TOPIC: MEDICAL IMAGE ANALYSIS AND CLASSIFICATION

Contents

List of Tables

List of Figures

AIMS OF MEDICAL IMAGE ANALYSIS AND CLASSIFICATION

- To build an interactive Deep Learning based system that allows users to classify medical images such as blood and breast using a web-based application.

OBJECTIVES

- To implement state-of-art novel machine learning solutions to support both BloodMNIST and BreastMNIST datasets.
- To provide a complete standalone solution with dynamic interactions.
- To analyze the performance of existing and pre-trained models.
- To accurately classify the image dataset.

INTRODUCTION

- The datasets are images of patients who have a type of blood cancer. The dataset was constructed for 8 types of blood cancer.
- The other dataset which used was of breast cancer, to predict if a patient has breast cancer. The classes are 0(benign) and 1(malignant)
- The given image sizes were 28x28, which makes them very less in resolution. So, for a human to segregate them will be a task next to impossible.
- Since the images are already grouped in classes by the MedMNIST team. We will use the dataset to build a neural network using TensorFlow and Keras packages provided by python to assess the performance of the neural network.
- We have built 6 neural networks, 3 of which are our models and the other three are the pre-trained models that we have used on the given datasets.

ANALYSIS OF EXSISTING DEEP LEARNING MODELS

Two existing models were implemented for the given dataset which are mentioned below:

1. VGG16 for BreastMNIST
2. ResNet50 for BloodMNIST

VGG16 for BreastMNIST

- VGG16(Builtin.com, no date) is a 16-layer deep neural network. A VGG network consists of small convolution filters. VGG16 has three fully connected layers, 13 convolutional layers, five Max Pooling Layers.
- VGG16 network basically receives 224X224 image input with 3 RGB channels and the image input size is always kept constant by cropping a 224X224 section from the center of each image.
- Conv-1 Layer has 64 filters, Conv-2 has 128 filters, Conv-3 has 256 filters, and Conv-4 and Conv-5 have 512 filters.
- Three Fully-Connected (FC) layers follow a stack of convolutional layers:
  - First and second have 4096 channels each
  - Third has 1000 channels (one for each class).
  - The final layer is the soft-max layer.
- We have used the BreastMNIST dataset for the VGG16 model and since this is an existing/pre-trained model where VGG16 model requires a minimum input size of 32X32 whereas the data size available is 28X28, we have developed a logic to convert the .npz into .jpg so that the data could be augmented easily.
- The augmentation and normalization were performed for the Train, Test, and Validation datasets with parameters as rescaling by 255, featurewise_center and featurewise_std_normalization set as True, rotation_range as 20, width_shift_range and height_shift_range as 0.2.
- After a series of observation of performance change w.r.t few parameters the model achieved an accuracy of 71% with 450 epochs as shown in Fig.2.

Classification Report

	precision	recall	f1-score	support
0	0.42	0.24	0.30	42
1	0.76	0.88	0.81	114
accuracy			0.71	156
macro avg	0.59	0.56	0.56	156
weighted avg	0.67	0.71	0.68	156

Figure 2 Classification report of Breast MNIST dataset

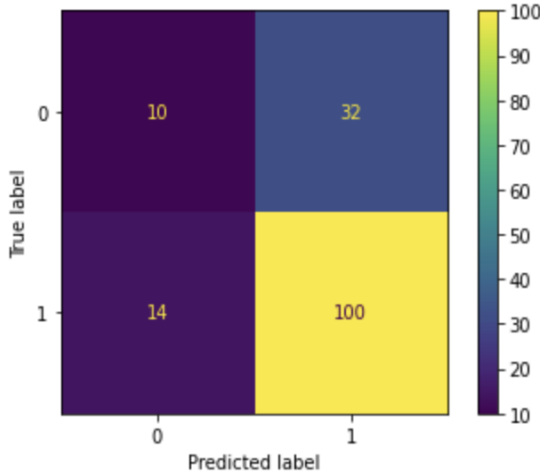

Figure 1 Confusion Matrix of VGG16 model for Breast MNIST dataset

ResNet50 for BloodMNIST

- ResNet50(Kaggle.com, no date) was implemented as an existing deep learning model with input data as BloodMNIST as the dataset consisted of large number of samples i.e., 17,092.
- The model is built of 48 convolution layers, one MaxPool layer and Average Pool layer. Altogether, the architecture is a 50-layer Convolutional Neural Network.
- This type of model stacks up the residual blocks and forms a network.
- Due to the fact that ResNet50 is a pre-trained model with few requirements for input, it is not able to handle .npz input data. Hence, the data which was in .npz format was converted to .jpg to perform data augmentation using Python mentioned in .ipynb file.
- A few parameters were used to enhance the data, such as rescaling the images by 255, zooming to 2, rotating 90 degrees and flipping horizontally and vertically. This was performed using ImageDataGenerator keras library.
- The target size of the image was set to 100X100 during augmentation.
- Since the BloodMNIST dataset consisted of 8 classes the loss was set to categorical_crossentropy and used Adam optimizer with learning rate as 0.001 during compilation.
- After carefully analyzing the change in performance with different input image size and decreasing learning rate, the model was run for 100 epochs and achieved an overall accuracy of 15% which is observed in Fig.3.

	precision	recall	f1-score	support
0	0.00	0.00	0.00	244
1	0.16	0.05	0.08	624
2	0.00	0.00	0.00	311
3	0.18	0.50	0.27	579
4	0.06	0.17	0.09	243
5	0.00	0.00	0.00	284
6	0.18	0.18	0.18	666
7	0.11	0.07	0.09	470
accuracy			0.15	3421
macro avg	0.09	0.12	0.09	3421
weighted avg	0.11	0.15	0.11	3421

Figure 3 classification report of ResNet50 for Breast MNIST dataset

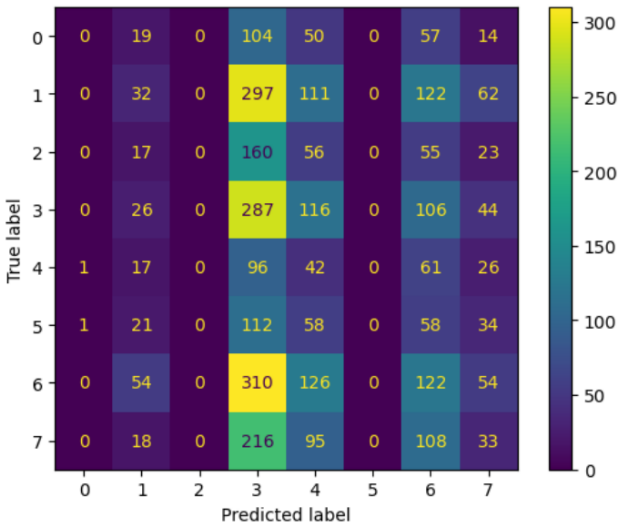

Figure 4 Confusion Matrix of ResNet50 model for BloodMNIST dataset

ANALYSIS OF DESIGNED DEEP NEURAL NETWORKS

A total of 4 novel Deep Neural Networks (DNN) were designed for both BloodMNIST and BreastMNIST dataset.

Novel DNN for BloodMNIST data with 95% accuracy

- The neural network built from scratch has produced the best result out of all 6 models.
- The model has produced 95% accuracy on unseen samples. The model architecture was built with 3 convolutional layers with batch normalization after layer. Then followed by a dense layer with 786 neurons, which is derived from the image size (28X28 = 786). Then these outputs are reduced to the next nearest radix 2 number which is 512. Then the output is classified by 8 neurons with softmax as the activation function.
- All the layers except the output layer are activated by the sigmoid function.
- The data augmentation was implemented with parameters as rescaling by 255, featurewise_center and featurewise_std_normalization set as True, rotation_range as 20, width_shift_range and height_shift_range as 0.2.
- After carefully observing the change in performance metrics the model achieved an accuracy of 95% which was the best of all the models implemented.

```
In [151]: 1 from sklearn.metrics import classification_report

In [152]: 1 print(classification_report(true_arr, class_arr))
```

	precision	recall	f1-score	support
0	0.87	0.96	0.91	244
1	1.00	0.99	0.99	624
2	0.99	0.91	0.95	311
3	0.91	0.84	0.87	579
4	0.91	0.97	0.94	243
5	0.85	0.91	0.88	284
6	0.96	0.98	0.97	666
7	1.00	0.99	0.99	470
accuracy			0.95	3421
macro avg	0.94	0.94	0.94	3421
weighted avg	0.95	0.95	0.94	3421

Figure 5 Classification report of Novel DNN BloodMNIST with 95% accuracy

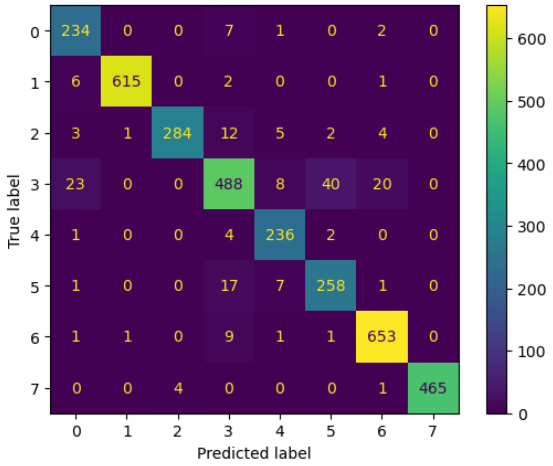

Figure 6 Confusion matrix of Novel DNN BloodMNIST with 95% accuracy

Novel DNN for BloodMNIST with 85% accuracy

- Among the six models, this novel DNN (Coursera.org, no date) placed second in terms of accuracy.
- The Convolutional Neural Network (CNN) architecture consisted of a Sequential layer followed by a Conv2D layer with ReLU as activation function and input image size of 28X28X3 followed by a MaxPool layer with (2,2) as pool size.
- Later, the output from the MaxPool layer was flattened and fed as input to 100 Dense layers with ReLU as activation function.
- The last layer in the network consisted of 8 Dense layers and softmax as activation function.
- CNN cannot accept the image as it is as an input. In order to flatten images, we must convert them into one-dimensional vectors of size 1X(28X28) = 1X784. The reshape function was used to accomplish this. The vectors were normalised to be I between 0 and 1 by dividing each of them with 255 because the pixel values range from 0 to 255.
- As a result of the huge dataset and data imbalance, this network underperformed when data augmentation was implemented. As a result, the dataset was used in .npz format without any data augmentation. After a series of training and evaluation the model achieved an accuracy of 85% with 200 epochs.

----------Classification report----------

	precision	recall	f1-score	support
0	0.92	0.45	0.60	244
1	0.91	0.98	0.94	624
2	0.67	0.93	0.78	311
3	0.74	0.77	0.76	579
4	0.92	0.69	0.79	243
5	0.74	0.77	0.75	284
6	0.96	0.91	0.93	666
7	0.97	1.00	0.99	470
accuracy			0.85	3421
macro avg	0.85	0.81	0.82	3421
weighted avg	0.86	0.85	0.85	3421

Figure 7 Classification report for Novel DNN BloodMNIST 85% accuracy

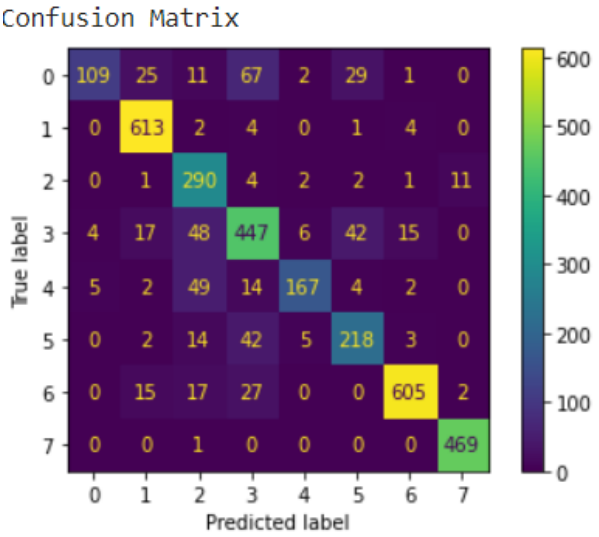

Figure 8 Confusion Matrix for Novel DNN BloodMNIST 85% accuracy

Novel DNN for BreastMNIST with 83% accuracy

- The neural network was trained for 2 different configurations, one without data augmentation and the same neural network with data augmentation.
- The dataset was imbalanced, and to address this situation, we must make the dataset balanced. But this will result in data loss. On the other hand, if we use data augmentation; the imbalance will not be resolved. But to resolve this problem. We will use the data without augmentation and train the neural network, retraining the same model with data augmentation. This will help the neural network address the dataset imbalance and the accuracy significantly increase.
- The neural network architecture consists of 3 convolution layers, with kernel regularizers(Towardsdatascience.com, 2017). And 3 fully connected layers with batch normalization set with a momentum of 0.1.
- Finally, the output layer has 2 neurons, and the softmax activation function.
- The network achieved an accuracy of 83% as shown in Fig. 9.

-------without data_augmentation-------

	precision	recall	f1-score	support
0	0.74	0.55	0.63	42
1	0.85	0.93	0.89	114
accuracy			0.83	156
macro avg	0.79	0.74	0.76	156
weighted avg	0.82	0.83	0.82	156

Figure 9 Classification report of Breast MNIST dataset without Data Augmentation

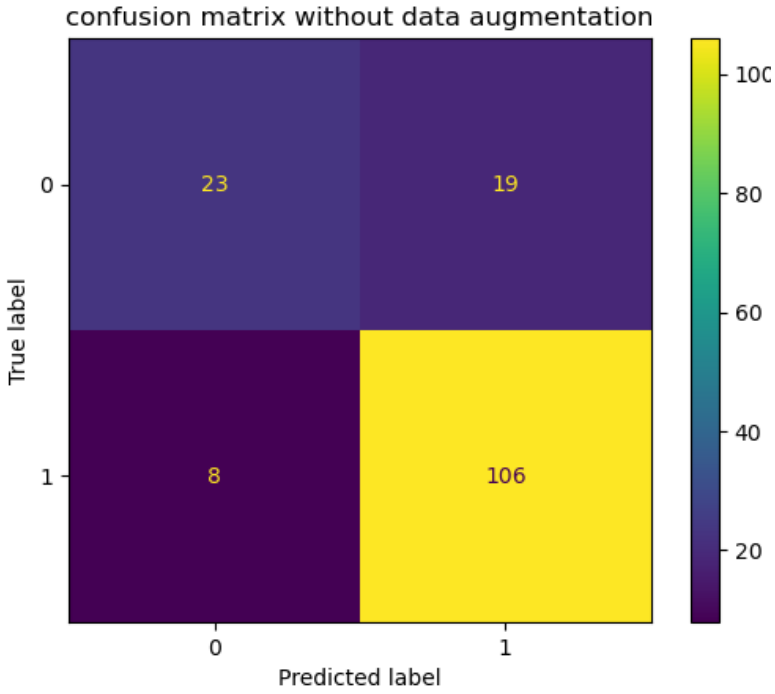

Figure 10 Confusion matrix for Breast MNIST without Data Augmentation

Novel DNN for BreastMNSIT with 77% accuracy

- The neural network is same as used in 3. Was trained with data augmentation to test the varying performance.
- The data augmentation was implemented with parameters as rescaling by 255, featurewise_center and featurewise_std_normalization set as True, rotation_range as 20, width_shift_range and height_shift_range as 0.2(Stackoverflow.com, 2019).
- After applying data augmentation on BreastMNIST the accuracy was reduced to 77% due to imbalance dataset as shown in Fig.11.

-------with data_augmentation-------

	precision	recall	f1-score	support
0	0.88	0.17	0.28	42
1	0.76	0.99	0.86	114
accuracy			0.77	156
macro avg	0.82	0.58	0.57	156
weighted avg	0.79	0.77	0.71	156

Figure 11 Classification report of Breast MNIST dataset with Data Augmentation

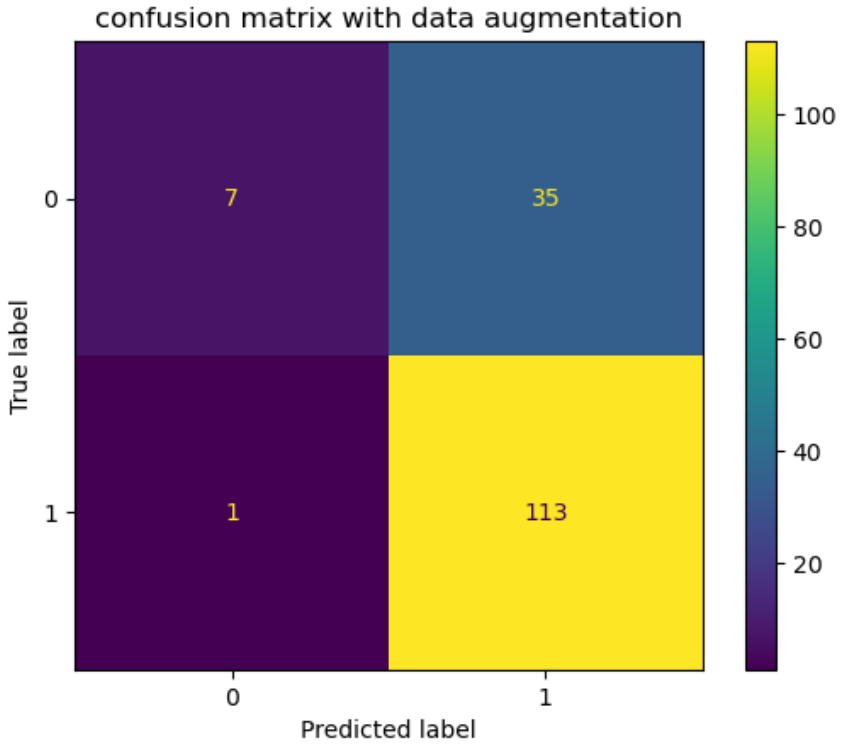

Figure 12 Confusion matrix of breast MNIST dataset with Data Augmentation

ANALYSIS OF THE TRAINING PROCESS

The analysis of all the model's training process is depicted below.

ResNet50 for BloodMNIST

- The minimum input size for ResNet50 model is 32X32, the model was trained with the target_size of the same with 200 number of epochs. This resulted in a very slow training process as there are 50 layers in ResNet50 model and the pixel size was too small.
- Later, to observe the change in performance the input image size was varied with 50X50, 100X100, 150X150 and 224X224 with varying number of epochs and learning rates. At last, it was observed that the image size with 100X100 increased the performance of the model without any issues of overfitting and underfitting of the data and with slight increase in training time.

VGG16 for BreastMNIST

- While training the VGG16 model with the BreastMNIST datasets we set the number of epochs to 250 and after model evaluation for the test dataset we found that the accuracy obtained was approximately 63%. Then after increasing the number of epochs to 450 and the accuracy was improved which was approximately 71% without any overfitting of data respectively.

Novel DNN for BloodMNIST data with 95% accuracy

- The augmented data was trained initially for 200 epochs, for every 5 epochs approximately we have observed a steep change in the validation loss. This meant that for every 5 epochs, the augmented data changed, and the augmentation is working properly to put the neural network into a proper learning process. So, after 200 epochs, the test accuracy or the accuracy on the unseen samples was about 87%.
- Then we ran for another 9 epochs, after this retraining of the network, the network accuracy was about 94.5%

Novel DNN for BloodMNIST with 85% accuracy

- The data was initially trained with 50 epochs and was observed that for every 3 epochs the validation_loss was decreasing steeply. This indicated to train the network with a greater number of epochs to increase the accuracy.
- The validation loss parameter change was not as expected when the model was run with 300 epochs, but it performed well with 200 epochs, i.e., the validation loss decreased significantly and there was no overfitting or underfitting problems.
- Hence, the model was trained for 200 epochs and achieved an accuracy of 85% on unseen samples.

Novel DNN for BreastMNIST with 83% accuracy and Novel DNN for BreastMNSIT with 77% accuracy

- From both neural networks, one thing is clear when the data is less, data augmentation is the best procedure to get more samples of different classes. We did a trial run of saving these images for BloodMNIST. The total number of augmented images crossed 150,000. Which is a good amount of data to train a neural network from scratch.

- One thing we have understood from this experiment is that neural networks behave like a human brain, and they are more realistic to our understanding process and produce results just like us.
- If Data augmentation does not work on these neural networks that are built from scratch, we can opt to use pre-trained models to get more efficiency, because they have already learnt using big datasets like ImageNet dataset, this is called transfer learning, projecting those weights to optimise to our problem, this will help us in training time.
- Since we have also used pre-trained models, we have to satisfy the requirements of these neural networks, for example, VGG16 requires a minimum input of 32x32 whereas the data is available in 28x28, we have converted the arrays into images and resized them to the requirement of the pre-trained models.
- This is the advantage of having our neural networks, we do not have to do much pre-processing. Just normalization and regularization will be sufficient.

ANALYSIS OF THE DESIGNED INTERACTIVE APPLICATION AND THE RELATED GUI

- Using the Streamlit application we have allowed the user to upload an image and select one of six pre-trained models to predict the image label.
- The models are loaded from the saved .h5 model files, and the user can select the appropriate model for the uploaded image from a dropdown menu.
- We have added a custom CSS styling to improve the visual appeal of the interface.
- Once an image is uploaded and a model is selected, the application resizes the image to the appropriate dimensions for the selected model and pre-processes the image data.
- When the user clicks the 'Predict' button, the application uses the selected model to make a prediction on the pre-processed image data and displays the predicted label to the user.
- There are six models available out of which three are for analyzing blood images and three for analyzing the breast images. The user can select any of the six models from the drop-down menu.
- The models listed below are for Blood image analysis:
    - ResNet50
    - Novel DNN model with 95% accuracy
    - Novel DNN model with 85% accuracy
- The models listed below are for Breast image analysis:
    - VGG16
    - Novel DNN model 77% accuracy
    - Novel DNN model 83% accuracy
- Here is a list of all the libraries used in the code:
    - **Streamlit**: A python library for building interactive web applications.
    - **tensorflow**: An open-source machine learning framework developed by Google.
    - **numpy**: A python library for numerical computing.
    - **PIL**: A python image library that adds support for opening, manipulating, and saving many different image file formats.

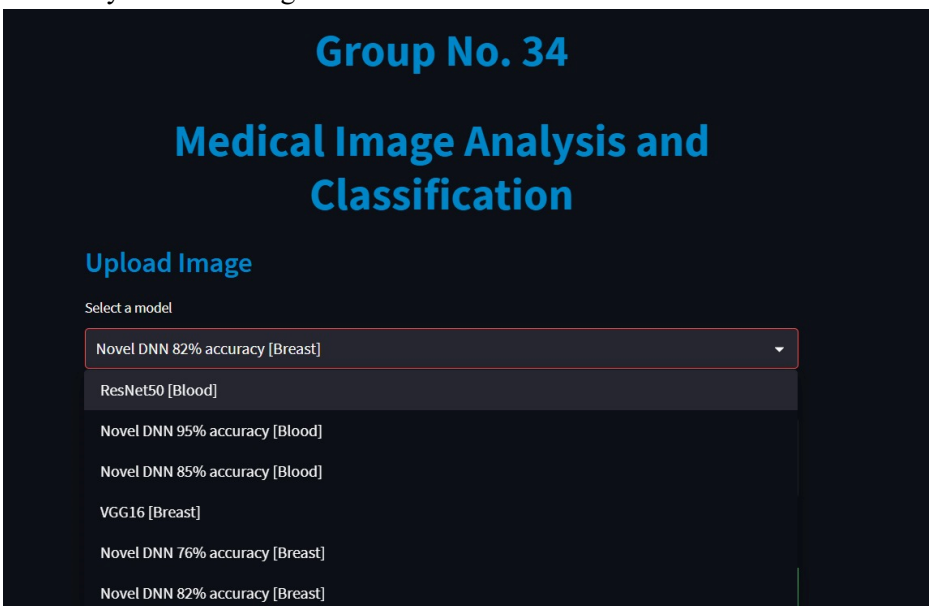

Figure 13 DropDown menu to select a desired model out of 6 models

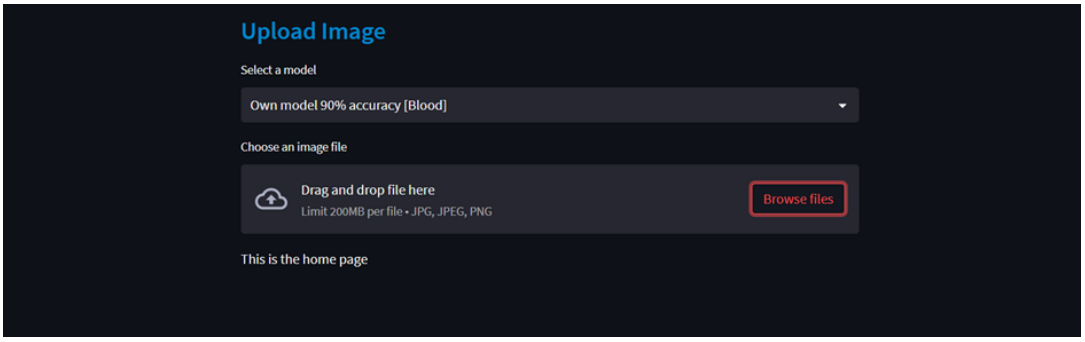

Figure 14 Drag and Drop or Upload image from local system

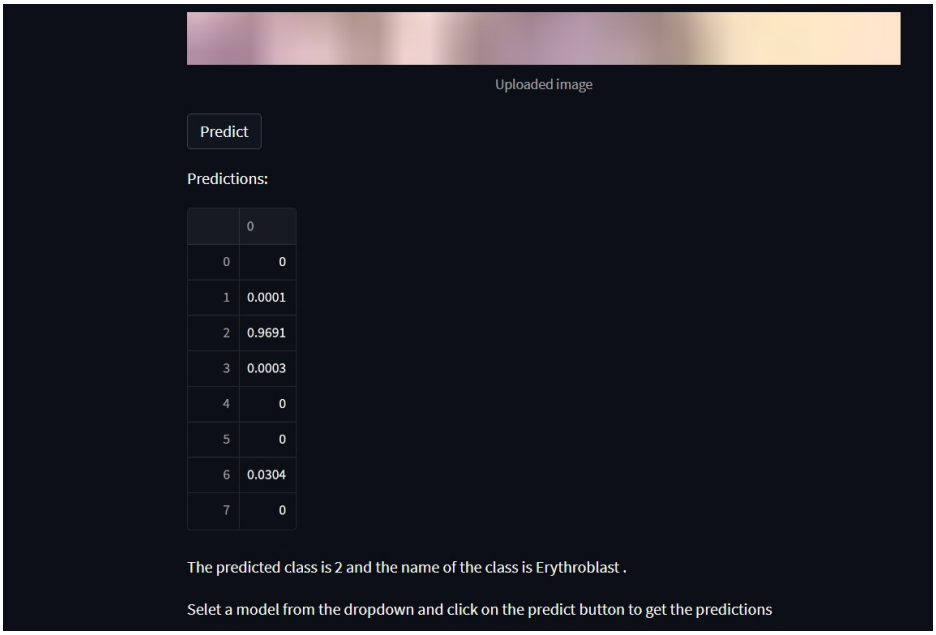

Figure 15 Prediction of class after the image is uploaded

COMPARATIVE ANALYSIS AND PERFROMANCE EVALUATION OF ALL THE DNNs USED

- A total of 6 DNNs were implemented and deployed which are displayed in the tables Tabel.1 and Table.2 along with their performance metrics.

Existing Model's Performance Metrics

SI NO.	MODEL NAME	ACCURACY (%)	PRECISION	RECALL	F1-SCORE
1.	VGG16	71	0.67	0.71	0.68
2.	ResNet50	15	0.11	0.15	0.11

Table 1 Performance metrics of pretrained models

Own Novel DNN Model's Performance Metrics

SI NO.	MODEL NAME	ACCURACY (%)	PRECISION	RECALL	F1-SCORE
1.	Novel DNN for BloodMNIST with 95% accuracy	95	0.95	0.95	0.94
2.	Novel DNN for BloodMNIST with 85% accuracy	85	0.86	0.85	0.85
3.	Novel DNN for BreastMNIST with 83% accuracy	83	0.82	0.83	0.82
4.	Novel DNN for BreastMNIST with 77% accuracy	77	0.79	0.77	0.71

Table 2 Performance Metrics of Own Novel DNN Models

- The above tables Table.1 and Table.2 unmistakably show that the Own Novel DNNs outperformed the state-of-the-art models.
- The primary cause of this is that, while networks of custom models created from scratch can be fine-tuned in accordance with the given data, pre-trained models cannot.
- The prediction accuracy of the own models is high as compared to the pre-trained models.
- One of the own novel DNN achieved 95% accuracy which is the highest of all the models implemented and the lowest achieved accuracy is 15% which is of ResNet50.

REFERENCES

[1] Garg, G. (2022) 'Image Classification Using Resnet-50 Deep Learning Model', *Analytics Vidhya*, 20 September. Available at: https://www.analyticsvidhya.com/blog/2022/09/image-classification-in-stl-10-dataset-using-resnet-50-deep-learning-model/ (Accessed: 8 Mar 2023).

[2] Builtin.com (no date) *Beginner's Guide to VGG16 Implementation in Keras*. Available at https://builtin.com/machine-learning/vgg16 (Accessed on: 8 Mar 2023)

[3] Coursera.org (no date) *Introduction to Deep Learning & Neural Networks with Kears*. Available at: https://www.coursera.org/learn/introduction-to-deep-learning-with-keras (Accessed on: 28 Feb 2023)

[4] Datagen.tech (no date) *ResNet-50: The Basics and a Quick Tutorial*. Available at: https://datagen.tech/guides/computer-vision/resnet-50/ (Accessed on 5 Mar 2023)

[5] Datagen.tech (no date) *Understanding VGG16: Concepts, Architecture, and Performance*. Available at: https://datagen.tech/guides/computer-vision/vgg16/#:~:text=The%20VGG16%20model%20can%20achieve,smaller%203%C3%973%20filters (Accessed on: 27 Feb 2023)

[6] Github.com (no date) *Image-processing-Using-VGG16*. Available at: https://github.com/afsanamimii/Image-processing-Using-VGG16 (Accessed on: 6 Mar 2023)

[7] Kaggle.com (no date) *Car Image Classification Using ResNet50*. Available at: https://www.kaggle.com/code/dskagglemt/car-image-classification-using-resnet50 (Accessed **on**: 9 March 2023).

[8] Keras.io (no date) *Keras Applications*. Available at: https://keras.io/api/applications/ (Accessed on: 2 Mar 2023)

[9] Keras.io (no date) *Layer weight regularizers*. Available at: https://keras.io/api/layers/regularizers/ (Accessed on 8 Mar 2023)

[10] Medium.com (no date) *Everything you need to know about VGG16*. Available at: https://medium.com/@mygreatlearning/everything-you-need-to-know-about-vgg16-7315defb5918 (Accessed on: 28 Feb 2023)

[11] TensorFlow.org (no date) *tf.keras.layers.Conv2D*. Available at: https://www.tensorflow.org/api_docs/python/tf/keras/layers/Conv2D (Accessed on: 29 Feb 2023)

[12] TensorFlow.org (no date) *tf.keras.layers.MaxPool2D*. Available at: https://www.tensorflow.org/api_docs/python/tf/keras/layers/MaxPool2D (Accessed on: 6 March 2023)

[13] Towardsdatascience.com (2017) *Regularization in Machine Learning*. Available at: https://stackoverflow.com/questions/54302953/keras-poor-performance-with-imagedatagenerator (Accessed on: 4 March 2023)

[14] Stackoverflow.com (2019) *Keras: poor performance with ImageDataGenerator*. Available at: https://stackoverflow.com/questions/54302953/keras-poor-performance-with-imagedatagenerator (Accessed on: 2 March 2023)

[15] Y. Yari, H. Nguyen and T. V. Nguyen, "Accuracy Improvement in Binary and Multi-Class Classification of Breast Histopathology Images," *2020 IEEE Eighth International Conference on Communications and Electronics (ICCE)*, Phu Quoc Island, Vietnam, 2021, pp. 376-381, doi: 10.1109/ICCE48956.2021.9352142.