but only some of them are cancerous. Patches that are labeled "1" contain cells that are characteristic of invasive ductal carcinoma.

- The dataset was downloaded from Kaggle, breast-histopathology-images.
- Spyder was used as a module to train and test the data.
- The model architecture used was CancerNet taken from Pyimagesearch.
- Preprocessing and data augmentation were carried out on downloaded data.
- The module was first supplied with training split, where the model was trained and weights were saved and then the accuracy of testing split was tested using CNN and confusion matrix respectively.
- The training data was processed by applying feature extraction.

- The test data obtained was tested for accuracy. The results were noted and recorded for further analysis.
- The model was also used for making predictions on raw images taken directly from the internet.

2. METHODS USED TO ACHIEVE OBJECTIVES

2.1 Workflow of the Project

Understanding the problem Statement.

⇩

Researching about problem statement and getting an insight into the possible solutions.

⇩

Structuring my own solution to the proposed problem.

⇩

Understanding the dataset and its structure.

⇩

Learning about deep learning algorithms to be used.

⇩

Learning about CNNs and usage as classifier.

⇩

Pre-processing the data given.

⇩

Performing EDA using python.

⇩

Testing different CNN architectures to select the best suited and accurate one for the model.

⇩

Started working on model.

⇩

Obtained a model to give predictions on histology images and classify them as benign or malignant.

2.2 Approach Used

To achieve the proposed result, Deep Learning technique was used for feature extraction and eventually classification of the images in two categories i.e. benign and malignant.

CancerNet a network appropriately designed for histology image classifications was used which is designed using Keras deep learning library which:

- Uses exclusively *3×3* CONV filters, similar to VGGNet.
- Stacks multiple *3×3* CONV filters on top of each other prior to performing max-pooling (again, similar to VGGNet).
- But unlike VGGNet, uses depth wise separable convolution rather than standard convolution layers.

The depth wise separable convolution:

- Is more efficient.
- Requires less memory.
- Requires less computation.
- Can perform better than standard convolution in some situations.

2.3 Implementation and Tools

2.3.1. Tools Used:

- Spyder IDE
- Python Packages

2.3.2. Implementation:

Dataset Split:

The dataset was separated into train, test and validation set. The train set consisted 80% of the total while the rest resided with test set. The validation set was taken out from tests set as a small part.

Data Augmentation:

Data augmentation, a form of regularization, is important for nearly all deep learning experiments to assist with model generalization. The method purposely perturbs training examples, changing their appearance slightly, before passing them into the network for training. This partially alleviates the need to gather more training data, though more training data will rarely hurt your model. Our images were of size 50*X*50. Random rotations, shifts, shears and flips were applied to our data. Rescaling of our image pixel intensities to the [0,1] was done.

Network Architecture:

Our network consists of 3 Conv blocks stacked one over the other comprising of batch normalizing, max-pooling and dropout layers in each one. The activation function used is Relu and Adagrad optimizer has been used. The last two layers consist of fully connected layers, output of which is then given to SoftMax classifier for final output.

Classification:

The highest prediction indices are grabbed for each sample and then a classification report is prepared conveniently. Confusion matrix is computed and then the accuracy is derived.

Saving model and weights:

Model and its weights were saved in a Json and H5 file format respectively which was then used to make predictions on raw images.

3. RESULTS AND DISCUSSIONS

Our model achieved the accuracy of around 84%, however, that raw accuracy is heavily weighted by the fact that we classified *"benign/no cancer"* correctly 93% of the time.

To understand our model's performance at a deeper level we compute the sensitivity and the specificity.

Our sensitivity measures the proportion of the *true positives* that were also predicted as positive (83.07%).

Conversely, specificity measures our *true negatives* (85.29%).

A total of 277,524 images belonging to two classes are included in the dataset:

Positive (+): 78,786

Negative (-): 198,738

Here we can see there is a class imbalance in the data with over *2x* more negative samples than positive samples.

The class imbalance, along with the challenging nature of the dataset, lead to us obtaining ~83% classification accuracy, ~83% sensitivity, and ~85% specificity.

4. CONCLUSIONS AND RECOMMENDATIONS

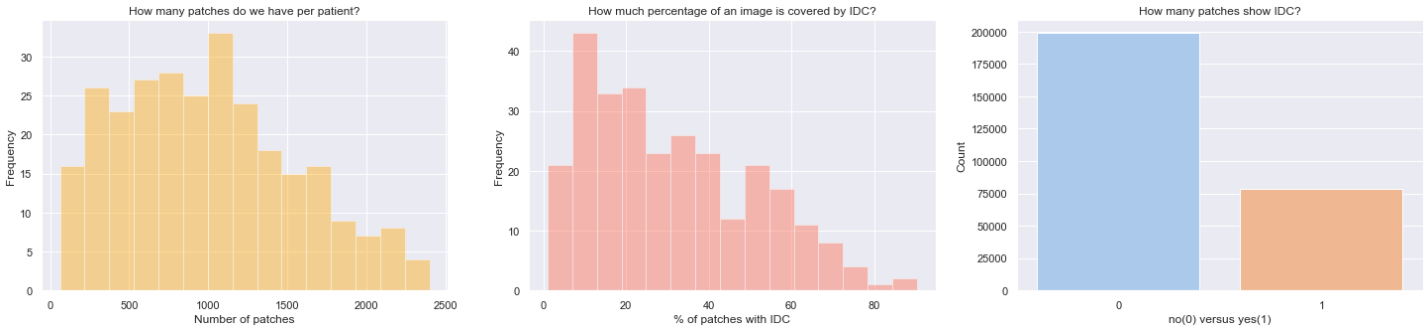
Given 277,524 images, with 78,786 belonging to positive class and the rest to negative, our model gives 83% of classification accuracy, 83% sensitivity and 85% specificity.

Breast cancer detection by using digital/digitized histopathology images is a milestone in the field of medical pathology. It has also opened a door to new opportunities for research as there are many undiscovered areas that can be revealed by techniques and tools of machine learning and deep learning. We may obtain improved results by altering the network design and parameters. From the point of method improvement, we can incorporate spectral imaging. Spectral imaging is used to obtain images with various wavelengths, which is different from the trivial three channel RGB image. Additionally, we may combine various imaging technologies such as MRI, CT Scan, ultrasound, and mammographic images, and determine their collective results. This technique is known as multimodal fusion. Problems stated above can again readily be solved by deep learning, and can be used to perform high quality research that might provide even better results.

5. REFERENCES

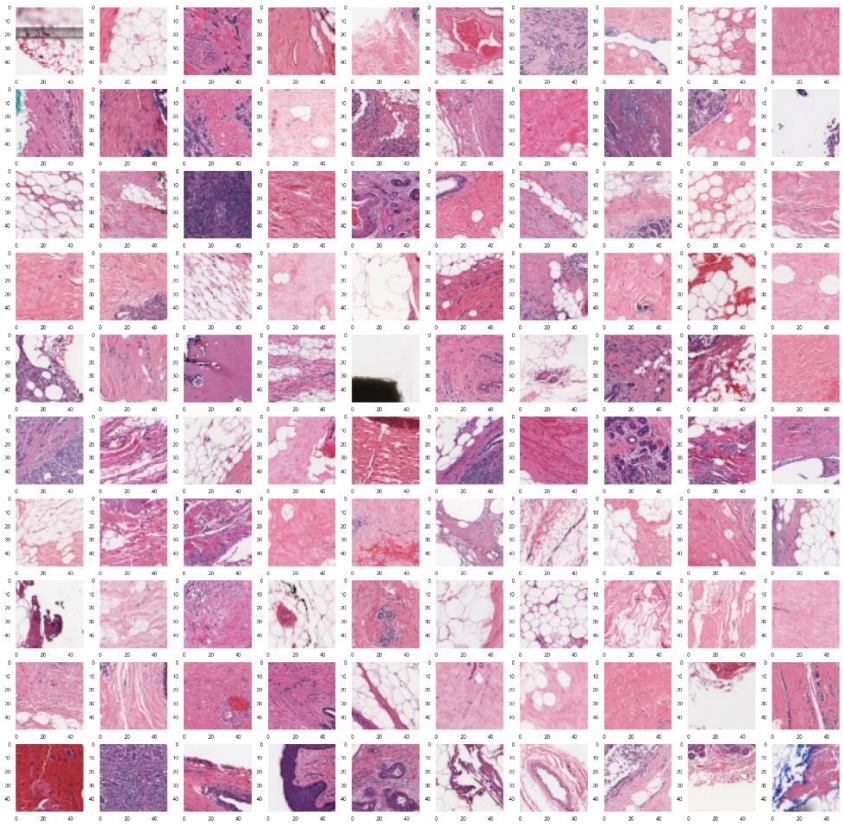
1. Dataset, Kaggle, https://www.kaggle.com/paultimothymooney/breast-histopathology-images
2. Adrian Rosebrock, pyimagesearch, Breast Cancer Classification with keras and Deep Learning.
3. Sumaiya Dabeer, Maha Mohammed Khan, and Saiful Islam, “Cancer diagnosis in histopathological image: CNN based approach”, Department of Computer Engineering Zakir Husain College of Engineering and Technology Aligarh, India
4. Breast cancer Classification End to End, Kaggle.com

APPENDICES

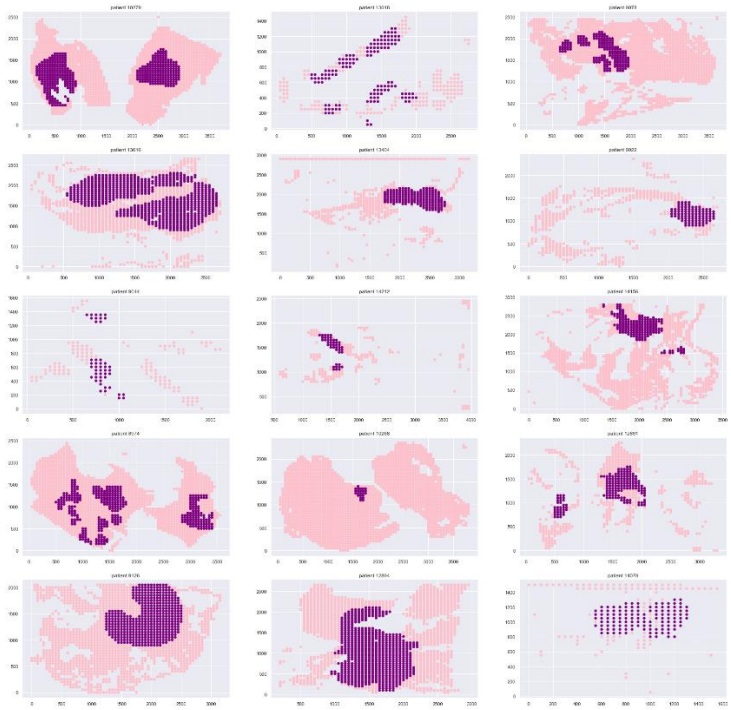


EDA Plots: 1. Patches per patient, 2. Percentage of images covered by IDC, 3. Total no. of patches showing IDC

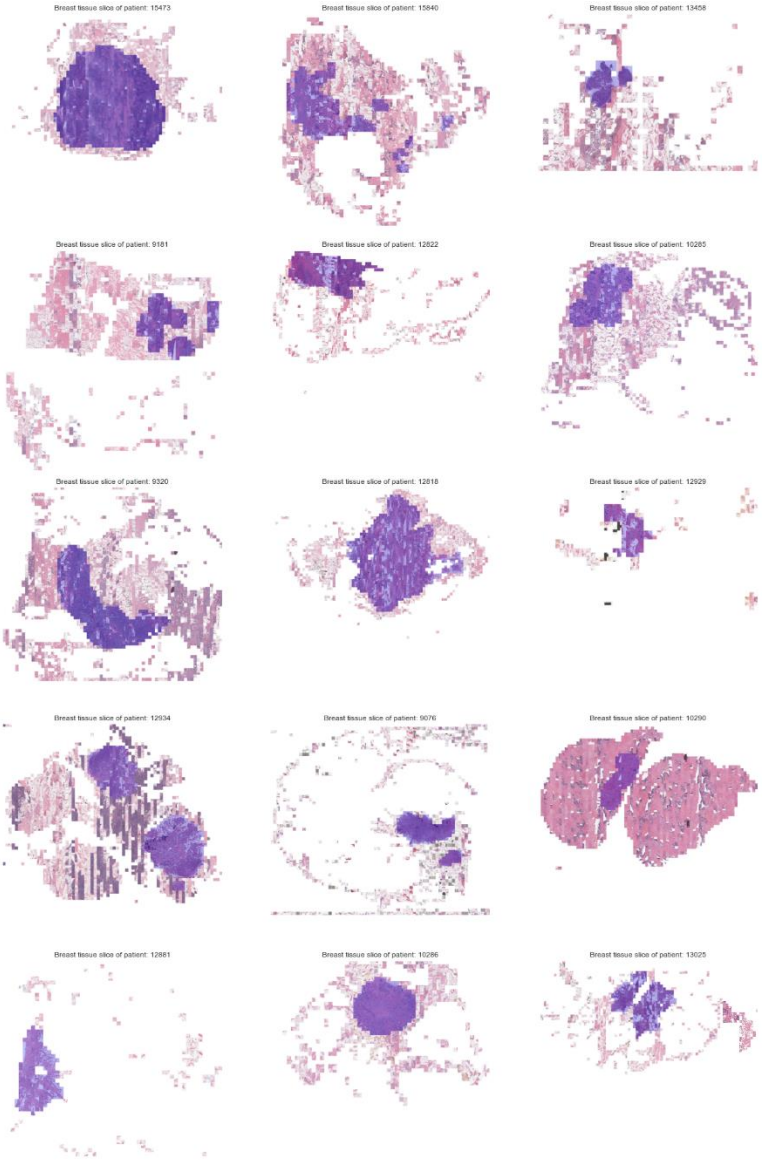
Patches infected with IDC showing darker and greater violet regions



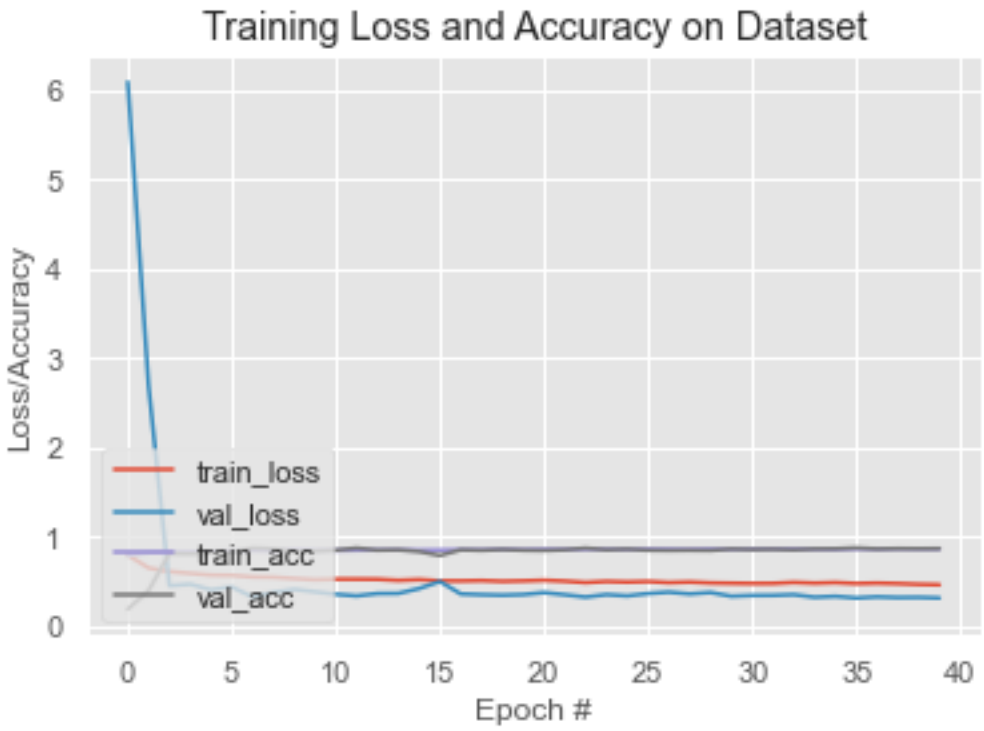
Healthy Patches



Reconstructed breast tissue heatmaps: violet areas depicting IDC infected regions



Reconstruction of breast tissue using actual patches and concatenating them to form the complete region.



Accuracy Plot

```
In [5]: runfile('F:/cancer_project/Exploratory.py', wdir='F:/cancer_project')
import complete
(277524, 5)
  patient_id     x  ...  target                                               path
0      10253  1001  ...       0  F:\cancer_project\Dataset\IDC_regular_ps50_idx...
1      10253  1001  ...       0  F:\cancer_project\Dataset\IDC_regular_ps50_idx...
2      10253  1001  ...       0  F:\cancer_project\Dataset\IDC_regular_ps50_idx...
3      10253  1001  ...       0  F:\cancer_project\Dataset\IDC_regular_ps50_idx...
4      10253  1001  ...       0  F:\cancer_project\Dataset\IDC_regular_ps50_idx...

[5 rows x 5 columns]
```

Output on terminal while running exploratory.py script

```
In [9]: runfile('F:/cancer_project/breast-cancer-classification/build_dataset.py', wdir='F:/
cancer_project/breast-cancer-classification')
Reloaded modules: main, main.config
[INFO] building 'training' split
[INFO] 'creating F:\cancer_project\Images2\training' directory
[INFO] 'creating F:\cancer_project\Images2\training\0' directory
[INFO] 'creating F:\cancer_project\Images2\training\1' directory
[INFO] building 'validation' split
[INFO] 'creating F:\cancer_project\Images2\validation' directory
[INFO] 'creating F:\cancer_project\Images2\validation\1' directory
[INFO] 'creating F:\cancer_project\Images2\validation\0' directory
[INFO] building 'testing' split
[INFO] 'creating F:\cancer_project\Images2\testing' directory
[INFO] 'creating F:\cancer_project\Images2\testing\0' directory
[INFO] 'creating F:\cancer_project\Images2\testing\1' directory

In [10]:
```

Output on terminal while running build_dataset.py script

```
In [11]: runfile('F:/cancer_project/breast-cancer-classification/train_model.py', wdir='F:/
cancer_project/breast-cancer-classification')
Reloaded modules: main, main.cancernet, tmpf77a___q, main.config
Found 199818 images belonging to 2 classes.
Found 22201 images belonging to 2 classes.
Found 55505 images belonging to 2 classes.
Epoch 1/40
WARNING:tensorflow:AutoGraph could not transform <function
Model.make_train_function.<locals>.train_function at 0x000001900439ED30> and will run it as-
is.
Please report this to the TensorFlow team. When filing the bug, set the verbosity to 10 (on
Linux, `export AUTOGRAPH_VERBOSITY=10`) and attach the full output.
Cause:
To silence this warning, decorate the function with @tf.autograph.experimental.do_not_convert
WARNING: AutoGraph could not transform <function
Model.make_train_function.<locals>.train_function at 0x000001900439ED30> and will run it as-
is.
Please report this to the TensorFlow team. When filing the bug, set the verbosity to 10 (on
Linux, `export AUTOGRAPH_VERBOSITY=10`) and attach the full output.
Cause:
To silence this warning, decorate the function with @tf.autograph.experimental.do_not_convert
 282/6244 [>.............................] - ETA: 1:25:17 - loss: 0.7971 - accuracy: 0.7541
```

Output on terminal while running train_model.py script

```
Epoch 2/40
6244/6244 [==============================] - 3240s 519ms/step - loss: 0.5598 - accuracy:
0.8288 - val_loss: 0.4129 - val_accuracy: 0.8389
Epoch 3/40
6244/6244 [==============================] - 3566s 571ms/step - loss: 0.5505 - accuracy:
0.8326 - val_loss: 0.4059 - val_accuracy: 0.8416
Epoch 4/40
6244/6244 [==============================] - 3432s 550ms/step - loss: 0.5461 - accuracy:
0.8336 - val_loss: 0.4097 - val_accuracy: 0.8413
Epoch 5/40
6244/6244 [==============================] - 3518s 563ms/step - loss: 0.5441 - accuracy:
0.8344 - val_loss: 0.4038 - val_accuracy: 0.8434
Epoch 6/40
6244/6244 [==============================] - 3731s 598ms/step - loss: 0.5436 - accuracy:
0.8350 - val_loss: 0.4038 - val_accuracy: 0.8420
Epoch 7/40
6244/6244 [==============================] - 3505s 561ms/step - loss: 0.5422 - accuracy:
0.8344 - val_loss: 0.4167 - val_accuracy: 0.8416
Epoch 8/40
6244/6244 [==============================] - 3416s 547ms/step - loss: 0.5383 - accuracy:
0.8359 - val_loss: 0.4092 - val_accuracy: 0.8431
Epoch 9/40
6244/6244 [==============================] - 3534s 566ms/step - loss: 0.5383 - accuracy:
0.8354 - val_loss: 0.4068 - val_accuracy: 0.8430
Epoch 10/40
6244/6244 [==============================] - 2797s 448ms/step - loss: 0.5363 - accuracy:
0.8365 - val_loss: 0.4067 - val_accuracy: 0.8437
Epoch 11/40
6244/6244 [==============================] - 1740s 279ms/step - loss: 0.5373 - accuracy:
0.8365 - val_loss: 0.4093 - val_accuracy: 0.8438
Epoch 12/40
6244/6244 [==============================] - 3377s 541ms/step - loss: 0.5376 - accuracy:
0.8358 - val_loss: 0.4107 - val_accuracy: 0.8433
Epoch 13/40
6244/6244 [==============================] - 2361s 378ms/step - loss: 0.5374 - accuracy:
```

Output on terminal while running train_model.py script: showing number of epochs.

```
Epoch 39/40
6244/6244 [==============================] - 2038s 326ms/step - loss: 0.5326 - accuracy:
0.8388 - val_loss: 0.4093 - val_accuracy: 0.8445
Epoch 40/40
6244/6244 [==============================] - 2016s 323ms/step - loss: 0.5318 - accuracy:
0.8384 - val_loss: 0.4095 - val_accuracy: 0.8445
[INFO] evaluating network...
              precision    recall  f1-score   support

           0       0.89      0.89      0.89     39793
           1       0.73      0.71      0.72     15712

    accuracy                           0.84     55505
   macro avg       0.81      0.80      0.80     55505
weighted avg       0.84      0.84      0.84     55505

[[35560  4233]
 [ 4530 11182]]
acc: 0.8421
sensitivity: 0.8936
specificity: 0.7117
```

Output on terminal while running train_model.py script: showing our evaluation metrices, confusion metric, accuracy, sensitivity and specificity.

```
In [3]: runfile('F:/cancer_project/cancer/new_image.py', wdir='F:/cancer_project/cancer')
Reloaded modules: main, main.cancernet
Loaded model from disk
Input image shape: (1, 48, 48, 3)
class prediction vector [p(0), p(1) =
[[0.00746956 0.9925304 ]]
```

Output on terminal while new_image_pred.py script: Final prediction probability on complete raw images (model was trained using segmented patches) taken directly from other sources having size far greater than the training images.