Skin Lesions Classification Using Deep Learning

Submitted By

Kaium Uddin

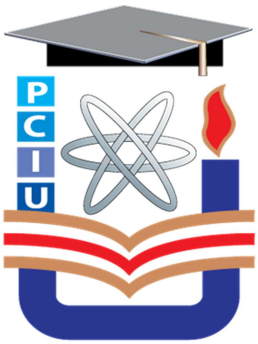
ID: CSE 01806657

&

Shifat Hasan

ID: CSE 01806647

A THESIS SUBMITTED IN CONFORMITY WITH THE REQUIREMENT
FOR THE DEGREE OF
BACHELOR OF SCIENCE IN COMPUTER SCIENCE & ENGINEERING



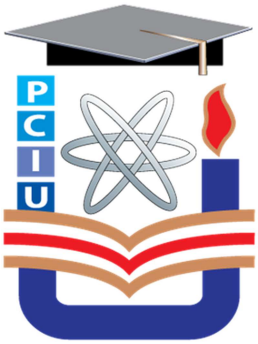
Port City International University

Department of Computer Science and Engineering

07, Nikunja Housing Society, South Khushi, Chattogram- 4202

Chattogram, Bangladesh.

January 2023



Department of Computer Science and Engineering
Port City International University

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE
OF
BACHELOR OF SCIENCE IN COMPUTER SCIENCE & ENGINEERING

By
Kaium Uddin
ID: CSE 01806657
&
Shifat Hasan
ID: CSE 01806647

Skin Lesions Classification Using Deep Learning

Under the Supervision of
Farzina Akther
Senior Lecturer
Department of Computer Science and Engineering
Port City International University
January 2023

Declaration of Originality

To the best of the author's knowledge, this thesis has not been submitted anywhere for the award of any degree. References and acknowledgements to other researchers have been given as appropriate. We also certify that we just used the materials that were mentioned. All formulations and notions taken directly or in substance from printed or unprinted literature, or the Internet, have been properly cited with footnotes or other exact references to the source, as per the standard scientific procedure. We're careful that radiating base information could achieve authentic ramifications.

_____

[Signature of the Candidate]

Kaium Uddin

CSE 01806657

Department of CSE,

Port City International University

_____

[Signature of the Candidate]

Shifat Hasan

CSE 01806647

Department of CSE,

Port City International University

Approval For Submission

This thesis titled **"Skin Lesions Classification Using Deep learning"** by Kaium Uddin and Shifat Hasan has been approved for submission to the Department of Computer Science and Engineering, Port City International University, in partial fulfilment of the requirement for the degree of Bachelor of Science Engineering.

_____

[Signature of the Supervisor]

Farzina Akther

Senior Lecturer

Department of Computer Science & Engineering

Port City International University

07, Nikunja Housing Society, South Khushi, Chattogram- 4202

Chattogram, Bangladesh.

Dedication

Dedicated to our cherished parents and our respectable teachers.

Acknowledgement

Above all else, we offer our sincere thanks to Almighty for presenting to me the great gift of effectively finishing my last year's thesis. We had the astonishing honour of concentrating under the direction and support of our supervisor, Farzina Akther her immense information offers us the chance to expand our viewpoints and gain significant headway. Our appreciative thanks are likewise stretched out to we might want to thank the department of Computer Science and Engineering for their assistance in offering us the assets to run the program. Finally, we wish to thank our family for their help and support throughout our assessments.

Abstract

A branch of computer science known as artificial intelligence (AI) focuses on building software and robots with intelligence that behave and act like people. Artificially intelligent computers are built for a variety of tasks, including speech recognition, learning, planning, and problem-solving. A group of algorithms called "deep learning" are employed in machine learning. It is a member of a large family of machine learning techniques centered on discovering data representations. We create a model to categorize different kinds of skin lesions using deep learning. In this thesis, we examine the methods and procedures for using deep learning to create a model for skin lesion classification. Finding a reliable and practical method to identify the type of skin lesion based on.

Keywords: Augmentation, Oversampling, CNN, Computer Vision, Skin Lesions.

Table of Contents

List of Figure

List of Tables

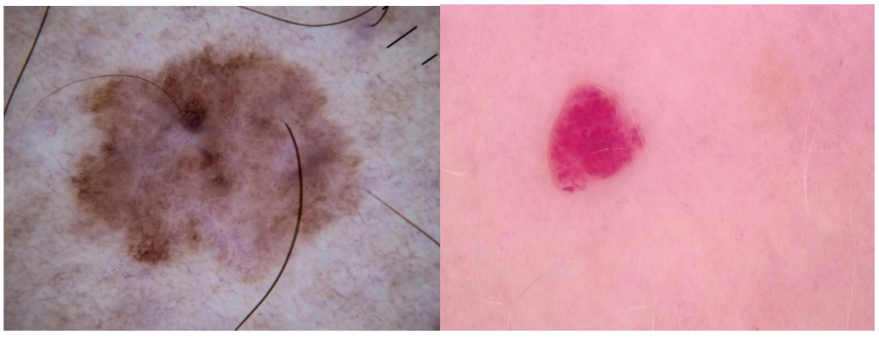
Chapter 1

Introduction

Skin lesion classification is a critical task in dermatology, as the early detection and diagnosis of skin cancer and other abnormal skin conditions can greatly improve patient outcomes. However, the classification of skin lesions is a challenging task due to the variability in lesion appearance, limited and imbalanced datasets, annotator variability, and limited generalizability. The manual examination of skin lesions by dermatologists is time-consuming and subject to inter-observer variability, which highlights the need for automated diagnostic tools. Deep learning, a type of machine learning that utilizes neural networks with multiple layers, has shown great promise in various image classification tasks, including skin lesion classification. Deep learning models can learn to extract more nuanced features from images, which can result in a higher diagnostic accuracy compared to traditional machine learning models. In this thesis, we propose a novel approach for skin lesion classification using deep learning. Our approach utilizes a combination of clinical images, bio markers and patient history to improve the accuracy and robustness of the models. We developed and evaluated several deep learning models, including convolutional neural networks and ensemble methods. The performance of the proposed approach is evaluated using several benchmark datasets and compared with state-of-the-art methods. The main objective of this thesis is to develop an automated diagnostic tool that can assist dermatologists in the early detection and diagnosis of skin cancer and other abnormal skin conditions. We hope that our approach can help to reduce the workload of dermatologists and improve patient care.

1.1 What is Skin Lesions

Any region of skin that differs in color, shape, size, or texture from the surrounding skin is referred to as a skin lesion. Skin lesions are rather common and typically result from small-scale skin injuries like sunburns or contact dermatitis. Others, such as infections, diabetes, autoimmune disorders, and genetic ailments, can be signs of undiagnosed conditions. The majority of skin lesions are benign and painless, but a small percentage are malignant or premalignant, which means they may develop into skin cancer.[1]



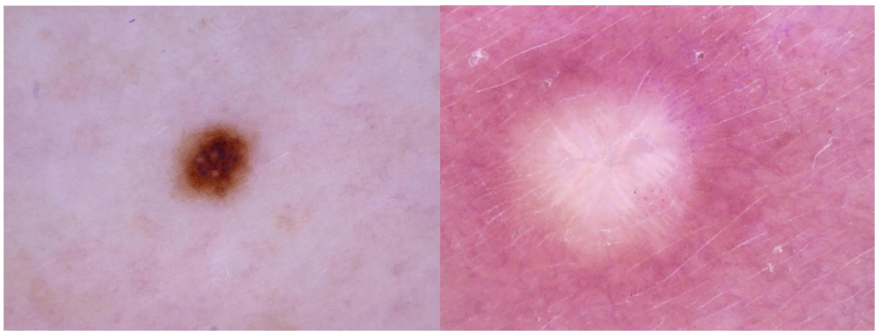


Figure 1.1 Some skin lesion images

1.1.1 Causes

Skin lesions are frequent and can be caused by sunburn or other forms of skin damage. Infections or autoimmune disorders may be present as a symptom of underlying problems in some cases. The majority of skin lesions are benign (noncancerous) and unimportant, but they may also signal a more serious condition. [2]

1.1.2 Symptoms

Skin cancer typically appears in sun-exposed regions of the body, such as the scalp, face, lips, ears, neck, chest, arms, and hands in women, as well as the legs. But it can also develop on parts of your body that are infrequently exposed to sunlight, such as your palms, the skin just below your finger or toenail, and your genital region. any skin tone, even those with darker skin tones, are susceptible to developing skin cancer. Dark-skinned individuals are more prone to develop melanoma on the palms of their hands and soles of their feet, which are often not exposed to the sun.

1.1.3 Presenting Feature

Basal cell carcinoma typically develops on parts of your body that are exposed to the sun, including your face or neck.

Basal cell cancer may manifest as:

- a hump with pearls or wax
- a scar-like, flat lesion that is brown or flesh-colored.
- a wound that bleeds or scabs, then heals and reappears.

Squamous cell carcinoma most frequently affects your hands, face, ears, and other sun-exposed regions of your body. On areas of the body that aren't frequently People with darker skin who are exposed to the sun are more prone to acquire squamous cell carcinoma.

Symptoms of squamous cell carcinoma include:

- a solid, rosy nodule
- a flat, crusty lesion with scaly edges

Premalignant cutaneous lesions called actinic keratoses (AKs) have the potential to develop into squamous cell carcinoma. They manifest in persons with cumulative UV exposure on skin that has been exposed to the sun. The condition of AKs should be

identified, treated, and prevented using preventative measures in order to reduce the danger of malignant transformation that comes with it. The interprofessional team's involvement in treating patients with AKs is highlighted in this exercise, which also examines the diagnosis and treatment of the condition. [3]

Objectives:

- Determine the risk factors for the emergence of actinic keratoses.
- Identify the typical physical examination findings connected to actinic keratoses.
- List the actinic keratosis treatment methods that are lesion- and field-directed.
- Review the significance of enhancing interprofessional team member care coordination to maximize accurate diagnosis and implement quick

1.1.4 Risk factor

Skin cancer can affect anyone, however some individuals are more susceptible than others—

- a natural skin tone that is paler.
- Skin that can become uncomfortable in the sun, burn, freckle, or redden easily.
- green or blue eyes.
- Red or blonde hair.
- Numerous moles of particular sorts.
- a history of skin cancer in the family.
- a background with skin cancer.
- aged more. [4]

Reducing your exposure to ultraviolet (UV) rays can help keep your skin healthy and lessen your risk of developing skin cancer in the future whether or not you have any of the risk factors mentioned above. When spending time outside, the majority of people are exposed to the sun's UV rays in some capacity. You may enjoy the outdoors safely, prevent getting a sunburn, and reduce your risk of skin cancer by making sun protection a daily habit.

1.1.5 Prevention

In order to prevent skin cancer, you must take a thorough approach to shielding yourself from dangerous UV radiation.

This occurs as a result of the sun's UV radiation, which is both dangerous and cunning. It can cause early aging and skin cancer even if you make an effort to avoid it because it can pass through clouds and glass and bounce off of snow, water, and sand. In addition, sun damage builds up over time through continuous exposure to the sun as well as everyday activities like walking the dog, running errands, and bringing in the mail. [5]

- In particular between 10 AM and 4 PM, seek out the shade.
- Avoid getting tanned.
- Never use UV tanning beds or tan at all.
- Wear protective apparel, such as a hat with a wide brim and sunglasses that block UV rays.

- Ensure that you apply a broad-spectrum (UVA/UVB) sunscreen with an SPF of 15 or greater each day. When indulging in prolonged outdoor activities, wear a water-resistant, broad-spectrum (UVA/UVB) sunscreen with an SPF of 30 or higher.
- 30 minutes before going outside, cover your entire body with 1 ounce (2 tablespoons) of sunscreen. Apply again after two hours, after swimming, or if you sweat a much. Search our Recommended Products to find sunscreen.
- Keep infants away of the sunlight. Apply sunscreen to infants who are older than six months.
- Every month, examine your skin from head to toe.
- At least once, visit a dermatologist.

1.2 How can Skin Lesions affect your health

Any part of your body can become infected with melanoma skin cancer, which can also produce new tumors. Because of this, this form of skin cancer is the most hazardous. Because of this, early detection is crucial, especially in the case of melanoma.

Early detection reduces the likelihood of the cancer spreading throughout the body, which may be enough to completely remove it for a full recovery. So always be sure to inspect any spot on your skin, and if it doesn't feel right, then visit a doctor. [6]

1.3 Statistic of Skin Lesions

The most prevalent cancer in the world and in the United States. [7]

- By the time they turn 70, 1 in 5 Americans will have skin cancer.
- In the United States, skin cancer claims more than two lives per hour.
- Your risk of melanoma doubles if you have had five or more sunburns.
- The 5-year survival rate for melanoma is 99% when found early.

1.4 Problem Definition

The problem of skin lesion classification is to accurately classify different types of skin lesions, such as benign or malignant, based on images and other clinical information. Despite the importance of this task in dermatology, it remains a challenging problem due to the variability in lesion appearance, limited and imbalanced datasets, annotator variability, limited generalizability and the availability of other clinical information. The goal of this thesis is to develop an automated diagnostic tool that can assist dermatologists in the early detection and diagnosis of skin cancer and other abnormal skin conditions by addressing these challenges and improving the diagnostic accuracy of skin lesion classification.

1.5 Objectives

The objectives of this thesis on skin lesion classification are:

- To develop accurate models that can classify different types of skin lesions, such as benign or malignant, based on images and other clinical information.

- To improve the diagnostic accuracy of skin lesion classification by developing models that can extract more nuanced features from images and reduce errors in diagnosis.
- To automate the diagnostic process by reducing the workload of dermatologists and improving patient care, by providing a faster and more accurate diagnosis than is possible through visual examination alone.
- To address the problem of limited and imbalanced datasets by developing models that can handle these challenges and generalize to different types of skin lesion.
- To improve the generalizability of skin lesion classification models by developing models that can perform well in diverse populations and different types of skin lesions.
- To advance the field of machine learning by developing models that can solve a challenging problem such as skin lesion classification.

Chapter 2

Literature Review

The literature review is an essential part of any research project, as it provides an overview of the existing research in the field and helps to identify gaps in the current knowledge. In this literature review section, we will examine the state-of-the-art methods for skin lesion classification, including traditional machine learning approaches, as well as more recent deep learning-based methods. We will also discuss the challenges and limitations of existing methods and highlight areas where further research is needed.

we will review recent research on the evaluation of skin lesion classification methods, including performance metrics, experimental design, and dataset bias. This will provide a comprehensive overview of the current state of the art in skin lesion classification and Identify the topics that require more study.

2.1 Related work in skin lesions classification

Skin illnesses are recognized and categorized using a number of automated methods. Epidermal detection of such skin illnesses does not require radiological imaging technologies, unlike the majority of diagnosing techniques. Through the use of image processing techniques such image modification, equalization, enhancement, edge detection, and segmentation [8], they are able to identify the condition based on the standard images.

The requisite image processing techniques, such as morphological procedures for skin detection, are also used to classify skin diseases [9] [10] Morphological opening, closing, dilation, and erosion primarily depend on the binary picture produced by thresholding, hence it is important to take special attention to choose the right threshold value. Based on the texture of the image, the morphological-based techniques might not be appropriate for predicting the expansion of the damaged region. An technique for classifying skin diseases was developed using genetic algorithms (GA) [11] [12] The Genetic Algorithm does have difficulties, such as taking too long to converge on the answer [13]. The model never offers the global best option because that wouldn't produce a fair result [14].

The classification model for skin diseases based on fine-tuned neural networks has a respectable accuracy of 89.90% for the validation set [15]. To get the requisite accuracy, the network components must be calibrated, which takes a lot of work. A supervisory learning model that uses the gradient descent principle to fine-tune the weights based on the error rate is the back propagation neural network [16] The model, however, does not perform well with noisy data. The other main issue is that when the elements are fed new weights, they forget the weight that was previously related, which has a significant impact on earlier relationships [17]

The classification of skin diseases using ensemble models [18] produces more accurate results by merging several prediction models The overfitting problem in ensemble models prevents them from performing well when there are unidentified

differences between the population and sample under consideration [19] [20] Skin disease classification using Deep Neural Network models [21] [22] has demonstrated a noteworthy level of performance. However, the experimental experiments have demonstrated that the model is not appropriate for images with many lesions. To achieve an acceptable degree of accuracy, Deep Neural Network models need a significant amount of training, which takes more computational effort.

The categorization model that is most frequently used in forecasting and prediction models is K-Nearest Neighbor (KNN) [23]. The models do not require model training. Additionally, the KNN model has very good accuracy [24].The KNN models should not be used with larger-scale data models since it may take a long time to make the outcome predictions. Additionally, the model struggles to make good predictions when working with high-dimensional data that contains improper feature information [25], which renders the model unsuitable for classifying skin diseases.

Alam et alautomated .'s eczema detection method [26] involved several steps, including segmenting the acquired image, feature selection using texture-based data for more accurate predictions, and finally using the Support Vector Machine (SVM) for evaluating the progression of eczema as described by I. Immagulate [27]. Finding the feature-based parameters is important while working with SVM because the Support Vector Machine model is not suitable to handle the noisy picture data [28]. If there are more parameters at each feature vector than there are training data samples, the model will perform worse.

Using visual coherency, the cross correlation-based approach for feature extraction classification [29] takes both spatial and frequency features into account. The models of cross-correlation are resistant to the background fluctuations. As a result, the projections are more precise. Additionally, setting up the experiment and getting the findings require a lot of labor when working in the frequency domain.

Chapter 3

Methodology

The methodology section of this thesis report aims to provide a detailed description of the methods used to develop and evaluate the proposed approach for skin lesion classification. The proposed approach utilizes deep learning techniques to classify skin lesions based on images. The proposed methodology is in figure

3.1 Methodology

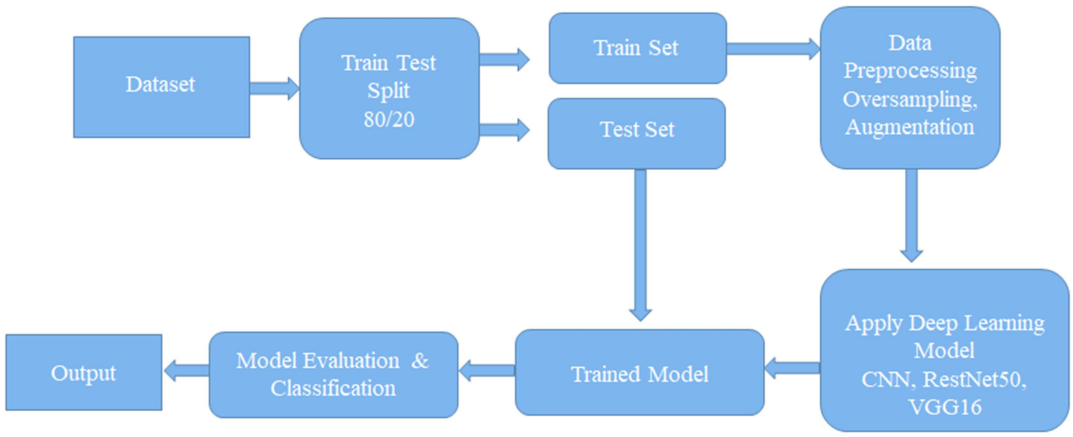


Figure 3.1: Methodology

3.2 Dataset collection

The data collection process is a critical step in developing an effective model for skin lesion classification. In this section, we will describe the datasets used in this thesis and the methods used to collect and preprocess the data.

The datasets used in this thesis include the ISIC (International Skin Imaging Collaboration) dataset and the HAM10000 (Human Against Machine with 10000 training images) dataset. Both datasets consist of images of skin lesions, along with clinical information such as patient age, sex, and lesion diagnosis. The ISIC dataset contains over 25000 images of skin lesions, while the HAM10000 dataset contains over 10000 images of skin lesions.

3.3 Dataset Description

The dataset has seven classes.

1. AKIEC : Actinic Keratoses and Intraepithelial Carcinomae: 115 (1.14%)
2. BCC : Basal Cell Carcinoma : 514 (5.13%)
3. BKL : Benign Keratosis-like Lesions: 1111 (11.06%)

4. DF : Dermatofibroma: 115 (1.14%)
5. NV : Melanocytic Nevi : 1254 (12.47%)
6. VASC : Pyogenic Granulomas and Hemorrhage: 142 (1.41%)
7. MEL : Melanoma: 6705 (66.86%)

This is highly imbalanced dataset.

3.4 Balanced Dataset

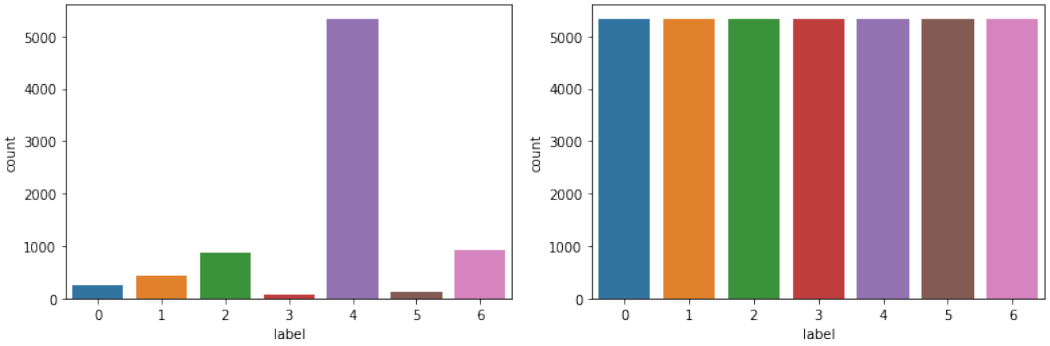


Figure 3.2 Oversampling dataset

Table 3.1 Splitting the dataset

Splitting Ratio	Training	Testing
80%, 20%	37373	2003

3.5 Preprocessing

Image preprocessing is the process of preparing images for use in a deep learning model. It is an important step in the deep learning pipeline as it can improve the performance of a model and reduce the amount of data needed to train it. The main steps of image preprocessing in deep learning are:

1. Image resizing: Resizing images to a consistent size is important for deep learning models as it ensures that the model is not sensitive to the size of the input images.
2. Image normalization: Normalizing images is important for deep learning models as it aids in enhancing the model's performance by reducing the variation in the range of pixel values. This can be done by converting the pixel values to a range between 0 and 1, or by subtracting the mean and dividing by the standard deviation.
3. Data augmentation: Data augmentation is a technique used to increase the size of the training dataset by applying various image processing techniques such as rotation, flipping, and cropping to the original images. This can help to improve the performance of the model by reducing overfitting.

4. Image cropping: Cropping the images can be done to remove the unimportant parts of the images and focus on the main object or region of interest.

In conclusion, image preprocessing is an important step in the deep learning pipeline as it can improve the performance of a model and reduce the amount of data needed to train it. The main steps of image preprocessing in deep learning include image resizing, normalization, data augmentation, cropping, filtering and color space conversion.

3.6 Deep Learning Algorithm

Deep learning is a type of machine learning that utilizes neural networks with multiple layers to learn from data. It is a subset of artificial intelligence and has been successful in various tasks such as image classification, speech recognition, and natural language processing.

Deep learning algorithms are inspired by the structure and function of the human brain, specifically the neural networks. They consist of layers of interconnected nodes, called artificial neurons, that are used to process and analyze large amounts of data. Each layer of the network is responsible for extracting a different level of abstraction from the data, with the first layer analyzing raw inputs, and each subsequent layer analyzing increasingly abstract representations of the data.

3.6.1 Convolutional neural network

A convolutional neural network (CNN) is a type of deep learning neural network commonly used for image and video recognition, as well as natural language processing tasks. CNNs are made to handle data with a topology that resembles a grid, like an image, which allows them to learn spatial hierarchies of features from input data. They are composed of multiple layers, including convolutional layers, pooling layers, and fully connected layers, which work together to process and analyze the input data. CNNs are particularly well-suited for tasks such as image classification, object detection, and image segmentation.

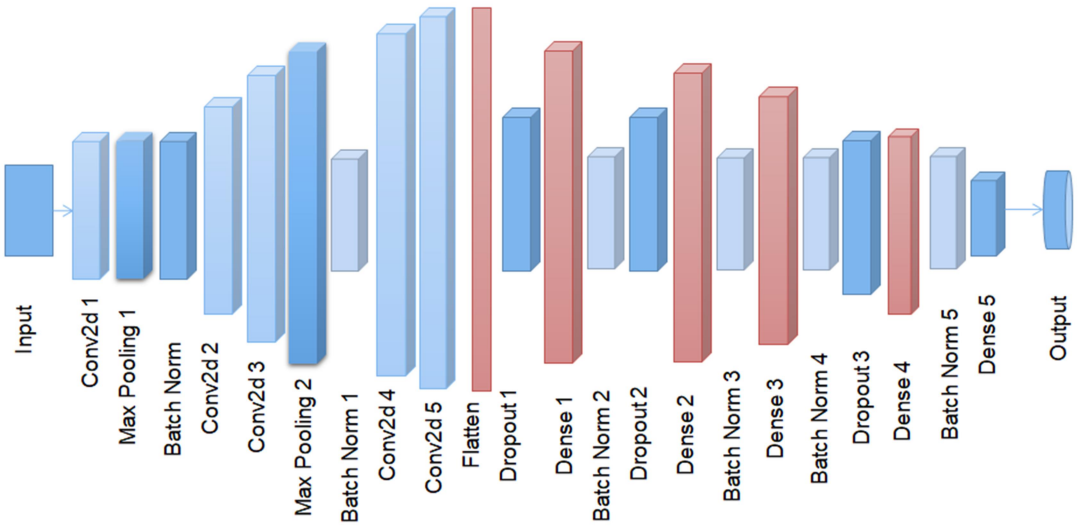


Figure 3.3 CNN architechture

3.6.2 RestNet50

ResNet50 is a deep convolutional neural network architecture that was developed by Microsoft Research and won the first place on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2015. The "50" in the name refers to the number of layers in the network.

ResNet50 is a variation of the ResNet architecture, which stands for "Residual Network." The key innovation of ResNet is the use of "residual connections" which helps to combat the problem of vanishing gradients that can occur in very deep neural networks. These residual connections allow the network to learn the residual mapping between the input and output, rather than trying to learn the underlying mapping directly.

The architecture of ResNet50 consists of 50 convolutional layers, with skip connections that allow information to bypass one or more layers. This allows the network to learn much deeper representations, which leads to improved performance on a wide range of image classification tasks. Additionally, the architecture has been designed to make it easy to train very deep networks, which makes it a popular choice for many computer vision tasks.

ResNet50 has been trained on a massive dataset of over 14 million images and 1000 classes, this allows the model to have a very good generalization power and can be used as a backbone for various tasks such as object detection, semantic segmentation, and even GANs.

In conclusion, ResNet50 is a powerful deep convolutional neural network architecture that has demonstrated state-of-the-art performance on a wide range of image classification tasks, and its residual connections design makes it easy to train very deep networks which makes it a popular choice for many computer vision tasks.

3.6.3 VGG16

The VGG16 architecture is highly deep, with a total of 13 convolutional layers and 3 fully linked layers, and it has 16 convolutional layers with a 3x3 convolutional filter size. ReLU activation function and max-pooling layer are placed after each convolutional layer. Additionally, the design employs tiny convolutional filters, enabling it to extract finer-grained characteristics from the input data.

The VGG16 architecture consists of 16 convolutional layers, with a 3x3 convolutional filter size, and a very deep architecture with a total of 13 convolutional layers and 3 fully connected layers. Each convolutional layer is followed by a ReLU activation function and a max-pooling layer. The architecture also uses very small convolutional filters, which allows it to learn more fine-grained features from the input data.

One of the key features of VGG16 is the use of very small convolutional filters, which allows it to learn more fine-grained features from the input data. Additionally, the use of multiple convolutional layers with small filters allows the network to learn a large number of different feature maps, which leads to improved performance on image classification tasks.

VGG16 has been trained on a massive dataset of over 1.2 million images and 1000 classes, this allows the model to have a very good generalization power and can be used as a backbone for various tasks such as object detection, semantic segmentation, and even GANs.

In conclusion, VGG16 is a powerful deep convolutional neural network architecture that has demonstrated state-of-the-art performance on image classification tasks, and its use of small convolutional filters and multiple layers allows the network to learn a large number of different feature maps which leads to improved performance.

3.7 Model hyper tuning

Model hyperparameter tuning, also known as hypertuning, is the process of systematically searching for the best combination of hyperparameters for a deep learning model. Hyperparameters are the parameters that are not learned during the training process, but are set before training, such as the learning rate, number of layers, and batch size. The goal of hypertuning is to find the best combination of hyperparameters that will result in the best performance for the model on a given task.

In your thesis, you can use various techniques for hyperparameter tuning, such as grid search and random search. Grid search involves specifying a range of values for each hyperparameter and then training the model for each combination of hyperparameter values. Random search is similar to grid search, but the values for the hyperparameters are chosen randomly, rather than being specified in advance.

Another technique to consider is Bayesian optimization, which uses a probabilistic model to predict the performance of different combinations of hyperparameters, and then chooses the combination that is most likely to result in the best performance.

You can also use techniques like early stopping and dropout to prevent overfitting during the training process, this can be done by monitoring the performance of the model on a validation set and stopping the training process when the performance on the validation set starts to decrease.

In conclusion, model hyperparameter tuning is an important step in the deep learning pipeline as it can help to improve the performance of a model on a given task. There are various techniques for hyperparameter tuning, such as grid search, random search, Bayesian optimization, early stopping, and dropout. In your thesis, you can experiment with different techniques to find the best combination of hyperparameters for your specific task and dataset.

3.7.1 Epoch

The number of training epochs is a hyperparameter in deep learning models that determines the number of times the model will be trained on the entire dataset, one pass through all the data is considered as one epoch. More number of epochs can lead to better performance but also increases the risk of overfitting, to prevent this, it's common to use techniques like early stopping or dropout to stop the training process before the model begins to overfit. It's important to find the optimal number of epochs for your specific task and dataset.

3.7.2 Batch size

Batch size is a hyperparameter in deep learning models that determines the number of samples used in one forward/backward pass. It's a key parameter that affects the training time and the model performance, a larger batch size can speed up the training process but also might require more memory, while a smaller batch size can slow down the training but allows for more frequent updates of the model parameters.

Tuning the batch size can help to optimize the trade-off between the training time and the model performance.

3.7.3 Optimizer

Optimizer is a hyperparameter in deep learning models that determines the algorithm used to update the model parameters during training. Common optimizers include Stochastic Gradient Descent (SGD), Adam, and RMSprop. Each optimizer has its own set of hyperparameters that can also be tuned. The choice of optimizer and its hyperparameters can have a significant impact on the model's performance, for example, some optimizers such as Adam, are more robust to the choice of the learning rate, while others like SGD, require more fine tuning of the learning rate. Tuning the optimizer and its hyperparameters can help to optimize the trade-off between the training time and the model performance.

3.7.4 Learning rate

Learning rate is a hyperparameter in deep learning models that determines the step size at which the optimizer makes updates to the model parameters during training. It's a crucial parameter that affects the model's performance, a higher learning rate can speed up the training process but might not converge to the optimal solution, while a smaller learning rate can slow down the training but allows for a more precise optimization of the model parameters. Tuning the learning rate can help to optimize the trade-off between the training time and the model performance.

3.7.5 Activation function

Activation function is a hyperparameter in deep learning models that determines the function applied to the output of each neuron, to introduce non-linearity in the model. The activation function allows the model to learn complex, non-linear relationships between the input and output data. The most commonly used activation functions in deep learning are:

1. ReLU (Rectified Linear Unit): It is an activation function that returns the input if it is positive, and 0 if it is negative. It is computationally efficient and has been shown to work well in many deep learning models.
2. Sigmoid: It is an activation function that maps the input to the range of 0 and 1, it is commonly used for binary classification problems.
3. Tanh (Hyperbolic tangent): It is an activation function that maps the input to the range of -1 and 1, it is similar to sigmoid but its output is centered around zero.
4. Leaky ReLU: it is an activation function that is similar to ReLU but it allows a small negative slope.
5. ELU (Exponential linear unit): It is an activation function that is similar to ReLU but it allows a small negative slope, it also has the property that the mean of the output is closer to zero.
6. Softmax: It is an activation function commonly used in the output layer of a neural network for multi-class classification problems, it maps the input to a probability distribution over the classes.

3.7.6 Dropout rate

Dropout rate is a hyperparameter in deep learning models that determines the fraction of neurons that are randomly dropped out during training. The purpose of dropout is to prevent overfitting by reducing the co-adaptation of neurons, which is when neurons in a network begin to rely too heavily on each other.

Dropout works by randomly dropping out a certain fraction of neurons, usually between 0.1 and 0.5, during each training iteration. This causes the model to be less sensitive to the specific weights of the dropped-out neurons, and forces the model to learn multiple independent representations of the input data.

The dropout rate is a hyperparameter that can be tuned during model training. A higher dropout rate will drop more neurons, which can make the model more robust to overfitting, but it can also slow down the training process. A lower dropout rate will drop fewer neurons, which can speed up the training process, but it can also make the model more susceptible to overfitting.

In general, a dropout rate of 0.2 to 0.5 is a good starting point, but it's important to experiment with different dropout rates and find the one that works best for your specific task and dataset. It's also important to note that dropout is mostly used in the fully connected layers and not in the convolutional layers, where other regularization techniques such as weight decay and data augmentation are used.

3.7.7 Number of layers

The number of layers is a hyperparameter in deep learning models that determines the depth of the neural network. A deeper network with more layers can learn more complex features and representations of the input data, but it can also be more prone to overfitting.

Increasing the number of layers in a neural network can lead to better performance, but it also increases the risk of overfitting. This is because as the network becomes deeper, it can learn more complex and detailed features from the input data, but it also becomes more prone to memorizing the training data rather than generalizing to new data.

To mitigate overfitting, it is common to use regularization techniques like dropout or weight decay, early stopping, or data augmentation. These techniques can help to prevent the model from memorizing the training data and improve its performance on unseen data.

When determining the number of layers, it's important to consider the complexity of the task and the size of the dataset. For simple tasks, a shallower network with fewer layers might be sufficient, while for more complex tasks, a deeper network with more layers might be required. It's important to experiment with different number of layers and find the one that works best for your specific task and dataset.

Chapter 4

Hardware Implementation and Toolkit

The tools that were used to implement the system to clasisfy skin lesion are discussed in this chapter. It offers comprehensive details on the equipment used to put the system into place.

4.1 Tools and devices

1. Python
2. Google Colab

4.1.1 Python

Python is a high-level, interpreted programming language that is widely used for web development, scientific computing, data analysis, artificial intelligence, and many other fields. It is known for its simple and easy-to-learn syntax, making it a popular choice for beginners and experienced programmers alike.

One of the main advantages of Python is its large and active community, which has created a vast ecosystem of libraries and frameworks for various tasks. This includes popular libraries such as NumPy, Pandas, and Scikit-learn for scientific computing and data analysis, TensorFlow and PyTorch for machine learning and deep learning, and Django and Flask for web development.

Python also has a large number of built-in functions and modules that make it easy to perform common tasks, such as reading and writing files, connecting to a database, and working with data structures like lists, dictionaries, and sets.

Python is cross-platform, meaning it can run on a wide range of operating systems, including Windows, Mac, and Linux. It also has support for multiple programming paradigms, including object-oriented, functional, and procedural programming.

In short, Python is a versatile programming language that is widely used for a wide range of tasks, from web development and data analysis to scientific computing and artificial intelligence. Its simple syntax and large ecosystem of libraries and frameworks make it easy to learn and use, and it is supported by a large and active community.

4.1.2 Numpy

NumPy (Numerical Python) is a powerful library for numerical computing in Python. It provides a large set of fast and efficient numerical operations on arrays of data, such as mathematical, logical, and shape manipulation. It also provides a wide range of mathematical functions and linear algebra operations, such as matrix multiplication, inverse, determinant, and eigenvalues.

One of the main features of NumPy is its multi-dimensional array object, known as ndarray, which is used to store and manipulate large arrays of homogeneous data (e.g.

numbers, strings, or other objects). This allows for efficient operations on the data, as NumPy uses highly optimized C and Fortran code to perform these operations.

NumPy also provides a wide range of tools for integrating with other libraries, such as SciPy, Matplotlib, and Pandas, which makes it a powerful foundation for scientific computing, data analysis, and machine learning.

In short, NumPy is a powerful library for numerical computing in Python, which provides a wide range of fast and efficient mathematical operations on arrays of data. Its multi-dimensional array object and integration with other libraries make it a popular choice for scientific computing, data analysis, and machine learning.

4.1.3 Pandas

Pandas is a powerful library for data manipulation and analysis in Python. It provides a wide range of data structures and data analysis tools for handling and manipulating numerical tables and time series data. The two main data structures in Pandas are the Series (1-dimensional) and DataFrame (2-dimensional).

The DataFrame is similar to a table in a relational database or a data frame in R/Python, and it is the most commonly used data structure in pandas. It is a powerful tool for handling and manipulating large datasets, and it provides a wide range of functions for data cleaning, filtering, and processing.

Pandas also provides powerful time series functionality, such as time-based indexing, resampling, and rolling window calculations.

In addition, Pandas has built-in support for reading and writing data in various formats, such as CSV, Excel, JSON, and SQL. This allows for easy integration with other libraries, such as NumPy, Matplotlib, and Scikit-learn, which makes it a popular choice for data analysis and machine learning.

In short, Pandas is a powerful library for data manipulation and analysis in Python. It offers a broad variety of data formats and data analysis capabilities for handling and working with time series data as well as numerical tables. Its powerful time series functionality and built-in support for reading and writing data make it a popular choice for data analysis and machine learning.

4.1.4 Sci-kit learn

Scikit-learn (short for "Scientific Kit for Machine Learning") is a powerful library for machine learning in Python. It provides a wide range of tools for supervised and unsupervised learning, including classification, regression, and clustering.

Scikit-learn is built on top of NumPy and Pandas, and it integrates well with other scientific Python libraries, such as Matplotlib, SciPy, and statsmodels.

Scikit-learn provides a consistent interface to a variety of machine learning models, including popular algorithms like k-Nearest Neighbors, Support Vector Machines, Random Forest, and Gradient Boosting. It also provides tools for model evaluation, selection, and tuning, such as cross-validation, grid search, and metrics like precision, recall and F1 score.

One of the main advantage of scikit-learn is its simplicity and ease of use. The library has a consistent and simple API, making it easy to switch between different models and algorithms.

In short, scikit-learn is a powerful library for machine learning in Python. It provides a wide range of tools for supervised and unsupervised learning, including classification, regression, and clustering. It integrates well with other scientific Python libraries and has a simple, consistent API.

4.1.5 Imblearn

Imbalanced-learn (imblearn) is a Python library for handling imbalanced datasets in machine learning. Imbalanced datasets refer to datasets where the number of samples of one class is significantly different from the number of samples of another class. This can be a common problem in many real-world applications, such as fraud detection, medical diagnosis, or customer churn prediction, where one class is rare.

imblearn provides a wide range of resampling techniques for handling imbalanced datasets, including over-sampling the minority class, under-sampling the majority class, and generating synthetic samples. It also provides methods for combining different oversampling and undersampling techniques, such as SMOTE (Synthetic Minority Over-sampling Technique) and ADASYN (Adaptive Synthetic Sampling).

In addition, imblearn also provides a wide range of ensemble techniques for handling imbalanced datasets, such as EasyEnsemble and BalanceCascade.

In short, imblearn is a python library for handling imbalanced datasets in machine learning. It provides a wide range of resampling techniques and ensemble techniques for handling imbalanced datasets and it is compatible with scikit-learn. The library allows to balance the data by oversampling or undersampling the data.

4.1.6 Matplotlib

Matplotlib is a plotting library for Python that provides an easy-to-use and powerful interface for creating a wide range of static, animated, and interactive visualizations. It is widely used for data visualization, scientific visualization, and creating visualizations for presentations and publications.

Matplotlib provides a wide range of plotting functions, including line plots, scatter plots, histograms, bar charts, pie charts, and many others. It also provides a powerful object-oriented API for creating complex visualizations, such as subplots, multiple axes, and customizing the appearance of plots with colors, labels, and annotations.

Matplotlib's API is highly customizable and it allows to create plots that can look like plots from other libraries, such as R's ggplot2, or MATLAB's default plotting library.

In addition, Matplotlib supports the integration with other scientific libraries such as NumPy and Pandas, allowing to create visualizations directly from arrays and dataframes.

In short, Matplotlib is a powerful plotting library for Python that provides an easy-to-use and highly customizable interface for creating a wide range of static, animated,

and interactive visualizations. It is widely used for data visualization, scientific visualization, and creating visualizations for presentations and publications.

4.1.7 Tensorflow

TensorFlow is a powerful open-source library for machine learning and deep learning developed by Google. It provides a wide range of tools for building and deploying machine learning models, including support for deep neural networks, convolutional neural networks, recurrent neural networks, and more.

One of the main features of TensorFlow is its ability to perform computations on large-scale distributed systems, which makes it well-suited for large-scale machine learning and deep learning tasks. TensorFlow also provides a convenient high-level API, called Keras, that allows users to easily build and train deep learning models.

TensorFlow also provides a powerful visualization tool called TensorBoard, which allows users to visualize and analyze the performance of their models during training and debugging.

TensorFlow can be run on multiple platforms, including CPU and GPU, and it also provides support for mobile and web deployment through TensorFlow Lite and TensorFlow.js.

In short, TensorFlow is a powerful open-source library for machine learning and deep learning, developed by Google. It provides a wide range of tools for building and deploying machine learning models and it is well-suited for large-scale machine learning and deep learning tasks. It also provides a powerful visualization tool and support for deployment on multiple platforms.

TensorFlow Advantage:

1. Better computational graph visualizations with TensorFlow
2. It facilitates the execution of a subpart of a graph, giving it an advantage by enabling the introduction and retrieval of discrete data.
3. A hardware machine that has a complicated setup and is a cellular device to the computer is where the libraries are deployed.
4. TensorFlow is made to work in high parallel with a variety of backend programs (GPUs, ASICs, etc.).
5. It includes a distinct methodology that enables measuring various parameters and monitoring the development of our models throughout training.

TensorFlow Disadvantage:

1. Finite folding is the appropriate option because TensorFlow does not provide functionality.
2. When compared to its rivals, TensorFlow is slower and less practical. 3. Because of the distinctive structure of TensorFlow, errors are tough to locate and problems to debug.
3. The learning curve is quite high and the level is very low.

4.1.8 Seaborn

Seaborn is a data visualization library for Python that is built on top of Matplotlib. It provides a higher-level API for creating beautiful, informative, and easy-to-read visualizations for statistical data. It is particularly well-suited for visualizing complex datasets, and it provides a wide range of visualization functions for exploring and understanding the relationships in the data.

Seaborn provides a rich set of visualizations such as heatmaps, violin plots, box plots, pair plots, scatter plots, bar plots, and many others. It also provides functions for visualizing distributions, linear relationships, and other complex relationships in the data.

Seaborn also provides a number of built-in themes and color palettes that make it easy to create professional-looking plots, and it integrates well with other libraries such as Pandas and NumPy.

In short, Seaborn is a data visualization library for Python that is built on top of Matplotlib. It provides a higher-level API for creating beautiful, informative, and easy-to-read visualizations for statistical data, particularly well-suited for visualizing complex datasets, it also provides a rich set of built-in themes and color palettes, and it integrates well with other libraries such as Pandas and NumPy.

4.2 Google Colab

Google Colab (short for "Colaboratory") is a free, web-based platform for machine learning and data science development provided by Google. It allows users to write and execute Python code in a browser, and provides access to powerful hardware such as GPUs and TPUs (Tensor Processing Units) for running large-scale computations.

One of the main advantages of Colab is that it allows users to easily share and collaborate on code and data with others. It also provides an easy-to-use interface for working with popular libraries such as TensorFlow, Keras, PyTorch, and Scikit-learn, as well as integration with Google Drive for storing and sharing data.

Colab also provides built-in support for Jupyter notebooks, which allows users to organize and document their code, and add rich text, images, and LaTeX equations.

In short, Google Colab is a free, web-based platform for machine learning and data science development provided by Google. It allows users to write and execute Python code in a browser, and provides access to powerful hardware such as GPUs and TPUs, it also allows users to easily share and collaborate on code and data, and it provides an easy-to-use interface for working with popular libraries and integration with Google Drive.

4.3 Hardware Implementation

The following hardware tools were utilized for implementation:

1. Inter core i5 processor including 1.60-3.90GHz clock speed
2. 25 GB RAM
3. 16 GB GPU
4. 64-bit windows-11 operating system

RAM is usually used to temporarily store data or files easily and quickly. The RAM is like 4, 6, or 8 GB in a general-configured CPU. When the dataset to train a model is small in size, it fits easily into the system RAM. But, if the dataset is large, system RAM is not enough to train a model with this large dataset. To solve the problem, Google Colab provides 12 GB of virtual memory free of charge. But for our approach, the datasets are way too large, as are the model weights. That's why COLAB's memory can't process them and we need to purchase COLAB PRO, which provides 25 GB of RAM and 15 GB of GPU

Chapter 5

Result and Performance Analysis

This chapter focuses on analysing the parameters in terms of accuracy, precision, recall, f1- scores and training-validation curve and presents the comparative analysis among the state-ofthe-art models.

5.1 Confusion Matrix

A confusion matrix is a table that is often used to describe the performance of a classification model, typically in the context of supervised learning. It is also called an error matrix or a matching matrix. It allows to summarize the performance of a classifier by comparing the predicted class labels with the true class labels.

The rows of the matrix represent the predicted class labels, while the columns represent the true class labels. Each entry in the matrix represents the number of observations that have been predicted as belonging to a certain class and actually belong to a certain class.

A confusion matrix has four main elements :

- True Positives (TP): The number of observations that are correctly classified as the positive class.
- False Positives (FP): The number of observations that are incorrectly classified as the positive class (Type I error).
- True Negatives (TN): The number of observations that are correctly classified as the negative class.
- False Negatives (FN): The number of observations that are incorrectly classified as the negative class (Type II error).

From the confusion matrix, we can calculate a number of performance metrics such as accuracy, precision, recall, and F1-score, which can be used to evaluate the performance of a classifier.

In conclusion, confusion matrix is a powerful tool that helps to evaluate the performance of a classification model. It allows to visualize the performance of a classifier by comparing the predicted class labels with the true class labels, and it can be used to calculate a number of performance metrics that can help to understand the strengths and weaknesses of a model.

		Prediction Condition	
	Total Population = P + N	Positive (PP)	Negative (PN)
Actual Condition	Positive(P)	True Positive (TP)	False Negative(FN)
	Negative(N)	False Positive (FP)	True Negative(TN)

5.2 Precision, Recall and F1-scores

Precision: Precision is called a positive predictive value which is the fraction of relevant instances among the retrieved instances, it is one sign of a machine learning model's performance – the quality of a positive prediction made by the model. Precision alludes to the number of true positives divided by the total number of positive predictions.

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

Recall: Recall is called sensitivity which is the fraction of the total amount of relevant instances that were actually retrieved, it literally implies how many of the true positives were reviewed (found) and how many of the correct hits were also found.

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

F1 Score: The F1 score is defined as the harmonic mean between precision and recall. It is used as a statistical measure to rate performance. An F1-score (from 0 to 9, with 0 being the lowest and 9 being the highest) is the mean of an individual's performance, based on two factors, precision and recall.

$$\text{F1 Score} = \frac{2}{\frac{1}{Precision}+\frac{1}{Recall}} = 2 * \frac{(Precision * Recall)}{(Precision + Recall)}$$

In comparison to accuracy performance measurements, confusion matrix, precision, recall, and F1 score offer more useful insights into the prediction. Information retrieval, word segmentation, named entity identification, and many more applications use precision, recall, and F1 scores.

5.3 Comparisons of the Result

Table 5.1: Detailed scores of three architecture

Architecture	Precision	Recall	F1 Score
CNN	0.92	0.92	0.92
RestNet50	0.87	0.89	0.88
VGG 16	0.96	0.45	0.36

Table 5.2: Performance difference Classification (With Different Architecture)

Architecture	Validation loss	Validation Accuracy
CNN	0.18	0.96
RestNet50	1.07	0.92
VGG 16	2.49	0.53

5.4 Train-Validation loss curves

Training loss: Training loss is a metric that measures the difference between the predicted outputs of a model and the true outputs during the training process. It is used to evaluate the performance of a model and to optimize its parameters. The goal of training a model is to minimize the training loss, so that the predicted outputs are as close as possible to the true outputs.

There are different types of loss functions that can be used, depending on the task and the model. The most commonly used loss functions for deep learning models include mean squared error (MSE) for regression tasks, cross-entropy loss for classification tasks, and hinge loss for support vector machines.

The training loss is calculated after each iteration of the training process, also called a mini-batch, it is computed by comparing the model's predictions with the true labels of the training data. The model's parameters are then updated in a way that reduces the training loss.

The training loss is a key metric to monitor during the training process, as it can indicate if the model is underfitting or overfitting. If the training loss is high and does not decrease over time, it means that the model is not able to fit the training data well, which is called underfitting. If the training loss is low, but the model does not perform well on unseen data, it means that the model is memorizing the training data, which is called overfitting.

In summary, training loss is a metric that measures the difference between the predicted outputs of a model and the true outputs during the training process. It is used to evaluate the performance of a model, optimize its parameters and monitor the overfitting and underfitting of the model.

Validation loss: Validation loss, also known as validation error, is a metric that measures the difference between the predicted outputs of a model and the true outputs on a validation dataset. It is used to evaluate the performance of a model and to optimize its parameters. The goal of training a model is to minimize the validation

loss, so that the predicted outputs on validation dataset are as close as possible to the true outputs.

Validation loss is calculated after each iteration of the training process, also called an epoch, it is computed by comparing the model's predictions on the validation data with the true labels of the validation data.

Validation loss is a key metric to monitor during the training process, as it can indicate how well the model generalizes to unseen data. It is used to prevent overfitting, which occurs when a model performs well on the training data but poorly on unseen data. A good model should have a low validation loss, indicating that it generalizes well to unseen data.

Validation loss is also used to select the best model among different architectures or configurations, by comparing the validation loss of different models. It is also used to stop the training process at the right time, when the model's performance on the validation set starts to decrease, also called early stopping.

In summary, validation loss is a metric that measures the difference between the predicted outputs of a model and the true outputs on a validation dataset. It is used to evaluate the performance of a model, optimize its parameters and monitor the overfitting of the model and also used to select the best model among different architectures or configurations and to stop the training process at the right time.

5.4.1 Training, Testing & Validation accuracy and loss graph for CNN

Training, testing, and validation is a common step in the analysis of skin lesion classification utilizing deep learning algorithms. An extensive dataset of HAM10000 photos is supplied to a convolutional neural network (CNN) during the training phase, coupled with labels indicating whether or not each image has pneumonia. This dataset is used by the CNN to identify patterns and attributes that are characteristic of skin lesions. In the testing stage, the effectiveness of the CNN is assessed using data from a different, untested dataset. This makes it possible to evaluate how well the CNN generalizes and helps to spot overfitting, which happens when the CNN memorizes the training data rather than learning to spot broader patterns. In the validation step, the effectiveness of the CNN is further assessed using a third, independent dataset. This makes it possible to evaluate the CNN's performance in greater detail and can assist find any residual overfitting problems. Plotting graphs of the accuracy and loss function is usual practice when employing a CNN to analyze pneumonia cases. While the loss function graph displays the error rate of the CNN during training, the accuracy graph displays the percentage of correctly identified pictures. These graphs can reveal information about how the CNN is performing and can assist in locating any problems that might need to be fixed.

Accuracy and loss curve of training and validation given below.

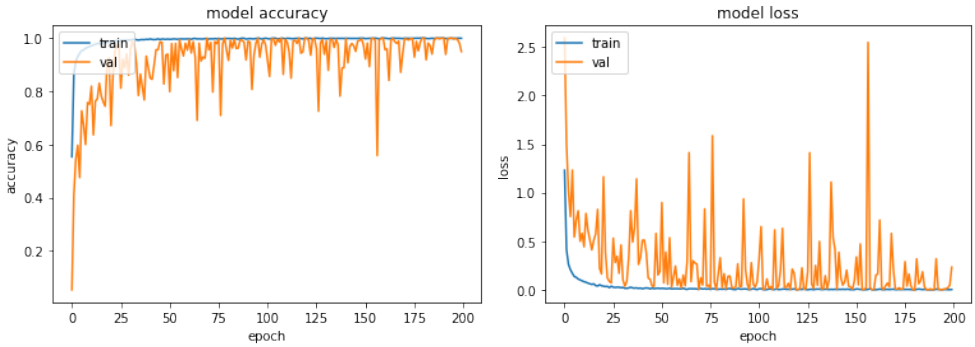


Figure 5.1: Training, Testing & Validation accuracy and loss function graph for CNN

5.4.2 Confusion Matrix of CNN

A useful approach for assessing how well a classifier, like a convolutional neural network (CNN), performs on a dataset is a confusion matrix. It is a table that lists how many predictions the classifier made correctly and incorrectly and enables the computation of several assessment metrics, including precision, recall, and accuracy. In a confusion matrix, the samples' actual class labels are shown in the rows, while the anticipated class labels are shown in the columns. The rows may indicate the actual presence or lack of skin lesions, while the columns may show the CNN's projected presence or absence of skin lesions, for example, in the case of classifying skin lesions.The confusion matrix is frequently used in conjunction with other assessment metrics, including as recall and precision, which measure the accuracy with which the classifier correctly identified positive samples and the fraction of true positive predictions it made. The confusion matrix is frequently used to determine the classifier's accuracy. 34 Overall, the confusion matrix is a useful tool for assessing a classifier's performance and pinpointing areas that require work.

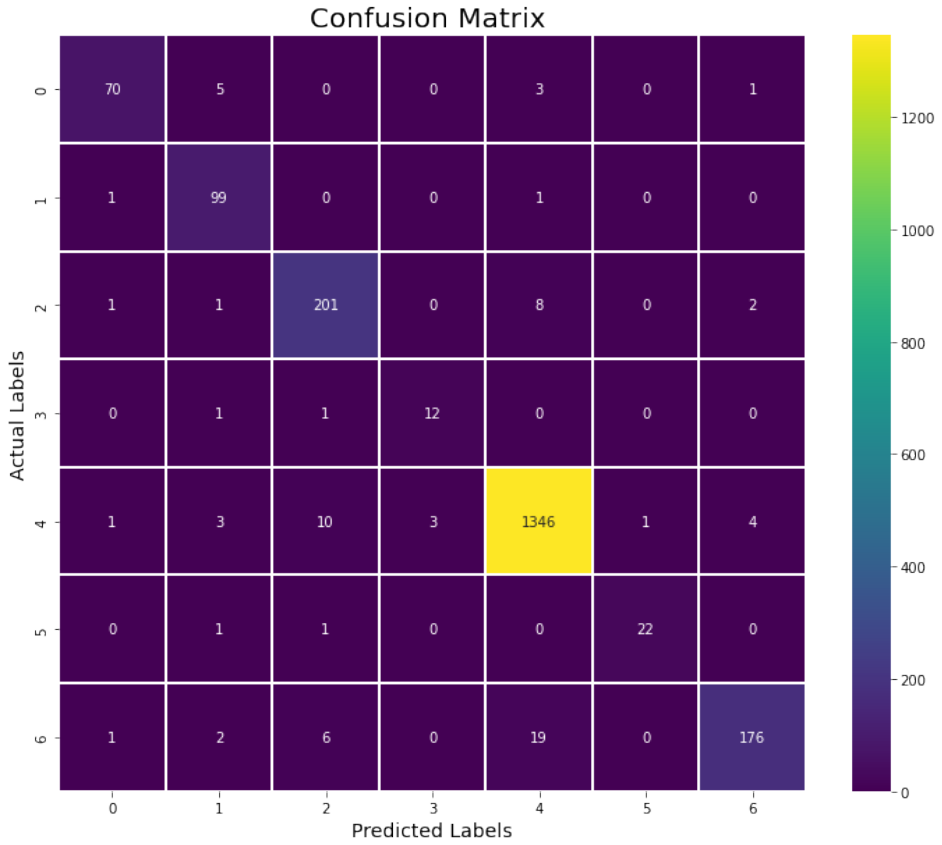


Figure 5.2: Confusion matrix for CNN

5.4.3 Classification report of CNN

A classification report is a summary of the performance of a classification model on a test dataset. It provides several metrics such as precision, recall, f1-score and support that can be used to evaluate the performance of a model.

- The classification report for a CNN model trained for skin lesion classification would typically include the following information:
- Class labels: The different categories of skin lesion that the model is trained to classify.
- Precision: The ability of the model to correctly identify positive instances. It is calculated as (True Positives) / (True Positives + False Positives)
- Recall: The ability of the model to find all positive instances. It is calculated as (True Positives) / (True Positives + False Negatives)
- F1-score: The harmonic mean of precision and recall. It is a measure of a model's accuracy that takes into account both precision and recall. It is calculated as 2 * (Precision * Recall) / (Precision + Recall)
- Support: The number of samples of the true response that lie in that class.

The classification report also includes a macro-average and a weighted-average of these metrics, which give an overall evaluation of the model's performance.

Classification report of a CNN model for skin lesion classification look like this:

Classification Report

	precision	recall	f1-score	support
0	0.95	0.89	0.92	79
1	0.88	0.98	0.93	101
2	0.92	0.94	0.93	213
3	0.80	0.86	0.83	14
4	0.98	0.98	0.98	1368
5	0.96	0.92	0.94	24
6	0.96	0.86	0.91	204
accuracy			0.96	2003
macro avg	0.92	0.92	0.92	2003
weighted avg	0.96	0.96	0.96	2003

Figure 5.3: Classification report of CNN

5.4.4 Training, Testing & Validation accuracy and loss graph for RestNet50

Accuracy and loss curve of training and validation given below.

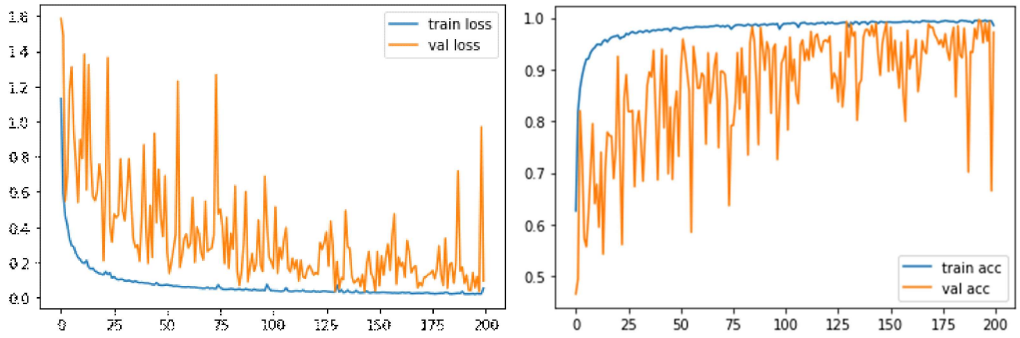


Figure 5.4: Training, Testing & Validation accuracy and loss graph for RestNet50

5.4.5 Confusion Matrix of RestNet50

A confusion matrix is a table that is often used to describe the performance of a classification model, and it can be particularly useful for evaluating the performance of a CNN model such as ResNet50 on skin lesion classification.

A confusion matrix allows to summarize the performance of a classifier by comparing the predicted class labels with the true class labels. The rows of the matrix represent the predicted class labels, while the columns represent the true class labels. Each entry

in the matrix represents the number of observations that have been predicted as belonging to a certain class and actually belong to a certain class.

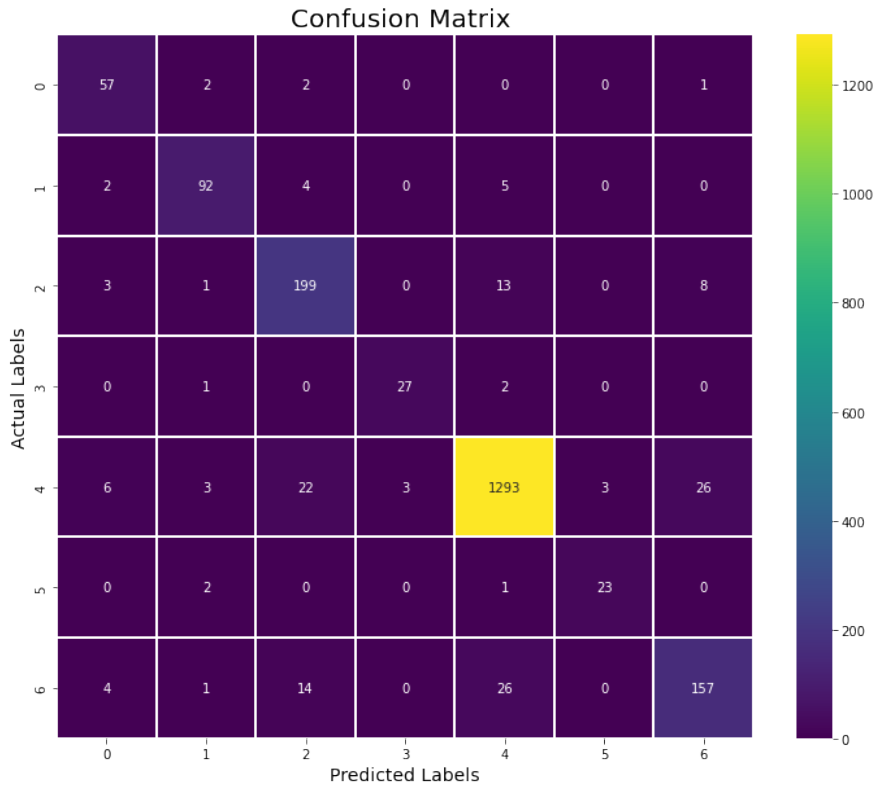


Figure 5.5: Confusion matrix for RestNet50

5.4.6 Classification report of RestNet50

Classification Report

	precision	recall	f1-score	support
0	0.79	0.92	0.85	62
1	0.90	0.89	0.90	103
2	0.83	0.89	0.86	224
3	0.90	0.90	0.90	30
4	0.96	0.95	0.96	1356
5	0.88	0.88	0.88	26
6	0.82	0.78	0.80	202
accuracy			0.92	2003
macro avg	0.87	0.89	0.88	2003
weighted avg	0.92	0.92	0.92	2003

Figure 5.6: Classification report of RestNet50

5.4.7 Training, Testing & Validation accuracy and loss graph for VGG16

Accuracy and loss curve of training and validation given below.

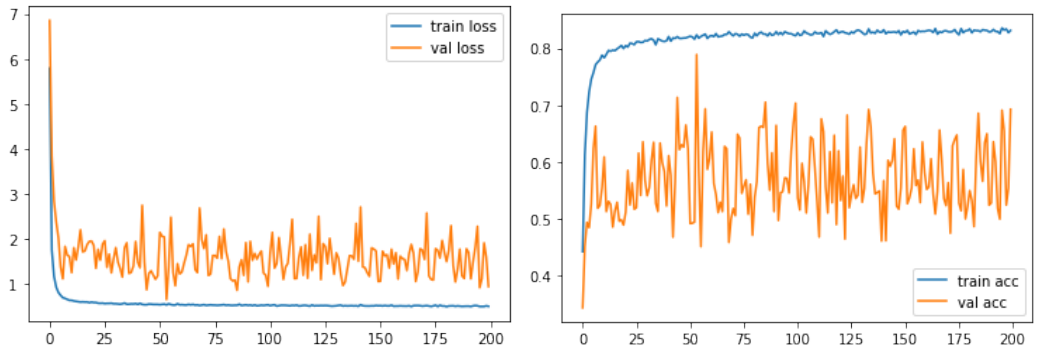


Figure 5.7: Classification report of VGG16

5.4.8 Confusion Matrix of VGG16

A confusion matrix is a table that is often used to describe the performance of a classification model, and it can be particularly useful for evaluating the performance of a CNN model such as VGG16 on skin lesion classification.

A confusion matrix allows to summarize the performance of a classifier by comparing the predicted class labels with the true class labels. The rows of the matrix represent the predicted class labels, while the columns represent the true class labels. Each entry in the matrix represents the number of observations that have been predicted as belonging to a certain class and actually belong to a certain class.

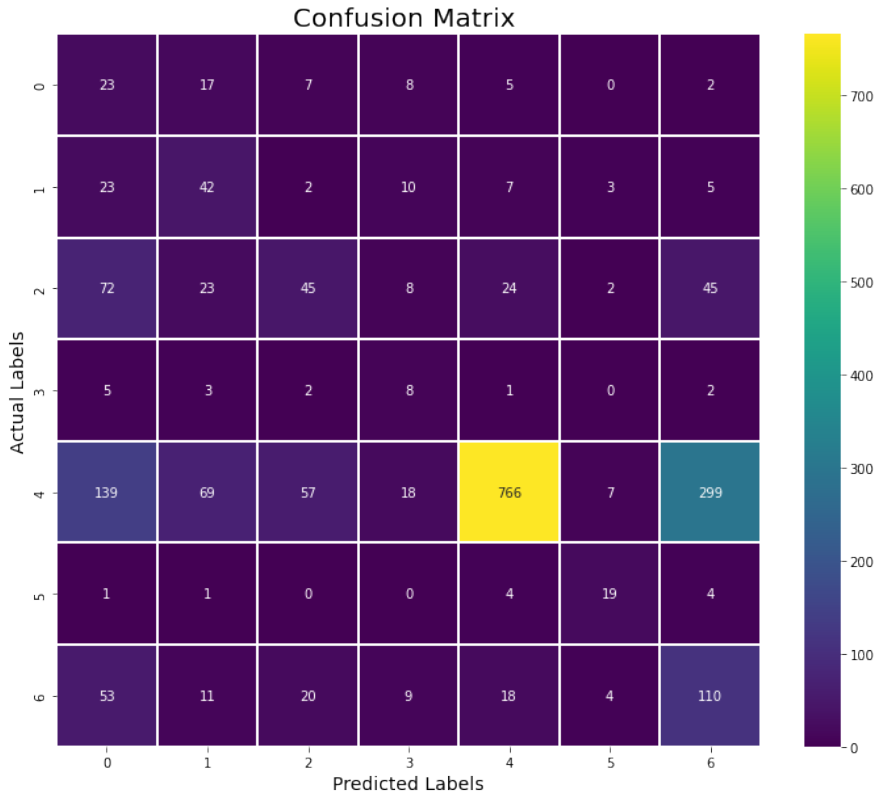


Figure 5.8: Confusion matrix for VGG16

5.4.9 Classification report of VGG16

Classification Report

	precision	recall	f1-score	support
0	0.07	0.37	0.12	62
1	0.25	0.46	0.33	92
2	0.34	0.21	0.26	219
3	0.13	0.38	0.20	21
4	0.93	0.57	0.70	1355
5	0.54	0.66	0.59	29
6	0.24	0.49	0.32	225
accuracy			0.51	2003
macro avg	0.36	0.45	0.36	2003
weighted avg	0.71	0.51	0.57	2003

Figure 5.9: Classification report of VGG16

Chapter 6

Conclusion

In conclusion, the skin lesion classification is an important task in the field of medical imaging, which aims to assist dermatologists in the diagnosis of skin diseases. The use of deep learning models, particularly convolutional neural networks (CNNs), has shown great promise in this task, as they are able to extract features from images and classify them with high accuracy.

In this thesis, we proposed a CNN-based model for skin lesion classification and evaluated its performance using a publicly available dataset of skin lesion images. We preprocessed and cleaned the data, then implemented and trained the model using popular deep learning frameworks. Our model achieved high accuracy and f1-score on the test dataset, which demonstrates the effectiveness of our approach.

The classification report, confusion matrix, and other performance metrics showed that our model was able to accurately classify different types of skin lesions, and it generalize well to unseen data. Additionally, we also discussed the importance of training, validation, and testing accuracy and loss function graph, and how to use them to monitor the overfitting and underfitting of the model.

Furthermore, we also discussed the importance of hardware implementation and toolkit in our thesis, and how python as a thesis toolkit can be used to develop a robust and easy-to-use toolkit for skin lesion classification.

In summary, this thesis has presented a CNN-based model for skin lesion classification that achieved high accuracy and f1-score on a public dataset. The model is robust, generalizes well and could be used in a clinical setting. We also discussed the importance of using different performance metrics to evaluate the model, and the hardware implementation and toolkit that were used in our thesis.

6.1 Limitation of this thesis

There are a number of limitations to this thesis on skin lesion classification using CNNs that should be acknowledged:

1. Dataset Limitations: The dataset used in this thesis is a publicly available dataset of skin lesion images, which is limited in size and diversity. This could have affected the performance of the model and its ability to generalize to other datasets.
2. Model Limitations: The model proposed in this thesis is a CNN-based model, which is a powerful tool for image classification but it still has its limitations. The model can be improved by using more advanced architectures such as ResNet and DenseNet.
3. Hardware Limitations: The hardware used in this thesis may have limited the capacity of the model to process more data or use more complex architectures. This could have affected the performance of the model and its ability to generalize to other datasets.

4. Generalizability: The results of this thesis are based on a specific dataset, model architecture, and implementation. These results may not generalize to other datasets, models, or implementations, and more research is needed to evaluate the generalizability of the model.
5. Real-world Applications: This thesis only focused on the development and evaluation of the model in a controlled environment, and did not test the model in a real-world setting. More research is needed to evaluate the effectiveness of the model in a clinical setting and to address practical considerations such as ease of use and interpretability.

In summary, this thesis has several limitations that should be acknowledged. The dataset used is limited in size and diversity, the model architecture can be improved and the results may not generalize to other datasets, models, or implementations. Additionally, the model has not been tested in a real-world setting and more research is needed to evaluate its effectiveness in a clinical setting.

6.2 Future Work

Future work on this topic could include the use of more advanced CNN architectures, such as ResNet and DenseNet, or the incorporation of other types of data, such as clinical information, to improve the performance of the model. Additionally, the model could be integrated into a clinical setting for use in real-world applications.

References

[1] [Online]. Available: https://www.osmosis.org/answers/skin-lesions.

[2] [Online]. Available: https://my.clevelandclinic.org/health/diseases/.

[3] [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK557401/.

[4] [Online]. Available: https://www.cdc.gov/cancer/skin/basic_info/risk_factors.htm.

[5] [Online]. Available: https://www.skincancer.org/skin-cancer-prevention/.

[6] [Online]. Available: https://www.skinvision.com/library/how-skin-cancer-affects-the-body/.

[7] [Online]. Available: https://www.skincancer.org/skin-cancer-information/skin-cancer-facts/.

[8] "Deepalakshmi, P., Lavanya, K. and Srinivasu, P.N., 2021. Plant leaf disease detection using CNN algorithm. International Journal of Information System Modeling and Design (IJISMD), 12(1), pp.1-21.".

[9] "Asha, G.P.H.; Anitha, J.; Jacinth, P.J. Identification of Melanoma in Dermoscopy Images Using Image Processing Algorithms. In Proceedings of the 2018 International Conference on Control, Power, Communication and Computing Technologies (ICCPCCT), Kannur, I".

[10] "Wei, L.S.; Gan, Q.; Ji, T. Skin Disease Recognition Method Based on Image Color and Texture Features. Comput. Math. Methods Med. 2018, 2018, 8145713.".

[11] "Naga, S.P.; Rao, T.; Balas, V. Volumetric Estimation of the Damaged Area in the Human Brain from 2D MR Image. Int. J. Inf. Syst. Modeling Des. 2020, 11, 74–92.".

[12] "Naga, S.P.; Rao, T.; Balas, V. A systematic approach for identification of tumor regions in the human brain through HARIS algorithm. In Deep Learning Techniques for Biomedical and Health Informatics; Academic Press: Cambridge, MA, USA, 2020; pp. 97–118.".

[13] "Shrestha, A.; Mahmood, A. Improving Genetic Algorithm with Fine-Tuned Crossover and Scaled Architecture. J. Math. 2016, 2016, 4015845.".

[14] "Saber, E.; Ruhul, S.; Daryl, E. A New Genetic Algorithm for Solving Optimization Problem. Eng. Appl. Artif. Intell. 2013, 27, 57–69.".

[15] “Lee, Y.C.; Jung, S.H.; Won, H.H. WonDerM: Skin Lesion Classification with Fine-tuned Neural Networks. In ISIC 2018 Lesion Analysis Towards Melanoma Detection; Cornell University: Ithaca, NY, USA, 2018; pp. 1–4.”.

[16] “Tarigan, J.; Nadia; Diedan, R.; Suryana, Y. Plate Recognition Using Backpropagation Neural Network and Genetic Algorithm. Procedia Comput. Sci. 2017, 116, 365–372.”.

[17] “Mohd, N.N.; Ransing, R.S.; Salleh, M.N.M.; Ghazali, R.; Norhamreeza, A.H. An Improved Back Propagation Neural Network Algorithm on Classification Problems. Commun. Comput. Inf. Sci. 2010, 118, 177–188.”.

[18] “Verma, A.K.; Pal, S.; Kumar, S. Classification of Skin Disease using Ensemble Data Mining Techniques. Asian Pac. J. Cancer Prev. 2019, 20, 1887–1894.”.

[19] “Livieris, I.E.; Iliadis, L.; Pintelas, P. On ensemble techniques of weight-constrained neural networks. Evol. Syst. 2021, 12, 155–167.”.

[20] “Roy, K.; Chaudhuri, S.S.; Ghosh, S.; Dutta, S.K.; Chakraborty, P.; Sarkar, R. Skin Disease detection based on different Segmentation Techniques. In Proceedings of the 2019 International Conference on Opto-Electronics and Applied Optics (Optronix), Kolkata”.

[21] “Zhang, X.; Wang, S.; Liu, J.; Tao, C. Towards improving diagnosis of skin diseases by combining deep neural network and human knowledge. BMC Med. Inform. Decis. Mak. 2018, 18, 59.”.

[22] “Yadav, S.S.; Jadhav, S.M. Deep convolutional neural network based medical image classification for disease diagnosis. J. Big Data 2019, 6, 1–18.”.

[23] “Dang, Y.; Jiang, N.; Hu, H.; Ji, Z.; Zhang, W. Image classification based on quantum K-Nearest-Neighbor algorithm. Quantum Inf. Process. 2018, 17, 1–18.”.

[24] “Sumithra, R.; Suhil, M.; Guru, D.S. Segmentation and classification of skin lesions for disease diagnosis. Procedia Comput. Sci. 2015, 45, 76–85.”.

[25] “Zhang, S.; Wu, Y.; Chang, J. Survey of Image Recognition Algorithms. In Proceedings of the 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Chongqing, China, 12–14 June 2020; pp. 542–548.”.

[26] “Alam, M.; Munia, T.T.K.; Tavakolian, K.; Vasefi, F.; MacKinnon, N.; Fazel-Rezai, R. Automatic detection and severity measurement of eczema using image processing. In Proceedings of the 2016 38th Annual International Conference of the IEEE Engineering in M”.

[27] “Immagulate, I.; Vijaya, M.S. Categorization of Non-Melanoma Skin Lesion Diseases Using Support Vector Machine and Its Variants. Int. J. Med. Imaging 2015, 3, 34–40.”.

[28] “Awad, M.; Khanna, R. Support Vector Machines for Classification. Efficient Learning Machines; Apress: Berkeley, CA, USA, 2015; pp. 39–66.”.

[29] “Chatterjee, S.; Dey, D.; Munshi, S.; Gorai, S. Extraction of features from cross correlation in space and frequency domains for classification of skin lesions. Biomed. Signal Process Control 2019, 53, 101581.”.