Melanoma Detection Assignment

Team Members: Aishwarya Avinash, Nikhil Panchagnula and Shan Salance

Problem Statement:

We will build a multiclass classification model using a custom convolutional neural network in TensorFlow

The problem

Melanoma is a type of cancer that can be deadly if not detected early. It accounts for 75% of skin cancer deaths. A solution that can evaluate images and alert dermatologists about the presence of melanoma has the potential to reduce a lot of manual effort needed in diagnosis. Hence build a CNN model which can detect melanoma.

The dataset

The dataset consists of 2357 images of malignant and benign oncological diseases, which were formed from the International Skin Imaging Collaboration (ISIC). The list of class labels i.e. lst of diseases are as follows:

1. Actinic keratosis
2. Basal cell carcinoma
3. Dermatofibroma
4. Melanoma
5. Nevus
6. Pigmented benign keratosis
7. Seborrheic keratosis
8. Squamous cell carcinoma
9. Vascular lesion

So in total there are 9 labels to be detected in the assignment

Assumptions

While solving the guidelines of the problem statement are followed:

- Keeping directory batch size 32
- Resizing image 180*180
- Number of epochs is 20
- Number of epochs in the augmented data = 20
- Number of epochs in rectified imbalanced data is 30

Link to my google drive where I kept and used my **zip dataset**
https://drive.google.com/drive/folders/13d9icXqHbgyGDKn2wtYg-XiBhdfnzf2B?usp=drive_link

For processing I have created a CNN_Assignment folder in my google drive in which I have kept this CNN_assignment.zip as provided in the project page.

Approach

- Understanding the dataset
- Importing essential libraries
- Defining the path for train and test images

- Defining parameters for the loader
- Creating respective training and validation datasets
- Visualizing classes
- Creating a base model
- Visualizing model results
- Treating overfitting by augmentation
- Re-building model with dropouts
- Checking presence of class imbalance
- Rectification of class imbalance
- Building final model
- Results
- Conclusion

Understanding the dataset

Training data: consist of 2239 images

Test data: consist of 118 images

The dataset consists of 2357 images of malignant and benign oncological diseases, which were formed from the International Skin Imaging Collaboration (ISIC). These are supposed to be used for training purpose of the model.

Importing essential libraries

Libraries that are essential for the processing for the model are imported which includes:

- Pathlib
- Tensorflow
- Matplotlib
- Numpy
- Pandas
- Os
- PIL

Defining the path for train and test images

Path to train and test images are defined in the code file to make them accessible for processing.

Defining parameters for the loader

As provided :

- Batch size o 32
- Image height of 180
- Image width of 180
- Seed of 123

Is used for loader

Creating respective training and validation datasets

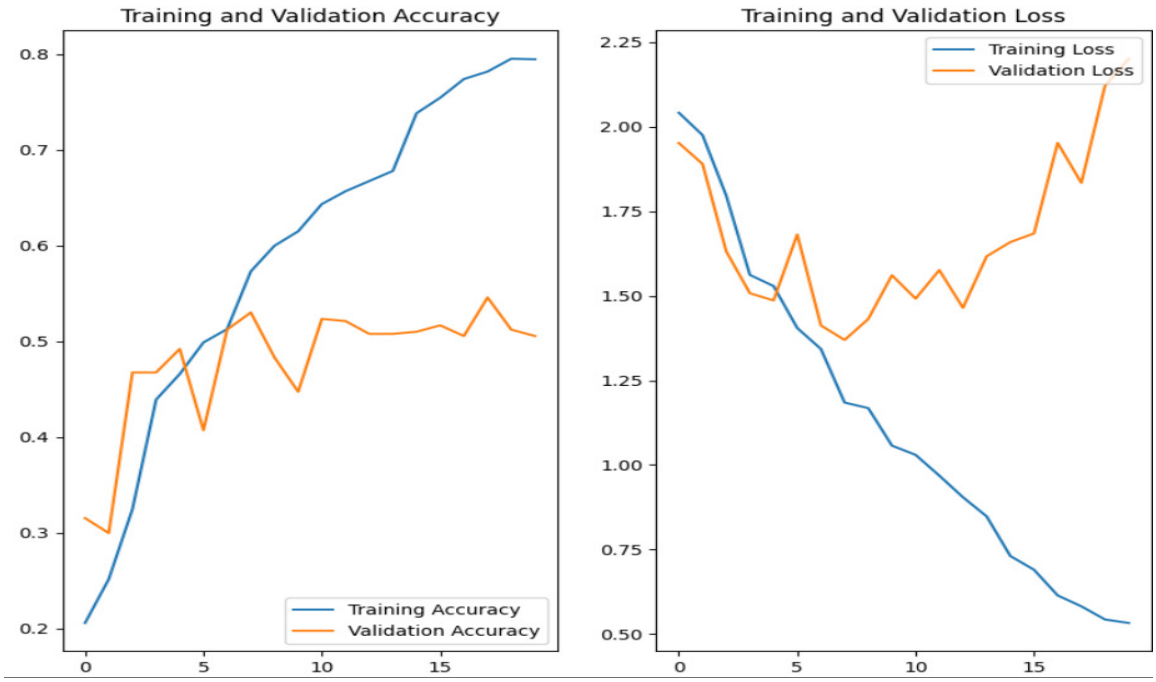
- For the training subset 80% of the records are kept
- For validation subset 20% of the records are kept

Creating a base model

A base model is created for processing in which:

- 3 convolution layers are used
- Max pooling
- Flatten
- And Dense output is used
- With **relu** activation function as it has better performance for non linear problem

Observation:

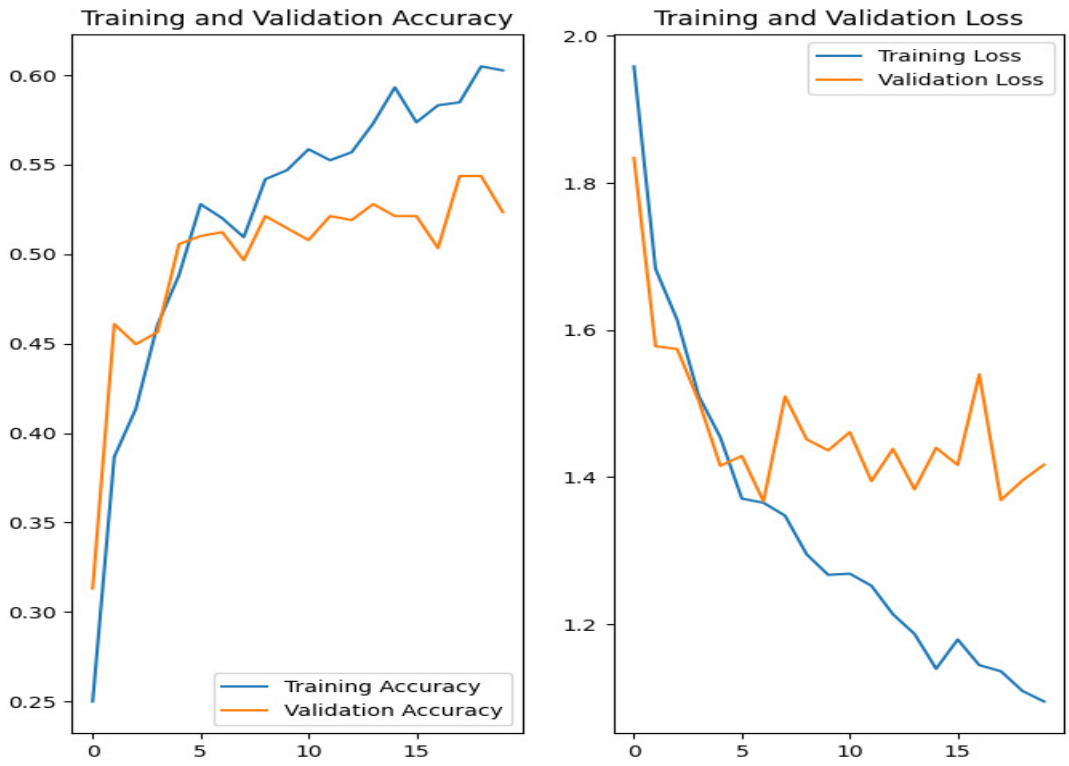


- Training accuracy is increasing but validation stops to follow the same path after few epochs
- Validation loss is also increasing while training loss decreases
- **This shows clear sign of OVERFITTING.**

Treating overfitting by augmentation

To rectify OVERFITTING data is augmented by performing **rotations, zooming** and **flip.** This provided model to learn from the same images but created different instances of it. After this the model was re-built.

Observations:



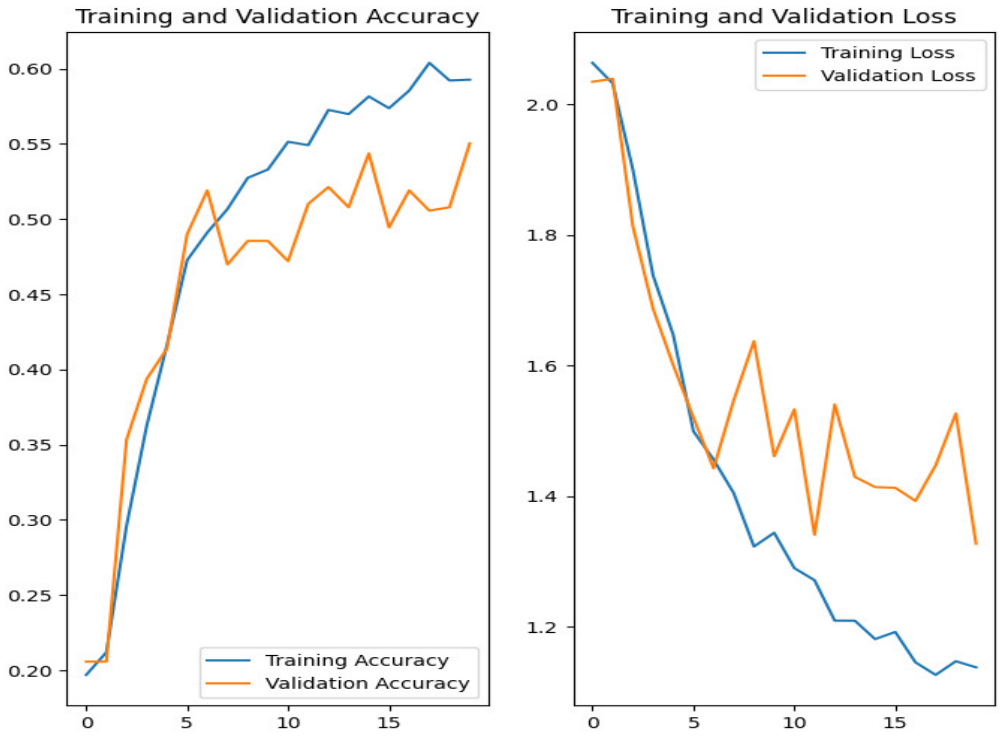
- We can observe that after applying augmentation the accuracy of validation is following that of train. Which shows that we have **improved** model in terms of **overfitting.**
- **But,** the model performance dropped a lot as compared to previous result

for further improvement adding a dropout layer to the same model.

Re-building model with dropouts

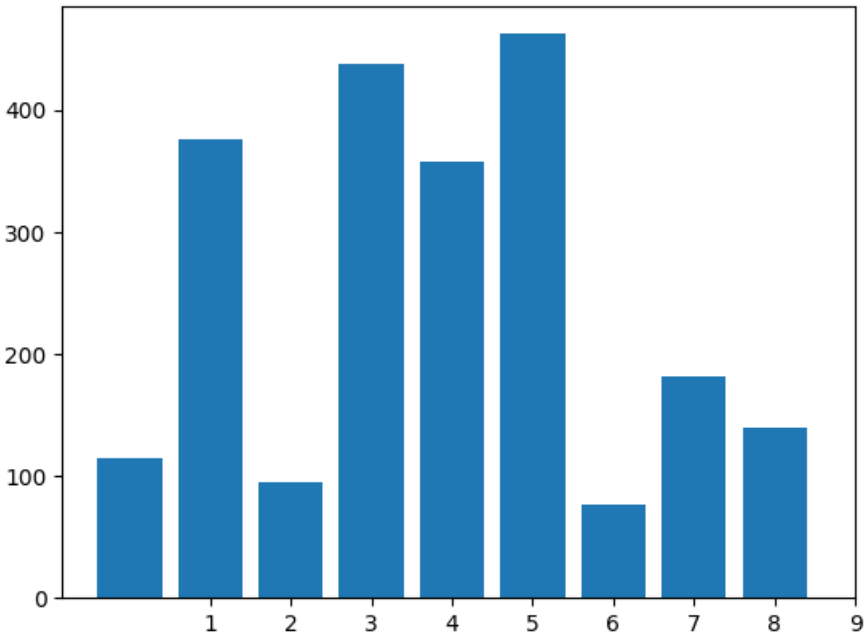
There were no remarkable changes in accuracies with dropouts **slight improvement in losses**, but this will help to get rid of any redundancies and will lower the workload on the model overall.

I have used the very least value for the dropout so that not too much of information loss is there



Checking presence of class imbalance

As the model was not improving after basic augmentation or dropouts, looked for presence of any class imbalance.



As we can see there are classes in majority and in minority, this creates a biased learning and hence the model is not getting equal chance to learn about each class.

Rectification of class imbalance

In such real-world data, there is one more factor to take into account and that is class imbalance. To overcome this again using augmentation and using it adding 500 instances of images for each class. This will give model equal opportunity to learn about each and every label available

```
pigmented benign keratosis    14.275115
melanoma                      13.918979
basal cell carcinoma          12.998961
nevus                         12.717020
squamous cell carcinoma       10.105357
vascular lesion                9.482119
actinic keratosis              9.111144
dermatofibroma                 8.829203
seborrheic keratosis           8.562101
Name: Label, dtype: float64
```

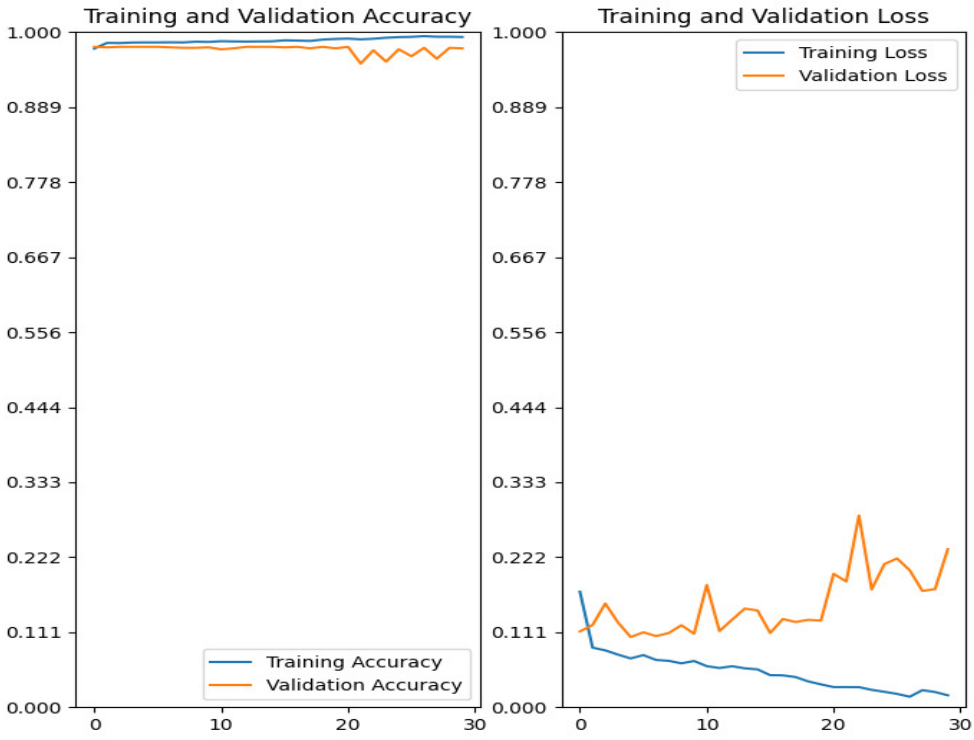
As observed above we can see that the classes now appear to be more evenly distributed.

Building final model

After rectification model is re trained for which Sequential model is used with the following layers:

- Conventional layers followed by MaxPooling
- Using normalization to normalize data for cnn
- Dropouts to avoid any redundancies
- Followed by more conventional layers
- And finally dropout flatten and dense for the output
- Using number of epochs as 30 as mentioned in the portal

Observations:




```
loss: 0.0176 - accuracy: 0.9927 - val_loss: 0.2340 - val_accuracy: 0.9759
```

- As we can see the model is performing very well with both training and validation accuracies are > 97% and close-by
- We can see the losses in train and validation are also <1 and has dropped eventually more as compared with the previous models.

Conclusion

- At first, we started with a basic model which led to highly overfit data
- we tried to resolve it by applying augmentation strategies like rotation, zoom and flip, it improved the variation between the train and validation accuracies but the overall accuracy dropped a lot
- another model with dropouts was introduced along with the augmentation it did help to get rid of redundancies
- till this time the performance was not good but the overfitting was not a detectible problem
- so, to improvise, we looked into the class distribution in the dataset, it was found that there is imbalance in the dataframe.
- to resolve these 500 more instances were added to the existing images to overcome the data imbalance
- because of class rebalance to which model got an equal chance in learning about each class
- and in the final result we can see that the training and validation accuracies have gone 99% and 97% respectively, also the losses have reduced. Concluding the final model