WESTERN UNIVERSITY
FACULTY OF ENGINEERING
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

ECE 9063 PROJECT: SKIN LESION CLASSIFICATION USING CONVOLUTIONAL NEURAL NETWORKS

ECE 9063 Data Analytics Foundation

Group Members:
Ashish Joshi, MEng.
Nittin Sant Sethi, MEng.
Tejasvi Roop Singh, Meng.

CONTENTS

1 INTRODUCTION 3
2 BACKGROUND 3
3 DATASET 4
3.1 EXPLORATORY DATA ANALYSIS (EDA) 7
4 MODEL USED 9
4.1 CONVOLUTIONAL NEURAL NETWORK 9
4.1.1 ARCHITECTURE 9
4.1.2 WHY CNN? 11
4.2 RESNET50 12
4.2.1 COMPARISON WITH CNN 12
4.2.2 ARCHITECTURE 12
5 METHODOLOGY 14
5.1 OVERALL OVERVIEW 14
5.2 DATA PRE-PROCESSING 14
5.2.1 DATA CLEANING 14
5.2.2 REMOVING CLASS IMBALANCE 15
5.2.3 IMAGE SEGMENTATION 15
5.2.4 FEATURE ENGINEERING 16
5.2.5 TRAIN/TEST/VALIDATION SPLIT 17
5.2.6 DATA NORMALIZATION (STANDARDIZATION) 17
5.2.7 ONE-HOT ENCODING 18
5.3 MODEL TRAINING 19
5.3.1 PARAMETERS BEING TUNED 19
5.3.2 UPDATING WEIGHTS AND BIASES 21
5.4 MODEL EVALUATION 22
6 RESULTS 24
6.1 BEFORE TUNING 24
6.2 AFTER TUNING 25
6.3 TEST RESULTS 27
6.4 COMPARISON BETWEEN THE 3 MODELS 29
7 MODIFICATIONS AFTER PRESENTATION - OVERSAMPLING 30
8 CONCLUSION AND FUTURE SCOPE 32
8.1 CONCLUSION 32
8.2 FUTURE SCOPE 33
9 REFERENCES 34

ABSTRACT

There have been numerous advancements in medical technology so far and with our project, we aim to contribute more towards the same by helping those who have shown symptoms while getting diagnosed with skin cancer.
The main purpose of our project is to analyze the dermatoscopic images of pigmented lesions while analysing also used image segmentation to find the cancerous part of the input images and then automatically classify them into one of the seven diagnostic categories using deep learning models Convolutional network and Residual Network-50.
This will make the patient and doctors aware of any issues present so that necessary action can be taken as soon as possible.
With our project, we conclude that classification with unsegmented images performed better than segmented images. The performance was also affected by the class imbalance which was later countered by oversampling technique.

1 INTRODUCTION

Skin cancer is the most prevalent in humans amongst all other cancer types. One of the main reasons for skin cancer is the exposure to harmful Ultra-Violet rays from the sun. Moreover, according to the Government of Canada's website, about one-third of all the new cases of cancer in Canada are skin cancers and the trend is on the rise [1]. Over 80,000 cases of skin cancer are diagnosed in Canada each year, more than 5,000 of which are melanoma, the deadliest form of skin cancer [2]**.** Therefore, it is necessary to detect skin cancer so that it can be treated on time and does not spread or become fatal.
In this project, we are working with cancer cells present in the skin. The main aim of the project is to classify skin lesions from images into 7 different categories of skin lesion namely:

1. Melanocytic nevi
2. Melanoma
3. Benign keratosis-like lesions
4. Basal cell carcinoma
5. Actinic keratoses
6. Vascular lesions
7. Dermatofibroma

This project uses Convolutional Neural Networks to classify skin lesions into the various categories of skin cancer. Firstly, the images are segmented so that only the significant parts of the image are left. It is then passed through the model which classifies the image accordingly. We have actually passed both segmented as well as unsegmented images and compared the performance of the model in both the cases. Furthermore, we have used the segmented images as the input for ResNet50 and compared its performance with that of CNN.

The rest of the report is organized as follows. First, we present Background in Section 2 which gives a little information about the models and accuracy measures used. In Section 3 we explain the dataset and depict the results of exploratory data analysis. Section 4 discusses the model used and the architecture of each model. Section 5 consists of the steps we have carried out from data preprocessing to training and testing and the metrics to evaluate the models. In Section 6 we have discussed the results obtained. Section 7 consists of Modifications made after Presentation which is finally followed by Conclusion and Future Work explained in Section 8.

2 BACKGROUND

Convolutional Neural Networks (CNN) is one of the most important and widely used forms of an artificial neural network which specializes in image recognition and classification tasks. Traditional neural networks or multilayer perceptron are not suitable for image classification tasks especially for large images because of the increasing number of weights with each hidden layer. A CNN performs image classification tasks better as the requirements for processing are lower as compared to an MLP and hence there are a lesser number of weights which a CNN must learn. A CNN uses a filter or a kernel which is mapped over the image to calculate values used in the training process. A CNN also consists of an input layer, convolutional layers, pooling layers and fully connected layers which give the output of the classification problem.

Image segmentation is a technique where the image is divided into different segments so as only important parts of an image can be utilized.

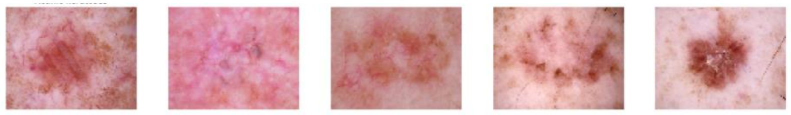
RESNET50 is an extension of Convolutional Neural Network (CNN) and has performed exceptionally in many image classifications tasks.

Confusion Matrix measure the performance helps in the evaluation of the model and various metrics can be calculated using it namely Accuracy, Precision, Recall and F1 Score. These metrics give a fair idea on how the model has performed which have been explained in Section 5.4 in detail.

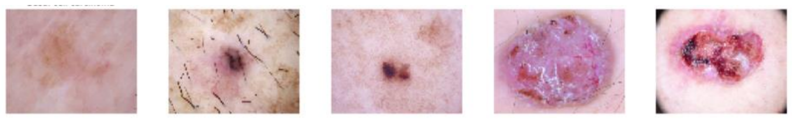
3 DATASET

The dataset for the project is from Harvard Dataverse HAM10000(Humans Against Machine with 10000 training images) which was used in a challenge hosted by the International Skin Imaging Collaboration (ISIC) in 2018. ISIC has developed technologies and techniques which address issues like patient privacy and interoperability [3]. Dermatoscopic images for this dataset were collected from various populations and finally collected to form a training dataset of 10015 images. The dataset also consists of a metadata csv file which contains information related to the images. The images are of seven different diagnostic categories into which we will classify our data. The skin lesion classes along with sample images of each class are described below –

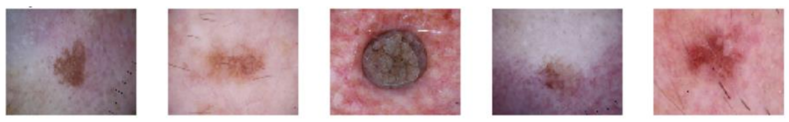
- **Actinic Keratoses (akiec)** - An actinic keratosis is a rough, scaly patch on the skin that develops from years of exposure to the sun. It's most commonly found on face, lips, ears, back of the hands, forearms, scalp or neck.[4]



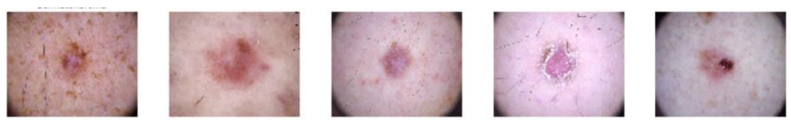
- **Basal Cell Carcinoma (bcc)** – Basal cell carcinoma is a type of skin cancer. Basal cell carcinoma begins in the basal cells, a type of cell within the skin that produces new skin cells as old ones die off. Basal cell carcinoma often appears as a slightly transparent bump on the skin, though it can take other forms. [5]



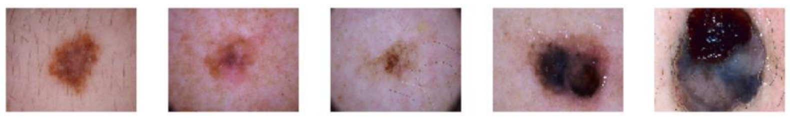
- **Benign Keratosis Like Lesions (bkl)** – Benign or Seborrheic keratosis is noncancerous (benign) skin growths that some people develop as they age. They often appear on the back or chest but can occur on any part of the body. [6]



- **Dermatofibroma (df)** - A dermatofibroma is a harmless, non-cancerous bump. Dermatofibromas are harmless growths within the skin that usually have a small diameter. They can vary in colour, and the colour may change over the years. [7]

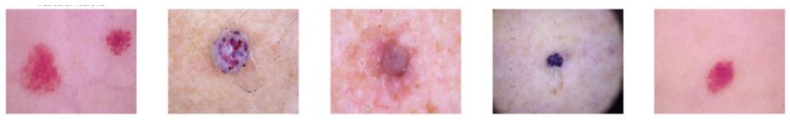


- **Melanoma (mel)** - Melanoma, also known as malignant melanoma, is a type of cancer that develops from the pigment-containing cells known as melanocytes. Melanomas typically occur in the skin, but may rarely occur in the mouth, intestines, or eye. [8]



- **Melanocytic Nevi (nv)** - Melanocytic nevi are benign neoplasms or hamartomas composed of melanocytes, the pigment-producing cells that constitutively colonize the epidermis. This often leads to a higher risk of melanoma, a serious skin cancer. [9]

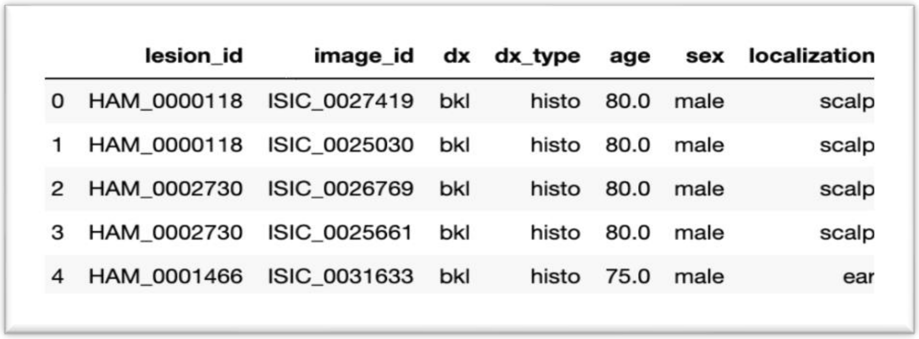
- **Vascular Lesions (vasc)** - Vascular lesions include acquired lesions (e.g., pyogenic granuloma) and those that are present at birth or arise shortly after birth (vascular birthmarks). Vascular tumours may be benign (not cancer) or malignant (cancer) and can occur anywhere in the body. [10]



The raw metadata file contains important information about each image. The dataset file contains the following columns –

- **Lesion_ID** – This column represents an ID for a particular lesion type.
- **Image_ID** – This column represents the ID for the particular image.
- **Dx** – This column represents the diagnostic categories which are discussed above
- **Dx_type** - This column represents the type of the diagnostic categories and the measures through which the corresponding diagnostic category was decided.
- **Age** – This column represents the age of the individual whose image sample has been taken.
- **Sex** - This column represents the gender of the individual whose image sample has been taken.
- **Localization** – This column represents the skin areas or regions which have been affected due to suspected cancer.

Given below is a sample of the raw metadata data frame –



	lesion_id	image_id	dx	dx_type	age	sex	localization
0	HAM_0000118	ISIC_0027419	bkl	histo	80.0	male	scalp
1	HAM_0000118	ISIC_0025030	bkl	histo	80.0	male	scalp
2	HAM_0002730	ISIC_0026769	bkl	histo	80.0	male	scalp
3	HAM_0002730	ISIC_0025661	bkl	histo	80.0	male	scalp
4	HAM_0001466	ISIC_0031633	bkl	histo	75.0	male	ear

Fig. 1. Sample raw dataset

3.1 EXPLORATORY DATA ANALYSIS (EDA)

Given below is the EDA of the dataset-

Cell Type

The graph shows the number of images for each diagnostic category or cell type.

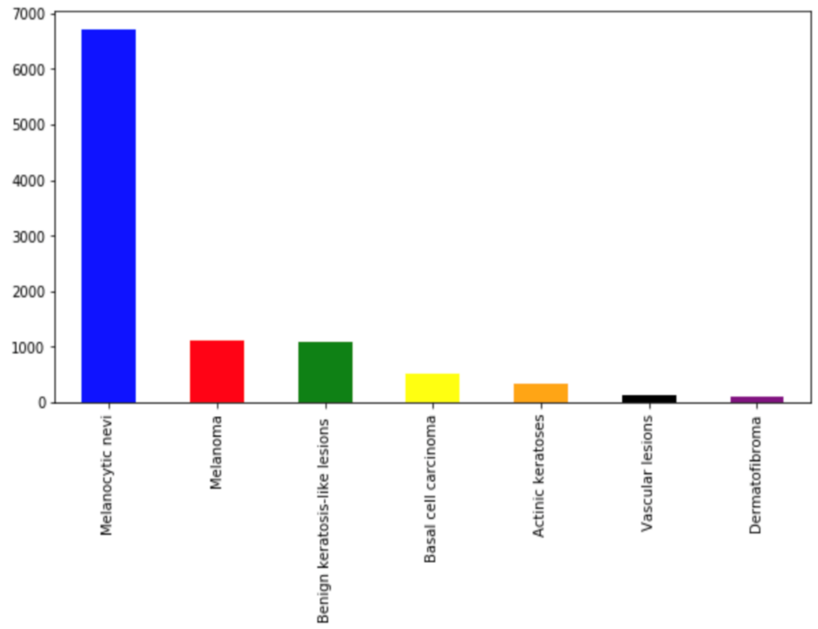


Fig.2 Analysis of cell type (different classes)

It is clear from the analysis that Melanocytic nevi are the most frequent sample in the dataset.

Localization

The graph for localization depicts the number of image samples for each area of skin compromised due to cancer.

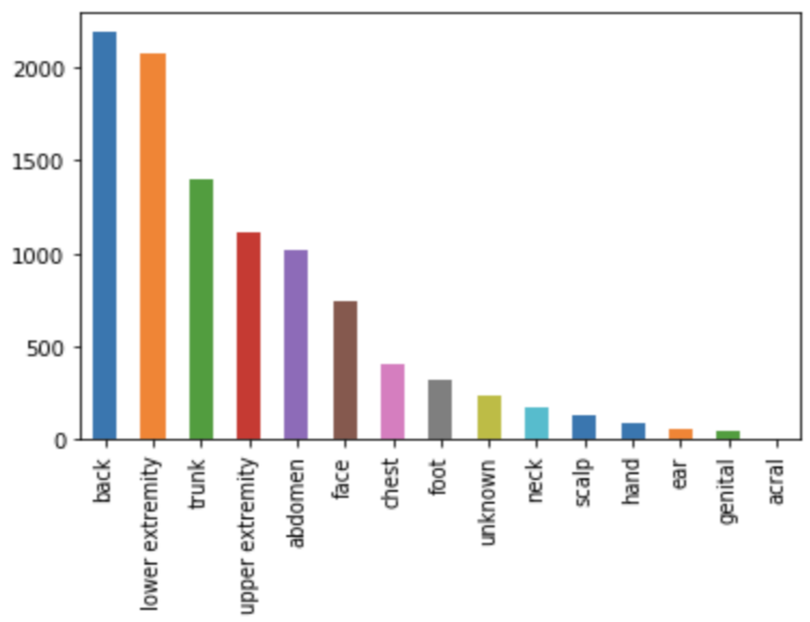


Fig.3 Analysis of Localization

The analysis of the graph shows that back and lower extremity regions are the most affected areas.

Age

The graph for age will depict the number of image samples per age group.

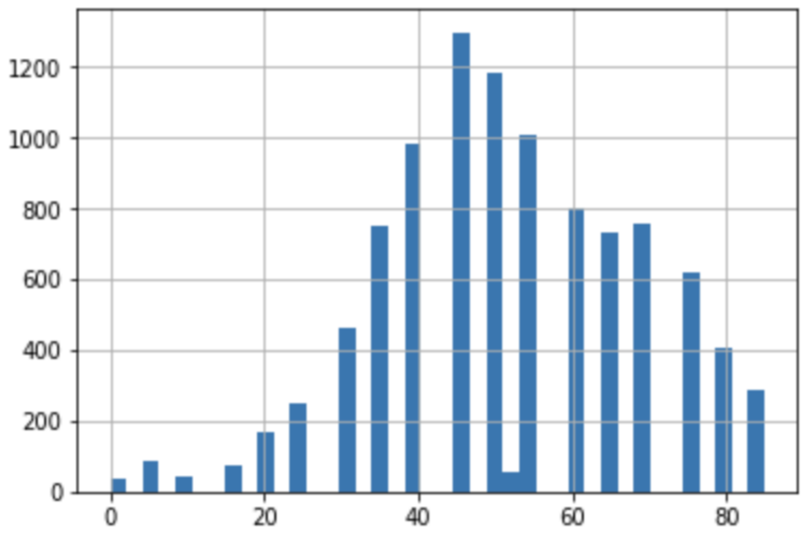


Fig.4 Analysis of Age

This analysis shows that the majority of samples are from people in the age group of 40-60.

4 MODEL USED

4.1 CONVOLUTIONAL NEURAL NETWORK

Convolutional Neural Networks (ConvNets/CNNs) are the most popular class of deep learning neural networks in image recognition field. CNN learn from the image data by finding patterns in the images and using them to classify images.

4.1.1 ARCHITECTURE

There are 3 different layers performing major operations in ConvNets [11]:

Convolutional Layers:
To perform convolutional operations to the input image, extract features and passing information to the next layer. The spatial relationship between pixels is preserved by convolution where it learns image features using a small square matrix along with input data. In CNN terminology, that small square matrix is called 'Kernel' or 'Filter' or 'Feature Detector'. The filter is slide over the image and for each slide dot product with image pixel value and filter value is calculated to form 'Convolved Features' or 'Activation Map' or 'Feature Map'. The size of the feature map depends on three factors:

- Depth: the numbers of filters we used for convolution

- Stride: the number pixels by which filter is slide
- Zero-padding: Using padding of zeroes for border pixels to control the feature map's size.

Mathematically convolutional layer output can be denoted as:

$$\text{out}(N_i, C_{\text{out}_j}) = \text{bias}(C_{\text{out}_j}) + \sum_{k=0}^{C_{\text{in}}-1} \text{weight}(C_{\text{out}_j}, k) \star \text{input}(N_i, k)$$

Where *'N'* is batch size, *'C'* denotes the number of channels, *'H'* is a height of input planes in pixels, and *'W'* is the width in pixels.[12]

ReLU (Rectified Linear Unit) a non-linear function is used after every convolutional operation. In order to handle real-world data convolution, need to learn nonlinearity, so ReLU provides the nonlinearity to convolution (which is a linear operation) by replacing negative pixel values with zero.

Pooling Layers:

To perform 'spatial pooling' or 'sub-sampling' or 'down-sampling', where for each feature map dimensionality is reduced but the most important information is retained. It is of different types such as Max, Sum, Average etc.
For commonly used Max Pooling, the largest pixel is taken from the feature map within the spatial neighbourhood (predefined small matrix). If instead of the largest element, we use the sum of all pixels it is call Sum Pooling or if we use the average of all then it is Average Pooling.

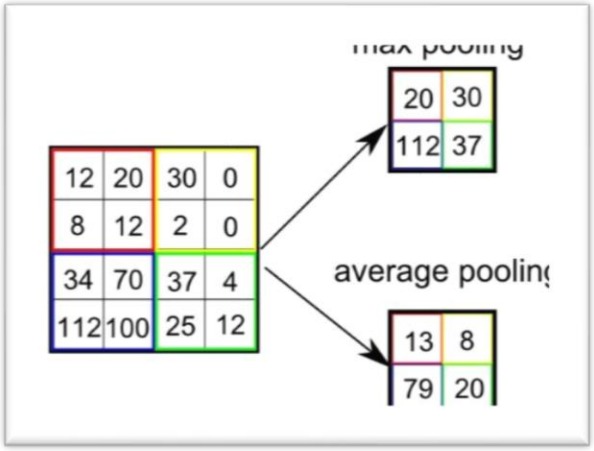


Fig.5 Pooling Techniques [11]

Mathematical notation for Max pooling:

$$out(N_i, C_j, h, w) = \max_{m=0,\ldots,kH-1} \max_{n=0,\ldots,kW-1} \text{input}(N_i, C_j, \text{stride}[0] \times h + m, \text{stride}[1] \times w + n)$$

Where *'N'* is batch size, *'C'* denotes the number of channels, *'h'* is a height of input planes in pixels, and *'w'* is the width in pixels.[13]

Fully Connected Layer:
It is a Multi-Layer Perceptron (every neuron in one layer is connected to every neuron in the next layer) which uses 'softmax' activation function in the output layer. Mainly the fully connected layer serves as a classifier on the top of the extracted features which assigns the probability of the image being predicted by the algorithm. It also helps in learning nonlinear combinations of the extracted features. It gets trained with backpropagation. Softmax function at the output layer processes the arbitrary real values and compress it to a vector of values between zero to one that sums to one.

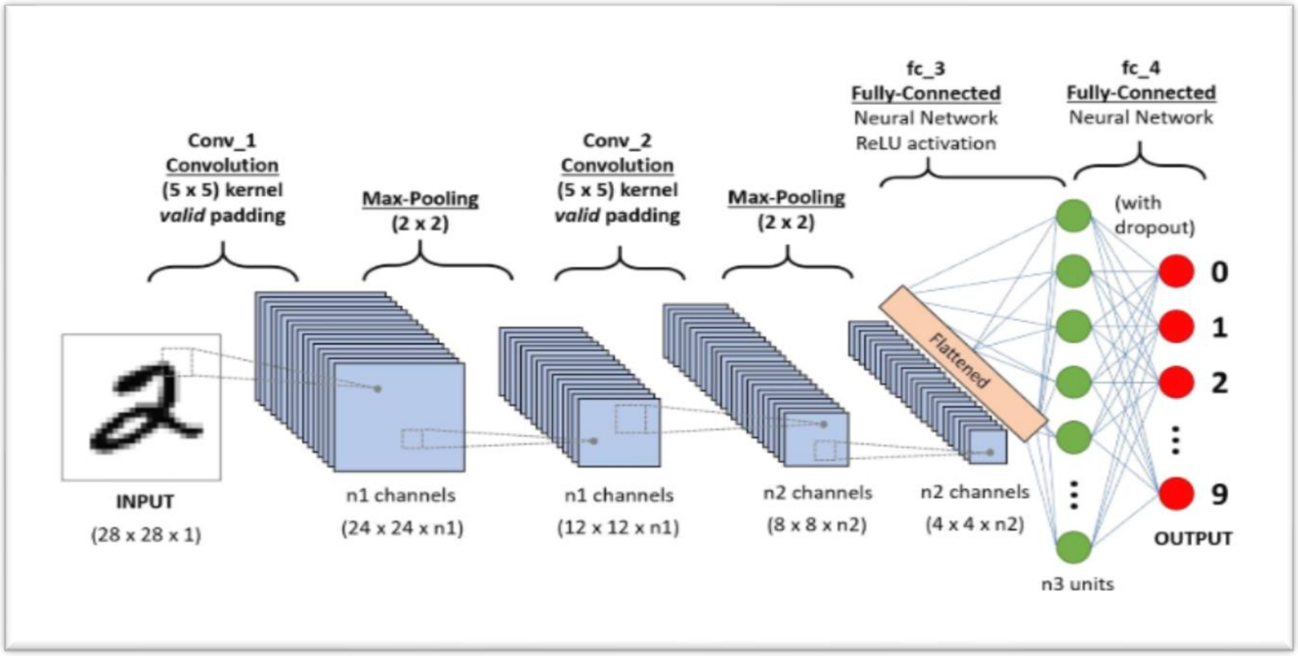


Fig.6 A CNN sequence to classify handwritten digits [11]

4.1.2 WHY CNN?

CNN because it provides a high representation of an image, which is easy to predict by using different sized kernels, which can convolve on image to detect different shapes and edges. Unlike dense networks, which might easily fail if the image is not centred, or the Recurrent Neural Networks which has a single weight matrix used by recurrent units, which is not helpful in finding spatial features and shapes.

4.2 RESNET50

It is a type of ResNet (Residual Network). Residual networks are arguably one of the most ground-breaking work in the computer vision/deep learning community and are from the same ConvNet family. ResNet was the winner of ImageNet challenge in 2015, it outperformed AlexNet. It follows the state-of-the-art extremely deep CNN architecture; it allows to train hundred of layers successfully. ResNet-152 has 152 convolution layers and ResNet-50 has fewer layers than ResNet-152 and it is frequently used.[14]

4.2.1 COMPARISON WITH CNN

The basic difference between ResNet and CNN is that it has a greater number of combination of layers. In order to achieve better results than single-layer networks, it is a common trend of using a deep network architecture, though Increasing network depth will not work if all the layers are just stacked together. Deep networks have a problem of vanishing gradient resulting in hard to train, during back-propagation gradient may become extremely small with repeated multiplication. As a result, network performance gets saturated or even starts degrading, as we go deep in the network. ResNet resolves this issue.[15]

4.2.2 ARCHITECTURE

ResNet's core idea for resolving performance problem is to introduce an "identity shortcut connection" (creating residual blocks) that skips one or more layers, skip connections are also known as gated units or gated recurrent units and have a strong similarity to recent successful elements applied in RNN.[16] The following figure shows a residual block:

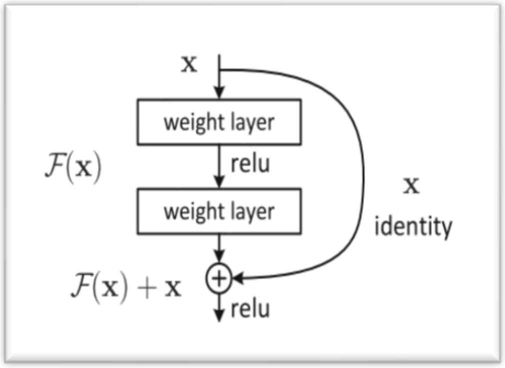

x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Fig.7 Residual block [17]

With reference to the block input, intermediate layers of a block learn a residual function. Residual function act as a refinement step where it learns to adjust the feature map for higher quality features.

In ReNet50, the two-layer residual block is replaced with a three-layer bottleneck block. It uses smaller sized convolutions to reduce and restore the channel depth which results in reduced computational load while calculating the convolution.

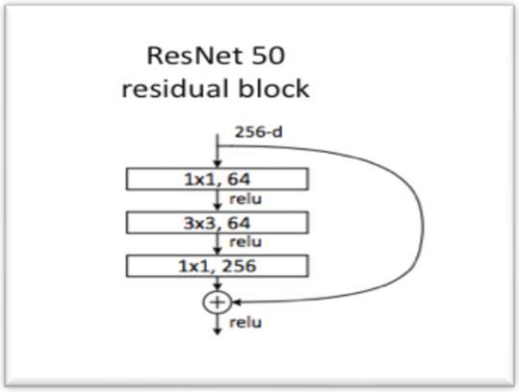


Fig.8 ResNet50 bottleneck block [18]

Following diagram shows the complete structure of ResNet-50 model, where periodically the feature mapping is downsampled by sliding convolution along with an increase in channel depth to preserve the time complexity per layer.

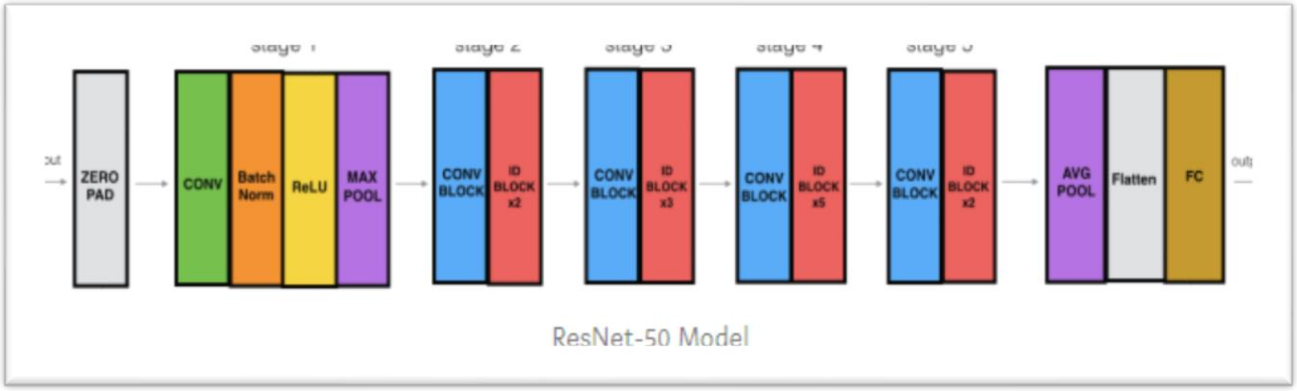


Fig.9 Complete ResNet-50 Architecture [19]

5 METHODOLOGY

5.1 OVERALL OVERVIEW

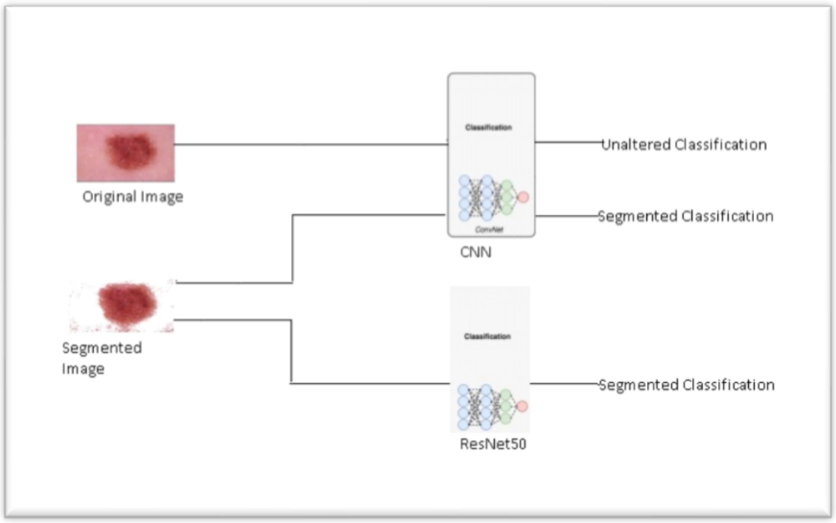


Fig.10 Methodology Overview

As you can see in the above figure, we have passed the original and the segmented images separately through the Convolutional Neural Network. Our CNN consists of a convolutional layer followed by a pooling layer, a dropout layer (to prevent overfitting) and finally a fully connected layer.
The segmented images were also fed in as the input for the ResNet50 model. The ResNet50 model consists of ResNet block (Explained in Section 4.2) followed by a series of convolutional blocks which have the same layers as mentioned above.
Both models classify the images into the different classes of cancer types.

5.2 DATA PRE-PROCESSING

Various data pre-processing techniques were performed on the dataset to make it suitable for training purposes. Given below are the steps performed in order to make the dataset ready for training.

5.2.1 DATA CLEANING

Each column of the dataset was checked for missing values. Upon elaboration, it was found that only the age column had around 57 missing values. The null values were then replaced by the mean of remaining values.

5.2.2 REMOVING CLASS IMBALANCE

As it was observed during Exploratory Data Analysis, there is some class imbalance which can be seen in Fig 2. It is clear that Melanocytic Nevi cell type has the most number samples for the diagnostic category, i.e. 6705/10015. Hence class imbalance needs to be removed. Below is the graph which shows the number of samples per class after reducing the dataset according to requirements.

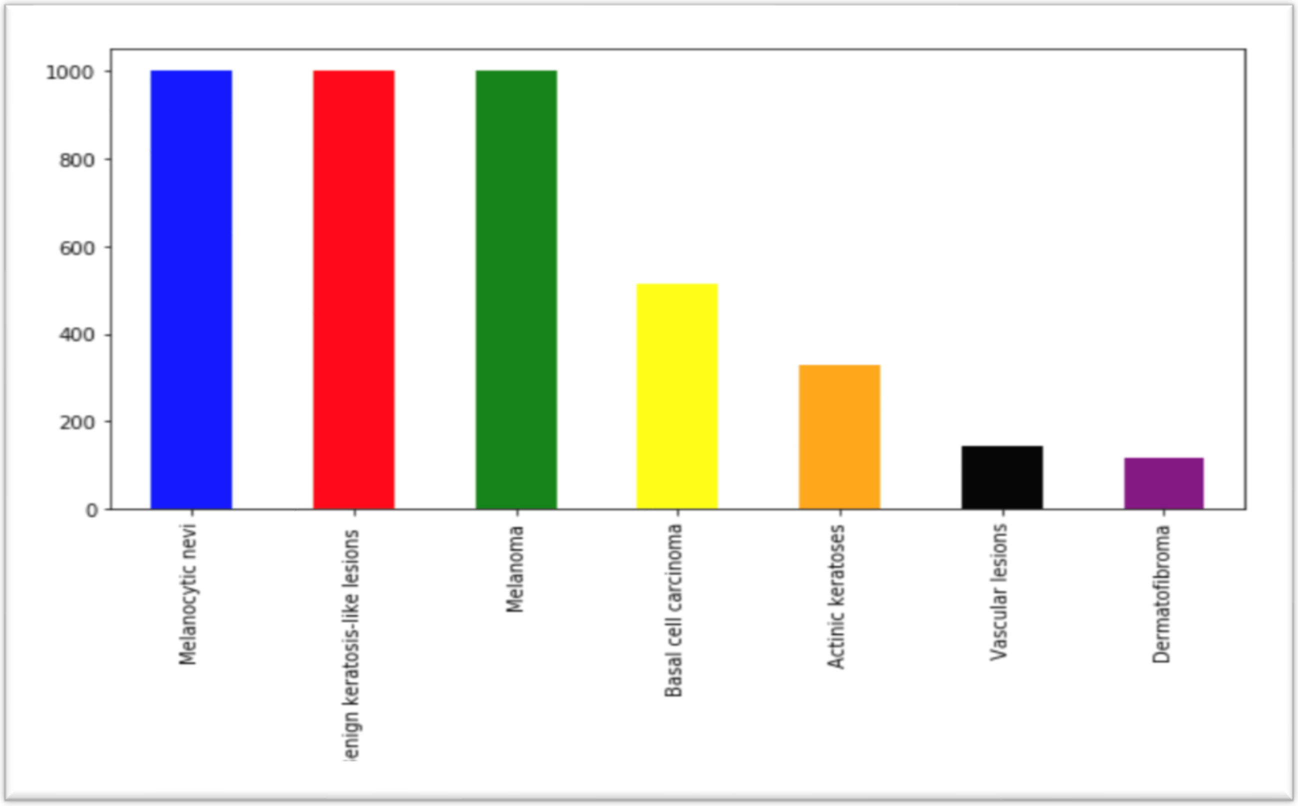


Fig.11 Class Imbalance removed for first 3 classes

After removing the class imbalance, Melanocytic nevi, Benign keratoses like lesions and Melanoma are now uniform at 1000 samples each and the size of image dataset is now reduced to 4098 images.

5.2.3 IMAGE SEGMENTATION

Image segmentation is a technique where the image is divided into different segments so as only important parts of an image can be utilized. Redundant parts are removed and only a significant portion of an image is left to analyse. For example, in this case, we would only like to retain the supposedly cancerous part of the skin and focus on it and remove the rest of the skin portion. The redundant skin area will be removed, and the deep learning model will only focus on the cancer cells. There are various ways to perform image segmentation tasks depending upon the requirement. However, it may not always be beneficial as in some cases, image segmentation can result in loss of important information which was otherwise required. Hence a comparison of both approaches that are image classification with original images as well as with segmented images can give a clear picture. Below is a flowchart which shows the steps involved in converting the original image to a segmented image, where we have used **thresholding** technique.

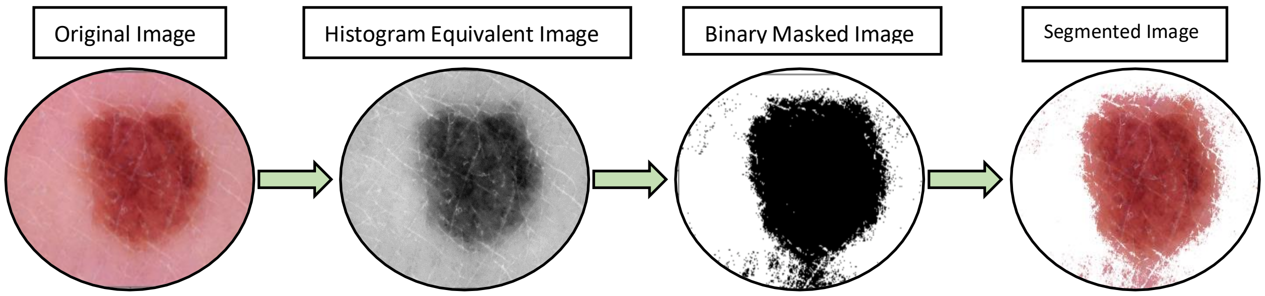


Fig.12 Process of Image Segmentation

Firstly, the original image, as shown in Fig 12 is converted to a Histogram Equivalent Grayscale Image as shown in Fig 12 by applying histogram equalization technique (contrast enhancement). Then we create a binary masked image using thresholding technique where calculate the mean of each pixel value of the obtained grayscale image and compare the 75% of the mean of all pixels (as a threshold) to the value of each pixel. If pixel values are greater than the adjusted mean, the pixel becomes black, otherwise white. Hence a Masked Image is obtained as shown in Fig 12.
Finally, the masked image is superimposed on the original image to get the final output i.e. a Segmented Image as shown in Fig 12.

5.2.4 FEATURE ENGINEERING

Currently, the metadata csv file contains only information about the image. Four new features were added which are as follows –

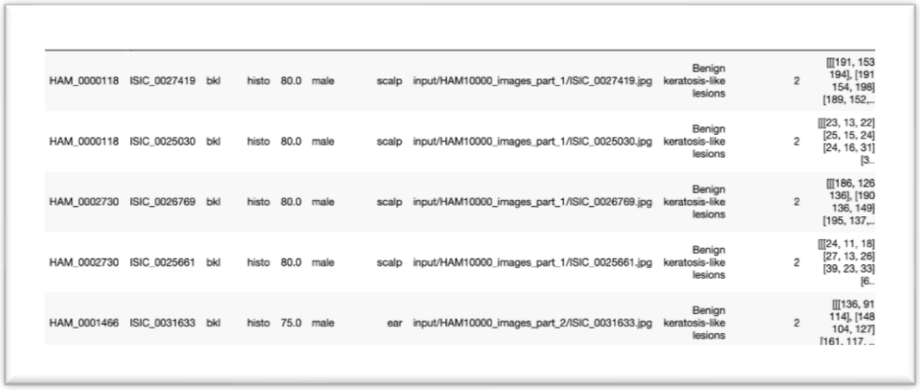
- Path – Path column was added using the Image_ID column which contains the path of the image which must be opened and used.

- Cell_type – Cell_type column was added using the dx column. Dx column contains the abbreviated names for the diagnostic categories. Hence to show more readable names, cell_type column with the full name of the classes is displayed. For example, if dx is 'bkl', the cell_type would be 'Benign keratosis-like lesions'. This was done to make the classes more readable.

- Cell_type_idx – This feature was engineered from cell_type column as discussed above. This is the target variable and converts the classes into numbers ranging from 0-6.

- Image – Finally, image feature was added which contains the array of the image to be classified and was engineered using the path column as discussed above. The images were also resized to meet the requirements.

Sample of the dataset after feature engineering is shown below-



HAM_0000118	ISIC_0027419	bkl	histo	80.0	male	scalp	input/HAM10000_images_part_1/ISIC_0027419.jpg	Benign keratosis-like lesions	2	[[[191, 153 194], [191 154, 198] [189, 152,...
HAM_0000118	ISIC_0025030	bkl	histo	80.0	male	scalp	input/HAM10000_images_part_1/ISIC_0025030.jpg	Benign keratosis-like lesions	2	[[[23, 13, 22] [25, 15, 24] [24, 16, 31] [3...
HAM_0002730	ISIC_0026769	bkl	histo	80.0	male	scalp	input/HAM10000_images_part_1/ISIC_0026769.jpg	Benign keratosis-like lesions	2	[[[186, 126 136], [190 136, 149] [195, 137,...
HAM_0002730	ISIC_0025661	bkl	histo	80.0	male	scalp	input/HAM10000_images_part_1/ISIC_0025661.jpg	Benign keratosis-like lesions	2	[[[24, 11, 18] [27, 13, 26] [39, 23, 33] [6...
HAM_0001466	ISIC_0031633	bkl	histo	75.0	male	ear	input/HAM10000_images_part_2/ISIC_0031633.jpg	Benign keratosis-like lesions	2	[[[136, 91 114], [148 104, 127] [161, 117, ...

Fig.13 New features added

The cell_type_idx was the target column whereas all the rest of the columns were used as the input data.

5.2.5 TRAIN/TEST/VALIDATION SPLIT

The Hold-Out evaluation has been done for this project. This is to ensure that the model is tested on unseen data, which makes sure that our output is not biased. Hence a train/test/validation split was done in the ratio of 70:20:10.

5.2.6 DATA NORMALIZATION (STANDARDIZATION)

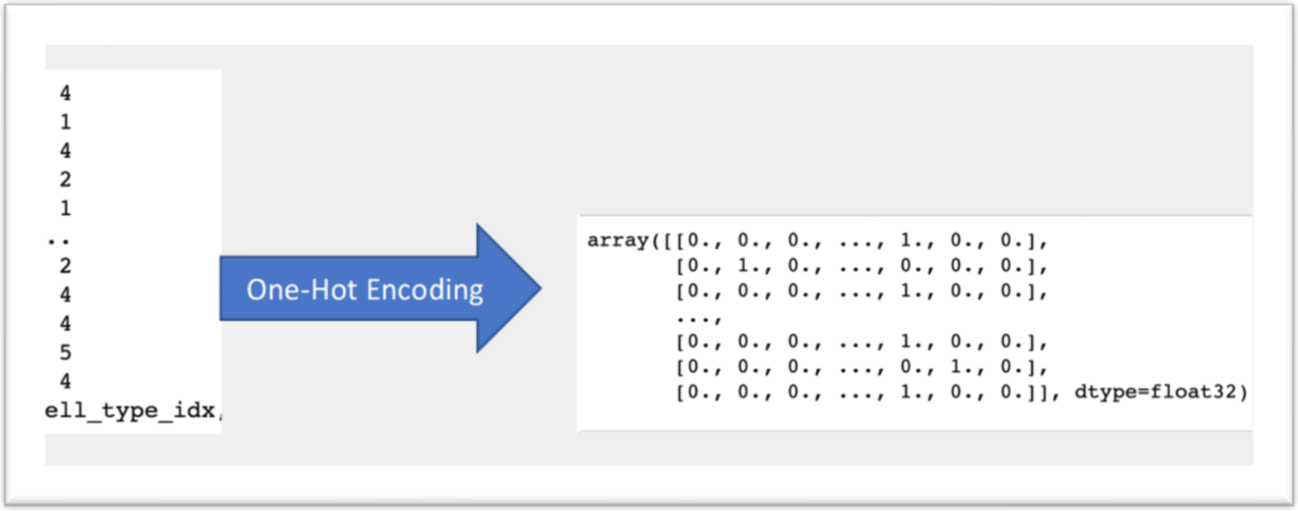
Before feeding data to a neural network, it's a good practice to normalize the data as it increases the learning process and moreover helps the data to converge faster**.** Normalization brings every attribute to the same required range. For this problem, the Standardization technique was used. Features are rescaled in a way so that they have a mean = 0 and standard deviation = 1. The final output is obtained by subtracting the mean from the input and finally dividing it by the standard deviation. Given below is the formula to obtain the normalized outputs after standardization –

$$z = \frac{x - \mu}{\sigma}$$

Where μ is the mean of input variables,
σ is the standard deviation,
x is the original value,
z is the normalized value.

5.2.7 ONE-HOT ENCODING

Finally, one-hot encoding was performed on labels i.e. both the train and test output variables. One-hot encoding is mainly performed on categorical data. Doing one-hot encoding allows the neural network algorithm to predict data more accurately as model won't assume natural ordering between categories. To perform one-hot encoding, "binarization" technique is performed on the target variables. All classes from 0-6 are one-hot encoded using this technique. For example, category 2 is converted to [0,0,1,0,0,0,0] to specify that this class is 2 as 0 represents the value is non-existent whereas 1 represents that the value is existent. Given below is the conversion of the categorical variable to one-hot encoded variables.



```
 4
 1
 4
 2
 1
..
 2
 4
 4
 5
 4
ell_type_idx,
```

```
array([[0., 0., 0., ..., 1., 0., 0.],
       [0., 1., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 1., 0., 0.],
       ...,
       [0., 0., 0., ..., 1., 0., 0.],
       [0., 0., 0., ..., 0., 1., 0.],
       [0., 0., 0., ..., 1., 0., 0.]], dtype=float32)
```

Fig.14 Showing One-Hot Encoding

5.3 MODEL TRAINING

5.3.1 PARAMETERS BEING TUNED

- ***Activation Function:***

  It is important for a neural network to understand something complicated and introduce non-linear properties to the network. It converts the input signal to an output signal.
  We have used **Rectified Linear Unit (ReLU)** as our activation function. It is a widely used activation function and has a range of 0 to infinity. It applies the element-wise activation function on the input data. Mathematically,

  $$\text{ReLU}(y) = \max(0, y)$$

  Where y is the input to the function.

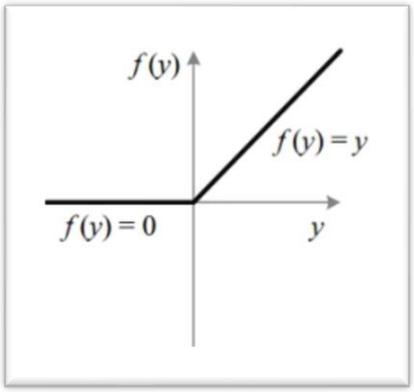


Fig.15 ReLU function [20]

- ***Number of Layers:***

  It determines the capability of the network to represent the structure in data. It denotes the number of hidden layers the model has. Generally, if you have too many layers it can lead to overfitting whereas too less can lead to underfitting and the model is not trained properly in either case. It affects the number of parameters a network has. Also, the number of layers a network has, the more time it takes to train it.

- ***Number of Neurons in Hidden Layers:***

  It represents the number of neurons in each hidden layer. If we have too many neurons per layer it increases the chance of model overfitting the data.

- ***Learning Rate:***

  It has a huge impact on training and stability of the model. If we have a large learning rate, then there is a chance that we won't reach convergence and the model would not be optimally trained. On the other hand, if we have a very small learning rate, it becomes very difficult to train the model and there is a chance of getting stuck in the local minima.

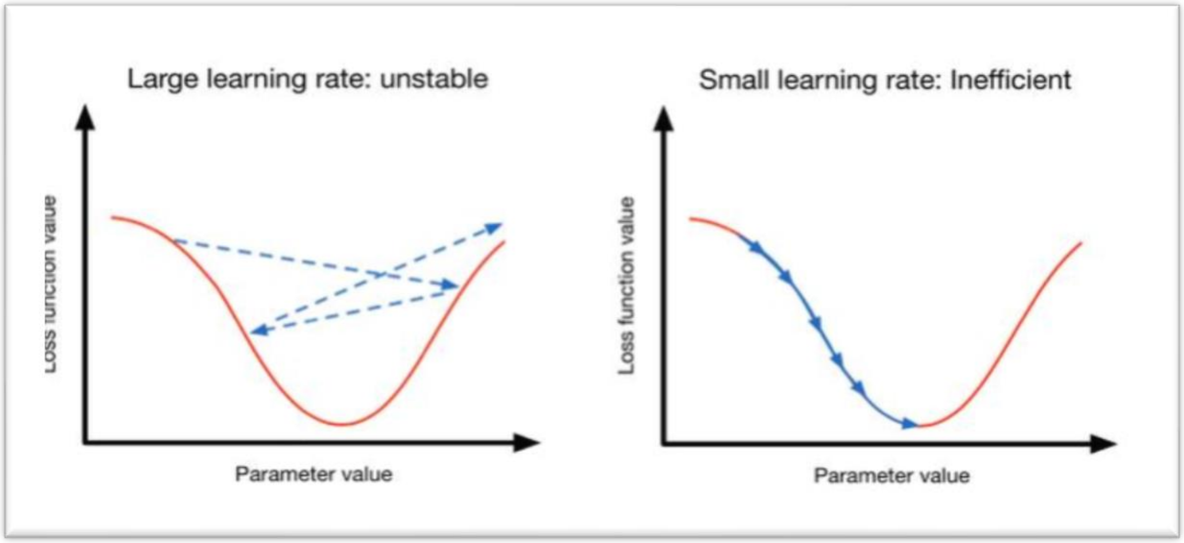


Fig.16a Importance of Learning Rate [20]

For tuning the learning rate, we have used **Adam optimiser**. It uses estimations of first and second moments of the gradient to adapt the learning rate for each weight of the neural network. Also, momentum has been set in Adam so that the model does not get stuck in local minima. This improves model accuracy and reduces loss.
Mathematically,

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t}$$

Where, m(hat) and v(hat) are corrected first and second moments respectively,
m and v are the moving averages
B_1, B_2 are the exponential decay rate for the first and second moments respectively.

Loss Function:

Loss Functions let the optimisation function know how well it is performing. We have used **Categorical Cross Entropy**. This loss function would train the model to predict a probability over the *C* classes for each image. It is a combination of Softmax activation and Cross entropy loss. We chose this because in our case, one image can belong to only one class.

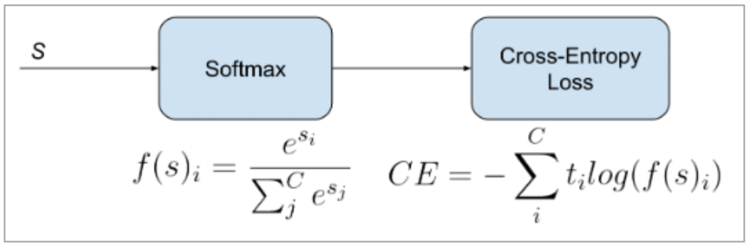

s
Softmax
Cross-Entropy Loss
$f(s)_i = \frac{e^{s_i}}{\sum_j^C e^{s_j}}$
$CE = -\sum_i^C t_i log(f(s)_i)$

Fig.16b Categorical-Cross Entropy [21]

Where CE = loss function output,
t = ground truth,
s = CNN score

5.3.2 UPDATING WEIGHTS AND BIASES

- We try to minimize the cost function by updating weights and biases.
- The cost function is given by the formula [20]:

$$Cost = \frac{1}{N}\sum_{i=1}^{N}(Y_i' - Y_i)^2$$

where N = number of samples
Y' = calculated value and Y is the target

- We feed forward the input samples and for each sample calculate [20]:

$$z^{x,l} = w^l a^{x,l-1} + b^l$$

Where, z= weighted input
W = weight, b= bias and a = activation function

- Error at the output layer is calculated as [20]:

$$\delta^{x,L} = \nabla_a C_x \odot \sigma'(z^{x,L}).$$

Where, δ^a = error

- This error is then backpropagated through each layer from output to input layer. It is calculated as [20]:

$$\delta^{x,l} = ((w^{l+1})^T \delta^{x,l+1}) \odot \sigma'(z^{x,l}).$$

Where l is the layer

- Finally, we compute the gradient descent to update the weights and biases using the formula [20]:

$$w_k \rightarrow w'_k = w_k - \eta \frac{\partial C}{\partial w_k}$$
$$b_l \rightarrow b'_l = b_l - \eta \frac{\partial C}{\partial b_l}.$$

Where n is the learning rate

5.4 MODEL EVALUATION

For testing the accuracy of the model, the following metrics have been considered:

- ***Confusion Matrix:*** The confusion matrix has the predicted classes as its columns and the actual classes are the rows of the matrix. There are 4 cases possible:
  - *True Positive (TP):* When the classifier predicts the correct class of the image. For instance, the classifier predicts that the image belongs to class Melanoma and it actually does
  - *True Negative (TN):* When the classifier predicts that the image does not belong to the class and it actually does not. For example, the classifier predicts that the image does not belong to class Melanoma and it actually does not
  - *False Positive (FP):* When the model predicts that the image belongs to the class, but it actually does not. For instance, the classifier predicts that the image belongs to class Melanoma, but it does not
  - *False Negative (FN):* When the model predicts that the image belongs to some other class, but it does not. For example, the classifier predicts that the image belongs to some other class instead of Melanoma, but it actually belongs to Melanoma.

Basically, "True", "False" indicates whether the classifier predicts the correct class of the image or not whereas "Positive", "Negative" indicate whether the classifier predicted the desired class or not.
From the confusion matrix, we get the following metrics:

a. Accuracy:

Accuracy is defined as out of all the classified items, how many of those were correctly classified. Mathematically,

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

Where TP = True Positive, TN = True Negative, FP = False Positive and FN = False Negative.

b. Precision:

Precision answers the question that whenever the model predicts the desired class, how often is it correct. Its formula is:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

c. Recall:

Recall tells us that if the image actually belongs to the desired class, how often it was predicted correctly. Mathematically,

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

d. F1-score:

It is just a harmonic mean of Precision and Recall. It is given by the formula:

$$f_1\text{-score} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Note:

We have used Tensorflow as the framework for our project and all of the code is written in Python. We used Google Colab (cloud service built on top of Jupyter Notebook) for running our model as it provides free GPU service. For running the model and performing the steps illustrated in Methodology, you will have to import the libraries listed below:

- Numpy
- Pandas
- Operating System(os)
- Glob from Glob module
- Seaborn
- Image from PIL (Python Imaging Library) module
- Label Binarize from Sklearn.preprocessing
- Confusion Matrix from Sklearn.metrics module
- itertools
- Tensorflow
- Drive from Google Colab module (Only if you are using Colab and need to mount the drive)

6 RESULTS

6.1 BEFORE TUNING

Before tuning our model, we passed the unsegmented images through CNN and then the segmented images through the CNN model.

Training results for unsegmented images:

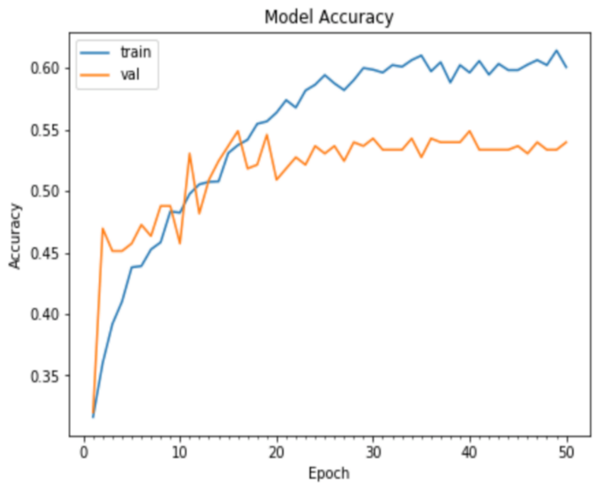


Fig 17a Training Accuracy

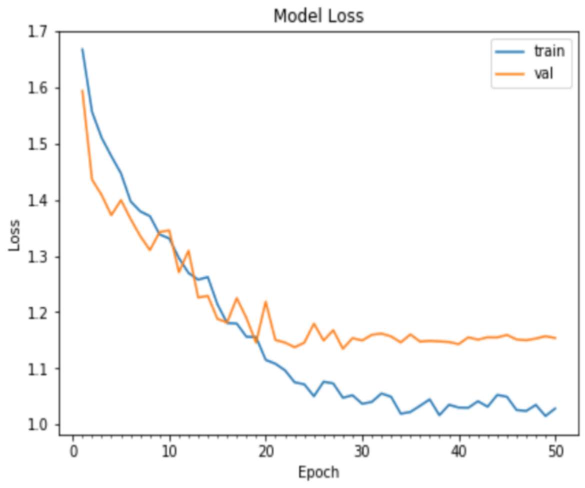


Fig 17b Training Loss

Figure 17(a) and 17(b) shows the train and validation accuracy and loss achieved during training. It has a training accuracy of 60% but the validation accuracy is just 51%.

Training results for segmented images:

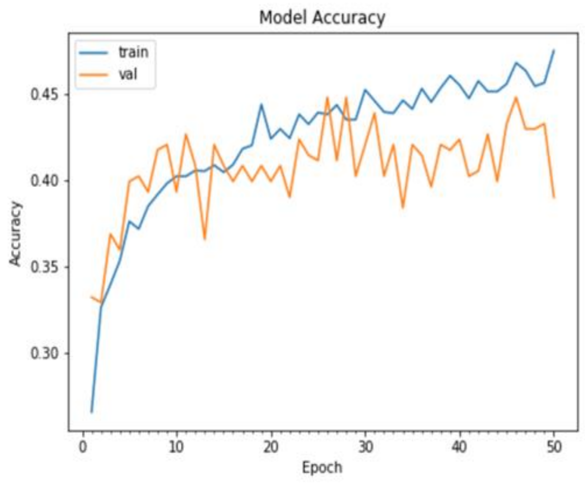


Fig 18a Training Accuracy

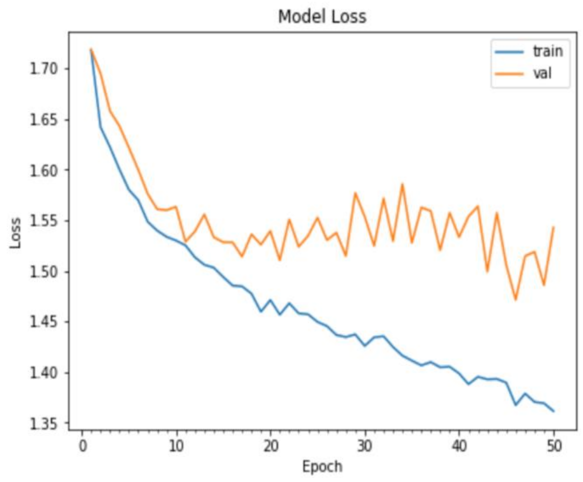


Fig 18b Training Loss

As you can see from figure 18(a) that the accuracy achieved was only 39% for validation and 47% for training and from figure 18(b), we can see that the loss is really high too.

From Figures 17 and 18, it is clear that there is a need to tune the model.

6.2 AFTER TUNING

After tuning, we fixed the parameters to the following values:

1. Number of layers: 12
2. Number of Neurons in Hidden Layers: 320
3. Learning Rate: 0.001

Now, the unsegmented images are passed through the tuned CNN model and the segmented images are passed through both tuned CNN and ResNet50 as shown below.

Training results for unsegmented CNN:

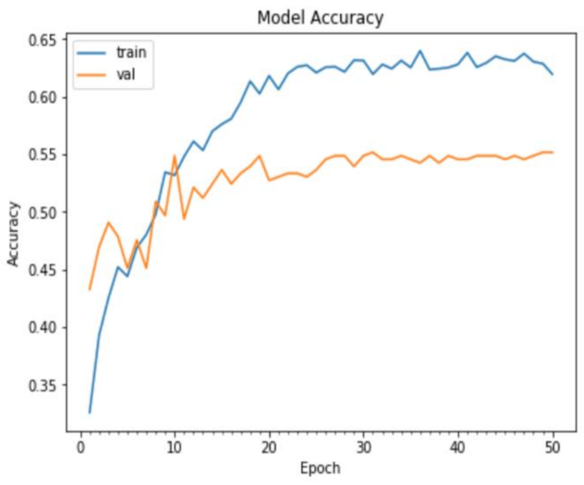


Fig 19a Training Accuracy

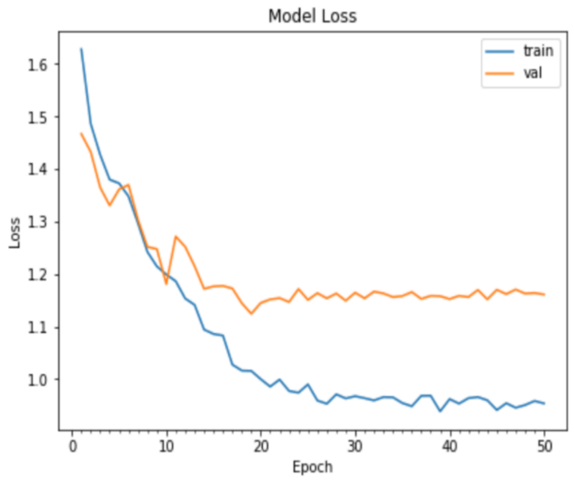


Fig 19b Training Loss

From figure 19a, you can see that the training accuracy reached up to 65% and validation accuracy was around 57%. Moreover, you can see that the validation accuracy becomes almost constant after like 20 epochs and the reason for that could be that even though we downsampled our dataset from 10015 to 4098 images there is still a considerable imbalance as some of our classes have just 150 images compared to 1000 images in other. This imbalance could have led to the training set being dominated by the dominating class and thus the model is not able to learn properly.

From figure 19b, you can see that after tuning the loss decreases with each epoch for the training set and finally reaches its minimum value meaning the model is efficiently trained.

Training results for segmented CNN:

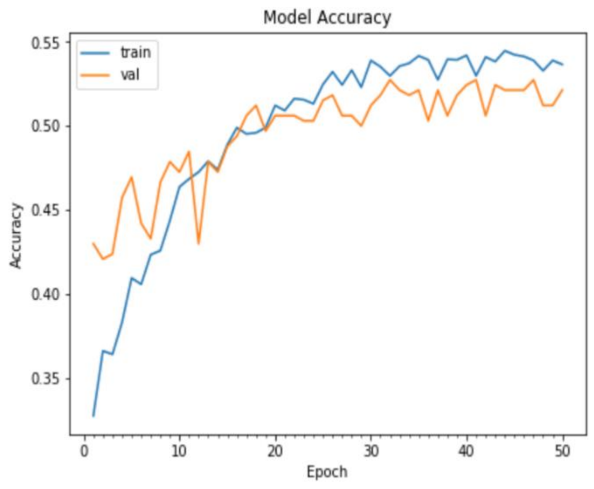


Fig 20a Training Accuracy

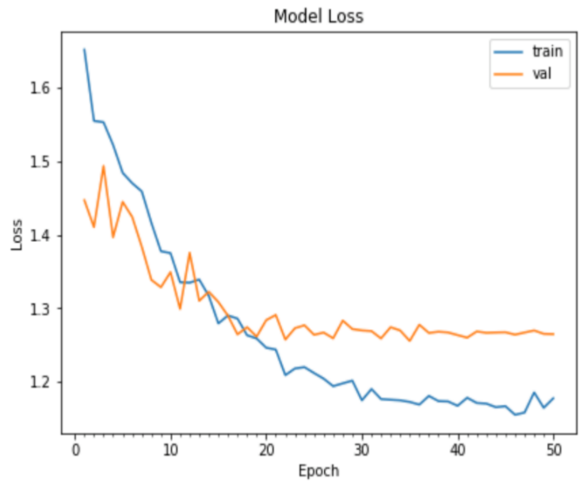


Fig 20b Training Loss

From figure 20a, you can see that the training accuracy reached up to 54% and validation accuracy was around 51%.
From figure 20b, you can see that after tuning the loss has declined from 137 in figure 18b to around 10.

Training results for segmented ResNet:

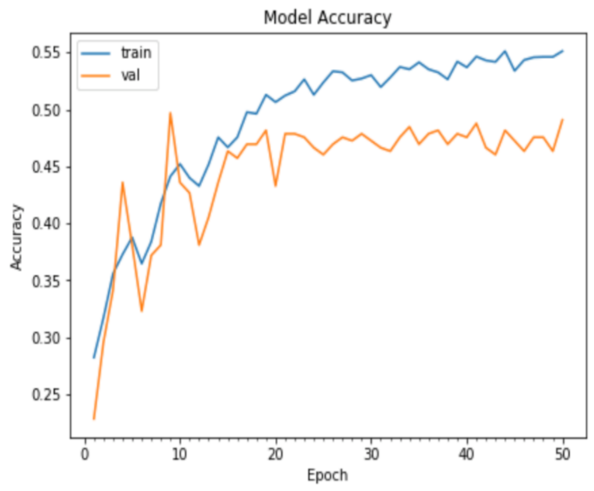


Fig 21a Training Accuracy

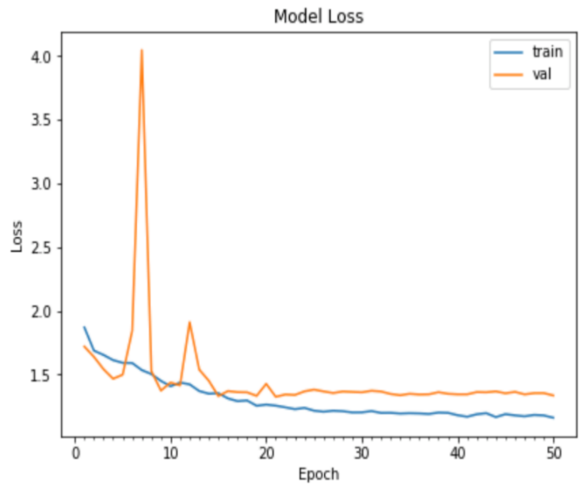


Fig 22b Training Loss

Figure 21(a) represents the accuracy achieved during the training process. You can see that the training accuracy reached up to 55% and validation accuracy was around 52%.
From figure 21(b) we can see that the loss curve is also a smooth exponential decreasing curve and achieves its minimum value towards the end meaning the model was efficiently trained.

6.3 TEST RESULTS

Test results for unsegmented CNN:

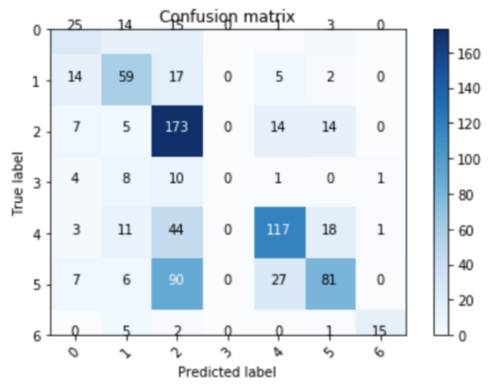


Fig. 22 Confusion Matrix for unsegmented CNN

Figure 22 shows the confusion matrix. The x-axis has the predicted class and the y-axis has the actual class. The number in the boxes indicate how many images were predicted. The higher the number the darker the colour. The diagonal elements represent True Positives and it shows that the model performed well for almost all the classes except for class 3 as it has really fewer images in the dataset.

CLASS	PRECISION	RECALL	F1-SCORE
0	0.42	0.43	0.42
1	0.55	0.61	0.58
2	0.49	0.81	0.61
3	0.00	0.00	0.00
4	0.71	0.60	0.65
5	0.68	0.38	0.49
6	0.88	0.65	0.75
ACCURACY		**57 %**	

Table 1: Test Accuracy for unsegmented CNN

Table 1 shows the evaluation metrics achieved for each class. We achieved a test accuracy of 57% for this model.

Test results for segmented CNN:

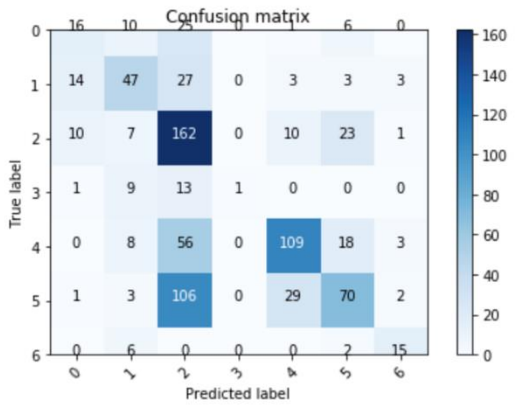


Fig. 23 Confusion Matrix for unsegmented CNN

Figure 23 shows the confusion matrix. The x-axis has the predicted class and the y-axis has the actual class. The number in the boxes indicate how many images were predicted. The higher the number the darker the colour. The diagonal elements represent True Positives and it shows that the model performed well for almost all the classes except for class 3 as it has really fewer images in the dataset. Also, you can see that the model was not able to predict class 5 properly either because even it had fewer images.

CLASS	PRECISION	RECALL	F1-SCORE
0	0.38	0.28	0.32
1	0.52	0.48	0.50
2	0.42	0.76	0.54
3	1.00	0.04	0.08
4	0.72	0.56	0.63
5	0.57	0.33	0.42
6	0.62	0.65	0.64
ACCURACY		**51 %**	

Table 2: Test Accuracy for segmented CNN

Table 2 shows the evaluation metrics achieved for each class. We achieved a test accuracy of 51% for this model.

Test results for segmented ResNet:

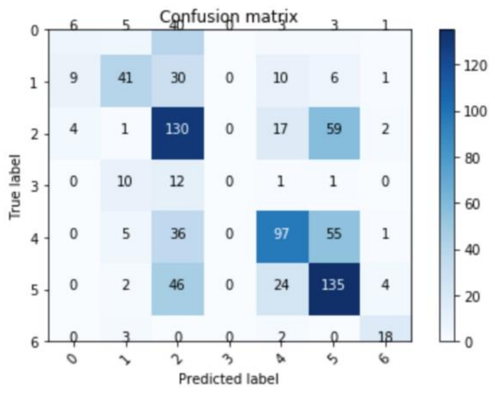


Fig. 24 Confusion Matrix for segmented ResNet

Figure 24 shows the confusion matrix. The x-axis has the predicted class and the y-axis has the actual class. The number in the boxes indicate how many images were predicted. The higher the number the darker the colour. The diagonal elements represent True Positives and it shows that the model performed well for almost all the classes except for class 3 as it has really fewer images in the dataset.

CLASS	PRECISION	RECALL	F1-SCORE
0	0.32	0.10	0.16
1	0.61	0.42	0.50
2	0.44	0.61	0.51
3	0.00	0.00	0.00
4	0.63	0.50	0.56
5	0.52	0.64	0.57
6	0.67	0.78	0.72
ACCURACY		**52 %**	

Table 3: Test Accuracy for segmented ResNet

Table 3 shows the evaluation metrics achieved for each class. We achieved a test accuracy of 52% for this model.

6.4 COMPARISON BETWEEN THE 3 MODELS

- As you can see from Table 1 and Table 2 that the unsegmented CNN performed better and had better accuracy compared to that of segmented CNN. This could be because segmentation can sometimes lead to loss of information or addition of noise.

- From Table 2 and Table 3 it is evident that ResNet performed slightly better than Convolutional Neural Network and had better accuracy than segmented CNN. This was expected because ResNet has a more advanced architecture compared to CNN.

7 MODIFICATIONS AFTER PRESENTATION - OVERSAMPLING

After the presentation, we decided to work on our segmented CNN model and try to improve it by completely removing the class imbalance. Hence, we decided to do Oversampling so that all classes have equal images.

OVERSAMPLING

Class imbalance is a major issue which can affect the performance of a deep learning model. This is because the results after classification would be biased and would largely depend on the majority class. Hence, oversampling comes into the picture where such a problem can be handled. There are various ways to perform over-sampling tasks and one of the most widely used methods is to increase the number of samples of minority classes.

After applying CNN and ResNet50 models, over-sampling was done and the number of images from each class was brought to 1000 each and hence the size of dataset was now 7000 images. Below is the graph which shows the number of samples per class after performing over-sampling.

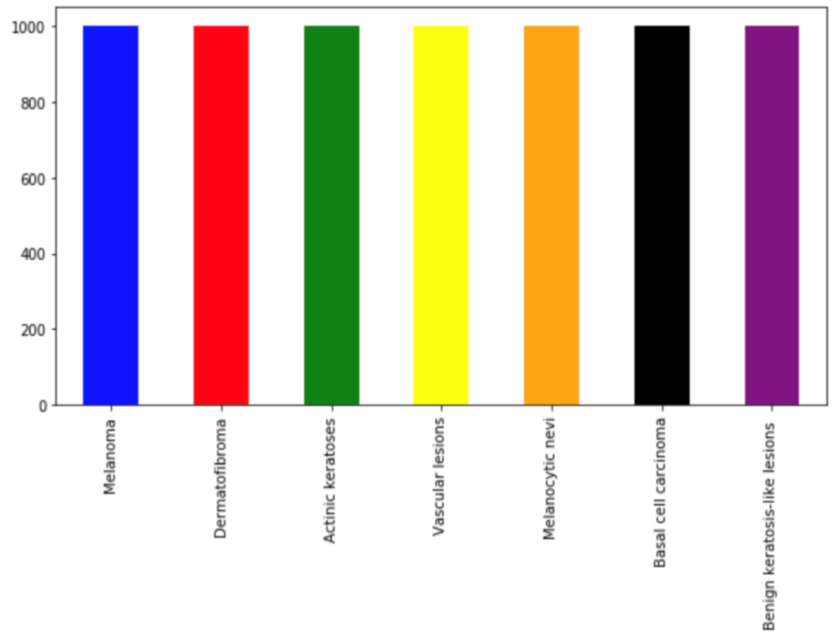


Fig. 25 Class Distribution after Oversampling

We trained the model with this oversampled dataset and the following results were obtained:

- **Training Results:**

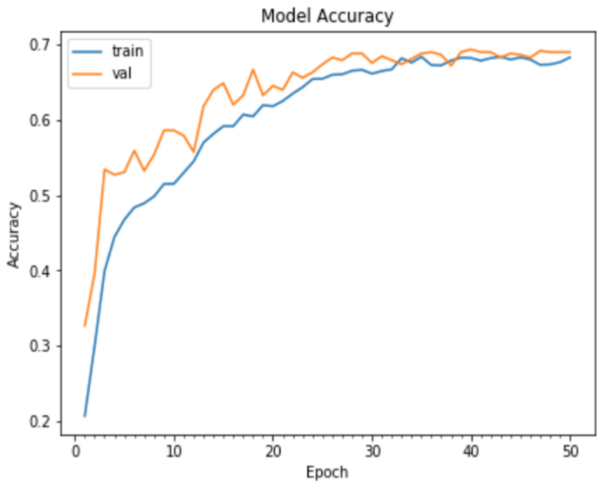


Fig 26a Training Accuracy

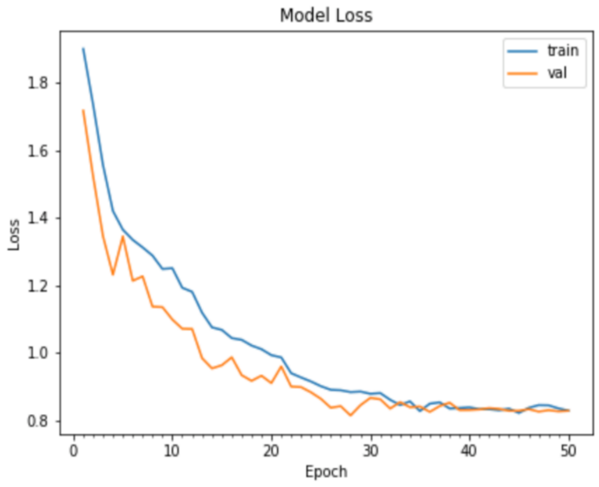


Fig 26b Training Loss

Figure 26(a) represents the accuracy achieved during the training process. You can see that the training accuracy reached up to 69% and validation accuracy was 68%.
From figure 26(b) shows the Loss vs Epochs curve and it shows that training as well as validation loss decreases with each epoch and finally reaches its minimum value.

- **Test Results:**

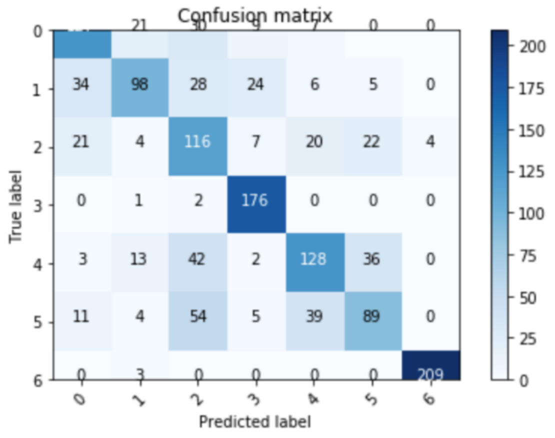


Fig. 27 Confusion Matrix for oversampled segmented CNN

Figure 27 shows the confusion matrix. The x-axis has the predicted class and the y-axis has the actual class. The number in the boxes indicate how many images were predicted. The higher the number the darker the colour. The diagonal elements represent True Positives and it shows that the model performed well for all the classes, unlike the previous models.

CLASS	PRECISION	RECALL	F1-SCORE
0	0.65	0.65	0.65
1	0.68	0.50	0.58
2	0.43	0.60	0.50
3	0.79	0.98	0.88
4	0.64	0.57	0.60
5	0.59	0.44	0.50
6	0.98	0.99	0.98
ACCURACY		**67 %**	

Table 4: Test Accuracy for oversampled CNN

Table 4 shows the evaluation metrics achieved for each class. We achieved a test accuracy of 67% for this model.

We can clearly see from Tables 1, 2, 3 and 4 that after oversampling the model performed the best and gave much better predictions for each class. This shows that class imbalance has a negative impact on model training and learning.

The performance of models in ascending order is as follows:

Segmented CNN **<** Segmented ResNet **<** Unsegmented CNN **<** Oversampled segmented CNN

8 CONCLUSION AND FUTURE SCOPE

8.1 CONCLUSION

This project was intended to analyze the dermatoscopic images of pigmented lesions and classify the results into seven different diagnostic categories of skin cancer which will help the medical community. Different methods were tried to achieve this target. A thorough analysis of data followed by proper data pre-processing was done which included data cleaning, removing of class imbalance, image segmentation, feature engineering, train/test/validation split, normalization and one-hot encoding.
Convolutional Neural Network was mainly used for image classification. Also, Image segmentation was performed which removed the redundant parts of the image. Our approach was to train the CNN model using original images as well as segmented images. Segmented images were also trained on RESNET50 model which is an extension of CNN. Results from all the three approaches were compared for the final result. After a thorough comparison, it was concluded that original images performed better when trained on deep learning models. This is

possible due to some loss of important information during the segmentation process. Moreover, it has been concluded that even after reducing the dataset from 10000 to around 4000 images, still the problem of class imbalance persists which results in reduced accuracy.
In order to encounter that we have used oversampling, by increasing the samples of classes which have low samples and it has provided the best results.

8.2 FUTURE SCOPE

During image segmentation, we have used thresholding technique, but currently, there are efficient alternate ways to perform image segmentation. Deep learning is one of the modern technologies being used for the same. U-Net, Masked R-CNN and YOLO (You Only Look Once) with Grabcut are few Deep learning models which can be used to segment the cancerous part of the image. Since images also contain hair which adds unwanted information (noise) to the image, several techniques can be used to remove hairs from the images before segmentation.

Morden application of our project can be in the field of digital dermatoscopy. Smartphones are a vital part of the modern lifestyle, already there is a lot of research going on smartphone-based dermatoscopy, where a patient can use their mobile digital camera to generate a dermatoscopic image. Same technology and results from our project can be combined in the form of an application, where on front-end patient will take the required picture and, in the back end it will be automatically classified and then result notification will be sent to patient and doctor as well. This will help in diagnosing cancer on time.

9 REFERENCES

[1] https://www.canada.ca/en/public-health/services/sun-safety/skin-cancer.html
[2] https://www.canadianskincancerfoundation.com/about/
[3] ISIC - https://www.isic-archive.com/#!/topWithHeader/tightContentTop/about/isicArchive
[4] Akiec - https://www.mayoclinic.org/diseases-conditions/actinic-keratosis/symptoms-causes/syc-20354969
[5] BCC - https://www.mayoclinic.org/diseases-conditions/basal-cell-carcinoma/symptoms-causes/syc-20354187
[6] BKL - https://www.webmd.com/skin-problems-and-treatments/picture-of-seborrheic-keratosis
[7] DF - https://www.medicalnewstoday.com/articles/318870.php
[8] Mel - https://www.cancer.ca/en/cancer-information/cancer-type/skin-melanoma/melanoma/?region=on
[9] NV - https://emedicine.medscape.com/article/1058445-overview
[10] Vasc - https://www.merckmanuals.com/en-ca/professional/dermatologic-disorders/benign-skin-tumors,-growths,-and-vascular-lesions/vascular-lesions-of-the-skin
[11] https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53
[12] https://pytorch.org/docs/stable/nn.html#conv2d
[13] https://pytorch.org/docs/stable/nn.html#maxpool2d
[14] https://towardsdatascience.com/understanding-and-coding-a-resnet-in-keras-446d7ff84d33
[15] https://towardsdatascience.com/an-overview-of-resnet-and-its-variants-5281e2f56035
[16] https://towardsdatascience.com/an-overview-of-resnet-and-its-variants-5281e2f56035
[17] https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53
[18] https://www.jeremyjordan.me/convnet-architectures/
[19] https://towardsdatascience.com/understanding-and-coding-a-resnet-in-keras-446d7ff84d33
[20] Course lecture slides https://owl.uwo.ca/portal
[21] https://gombru.github.io/2018/05/23/cross_entropy_loss/

Also Followed:

- https://github.com/adriaromero/BSc_Thesis_Skin_Lesion_CNN
- https://www.mdpi.com/2075-4418/9/3/72/pdf