CHEST OPACITY CLASSIFICATION

By

Sulaiman Kagumire

2100702296
2021/HD05/2296U

Outline

- Data balancing
- Train-valid-test-split
- Image Preprocessing
- Data Augmentation
- Model Architecture
- Results
- Model Evaluation
- Model Deployment
- Model Testing
- Work in Progress

Balancing the Data

- First step was to balance the data so that all classes have the same number of training data images.
- Though this was not that necessary because the data was slightly balanced, but however I did.

Data Balancing

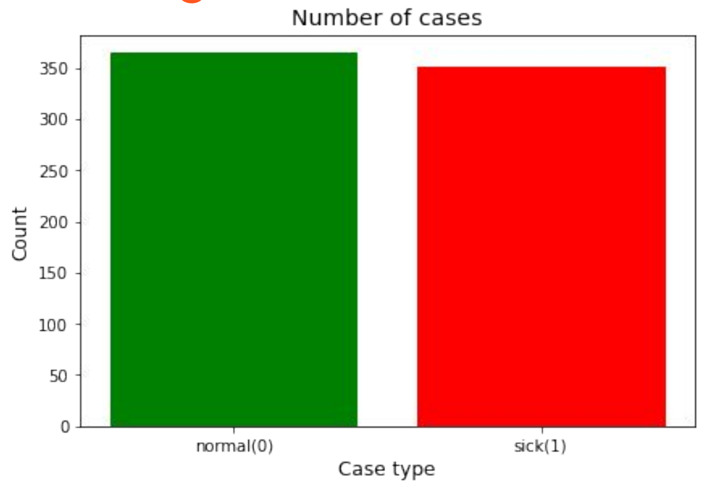
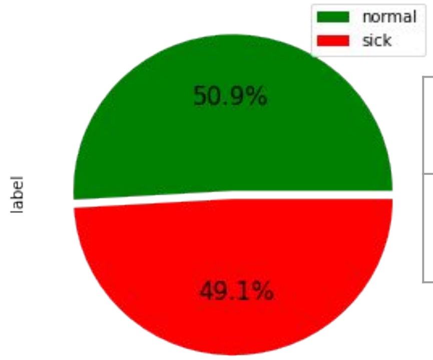


Fig.1 Imbalance Data



normal	364
sick	351

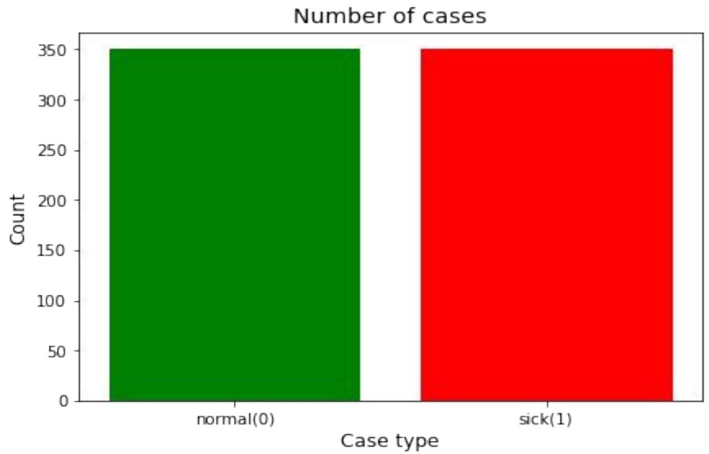
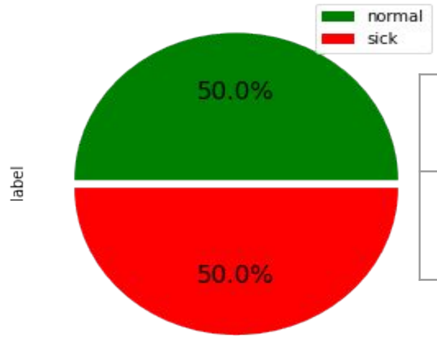


Fig.2 Balanced Data



normal	350
sick	350

Train-test-Split

- After Balancing, the data was split into training, validation, and testing, as shown in the next page slide.
- Emphasis was put on having the same number of classes in all the three data sets.

Train-valid-test-split

	normal	sick	Total
train	224	224	448
valid	70	70	140
test	56	56	112

Image Preprocessing

- After splitting, The data was read through a pipeline that preprocesses the images.
- Images were resized to 150X150. This was a random choice of size. Another reason would be to reduce the number of training parameters.
- Labels were converted to a categorical matrix which has binary values for each label column.

Data Augmentation

- Augmentation means to artificially increase the amount of data by generating new data points from existing data by adding minor alterations to data.

- The augmented images were normalized by rescaling pixel values from the range of 0-255 to the range 0-1 preferred for neural network models.

Training and Validation

- A decision to use a pretrained model was taken based on the fact that the data was too small. Pretrained models are trained on large datasets, like ImageNet.

- The hard work of optimizing the parameters has already been done for you, now what you have to do is fine-tune the model by playing with the hyperparameters so in that sense, a pre-trained model is a life-saver.

Training and Validation

- Model used - ResNet50V2
- Trainable layers were frozen so that their state won't be updated during training.
- One Dense layer, activated with Relu, and output layer, activated with softmax, were added on top of the model.

The following Callbacks were added at given stages of the training procedure.

1. ReduceOnPlateau - **Reduce learning rate when a metric (loss in this case) has stopped improving**. Models often benefit from reducing the learning rate
2. ModelCheckpoint - To save a best model or weights (in a checkpoint file) at some interval, so the model or weights can be loaded later to continue the training from the state saved.

Hyperparameters

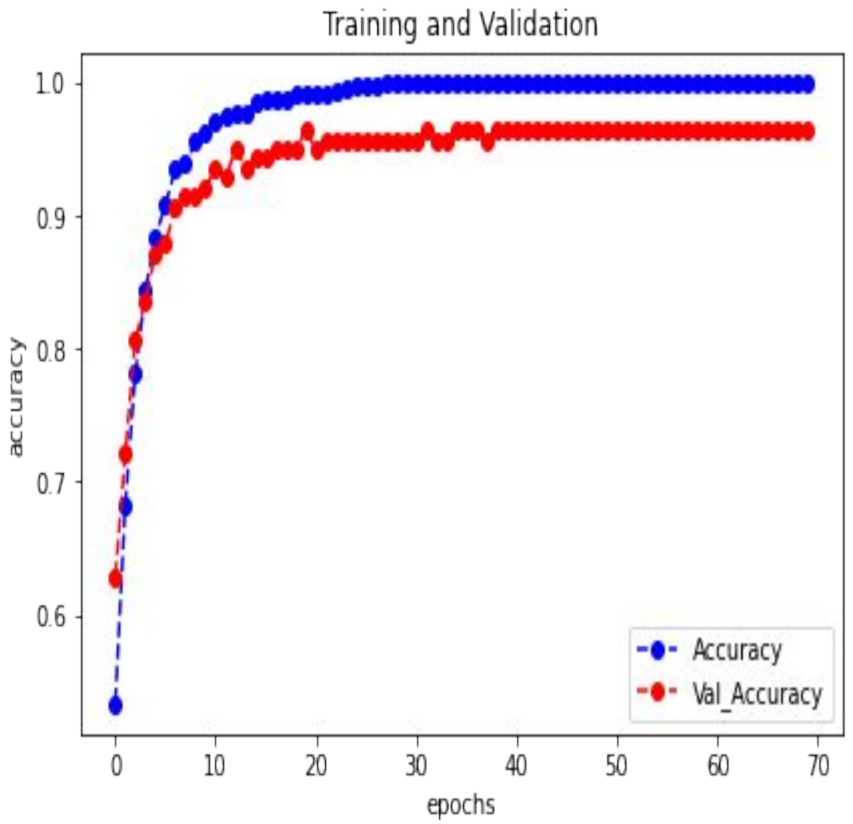
Learning rate = 0.0001

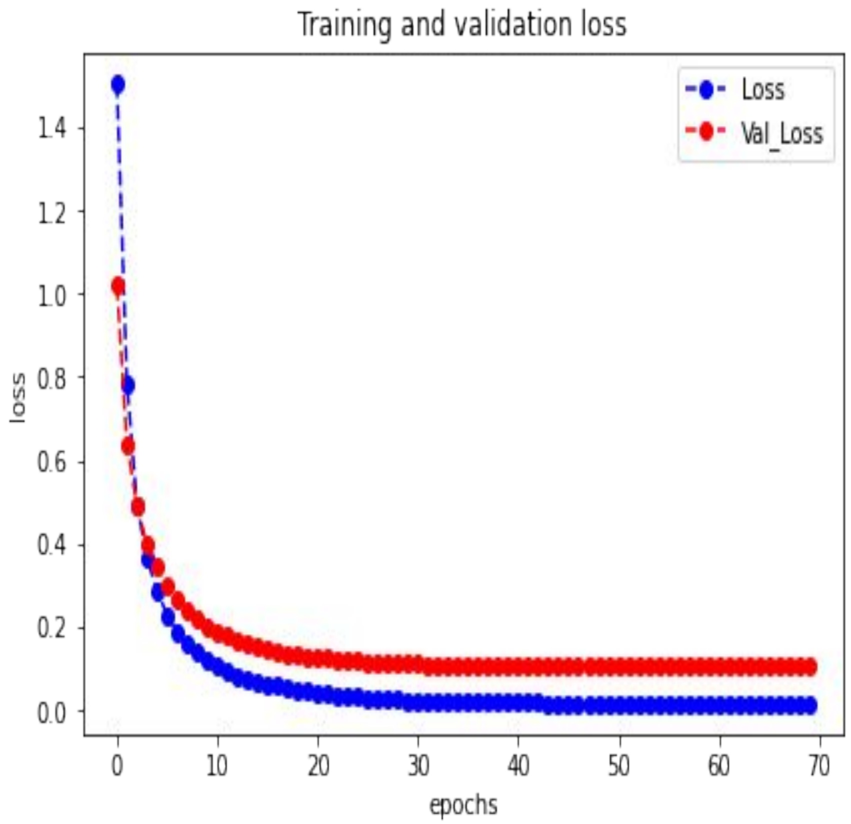
Optimizer - Adam

Loss metric = Categorical cross entropy

Performance metrics = Accuracy and AUC

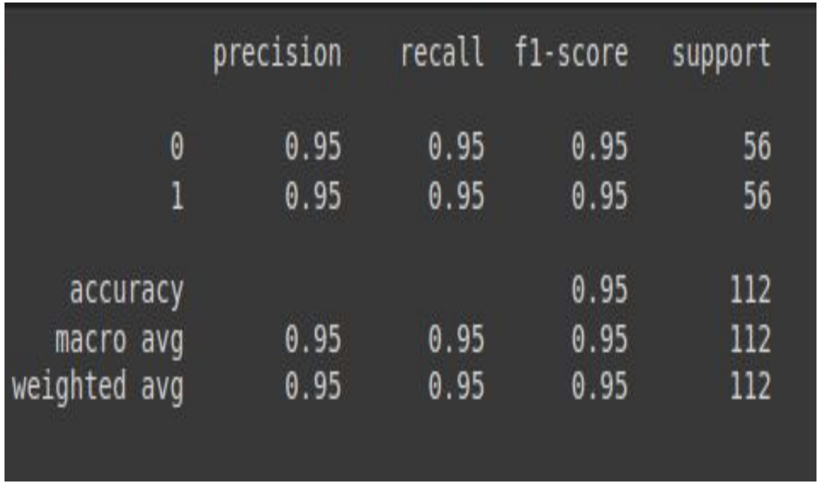
Results


Training and Validation
accuracy
epochs
Accuracy
Val_Accuracy
1.0
0.9
0.8
0.7
0.6
0
10
20
30
40
50
60
70


Training and validation loss
loss
epochs
Loss
Val_Loss
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
30
40
50
60
70

Model evaluation

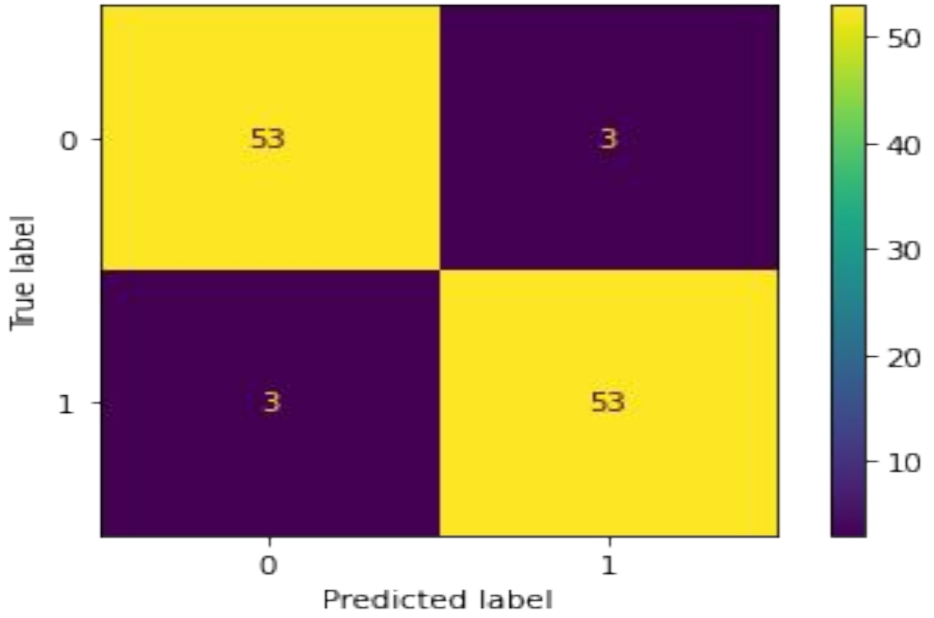
Evaluating the model on the test data



```
              precision    recall  f1-score   support

           0       0.95      0.95      0.95        56
           1       0.95      0.95      0.95        56

    accuracy                           0.95       112
   macro avg       0.95      0.95      0.95       112
weighted avg       0.95      0.95      0.95       112
```



Model Deployment

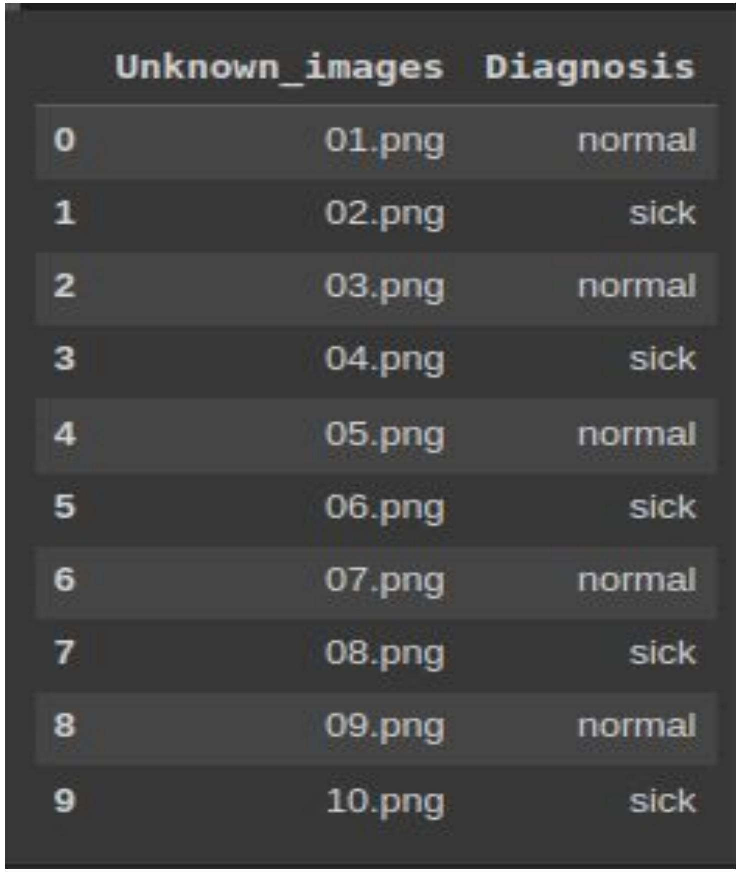
The model was deployed using Streamlit at https://ksulaiman1-mcs7204-deep-learning-project-deploy-6uzk77.streamlitapp.com/

- Upload an image and click and the backend model will predict or determine whether the image has opacities or it is a normal image.

Testing - Unknown images

Using the given unknown images, the predicted labels for the first 10 image are shown below.

The unknown images were passed through the same preprocessing as the training images.



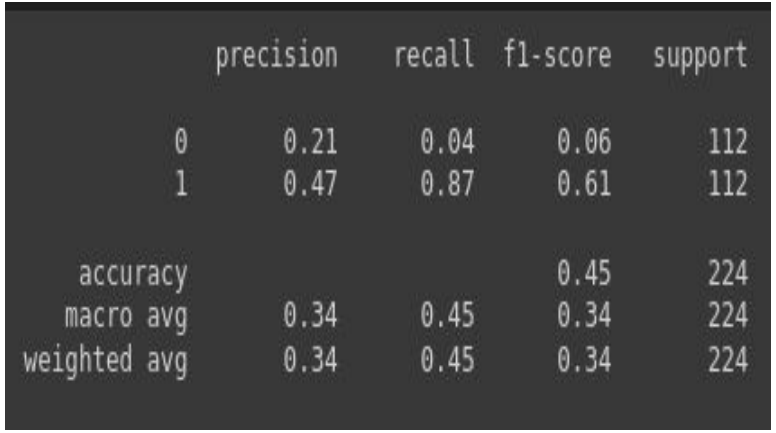
	Unknown_images	Diagnosis
0	01.png	normal
1	02.png	sick
2	03.png	normal
3	04.png	sick
4	05.png	normal
5	06.png	sick
6	07.png	normal
7	08.png	sick
8	09.png	normal
9	10.png	sick

Testing on Dataset2

- SImilarly, Dataset2 was passed through the same preprocessing steps as the training images.

- The model results on Dataset2 is shown in the confusion matrix in the next slide page.

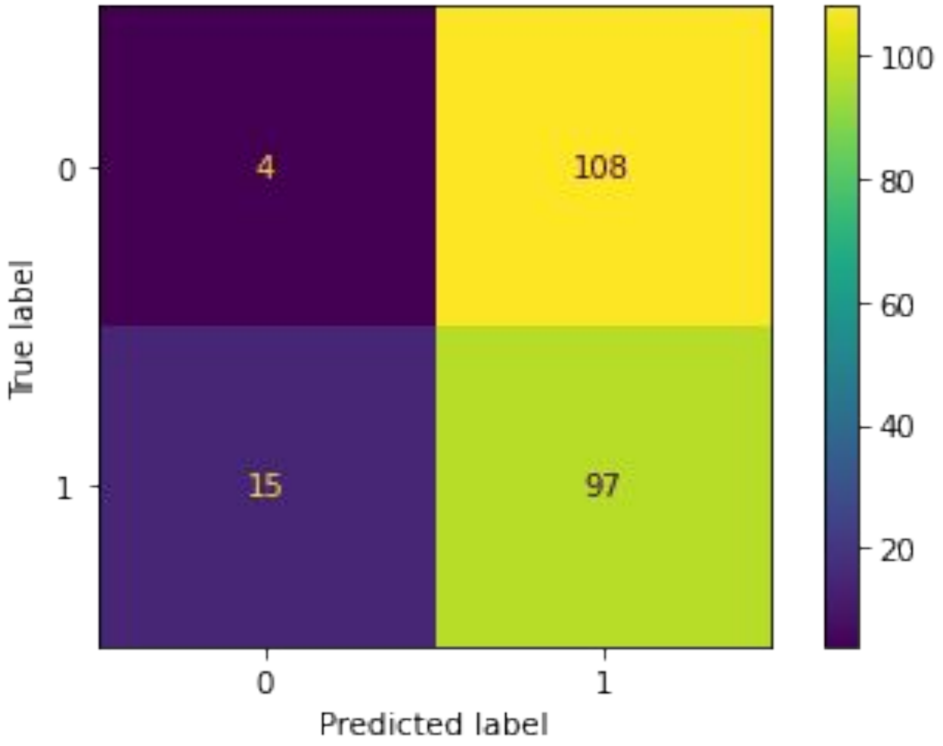
Testing - Dataset2



```
              precision    recall  f1-score   support

           0       0.21      0.04      0.06       112
           1       0.47      0.87      0.61       112

    accuracy                           0.45       224
   macro avg       0.34      0.45      0.34       224
weighted avg       0.34      0.45      0.34       224
```



Results Discussion

- The model performed quite well on the test data. But however it performed poorly on Dataset2. One reasons for this might be because Dataset2 was from a different environment.
- To improve this model, Vision Transformer models might be used. But however, these models need a lot of training data points.