Skin Cancer Detection

Using Deep Convolutional Neural Networks

ISIC 2019

—

Submitted by:
Mohamed Ghayaas Anjum (100364934)
Leo Liang (100244775)
Avneet Kaur (100368395)

Table of Contents

Background

Cancer begins in the body or an organ when healthy cells change and grow out of control, forming an extra mass called a ‘tumor’. A tumor can be cancerous or benign. A cancerous tumor is ‘malignant’ which means that it can grow and spread to other parts of the body. A ‘benign’ tumor means the tumor can grow but won’t spread. (Cancer.Net Editorial Board, 2020)

Skin cancer is globally the most commonly occurring cancer, with melanoma being the deadliest form. Doctors diagnose skin cancer in approximately more than 3 million Americans each year. If skin cancer is found early, it can be treated with topical medications or procedures done by a dermatologist. Sometimes an outpatient surgery also becomes necessary in the treatment. Since cancers on the skin can be detected early, it is responsible for less than 1% of all cancer deaths.

Dermoscopy is a skin imaging technology that has demonstrated commendable improvement in the diagnosis of skin cancer compared to unaided visual inspection. (Pacheco et al., 2019) But clinicians should receive adequate training for those improvements to be realized. The use of computer-aided diagnosis (CAD) systems for skin cancer detection has been increasing over the past decade. Recently, deep learning models have been achieving remarkable results in the analysis of various medical images. (Pacheco et al., 2019) Specifically, the convolutional neural networks (CNN) have become a standard approach to handling image analysis problems and also extended to medical images. To make this expertise widely available, the International Skin Imaging Collaboration (ISIC) has developed the ISIC Archive, an international repository of dermoscopic images, for the purpose of both clinical training as well as for supporting technical research toward automated algorithmic analysis by hosting the ISIC Challenges.

Objective

The objective of this project is to develop a deep learning model in the context of the multi-class classification of skin lesions. The proposed Deep Convolutional Neural Network model is carefully designed with several layers, and multiple filter sizes, and manipulates filters and layers to improve the efficacy and accuracy. Dermoscopic images are acquired from the International Skin Imaging Collaboration databases (ISIC-19) for experiments.

The goal of ISIC 2019 is to classify dermoscopic images among eight different diagnostic categories: Melanoma, Melanocytic nevus, Basal cell carcinoma, Actinic keratosis, Benign keratosis, Dermatofibroma, Vascular lesion, and Squamous cell carcinoma.

Overview of the Dataset

The ISIC – 2019 training dataset consists of 25,331 dermoscopic images of skin lesions belonging to 9 different cases of skin cancers. However, the 9th class is an outlier and is excluded from the analysis as there are no images present corresponding to that class. The dataset is a combination of the earlier published datasets – BCN_20000 Dataset, HAM10000 Dataset, and MSK Dataset.

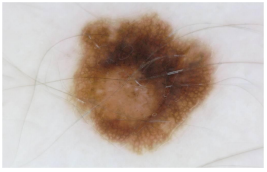
The resolutions of the images in the ISIC 2019 are varying between **600 x 450, 1024 x 1024, 1022 x 767, and 1024 x 768**. The ground truth of these images is stored in another dataset (ISIC_2019_Training_GroundTruth.csv) which consists of the names of all the images and mapping of these images to their corresponding labels (skin lesion class). The 8 classes and the counts of images in each class are shown below:

CLASS	NAME	COUNT
NV	Melanocytic Nevus	12875
MEL	Melanoma	4522
BKL	Benign Keratosis Lesion	2624
BCC	Basal Cell Carcinoma	3323
AK	Actinic Keratosis	867
VASC	Vascular Lesions	253
DF	Dermatofibroma	239
SCC	Squamous Cell Carcinoma	628
Total		25331

Table 1: Count of training images

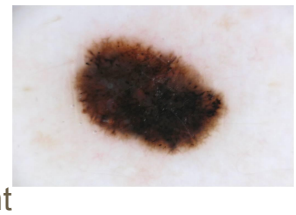
Description of Classes

- **Melanocytic Nevus (NV)**



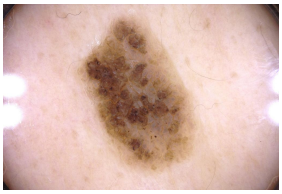
A melanocytic naevus (American spelling 'nevus'), or mole, is a common benign skin lesion caused by local proliferation of pigment cells known as melanocytes. A melanocytic naevus can be present in a person at birth, (a congenital melanocytic naevus) or appear later (an acquired naevus). Almost every person has at least one melanocytic naevus. Although the exact reason for the local proliferation of naevus cells is unknown, the number of melanocytic naevi a person has depends on genetic factors, sun exposure, and or immune status. (Oakley, 2016)

- **Melanoma (MEL)**



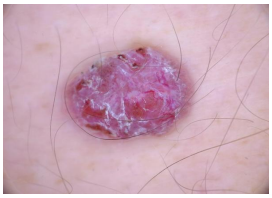
Melanoma is a serious and lethal form of skin cancer that begins in melanocyte cells. While it is less common than basal cell carcinoma (BCC) and squamous cell carcinoma (SCC), melanoma is more dangerous because it is malignant in nature which means that it could rapidly spread to other parts of the body or skin if not identified or treated at an early stage.

- **Benign Keratosis Lesions (BKL)**



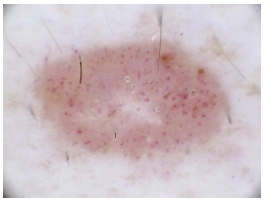
A keratosis lesion is a noncancerous (benign) growth on the Skin. The color can range from white, tan, brown, or black. Most are raised and appear "stuck on" to the skin. They may look like warts and often appear on a person's chest, arms, back, or other areas. They're very common in people older than age 50, but younger adults can get them as well. With age, more and more people get 1 or more of these growths. (Cedars-Sinai, n.d.)

- **Basal Cell Carcinoma (BCC)**



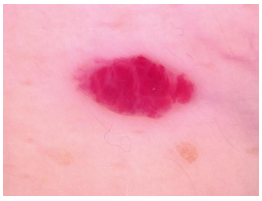
Basal cell carcinoma (BCC) is the most frequently occurring and the most common form of skin cancer. They usually arise from abnormal and or uncontrolled growth of basal cells. These lesions can look like open sores, red patches, pink growths, shiny bumps, or scars. The lesions commonly arise in areas of the body more exposed to the sun. (Karen & Moy, 2022)

- **Actinic Keratosis (AK)**



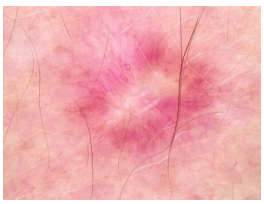
Actinic keratosis (AK) is the most common precancer that forms on skin damaged by chronic exposure to ultraviolet rays from the sun. They often appear as a small dry, scaly, or crusty patch on the skin. They may be red, light or dark tan, pink, or white and are sometimes raised. The lesions frequently develop on sun-exposed areas of the face, lips, ears, scalp, shoulders, neck, and the back of the hands and forearms. (Goldberg & Lebwohl, 2022)

- **Vascular Lesions (VASC)**



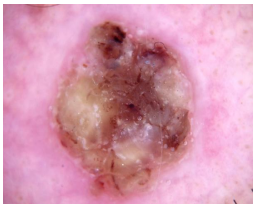
Vascular lesion is an umbrella term for different varieties of skin irregularities such as Cherry Angiomas, Vascular Birthmarks, Port-Wine Stains, Vascular malformations, etc., that result from an overproduction of very small capillaries at, or on top of, the skin surface.

- **Dermatofibromas (DF)**



DF is a small, benign skin growth that is mostly found on the lower legs. They are usually of the size of around 2-3 mm, purplish-brown color, and hard in structure. They can be pink, red, grey, or brown in color.

- **Squamous Cell Carcinoma (SCC)**



Squamous cell carcinoma is a common form of skin cancer that develops in the squamous cells. Although they can spread aggressively, they are usually not life-threatening. Mostly arise from prolonged exposure to ultraviolet (UV) radiation. (*Squamous Cell Carcinoma of the Skin*, n.d.)

Pre-processing

The goal of the pre-processing is to improve the training data for the CNN model by removing undesirable noise and enhancing the images to improve the performance of the model. The following steps were taken in the pre-processing stage:

1. Mapping – Separate all 8 class images into 8 directories.
2. Train test valid split – split the training images in the ratio of 70:15:15
3. Augmentation – Balance the amount of data in each class and increase the training data.
4. Resizing – Reduces time taken at each epoch of training CNN model.
5. Cropping – Avoid unwanted areas of an image (dark corners, ruler marks, normal skin area)
6. Hair Removal – Eliminate the element of noise that might be learned as a feature by model
7. Normalization – Ensure each pixel has similar distribution making the convergence of CNN faster.

Mapping

With the help of python script (“image_placement_2019.ipynb”), all the 25331 training set images from the 2019 dataset were contained in a single folder (orig) and were moved to their 8 respective distinctive directories according to the mapping present in the ground-truth .csv dataset.

Train, Test, and Validation split

We perform the splitting of images to be used for the training, testing, and validation purpose while building a model. A python script is used to create three separate directories: ‘train_70%’, ‘test’, and ‘valid’. This script takes the training images from each of the 8 subfolders created from the mapping step above and places 70% of the images of each class into a train folder, 15% of images into the test folder, and the remaining

15% images into a valid folder. The number of images in each class after the split is as below:

Sl No.	Class	Count	Training	Testing	Validation
1	NV	12875	9012	1931	1932
2	MEL	4522	3165	678	678
3	BKL	2624	1836	394	394
4	BCC	3323	2326	498	498
5	AK	867	606	130	131
6	VASC	253	177	38	38
7	DF	239	167	36	36
8	SCC	628	439	94	95

Table 2: Count of images after split

All further augmentations and pre-processing techniques are applied to the 70% training data. The test and valid data will be used in the model unprocessed.

Augmentations

Augmentation of images is done to deal with the problem of skewed classes, overfitting, and training image scarcity. For example, the class NV has 9012 training images whereas the class DF has 167. This large difference in the number of training data creates a bias in the model for the biggest class. Therefore, a series of suitable augmentations are performed to take the initial 167 DF images and obtain the same number (9012) images for DF, thus matching with the biggest class. Likewise, all the other class images are augmented to obtain 9012 images.

This sequence of augmentations was chosen with a focus that the resulting images are obtained as natural, and real, as possible and avoid duplications. The various kinds of

augmentations performed in this step are Shift scale rotation, Horizontal flipping, vertical flipping, center cropping, rotations without shear shifting, and shear shifting without rotations. No augmentations are applied to the class NV.

Key Features of the applied augmentations:

1. Augmentations are applied to match the training size of the biggest class NV, size = 12875. Training input of 70% = 0.7x12875 = 9012. Therefore, the target training size range of other classes - (8700, 9300).
2. All augmentations are applied using the 'Albumentations' library.
3. For images where center cropping was applied, the input images were resized to 256x256 before applying and then cropped to a specified resolution.
4. Centre cropping was tested with a crop resolution of (100*100) (150*150) (175*175) and the best was chosen.
5. The final resolution of images after all augmentations were maintained to 256x256.
6. The characteristics of lesions like shape (circular or irregular or patch like), size (small or spread through the dimension of image), (placement weather placed in the middle of the image randomly) were also considered to decide the sequence of augmentations, degrees of rotations, and other parameters chosen.
7. Augmentations are applied to all the resulting images from the previous augmentation of a class except the last augmentation which is applied on a selected number of randomly chosen and shuffled images to match the final training size.

Note:

1. Refer to Appendix 2: Augmentation Visuals. For original and augmented images of all classes
2. Refer to Appendix 3: Augmentations Summary. For Sequence and types of augmentations in each class

3. The details, sequence, and parameters of augmentations for the 2019 dataset are also summarized in the file "*ISIC_2019_augmentations_summary.xlsx*". Please refer to the file for details of the augmentation process.

Resizing

Resizing images is a critical step in Image pre-processing as CNN models run faster on smaller images. In the 2019 dataset, we have varying resolutions for the images like 600 x 450, 1024 x 1024, 1022 x 767, and 1024 x 768. The resizing has been applied as an exit step in the augmentation process for the 2019 dataset. All the resulting images from the augmentation process are stored with a resolution of 256 x 256 which is the same for all classes. The resulting augmented and resized images of each class has a size of about 150 MB per class. The images can be further resized to 128 x 128, or 64 x 64 as an input step during the model creation if required. We consider square dimensions as the TensorFlow library which we are using to create our model, expects that dimension.

Image Cropping

The image cropping was applied to 5 classes in the following instances:

1. Class AK: on 4164 shuffled and randomly chosen images as the last augmentation step (after 3 augmentations). The cropped image size is 175 x 175.
2. Class BKL: 1st augmentation step for class BEL on 1836 training images. The cropped image size is 100 x 100.
3. Class DF: As a 2nd augmentation step on 334 images obtained from the first augmentation. The cropped image size is 125 x 125.
4. Class SCC: As a 4th augmentation step on 3512 images. The cropped image size is 175 x 175.
5. Class VASC: 2nd augmentation step on 354 images with a cropped image size of 125 x 125.

We chose a smaller crop size for classes BKL, DF, and VASC as the lesions here are mostly small in shape compared to the other two classes mentioned above. Likewise, a bigger resolution of 175 x 175 was chosen to accommodate a slightly bigger lesion size of the classes AK and SCC.

The cropping was not applied to the remaining 2 classes MEL and BCC because the lesions in the remaining classes were unevenly shaped and spread throughout the dimension of the image. Therefore, cropping such images would have resulted in the elimination of important features of the model.

Digital Hair Removal

Artifacts such as hair especially dark-colored hair are the major obstacles in creating a robust model. A CNN model can learn every feature of the image given to it as training data. Hair in images is processed as a feature of skin lesion itself resulting in poor model performance. Hence, it becomes essential to eliminate any such noise.

The following methodology has been employed to tackle this problem:

A different approach was formulated for removing hairs in the 2019 dataset. As the size of images in each class is very large, manual selection and separation of only the haired image from each class and application of hair removal was not an optimal approach. Thus, it was decided that this pre-processing step will be applied class-wise. We visually inspected all 8 classes from the training dataset to check where this problem exists predominantly. It was found that 5 classes out of 8 had a higher percentage of images with hairs. The hair removal was applied on 9012 images of the 5 classes after the augmentations were performed on the classes mentioned below:

- MEL – Melanoma
- BKL - Benign Keratosis Lesion
- NV - Melanocytic Nevus
- BCC - Basal Cell Carcinoma
- VASC - Vascular Lesions

The technique used for hair removal is called Dullrazor (Lee et al., 1997) which is a well-known mechanism for removing hair from dermatoscopic images. The algorithm has been developed after research and is efficient, especially for skin cancer images. It performs better on especially dark thick hair as they are the main culprit that confuses 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 color 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 color 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 the OpenCV library.

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

Normalization

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

Model Building

Phase 1 – Model Testing

➢ Overview of Modeling Approach

This section summarizes the models created for classifying the 8 classes of cancers for the ISIC 2019 dataset. The metric used for evaluating the performances of the models is “Balanced accuracy”. The same metric has been used to evaluate the models on the ISIC leaderboard. All models are evaluated for new data or “test data” which is not seen by the model during its training and validation (15% of the initial training data totaling 3799 images retained during the train test split). As a start, this test data is only resized to the input image size of the respective model and is not subjected to any preprocessing or hair removal. Any subsequent changes to the type of validation data provided for training and test data for predictions are described in the respective models.

➢ Hardware Specifications

All models described in this report are trained on a local machine with the following hardware specifications:

1. AMD Ryzen 5 3550H, Quad-Core 2.10 GHz speed processor
2. Dedicated GDDR5, 4Gb, NVIDIA GeForce GTX 1650 Graphics Controller Model (CUDA enabled)
3. 8GB DDR4 SD RAM

➢ **Metrics of Evaluation**

Precision: In the simplest terms, Precision is the ratio between the true Positives and all the Positives. For our problem statement, that would be the measure of the number of images that the model correctly identifies in a particular class of cancer out of all the images predicted as positive in that class of cancer. (Purva91, 2020)

Mathematically:

$$Precision = \frac{True\ Positive(TP)}{True\ Positive(TP) + False\ Positive(FP)}$$

Recall: The recall is the measure of our model correctly identifying the True Positives. Thus, for all the images belonging to a particular cancer class, recall tells us how many the model correctly identified or predicted as images of that particular class. (Purva91, 2020)

Mathematically:

$$Recall = \frac{True\ Positive(TP)}{True\ Positive(TP) + False\ Negative(FN)}$$

We refer to it as True Positive Rate or Sensitivity. This measure is particularly important when our model involves the prediction of lethal cancer classes like melanoma. What if a patient has melanoma, but there is no treatment given to him/her because our model wrongly predicted other non-lethal cancer classes? This is a situation we would like to avoid and therefore recall is a more important metric to maximize in our modeling.

Accuracy: Accuracy is the ratio of the number of total correct predictions and the total number of predictions.

$$\text{Accuracy} = \frac{\text{True Positive + True Negative}}{\text{(True Positive + False Positive + True Negative + False Negative)}}$$

Using accuracy as a defining metric for our model does make sense intuitively, but often, it is advisable to use Precision and Recall. There may be some situations where the model accuracy is high, but our precision or recall is low. Although we tend to aim for high precision and high recall value but achieving both at the same time is not possible. For example, if we change the model for giving us a high recall value, we might detect all the patients who have melanoma cancer, but we might end up giving treatments to a lot of extra patients who were wrongly classified as melanoma but they actually do not suffer from it.

Balanced Accuracy: Normalized (or balanced) multi-class accuracy is defined as the accuracies of each category, weighted by the category prevalence. Balanced accuracy is a metric we can use to assess the performance of a classification model. (Zach, 2021) Specifically, it is the arithmetic mean of the (<category>_true_positives / <category>_positives) across each of the diagnostic categories. This metric is semantically equivalent to the ***average recall score***. (*ISIC 2019 Challenge*, n.d.)

Accuracy doesn't make us see the problem with the model. Here, model positives are represented well. So, in a case like this, balanced accuracy is better than accuracy. ***If the dataset is well-balanced, Accuracy and Balanced Accuracy tend to converge at the same value.*** (Olugbenga, 2022)

Extra: *A sample screenshot of the balanced accuracies obtained by the participants of the ISIC 2019 challenge has also been included in the "Appendix 5" section of the report as a comparison of the performance of the models created by our team.*

Model 1: Base Model

Reference File: "*model_researchpaper_processed.ipynb*"

We build a custom CNN model having 3 hidden layers. The first layer has 16 (3X3) convolution filters with "relu" activation, 32 in the second and 64 in the third. After each convolution layer, a (2x2) max pooling is applied totaling to 3 MaxPool2D layers after each Conv2D layer. A 50% dropout layer is applied after the hidden layers to prevent the overfitting of the model. The features thus learned by the model are flattened, and a dense prediction layer is applied with SoftMax activation to predict the 8 classes of skin cancer in the dataset. (Fu'adah et al., 2020)

The model is compiled with 'Adam' optimizer with 'Categorical Crossentropy' loss function and trained for maximizing "accuracy".

We use a total of 72096 images of skin cancer belonging to 8 classes. These images are obtained after balancing the skewed classes by augmentations and appropriate preprocessing. The validation images used have also been subjected to the same preprocessing as the training images.

Train Summary:

Image size: 128 x 128
No. of training images: 72096 (Augmented, balanced hair removal) (8 classes)
No. of validation images: 3799 (Preprocessed - Hair removal) (8 classes)
No. of test images: 3799 (8 classes)
No. of Epochs: 50
Training metrics: After 50 epochs
loss: 0.4458 - accuracy: 0.8294 - val_loss: 1.0900 - val_accuracy: 0.6570
Accuracy Score: 0.056
Balanced Multiclass Accuracy: 0.1303
Model Structure: Appendix. 6

Confusion matrix and classification report: Appendix. 7: Model 1
Results and Observations: A prediction accuracy of 0.0526, training accuracy of 0.8294, and validation accuracy of 0.6570 are obtained for this model structure.

No overfitting is observed during the training and after the completion of 50 epochs as we could see a gradual decrease in the validation loss and increase in the validation accuracy during training.

Con: Here, we are providing 3799 validation images for 72096 training images which constitute only about 5% of the training data. The training can be further improved by providing more percentage of validation images with respect to the training data.

Model 2: Validation data is given as a 25% split from training data

Reference File: "*model_researchpaper_processed_2.ipynb*"

To overcome the drawbacks of model 1, we ran another trial on the same model maintaining all the characteristics of the data same as model 1, only changing the percentage of validation images provided during the training. Here, we provide about 18024 processed images as validation images to the model for training, given as a 25% split from 72096 training images. As a result, we have 54072 training images, and the remaining 18024 are used as validation for training.

Train Summary:

Image size: 128 x 128
No. of training images: 54072 (Augmented, balanced hair removal)
No. of validation images: 18024 (25% split from training data)
No. of test images: 3799
No. of Epochs: 50
Training metrics: After 50 epochs
loss: 0.3995 - accuracy: 0.8488 - val_loss: 2.1264 - val_accuracy: 0.5623
Accuracy Score: 0.076

Balanced Multiclass Accuracy: 0.1203

Model Structure: Same as Model 1. Appendix 6

Confusion matrix and classification report: Appendix 7, Model 2

Results and Observations: This trial is carried out for providing enough validation images to evaluate the fit of the model to obtain a more generalized model. The training accuracy is found to increase to 0.8488 from 0.8294 but we see a slight decrease in the validation accuracy from 0.6570 (model 1) to 0.5623.

The balanced accuracy for the test data is also found to decrease by 1% from 0.1303 to 0.1203 upon providing a sufficient number of validation images. With this decrease in the validation and test accuracies, we can see that this model has generalized more than the model earlier.

Model 3: Effect of Pre-processing / Hair removal on the model performance

(Both training and validation datasets are unprocessed)

Reference File: "*model_researchpaper_without HR.ipynb*"

All characteristics of this model, and the number of training and validation images remain the same as model 1. The only difference in the training and validation images provided to this model is that they are not subjected to the hair removal preprocessing as they were in model 1.

Train Summary:

Image size: 128 x 128

No. of training images: 72096 (Augmented, balanced and NO hair removal) (8 classes)

No. of validation images: 3799 (NO hair removal) (8 classes)

No. of test images: 3799 (8 classes)

No. of Epochs: 50

Model Structure: Same as Model 1. Appendix 3

Confusion matrix and classification report: Appendix 7, Model 3

Training metrics: After 50 epochs

loss: 0.6791 - accuracy: 0.7498 - val_loss: 1.3141 - val_accuracy: 0.5969

Accuracy Score: 0.2648

Balanced Multiclass Accuracy: 0.1112

Results and Observations: The key points to note from this model are

1. The training accuracy of this model after 50 epochs is 0.7498 and validation accuracy is 0.5969 as opposed to the training and validation accuracies of model 1 which are 0.8294 and 0.6570 respectively. We can observe that for the data without hair removal, as in this model, the training and validation accuracies lowered significantly as compared to model 1 in which the data was processed by hair removal.
2. The balanced accuracy of this model is 0.1112 as opposed to the balanced accuracy of 0.1303 of model 1, and 0.1203 of model 2. The overall test prediction recall has significantly decreased for this model without any preprocessing when compared to the model with processed training images.
3. The accuracy score metric is 0.26. Though the accuracy is quite high when compared to the previous models, it can be attributed to the existence of a noise element or hairs in both the training and test. Our rationale for this higher accuracy is that the model has been trained in the presence of hairs in the images and has also been predicted on images with hairs. However, the absence of preprocessing or the presence of hairs restricts us from qualifying this model as a better one.

Model 4: Transfer Learning – NasNetMobile (Fine Tuned)

Reference File: "*model_NasNetMobile.ipynb*"

Next, we try to use a transfer learning technique to classify the 8 classes of skin cancer. We start by developing a smaller architecture, so that the hardware configuration available at hand, is able to train the model. (Tsang, 2019)

NAS stands for Neural Search Architecture. It belongs to the category of AutoML architectures. It was developed and introduced by Google which framed the problem of finding the best CNN architecture as a Reinforcement Learning problem. (Yanhui, 2021)

Basically, the idea of NAS in its core was to search for the best combination of parameters of the given search space of filter sizes, strides, output channels, and or a number of layers. In this Reinforcement Learning setting, the reward after each search action was the accuracy of the searched architecture on the given dataset.

NASNet achieved state-of-the-art results in the ImageNet competition. We select this model because the total number of parameters to train is about 5.3 million which is far less when compared to other state-of-the-art algorithms like Efficient Net or Inception Net which has about over 50M parameters to be trained. Nonetheless, the top 5 accuracy of NasNet Mobile is comparable and close to its counterparts to about 91.9% whereas that of the efficient net and inception net is in the range of 93 to 97% for their various versions. (keras.io, n.d.)

The feature vector of the NasNetMobile model (Nair, n.d.) trained on the ImageNet dataset is imported and fine-tuned for our training images. The methodology to use this transfer learning model and fine-tuning is available in the documentation of this model provided in the footnote. (Google, n.d.)

Train Summary:

Image size: 224 x 224
No. of training images: 57680 (Augmented, balanced, hair removal)
No. of validation images: 14416 (20% split from training data)
No. of test images: 3799
No. of Epochs: 30
Training metrics: After 30 epochs

loss: 0.2906 - accuracy: 0.0093 - precision: 0.9747 - recall: 0.9713 - auc: 0.9991 - val_loss: 2.6806 - val_accuracy: 0.0133 - val_precision: 0.6455 - val_recall: 0.6348 - val_auc: 0.8606

Balanced Multiclass Accuracy: 0.1405

Model Structure: Appendix 6, Model NasNetMobile

Confusion matrix and classification report: Appendix 7, Model NasNetMobile

Results and Observations: 30 epochs took about 12 to 13 hours to train. No overfitting was observed as the validation loss decreased after each epoch and the validation accuracy was found to gradually increase. Also, the model is not yet saturated after 30 epochs. It could be configured for a greater number of epochs to improve the accuracy. We obtain a better-balanced accuracy of 0.1405 on the evaluation of test data than other models described in this report.

In order to further improve the accuracy of predictions, we can use bigger transfer learning architectures like Inceptions, Efficient Nets, Xception, and Resnets which are more preferred in the classification of medical images. However, training and fine-tuning these architectures require a high configuration of RAM and GPU.

Phase 2 – Advancements

As an effort to improvise the performance of the classification for the 8 cancer classes, and to study the root cause of the accuracy issue of models on the 2019 dataset, we try to implement the learnings from the above model trials and test other models, different augmentation approaches, other ways of balancing the skewed classes and preprocessing techniques.

As a start, we tried to use an alternate approach to solve the problem of skewed classes. In the trials above, we performed augmentations and oversampling of the images and brought the number of each cancer class to 9012. Here, we implemented the training of the model with “Class Weights”.

Class Weights

The problem with most machine learning algorithms is that they assume the data is equally distributed. So, when we have a class imbalance, the machine learning classifier tends to be more biased towards the majority class (NV in our case), causing bad classification of the minority classes (DF, SCC, VASC). But we can modify our training algorithm to consider the skewed distribution of the classes. This can be achieved by giving different weights to both the majority and minority classes.

On the whole, The class_weight parameter of the fit() function is a dictionary mapping class to a weight value. For example, if we have 500 samples of class 1 and 1500 samples of class 2, then we feed in class_weight = {1:3, 2:1}. That gives class 1 three times the weight in the training than class 2.22

Class weights give all the classes equal importance on gradient updates, on average, regardless of how many samples we have from each class in the training data. This prevents models from predicting the more frequent class more often just because it's more common.

CLASS	WEIGHTS
AK	3.65
BCC	0.95
BKL	1.20
DF	13.26
MEL	0.7
NV	0.24
SCC	5.04
VASC	12.51

Table 3: Calculation of Class Weights

```
class_weights

{0: 3.6567656765676566,
 1: 0.9527085124677558,
 2: 1.2069716775599129,
 3: 13.269461077844312,
 4: 0.7001579778830964,
 5: 0.2458943630714603,
 6: 5.047835990888383,
 7: 12.519774011299434}
```

Xception Model

We use another, powerful and more reliable transfer learning model – Xception with a total of 22.9M trainable parameters and top 1 and top 5 accuracies on the imagenet dataset of 79% and 94.5% respectively. (keras.io, n.d.) Xception is a deep convolutional neural network architecture that involves Depth wise Separable Convolutions. (Akhtar, n.d.) It is also known as the “extreme” version of an Inception module and is 71 layers deep. You can load a pre-trained version of the network trained on more than a million images from the ImageNet database. (The MathWorks, Inc., n.d.)

For the purpose of our problem, we fine-tune the Xception architecture on our training images as well as reconfigure the prediction block by adding newly flattened, and dense layers coupled with batch normalization layers and dropouts to predict 8 classes of cancers. (Appendix 9: Xception Architecture)

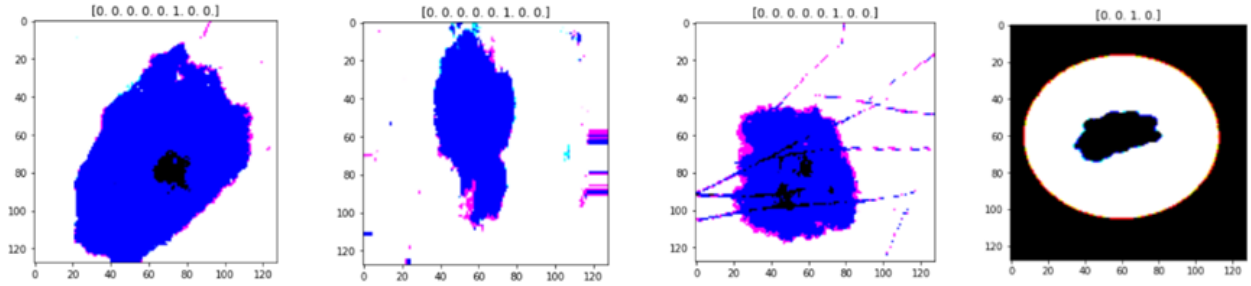
Model 5: Xception with Class Weights

Reference File: “*model_Xception_CW_P.ipynb*”

In this trial, We introduce the class_weights parameter with values as mentioned in “Table 3: Calculation of Class Weights” in the model.fit() function while training. In this trial, raw and resized images are used, that is, we have eliminated the augmentations and preprocessing/ hair removal steps as described in the former portions of this report. However, we do provide some basic and random augmentations like shear shift, zoom range, horizontal flip, and vertical flip for the training images while loading images via Image Data Generator.

Additionally, in order to remove or control the noise of images, we propose an alternate preprocessing approach. We used Resnet’s predefined “preprocess_input” function. The function is applied alike in the training, validation as well as test data to maintain similarity in the nature of images for this trial.

Preprocess_Input: The preprocess_input function is meant to adequate the images to the format the model requires. In Resnet's preprocess_input, the images are converted from RGB to BGR, then each color channel is zero-centered with respect to the ImageNet dataset without scaling. (www.tensorflow.org, 2022) The images obtained after applying this preprocessing function are as below.



As seen from the above images, some hairs still exist in the images. But this method helps separate or segment the cancer lesion from the rest of the skin. It also helps eliminate other noise elements such as ink marks, ruler marks, etc. The particulars of training are as mentioned below.

Train Summary:

Image size: 128x128

No. of training images: 17728 (These are the initial number of training images of 8 cancer classes we obtained before subjecting to augmentations or hair removal as mentioned in "Table 2: Count of images after split". Preprocessing is applied as described above)

No. of validation images: 3796 (15% split as mentioned in "Table 2: Count of images after split", preprocessing is applied)

No. of test images: 3799 (Preprocessing is applied.)

No. of Epochs: 100

Training metrics: After 72 epochs (Training stopped at 72 epochs by the Early stopping callback)

loss: 0.5395 - accuracy: 0.6997 - val_loss: 0.8753 - val_accuracy: 0.6745

Accuracy Score: 30.6

Balanced Multiclass Accuracy / Average Recall: 0.132

Confusion matrix and classification report: Appendix 7, Model 5: Xception with CW

Results and Observations: Though we have achieved a better overall accuracy on the test set of 30%, the average recall remains in the same range as the previous models (0.13). Also, by this trial, we can say that training the model with class weights helps us in better results than oversampling with augmentations.

The purpose of our modeling is to accurately predict an image belonging to a particular cancer class which can be identified by the recall of the model. In all our models, after various trials, though we see an overall betterment in the model accuracy (up to 30%), we see an average recall of a maximum of about 12 to 14%. The trend in all our models is that, on one hand, the NV class which has the highest number of original images is predicted with the highest precision and recall, on the other hand, the 4 cancer classes – AK, DF, SCC, and VASC are predicted with the least precision and recall in the range 0 to 5% in all models irrespective of the data balancing, augmentations, and preprocessing approaches.

The 4 dominant classes – NV, BCC, BKL, and MEL have original training images in the numbers of thousands (9012, 2326, 1836, 3165) whereas, on the other hand, the 4 minor classes – AK, DF, SCC, and VASC have a few hundred training images (606, 177, 167, 439). This bias also exists in the prediction data or test data used for predictions. The test data for dominant classes have about a few hundred images whereas that for the minor classes are less than 100.

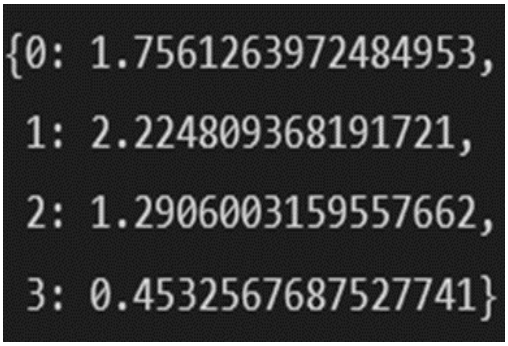
These numbers play an important role in the training and evaluation of the model. For example, the class NV has 1931 test images to predict and calculate the accuracy, precision, and recall. Whereas for a class DF, these metrics are calculated on 36 images. In a way, when we want to predict a lethal cancer class of MEL, we can say that the minor classes are acting as noise.

Model 6: Xception model with dominant classes

Reference File: "*model_Xception_CW_P_4.ipynb*"

The minor classes as mentioned above have a very low number of images. Therefore, we want to reduce the number of classes to be predicted in a model to further the performance of the model for the performance of a lethal and more important cancer class – MEL. Therefore, we retain the same Xception model used above but perform classification on only the 4 dominant classes - NV, BCC, BKL, and MEL. The train and test images are preprocessed using the preprocess_input function described above. The class weights also defer in this trial as the weights are distributed across only 4 classes.

CLASS	WEIGHTS
BCC	1.75
BKL	2.22
MEL	1.29
NV	0.45



Train Summary:

Image size: 128x128

No. of training images: 16339 (Initial number of training images of 4 dominant cancer classes Preprocessing is applied as described above)

No. of validation images: 3500

No. of test images: 3501 (Preprocessing is applied.)

No. of Epochs: 100

Training metrics: After 75 epochs (Training stopped at 75 epochs by the Early stopping callback)

loss: 0.0319 - accuracy: 0.9841 - precision: 0.9843 - recall: 0.9838 - auc: 0.9994 - val_loss: 0.6152 - val_accuracy: 0.8546 - val_precision: 0.8557 - val_recall: 0.8538 - val_auc: 0.9586

Accuracy Score: 37.7

Balanced Multiclass Accuracy / Average Recall: 0.238

Confusion matrix and classification report: Appendix 7, Model 6: Xception with dominant classes

Results and Observations: Referring to the classification report, we can see that the elimination of 4 cancer classes with fewer images does have a positive effect on the performance. The individual precisions and recalls of the classes BKL, MEL, and NV increased whereas that of BCC remained almost the same. The overall model accuracy increased by 6% here. Also, the average recall / balanced accuracy doubled from 0.132 to 0.238 in this model.

Model 7: Xception model for Binary classification

Reference File: "*model_Xception_CW_P_binary_can_noncan.ipynb*"

Our main interest in creating predictive models is to predict whether the lesion in an image is cancerous. We want to predict a cancer image with the highest certainty given by its recall. Looking into the core of the 8 cancer classes we have in the 2019 dataset, we already know that the 3 classes Melanoma MEL, Basal Cell Carcinoma BCC, and Squamous cell Carcinoma SCC lesions are dangerous cancers with MEL being the most aggressive and lethal. Therefore, in this trial, we try to group the above-mentioned three classes into a "Cancer" group and the rest of the 4 classes into a "non-Cancer" group. We created a binary classification model which predicts whether the image belongs to the cancer group or the non-cancer. We train the model using Xception with the training images subjected to the same random augmentations and resnet's preprocessing as described earlier. Owing to the grouping, the class weights in this trial are {cancer: 1.49, non-cancer: 0.75}. The model is optimized using "Adam" with the

'binary_crossentropy' loss function. Most importantly, here we used a balanced test dataset with 1270 images in each of the groups so that we eliminate the bias in test evaluation due to the imbalance in the number of test images. The particulars of the training are as below:

Train Summary:

Image size: 128x128
No. of training images: 17728 (5930 images belonging to classes BCC, MEL and SCC grouped into cancer, and 11798 images belonging to rest of the 5 classes grouped as non-cancer. Random augmentations and Preprocessing are applied)
No. of validation images: 3796 images belonging to 2 classes
No. of test images: 2540 (1270 images from cancer group and 1270 from non-cancer. Preprocessing is applied)
No. of Epochs: 100
Training metrics: After 100 epochs
loss: 0.0359 - accuracy: 0.9887 - precision: 0.9887 - recall: 0.9887 - auc: 0.9988 - val_loss: 0.4273 - val_accuracy: 0.8771 - val_precision: 0.8771 - val_recall: 0.8771 - val_auc: 0.9406
Accuracy Score: 49.17
Balanced Multiclass Accuracy / Average Recall: 0.49
Confusion matrix and classification report: Appendix 7, Model 7: Xception – Binary Classification
Accuracy and Loss training graphs: Appendix 8, Model 7: Xception Binary classifier – Training Performance

Results and Observations: Referring to the classification report, this trial yields us the highest performance with a test accuracy of 49.17% and an overall recall of 0.49 in predicting whether an image is cancerous or non-cancerous. We have also obtained very good values of precision and recall for the individual classes. Both the groups are

being predicted with a precision of 0.49. The recall for cancer is 0.41 and that for non-cancer is 0.58

Conclusions

We obtained a significant 10% more training accuracy, and 1% more balanced accuracy for our Hair removal preprocessing approach. The removal of hair, cropping of unwanted areas of images, and balancing the number of images with augmentations have proved to improvise CNN models.

Based on our experiments with respect to building CNN models, the simpler models on the 2019 dataset do not yield high-performing results. The low accuracy for the 2019 dataset can be attributed to the nature of the images, a high number of categories with a very high imbalance in the number of images in each class. Although the augmentations balance the number of images across all classes to an equal number for a model, it does not help a model to learn new features from the classes which can contribute to the improvement. For example, though the training images for class DF are brought to 9012 after augmentations, the number of features learned from this class are only the ones present in the initial 167 training images before augmentations.

Improvisation of these models was also taken place as part of the model building process by iterating through various approaches like

- increasing the number of hidden layers from 2 number to up to 4
- increasing and decreasing the number of convolution filters in each layer from 8 to up to 256
- providing a different quantity of validation splits for the model to adjust training weights like 15% to 30%
- providing processed and unprocessed validation data
- training on CPU, GPU, and testing the model for a new under sampled and balanced dataset (300 images of each class) of images.

All these approaches did not result in any significant increase in the overall model accuracy. These simpler models were trained with a training time range of 1 hour to 2 hours each.

One major problem encountered in the process of modeling was a limitation of hardware specifications to train more complex CNN models. However, we implemented a smaller, but state-of-the-art fine-tuned NasNetMobile transfer learning architecture which outperformed our base models with a better-balanced accuracy/ average recall of 0.14. The training time for this model on the above-mentioned hardware specifications was 13 hours.

In the advancement phase of the project, we checked the problem of imbalance in the number of images in cancer classes by an alternative approach of class weights. Although just using a more advanced transfer learning model like Xception in place of Nasnet did not improve the overall recall (which can be seen from the average recall of 0.132 in model 5), the overall performance and accuracy of the model were improved to 30.6 (in Model 5) by the class weight training.

We also tested this model with the earlier dull razor preprocessing, unprocessed data, as well as resnet's, preprocess function for training images. The resnet's preprocessing which changes the color channels and standardizes the images also helped in somewhat segmenting or distinguishing the lesions from the skin portion of the images, thereby yielding maximum performance when compared to the other preprocessing techniques used. We then tested the model for the unprocessed and the other two preprocessing techniques applied to test data as well in addition to the training data. It was found that applying the same preprocessing technique to the test data to which the training data is subjected, helps improve the performance as the model is able to predict the same nature of images that it has been trained on.

One reason for the lower recall of the models can be observed from the classification report of model 5. We can see that the 4 minor classes with a very less number of training images greatly impacted the model performance as the individual recalls of

these classes are still in the ranges of 0 to 5%. We assume that these extra 4 non-cancerous classes are acting as noise in our motive of predicting dangerous cancerous classes.

Therefore, we modeled to predict the 4 dominant classes by eliminating these 4 minor classes. As a result, the overall recall almost doubled from 0.13 in model 5 to 0.238 in model 6. The class NV which has 9012 original training images, and 1931 test images are being and have always been predicted with the highest precision and recall of about 0.55 and 0.57 respectively. The key takeaway from these two trials is that a greater number of original images (higher amount of data) for each class will help in a better and more accurate prediction of a cancer class.

In addition to the above, we also have to be careful that, if we want to prepare a model to predict a greater number of classes, we shall have an adequate amount of training data respective to an individual class for a model to learn a reasonable number of features of that class. Since in our scope of the project, we did not gather any new data for these minor classes, we restructured our model and grouped the 8 cancer classes into 2 – cancer and non-cancer. This restructuring helped us in two ways – one is we reduced the number of classes from 8 to 2, thereby eliminating the problem of high cardinality with less amount of data in each class. Two, we also obtained a reasonable quantity of training images in both cancer and non-cancer class (5930 cancer images and 11798 noncancer images).

The same advantage was also obtained in the test dataset. In addition to grouping, we extended the advantage by taking a balanced test data of 1270 images in each class for prediction. As a result, we obtained the highest performance so far with an overall accuracy of 49%, balanced accuracy/ average recall of 49%, and highest individual recalls of 0.41 and 0.58 for cancer and non-cancer groups respectively.

Future Work

It was learned from our research that the ISIC 2019 dataset provides better results with keras's Transfer learning architectures and ensemble models of these architectures instead of simpler models. We can improve the performance for 2019 by training and fine-tuning an ensemble of larger architectures like EfficientNet, InceptionNet, and ResNet which are not only the preferred architectures in modeling medical images but have also proven to yield robust and significant results in the range of 50 to 65% balanced accuracy for other researchers and the participants of the ISIC 2019 challenge. However, we were prevented from implementing these ensemble models due to insufficient hardware capabilities required to train these models.

After applying resnet's preprocessing function, we observed that some distinct objects in the images like hair, and black edges are still retained. Therefore, in order to get the maximum advantage from this technique, we suggest that the preprocessing is applied phase-wise. That is removal of hairs, then cropping the images to separate out the image from the dark borders, and then applying the resnet preprocess which would then help in distinguishing the lesion from the skin. Thereby, we shall eliminate more noise elements from the images. Alternatively, it is determined from our research that the segmentation procedure applied to cancer images has helped in improving the performance of the models.

In our trials, we have used the "Adam" optimizer and "categorical_crossentropy" as loss functions for multiclass classification and "binary_crossentropy" loss for binary classification. We can explore other optimizers and loss functions to test the model performance as tuning these model parameters will help improve the models. Alternatively, we can also use "Keras Auto Tuner", to automatically tune the hyperparameters of the models we build. "Auto Keras" is also another very innovative AutoML technique that automatically searches a specified number of (up to 100) available model architectures as well as tunes model hyperparameters to best suit any training data and delivers the best performing model.

There is a popular saying in the machine learning domain - “A machine learning model is only as good as the data it is fed”. (Motroc, 2018) Thus, a model can be improved by improvising the quantity and quality of the input data. Though our preprocess approach has proven to help us in our process, we can incorporate newer technologies and better approaches to handling and preprocessing the data.

The problem of a very less number of images in certain classes should also be tried to address by gathering sufficient and newer original images of underrepresented classes in the 2019 dataset so that the classes are balanced with an equal amount of original images and features to be learned by the model. This balance should also be extended to the test dataset used for predictions.

References

Akhtar, Z. (n.d.). *Xception: Deep Learning with Depth-wise Separable Convolutions*. iq.opengenus.org. Retrieved April, 2022, from https://iq.opengenus.org/xception-model/

Cancer.Net Editorial Board. (2020, October). *Skin Cancer (Non-Melanoma): Introduction*. Cancer.Net. https://www.cancer.net/cancer-types/skin-cancer-non-melanoma/introduction

Cedars-Sinai. (n.d.). *Seborrheic Keratosis*. www.cedars-sinai.org. Retrieved February 27, 2022, from https://www.cedars-sinai.org/health-library/diseases-and-conditions/s/seborrheic-keratosis.html

Fu'adah, Y. N., Pratiwi, N. C., Pramudito, A., & Ibrahim, N. (2020, December). *Convolutional Neural Network (CNN) for Automatic Skin Cancer Classification System*. ResearchGate. https://www.researchgate.net/publication/347930245_Convolutional_Neural_Network_CNN_for_Automatic_Skin_Cancer_Classification_System

Goldberg, L. H., & Lebwohl, M. (2022, January). *Actinic Keratosis Overview*. The Skin Cancer Foundation. https://www.skincancer.org/skin-cancer-information/actinic-keratosis/

Google. (n.d.). *Feature vectors of images with NASNet-A (mobile) trained on ImageNet (ILSVRC-2012-CLS). Publisher: Google License: Apache-2.0*. tfhub.dev.

Retrieved April, 2022, from https://tfhub.dev/google/imagenet/nasnet_mobile/feature_vector/5

ISIC 2019 Challenge. (n.d.). challenge.isic-archive.com. https://challenge.isic-archive.com/landing/2019/

Karen, J. K., & Moy, R. L. (2022, January). *Basal Cell Carcinoma Overview*. The Skin Cancer Foundation. https://www.skincancer.org/skin-cancer-information/basal-cell-carcinoma/

keras.io. (n.d.). *Keras Applications*. keras.io. https://keras.io/api/applications/

Lee, T., Ng, V., Gallagher, R., Coldman, A., & McLean, D. (1997, November). *DullRazor: a software approach to hair removal from images*. www.sciencedirect.com. https://www.sciencedirect.com/science/article/abs/pii/S0010482597000206

The MathWorks, Inc. (n.d.). *Xception convolutional neural network*. www.mathworks.com. Retrieved April, 2022, from https://www.mathworks.com/help/deeplearning/ref/xception.html

Motroc, G. (2018, April 6). *"A machine learning model is only as good as the data it is fed"*. jaxenter.com. https://jaxenter.com/apache-spark-machine-learning-interview-143122.html

Nair, D. (n.d.). *NASNet - A brief overview*. OpenGenus IQ. https://iq.opengenus.org/nasnet/

Oakley, D. A. (2016, January). *Melanocytic naevus*. DermNet NZ. https://dermnetnz.org/topics/melanocytic-naevus

Olugbenga, M. (2022, March 18). *Balanced Accuracy: When Should You Use It?* neptune.ai. https://neptune.ai/blog/balanced-accuracy

Pacheco, A. G. C., Ali, A. R., & Trappenberg, T. (2019, September). *Skin cancer detection based on deep learning and entropy to detect outlier samples*. ResearchGate. https://www.researchgate.net/publication/335737850_Skin_cancer_detection_based_on_deep_learning_and_entropy_to_detect_outlier_samples

Purva91. (2020, September 4). *Precision vs. Recall – An Intuitive Guide for Every Machine Learning Person*. www.analyticsvidhya.com. https://www.analyticsvidhya.com/blog/2020/09/precision-recall-machine-learning/

Squamous cell carcinoma of the skin. (n.d.). www.mayoclinic.org. Retrieved March 27, 2022, from https://www.mayoclinic.org/diseases-conditions/squamous-cell-carcinoma/symptoms-causes/syc-20352480

Tsang, S. H. (2019, May 18). *Review: NASNet — Neural Architecture Search Network (Image Classification)*. sh-tsang.medium.com. https://sh-tsang.medium.com/review-nasnet-neural-architecture-search-network-image-classification-23139ea0425d

www.tensorflow.org. (2022, February 3). *tf.keras.applications.resnet50.preprocess_input*. TensorFlow. https://www.tensorflow.org/api_docs/python/tf/keras/applications/resnet50/preprocess_input

Yanhui, C. (2021, February 26). *From AlexNet to NASNet: A Brief History and Introduction of Convolutional Neural Networks*. towardsdatascience.com. https://towardsdatascience.com/from-alexnet-to-nasnet-a-brief-history-and-introduction-of-convolutional-neural-networks-cf63bf3320e1

Zach. (2021, October 6). *What is Balanced Accuracy? (Definition & Example)*. www.statology.org. https://www.statology.org/balanced-accuracy/#:~:text=Balanced%20accuracy%20is%20a%20metric,model%20is%20able%20to%20detect

Appendix

1. Links for Data, Scripts, and Documentations

   - Link for “ISIC_2019_augmentations_summary.xlsx”

   https://github.com/Avi-k-dua/Melanoma_Detection/tree/master/ISIC_2019

   - Link for all python codes:

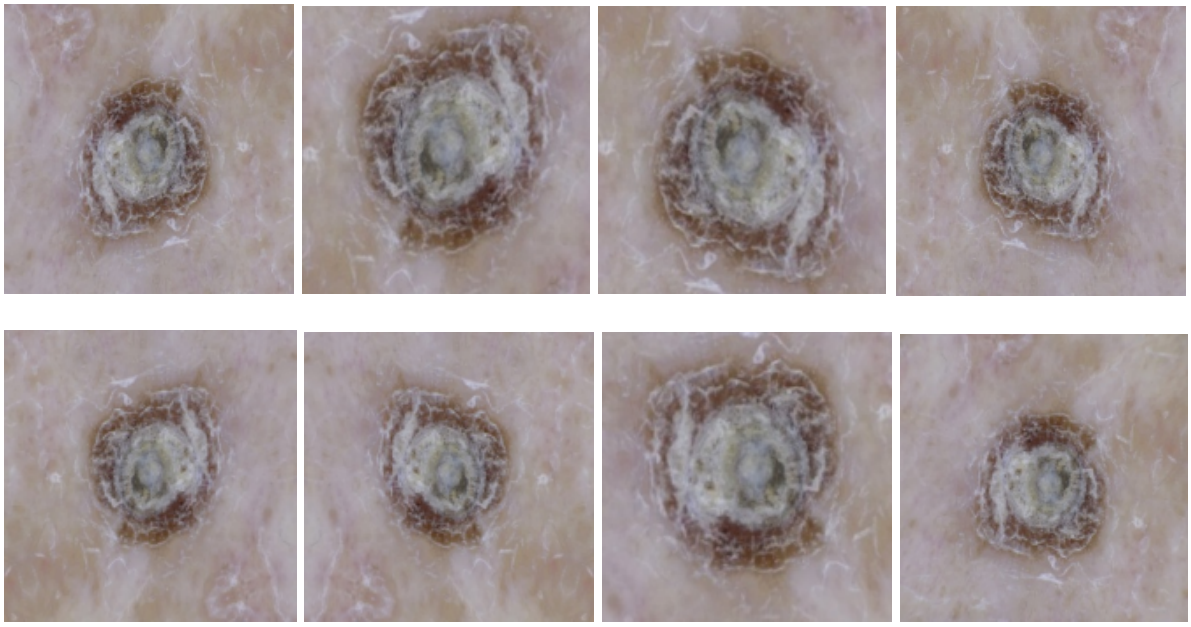
   https://github.com/Avi-k-dua/Melanoma_Detection/tree/master/ISIC_2019

   - Link for all data used in this project: One Drive: ISIC 2019
   - ISIC 2019 data archive: https://challenge.isic-archive.com/data/

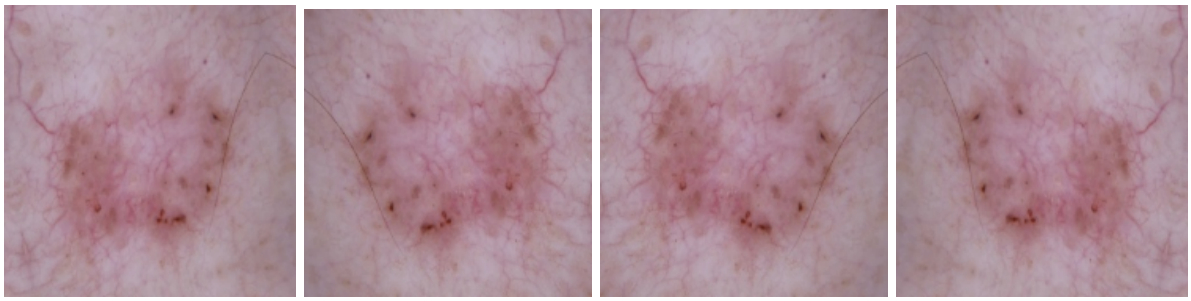
2. Augmentation Visuals

   The 1st image represents the original image. All other images are the images obtained from augmentations on the respective class.

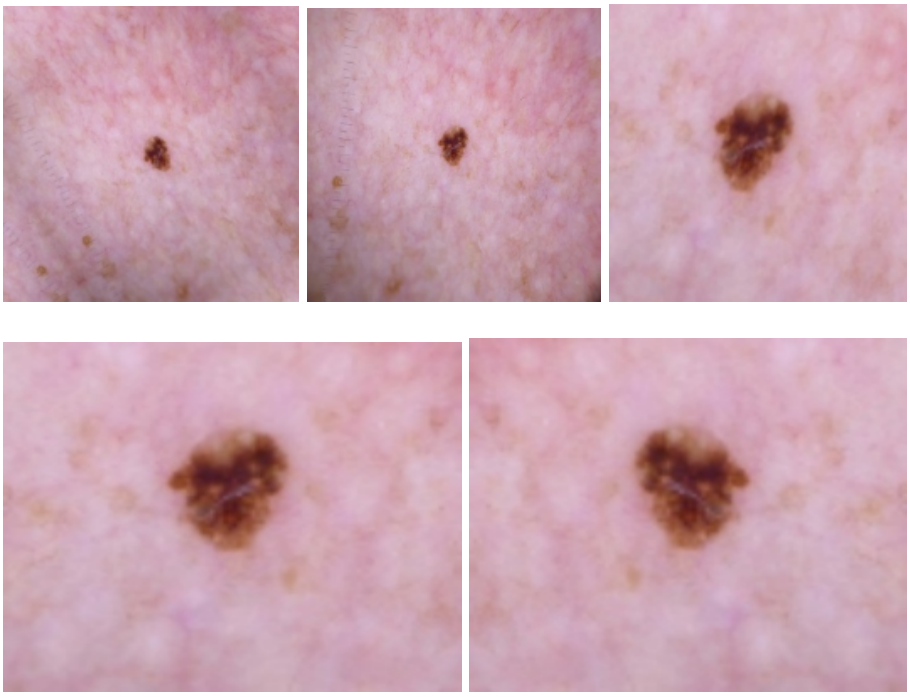
   AK:



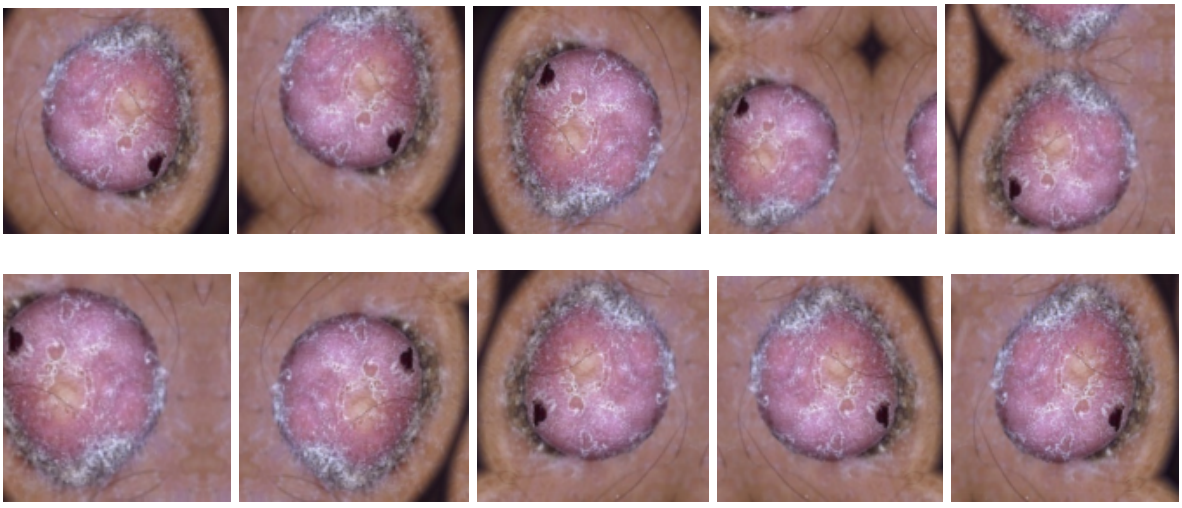
BCC:



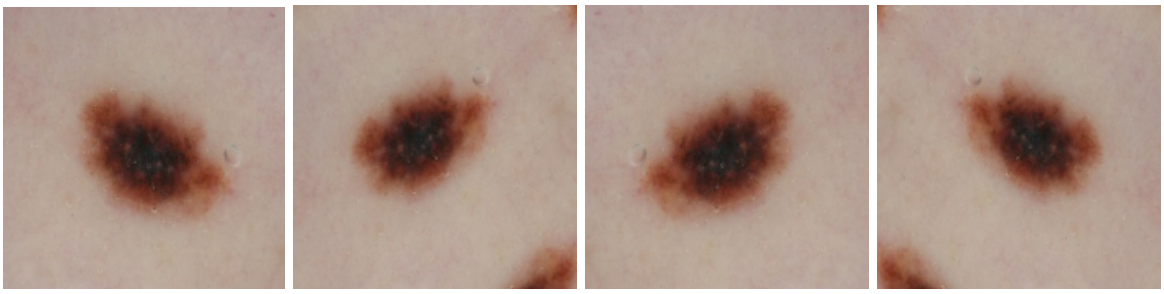
BKL:



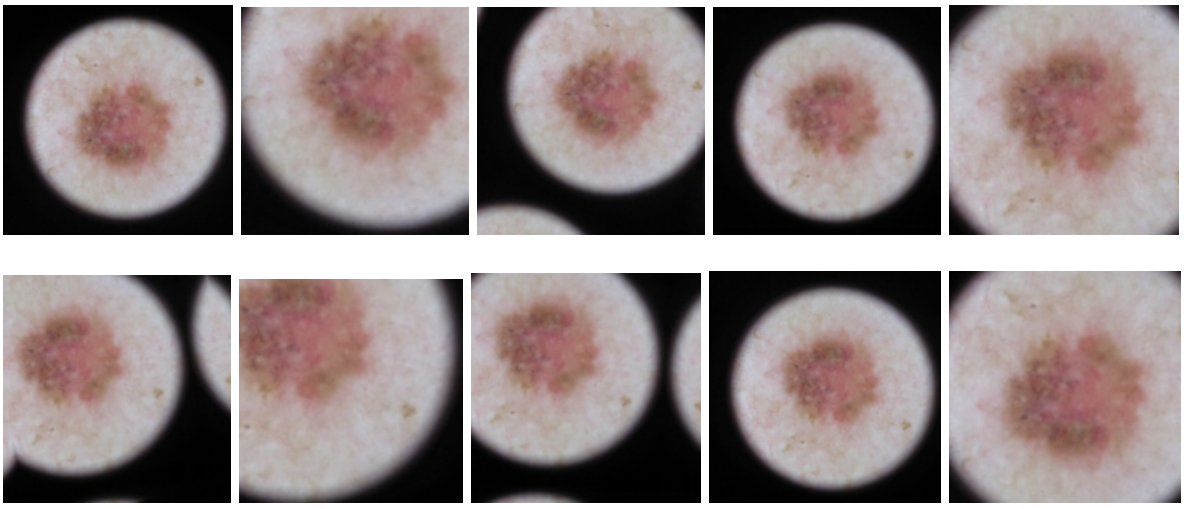
DF:



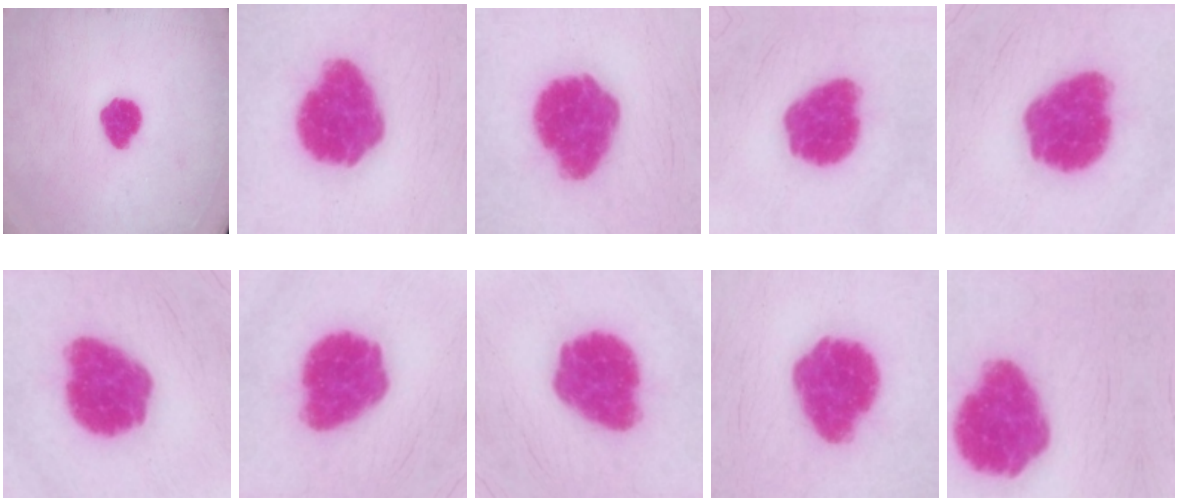
MEL:



SCC:



VASC:



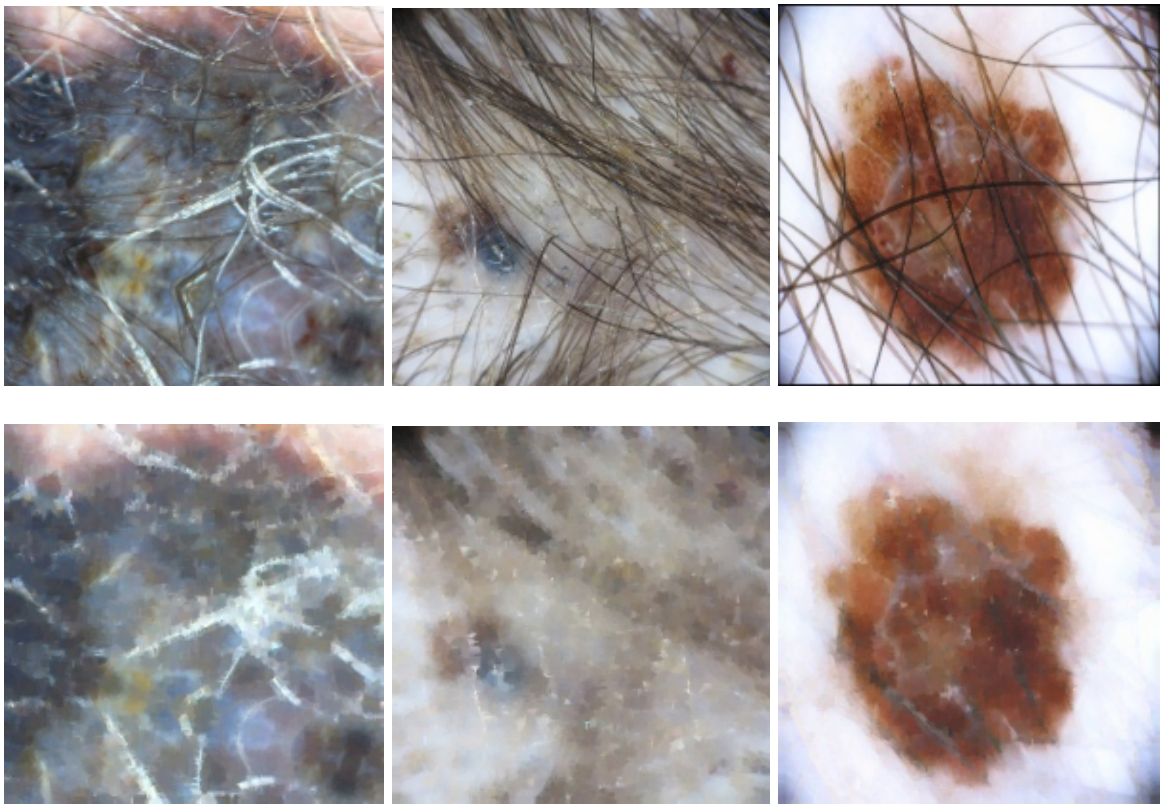
3. Augmentation Summary

ISIC 2019 Training Images - Augmentations Summary

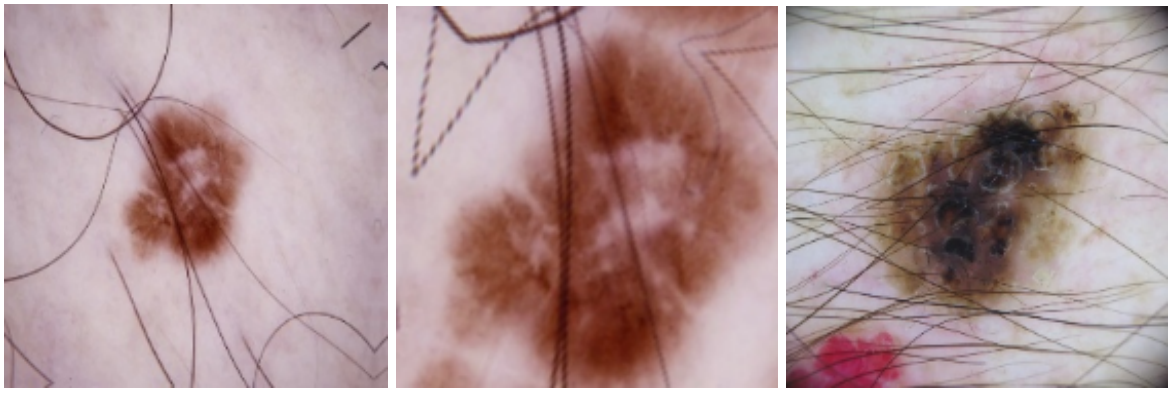
Sl. No	CLASS	Notation	Images before Aug	Aug_1	Aug_2	Aug_3	Aug_4	Aug_5	Aug_6	Training input after Aug
1	Actinic Keratosis	AK	606	ShiftScaleRotate. shift_limit=0.0625 (default), scale_limit=0.1 (default), rotate_limit=(-30,30) degrees	Horizontal flip: Flip the input horizontally around the y-axis	Vertical flip: Flip the input vertically around the x-axis.	Centre crop: cropping the centre square portion of the images of h x w =175*175. on 4164 randomly chosen images from last augmentation			9012
2	Basal Cell Carcinoma	BCC	2326	ShiftScaleRotate. shift_limit=0.0625 (default), scale_limit=0.1 (default), rotate_limit=(-40,40) degrees	Horizontal flip: Flip the input horizontally around the y-axis. on 4360 randomly chosen images from last augmentation					9012
3	Benign Keratosis	BKL	1836	Centre crop: cropping the centre square portion of the images of h x w =100*100.	ShiftScaleRotate. shift_limit=0.0625 (default), scale_limit=0.1 (default), rotate_limit=(-45,45) degrees. on 3764 randomly chosen images from last augmentation	Horizontal flip: Flip the input horizontally around the y-axis. on 1668 randomly chosen images from last augmentation				9012
4	Dermatofibroma	DF	167	ShiftScaleRotate. shift_limit=0.0625 (default), scale_limit=0.1 (default), rotate_limit=(-30,30) degrees	Centre crop: cropping the centre square portion of the images of h x w =125*125.	Rotate, without shear shifting (40 degrees)	Horizontal flip: Flip the input horizontally around the y-axis.	Vertical flip: Flip the input vertically around the x-axis.	Shift and Scale (shift_limit= 0.3, scale_limit=0.2, rotate_limit=0). without rotations. on 3668 randomly chosen images from last augmentation	9012
5	Melanoma	MEL	3165	ShiftScaleRotate. shift_limit=0.25, scale_limit=0.2, rotate_limit=(-40,40) degrees	Horizontal flip: Flip the input horizontally around the y-axis. on 2682 randomly chosen images from last augmentation					9012
6	Squamous Cell Carcinoma	SCC	439	Horizontal flip: Flip the input horizontally around the y-axis.	Vertical flip: Flip the input vertically around the x-axis.	ShiftScaleRotate. shift_limit=0.2, scale_limit=0.15, rotate_limit=(-35,35) degrees	Centre crop: cropping the centre square portion of the images of h x w =175*175.	Rotate, without shear shifting (20 degrees). on 1991 randomly chosen images from last augmentation		9012
7	Vascular Lesion	VASC	177	ShiftScaleRotate. shift_limit=0.0625 (default), scale_limit=0.1 (default), rotate_limit=(-30,30) degrees	Centre crop: cropping the centre square portion of the images of h x w =125*125.	Rotate, without shear shifting (40 degrees)	Horizontal flip: Flip the input horizontally around the y-axis.	Vertical flip: Flip the input vertically around the x-axis.	Shift and Scale (shift_limit= 0.3, scale_limit=0.2, rotate_limit=0). without rotations. on 3348 randomly chosen images from last augmentation	9012
8	Melanocytic Nevus	NV	9012	No Augmentations were applied as the training input is 12875 images. 70% of this size = 9012 images will be chosen for model building						9012
		Total	17728						Total	72096

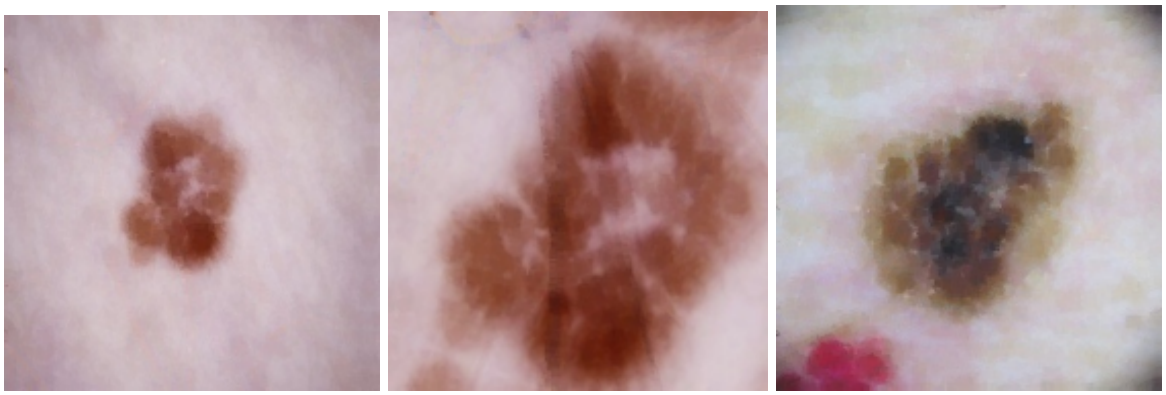
4. Image Pre-processing / Hair Removal

MEL:

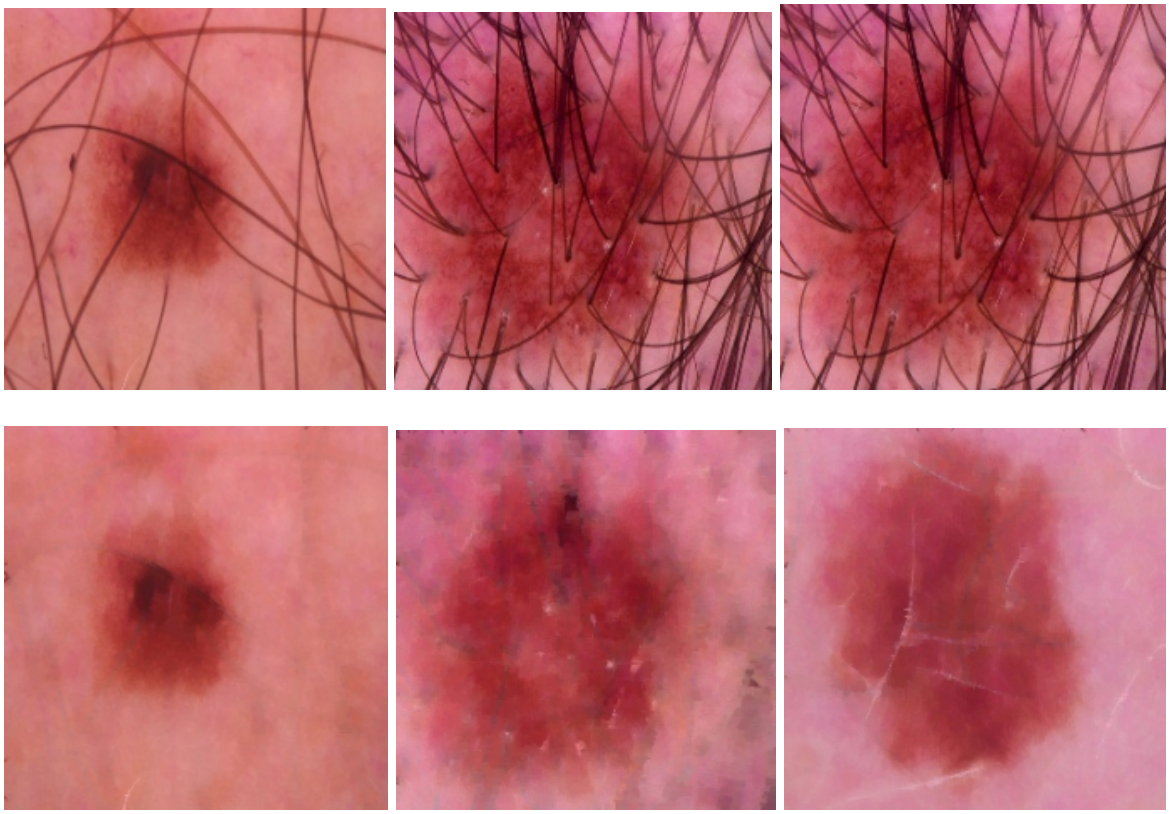


BKL:

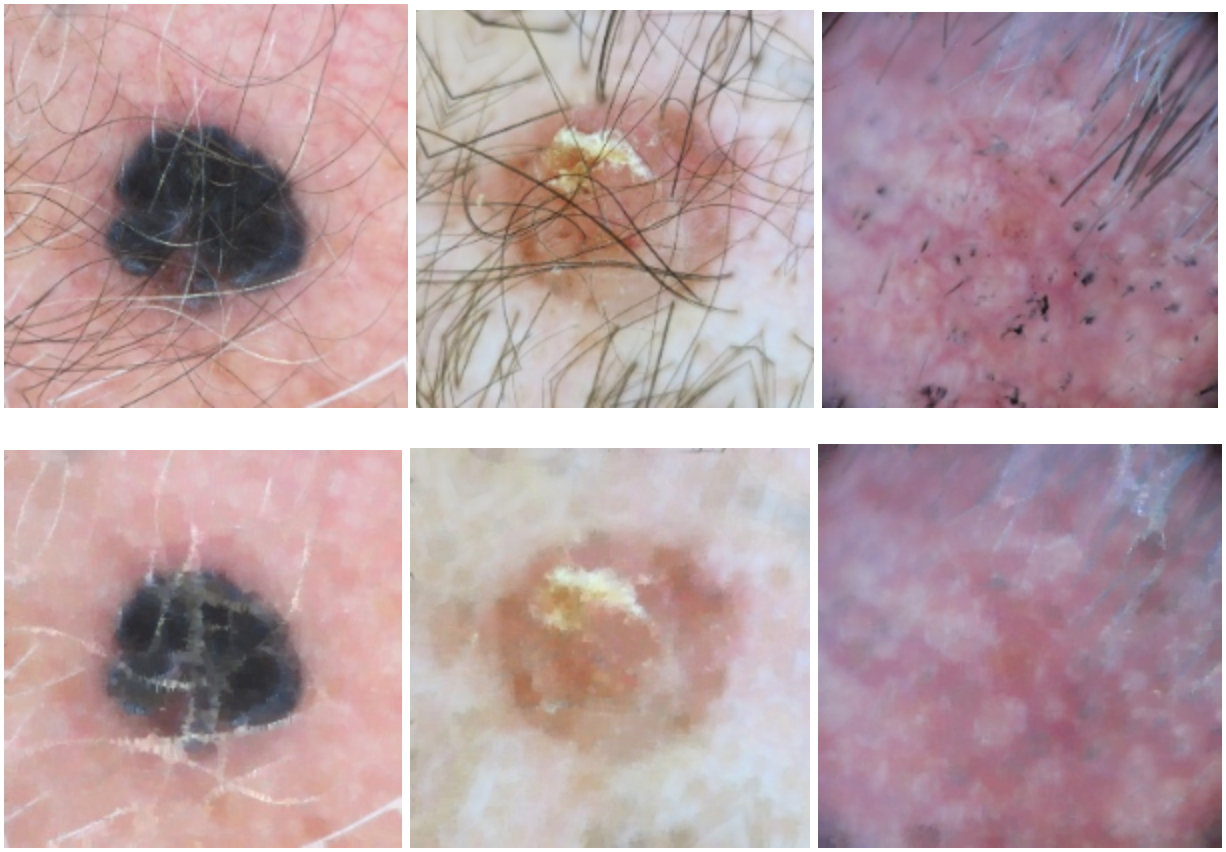




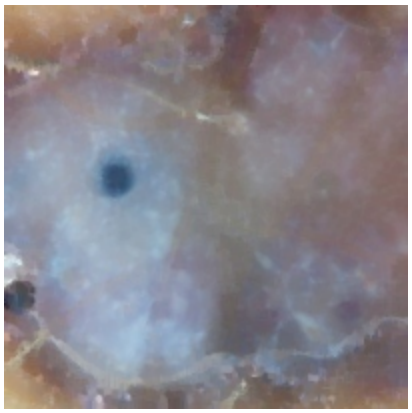
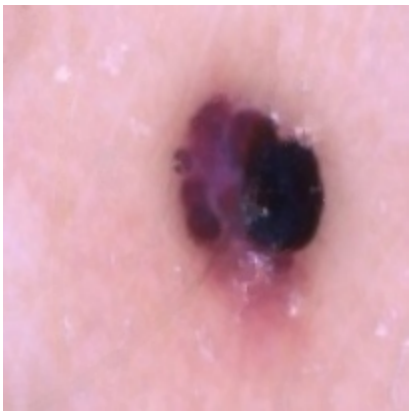
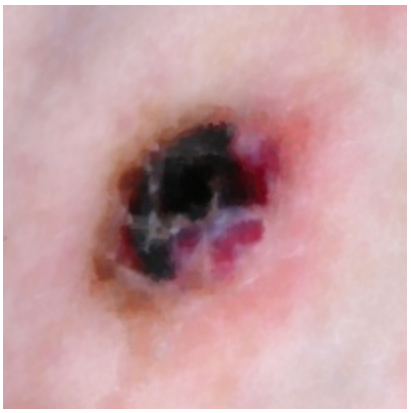
NV:



BCC:



VASC:

5. ISIC 2019 Leader board scores

ISIC 2019 Leaderboards

Group By Team

LESION DIAGNOSIS: IMAGES ONLY | LESION DIAGNOSIS: IMAGES AND METADATA

Rank <129 total>	Team <64 unique teams>	Approach Name	Manuscript	Used External Data <41 yes>	Primary Metric Value <Balanced Multiclass Accuracy>
1	DAISYLab Hamburg University of Technology/University Medical Center Hamburg-Eppendorf	Ensemble of Multi-Res EfficientNets + SEN1542		Yes	0.636
2	DAISYLab Hamburg University of Technology/University Medical Center Hamburg-Eppendorf	Skin Lesion Classification Using Multi-Resolution and Multi-Scale EfficientNet -based Ensembling		Yes	0.626
3	DysionAI DYSION AI, Inc, Beijing, China	Ensemble of EfficienetB3-B4-Seresnext101		No	0.607
4	DysionAI DYSION AI, Inc, Beijing, China	Ensemble of EfficienetB2-B3-B4-Seresnext101		No	0.604
5	DysionAI DYSION AI, Inc, Beijing, China	EfficienetB3-B4-Seresnext101-Seresnext50-Densenet121		No	0.601
6	AImageLab & PRHLT Unimore & UPV	ensemble, ood threshold 100%		No	0.593
7	AImageLab & PRHLT Unimore & UPV	ensemble, ood threshold 95%		No	0.584
8	DermaCode	13 models + hierarchical approach to select outliers		No	0.578
122	YouMe AI	Convolutional Neural Network with Class Weights - final		No	0.247
123	SkinLegion Persistent Systems Ltd	AnoGAN		No	0.147
124	LLCW Institute of Automation, Chinese Academy of Sciences	Resnet with Transfer Learning		Yes	0.113
125	IT Derm Lab DBE	Fine tune Resnet		No	0.109
126	IT Derm Lab DBE	Fine tune Resnet + mixup		No	0.109
127	jasdeep Singh	Asymmetrical loss function and cutout along the corners		Yes	0.108
128	TUKL NUST	ResNet50 ConvNet using cyclical learning rates and transfer learning		No	0.106
129	P	ResNet50		No	0.0476

6. Model Structures

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 128, 128, 16)	448
max_pooling2d (MaxPooling2D)	(None, 64, 64, 16)	0
conv2d_1 (Conv2D)	(None, 64, 64, 32)	4640
max_pooling2d_1 (MaxPooling2D)	(None, 32, 32, 32)	0
conv2d_2 (Conv2D)	(None, 32, 32, 64)	18496
max_pooling2d_2 (MaxPooling2D)	(None, 16, 16, 64)	0
dropout (Dropout)	(None, 16, 16, 64)	0
flatten (Flatten)	(None, 16384)	0
dense (Dense)	(None, 8)	131080

...
Total params: 154,664
Trainable params: 154,664
Non-trainable params: 0

Model 1: Base Model

Model: "sequential"

Layer (type)	Output Shape	Param #
keras_layer (KerasLayer)	(None, 1056)	4269716
dense (Dense)	(None, 8)	8456

Total params: 4,278,172
Trainable params: 4,241,434
Non-trainable params: 36,738

Model NasNetMobile

7. Confusion Matrices and Classification Reports

```
Confusion Matrix
[[ 58   1   0  13   3  11  44   0]
 [207   1   1  53   8  21 203   4]
 [146   0   0  48   4  14 180   2]
 [ 16   0   0   3   2   2  13   0]
 [248   1   2  85   8  37 297   0]
 [706   3   4 224  27 101 861   5]
 [ 38   0   0  12   1   1  42   0]
 [ 13   0   0   5   2   2  16   0]]
Classification Report
              precision    recall  f1-score   support

          ak       0.04      0.45      0.07       130
         bcc       0.17      0.00      0.00       498
         bkl       0.00      0.00      0.00       394
          df       0.01      0.08      0.01        36
         mel       0.15      0.01      0.02       678
          nv       0.53      0.05      0.10      1931
         scc       0.03      0.45      0.05        94
        vasc       0.00      0.00      0.00        38

    accuracy                           0.06      3799
   macro avg       0.11      0.13      0.03      3799
weighted avg       0.32      0.06      0.06      3799
```

Model 1

```
Confusion Matrix
[[ 57   1   0  17   4  10  41   0]
 [172   0   2  65  14  42 198   5]
 [130   1   1  80   6  36 137   3]
 [ 14   0   0   2   3   5  12   0]
 [216   1   2 114   9  68 267   1]
 [623   4   6 306  44 186 753   9]
 [ 38   0   0  15   1   9  30   1]
 [ 14   0   0   4   1   3  16   0]]
Classification Report
              precision    recall  f1-score   support

          ak       0.05      0.44      0.08       130
         bcc       0.00      0.00      0.00       498
         bkl       0.09      0.00      0.00       394
          df       0.00      0.06      0.01        36
         mel       0.11      0.01      0.02       678
          nv       0.52      0.10      0.16      1931
         scc       0.02      0.32      0.04        94
        vasc       0.00      0.00      0.00        38

    accuracy                           0.08      3799
   macro avg       0.10      0.12      0.04      3799
weighted avg       0.29      0.08      0.09      3799
```

Model 2

```
Confusion Matrix
[[ 10  18  14   4  26  54   1   3]
 [ 30  56  60   4 113 213  15   7]
 [ 26  58  45   3  87 149  18   8]
 [  6   4   3   0   8  14   1   0]
 [ 54 124  66  10 118 267  28  11]
 [162 284 208  26 396 776  53  26]
 [ 10  13  13   1  14  41   1   1]
 [  7   4   2   0   6  18   1   0]]
Classification Report
              precision    recall  f1-score   support

          ak       0.03      0.08      0.05       130
         bcc       0.10      0.11      0.11       498
         bkl       0.11      0.11      0.11       394
          df       0.00      0.00      0.00        36
         mel       0.15      0.17      0.16       678
          nv       0.51      0.40      0.45      1931
         scc       0.01      0.01      0.01        94
        vasc       0.00      0.00      0.00        38

    accuracy                           0.26      3799
   macro avg       0.11      0.11      0.11      3799
weighted avg       0.31      0.26      0.28      3799
```

Model 3

```
Confusion Matrix
[[ 40   0   0  36   8  21  23   2]
 [118   5   0 145  29 100  95   6]
 [ 84   4   1 118  24  77  84   2]
 [  9   0   1  12   6   5   3   0]
 [160   4   0 198  37 131 145   3]
 [433  12   5 550 126 414 378  13]
 [ 24   1   0  28   4  18  19   0]
 [  8   0   0  11   3  11   5   0]]
Classification Report
              precision    recall  f1-score   support

          ak       0.05      0.31      0.08       130
         bcc       0.19      0.01      0.02       498
         bkl       0.14      0.00      0.00       394
          df       0.01      0.33      0.02        36
         mel       0.16      0.05      0.08       678
          nv       0.53      0.21      0.31      1931
         scc       0.03      0.20      0.04        94
        vasc       0.00      0.00      0.00        38

    accuracy                           0.14      3799
   macro avg       0.14      0.14      0.07      3799
weighted avg       0.34      0.14      0.18      3799
```

Model NasNetMobile

Confusion Matrix

```
[[  4  10  28   1  27  57   1   2]
 [ 14  49  70   2  94 248  11  10]
 [ 11  59  63   3  66 178   8   6]
 [  3   4   4   1   7  17   0   0]
 [ 21  87  80  16 125 315  24  10]
 [ 72 212 277  18 377 918  38  19]
 [  3  11  11   4  18  42   5   0]
 [  4   2   6   0   5  20   0   1]]
```

Classification Report

	precision	recall	f1-score	support
ak	0.03	0.03	0.03	130
bcc	0.11	0.10	0.11	498
bkl	0.12	0.16	0.14	394
df	0.02	0.03	0.02	36
mel	0.17	0.18	0.18	678
nv	0.51	0.48	0.49	1931
scc	0.06	0.05	0.06	94
vasc	0.02	0.03	0.02	38
accuracy			0.31	3799
macro avg	0.13	0.13	0.13	3799
weighted avg	0.32	0.31	0.31	3799

Model 5: Xception with CW

Confusion Matrix

```
[[  50   76   88  284]
 [  57   39   78  220]
 [  98   61  123  396]
 [ 254  229  339 1109]]
```

Classification Report

	precision	recall	f1-score	support
bcc	0.11	0.10	0.10	498
bkl	0.10	0.10	0.10	394
mel	0.20	0.18	0.19	678
nv	0.55	0.57	0.56	1931
accuracy			0.38	3501
macro avg	0.24	0.24	0.24	3501
weighted avg	0.37	0.38	0.37	3501

Model 6: Xception with dominant classes

Confusion Matrix

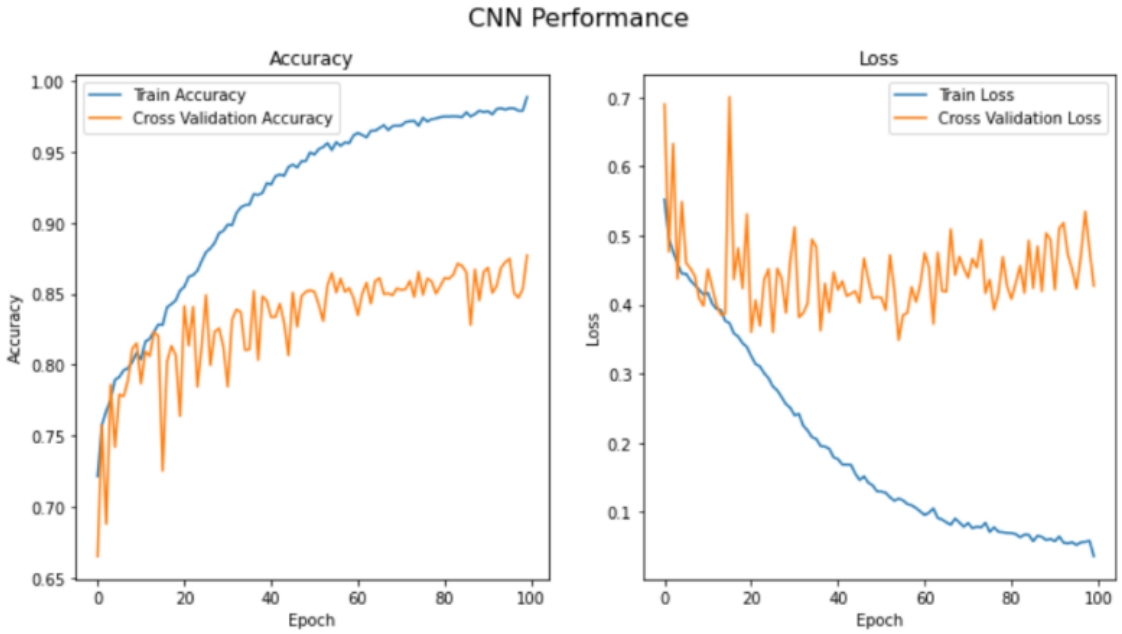
```
[[517 753]
 [538 732]]
```

Classification Report

	precision	recall	f1-score	support
Cancer	0.49	0.41	0.44	1270
Non-Cancer	0.49	0.58	0.53	1270
accuracy			0.49	2540
macro avg	0.49	0.49	0.49	2540
weighted avg	0.49	0.49	0.49	2540

Model 7: Xception – Binary Classification

8. Model 7: Xception Binary classifier – Training Performance



9. Xception Architecture

```
Model: "functional_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   Connected to
=================================================================
input_1 (InputLayer)         [(None, 128, 128, 3) 0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 63, 63, 32)   864         input_1[0][0]
_________________________________________________________________
block1_conv1_bn (BatchNormaliza (None, 63, 63, 32)   128         block1_conv1[0][0]
_________________________________________________________________
block1_conv1_act (Activation)   (None, 63, 63, 32)   0           block1_conv1_bn[0][0]
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 61, 61, 64)   18432       block1_conv1_act[0][0]
_________________________________________________________________
block1_conv2_bn (BatchNormaliza (None, 61, 61, 64)   256         block1_conv2[0][0]
_________________________________________________________________
block1_conv2_act (Activation)   (None, 61, 61, 64)   0           block1_conv2_bn[0][0]
_________________________________________________________________
block2_sepconv1 (SeparableConv2 (None, 61, 61, 128)  8768        block1_conv2_act[0][0]
_________________________________________________________________
block2_sepconv1_bn (BatchNormal (None, 61, 61, 128)  512         block2_sepconv1[0][0]
_________________________________________________________________
block2_sepconv2_act (Activation (None, 61, 61, 128)  0           block2_sepconv1_bn[0][0]
_________________________________________________________________
block2_sepconv2 (SeparableConv2 (None, 61, 61, 128)  17536       block2_sepconv2_act[0][0]
...
Total params: 25,188,528
Trainable params: 25,068,208
Non-trainable params: 120,320
```

10. Steps to reproduce preprocessing results for ISIC 2019

I. The preprocessing steps have been divided into 4 pipelines explained in 4 steps below. The successful completion of the preprocessing requires the successful execution of 4 python scripts one after the other.
II. Please follow the exact upper/ lower case for the folder names as mentioned in this document.
III. All python files will be placed outside the "Data" directory
IV. Classes representation: ["ak", "bcc", "bkl", "df", "Mel", "NV", "SCC", "vasc"]
V. All code files link:

https://github.com/Avi-k-dua/Melanoma_Detection/tree/master/ISIC_2019

VI. Data link: ISIC 2019

STEP 1: Training data download and class separation:

1. Download the ISIC 2019 training data (9.1GB), training metadata.csv (1MB), and training ground truth.csv (1MB) from the link https://challenge.isic-archive.com/data/
2. Place all the 25,331 training images into a folder named "orig". Or alternatively, rename the downloaded folder to "orig".
3. Place the "orig" folder and all the four python notebooks(.ipynb) files in a single directory (like "ISIC_2019").
4. Run the script "image_placement_2019.ipynb". This automatically creates the directories required for placing the training images into separate subfolders representing their respective classes.

Note: *Verify that after running the above script, a folder named Data gets created in the same directory.*

STEP 2: Train, Test, Validation Split:

Run the script "train_test_valid_split_2019.ipynb". This will automatically create the required directories and place the 25331 images belonging to 8 classes into the folder paths as described below:

Train – "Data/train_70%/"

Test – "Data/Processed_Data/test/"

Valid – "Data/Processed_Data/valid/"

The splitting of the training data will be performed in the following manner:

- 70% of the training data will be used for training. All preprocessing steps shall be applied to this data.
- 15% of the training data will be reserved for testing the model (unprocessed).
- 15% of the training data will be reserved for validation / making predictions on the model (unprocessed).

Note: The above-mentioned split is performed respective to individual class, after separating the classes in step 1. For example, out of 867 training images for class AK, 70% of images are split into training, 15% into the test, and 15% into valid. Likewise for all classes.

STEP 3: Training Augmentations:

1. Run the python script – "*ISIC_2019_Augmentations.ipynb*". This file contains the augmentations for all the 8 classes. This file takes input data from the 8 subfolders of "train_70%" and places the augmented files in the 8 respective subfolders of the "train_augmented".
2. The directories required for saving the augmented files will be automatically created from the script.
3. The summary of each augmentation applied in each class is described in "*ISIC_2019_augmentations_summary.xlsx*"

STEP 4: Hair Removal:

1. This preprocessing is performed on only 5 classes. After the augmentation step.
2. Manual action: Create a folder named "hair removal" in the "Data" directory and copy the following 5 subfolders from the augmentation step above(Data/train_augmented) and place them in the "Data/hair removal" folder: "train_bcc_aug" "train_bkl_aug", "train_mel_aug", "train_nv_aug", "train_vasc_aug"
3. Run the file "Hair_removal_2019.ipynb". It will perform the hair removal by taking input from the 5 classes placed in the "hair removal" folder and placing the results in the 5 folders inside the "Data/Processed_Data" (Automatically created through the script).
4. Manual action: Copy the remaining 3 subfolders "train_ak_aug" "train_df_aug", "train_scc_aug" from the "Data/train_augmented" and place them along with the 5 subfolders inside the "Data/Processed_Data/train" folder will form the training input for the model.

The "train", "test" and "valid" folders inside Data/Processed_Data containing 8 subfolders for each class thus obtained after the successful completion of the above preprocessing steps, shall be used for model building.