Skin Cancer Detection

Using Deep Convolutional Neural Networks

ISIC 2018

By

Avneet Kaur

Submitted To

Dr. Quynh (Monica) Nguyen

Contents

Abstract

Skin Cancer is the most common form of cancer that occurs due to an abnormal growth of skin cells. It is usually developed on skin areas that are exposed to the Sun. One in five Americans develop skin cancer by the age of 70 and having more than 5 sunburns can double the risk for melanoma, a type of skin cancer[1]. If detected early, skin cancers can be treated effectively thereby increasing the survival rate. However, the process of detecting skin cancer by an expert dermatologist includes various examination tests from clinical screening to biopsy, which can be time consuming and challenging. Therefore, significant research has been conducted and many are in progress to automate the crucial task of classifying skin lesions into benign, cancerous, or precancerous states. In this work, we have developed a model for skin lesion classification by implementing Image Augmentation, hair removal, cropping, Deep Convolutional Neural Networks (DCNN) and Ensemble Learning. The model is trained and validated on ISIC 2018 dataset consisting of 10,015 images belonging to seven different classes of skin lesions. Model achieved an overall validation accuracy of 96% on randomly generated test dataset and 88% accuracy on test dataset (generated from ISIC 2019 training dataset)

Introduction

Skin cancer is a major public health problem, with over 5,000,000 newly diagnosed cases in the United States every year The three major types of skin cancers are Basal Cell Carcinoma, Melanoma, and Squamous Cell Carcinoma Melanoma is the deadliest form of skin cancer, responsible for an overwhelming majority of skin cancer deaths. Although the mortality is significant, when detected early, melanoma survival exceeds 95% [2]. Medical imaging such as Optical Coherence Tomography, Confocal Scanning Laser Microscopy, Magnetic Resonance Imaging and Dermoscopic Imaging are used in skin cancer diagnosis wherein lesion images are inspected by dermatologists visually [3]. CAD(Computer-Aided Diagnosis) systems are being used to relieve this time consuming & tedious process as well as improve the precision of diagnosis [4]. These systems are utilized for pre-processing, capturing region of Interest, and extracting features. Since the images contain artifacts like excessive hair, dark borders, ink marks etc., have diverse size, texture and skin colour which prevent accurate recognition of type of lesion, the steps of pre-processing are essential to be able to classify the images accurately [5]. Once images get processed, the final task is to perform SLC(Skin Lesion Classification) using appropriate techniques that can optimally identify the type of skin lesion based on various features of lesions such as colour, shape, size, aperture etc.

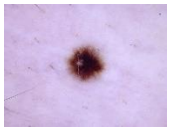
This report discusses various research and implementations performed in this field using techniques developed over time in the field of machine learning. Deep CNN based classifiers along with dense neural networks is the most popular and effectives approach that has proved in demonstrating good accuracy of classification and avoiding tedious process of feature extraction. Based on research, we have integrated techniques of Deep CNN, image augmentation and ensemble learning to train models on ISIC 2018 competition dataset consisting of 7 different classes of skin lesion images. Due to unbalanced dataset, several image augmentation techniques are applied to balance the number of images in each class. We applied various techniques to process these images by cropping, removing artifacts and fed them into different CNN models. The study discusses various models that were tried building up to an Ensemble of most suitable model. We used Adam optimizer method to optimize our model over 30 epochs based on accuracy and categorical cross entropy loss.

Dataset

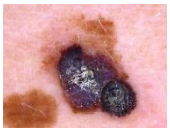
The ISIC – 2018 training dataset consists of 10,015 dermatoscopic images of skin lesions belonging to different cases of skin cancers. The ground truth of these images is stored in another dataset which consists mapping of these images to their corresponding labels(skin lesion class). There are 7 cases namely :

CLASS	NAME	COUNT
NV	Melanocytic Nevus	6705
MEL	Melanoma	1113
BKL	Benign Keratosis Lesion	1099
BCC	Basal Cell Carcinoma	514
AKIEC	Actinic Keratosis	327
VASC	Vascular Lesions	142
DF	Dermatofibroma	115
	TOTAL	***10015***

- ***Melanocytic Nevus (NV)***
  Melanocytic Nevus are benign form of skin lesions and commonly known as moles. They are usually brown, pink, tan or black coloured. They are circular or oval shaped and are usually small sized. However, some of them can be of larger sizes and produce dark, coarse hair.



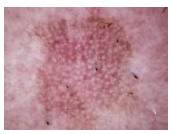
- ***Melanoma (MEL)***
  Melanoma is the deadliest form of skin cancer and frequently develops in a mole or suddenly appears as a new dark spot on the skin.



- ***Benign Keratosis Lesions (BKL)***
  BKL are non-cancerous skin lesions that many people tend to develop as they age older. They are usually black, brown, or light tan in colour with waxy or scaly texture.

- ***Bassal Cell Carcinoma (BCC)***
  BCC is the most common type of skin cancer. It often looks like a flesh-coloured, pearl-like bump, or pinkish patch of skin. It is most frequently seen in people who have fair skin.

- ***Actinic Keratosis (AKIEC)***
  AKIEC are rough, scaly patches on the skin that develop from years of sun exposure. They are often found on the face, lips, ears, forearms, scalp, neck or back of the hands. If left untreated they can turn into squamous cell carcinoma skin cancer.



- ***Vascular Lesions (VASC)***
  Vascular lesion is an umbrella term for different varieties of skin irregularities such as Cherry Angiomas, Vascular Birthmarks, Port-Wine Stains, Vascular malformations etc., that result from an over production of very small capillaries at, or on top of, the skin surface. According to our observation, our dataset consists of mainly capillary malformations , a type of Vascular malformation, that are of pink or deep purple colour.

- ***Dermatofibromas (DF)***
  DF are small, benign skin growths that are most commonly found on the lower legs. Characteristics of dermatofibromas include size of around 2-3 mm, purplish brown colour, and hard structure. They are usually painful when pressed.

Proposed Methodology

DCNN i.e., Deep Convolutional Neural Networks are the most effective and well-recognized technique for image classification currently. Although, we explored some DCNN models developed by companies like Google and researchers based on ImageNet dataset by training them on ISIC 2018 Skin Lesion Image dataset to verify whether they provide expected accuracy. However, as described above they could not perform well on this dataset. Based on study of various research papers & related work, we have developed an Ensemble of 10 Deep CNN based on a custom DCNN base model.

The dataset as described above is highly imbalanced in terms of number of images in each class. Moreover, the images contain unwanted artifacts such as hair, ink marks, ruler marks etc., that can result in improper training of machine learning model. Hence, we have applied a variety of pre-processing technique on the images to make them consumable by the model. The benefit of these techniques is two-fold. Firstly, they make the images clean such that they reflect the actual features of the lesions. Secondly, it eliminates the problem of class imbalance thus making training more robust.

Pre-Processing

We applied the following techniques to enhance the quality of training data for model training by removing undesirable noise and improving the features' visibility of images. It dramatically improves the model performance.

Stages of pre-processing:

Stage 1: Mapping - *Organize images for efficient importing*

As we implemented our model using python keras library, we utilized ImageDataGenerator class that accepts images in a specific directory structure wherein the parent directory contains as many directories as there are classes (in this case 7). Each of these sub-directories contain images of corresponding class. With the help of python script, all the 10,015 training set images (that were

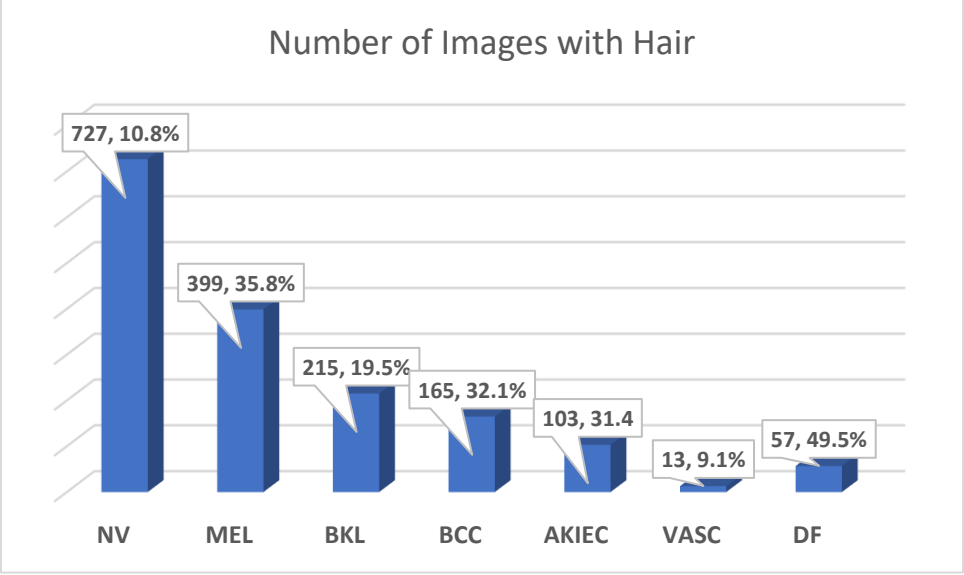
obtained in a single folder from source) were organized into respective directories based on mappings available in the ground-truth csv dataset.

Stage 2 : Digital Hair Removal - *Eliminate noise that might be learned as a feature*

A major obstacle for creating a robust model for a skin lesion classification is presence of hair especially dark coloured hair. DCNN models develop feature maps by applying various filters on image , and thus, can learn hair as a feature of skin lesion. This adversely affects the model performance. Hence, it becomes essential to eliminate any such noise. We have employed the below mentioned approach to tackle this problem

- *Manual Selection*:
  As any kind of pre-processing/filter reduces the quality of image, applying it to all images for removing hair is not optimal. Although we attempted to identify patterns of the presence of hair in different classes, it turned out that not all the classes have significant number of images with hair.
  However, even if a class has a smaller number of images with hair, we cannot ignore them. Since, we have performed augmentations(discussed later), those images can increase. Hence, generalizing classes with hair or without hair might result in degrading quality of non-noisy images and leaving some images with hair, unprocessed, thereby affecting the accuracy.

  Therefore, we manually looked at all the 10,015 images and annotated the images with dark coloured hair for hair removal. The count of images with hair selected from each class is shown below:



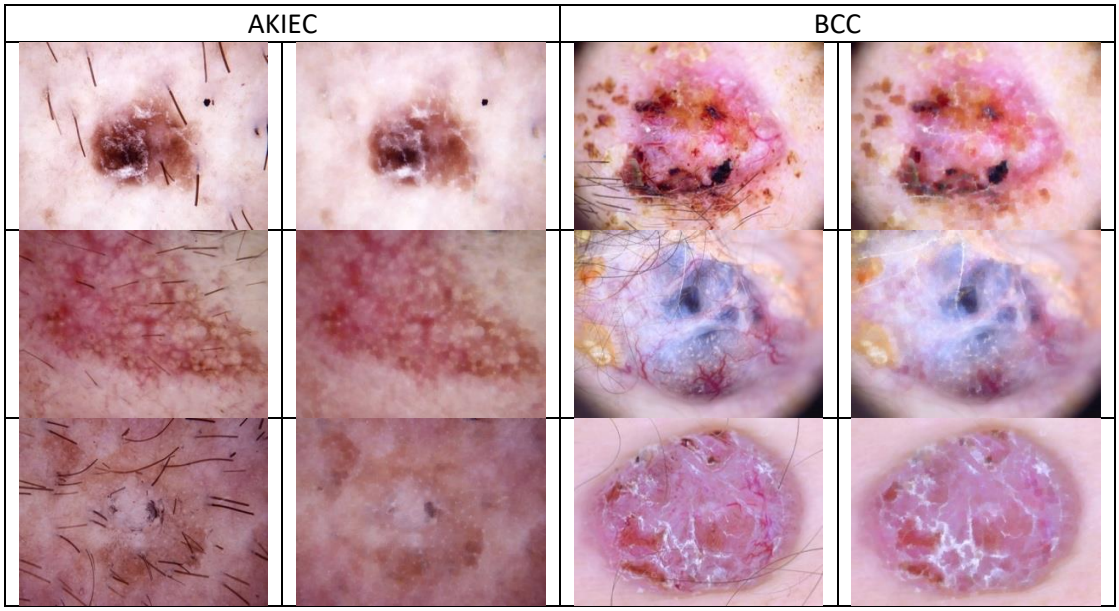
  Hence, 16% of images are considered for hair removal processing.

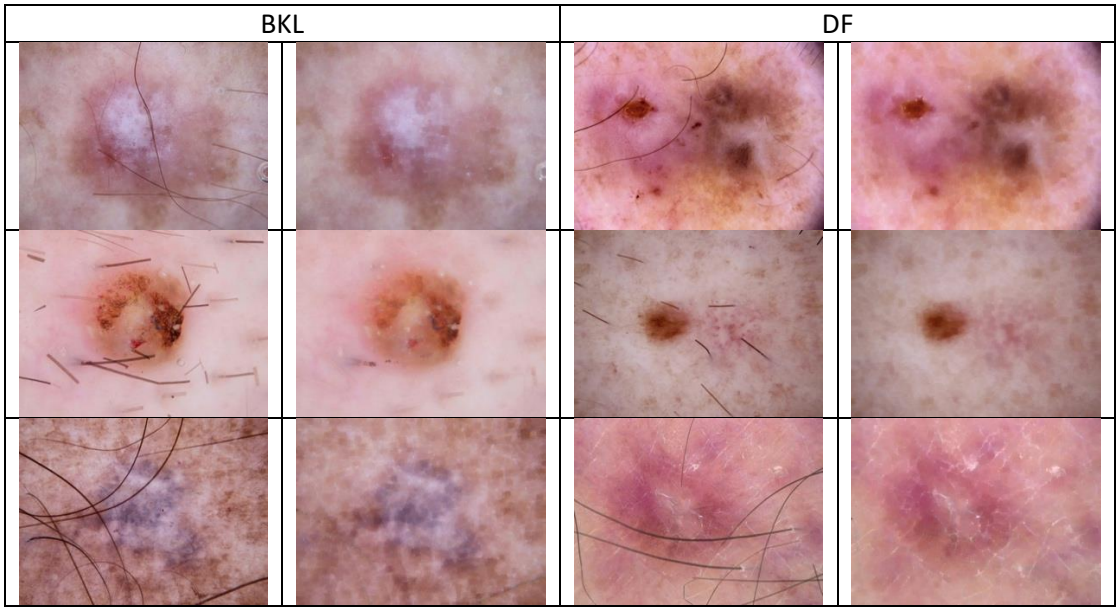
- *DullRazor [6]*:
  We have employed Dullrazor, a well-known mechanism for removing hair from dermatoscopic images. It performs better especially on dark thick hair as they are the main culprit that affect

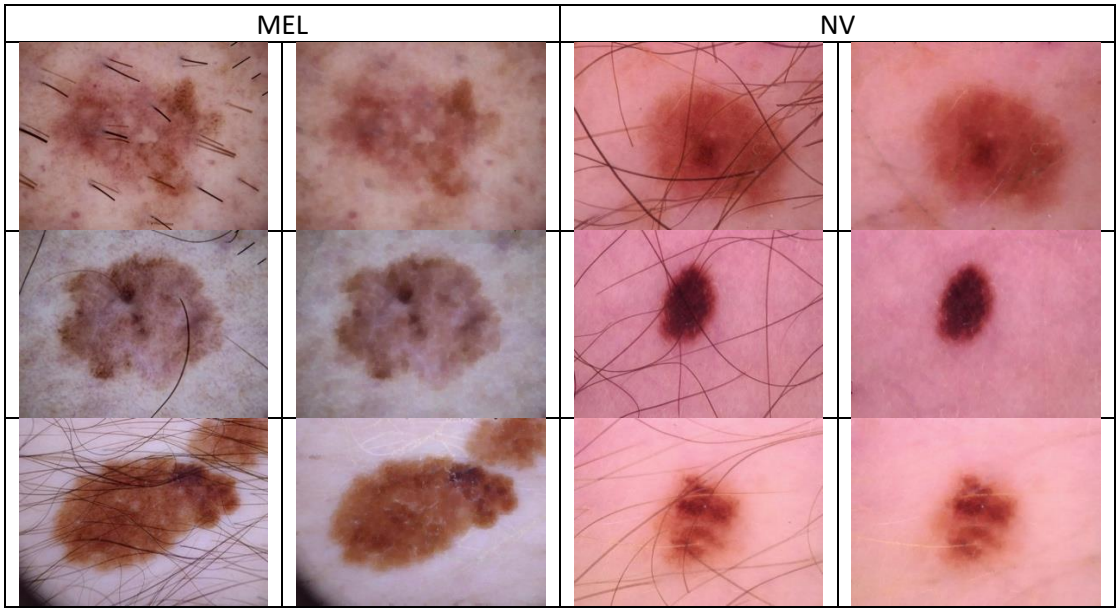
the CNN model resulting in unsatisfactory results. The Dullrazor algorithm and its working are described briefly :

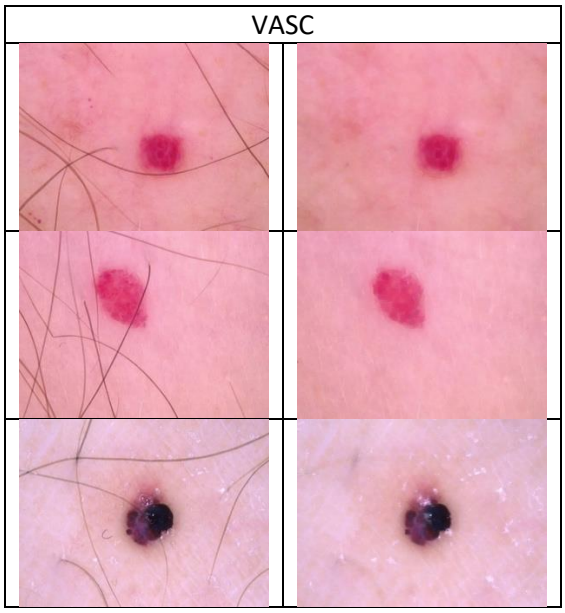
- The hair regions are initially detected through the morphological closing operator on each RGB colour channel separately and with three structuring elements having different directions .
- To generate the binary mask, a thresholding process is applied to the absolute difference between the original colour channel and the image generated by the closing .
- The mask pixels undergo a bilinear interpolation between two nearby not-mask pixels.
- Finally, to the resulting image, an adaptive median filter is applied.

We have implemented this algorithm in Python using OPENCV library. Below are the unprocessed & processed images:




BKL
DF


MEL
NV



Note : Dull razor algorithm does not work efficiently with white or light-coloured hair. But they are automatically handled by the CNN and are not as much a hinderance as compared to dark thick hair.

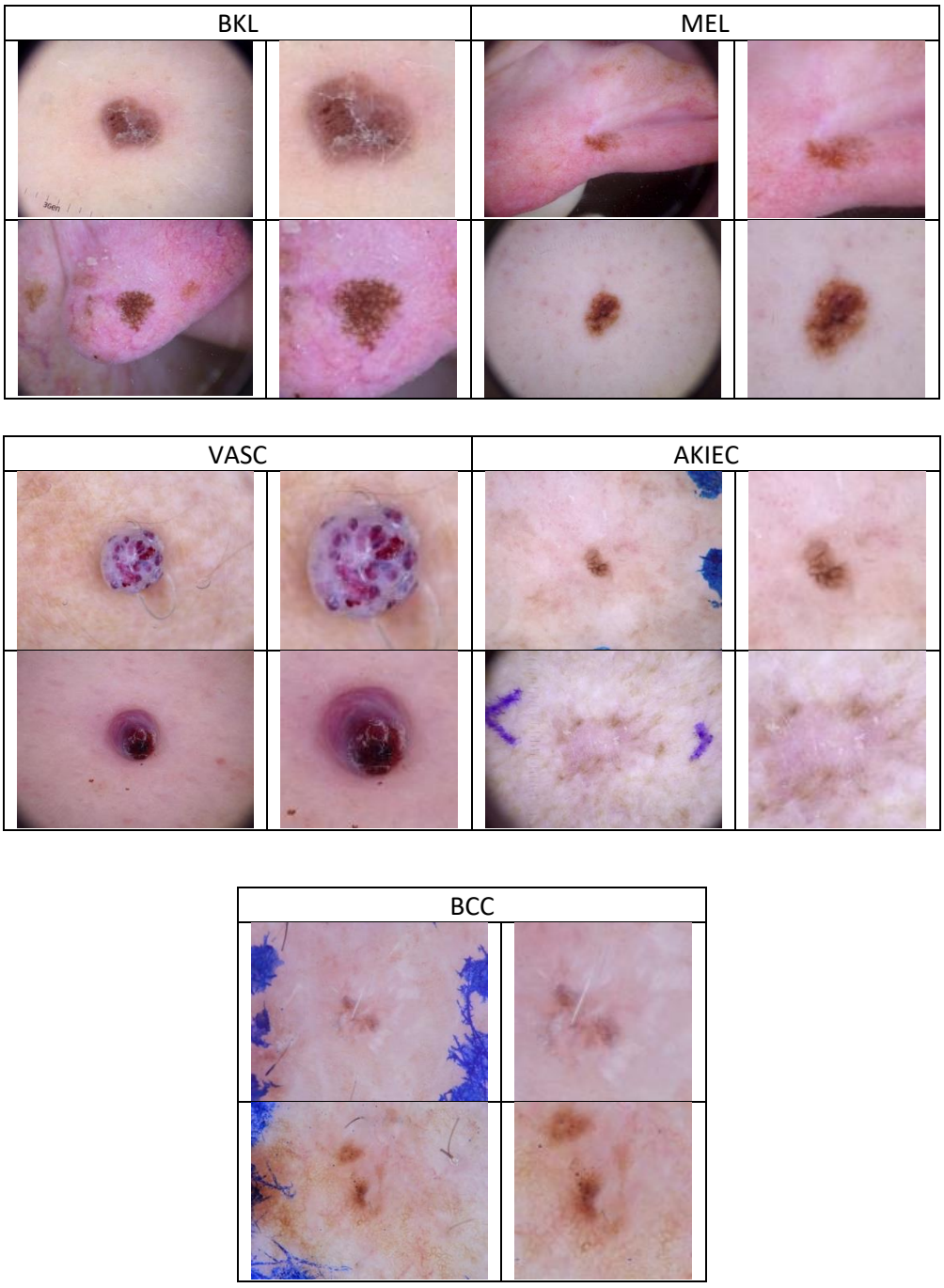
Stage 3 : Resizing - *Reduces time taken at each epoch of training CNN model*

Resizing images is a critical step in Image pre-processing. CNN models run faster on smaller images due to less number of pixels and thus, less data points. In our dataset all the images have size of 600 x 490 which results in ~300K pixel data points. Therefore, prior to any other image processing, we have resized all the images to 128 x 128 (~16K pixel data points) reducing amount of data by ~94%. Moreover, python library used for the project i.e., TensorFlow expects a square dimension.

Step 4 : Image Cropping - *Trim unwanted areas of image (dark corners, ruler & ink marks, etc.)*

The dataset consists of some images that have dark borders (called Vignette effect) and Ink/Ruler marks that lowers the training efficiency of the model. Hence, we applied centre cropping make the lesion prominent.

We identified the classes **BKL**, **MEL** and **VASC** having such noise in images by manually checking the location of the lesion in the images. According to our observations, we selected such images from each class on which central crop can be applied. We filtered out **124** images from BKL, **194** images from MEL, and **119** images from VASC class. We also applied cropping to 2 images from AKIEC, 2 images from BCC, as they contained ink marks on the sides. Below are some of the unprocessed & processed images:



Stage 5 : Augmentation - *Balance data in each class & increase training data*

Skewed classes, overfitting and training image scarcity were the problems that were solved using Augmentation of images. The NV class dominates with approximately 67% of images in training data as can be seen from the frequency table of classes. Hence, various augmentation techniques were implemented to increase the size of each class and balance the distribution.

The below table summarizes the Augmentations applied on different classes:

CLASS	Actinic Keratosis	Basal Cell Carcinoma	Benign Keratosis	Dermatofibroma	Melanoma	Vascular Lesion	Melanocytic Nevus	
Notation	AKIEC	BCC	BKL	DF	MEL	VASC	NV	Total
Training Images before Aug	228	359	769	80	779	99	4693	7007
Aug_1	ShiftScaleRotate. shift_limit=0.0625 (default), scale_limit=0.2 (default), rotate_limit=(-40,40) degrees	ShiftScaleRotate. shift_limit=0.0625 (default), scale_limit=0.2 (default), rotate_limit=(-40,40) degrees	ShiftScaleRotate. shift_limit=0.0625 (default), scale_limit=0.2 (default), rotate_limit=(-40,40) degrees	ShiftScaleRotate. shift_limit=0.0625 (default), scale_limit=0.2 (default), rotate_limit=(-40,40) degrees	Vertical flip: Flip the input vertically around the x-axis.	ShiftScaleRotate. shift_limit=0.0625 (default), scale_limit=0.2 (default), rotate_limit=(-40,40) degrees	No Augmentations were applied as the training input is 6705 images. 70% of this size i.e near to 4693 images will be chosen for model building	TOTAL
Aug_2	Horizontal flip: Flip the input horizontally around the y-axis	Horizontal flip: Flip the input horizontally around the y-axis	Vertical flip: Flip the input vertically around the x-axis.	Horizontal flip: Flip the input horizontally around the y-axis	Bright Contrast : Adjusts the brightness and contrast of the images, brightness= 0.2, contrast=0.2	Horizontal flip: Flip the input horizontally around the y-axis		
Aug_3	Vertical flip: Flip the input vertically around the x-axis.	Vertical flip: Flip the input vertically around the x-axis.	Rotate, with shear shifting (80 degrees, 0.2) on randomly chosen 1617 images from last augmentation	Vertical flip: Flip the input vertically around the x-axis.	Rotate, with shear shifting (80 degrees, 0.2) on randomly chosen 1577 images from last augmentation	Vertical flip: Flip the input vertically around the x-axis.		
Aug_4	Bright Contrast : Adjusts the brightness and contrast of the images, brightness= 0.2, contrast=0.2	Rotate, with shear shifting (80 degrees, 0.2) on randomly chosen 1821 images from last augmentation		Bright Contrast : Adjusts the brightness and contrast of the images, brightness=0.2 , contrast=0.2		Bright Contrast : Adjusts the brightness and contrast of the images, brightness=0.2 , contrast=0.2		
Aug_5	Scaling of 0.3 on randomly chosen 1045 images from last augmentation			Rotate, without shear shifting (40 degrees)		Rotate, without shear shifting (40 degrees)		
Aug_6				Rotate, with shear shifting (80 degrees, 0.2) on randomly		Rotate, with shear shifting (80 degrees, 0.2) on randomly		

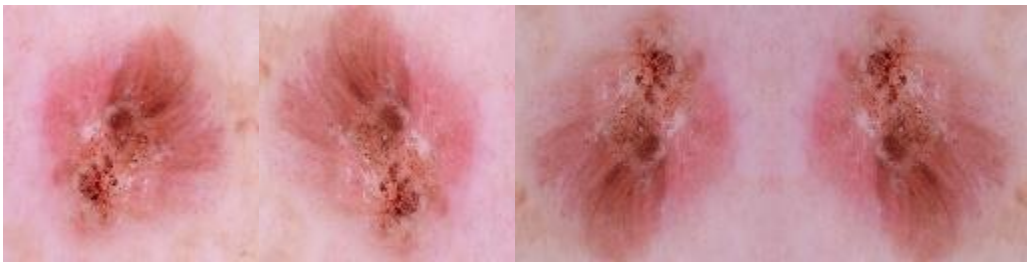
				chosen 2133 images from last augmentation		chosen 1525 images from last augmentation		
Training input after Aug	4693	4693	4693	4625	4693	4641	4693	32731
Validation Data images count	99	155	330	35	334	43	2012	3008

Notes:

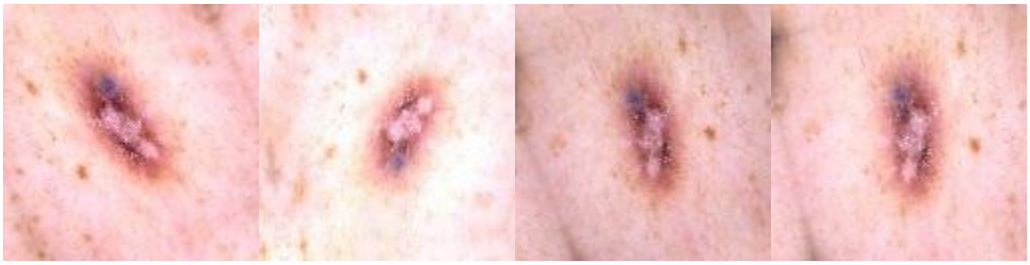
1. Augmentations are applied to match the training size of the biggest class NV, size = 6705. Training input of 70% = 0.7x6705 = 4693(approx). Therefore, target training size range of other classes - [4625, 4693].
2. All augmentations are applied using the 'Albumentations' library.
3. The final resolution of images after all augmentations was maintained to 128 x 128 .
4. Effort was made to make augmented images as natural as possible by choosing the sequence of augmentations functions as well as the augmenting parameters to avoid duplications from repeated augmentations.
5. Augmentations are applied to all the resulting images from the previous augmentation of a class except the last augmentation of DF class wherein it was applied on a selected number of randomly chosen and shuffled images to match the final training size.

Augmentation Results

AKIEC:



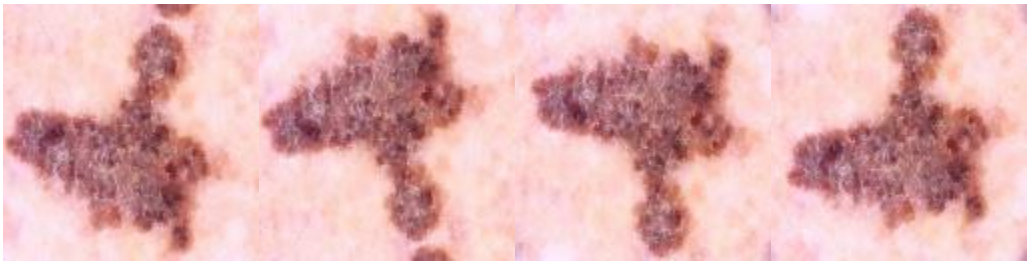
DF:



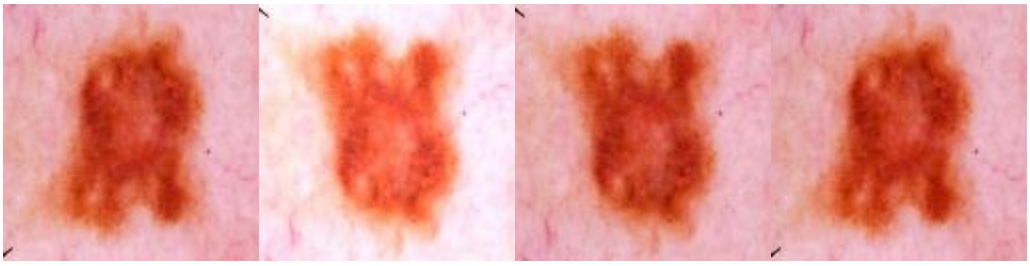
VASC:

BCC:

BKL:



MEL :



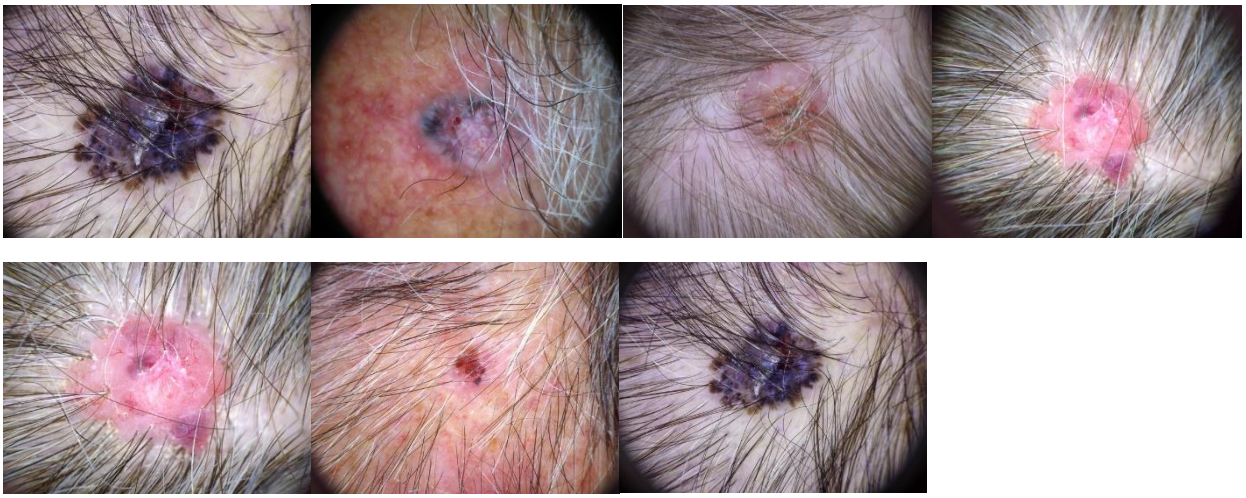
Stage 6 : Normalization - *Ensure each pixel has similar distribution making convergence of CNN faster*

The final step of pre-processing is standardizing each pixel value of image from 0-255 to 0-1 called Normalization. The main objective of performing normalization is to achieve consistency in dynamic range of pixel intensity of images of different skin lesions. Furthermore, it will help CNN model converge faster. Normalization will be applied while developing the model as it comes as a built-in feature of ImageDataGenerator class in Tensorflow library of python.

Apart from the pre-processing we applied to enhance quality of images of dataset, we identified 9 images that were not recoverable in a good quality form due to too much hair. We have deleted these images from the dataset as they were causing reduced accuracy.

Below are the images we removed from different classes:

BCC:



BKL:

MEL:

Experiments

Convolutional Neural Networks (CNNs) are currently the most popular and effective technique for image classification. In this project, we aim to develop a CNN model for classifying Skin Lesion images into the given 7 classes of lesions. There are various CNN models developed by companies like Google and researchers based on ImageNet dataset such as InceptionNet, ResNet etc. We have explored some of these models by training them on ISIC 2018 Skin Lesion Image dataset to verify whether they provide expected accuracy. Moreover, we have developed custom models based on study of various research papers. In this section, we have described all the attempts made to develop a robust CNN model for classification.

We did extensive research to understand the intuition behind making a robust CNN model especially for Skin Lesion classification. Most of the research work we reviewed were performing binary classification to detect whether a Lesion is Melanoma or not. These models achieved good accuracy due to binary nature of classification. Some of the research work was for all the 7 classes of ISIC dataset, however, the dataset used to train was a small subset of 10k images that are available in ISIC archives. Although the models developed in these papers achieved good accuracy, they were not generalized model since the amount of data used to train was less.

As we developed an idea about the purpose of different convolution layers, number of filters, pooling and dropout, we were able to tweak proposed and pre-implemented models to train them for our ISIC 2018 dataset. Below is a detailed description of different attempts made to develop an efficient model along with limitations of each model.

1. Proposed Model by researchers at Telkom University, Bandung, Indonesia

Model Description:

CNN model proposed by researchers at Telkom University was first utilized to develop a CNN model for ISIC 2018 images. The researchers developed model for 4 different classes of images while for our dataset, we tweaked the model to work for 7 classes of output layer. We used this model as a base reference because it was used for skin lesion classification & achieved good accuracy. Also, it is a simpler model for implementation and understanding.[7]

Model Architecture:

The following is the layer-by-layer description:

Layer (type)	Output Shape	Param #
conv2d_40 (Conv2D)	(None, 128, 128, 16)	448
average_pooling2d_10 (AveragePooling2D)	(None, 64, 64, 16)	0
conv2d_41 (Conv2D)	(None, 64, 64, 32)	4640
max_pooling2d_30 (MaxPooling2D)	(None, 32, 32, 32)	0
conv2d_42 (Conv2D)	(None, 32, 32, 64)	18496
max_pooling2d_31 (MaxPooling2D)	(None, 16, 16, 64)	0
conv2d_43 (Conv2D)	(None, 16, 16, 128)	73856
max_pooling2d_32 (MaxPooling2D)	(None, 8, 8, 128)	0
dropout_10 (Dropout)	(None, 8, 8, 128)	0
flatten_10 (Flatten)	(None, 8192)	0
dense_10 (Dense)	(None, 7)	57351

Total params: 154,791
Trainable params: 154,791
Non-trainable params: 0

- CN Layers : x3, Kernel: 3x3
  - Layer 1: Filters x16
  - Layer 2: Filters x32
  - Layer 3:Filters x64
- Max Pooling Layers: 2x2
- Dropout : 50%
- Fully Connected Layer : x1
- Activation:
  - CNN: RELU
  - Output: SoftMax
- Optimizer: ADAM
- Padding: SAME
- Loss Metric: Categorical Cross Entropy
- Accuracy Metric: Accuracy

Results:

Confusion Matrix

```
[[  44   18    3    7    7   20    0]
 [  25   94    5    6    1   21    1]
 [  25   13  151   11   33   97    0]
 [   2    2    0   22    2    7    0]
 [  25   10   39    6  154   98    2]
 [  21   34   76   30  105 1738    8]
 [   0    1    0    0    1    1   40]]
```

Classification Report

	precision	recall	f1-score	support
akiec	0.31	0.44	0.37	99
bcc	0.55	0.61	0.58	153
bkl	0.55	0.46	0.50	330
df	0.27	0.63	0.38	35
mel	0.51	0.46	0.48	334
nv	0.88	0.86	0.87	2012
vasc	0.78	0.93	0.85	43
accuracy			0.75	3006
macro avg	0.55	0.63	0.57	3006
weighted avg	0.76	0.75	0.75	3006

- Number of Epochs : 50
- Overall Accuracy: **75%**

2. Modified Proposed Model

Model Description:

Since the above CNN model didn't achieve good accuracy and was a basic model with few layers of convolution and no ANN layer, we enhanced the model by adding more layers of Convolution and ANN dense layers. This dramatically increased the number of trainable parameters. The intention was to enable model to learn more from images. Also, the accuracy achieved for NV and VASC classes was higher as compared to others. Thus, we developed model to tune hyperparameters only for rest of the 5.

Model Architecture:

The following is the layer-by-layer description:

Layer (type)	Output Shape	Param #
conv2d_27 (Conv2D)	(None, 128, 128, 32)	896
conv2d_28 (Conv2D)	(None, 128, 128, 32)	9248
max_pooling2d_18 (MaxPooling2D)	(None, 64, 64, 32)	0
conv2d_29 (Conv2D)	(None, 64, 64, 64)	18496
conv2d_30 (Conv2D)	(None, 64, 64, 64)	36928
max_pooling2d_19 (MaxPooling2D)	(None, 32, 32, 64)	0
conv2d_31 (Conv2D)	(None, 32, 32, 128)	73856
conv2d_32 (Conv2D)	(None, 32, 32, 128)	147584
max_pooling2d_20 (MaxPooling2D)	(None, 16, 16, 128)	0
flatten_6 (Flatten)	(None, 32768)	0
dense_16 (Dense)	(None, 512)	16777728
dense_17 (Dense)	(None, 512)	262656
dense_18 (Dense)	(None, 512)	262656
dropout_11 (Dropout)	(None, 512)	0
dense_19 (Dense)	(None, 5)	2565

Total params: 17,592,613
Trainable params: 17,592,613
Non-trainable params: 0

- CN Layers : x6, Kernel: 3x3
  - Layer 1,2 : Filters x32
  - Layer 3,4 : Filters x64
  - Layer 5,6 : Filters x128
- Max Pooling Layers: 2x2
- Dropout : 50%
- Fully Connected Layer : x4
  - 3 Layers: 512 Nodes
  - Output Layer: 5 nodes
- Activation:
  - CNN: RELU
  - Dense: RELU
  - Output: Softmax
- Optimizer: ADAM
  - Learning rate = 0.001
- Padding: SAME

Results:

The results below show that the model accuracy is too low. Also, it started decreasing after 10 Epochs, so we stopped the training. It can be observed from epochs that we achieved the highest validation accuracy of 36.8%

```
Epoch 3/20
365/365 [==============================] - 93s 256ms/step - loss: 1.6096 - accuracy: 0.1962 - val_loss: 1.6072 - val_ac
curacy: 0.1708
Epoch 4/20
365/365 [==============================] - 94s 257ms/step - loss: 1.6096 - accuracy: 0.1995 - val_loss: 1.6039 - val_ac
curacy: 0.1708
Epoch 5/20
365/365 [==============================] - 93s 254ms/step - loss: 1.6096 - accuracy: 0.1991 - val_loss: 1.6040 - val_ac
curacy: 0.3114
Epoch 6/20
365/365 [==============================] - 92s 252ms/step - loss: 1.6096 - accuracy: 0.1955 - val_loss: 1.6076 - val_ac
curacy: 0.1708
Epoch 7/20
365/365 [==============================] - 94s 258ms/step - loss: 1.6096 - accuracy: 0.1975 - val_loss: 1.6047 - val_ac
curacy: 0.1708
Epoch 8/20
365/365 [==============================] - 93s 254ms/step - loss: 1.6096 - accuracy: 0.1983 - val_loss: 1.6100 - val_ac
curacy: 0.1708
Epoch 9/20
365/365 [==============================] - 92s 251ms/step - loss: 1.6095 - accuracy: 0.1992 - val_loss: 1.6072 - val_ac
curacy: 0.3683
Epoch 10/20
365/365 [==============================] - 92s 252ms/step - loss: 1.6096 - accuracy: 0.1913 - val_loss: 1.6078 - val_ac
curacy: 0.1105
```

3. RESNET50 Model

Model Description:

RESNET50 is one of the most popular models used for image classification. It is a model which is originally trained on ImageNet dataset where it achieved 92.1% Top-5 accuracy. We utilized its architecture and trained it using ISIC dataset. We didn't use the pre-trained weights of ImageNet since they are not relevant to Skin Lesion images.

Model Architecture:

The model architecture is huge as it contains 50 DCNN layers with Max Pool, Average Pool and Batch Normalization layers etc.

Results:

```
Confusion Matrix
[[  76    4   10    0    3    5    1]
 [  57   51   15    2    3   24    1]
 [  53    4  181    0   23   69    0]
 [   8    1    1   15    3    7    0]
 [  35   16   83    0  125   75    0]
 [ 173   66  327    7  202 1220   17]
 [   0    2    1    0    5    1   34]]
Classification Report
              precision    recall  f1-score   support

       akiec       0.19      0.77      0.30        99
         bcc       0.35      0.33      0.34       153
         bkl       0.29      0.55      0.38       330
          df       0.62      0.43      0.51        35
         mel       0.34      0.37      0.36       334
          nv       0.87      0.61      0.71      2012
        vasc       0.64      0.79      0.71        43

    accuracy                           0.57      3006
   macro avg       0.47      0.55      0.47      3006
weighted avg       0.69      0.57      0.60      3006
```

- Number of Epochs = 20
- Overall Accuracy: **57 %**

4. InceptionV3

Model Description:

Like RESNET50, InceptionV3 is another popular model developed by GoogleLeNet used in Transfer learning. Various research papers described the usage of this model for skin lesion multiclass classification with sufficiently good accuracies.

Model Architecture:

The model architecture is huge and is 189 layers deep. We did not use pre trained weights from ImageNet Datasets. Instead, we trained the model based on our ISIC 2018 dataset.

Results:

```
Confusion Matrix
[[ 20   1   1   0   0  77   0]
 [  2  12   1   0   0 138   0]
 [  5   0  26   2   8 288   1]
 [  1   0   0   9   0  25   0]
 [  2   0  16   0  48 268   0]
 [  0   1   8   1   9 1990  3]
 [  0   0   0   1   3   4  35]]
Classification Report
              precision    recall  f1-score   support

       akiec       0.67      0.20      0.31        99
         bcc       0.86      0.08      0.14       153
         bkl       0.50      0.08      0.14       330
          df       0.69      0.26      0.37        35
         mel       0.71      0.14      0.24       334
          nv       0.71      0.99      0.83      2012
        vasc       0.90      0.81      0.85        43

    accuracy                           0.71      3006
   macro avg       0.72      0.37      0.41      3006
weighted avg       0.70      0.71      0.63      3006
```

- Number of Epochs = 20
- Overall Accuracy = **71%**

Ensemble of Convolutional Neural Network

Ensemble Learning is a boosting technique often used in machine learning to improve the robustness, efficiency and accuracy of models. In this project, we have developed a custom ensemble of DCNN model experimented initially as it provided good performance. We chose to take 10 models in our ensemble as we did not want to overfit the model and make it complex.

We tried 3 different approaches to Ensemble Budling eventually discovering the best ensemble with high accuracy.

Ensemble Development Technique

Step 1: Base CNN Model
Selected a Base DCNN Model that will be used to develop ensemble.

Step 2: Develop a Collection of Compiled Models
Created a **List** of 10 DCNN Base Models and compiled them using appropriate metrics. This is the Ensemble Model, a collection of base models compiled with same hyperparameters

Step 3: Set Hyperparameters for Model Development

- **TRAIN DIRECTORY** : The directory on system containing pre-processed images
- **Epochs**: The number of epochs to run for training the ensemble. We chose this value as 50 based on our research review where most of the CNN models were trained for at least 30 epochs. This helps in better trained model development when the number of images is huge and variation among them is significant.
- **Batch Size**: The model is trained on batches of images. This is the number of images used to train a single forward & backward pass(epoch). If this value is too high, it can make the CNN take too long to achieve convergence(no more gain in accuracy). However, if it is too low, it will take more time for accuracy to stabilize (the accuracy will bounce up and down in subsequent epochs.). Therefore, based on several attempts, we fixed the value to be 64 for training the network better.

Step 4: Pre-process Data & Train Each Model of Ensemble:

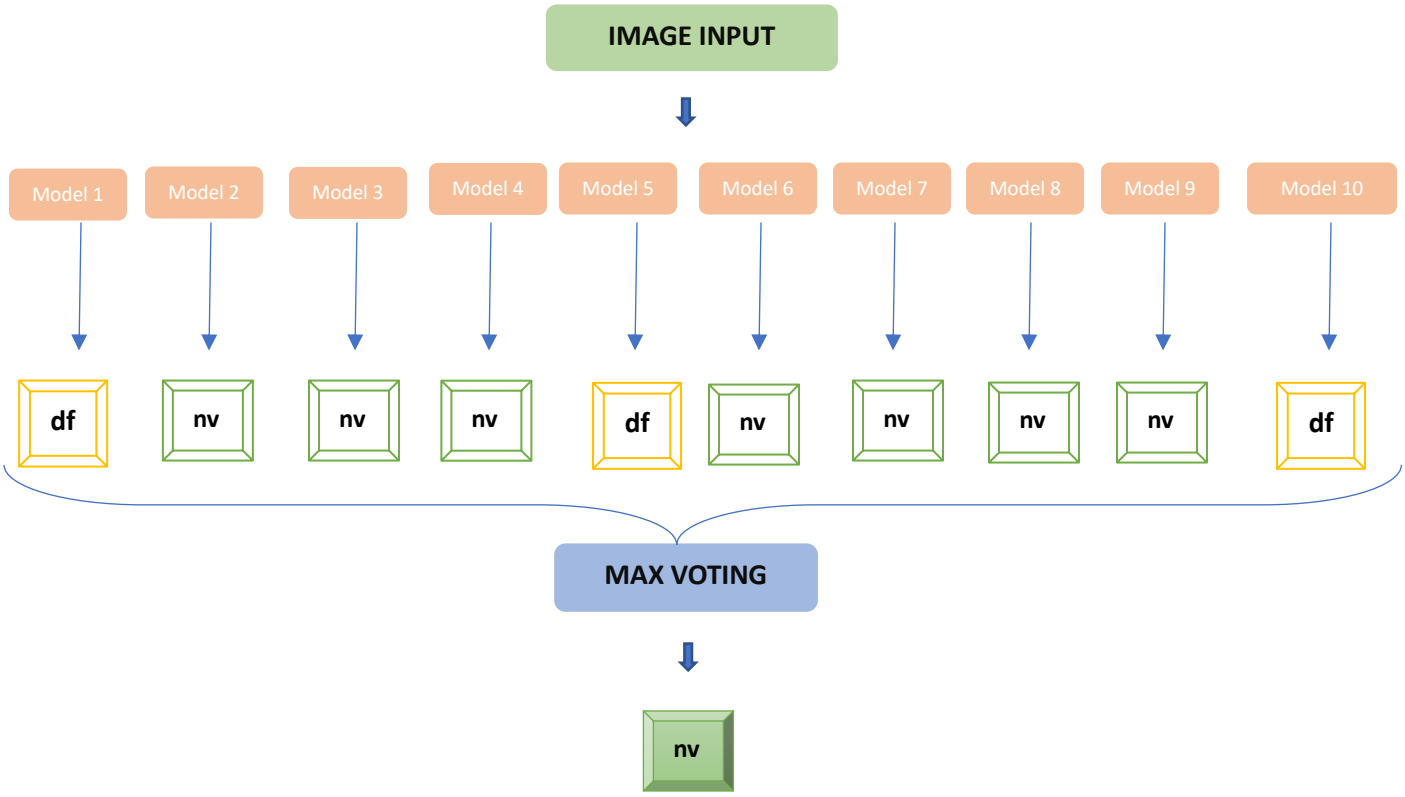
The following steps were applied in each iteration of for loop that run for 10 iterations training a distinct model in each:

i. Pick a random seed value which is different in every iteration
ii. Shuffle the pre-processed images using this random seed value such that different set of images are picked in every iteration.
iii. Split the data with 70% images as training set and 30% as validation set.
iv. Perform Augmentations with an objective to balance the images in each class. The functions to augment images are separately written based on the number of images for each class.
v. Keep the augmented training images and validation images in separate directories with following structure:

```
augmented|                    test |
         |-akiec                   |-akiec
         |-bcc                     |-bcc
         |-bkl                     |-bkl
         |-df                      |-df
         |-mel                     |-mel
         |-vasc                    |-vasc
         |-nv                      |-nv
```

vi. Use ImageDataGenerator class of keras module in python to develop an efficient structure to provide training and validation data for model training. It also scales the images so that pixels are in range of 0 to 1 instead of 0 to 255.
vii. Train the model using the hyperparameters set above and save the model in the list 'ensemble'
viii. Delete the augmented and validation images at end of each iteration for new set to be generated in the next iteration.

The working of the ensemble prediction is explained below:



Ensemble Models

The ensembles were developed using technique mentioned above however, the base model used was tweaked 3 times to attain the best model accuracy.

Ensemble 1: Basic Proposed Model Ensemble

Model Description:
Based on few research papers, we got an intuition about using Ensemble of CNN's to develop our classification model. Thus, we applied the ensemble learning technique to develop a robust model. Similar to ensemble learning techniques of Machine Learning like Random Forest where each Random Forest is an ensemble of decision trees trained on randomly picked different train-validation split of

dataset, we utilized Proposed Model as specified in [8] and described above. We developed an ensemble of 10 such CNN models each of which was trained on a randomly picked Train-Validation Split (based on different random seed). The ensemble was used with Max Voting technique to classify the images while testing.

Ensemble 2: Proposed Model Modified with 4 layers and 0.25 dropout

Model Description:

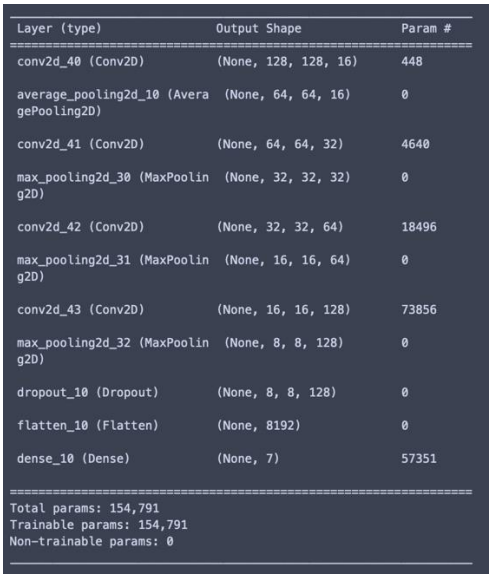
The Ensemble developed with basic model provided with substantial improvement in Accuracy. However, the recall for some of the classes i.e., bkl, df and mel were low. Thus, to improve the recall of these classes, we applied certain modifications explained below.

Modifications:

1. Switch to Average Pooling from Max Pooling: We modified the pooling technique used after the first layer of CNN from Max pooling to Average pooling. This affects the pixel value being taken from portions of images to develop feature map. Average pooling helps in extracting overall features such as image contrast, whereas max pooling is useful for edge detection.
2. Added one more convolution layer of 128 filters. In CNN, an addition of a CNN layer results in extraction of more features from the images. In a dataset with large number of images with significant variations, increasing the convolution layers increases the accuracy and more details from the images are extracted. We did not increase more as it will increase the computational power usage and memory during training. It is a best practice to increase the number of filters in powers of 2 so we used 128 filters.
3. Dropping dropout to 0.25: We decreased the dropout after the third CNN +Pooling Layer from 50% to 25%. The dropout is responsible for deactivating neurons in layers such that model doesn't overfit. But with a high value of 0.5 it may result in underfitting as we have large number of images with wide variations. Thus, we reduced the value to 0.25.
4. Decreasing Number of Epochs: From the training of Ensemble 1, we observed that the model doesn't improve significantly after 30 epochs but consumes resources. Thus, we reduced the number of epochs for Ensemble 2.

Model Architecture:

The Ensemble architecture used here is same as that of Base Ensemble Model. The only difference is that the Base CNN model used is modified as given below:



Layer (type)	Output Shape	Param #
conv2d_40 (Conv2D)	(None, 128, 128, 16)	448
average_pooling2d_10 (AveragePooling2D)	(None, 64, 64, 16)	0
conv2d_41 (Conv2D)	(None, 64, 64, 32)	4640
max_pooling2d_30 (MaxPooling2D)	(None, 32, 32, 32)	0
conv2d_42 (Conv2D)	(None, 32, 32, 64)	18496
max_pooling2d_31 (MaxPooling2D)	(None, 16, 16, 64)	0
conv2d_43 (Conv2D)	(None, 16, 16, 128)	73856
max_pooling2d_32 (MaxPooling2D)	(None, 8, 8, 128)	0
dropout_10 (Dropout)	(None, 8, 8, 128)	0
flatten_10 (Flatten)	(None, 8192)	0
dense_10 (Dense)	(None, 7)	57351

Total params: 154,791
Trainable params: 154,791
Non-trainable params: 0

Ensemble 3: Proposed Model with 4 layers and 0.4 dropout

Model Description:

The overall accuracy in the previous model was very high with great precision and recall scores as well. This created a doubt regarding model being a bit overfitted due to low percentage of dropout. Thus, we developed yet another Ensemble model with 0.4 or 40% drop out.

Modifications:

Increase Dropout: In order to make sure that model is not overfitting, we increased the dropout to 40%. Increasing dropout results in lowering the chances of overfitting.

Model Architecture:

The model is same as before just with a difference in the dropout ratio.

Results

The Ensemble technique provided a dramatic improvement in the accuracy. The results and improvements for each ensemble developed is mentioned ahead.

Ensemble 1

```
Confusion Matrix
[[  85    1    1    0    0   12    0]
 [   1  133    2    0    0   17    0]
 [   1    3  213    0    4  109    0]
 [   1    0    0   22    1   11    0]
 [   2    2    5    0  224  101    0]
 [   1    2    9    4    3 1993    0]
 [   0    0    0    0    0    0   43]]
Classification Report
              precision    recall  f1-score   support

       akiec       0.93      0.86      0.89        99
         bcc       0.94      0.87      0.90       153
         bkl       0.93      0.65      0.76       330
          df       0.85      0.63      0.72        35
         mel       0.97      0.67      0.79       334
          nv       0.89      0.99      0.94      2012
        vasc       1.00      1.00      1.00        43

    accuracy                           0.90      3006
   macro avg       0.93      0.81      0.86      3006
weighted avg       0.91      0.90      0.90      3006
```

- Number of Epochs : 50 for each model in Ensemble
- Overall Accuracy: 90%
- Improvement from Base Model: +15%

Ensemble 2

```
Confusion Matrix
[[  98    0    0    0    0    1    0]
 [   0  151    0    1    1    2    0]
 [   1    3  314    0    5    6    0]
 [   0    0    0   35    0    0    0]
 [   0    1    2    0  326    5    0]
 [   1    9   21    2   14 1964    1]
 [   0    0    0    0    0    0   43]]
Classification Report
              precision    recall  f1-score   support

       akiec       0.98      0.99      0.98        99
         bcc       0.92      0.97      0.95       155
         bkl       0.93      0.95      0.94       329
          df       0.92      1.00      0.96        35
         mel       0.94      0.98      0.96       334
          nv       0.99      0.98      0.98      2012
        vasc       0.98      1.00      0.99        43

    accuracy                           0.97      3007
   macro avg       0.95      0.98      0.97      3007
weighted avg       0.98      0.97      0.97      3007
```

- Number of Epochs for each Model : 30
- Overall Accuracy: 97%
- Improvement from Ensemble 1: +7%

Ensemble 3

```
Confusion Matrix
[[  98    0    0    0    1    0    0]
 [   1  151    1    0    0    2    0]
 [   5    6  301    0    3   14    0]
 [   0    0    0   35    0    0    0]
 [   1    0    2    0  323    8    0]
 [   7   13   36    5   30 1921    0]
 [   0    0    0    0    0    0   43]]
Classification Report
              precision    recall  f1-score   support

       akiec       0.88      0.99      0.93        99
         bcc       0.89      0.97      0.93       155
         bkl       0.89      0.91      0.90       329
          df       0.88      1.00      0.93        35
         mel       0.90      0.97      0.93       334
          nv       0.99      0.95      0.97      2012
        vasc       1.00      1.00      1.00        43

    accuracy                           0.96      3007
   macro avg       0.92      0.97      0.94      3007
weighted avg       0.96      0.96      0.96      3007
```

- Number of Epochs for each Model : 30
- Overall Accuracy: 96%
- Improvement from Ensemble 1: +6%
- Change in Accuracy from Ensemble 2 after changing dropout: -1%

The Ensemble Model 1 provided a dramatic increase of 15% from the base model that proved the benefit of using Ensemble Learning. The Model Validation results showed that Ensemble 3 performed best in terms of Overall Accuracy achieving a remarkable value of 96% within 30 epochs. Ensemble 2 provided better accuracy, but we were suspicious of overfitting. Although, the accuracy dropped by 1% in Ensemble 3 as compared to Ensemble 2, but it made the model more generalize preventing overfitting.

Conclusion

The study conducted on ISIC 2018 dataset revealed the importance of machine learning in field of medical diagnosis. The DCNN Ensemble model developed using a simple base model with just 4 convolutional layers (Ensemble 3) & 154K trainable parameters achieved an overall accuracy of 96% on validation set and 88% on test dataset (highest among various test sets). AKIEC class shows the best performance with the highest Recall(99%)and F1-sccore(95%). Most of the misclassified images were predicted as NV, as NV class constituted wide variety of images, thus capturing more training features than other classes. It can also be concluded that some of the misclassified images from the test set had noise such as hair and other artifacts.

The pre-processing of images proved to be a crucial step as it dramatically improved the performance of model. Although, the manual selection of images that require pre-processing is a challenging & time-consuming task, it is worth the benefits. It is to be noted that the objective is to capture as many relevant features as possible for better diagnosis thus, a robust model can save time as well as lives.

The Augmentations were performed for training data only as validation images must reflect unprocessed test dataset. Balancing the dataset via this step also helped in better training & thus, developing a generalized model.

From the perspective of diagnosis of skin cancer(BCC & MEL lesions), it can be concluded that BCC has a better recall than MEL as only 6 out of 100 lesions were misclassified as non-cancerous however, for Melanoma 23 lesions were misclassified. This is unacceptable as melanoma is highly cancerous and if remain undetected, can be fatal. The model can be improved in terms of better recall for Melanoma by obtaining more original images of melanoma. This will enable capturing of more varied features of lesion and thus, will improve its detection. Although BCC & MEL are cancerous lesions, AKIEC lesions can too turn into squamous cell carcinoma skin cancer if left untreated. The model shows excellent results for AKIEC lesions and thus can help in early diagnosis & treatment preventing it from turning cancerous.

Below table summarizes the results of all the experimented models along with overall Validation and Test Accuracies:

S.No.	***Model Name***	***Validation Accuracy (%)***	***Test Accuracy (%)***
1	Base Model	75	-
2	Modified Base Model	36.8	-
3	RESNET50	57	-
4	InceptionV3	71	-
5	Ensemble 1	90	65
6	Ensemble 2	97	13
7	**Ensemble 3**	**96**	**88**

Future Work

Although the objective of this study was successfully attained with an Ensemble Model providing good class wise as well as overall accuracies, our analysis & research helped us arrive at following extensions of this study:

- **Accuracy gains via better training**
  The augmented data does not capture varied features of skin lesions and thus, the model is able to learn only available cases. Any unknown case (with highly varying features) presented to model can result in invalid predictions. Thus, more data for each class of lesions must be collected such that training is performed on actual cases instead of computer-generated images. This can provide substantial gains in individual class accuracies as well as make the model more robust.

- **Improved Model with addition of Segmentation Masks:**
  The model can further be improved by using original Segmentation Masks for skin lesions as a pre-training step. This will enable capturing only the relevant features of lesion by the model thus, resulting in a highly accurate model. Although, the segmentation masks are available on ISIC website, those were developed using AI techniques. We suggest using better computer imaging techniques to capture masks of lesions while capturing their images.

- **Cloud Deployment & Web Service:**
  The model can be deployed to cloud services such as AWS and Azure that provide state-of-the-art machine learning environments for highly scalable and efficient web services. The model thus can be utilized in a real-time Web-Application where Doctors can upload the images of lesions to get accurate predictions of the class of lesions. The cloud also enables continuous learning as it can use the uploaded images as feedback to improve the model efficiency. Moreover, the model can be continuously trained on more data.

References

1. *https://www.skincancer.org/skin-cancer-information/skin-cancer-facts/*
2. *https://challenge.isic-archive.com/*
3. *https://pubmed.ncbi.nlm.nih.gov/21342292/, Smith L, Macneil S. State of the art in non-invasive imaging of cutaneous melanoma. Skin Res Technol. 2011 Aug;17(3):257-69. doi: 10.1111/j.1600-0846.2011.00503.x. Epub 2011 Feb 22. PMID: 21342292.*
4. *https://arxiv.org/abs/1907.04305 Hasan, M. K., Dahal, L., Samarakoon, P. N., Tushar, F. I., Marťı, R.: DSNet: Automatic dermoscopic skin lesion segmentation. Computers in Biology and Medicine 120, 103738 (2020).*
5. *https://arxiv.org/abs/1601.07843 Mishraa, N. K., Celebi, M. E.: An overview of melanoma detection in dermoscopy images using image processing and machine learning. arXiv:1601.07843 (2016)*
6. *https://www.aad.org/public/diseases/skin-cancer/types/common*
7. *https://www.mayoclinic.org*
8. *https://en.wikipedia.org/wiki/Melanocytic_nevus*
9. *https://iopscience.iop.org/article/10.1088/1757-899X/982/1/012005/pdf Yunendah Nur Fu'adah et al 2020 IOP Conf. Ser.: Mater. Sci. Eng. 982 012005*

Appendix

Code Usage

The results can be reproduced by following the steps mentioned below.

1. Place the images inside the directory 'Skin_cancer_detection/ISIC2018/orig'
2. Run the jupyter notebook placed under 'Skin_cancer_detection/ISIC2018_Preprocessing_Augmentation.ipynb' to perform pre-processing. All the directories get created internally in code.
3. Run Model1_2_3_4.ipynb file to create the first four models and their corresponding results. Note that the models ceated here are not getting saved since the results are not prominent.
4. Run Ensemble1.ipynb, Ensemble2.ipynb or Ensemble3.ipynb to develop ensembles of model. Make sure to follow instructions given in the notebook. Also, these models are saved and can be loaded without explicitly training them. All the notebooks contain code to load these models at the end.

Notes: Make sure to change path separator('/' to '\\') if running on Windows system.

Code File Description

S.No.	File	Description	Location
1.	ISIC2018_Preprocessing_Augmentation.ipynb	Consists compiled code for all data pre-processing, augmentation and train-test splitting for creation of final data for modelling	ISIC2018
2.	Model1_2_3_4.ipynb	Contains the first four models as described: • Model 1 : Proposed model from Research paper • Model 2- Modified version of Model 1. • Model 3 – ResNet50 • Model 4 – InceptionV3	ISIC2018
3.	Ensemble 1.ipynb	Contains the Ensemble of 10 Base Models (Model1)	ISIC2018

		trained on randomly shuffled different sets of test train validation splits.	
4.	Ensemble 2.ipynb	Contains the Ensemble of 10 modified Base Models with 1 extra CNN layer, 25% dropout, trained on randomly shuffled different sets of test train validation splits.	ISIC2018
5.	Ensemble 3.ipynb	Contains the Ensemble of 10 modified Models (Model1) with 1 extra CNN layer, 40% dropout, trained on randomly shuffled different sets of test train validation splits.	ISIC2018
6.	Cropping_2018_Image_Class_Map.csv	Image to Class mapping of manually selected images for cropping	ISIC2018/labels
7.	Hair_Removal_2018_Image_Class_Map.csv	Image to Class mapping of manually selected images for Hair Removal	ISIC2018/labels
8.	ISIC2018_Task3_Training_GroundTruth.csv	Ground Truth information of training dataset from ISIC Archive	ISIC2018/labels
9.	ISIC2018_Task3_Validation_GroundTruth.csv	Ground Truth information of Validation dataset from ISIC Archive	ISIC2018/labels
10.	Ensemble Saved Models	All the trained Ensemble models with trained weights to be re-used.	ISIC2018/Saved Models
	saved_model.pb keras_metadata.pb variables.data-00000-of-00001 variables.index Ensemble***X***_model***Y***.h5	Files that can be loaded using tensorflow libarary(code available in Model notebooks above) to load the trained model. X -> Ensemble Number Y -> Model Number in each Ensemble	EnsembleX/