Skin Lesion Classification Using Different CNN Models

by

Md. Yahea Sultan Oli
19166012

A project submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
Master of Engineering in Computer Science and Engineering

Department of Computer Science and Engineering
Brac University
May 2023

Declaration

It is hereby declared that

1. The project submitted is my own original work while completing degree at Brac University.
2. The project does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The project does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. I have acknowledged all main sources of help

Student's Full Name & Signature:

Md Yahea Sultan Oli
19166012

Approval

The project titled "Skin Lesion Classification Using Different CNN Models" submitted by submitted by

1. Md. Yahea Sultan Oli (19166012)

of Spring, 2023 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of Master of Engineering in Computer Science and Engineering on May 25, 2023.

Examining Committee:

Supervisor:
(Member)

Dr. Amitabha Chakrabarty
Associate Professor
Computer Science and Engineering
Brac University

Program Coordinator:
(Member)

Dr. Amitabha Chakrabarty
Associate Professor
Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Sadia Hamid Kazi
Associate Professor
Department of Computer Science and Engineering
Brac University

Abstract

The field of dermatoscopic image classification has gained significant attention as there is a growing demand for early diagnosis of specific diseases. The use of deep learning is increasingly significant in the quest for a more effective dermoscopic analysis method. The "HAM10000" (Human Against Machine) dataset has been used in this study for classification of 7 different types of skin lesions by using DenseNet-121, VGG16, ResNet50, and Inceptionv3 model. To improve the classifier's performance, data augmentation was applied. This study could help dermatologists in the clinic make more precise decisions when identifying skin lesions, which would be beneficial. With this project I have tried to improve the model so that dermatologists identify skin lesions more precisely. Through the implementation of data augmentation techniques, this project achieved an impressive categorical accuracy of 92% and a top2 accuracy of 97% using DenseNet-121. The remaining models, VGG16, ResNet50, Inceptionv3 achieved accuracy 80%, 78%, 84% respectively. This project could have a beneficial impact on dermatoscopic image recognition and can reduce time and valuable resources. It can also help to saves life where robust diagnosing is not available.

Keywords: Cancer; Distinguished; Melanoma; Lesion; Densenet121; VGG16; Inceptionv3; ResNet50

Acknowledgement

I want to specifically thank my sir Dr. Amitabha Chakrabarty for giving the opportunity to work at this project "Skin Lesion Classification Using Different CNN Models". I have done a lot of research for having this opportunity and I have learned so many new things as a result. I truly appreciate him. I would also like to express my gratitude to my parents and friends, whose immense support played a crucial role in enabling me to complete this project within the given timeframe. I want to thank Dr. Amitabha Chakrabarty, for everything that he did to support me in achieving the goal. I must express my gratitude for the advise given by the panels and the other supervisors, especially when it came to my presentation of the project, his remarks and ideas improved my documentation skills.

Table of Contents

List of Figures

List of Tables

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

akiec Actinic keratosis

ANN Artificial Neural Network

bcc Basal cell carcinoma

bkl Benign keratosis

CNN Convolutional Neural Network

df Dermatofibroma

DNN Deep Neural Networks

mel Melanoma

nv Melanocytic nevi

vasc Vascular lesions

Chapter 1

Introduction

Deep learning is an innovation that empowers machines to distinguish designs in huge measures of "solo" information and to perceive pictures or discourse. Machines are capable of preparing administered information - for instance, the information that engineers physically contribute to a framework to empower it to comprehend spoken orders. The objective of Deep learning is to make machines that learn without the requirement for human info, for example, by distinguishing the words verbally expressed in large numbers of recordings.

1.1 Problem Statement

Skin cancer is widely recognized as one of the most prevalent types of cancer. [33]. Due to its exceedingly poor chance of survival, Melanoma is considered as one of the deadliest types of skin cancer [24]. Since skin lesions are the first sign of skin cancer, early identification could increase survival rates while lowering treatment costs because skin cancer treatments are highly expensive.

A skin lesion refers to an area of the skin that has experienced abnormal growth and has the potential to develop into skin cancer. Different skin lesions may appear different to the human eye due to variations in the lesion's expanding length and color. However, accurate identification of diseases cannot be achieved with this observation alone. To accurately detect skin lesions, dermoscopy was previously used. However, it was time-consuming. The Artificial Neural Network (ANN) was commonly used for generating responses based on medical data during the early years of the twenty-first century. This was mainly due to its faster and more straightforward nature [16]. However, the lack of hardware units made ANN an ineffective technique. Convolutional Neural Networks (CNN) are among the Deep Learning techniques that have a significant impact on information analysis and medical data, where picture data is preferable to artificial neural networks [21].[21]. Early skin lesion identification has been the subject of some prior research, with generally excellent results. My goal is to enhance the precise identification and classification of seven types of skin lesions by using the more accurate models (DenseNet-121, VGG16, ResNet50, and Inceptionv3).

1.2 Objectives

The HAM10000 data set was used throughout the project's work [7]. The data includes 7 classes. I first pre pre-processed all the images and split them into train and validation sets for further work. Data augmentation was performed to alter the training set because the data set was highly unbalanced. The training-validation loss and training-validation accuracy curves were calculated for analyzing the model. Categorical accuracy, precision, recall and F-1 score were computed to for further insight of the model. The objective of the work is to:

- Address class imbalance problem using data augmentation techniques
- Try to classify skin lesions using a robust deep learning approach.

1.3 Project Structure

Following is how the rest of the theory is coordinated

Chapter-2: Background Study
I am going to discuss thoroughly about the backbone of this project work which is convolutional neural networks and how it's various layers work at this chapter.

Chapter 3: Methodology
This chapter reflects how the total experiment was done and which manner it was evaluated and approached. Here, how the dataset was pre-processed, trained and evaluated. Here I worked with skin lesion images and tried to make the result more robust.

Chapter 4: Result analysis
This chapter shows the gained result and its comparison with different models.

Chapter 5: Conclusion and Future Works
Here in this chapter, the summary of the whole project will be presented where all the limitations and contributions will be presented. Also, future scope of works will be there.

Chapter 2

Background Study

In this chapter, Various aspects of convolutional networks are going to be discussed and also relevant pre-trained architectures details will be presented

2.1 Related Works

Medical image segmentation using deep learning has been around for a while. Hardware advancements over time have made it simpler for hospitals all over the world to use. For this procedure, convolutional neural networks (CNNs) are used. While CNNs function similarly to a standard feedforward neural network, they are considerably better suited to handle images because they combine several approaches like convolutions, max-pooling, etc.

By using a 2D input image and applying 2D filters to it (done with a 2D CNN) is the main concept behind using CNNs. A different strategy is to use transfer learning, where models are trained using pre-trained cutting-edge models and the final few layers are frozen to acquire weights particular to the problem. The majority of the low-level features come from ImageNet. Using 2.5D CNN, which can handle some spatial information, is another novel strategy. They are able to balance computational costs and performance well. In comparison to a 2.5 CNN, 3D CNNs offer even higher performance and handle richer spatial information. Deep learning algorithms have exhibited remarkable performance in visual tasks, surpassing even human capabilities in certain video games, such as Go [19], Atari [14] and object identification [15], This has prompted researchers to explore the feasibility of automated skin cancer screening. A number of research have been conducted to compare the classification of skin cancer made by dermatologists and that made using Deep Learning [2]. In a benchmark study conducted by Esteva et al., where 129,450 clinical photos were compared, it was discovered that the CNN model performed at a comparable or superior level to dermatologists [29]. Deep Neural Networks (DNNs), which were developed to increase accuracy over earlier models, have been more popular in recent years [2]. DNNs have a compelling effect on image classification, but using them to classify medical images is difficult since they need a lot of training data [7]. The majority of the existing literature uses transfer learning, a method that reuses a model created for one task for another, to handle challenges with enormous datasets. We have covered current techniques and works that are

connected to them in this chapter. To achieve the best results and improve artificial intelligence's ability to recognize, segment, and classify a wide range of variances in medical images, many strategies were used.

2.2 Deep Learning

Given the iterative nature of computations involving profound learning, their complexity as the number of layers increases, and the vast amounts of information needed to create the organizations, a significant amount of computational power is anticipated to be applied to these problems. The dynamic nature of deep learning strategies—their potential to constantly advance and adapt to changes in the hidden data architecture—presents a fantastic opportunity to include more original behavior's into examination. One possibility is to more strongly personalize client examination. Another fantastic opportunity is to increase precision and performance in applications that have been using neural organizations for a while. We can add more pronounced profundity by using better calculations and more calculating power[17].

Most deep learning models are supported by artificial neural networks, a type of cutting-edge AI calculation. Deep learning may therefore occasionally be mentioned in the context of profound neural learning or profound neural systems management. Repetitive neural organizations, convolutional neural organizations, fake neural organizations, and feed-forward neural organizations are only a few of the distinctive forms of neural organizations. Each has benefits for particular use cases. Though they all manage information and let the model decide whether it has made the correct translation or choice for a certain information component on its own, they all operate in rather similar ways.

Neural organizations have an experimental cycle, thus they require enormous amounts of data to prepare. It's no accident that neural organizations gained popularity when the majority of endeavors mastered in-depth information analysis and gathered vast amounts of data. The information used during the preparation stage must be labelled because the model's initial not many cycles comprise to some extent informed guesses on the content of a picture or grammatical features. This allows the model to check whether its estimate was accurate [13]. However, this suggests that while many projects that employ a lot of data have a lot of data, unstructured data is less helpful. Unstructured data must be prepared and have a sufficient level of precision before being studied by a profound learning model; however, profound learning models cannot prepare on unstructured data.

2.3 Supervised Learning

Supervised learning techniques are utilized to construct a mathematical model that represents a dataset containing input and output information. [4] This dataset, known as training data, comprises examples with desired outputs (supervisory signals) and corresponding inputs. The training data is represented as a matrix at mathematical model, where every training sample is shown as a vector or an array,

generally referred to be a feature vector. Supervised learning techniques provide a prediction function capable of producing outputs for fresh inputs through iterative optimization of an objective function.[8] By employing an optimal function, the algorithm can accurately forecast outcomes for inputs that were not included in the training data. The learning process is considered successful when the algorithm's output or prediction accuracy improves over time.[1]

Supervised learning algorithms include various techniques, including regression classification. [3] While classification methods are used when outputs are constrained to a predetermined range of values, regression methods are appropriate for situations where outputs can take any numerical value within a given range. The main goal of similarity learning is finding the similarity or link between two items, a supervised machine learning's subfield known as regression and classification. It leverages instances to learn and can be applied to tasks such as speaker verification, visual identification tracking, rating systems, recommendation systems, face recognition, and voice recognition.

2.4 Convolutional Neural Network

Convolutional neural networks (CNNs) resemble conventional neural networks in many ways. CNNs are made with neurons and with trainable weights and biases, just as their analogues. Every neuron receives input, computes dot product, and, if desired, adds a nonlinearity. Starting with the original image pixels and concluding with class scores, the complete network continues to operate as a differentiable score function. Similar to normal neural networks, CNNs contain a loss function in the final (fully connected) layer, such as SVM/Softmax. The techniques and strategies developed for learning regular neural networks remain applicable in this context.

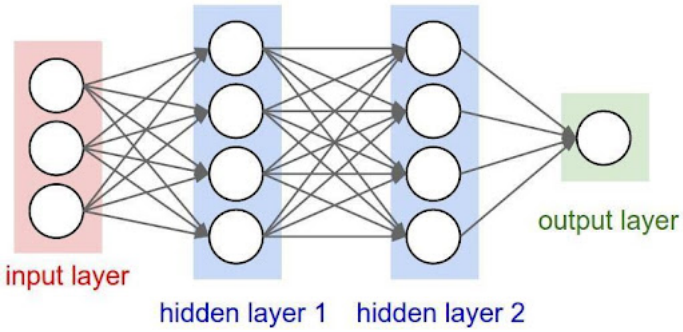


Figure 2.1: Simple CNN architecture [38].

A single vector is the input for regular neural networks, which process it through a series of hidden layers. Each set of neurons in a hidden layer is connected to every other neuron in the layer above it. One layer's worth of neurons each function independently and do not exchange connections. In classification settings, the "output layer"—the last completely linked layer—provides the class scores [20].

Regular Neural Networks have difficulties scaling up to process full-sized images, as we have already seen. A single fully connected neuron in CIFAR-10's initial

hidden layer, which uses images that are only 32x32x3 (32 wide, 32 height, and 3 color channels), would need 3072 weights. Even though it can appear manageable, this amount becomes useless when working with larger images. For instance, neurons with 120,000 weights would be produced by an image with a larger size, such 200x200x3. Furthermore, it would be advantageous to have several of these neurons, which would quickly amass an excessive amount of parameters. The fully integrated structure is obviously ineffective and causes overfitting.

Convolutional Neural Networks overcome these difficulties by imposing a more logical architectural design and utilizing the input's image-based nature. ConvNet layers arrange neurons in three dimensions—width, height, and depth—in contrast to standard neural networks. Instead of the depth of the entire neural network, depth in this context indicates the third dimension of activation volume. For instance, the input photos for CIFAR-10 create a volume of activations that has the following measurements: 32x32x3 (width, height, and depth, respectively). Notably, the neurons in a layer don't fully connect with all the neurons in the layer above; instead, they only link to a small portion of those neurons. Additionally, the CIFAR-10 final output layer would be 1x1x10 in size. By reducing the size of the image, the ConvNet architecture is able to turn the entire image class scores along the depth dimension into a single vector. [22].

2.5 Layers in CNN

A basic ConvNet consists of a series of layers, where each layer transforms one activation volume into another using a differentiable function. To construct ConvNet architectures, three essential types of layers are required: Convolutional Layer, Fully Connected Layer and Pooling Layer which are the same to those found in regular Neural Networks. By stacking these layers together, a complete ConvNet architecture is formed.

2.5.1 Convolutional Layer

The Conv layer does the majority of the computational labor and acts as the main building component of a Convolutional Network. Each filter in its set of parameters is a learnable one, with dimensions that are tiny (height and width) but cover the full depth of the input volume. For instance, a common first-layer ConvNet filter can be 5x5x3 in size (5 pixels wide and tall, and 3 for the input's color channels). At the time of forward pass, each filter is convoluted throughout the input volume's width and height, and computing dot products between each entry's corresponding input at each point.

Intuitively, the network learns filters that are activated in the first layer whenever they recognize particular visual elements, such as edges of different orientations or color patches. These filters might be able to identify more complicated patterns, such honeycomb or wheel-like formations, as the network advances to higher levels. There are a number of filters (for example, 12 filters) in each Conv layer, and each filter generates a unique 2-dimensional activation map. To create the output volume,

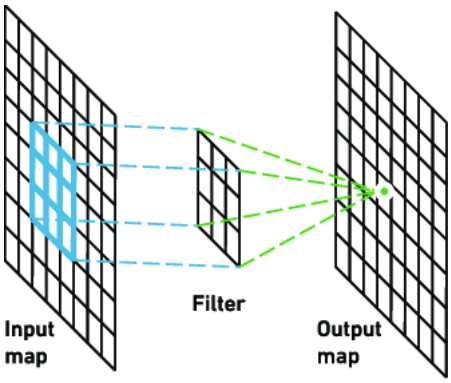


Figure 2.2: Convolution layer example[40].

these activation maps are stacked along the depth axis. As a result, while preserving full connectivity throughout the depth (all color channels), convolutional layer's each neuron is connected locally to a location inside the input volume in terms of spatial proximity. Along the depth dimension, several neurons (in this case, five) concentrate on the same input region.

Notably, the Conv layer and the neurons continue to use same method of computation as that discussed in the chapter on neural networks: they compute a non-linearity, then a dot product of their weights with the input. However, their connectivity is now limited to a small spatial area.

2.5.2 Activation Function

Mathematical Equations called activation functions are used to control output of a neural network's. By being associated with every neuron in the network, here a neuron should be activated ("fired") or remain inactive these functions decide that, for the model's prediction based on the significance of its input. Additionally, in bringing each neuron's output activation functions helps within a range between 1 and 0, or between -1 and 1.
Since, activation functions are computed across thousands or millions of neurons for each data sample, they also need to be computationally efficient. Modern neural networks train their models using a method called backpropagation, which puts more computational strain on the activation function and its derivative function.

Some of the activation functions are:

ReLU (Rectified Linear Unit)

Advantages

- **Efficient in computation** —For being computationally efficient, this feature enables the network to rapidly converge.
- **Non-linear**—Despite its appearance as a linear function, ReLU is actually non-linear and possesses a derivative function, enabling effective backpropagation.

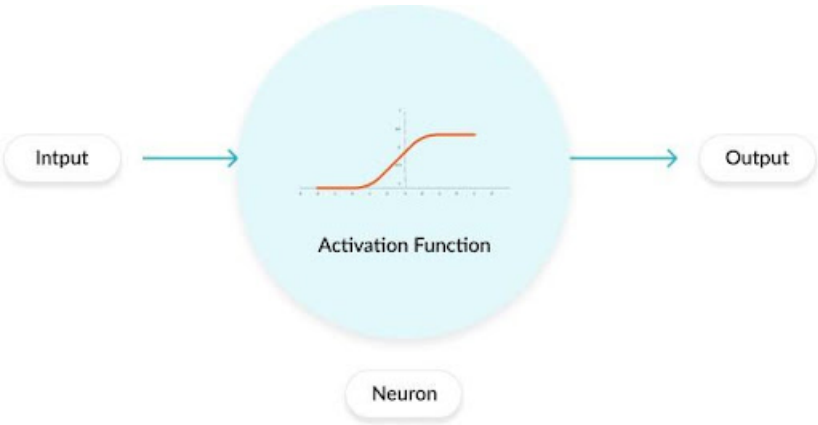


Figure 2.3: Activation Function[36].

Disadvantages

- **The Dying ReLU problem**—This problem arises when inputs become zero or negative, resulting in the function's gradient being zero. Consequently, the network faces difficulties in performing backpropagation and is unable to learn.

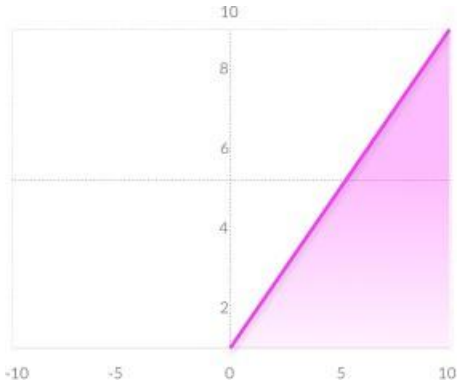


Figure 2.4: ReLU Function[36].

Softmax

- **Able to handle multiple classes** Unlike other activation functions that can only handle one class, softmax activation function is capable of handling multiple classes. It achieves this by normalizing the outputs for each class within the range of 0 and 1 and dividing them by their sum. This process yields the probability of the input value belonging to a specific class.
- **Useful for output neurons**—Softmax activation function is particularly beneficial for output neurons. It is commonly employed exclusively in the output layer of neural networks that require classification of inputs into multiple categories.

2.5.3 Dropout layer

In the context of fully connected layers, the initial dropout was covered. But dropout in convolutional layers is hardly seen. There are some debates about the dropout

effects in convolutional neural networks. Some people think dropout should not be used in convolutional layers because convolutional layers have fewer param- eters and are less likely to overfit. Randomly destroying nodes will slow down the training process because the gradient updates for the weights of convolutional layers are the average of all the gradients from all the convolutions. [22] Dropout does pro- vide regularization for any kind of neural network architectures.

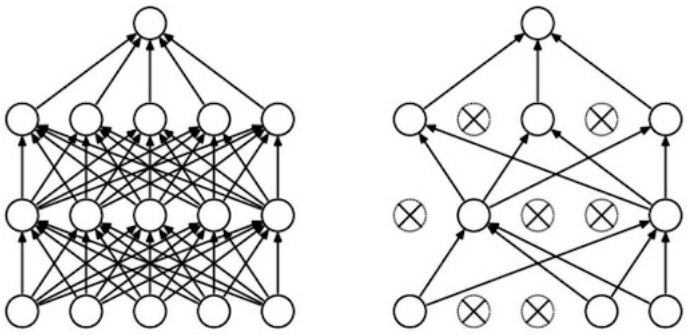


Figure 2.5: Dropout effect [12].

It also provides additional benefits from the perspective of Bayesian learning, which I might discuss in the future. But in practice, depending on the task, dropout may or may not affect the accuracy of your model. If you want to apply dropout in convolutional layers, just make sure that you test the training with and without dropout to see if it makes a difference.

2.5.4 Pooling Layer

The layer inserted after the convolutional layer is known as the pooling layer. More precisely, once a nonlinearity is used to implement the feature maps (such as ReLU) that a convolutional layer generates. In a model, the layers might look like this:

1. Input Image
2. Convolutional Layer
3. Nonlinearity
4. Pooling Layer

After convolutional layer, a pooling layer is typically included in the layer ordering of convolutional neural networks. This pattern might show up one or more times in a particular model.

In order to generate a fresh set pooled feature maps with the same number, on each feature map the pooling layer separately operates.

When deciding on a pooling operation for feature maps, the process is akin to selecting a filter. Typically, a common choice is a 2x2 pooling operation applied with a stride of 2 pixels. smaller pooling operation or filter size compared to the feature

map includes in this results.

This indicates each feature map is continuously reduced by a factor of 2, effectively halving each dimension, by the pooling layer. As a result, each feature map has only one-fourth of the original size. For example, a feature map of size 66 (36 pixels) would yield an output pooled feature map of size 33 (9 pixels) when a pooling layer is applied.

Instead of being taught, the pooling operation is defined. In the pooling operation, there are two often utilized functions:

- **Average Pooling:** Average pooling involves computing the average value within each patch or region of the feature map.
- **Maximum Pooling (or Max Pooling):** This technique involves identifying the maximum value within each patch or region of the feature map.

Using a pooling layer and down sampling or pooling feature maps, the input's retrieved features are condensed into a summary form. Because of this, even little changes in a feature's location that the convolutional layer detects will still be kept in the pooled feature map. This trait, called invariance to local translation, makes the model better at identifying features regardless of where exactly they are inside the input.[29].

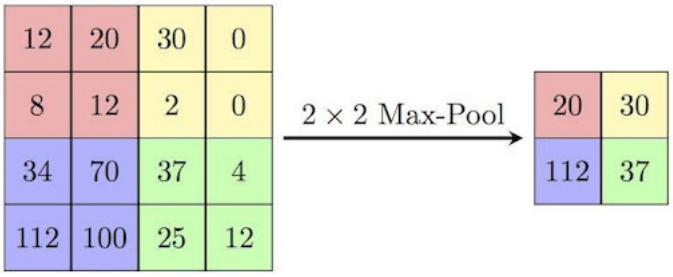


Figure 2.6: Max pooling layer [39].

2.5.5 Fully Connected Layer

Fully connected neural networks (FCNNs) are a particular architecture of artificial neural networks, wherein each node or neuron in one layer is linked to every neuron in the subsequent layer.

Although FCNNs are widely used for certain data types, they encounter challenges in tasks like image recognition and classification. These networks can be computationally demanding and susceptible to overfitting. Furthermore, when FCNNs are deep, meaning they possess numerous layers of neurons, they can become notably difficult for humans to comprehend.

Depending on the number of classes in the classification issue, the network's last fully connected layer generates the net output using a softmax function as an activation function. These kinds of various components make up a standard CNN

architecture. A typical architecture would look somewhat like this:

[(CONV-RELU) *N-POOL?] *M-(FC-RELU) *K, SOFTMAX (2.1)

where N is usually up to ˜5, M is large, 0 <= K <= 2.

2.5.6 Flatten and GlobalAveragePooling2D layer

A flatten layer condenses the input's spatial dimensions into the channel dimension. For instance, if the layer receives an H-by-W-by-C-by-N-by-S array as input (sequences of images), the output will be a flattened (H*W*C)-by-N-by-S array. Only sequence input is supported by this layer. The "Average Pool" function calculates the average value from the kernel size and moves by the stride value. comparable to Max Pool, which we have used several times and which takes the maximum value inside a particular kernel size. Therefore, the kernel size for the global average pool is H x W. Therefore, for H x W x C input, it calculates the global average over Height and Width and outputs a tensor with dimensions of 1xC.

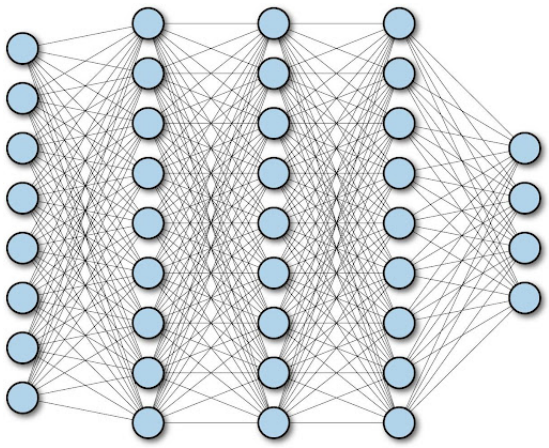
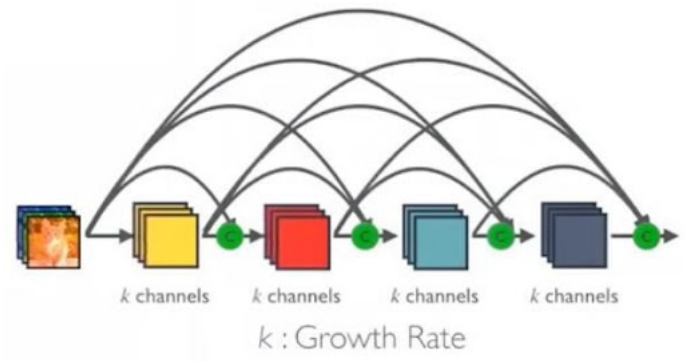


Figure 2.7: Fully Connected Layer [37].

2.6 DenseNet - 121

Convolutional neural networks are well known for efficiently analyzing picture data. The feature duplication issue in convolutional neural networks has recently come to light. This replication has been greatly decreased because to DenseNet [14]. It receives greater supervision from each tier since the dense blocks have closer connections [23]. We used the DenseNet-121 Architecture for the investigation. 121 here denotes the ImageNet Model's depth.

Every layer in DenseNet takes input additionally from all lower layers and sends feature maps of its own to all higher layers. Through concatenation this is achieved, allowing each layer to benefit from the "collective knowledge" accumulated from the preceding levels.



Plot

Figure 2.8: Dense Block in DenseNet with Growth Rate [32].

Direct propagation of the error signal to prior layers is simple. Because the final classification layer can directly supervise earlier layers, this is a type of implicit deep supervision. DenseNet has a number of benefits, including:

- It uses fewer parameters than CNN because
  - Redundant feature maps don't need to be learned again.
  - Densenet effectively differentiates between the information retained within the network and the information contributed to the network.
- Enhance gradients and information flow across the network, making training easier
  - Implicit deep supervision
  - Also has regularizing effects

These are somewhat modified versions of DenseNet. "Bottleneck" is the meaning of the "B" in DenseNet-B. This is as a result of the addition of a bottleneck layer, which is actually an 11 convolution layer. This means that the composite function will now be BN-RELU-1x1conv-3x3conv rather than BN-RELU-3x3conv. To make the feature map smaller before performing 33 conversion, this is done. According to the study, it is more computationally efficient.

The letter "C" stands for compression. The model can either maintain the amount of feature maps from one dense block to the next at transition layers or merely alter the resolution. The transition layer, however, might cut down on the number of feature maps by a factor in order to make the model more compact. The letter "C" is added if this strategy is used. When both of the aforementioned variants are used, DenseNet-BC is the model [28].

Layers	Output Size	DenseNet-121	DenseNet-169	DenseNet-201	DenseNet-264
Convolution	112 × 112	7 × 7 conv, stride 2			
Pooling	56 × 56	3 × 3 max pool, stride 2			
Dense Block (1)	56 × 56	[1 × 1 conv, 3 × 3 conv] × 6	[1 × 1 conv, 3 × 3 conv] × 6	[1 × 1 conv, 3 × 3 conv] × 6	[1 × 1 conv, 3 × 3 conv] × 6
Transition Layer (1)	56 × 56	1 × 1 conv			
	28 × 28	2 × 2 average pool, stride 2			
Dense Block (2)	28 × 28	[1 × 1 conv, 3 × 3 conv] × 12	[1 × 1 conv, 3 × 3 conv] × 12	[1 × 1 conv, 3 × 3 conv] × 12	[1 × 1 conv, 3 × 3 conv] × 12
Transition Layer (2)	28 × 28	1 × 1 conv			
	14 × 14	2 × 2 average pool, stride 2			
Dense Block (3)	14 × 14	[1 × 1 conv, 3 × 3 conv] × 24	[1 × 1 conv, 3 × 3 conv] × 32	[1 × 1 conv, 3 × 3 conv] × 48	[1 × 1 conv, 3 × 3 conv] × 64
Transition Layer (3)	14 × 14	1 × 1 conv			
	7 × 7	2 × 2 average pool, stride 2			
Dense Block (4)	7 × 7	[1 × 1 conv, 3 × 3 conv] × 16	[1 × 1 conv, 3 × 3 conv] × 32	[1 × 1 conv, 3 × 3 conv] × 32	[1 × 1 conv, 3 × 3 conv] × 48
Classification Layer	1 × 1	7 × 7 global average pool			
		1000D fully-connected, softmax			

Figure 2.9: Convolutional kernel sizes and output for various DenseNets on ImageNet. [30].

Contrary to intuition, DenseNets require fewer parameters than equivalent traditional CNNs by utilizing connections in a unique manner, need to learn redundant feature maps' eliminating. Additionally, certain forms of ResNets have demonstrated that numerous layers contribute minimally and can be discarded. Unlike ResNets, which possess a large parameter due to each layer having its own weights to learn, DenseNet layers are comparatively narrow, typically consisting of only 12 filters, while introducing a small set of new feature maps. Furthermore, the training difficulties associated with deep networks, attributed to information and gradient flow, are resolved by DenseNets. Each layer in DenseNets directly accesses gradients from the loss function and the original input image, mitigating these issues [26].

2.7 VGG16

The VGG16 model, developed by the Visual Geometry Group (VGG) at the University of Oxford, is a renowned convolutional neural network architecture that has gained significant popularity. It was first mentioned in the Simonyan and Zisserman study called "Very Deep Convolutional Networks for Large-Scale Image Recognition" from 2014. The VGG16 model outperformed the AlexNet model in the ILSVRC-2014 competition, which featured VGG16 prominently. This was accomplished in the first and second convolutional layers rather than bigger kernel sizes (11 and 5) by using several cascaded 33 kernel-sized filters. It took several weeks to train VGG16, and the acceleration was provided by NVIDIA Titan Black GPUs [31] At ImageNet dataset, which has 14 million images and those are divided into 1000 classes, also the model scored a top-5 test accuracy of 92.7%.

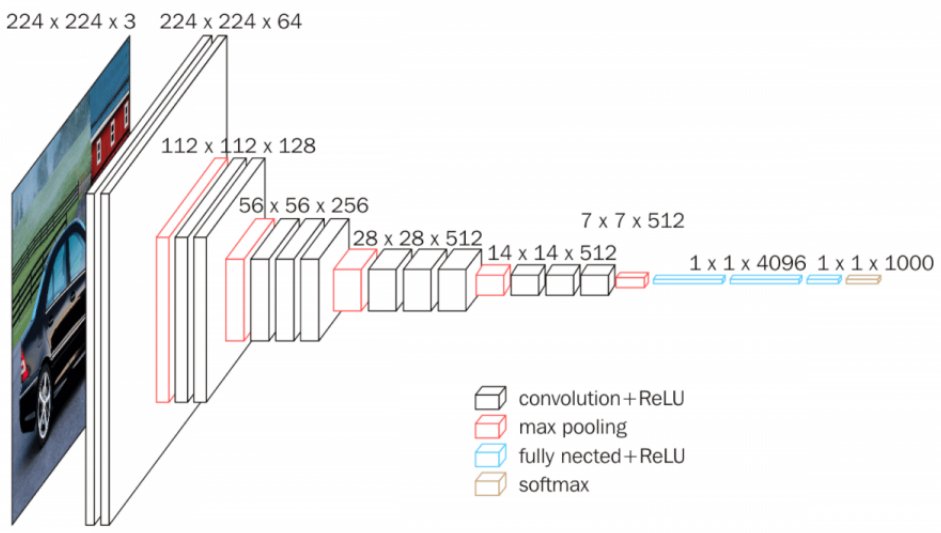


Figure 2.10: VGG-16 architecture

The Architecture: An image with the dimensions (224, 224, 3) is the network's input. The first two layers employ the same padding and have 64 channels with a 3x3 filter size. Following that, two convolutional layers are employed with a filter size of 128 and (3, 3), along with a subsequent max pooling layer with a stride of (2, 2). The same stride is used in another max pooling layer. Subsequently, there are two sets of convolutional layers, each comprising 256 filters and a filter size of (3, 3). This is followed by two sets of three convolutional layers, each consisting of 512 filters of size (3, 3), and a max pooling layer. Throughout these layers, the same padding technique is applied. The final step is running the resultant image through a stack of two convolutional layers. Instead of the larger 11x11 in AlexNet or 7x7 in ZF-Net, 3x3 filters are used across these convolutional and max pooling layers. Additionally, the number of input channels is adjusted using 1x1 pixels. To maintain the spatial details of the image, a 1-pixel padding (the same padding) is placed after each convolutional layer [34]

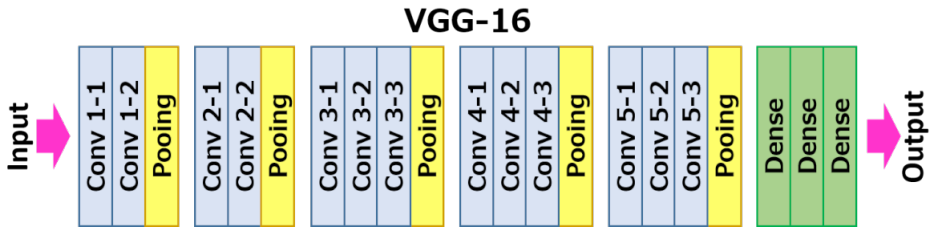


Figure 2.11: VGG-16 architecture Map

A (7, 7, 512) feature map is produced after the stack of convolutional and max pooling layers. To create a feature vector of size 1x25088, the resulting feature map is flattened. Then, three fully connected layers are utilized. Here the first layer takes produces a vector of size 1x4096 and the flattened feature vector as input. Similarly, the second layer generates another vector of size 1x4096. Finally, the third layer consists of 1000 channels, corresponding to the 1000 classes of the ILSVRC challenge. The softmax function is applied in the third fully connected layer to classify the input into the respective 1000 classes. Due to ReLU's computational effectiveness,

accelerated learning, and capacity to address vanishing gradient issues, it is utilized as the activation function in all hidden layers.

Configurations: The ConvNet configurations, depicted in figure 2, are named as A, B, C, D, and E. These configurations share a common architectural design and primarily differ in their depth. Here Network A consists of 11 weight layers (8 convolutional layers and 3 fully connected layers), while network E comprises 19 weight layers (16 convolutional layers and 3 fully connected layers). The convolutional layers have a relatively small width, starting from 64 channels in the first layer and doubling in size after each max-pooling layer, until reaching a final width of 512 channels.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure 2.12: ConvNet Configuration

2.8 ResNet-50

ResNet50 is a deep neural network architecture that has made significant contributions to the field of computer vision. It was introduced in 2015 by researchers at Microsoft Research Asia as a way to address the problem of vanishing gradients in very deep neural networks. In ResNet the "Res" stands for "residual", which refers to the residual connections used in the network.

ResNet-50 Architecture: ResNet50 consists of 50 layers, including convolutional layers, batch normalization layers, and fully connected layers. The architecture is based on a series of residual blocks, each of which contains two or more convolutional layers and a residual connection. The residual connection allows the network to learn the difference between the output of the convolutional layers and the input to the block, rather than learning the entire transformation from scratch.

While the regular network in ResNet is inspired by the VGG neural networks (such as VGG-16 and VGG-19), ResNet utilizes fewer filters and is less complex than VGGNet. For example, a 34-layer ResNet achieves high performance with significantly fewer floating point operations (FLOPs) compared to a VGG-19 network.

In addition to the residual connections, ResNet50 also uses global average pooling, which helps to reduce overfitting by producing a fixed-size output regardless of the input size. The network also includes a softmax activation function at the output layer, which is commonly used for multiclass classification problems. This architecture has been used for other deep neural network architectures, including ResNet101 and ResNet152.

Two essential design concepts are followed by the ResNet architecture. First, no matter how big the output feature map is, the number of filters in each layer stays constant. Second, to maintain the time complexity of each layer, the number of filters is doubled when the size of the feature map is cut in half. [35]

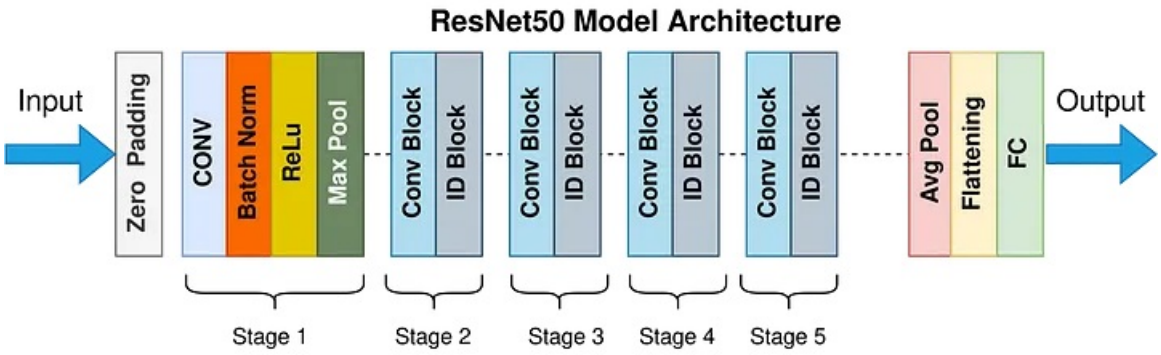


Figure 2.13: Resnet-50 Model architecture

2.9 Inception v3

Inception V3, a convolutional neural network originally developed as a module for GoogLeNet, serves as a valuable tool for object detection and image analysis. It is the third iteration of the Google Inception Convolutional Neural Network, initially introduced in the context of the ImageNet Recognition Challenge. Inceptionv3 was specifically designed to facilitate the construction of deeper networks while keeping the number of parameters manageable, boasting "under 25 million parameters" compared to AlexNet's 60 million.

In the realm of computer vision, Inception greatly contributes to object classification, similar to how ImageNet functions as a comprehensive database of categorized visual objects. Many applications have successfully employed the Inceptionv3 architecture, often leveraging pre-trained data from ImageNet. Notably, it has found application in leukemia research within the field of life sciences.

Interestingly, the original name "Inception" was given as a code name due to the widespread popularity of a viral internet meme referencing the famous line "we need to go deeper" from Christopher Nolan's film, Inception.

The Inception V3 model has 42 layers, which is a little more than the Inception V1 and V2 models. Despite its increased depth, the Inception V3 model exhibits impressive efficiency. In the following discussion, we will delve into the details of the components that comprise the Inception V3 model.

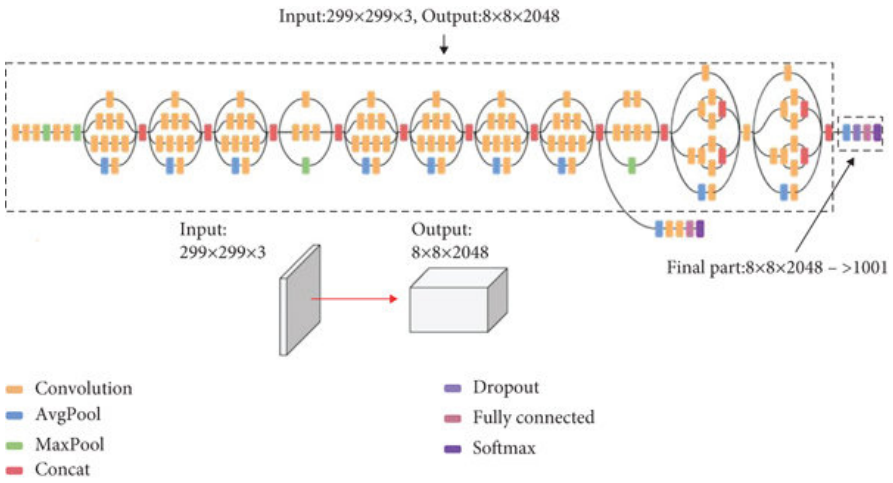


Figure 2.14: Architecture of Inception v3

Chapter 3

Methodology

Every scientific work is a flow of steps that is done in a sequence to get the most logical and corrected result. In this experiment, the following steps shown in the fig 3.1 were followed. We started with loading the data into the system and doing explanatory analysis of the data. Then cleaning the corrupted images and doing data augmentation. Then comes normalization of the image pixels and splitting the model into test and train sets. Finally, it is given into the model and then evaluated the model with various matrices.

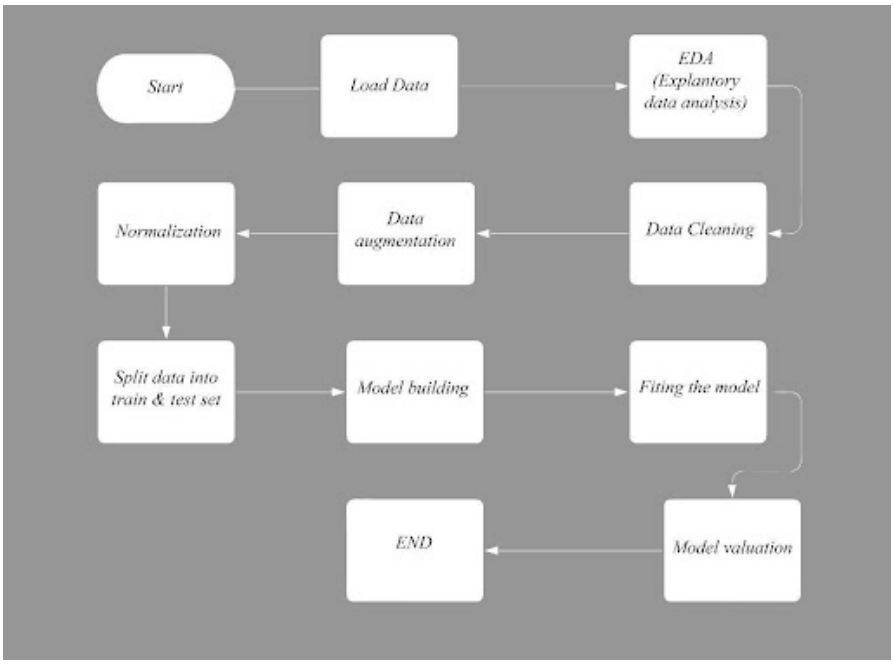


Figure 3.1: Methodology

3.1 Dataset analysis

The most well-known threat to humans, skin cancer, is primarily evaluated externally, beginning with a clinical screening at the base of the skin and maybe continuing with a dermoscopic examination, a biopsy, and histological evaluation. According to the fine-grained data, automating the organization of skin damage us-

ing photographs is a difficult task, changeability when there are open wounds.

This dataset is called **HAM10000** ("Human Against Machine with 10000 preparing images"). It consists of 10015 dermatoscopic images that are provided as a preparation set for academic AI applications and are made available to the public through the ISIC document. For AI and correlations with human experts, this benchmark dataset can be used. There are seven distinct categories of skin cancer, as follows:

1. **Melanocytic nevi:** Our study includes all of the many types of melanocytic nevi, which are non-cancerous growths of melanocytes. The dermatoscopic properties of these variants can vary greatly. They often exhibit symmetrical distribution of color and shape, unlike melanoma. [9].

2. **Melanoma:**Melanoma is a malignant tumor originating from melanocytes and can manifest in various forms. Melanoma can be successfully treated through simple surgical excision in Early Stage. Our study encompasses melanoma in situ including all variants of melanoma, while excluding subungual, non-pigmented, mucosal melanoma or ocular.

3. **Benign keratosis-like lesions:** "The classification of "Benign Keratosis" includes seborrheic keratoses, also it is known as "senile warts," as well as solar lentigo, considered as seborrheic keratosis's flat variant. Additionally, lichen-planus-like keratoses (LPLK) are part of this classification, representing seborrheic keratoses or solar lentigo with regression and inflammation [6]. We have grouped these subgroups together despite the fact that they may have different dermatoscopic characteristics because of their biological resemblance and regular reporting under the same histological name. Dermatoscopy is particularly difficult when dealing with keratoses that resemble lichen planus since they might visually resemble melanoma and are frequently subjected to biopsy or excision.

4. **Basal cell carcinoma:** A common epithelial skin cancer, seldom metastasizes but, if untreated, can have disastrous local effects. The varied morphological manifestations of it (flat, nodular, pigmented, and cystic) are further detailed in [11].

5. **Actinic keratoses:** Also known as solar keratoses and intraepithelial carcinoma (Bowen's disease) are prevalent forms of non-invasive squamous cell carcinomas. They can be treated locally without the need for surgery. While some scientists consider them to be precursors of squamous cell carcinoma rather than actual carcinomas, it is widely accepted that these lesions have the potential to progress into invasive squamous cell carcinoma. It is important to note that invasive squamous cell carcinoma is typically devoid of pigmentation.

6. **Vascular lesions:** Several different vascular skin lesions, including angiokeratomas, pyogenic granulomas, cherry angiomas [5]. as well as hemorrhages, are included in the dataset. Dermatoscopically, angiomas are recognized by their red or purple color and solid, well-defined features known as red clods or lacunae.

7. **Dermatofibroma:** Dermatofibroma, a benign skin lesion, is believed to be either a benign growth or an inflammatory reaction to minor trauma. The most typical dermatoscopic appearance consists of reticular lines around the periphery and a white patch in the center that denotes fibrosis [9].

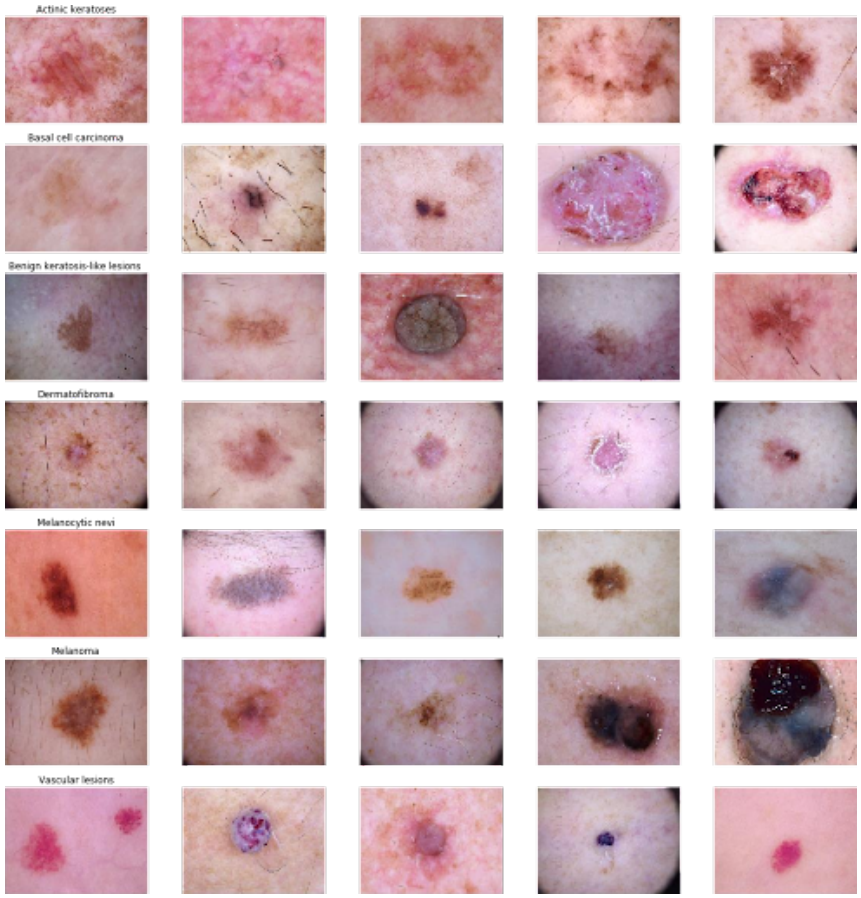


Figure 3.2: Samples from the Ham10000 Dataset. [27]

To analyze the distribution of the Technical Validation field (ground truth) categories, which are listed below, they were plotted:

1. **Histopathology (Histo):** Excised lesions were examined and diagnosed by specialized dermatopathologists using histopathology.

2. **Confocal:** A technique for in-vivo imaging with resolution that is near to the cellular level, reflectance confocal microscopy, was applied. A grey-world assumption in Lab-color space was used to process a few benign facial photos both before and after hand histogram adjustments.

3. **Follow-up:** Digital dermatoscopy was used to track nevi, and those that did not change after three follow-up visits, or for a period of 1.5 years, were deemed biologically benign. Only nevi were included in this ground-truth category; dermatologists often do not monitor benign conditions like vascular lesions, dermatofibromas or seborrheic keratoses.

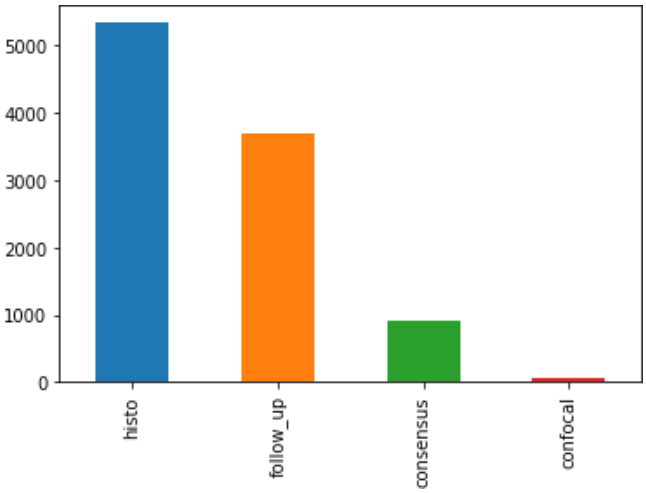


Figure 3.3: Technical Validation field of The Ham10000 data-set

4. **Consensus:** Authors PT and HK provided expert-consensus ratings for typical benign lesions in cases where histology or follow-up information was lacking. The consensus labels were only applied when both authors independently made the same clear benign diagnosis. Lesions with this form of ground truth were typically photographed for illustrative purposes only; no more follow-up or confirmation biopsies were required.

The dataset exhibits a class imbalance issue. Figure 3.3 provides a visual representation of the dataset's image distribution, allowing for easy interpretation.

3.2 Pre-processing

Data preprocessing is a crucial step in creating a machine learning model as it involves preparing raw data to make it suitable for analysis. In real-world machine learning projects, data is often not clean or properly formatted. To perform any operations on the data, it is necessary to clean and format it appropriately. This is where data preprocessing is necessary. Real-world data frequently has missing values, noise, and may be in an undesirable format, making it impossible for machine learning models to use it directly. To clean the data, data preparation is done, deal with missing values, and format it so that machine learning models can use it efficiently. By conducting data preprocessing, the efficiency and accuracy of the machine learning model can be improved. Preprocessing involves applying various transformations to the raw data before feeding it into a machine learning or deep learning algorithm. For instance, convolutional neural network training on raw images is likely to produce subpar classification results [18]. Our model was assessed using the HAM10000 data set. 10015 dermatoscopic photos were downsized from 450 x 600 to 224 x 224, where 3 stands for the RGB color channel. The dataset contains 7 classes, and upon examining the distribution of images across these classes, it becomes evident that there is an issue with class imbalance. Out of the 10,015 lesion IDs present, only 7,470 are unique [19]. The training set has 9077 photos, whereas the validation set has 938 images. To prevent having too many comparable images for our learning approach, a total of 4,501 images were eliminated from the

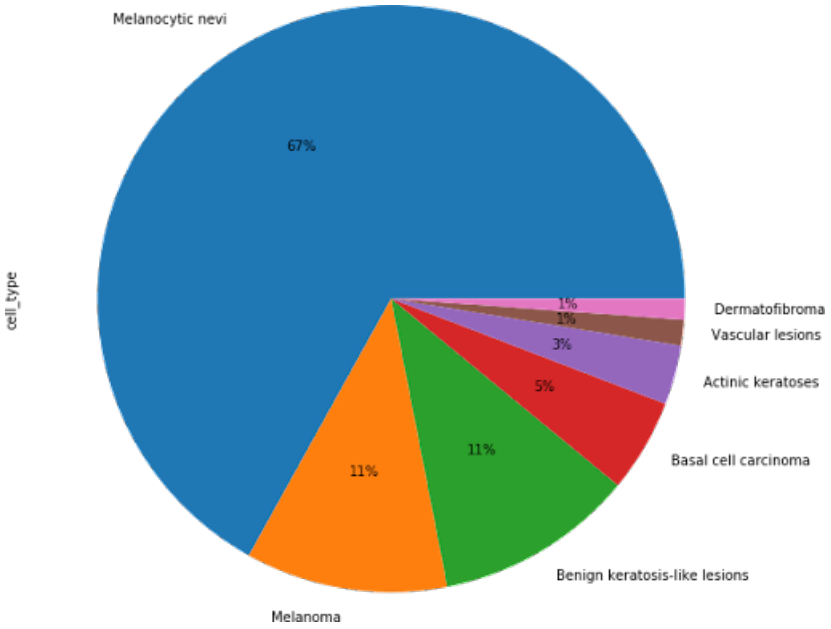


Figure 3.4: Image Distribution of the dataset.

dataset, as they shared the same lesion ID among the initial count of 10,015 images [17]. From 5,514 non-duplicate photographs, a subset of 938 photos was chosen for the validation set. Subsequently, the validation set was further reduced by removing 10,015 photos, resulting in the creation of the training set.

3.3 Data augmentation

Supervised Deep Learning model's prediction accuracy is strongly dependent on the quantity and diversity of data used for training. This relationship can be compared to the analogy of rocket engines requiring a significant amount of fuel to achieve the successful completion of a mission.

Deep learning models trained to do well on challenging tasks typically have a lot of hidden neurons. There are more trainable parameters as there are more hidden neurons. Modern Computer Vision models feature millions of parameters, such RESNET (60M) and Inception-V3 (24M).

In Natural Language Processing models like BERT (340M), the number of parameters can reach hundreds of millions. These deep learning models, developed to achieve high accuracy in tasks such as object detection and language translation, possess numerous tunable parameters. Consequently, to learn the values for these parameters during training, they require a significant amount of data.

In simple terms, the complexity of a task is directly related to the number of learnable parameters in the model, and consequently, the amount of data required is proportional to both factors.

In many cases, obtaining the large quantities of data necessary to train models for specific complex tasks, such as classifying weeds from crops or identifying novel

patient characteristics, is challenging. While transfer learning techniques can be helpful, there are difficulties in adapting pre-trained models to specific tasks.

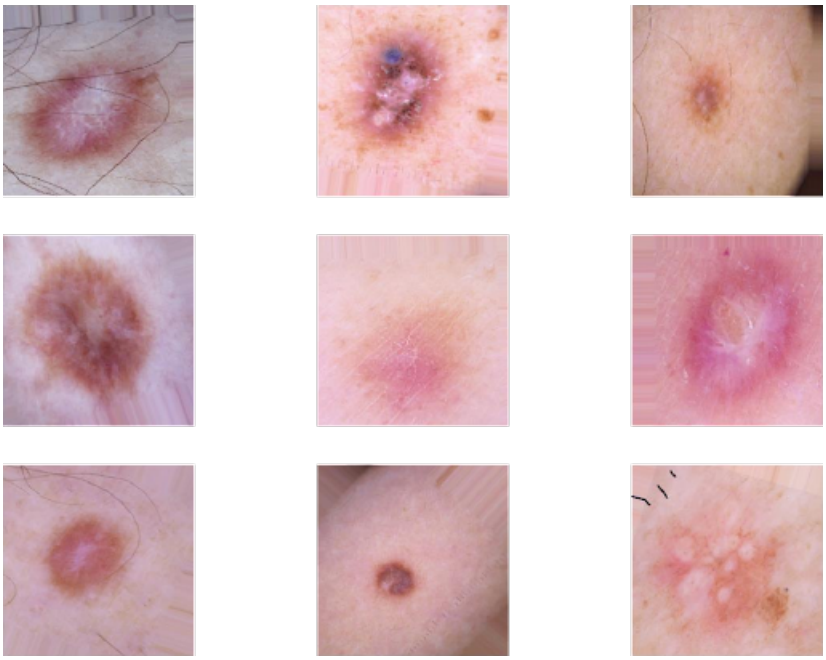


Figure 3.5: Examples of augmented images

Another approach to address the limited data problem is data augmentation, which involves applying various transformations to available data to generate synthetic data. Data augmentation serves both to increase the diversity of training data and to augment the data quantity. Additionally, augmented data can help alleviate class imbalance issues in classification tasks. [25].

Uneven class balance in the HAM10000 dataset is one of its primary issues. Melanocytic nevi (nv) has 6705 images, far more than any other class. We're going to employ a data augmentation strategy to solve this issue. Additionally, it will aid in lowering overfitting. There are numerous data augmentation techniques, including random nonlinear deformation and image rotation. In many circumstances, data augmentation shows to be a very effective way to increase accuracy [8]. For data augmentation, characteristics such as width and height shift range, zoom range, rotation range, and flip horizontally and vertically were used. On the training set, data augmentation was used, with the exception of the nv (melanocytic nevi) class because it is already well-represented. There were roughly 7000 photographs generated for each class, giving us a total of 43918 images. Further details are provided in Table 3.1.

Class	Before data augmentation		After data augmentation	
	Train set	validation set	Train set	validation set
Melanocytic nevi(nv)	5954	751	5954	751
Melanoma(mel)	1074	39	6894	39
Benign keratosis(bkl)	1024	75	6894	75
Basal cell carcinoma(bcc)	484	30	6826	30
Actinic keratosis(akiec)	301	26	6070	26
Vascular lesions(vasc)	131	11	6157	11
Dermatofibroma(df)	109	6	5123	6
Total	9077	939	43918	938

Table 3.1: Before and after applying data augmentation, distribution of images in the training and validation sets.

3.4 Model implementation

3.4.1 DenseNet-121

On our training set, we trained the DenseNet-121 pre-trained model. Here included Dropout, Global Average Pooling2D, and a dense layer using the SoftMax activation function in our sequential model. Dropout is a very simple method for preventing overfitting in neural networks [1]. The model's 7,044,679 total parameters. Among them, 83,648 parameters could not be trained whereas 6,961,031 parameters could. TensorFlow's Keras API was used to build the model. To determine the ideal weights, learning rate and dropout were set to 0.4 and 0.001, respectively. It is seen visually. Both batch size for training and validation were set to 10. The loss function used was categorical cross-entropy. When using a single label for many classes, it is typically used. Accuracy serves as a gauge of model performance as a whole. To assess the model, the Top2 accuracy as well as the average of Precision, Recall, and F1-score are examined.

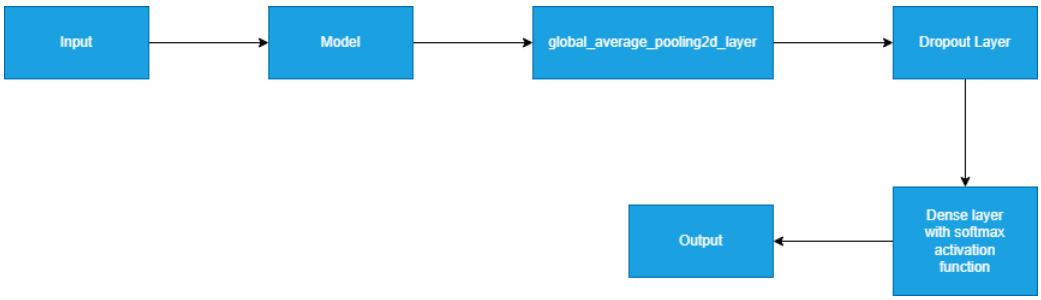


Figure 3.6: Model Implementation

Precision - Precision is defined as the ratio of correctly anticipated positive observations to all predicted positive observations. Low false positive rates are indicative of great precision. In our example, the precision we obtained was 0.788, which is regarded as good.

Precision = TP/TP+FP

Recall (Sensitivity) - Recall quantifies the proportion of accurately foreseen positive observations to all of the actual positive observations. It provides an answer to the query, "How many of the passengers who actually survived did we correctly label?" A recall of 0.631 was attained by our model, which is excellent as it is higher than 0.5.

Recall = TP/TP+FN

F1 score - A metric that combines recall and precision is the F1 score by taking their weighted average. It considers both false positives and false negatives when evaluating a model's performance. While it may be less intuitive than accuracy, the F1 score is often more valuable, particularly in cases of imbalanced class distribution. Accuracy is most effective when false positives and false negatives carry similar costs. However, when the costs of false positives and false negatives differ significantly, examining both precision and recall is recommended. In our case, the F1 score is calculated as 0.701, indicating a satisfactory performance of the model.

F1 Score = 2*(Recall * Precision) / (Recall + Precision)

3.4.2 VGG16

Here TensorFlow and Keras is used to implement the VGG16 model. It includes the following steps mentioned below:

Data Preprocessing: The code uses ImageDataGenerator to preprocess the images before feeding them into the network. The images are rescaled, rotated, zoomed, and flipped horizontally to augment the training data. The training and validation directories are set, and the batch size and image size are specified.

Loading the VGG16 Model: The VGG16 model is initialized with pre-trained weights from the ImageNet dataset. To adapt it for our purpose, the top layers of the model are excluded, retaining only the convolutional layers.

Adding New Top Layers: To enhance the loaded model, additional layers are incorporated above the existing convolutional layers. The output generated by the convolutional layers is flattened and subsequently fed into a fully connected layer comprising 256 neurons with a ReLU activation function. To mitigate overfitting, a dropout layer is included. To generate the output probabilities, a dense layer with a softmax activation function is then added.

Setting Layers to Non-Trainable: The original convolutional layers are frozen by setting their trainable attribute to False. This is done to prevent the weights of these layers from being updated during training, as they have already learned useful features from the ImageNet dataset.

Compiling the Model: The model is then compiled with the Adam optimizer, a learning rate of 0.001, and the categorical cross-entropy loss function. The model is also set to output the accuracy metric during training.

Training the Model: The model is trained on the training data for 3 epochs, with early stopping based on the validation loss. The training and validation accuracy and loss are recorded during training, and are used to plot the training and validation accuracy and loss graphs.

Evaluating the Model: The trained model is evaluated on the validation set, and the true labels and predicted labels are obtained. A confusion matrix is computed using these labels, and is plotted using the seaborn library. The classification report is also generated using the scikit-learn library.

3.4.3 Resnet50

To implement the model, firstly imports TensorFlow, Keras and all other necessary libraries. Then imports the ImageDataGenerator class from the Keras module for image data preprocessing, such as data augmentation and normalization. Imports the Resnet50 model, a convolutional neural network (CNN) architecture that has already been trained and is frequently employed for image classification tasks. Sets the size of the input images to be (224, 224) pixels. This is the input size expected by the ResNet50 model.

Then creates an instance of the ImageDataGenerator class for training data, which performs data augmentation and normalization. After that creates an instance of the ImageDataGenerator class for validation data, which performs normalization only by scaling the pixel values to a range of 0 to 1. Then creates a data generator using the flow from directory() method of the ImageDataGenerator class for training data. Here it generates batches of augmented and normalized images from the training directory, resizes the images to the specified target size of (224, 224) pixels, sets the batch size to 32, and specifies the class mode as 'categorical' for multi-class classification.

Applies the Global Average Pooling 2D layer to the tensor, which reduces the spatial dimensions of the tensor to a single value for each feature map, simply averaging all of the feature map's values. the model is then given a 0.5 rate Dropout layer, by randomly removing some of the input units during training which helps prevent overfitting, and a fully connected Dense layer with 1024 units and a ReLU activation function. The model is then given a second fully connected Dense layer with 512 units and a ReLU activation function.

Compiles the model, specifying Adam optimizer, categorical cross-entropy as the loss function for multi-class classification, and accuracy as the evaluation metric. Then Fits the model to the training data using the fit() method, specifying the data generator for training data, the number of epochs to train (3 in this case), the data generator for validation data, and the EarlyStopping callback for early stopping. Finally recompiles the model with a lower learning rate of 0.0001 using the Adam optimizer. The categorical crossentropy loss function is used for multi-class classification, and the accuracy metric is used for evaluation.

3.4.4 Inceptionv3

ImageDataGenerator class from the tensorflow.keras.preprocessing.image module. This class generates batches of augmented image data used for training deep learning models. Then imports the InceptionV3 class from the tensorflow.keras.applications.inception v3 module. This is a pre-trained model based on a convolutional neural network (CNN) designed specifically for image classification tasks. Dense and GlobalAverage-Pooling2D classes from the tensorflow.keras.layers module. These are layers that can be used to build deep learning models. ImageDataGenerator object for the training data and sets the rescale parameter to 1./255, this operation normalizes the pixel values to a range of 0 to 1.

DirectoryIterator object using the flow from directory() method of the valid datagen object. This iterator generates batches of validation data from the valid dir directory. InceptionV3 model specifies that the top layers should not be included and with pre-trained weights from the imagenet dataset. The input shape parameter is set to the desired input shape of the images. New layers to the Inceptionv3 model that we loaded earlier. I first create a new variable x which is set to the output of the base model. Then add a GlobalAveragePooling2D layer to x, which reduces the dimensions of the output from the base model to a fixed size and add a Dense layer with 1024 neurons and 'relu' activation to x.

Finally, add a Dense layer with the number of classes in our dataset as the number of output neurons and 'softmax' activation to x. This will be the final output layer of our model. Defines new model as a Model object with inputs from the base model and outputs from our new output layer. The chosen optimizer for this model is stochastic gradient descent (SGD) with a learning rate of 0.0001 and a momentum of 0.9. The selected loss function for this model is categorical cross-entropy, and the evaluation metric is accuracy.

Trained the model for 03 epochs using the train generator as the input data, valid generator as the validation data, and earlystop as the callback to monitor the validation loss. The fit method executes and returns a history object which stores the training and validation metrics information. Then compiles the model for fine-tuning. Fine-tunes the model by fitting it to the training data and validating it on the validation data. The train generator and valid generator are used to generate batches of augmented images for training and validation, respectively. The model is trained for 5 epochs. The trained model is evaluated on the validation set, and the true labels and predicted labels are obtained. Two confusion matrices are computed using these labels, and are plotted using the seaborn library before and after fine tuning. The classification report is also generated using the scikit-learn library.

Chapter 4

Result Analysis

This chapter includes all the data if four model and their analysis which were gathered during the experiment. Comparative analysis with some of the existing work will be included as well.

4.1 Model evaluation

4.1.1 DenseNet-121

The training data was used to train the DenseNet-121 model for three comparison epochs and thirty epochs to obtain the best results. The model's training-validation accuracy and training-validation loss curves were calculated for analysis. In Figure 4.1, the loss curve shows a downward trend, indicating improved performance. The small gap between the validation and train curves suggests a good alignment [10].

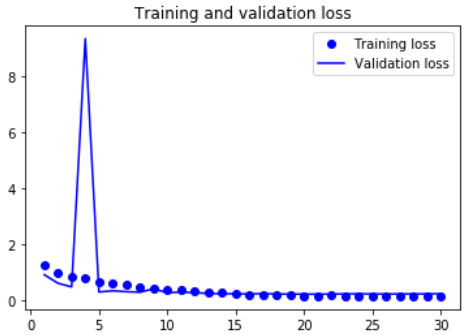


Figure 4.1: Train and validation loss

The model achieved a top2 accuracy of 97% and a categorical accuracy of 92%. Figure 4.2 shows the curve of train-validation top2 accuracy and the train-validation accuracy. The CNN's high connectivity proves advantageous, as it allows for the reuse of features

The performance measurement model's confusion matrix is shown in Fig. 4.3. We may determine the following facts from the matrix:

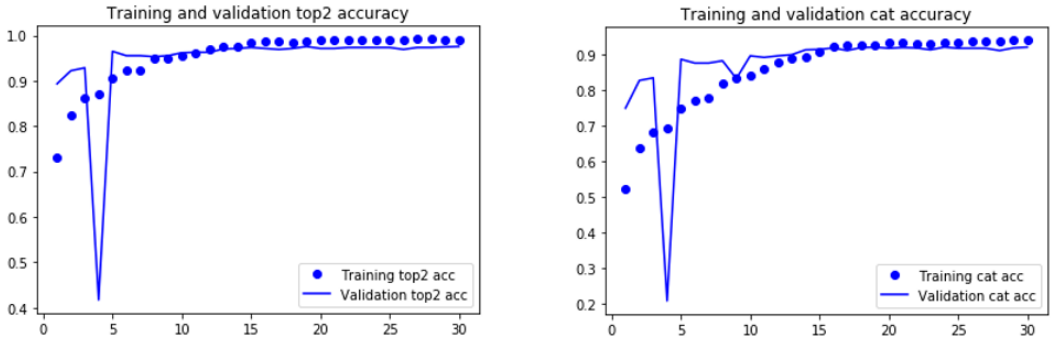


Figure 4.2: Categorical and Top 2 accuracy

- **Actinic keratosis(akiec)**- Out of the 26 images, 15 were correctly predicted.
- **Basal cell carcinoma(bcc)**- Out of the 30 images, 23 correctly predicted.
- **Benign keratosis(bkl)**- Out of the 75 images, 34 correctly predicted.
- **Dermatofibroma(df)**- Out of the 6 images, 3 correctly predicted.
- **Melanoma(mel)**- Out of the 39 images, 22 correctly predicted.
- **Melanocytic nevi(nv)**- Out of the 751 images, 675 correctly predicted.
- **Vascular lesions(vasc)**- Out of the 11 images, 10 correctly predicted.

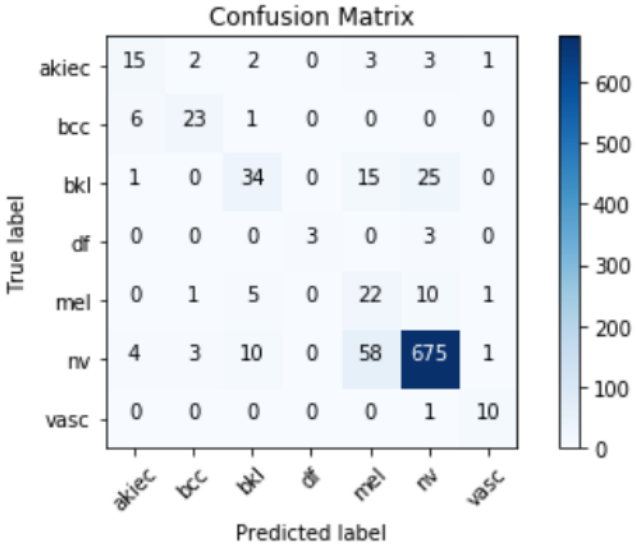


Figure 4.3: Confusion matrix of Densenet121

Table 4.1 displays the classification report for the densenet-121 model on the ham10000 dataset. Table 4.1 details each class's Precision, Recall, and F1-score results. This model's best predictions for Melanocytic Nevi (nv) and Melanoma (mel) are both fairly challenging to recognize. This model achieved an average Precision, Recall, and F1-score of 87%.

Class	Precision	Recall	F1-score	Support
Melanocytic nevi(nv)	0.94	0.90	0.92	751
Melanoma(mel)	0.22	0.56	0.32	39
Benign keratosis(bkl)	0.65	0.45	0.54	75
Basal cell carcinoma(bcc)	0.79	0.77	0.78	30
Actinic keratosis(akiec)	0.58	0.58	0.58	26
Vascular lesions(vasc)	0.77	91	0.83	11
Dermatofibroma(df)	1.0	0.50	0.67	6
average	0.87	0.83	0.85	938

Table 4.1: Classification report of DenseNet-121

4.1.2 VGG16

The VGG16 model is trained on the training data for 03 epochs, with early stopping based on the validation loss. The training and validation loss and accuracy are recorded during training, and are used to plot the training and validation loss and accuracy graphs shown in Fig. 4.4. This model achieves 80% accuracy.

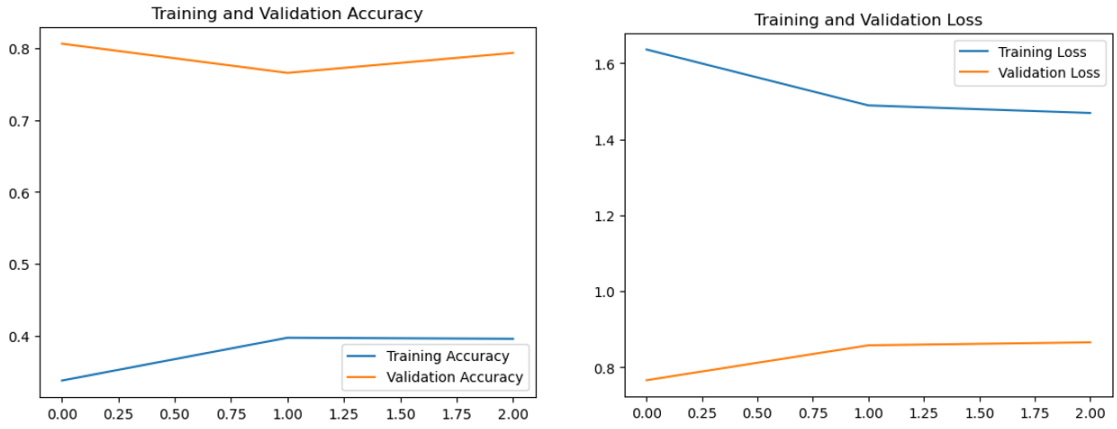


Figure 4.4: Training and Validation Accuracy and Loss

The performance measurement model's confusion matrix is shown in Fig. 4.5. We may determine the following facts from the matrix:

- **Actinic keratosis(akiec)**- Out of 26 images, 01 correctly predicted.
- **Basal cell carcinoma(bcc)**- Out of 30 images,02 correctly predicted.
- **Benign keratosis(bkl)**- Out of 75 images, 04 correctly predicted.
- **Dermatofibroma(df)**- Out of 6 images, 0 correctly predicted.
- **Melanoma(mel)**- Out of 39 images, 02 correctly predicted.
- **Melanocytic nevi(nv)**- Out of 751 images, 602 correctly predicted.
- **Vascular lesions(vasc)**- Out of 11 images, 00 correctly predicted.

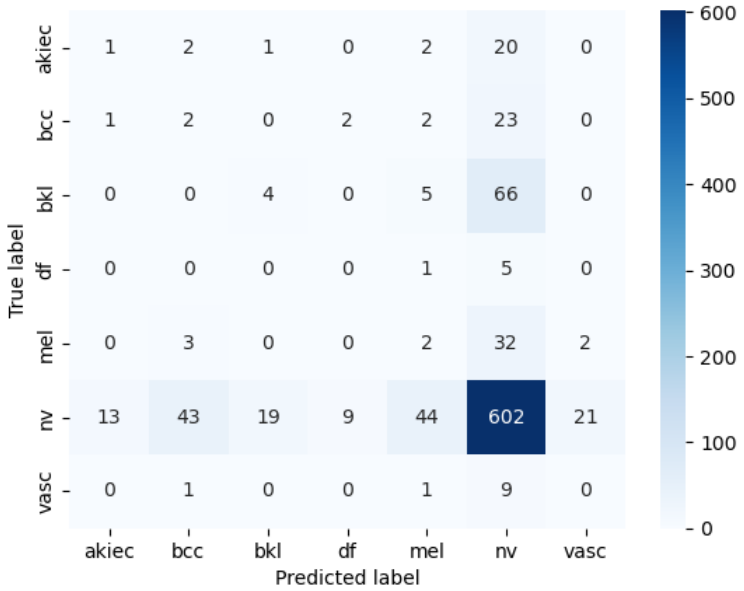


Figure 4.5: Confusion matrix of VGG16

Table 4.2 displays the classification report for the VGG16 model on the ham10000 dataset. Table 4.2 details each class's Precision, Recall, and F1-score results. The model struggles with identifying the other classes and demonstrates the highest accuracy in predicting Melanocytic nevi (nv). This model achieved an average Precision, Recall, and F1-score of 65%.

Class	Precision	Recall	F1-score	Support
Melanocytic nevi(nv)	0.80	0.80	0.80	751
Melanoma(mel)	0.04	0.05	0.04	39
Benign keratosis(bkl)	0.17	0.05	0.08	75
Basal cell carcinoma(bcc)	0.04	0.07	0.05	30
Actinic keratosis(akiec)	0.07	0.04	0.05	26
Vascular lesions(vasc)	0.00	0.00	0.00	11
Dermatofibroma(df)	0.00	0.00	0.00	6
average	0.65	0.65	0.65	938

Table 4.2: Classification report of VGG16

4.1.3 ResNet50

The ResNet50 model is trained on the training data for 03 epochs, with fine tuning the model. The training and validation loss and accuracy are recorded during training, and are used to plot the training and validation loss and accuracy graphs shown in Fig. 4.6. This model achieves 78% accuracy.

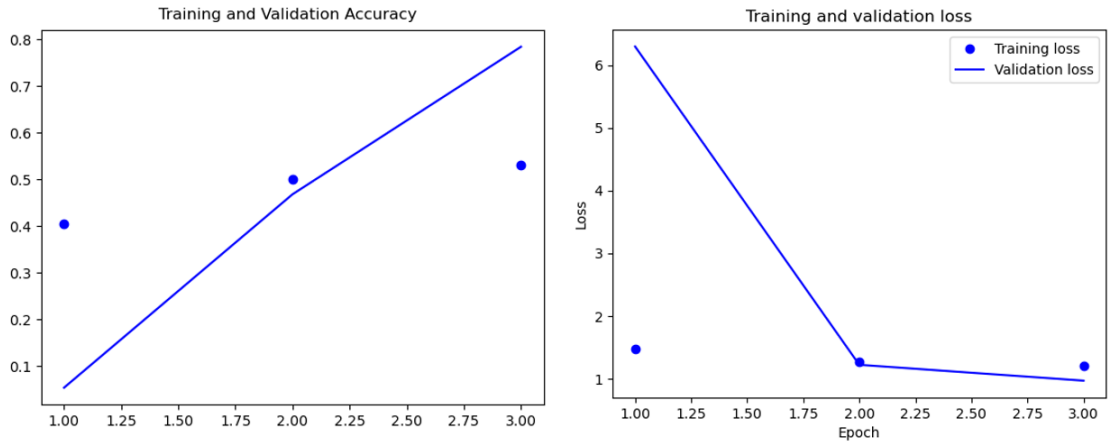


Figure 4.6: Training and Validation Accuracy and Loss

The performance measurement model's confusion matrix is shown in Fig. 4.7. We may determine the following facts from the matrix:

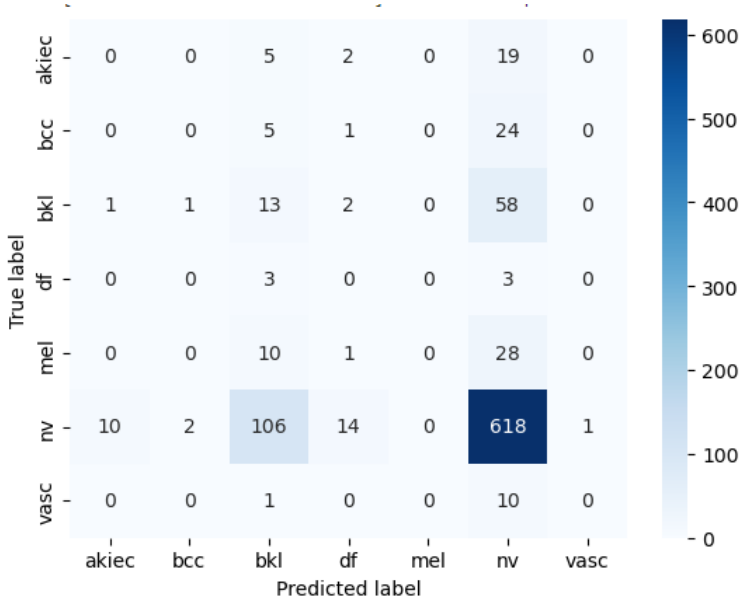


Figure 4.7: Confusion matrix of ResNet50

- **Actinic keratosis(akiec)**- Out of 26 images, 00 correctly predicted.
- **Basal cell carcinoma(bcc)**- Out of 30 images, 00 correctly predicted.
- **Benign keratosis(bkl)**- Out of 75 images, 13 correctly predicted.
- **Dermatofibroma(df)**- Out of 6 images, 00 correctly predicted.

- **Melanoma(mel)**- Out of 39 images, 00 correctly predicted.
- **Melanocytic nevi(nv)**- Out of 751 images, 618 correctly predicted.
- **Vascular lesions(vasc)**- Out of 11 images, 00 correctly predicted.

Table 4.3 displays the classification report for the ResNet50 model on the ham10000 dataset. Table 4.3 details each class's Precision, Recall, and F1-score results. The model struggles with identifying the other classes and demonstrates the highest accuracy in predicting Melanocytic nevi (nv). This model achieved an average Precision, Recall, and F1-score of 66%.

Class	Precision	Recall	F1-score	Support
Melanocytic nevi(nv)	0.81	0.82	0.82	751
Melanoma(mel)	0.00	0.00	0.00	39
Benign keratosis(bkl)	0.09	0.17	0.12	75
Basal cell carcinoma(bcc)	0.00	0.00	0.00	30
Actinic keratosis(akiec)	0.00	0.00	0.00	26
Vascular lesions(vasc)	0.00	0.00	0.00	11
Dermatofibroma(df)	0.00	0.00	0.00	6
average	0.66	0.67	0.66	938

Table 4.3: Classification report of ResNet50

4.1.4 Inceptionv3

The Inceptionv3 model is trained on the training data for 03 epochs, with fine tuning the model. The training and validation loss and accuracy are recorded during training, and are used to plot the training and validation loss and accuracy graphs shown in Fig. 4.8. This model achieves 84% accuracy.

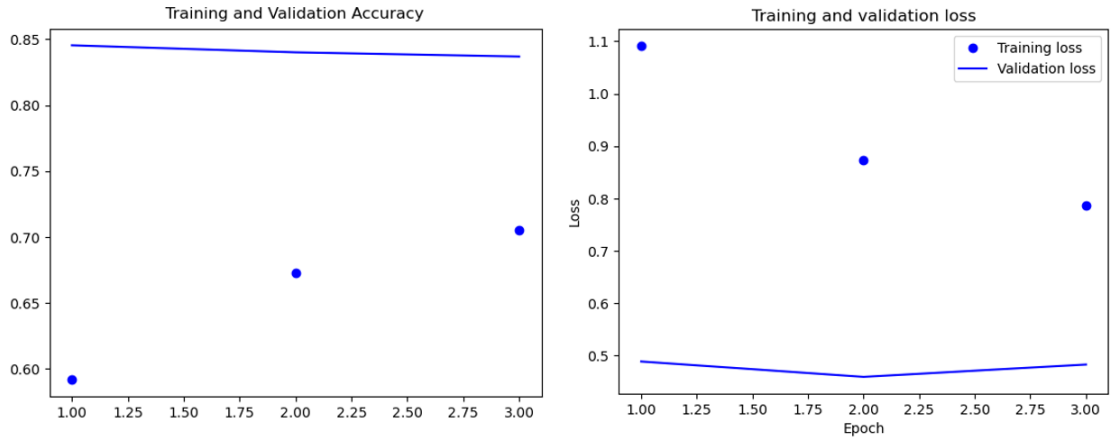


Figure 4.8: Training and Validation Accuracy and Loss

The performance measurement model's confusion matrix is shown in Fig. 4.9. We may determine the following facts from the matrix:

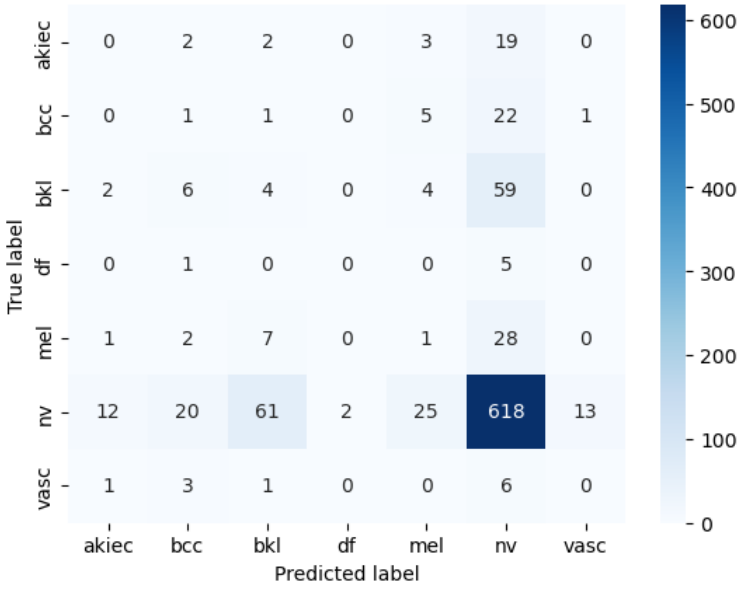


Figure 4.9: Confusion matrix of Inceptionv3

- **Actinic keratosis(akiec)**- Out of 26 images, 00 correctly predicted.
- **Basal cell carcinoma(bcc)**- Out of 30 images, 01 correctly predicted.
- **Benign keratosis(bkl)**- Out of 75 images, 04 correctly predicted.
- **Dermatofibroma(df)**- Out of 6 images., 00 correctly predicted
- **Melanoma(mel)**- Out of 39 images, 01 correctly predicted.
- **Melanocytic nevi(nv)**- Out of 751 image, 618 correctly predicteds.
- **Vascular lesions(vasc)**- Out of 11 images, 00 correctly predicted.

Table 4.4 displays the classification report for the Inceptionv3 model on the ham10000 dataset. Table 4.4 details each class's Precision, Recall, and F1-score results. The model struggles with identifying the other classes and demonstrates the highest accuracy in predicting Melanocytic nevi (nv). This model achieved an average Precision, Recall, and F1-score of 66

Class	Precision	Recall	F1-score	Support
Melanocytic nevi(nv)	0.82	0.82	0.82	751
Melanoma(mel)	0.03	0.03	0.03	39
Benign keratosis(bkl)	0.05	0.05	0.05	75
Basal cell carcinoma(bcc)	0.03	0.03	0.03	30
Actinic keratosis(akiec)	0.00	0.00	0.00	26
Vascular lesions(vasc)	0.00	0.00	0.00	11
Dermatofibroma(df)	0.00	0.00	0.00	6
average	0.66	0.67	0.66	938

Table 4.4: Classification report of Inceptionv3

4.2 Comparison

In this project I have shown comparison of four CNN models, DenseNet-121, VGG16, ResNet50 and Inceptionv3. From the result analysis we can see that, from all these four models DenseNet-121 has achieved best accuracy and also better classification. VGG16, ResNet50 and Inceptionv3 has achieved 80%, 78% and 84% accuracy, where DenseNet-121 model has achieved 97% top2 accuracy and 92% categorical accuracy.

Also, VGG16, ResNet50 and Inceptionv3 has been able to classify one class Melanocytic nevi(nv) perfectly from the seven class of the dataset used here and other classes have very low classification scores. On the other hand, DenseNet-121 has better classification scores on every seven class, but comparatively low score on one class Melanoma(mel). At table 4.5 comparison of four different modes is shown below:

Classifier Name	Number of Classes	Accuracy%
DenseNet-121	Seven	97%
VGG16	Seven	80%
ResNet50	Seven	78%
Inceptionv3	Seven	84%

Table 4.5: Comparison of Four CNN Models

Comparison of some previous related work:
Table 4.6 presents a comparison between several contemporary works and my current study. W. Sae-Lim et al. got 84% categorical accuracy on their validation set and S. S. Chaturvedi et al. got 83.15% categorical accuracy on their validation set. But from my work, DenseNet-121 model got 92% categorical accuracy and 97% top2 accuracy.

Classifier Name	Year	Number of Classes	Accuracy%
AlexNet and VGGNet [25]	2018	Three	79.9%
GoogleNet and VGGNet [25]	2018	Three	81.2%
VGGNet [25]	2018	Three	79.3%
GoogleNet and AlexNet [25]	2018	Three	80.7%
VGGNET [31]	2019	Seven	85.62%
MobileNet [32]	2019	Seven	83.23%
DenseNet-121	Current Work	Seven	97%
VGG16	Current Work	Seven	80%
ResNet50	Current Work	Seven	78%
Inceptionv3	Current Work	Seven	84%

Table 4.6: Comparison between some previous work and the current study

The experimental data was presented which was acquired during work using the Ham-10000 dataset using DenseNet-121, VGG16, ResNet50 and Inceptionv3. Among them DenseNet-121 model got an average of 87% Precision,83% Recall and 85% F1-score and the loss curve was in the satisfactory range.

Chapter 5

Conclusion and Future Works

This chapter briefly describes the outcome, limitations of the project work. Future work segment is also added.

5.1 Summary

In recent times, the incidence of skin cancer has risen significantly. Early detection plays a crucial role in ensuring successful treatment and recovery for patients. Deep Convolutional Neural Networks (CNNs) have gained immense popularity in the domain of visual recognition tasks. In the medical field, CNNs are steadily establishing their prominence. Leveraging CNN models, our aim was to develop a more reliable model that assists dermatologists in accurately identifying skin lesions. Among the models explored, the DenseNet-121 model achieved an impressive categorical accuracy of 92% and a top2 accuracy of 97%, thanks to the implementation of data augmentation techniques. It is worth noting that the availability of limited datasets and the imbalanced distribution of classes pose challenges for researchers in this field. Therefore, further research in this area is necessary to advance our understanding and capabilities.

5.2 Limitations

As there is less data set to test the model, we need to merge various datasets and perform more robust solutions for the dermatologists. Model needs to be more tuned to remove the overfitting.

5.3 Future Work

Various pre-processing techniques can be applied to see how it works for the dataset. TTA (Test time augmentation) can also be used to see if the result improves.

5.4 Conclusion

This project focused on skin lesion classification and presented an approach to improve the accuracy, precision, recall, and F1-score using CNN models on the HAM10000 dataset. The results demonstrated the effectiveness of this approach. Furthermore, it is anticipated that by integrating the model implementation approach with appropriate image pre-processing methods, the classification approach can be enhanced for other medical images as well. This highlights the potential for broader applications of the proposed methodology in the field of medical image analysis.

Bibliography

[1] T. M. Mitchell, *Machine Learning*. McGraw-Hill Science/Engineering/Math, Mar. 1997.

[2] M. Vestergaard, P. Macaskill, P. Holt, and S. W. Menzies, "Dermoscopy compared with naked eye examination for the diagnosis of primary melanoma: a meta-analysis of studies performed in a clinical setting," *British Journal of Dermatology*, ??? Sep. 2008. DOI: 10.1111/j.1365-2133.2008.08713.x.

[3] E. Alpaydin, *Introduction to Machine Learning*. MIT Press (MA), Jan. 2010.

[4] S. J. Russell and P. Norvig, *Artificial Intelligence*. Jan. 2010.

[5] P. Zaballos, M. Carulla, F. Ozdemir, *et al.*, "Dermoscopy of pyogenic granuloma: a morphological study," *British Journal of Dermatology*, vol. 163, no. 6, pp. 1229–1237, Dec. 2010. DOI: 10.1111/j.1365-2133.2010.10040.x.

[6] P. Zaballos, E. Salsench, P. Serrano, F. Cuellar, S. Puig, and J. Malvehy, "Studying Regression of Seborrheic Keratosis in Lichenoid Keratosis with Sequential Dermoscopy Imaging," *Dermatology*, vol. 220, no. 2, pp. 103–109, Jan. 2010. DOI: 10.1159/000265556.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Neural Information Processing Systems*, vol. 25, pp. 1097–1105, Dec. 2012. [Online]. Available: http://books.nips.cc/papers/files/nips25/NIPS2012_0534.pdf.

[8] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning*. The MIT Press, Aug. 2012.

[9] C. Rosendahl, A. Cameron, I. McColl, and D. Wilkinson, "Dermatoscopy in routine practice - 'chaos and clues'.," *Australian Family Physician*, vol. 41, no. 7, pp. 482–7, Jul. 2012.

[10] H. K. Jabbar and R. Z. Khan, "Methods to Avoid Over-Fitting and Under-Fitting in Supervised Machine Learning (Comparative Study)," *Computer Science, Communication and Instrumentation Devices*, Jan. 2014. DOI: 10.3850/978-981-09-5247-1\017. [Online]. Available: https://doi.org/10.3850/978-981-09-5247-1_017.

[11] A. Lallas, Z. Apalla, G. Argenziano, *et al.*, "The dermatoscopic universe of basal cell carcinoma," *Dermatology practical conceptual*, vol. 4, no. 3, Jul. 2014. DOI: 10.5826/dpc.0403a02.

[12] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, Jan. 2014. [Online]. Available: https://jmlr.csail.mit.edu/papers/volume15/srivastava14a/srivastava14a.pdf.

[13] Y. LeCun, Y. Bengio, and G. E. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015. DOI: 10.1038/nature14539.

[14] V. Mnih, K. Kavukcuoglu, D. Silver, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015. DOI: 10.1038/nature14236.

[15] O. Russakovsky, J. Deng, H. Su, *et al.*, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, Dec. 2015. DOI: 10.1007/s11263-015-0816-y.

[16] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, Jan. 2015. DOI: 10.1016/j.neunet.2014.09.003.

[17] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, Nov. 2016.

[18] K. Pal and K. S. Sudeep, "Preprocessing for image classification by convolutional neural networks," *IEEE International Conference on Recent Trends in Electronics, Information Communication Technology*, May 2016. DOI: 10.1109/rteict.2016.7808140.

[19] D. Silver, A. Huang, C. J. Maddison, *et al.*, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, Jan. 2016. DOI: 10.1038/nature16961.

[20] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," *2017 International Conference on Engineering and Technology (ICET)*, Aug. 2017. DOI: 10.1109/icengtechnol.2017.8308186. [Online]. Available: https://doi.org/10.1109/icengtechnol.2017.8308186.

[21] G. M. Farinella, C. Wong, F. Deligianni, *et al.*, "Deep Learning for Health Informatics," *IEEE Journal of Biomedical and Health Informatics*, vol. 21, no. 1, pp. 4–21, Jan. 2017. DOI: 10.1109/jbhi.2016.2636665. [Online]. Available: https://doi.org/10.1109/jbhi.2016.2636665.

[22] J. J. Fei-Fei Li and S. Yeung, *Class lecture, topic: "convolutional neural networks for visual recognition."* Spring 2017.

[23] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," *arXiv (Cornell University)*, Jul. 2017. DOI: 10.1109/cvpr.2017.243. [Online]. Available: http://arxiv.org/pdf/1608.06993.

[24] E. Jana, R. Subban, and S. Saraswathi, "Research on Skin Cancer Cell Detection Using Image Processing," *International Conference on Computational Intelligence and Computing Research*, Dec. 2017. DOI: 10.1109/iccic.2017.8524554.

[25] A. Mikołajczyk and M. Grochowski, "Data augmentation for improving deep learning in image classification problem," *2018 International Interdisciplinary PhD Workshop (IIPhDW)*, May 2018. DOI: 10.1109/iiphdw.2018.8388338. [Online]. Available: https://doi.org/10.1109/iiphdw.2018.8388338.

[26] P. Ruiz, *Understanding and visualizing DenseNets - Towards Data Science*, Oct. 2018. [Online]. Available: https://towardsdatascience.com/understanding-and-visualizing-densenets-7f688092391a.

[27] P. Tschandl, C. Rosendahl, and H. Kittler, "The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions," *Scientific Data*, vol. 5, no. 1, Aug. 2018. DOI: 10.1038/sdata.2018.161. [Online]. Available: https://www.nature.com/articles/sdata2018161.pdf.

[28] Admin, *DenseNet paper review - Chadrick039;s Blog*, May 2019. [Online]. Available: https://chadrick-kwag.net/densenet-paper-review/.

[29] J. Brownlee, *A Gentle Introduction to Pooling Layers for Convolutional Neural Networks*, Jul. 2019. [Online]. Available: https://machinelearningmastery.com/pooling-layers-for-convolutional-neural-networks.

[30] *CNN ARCHITECTURES: DENSENET*, Apr. 2020. [Online]. Available: https://machinelearningtokyo.com/2020/04/21/cnn-architectures-densenet/.

[31] M. U. Hassan, "VGG16 8211; Convolutional Network for Classification and Detection," *Neurohive*, Feb. 2021. [Online]. Available: https://neurohive.io/en/popular-networks/vgg16/.

[32] S.-H. Tsang, *Review: DenseNet — Dense Convolutional Network (Image Classification)*, Dec. 2021. [Online]. Available: https://towardsdatascience.com/review-densenet-image-classification-b6631a8ef803.

[33] T. S. C. Foundation, *Skin Cancer Facts Statistics - The Skin Cancer Foundation*, Mar. 2023. [Online]. Available: https://www.skincancer.org/skin-cancer-information/skin-cancer-facts/.

[34] GeeksforGeeks, "VGG 16 CNN model," *GeeksforGeeks*, Jan. 2023. [Online]. Available: https://www.geeksforgeeks.org/vgg-16-cnn-model/.

[35] *ResNet-50: The Basics and a Quick Tutorial*, Mar. 2023. [Online]. Available: https://datagen.tech/guides/computer-vision/resnet-50/.

[36] *7 Types of Neural Network Activation Functions: How to Choose?* [Online]. Available: https://missinglink.ai/guides/neural-network-concepts/7-types-neural-network-activation-functions-right/.

[37] *Chapter 4. Fully Connected Deep Networks.* [Online]. Available: https://www.oreilly.com/library/view/tensorflow-for-deep/9781491980446/ch04.html.

[38] *Convolutional Neural Networks for Visual Recognition.* [Online]. Available: https://cs231n.github.io/convolutional-networks/.

[39] *Online.* [Online]. Available: https://computersciencewiki.org/index.php/File:MaxpoolSample2.png.

[40] *Outline of the convolutional layer.* [Online]. Available: https://www.researchgate.net/figure/Outline-of-the-convolutionallayer_fig1_323792694.