# **Melanoma Detection Assignment**

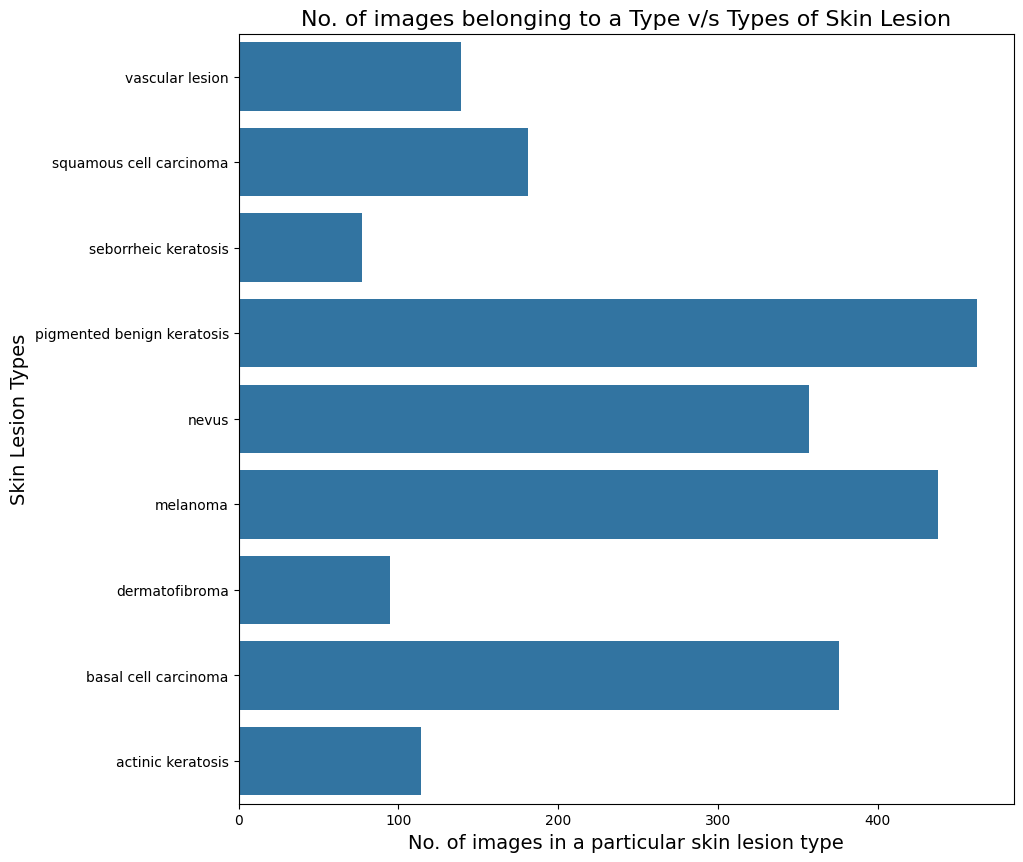
In this assignment, we have to build a **multiclass classification model using a custom convolutional neural network in TensorFlow**.

## **Table of Contents**

* [General Info](#4d34og8)
* [Technologies Used](#2s8eyo1)
* [Conclusions](#17dp8vu)
* [Acknowledgements](#3rdcrjn)

**General Information**

The dataset comprises 2357 images depicting malignant and benign oncological conditions, sourced from the International Skin Imaging Collaboration (ISIC). These images were categorized based on the classification provided by ISIC, with each subset containing an equal number of images.



We used the Augmentor library (<https://augmentor.readthedocs.io/en/master/>) to balance the dataset. This means we created more data for under-represented classes to ensure all classes had a fair chance of being learned by the model.

**Conclusions**

1. **Model architecture**

**Breakdown of the CNN Architecture:**

**1. Data Augmentation:** The augmentation\_data variable specifies techniques to artificially expand the training dataset. This involves random transformations like rotations, scaling, and flipping, enhancing the model's ability to generalize to unseen data.

**2. Normalization:** The Rescaling(1./255) layer normalizes pixel values to a 0-1 range. This stabilizes training and accelerates convergence.

**3. Convolutional Layers:** Three convolutional layers, each followed by a ReLU activation function, extract features from the input images. The padding='same' argument preserves spatial dimensions. The number of filters in each layer (16, 32, 64) determines feature map depth.

**4. Pooling Layers:** Max-pooling layers downsample feature maps, reducing computational cost and mitigating overfitting.

**5. Dropout Layer:** A dropout layer with a 20% dropout rate randomly deactivates neurons during training, preventing overfitting.

**6. Flatten Layer:** The Flatten layer transforms the 2D feature maps into a 1D array, preparing the data for fully connected layers.

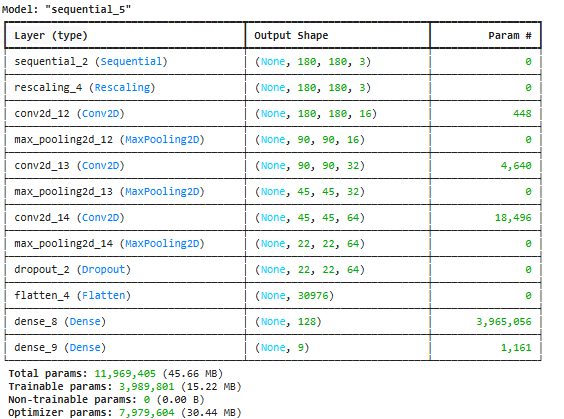
**7. Fully Connected Layers:** Two dense layers with ReLU activation functions learn complex patterns from the flattened features. The final dense layer outputs the probability distribution over class labels.

**8. Output Layer:** The number of neurons in the output layer matches the number of classes. It produces raw logits, which are fed into the loss function during training.

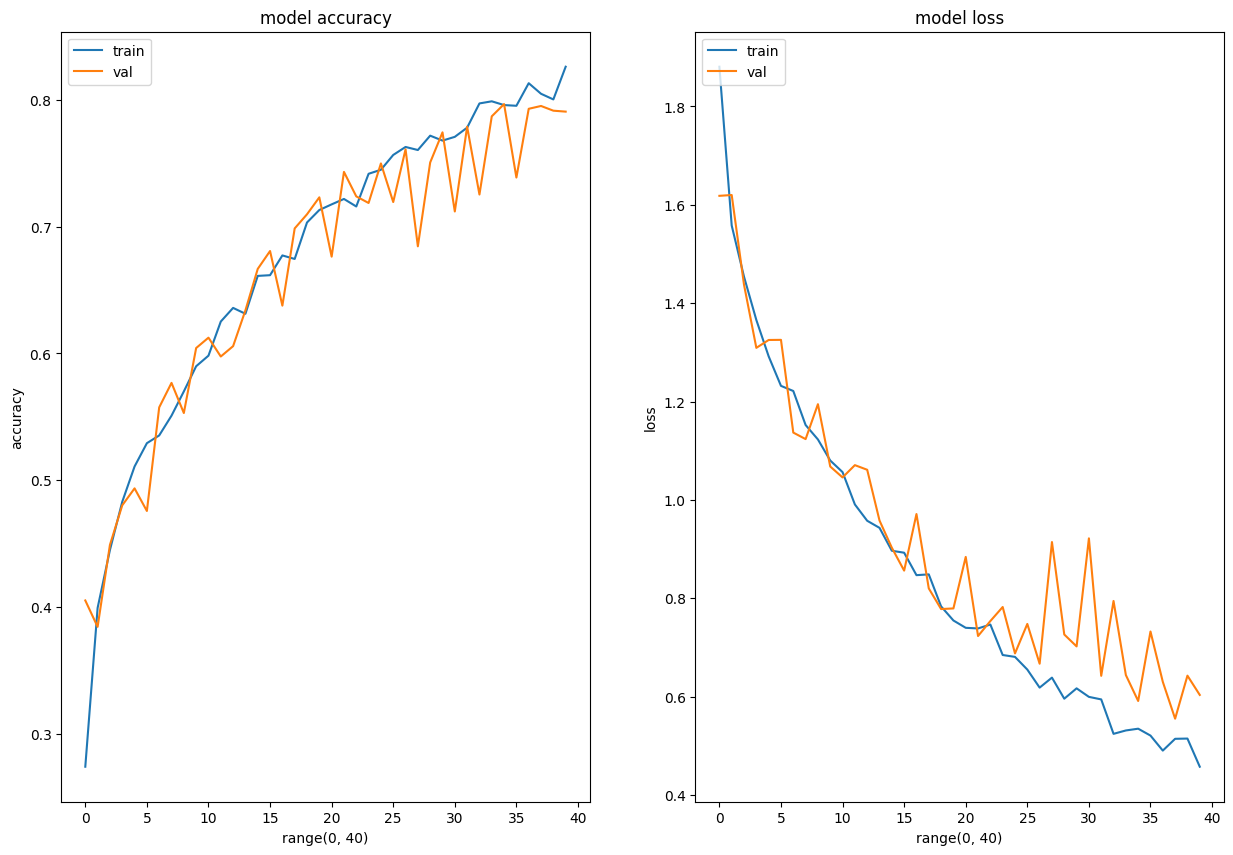
**9. Model Compilation:** The model is compiled using the Adam optimizer and Sparse Categorical Crossentropy loss, suitable for multi-class classification. Accuracy is used as the evaluation metric.

**10. Training:** The model is trained for 50 epochs. The ModelCheckpoint callback saves the best-performing model, and EarlyStopping halts training if validation accuracy doesn't improve for 5 epochs.

1. **Model Summary**

****

1. **Model Evaluation**

****

## **Technologies Used**

* [Python](https://www.python.org/) - version 3.11.4
* [Matplotlib](https://matplotlib.org/) - version 3.7.1
* [Numpy](https://numpy.org/) - version 1.24.3
* [Pandas](https://pandas.pydata.org/) - version 1.5.3
* [Seaborn](https://seaborn.pydata.org/) - version 0.12.2
* [Tensorflow](https://www.tensorflow.org/) - version 2.15.0

## **Acknowledgements**

* UpGrad tutorials on Convolution Neural Networks (CNNs) on the learning platform
* [Melanoma Skin Cancer](https://www.cancer.org/cancer/melanoma-skin-cancer/about/what-is-melanoma.html)
* [Introduction to CNN](https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/)
* [Image classification using CNN](https://www.analyticsvidhya.com/blog/2020/02/learn-image-classification-cnn-convolutional-neural-networks-3-datasets/)
* [Efficient way to build CNN architecture](https://towardsdatascience.com/a-guide-to-an-efficient-way-to-build-neural-network-architectures-part-ii-hyper-parameter-42efca01e5d7)

## **Contact**

Created by [@Suraj-Nair71] - feel free to contact me!