DESIGN AND DEVELOPMENT OF DECISION SUPPORT SYSTEM FOR SKIN DISEASE DETECTION USING DEEP LEARNING TECHNIQUES

MCA 4th Semester Dissertation Report

Submitted in partial fulfillment of the requirements
for the award of the degree of

MASTER OF COMPUTER APPLICATIONS

(MCA)

by

Jubin Saud
(Roll No. 22MCA9)
(Registration No. 22090446)

to

DEPARTMENT OF COMPUTER APPLICATION
NORTH-EASTERN HILL UNIVERSITY
Tura Campus, Tura
Meghalaya, India-794002

Under the Supervision of

Dr. Anindya Halder
Associate Professor

Department of Computer Application
North-Eastern Hill University
Tura Campus, Tura
July, 2024

पूर्वोत्तर पर्वतीय विश्वविद्यालय
कंप्यूटर अनुप्रयोग विभाग
तुरा परिसर, पश्चिम गारो हिल्स, तुरा- 794002
मेघालय (भारत)
ईमेल: hodca@nehu.ac.in

North-Eastern Hill University
Department of Computer Application
Tura Campus, West Garo Hills, Tura-794002
Meghalaya (India)
Email: hodca@nehu.ac.in

CERTIFICATE OF APPROVAL

This is to certify that the MCA 4th Semester dissertation work entitled **"DESIGN AND DEVELOPMENT OF DECISION SUPPORT SYSTEM FOR SKIN DISEASE DETECTION USING DEEP LEARNING TECHNIQUES "** carried out by **Jubin Saud** bearing Roll No: **22MCA9** and Registration No: **22090446** under the guidance of **Dr. Anindya Halder** has been found satisfactory and is approved as a project work carried out and presented in a manner required for its acceptance in partial fulfillment of the requirements for the degree of Master of Computer Applications (MCA) from North-Eastern Hill University, Tura Campus, Meghalaya.

INTERNAL EXAMINER	EXTERNAL EXAMINER
DATE:	DATE:
PLACE:	PLACE:

पूर्वोत्तर पर्वतीय विश्वविद्यालय
कंप्यूटर अनुप्रयोग विभाग
तुरा परिसर, पश्चिम गारो हिल्स, तुरा- 794002
मेघालय (भारत)
ईमेल: hodca@nehu.ac.in

North-Eastern Hill University
Department of Computer Application
Tura Campus, West Garo Hills, Tura-794002
Meghalaya (India)
Email: hodca@nehu.ac.in

CERTIFICATE

FROM

THE HEAD OF DEPARTMENT

This is to certify that the MCA 4^{th} Semester dissertation work entitled **"DESIGN AND DEVELOPMENT OF DECISION SUPPORT SYSTEM FOR SKIN DISEASE DETECTION USING DEEP LEARNING TECHNIQUES "** is submitted by **Jubin Saud** bearing Roll No: **22MCA9** and Registration No: **22090446** under the guidance of **Dr. Anindya Halder**, in partial fulfillment of the requirements for the degree of Master of Computer Applications (MCA) from North-Eastern Hill University, Meghalaya.

DATE: ____________________

PLACE: ____________________

____

Head of the department
Dept. of Computer Application
North-Eastern Hill University
Tura Campus, Meghalaya

पूर्वोत्तर पर्वतीय विश्वविद्यालय
कंप्यूटर अनुप्रयोग विभाग
तुरा परिसर, पश्चिम गारो हिल्स, तुरा- 794002
मेघालय (भारत)
ईमेल: hodca@nehu.ac.in

North-Eastern Hill University
Department of Computer Application
Tura Campus, West Garo Hills, Tura-794002
Meghalaya (India)
Email: hodca@nehu.ac.in

CERTIFICATE FROM THE SUPERVISOR

This is to certify that the MCA 4th Semester dissertation work entitled **DESIGN AND DEVELOPMENT OF DECISION SUPPORT SYSTEM FOR SKIN DISEASE DETECTION USING DEEP LEARNING TECHNIQUES** is a bonafide work carried out by **Jubin Saud** bearing Roll No: **22MCA9** and Registration No: **22090446** under my supervision and guidance. The report is found to be satisfactory for the partial fulfillment of the requirements for the degree of Master of Computer Applications (MCA) from North-Eastern Hill University Tura Campus, Meghalaya.

DATE: ____________________

PLACE: ____________________ ___________________________

Dr. Anindya Halder
Associate Professor
Dept. of Computer Application
North-Eastern Hill University
Tura Campus, Meghalaya

DECLARATION

I hereby declare that the dissertation work entitled **"DESIGN AND DEVELOPMENT OF DECISION SUPPORT SYSTEM FOR SKIN DISEASE DETECTION USING DEEP LEARNING TECHNIQUES "** submitted by **Jubin Saud** bearing Roll No: **22MCA9** and Registration No: **22090446** as a developer for the project, to the Department of Computer Application, North-Eastern Hill University, Meghalaya for the partial fulfillment of the requirements for the award of degree of Master of Computer Applications (MCA). I further declare that the dissertation work had not been submitted else where for any other degree or diploma before.

DATE: ____________________

PLACE: ____________________ ___________________________

Jubin Saud
Roll no. : 22MCA9
Reg no. : 22090446

ACKNOWLEDGEMENTS

On successful completion of my project, I would like to express my sincere gratitude to everyone who helped and supported me. However, I wish to make special mention of the following. Firstly, I am highly indebted to my guide **Dr. Anindya Halder** (Associate Professor) at Dept. of Computer Application, North-Eastern Hill University, Tura Campus, Meghalaya for the valuable guidance which has promoted my efforts in all stages of this project work. I am also thankful to **Dr. Raj Kumar Goel** (Associate Professor) and **Dr. Ansuman Kumar** (Assistant Professor) who gave me the opportunity to do this project.

My thanks and appreciation goes to my friends and staffs as well who willingly helped me without any hesitation.

Jubin Saud

Date: ....

ABSTRACT

Skin diseases' classification is one of the most essential processes in dermatology, as accurate differentiation between diseases is highly significant for proper treatment. The conventional technique of diagnosis, which depends on the work of specialists in the field of skin diseases, has certain disadvantages: it is lengthy, and it contains certain degrees of inaccuracy. This is an area of active research, and current developments in deep learning and image classification present the most effective remedies to this problem. This dissertation paper focuses on the use of deep learning classifiers for the categorization of several skin diseases through dermoscopy images. The model is trained using publicly available labelled skin disease images and is able to distinguish between the diseases on which the model has been trained on. With reference to the detailed classification of the preferred objects and the improved conception of the deep learning classifier, it is possible to mention that proposed approach is significantly more accurate and involve less time consumption in comparison with traditional methods. Besides underlining the capabilities of deep learning in medical diagnostics, this study also intends to present a feasible, general-use tool for dermatologists which doesn't only lead to improved patient care but also, positively impact the overall patient prognosis.

Keywords: skin disease; deep learning; Image classification; automation; pre-trained models; TensorFlow; CNN; medical diagnosis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

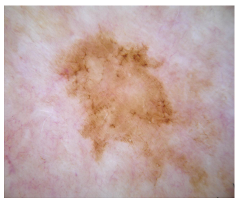
1.1 Preface

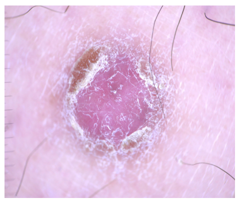
The skin is the largest and most vital organ in the human body. It absorbs vitamin D3, protects from harmful UV rays, illnesses, wounds, and temperature variations, and also prevents dehydration by holding in fluids. The skin enables us to feel sensations such as hot and cold. Skin diseases can cause rashes, inflammation, itchiness, and other abnormalities. They can be genetic or caused by lifestyle factors and impact millions of people globally. Some skin diseases can lead to potentially life-threatening problems. Due to their complexity, diversity, and similarity, skin diseases are challenging to diagnose and typically require experienced dermatologists [1].

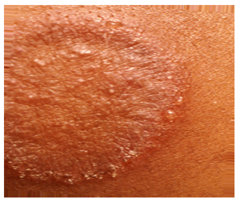
1.2 Skin Diseases and its Causes

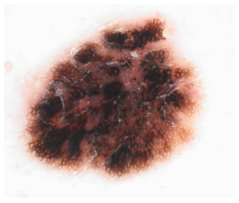
In assigning health priorities, skin diseases aren't usually one of the top concerns, and is regarded less significant compared to other illnesses such as typhoid, HIV/AIDS, community-acquired pneumonia and others [2]. However, skin diseases are generally the most common type of diseases globally affecting people of all ages, castes, nations, and beliefs. Skin diseases are conditions that affect the skin. It can range from small, minor inflammations to physical incapability or death [3]. These diseases may cause rashes, inflammation, itchiness, or other skin changes. Some skin conditions may be genetic, while lifestyle factors may cause others.

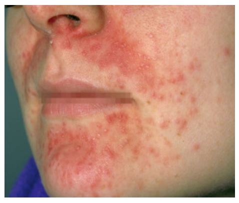
A few examples of skin diseases are shown is Figure 1.1.

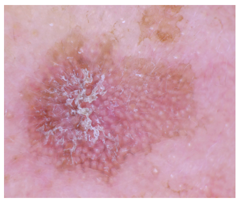

(a) Melanoma.


(b) Dermatofibroma.


(c) Tinea Ringworm Candidiasis.


(d) Melanocytic Nevus.


(e) Atopic Dermatitis.


(f) Actinic Keratosis.

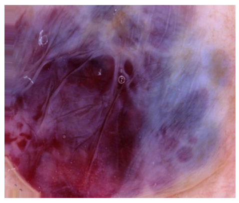

(g) Vascular Lesion.

Figure 1.1: Skin diseases.

Skin diseases are usually categorized into the following categories:

1. *Bacterial Skin Infections:* The skin bacterial infections are infections that occur when bacteria enter the skin or are present on it. It can penetrate the skin through a hair focile or through a wound. Bacterial infections can be of two types: systemic and local. Systemic infection can present symptoms all over the body like fever, while in local infection the signs and symptoms are only localized. There are infections resulting from bacteria, and these may start from one point and affect the whole body. It can be both contagious and non-contagious.

   Examples: Impetigo is one type of bacterial skin infections that may be passed from one person to another through skin contact or touch with contaminated food or water, or anything carrying the germs. Others like Cellulitis are not, meaning that they cannot be transmitted from one person to the other.

2. *Viral Skin Infections:* Viral skin infections are caused by viruses and can be contracted at any age in peoples lives. These infections are usually accompanied by skin lesions or rashes.

   Examples: Herpes simplex which causes cold sores (HSV-1) or genital herpes (HSV-2). Shingles (herpes zoster) causes reactivation of varicella-zoster virus that had been dormant in the body, which causes painful rashes.

3. *Fungal Infections:* Fungal infections usually caused by various types of fungi, including yeasts and molds. This usually thrive in warm, moist environments.

   Examples: Ringworm (tinea), it causes red circular itchy on the body. Athlete's foot, affects the feet causing itching, burning, and peeling skin between toes or on soles.

4. *Parasitic Infestations:* Parasitic infestations caused by the presence of tiny organisms making their habitat on or within the skin and manifesting different signs and symptoms leading to discomfort.

   Examples: Scabies, caused by small arthropods that are deeply embedded in the skin, with itching being the major symptom. Lice infestation, which influences the scalp, body or genital area accompanied by itchiness with evidence of eggs or nits on hair.

5. *Autoimmune Disorders:* Autoimmune disorders occurs when our body immune systems start mistakenly attacking healthy skin cells, causing various skin manifestations.

   Examples:Psoriasis which causes rapid production of skin cells resulting in thick, scaly patches. Vitiligo, another examples of this kind of disorder cause loss of skin pigmentation in patches.

6. *Hereditary dermatoses:* Hereditary dermatoses refers to skin conditions that are passed on from generation to generation through inheritance of genes.

Examples: Ichthyosis, a cluster of diseases that manifest themselves through symptoms such as skin dryness, skin thickness and scaling. Epidermolysis bullosa, results in skin breakdown, characterized by developing a rash and fungating ulcers on the skin.

7. *Allergic Reactions:* Skin allergies are conditions that stem from a person's immune response to a specific substance or elements that touch the skin.

   Examples: Contact dermatitis, a dermatitis that results from contact with a substance that produces an allergic reaction or does chemical damage to the skin. Atopic dermatitis (eczema), a long term disease that produces skin dryness, itching, and inflammation. Urticaria (hives), which causes Swollen, red, itchy lumps on the skin, commonly resulting from an allergic reaction.

8. *Inflammatory Disorders:* There are a number of types of inflammatory skin disorders that are common and can be caused by a number of different factors.

   Examples: Rosacea, it leads to skin redness, visible blood vessels, and small red spots on the face. Acne, which is caused by blocked hair follicles and results in pimples commonly known as the blackheads and the whiteheads.

9. *Neoplastic Diseases:* Some skin diseases are neoplastic diseases.These are diseases in which there's an abnormal growth of skin cells. This growth can be neoplastic benign (non-cancerous) or neoplastic malignant (cancerous).

   Examples: Melanoma, a severe type of the disease that affects the skin and develops from the pigment-producing cells. Luekemias also known as blood cancer, results in overproduction of immature blood cells, which leads to anemia, fatigue, and blood clotting problems.

10. *Environmental/Occupational Skin Diseases*: These skin conditions are proven to be fan by external stimuli such as environmental conditions or exposures at the workplace.

    Examples: Sunburn, which is caused by long-term exposure to sunlight and chemical burns, which are the effects of skin contact with chemicals which may cause skin harm.

1.3 Clinical Approach to Diagnose Skin Diseases

Diagnosing skin diseases fundamentally requires the taking of history and clinical examination of the patient, and, on occasions, further tests [4]. The initial assessment procedure involves general health history based on the start and the duration, characteristics, and change of symptoms, and any context as causative and aggravating factors [5]. Skin lesions' morphology, distribution, and color are then reviewing in physical examination. [6] For assessing potentially malignant skin lesions dermatologists often use the "ABCDE" (Asymmetry, Border irregularity, Color variation, Diameter $>$ 6mm, and Evolution) system. Other tests that can be employed for a proper assessment of skin structures include dermoscopy, Wood's lamp examination for specific infections, and pigmentary disorders, as well as skin biopsy to be followed by histopathological examination [7]. Occasionally, further investigations are indicated including blood tests including complete blood count, biochemical screen, specific allergy tests or bacterial culture to exclude other primary diseases [8].

1.4 Computational Approach to Diagnose Skin Diseases

The diagnostic methods involving skin diseases have received much attention in the last few years due to breakthroughs in machine learning and computer vision paradigms. These methods mainly incorporate deep learning approaches such as CNN in diagnosis of skin lesions by examining the images of the lesions and categorizing them into different diseases [9]. It generally starts with gathering of big data sets with dermatological images and these images are fed to the neural networks. Once trained, these models can then predict diagnostics for new, unseen images.

One of the advantages when using this approach is widely known from the study by Esteva et al. (2017) [10] that shows its capability to perform are even better compared to a human dermatologist in specific tasks. But there are still problems: For example, there is a lack of diverse and more importantly, racially and ethnically balanced databases used to test the efficacy of the models in different skin types and pathologies [11]. Other progress that has been made in the field are use cases of transfer learning where lower skin datasets pre-trained models are adopted to dermatological tasks, as well as the improving of the eXplainable Artificial Intelligence (XAI) solutions to gain information on how the decisions of the model are made [12]. Despite the promise of these computational methods, they are generally regarded as supplements to the clinicians' diagnostic ability in dermatology.

1.5 Challenges in Skin Disease Classification

While trying to classify skin diseases based on their images, the following challenges are to be faced:

1. *Intra-class Variability:* Skin conditions vary greatly in how they look, even when they fall into the same category. Factors like age, skin tone, how far the disease has progressed, and environmental factors can all play a role in this variation, making it difficult to classify skin lesions or rashes accurately.

2. *Inter-class Similarity:* Certain skin conditions may have visual similarities in color, texture, or shape, which can lead to confusion and misidentification. This likeness between different skin diseases can make it challenging to differentiate between them, making it necessary for the system to recognize even the subtlest distinctions.

3. *Limited and Imbalanced Datasets:* The datasets might not accurately represent all skin disease categories, leading to imbalances that could impact how well the classification system works.

4. *Interpret-ability and Explain-ability:* Although deep learning models have demonstrated impressive accuracy in categorizing images, their lack of transparency and interpret-ability can hinder our ability to comprehend how decisions are reached. In the realm of medical diagnosis, it is essential to have

explanations that can be trusted to promote the widespread adoption of reliable clinical practices.

1.6 Deep Learning

The deep learning (DL) is a sub-branch of the machine learning where artificial neural network is adopted with layered units successively formed to derive higher level attributes of the raw input data. In other words, deep learning can be defined as a process of training artificial neural networks that learn representations of a given data. Every layer, in a deep neural network, receives some data and changes it to a higher level and more composite form. For instance, in image recognition the first level can be the edge detection the next level could be texture recognition the next level is body part recognition which resembles the next level and so on.

1.6.1 Deep Learning Techniques

Deep Learning techniques is a broad term that refers to a range of approaches as well as neural network frameworks developed with the aim of addressing challenging problems in diverse fields. The approaches/techniques in deep learning are further discussed below.

- *Convolutional Neural Networks (CNNs):* CNNs are mostly employed in image and video based applications. They employ convolutional layers in the network which applies filters to the data to make the network to realize spatial arrangements and appearances of the images [13].

  Applications: Image recognition of different categories, identification of objects in an image, identification of specific faces in a picture, and image based diagnosis in medicine.

- *Recurrent Neural Networks (RNNs):* RNNs are special for the sequence data, such as time-series data or natural language data. Some of them have carry-over loops for the information flow from one time-step to the next, which makes them favorable for tasks that demand context from previous inputs [14].

Applications: Natural language processing, computer translation, automatic speech recognition and time series forecasting.

- *Long Short-Term Memory Networks (LSTMs):* Long Short-Term Memory Networks (LSTMs): LSTMs are another type of RNN that solves the difficulty of long-term dependencies for standard RNNs. The gates are also used to regulate the input and output data flow, since they are able to remember long-term dependencies in the data [15].

  Applications: Text to speech, generation of speech, and video analysis.

- *Transformers:* Transformers are a deep learning model that employs the self-attention mechanism for feeding the input data. They have effectively solved the long-standing problem of NLP and allows parallelization of computations and dealing with long-range dependencies [16].

  Applications: Its application consists of machine translation, text summarization, question answering and language modeling GPT-3, BERT etc.

- *Autoencoders:* Autoencoder is a kind of neural networks that can be used in unsupervised learning. Easy to understand as they are made of an encoder that encodes the input data and a decoder that decodes the compressed data. It is applied to tasks like feature extraction and classification, also it is applied in data mining, for instance identifying anomalies in huge data sets [17].

  Applications: The applications of CNNs include image denoising, feature learning and generative modeling.

- *Generative Adversarial Networks (GANs):* It is composed of two neural networks, a generator and a discriminator, that work in parallel during the training process. The generator provides fake data, and at the same time, the discriminator checks this information. Such an antagonistic process results in realistic data being generated [18].

  Applications: Generation and style synthesis, data augmentation.

- *Reinforcement Learning (RL):* RL is one of the subcategories of machine learning, where an agent interacts with an environment and learns to choose the best of sequential actions likely to yield the maximum cumulative reward. In contrast to supervised learning, where the model gets trained from a set of input-output pair data, RL is all about learning from the actions' outcomes, more often than not using an experimental approach [19].

Applications: Specific applications include game playing such as AlphaGo, robotics, self-driving cars, and recommendation systems.

These techniques are the backdrop of deep learning and work in different manners to solve different problems and handle data. The elaboration and improvement of these models remain key to enhancing the trends in bringing about artificial intelligence as a mechanism that endows machines the ability to perform tasks more and more accurately.

1.6.2 Deep Transfer Learning

Deep Transfer Learning refers to a method employed in machine learning that entails reusing an existing model for a given assignment to act as a foundation upon which a new model will be developed to address a different but somehow related assignment. In their effort to improve performance and reduce training time, this method allows one field to benefit from what is known in another field. Trained models that are usually general e.g., ImageNet helps in reducing the need for too many labels through the use of transfer learning from large datasets like ImageNet.

Deep transfer learning, in practice, typically involves feature extraction or fine-tuning. Feature extraction utilizes a pre-trained model to derive salient characteristics from new input, which can then be used to train another classifier. In fine-tuning, a pre-trained model is picked up, and its weights updated using the fresh dataset. This can happen by unfreezing certain layers and then training them at a smaller learning rate.

1.6.3 Deep Learning for Image Classification

Image Classification is a critical task in computer vision, where the goal is to assign a label or a category to an input image. Deep Learning, particularly Convolutional Neural Networks(CNN) has revolutionized this field, achieving state-of-the art performance across various image classification benchmarks.

Convolutional Neural Networks (CNNs) are the mainstay of many modern image classification systems. They are made of a combination of various layers namely, convolutional layers, pooling layers and fully connected layers [20]. Transfer learning which utilizes pre-trained models on massive datasets like ImageNet as starting points that can be customized for particular tasks is often used in order to improve their performance while at the same time reducing the amount of time taken during training [21].

The task of image classification generally consists of the following steps:

- *Data Collection and Preparation:* Get a diverse dataset of labeled images that represent the categories you expect to classify. It could be achieved by collecting your own data or using pre-existing datasets such as ImageNet or CIFAR-10 [22].

- *Data Preprocessing:* Make images the same sizes all through by normalizing pixel values as well as possibly resizing them and applying techniques such as rotation, flipping, or color jittering to increase dataset diversity [23].

- *Data Augmentation (optional):* Data augmentation is a technique used to artificially expand the training dataset by applying various transformations to existing images. Common augmentation tools include rotation, flipping, scaling, cropping, and adjusting brightness or contrast. This way during training the model will also be exposed to more diverse types of image variations, hence better generalization as well as robustness are realized through it by exposing the network to a broader spectrum of image variations during training.

- *Data Splitting:* When working with data it is best practice to split them into sections such as training data, validation data, etc. An optimal distribution often includes allocating 70-80% towards one part (training set), 10-15% another (validation set), and the rest to testing set.

- *Model Selection:* Use a suitable convolutional neural network architecture for your work, such as ResNet or EfficientNet, among others

- *Transfer Learning (Optional):* If you decide to utilize transfer learning, import a pretrained model and either deploy it as a feature extractor or adapt it for your particular job [24].

- *Model Configuration:* Establishing the model construction, which includes the number and types of layers, activation function and output layer which matches the number of classes we want our data to classify into.

- *Training:* Data should be taken through the network so that it can be learned from, then loss calculated with a suitable loss function such as cross entropy before updating model parameters through optimization techniques such as SGD or Adam [25].

- *Validation:* The model's performance in the validation set should be evaluated often, so that one can see if it is over-fitting or needs tuning of various hyperparameters.

- *Testing:* Carrying out a test on the model trained, and then we will be able to get an honest estimate of the model's performance.

- *Hyperparameter Tuning:* Fine-tune the hyperparameters of the algorithm such as learning rate, batch size, regularization method etc. This will help you improve model performance [26].

AlexNet, VGGNet, ResNet, Inception (GoogLeNet), DenseNet, EfficientNet, and Vision Transformer (ViT) are well-known architectures used in image classification. All of these have made important steps in the field of image classification.

The challenges and future directions in image classification using deep learning include enhancing model efficiency as well as decreasing computational requirements; rectifying bias and fairness issues in image classification systems; creating more tolerant models against domain shifting and adversarial attacks; and examining self-supervised and few-shot learning approaches.

1.7 Related Work on Deep Learning

Zeon et al. developed DOCAID [27], a disease prediction system for dengue, cholera, malaria, Yellow fever, and diarrhea using the Naive Bayes Classifier algorithm. Ra et al. [28] used segmentation and classification techniques to identify skin lesion regions. Wu et al. [29] used a convolutional neural network for face-related skin disease identification.

CNNs are Deep Learning algorithms that are used to recognize and classify visual features from input data such as road signs, building sites, and human faces [30]. CNN trains on a bunch of images based on architecture's parameters. The authors of [31] used a range of data feature extraction methods, such as Multi-layer Perceptron (MLP), Random Forest, Naive Bayes, and decision trees, to forecast Erythemato-Squamous Disease (ESD). With a classification accuracy of 97.4% (backslash percent), the Naive Bayes classifier performed the best in this scenario.

Skin disorders are also categorized using necessary image processing techniques, such as morphological operations for skin detection [32] [33]. The binary pattern created by threshold-holding is crucial to pathological raising, closure, distortion, and erosion; so, the optimal threshold value needs to be chosen very carefully. The development of the damaged region may not be predicted by morphologically based methods due to the roughness of the image.

A system for categorizing skin conditions was created by the Genetic Algorithm (GA) [34] [35]. Some disadvantages of the Genetic Algorithm include its protracted convergence time to a solution [36]. If the model does not yield a fair conclusion, it rarely provides the global best answer [37].

Skin disease categorization using ensemble models [38] yields more accurate findings by combining many prediction models. Ensemble models tend to overfit and perform poorly when there are unidentified imbalances in the sample data [39]. It has been demonstrated that classifying skin diseases using a deep learning model [40] [41] is successful. However, experiments have shown that the model is insufficient for multi-lesion images. More processing work is needed because deep learning models need a lot of training to reach a reasonable degree of accuracy.

1.8 Motivation of the Project

The motivation of the proposed project work are as follows:

1. *Increased Diagnostic Accuracy:* Using deep learning and computer vision techniques, the proposed system aims to accurately identify and classify skin diseases and maintain a consistent prediction history.

2. *Accessibility:* An automated skin disease detection system can be deployed on various platforms, including mobile phones, making it more accessible to a boarder population, especially in rural areas that lack proper healthcare services.

3. *Faster Results:* Based on provided images, the system aims to give quick and accurate diagnostics of skin diseases, which can facilitate timely intervention and treatment, potentially improving patient outcomes and preventing complications occurring due to delayed diagnosis.

4. *Reduced Subjectivity and Consistent Diagnosis:* Automated systems offer an unbiased approach to diagnosis, minimizing variations caused by biases and experience levels. This leads to treatment protocols being implemented effectively.

5. *Scalability and Efficiency:* The rising need, for healthcare services and the increasing occurrence of skin conditions have led to the development of an automated system that offers an effective way to screen and diagnose groups of people, easing the workload, on healthcare providers.

6. *Cost Reduction:* Automated systems have the potential to lower healthcare costs for diagnosing and treating skin diseases by simplifying the process and reducing the requirement for specialized personnel in some instances.

1.9 Organization of the Report

The organization of the rest of the report is as follows:

Chapter 2 describes the methodology of image classification of skin diseases. It covers dataset preparation, architectural details of the DL Image Classifiers and training procedures.

Chapter 3 provides the results from our experiments. The evaluation of the models are shown using the four metrics namely accuracy, precision, recall, and F1-measure. Accuracy curve and loss cure of the models are also shown in this chapter. It also contains a confusion matrix and a comparative study to determine the best performing model.

Chapter 4, we discuss the conclusion of this report. It also identifies areas for future research and improvement.

Chapter 2

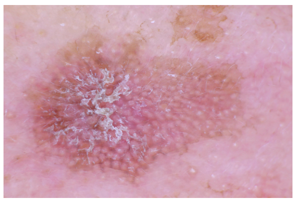
Materials and Methods

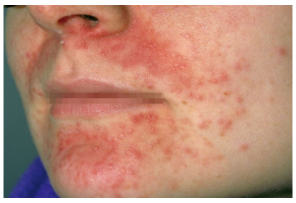
2.1 Introduction

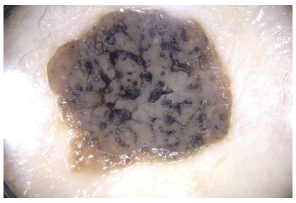
In this section, we discuss how deep learning approaches can be used to classify images. Image classification in deep learning is done based on some features such as edge, color, pattern, texture etc.

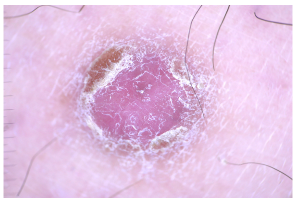
2.2 Skin Disease Dataset Used

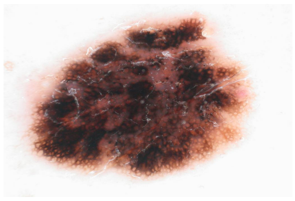
The publicly available dataset from [42] is used to carry out experiments for this report. The dataset consists of 696 sample images across 9 classes namely, Actinic Keratosis (2.1a), Atopic Dermatitis (2.1b), Benign Keratosis (2.1c), Dermatofibroma (2.1d), Melanocytic Nevus (2.1e), Melanoma (2.1f), Squamous Cell Carcinoma (2.1g), Tinea Ringworm Candidiasis (2.1h), Vascular Lesion (2.1i). Sample images from all the classes present in this dataset are shown in Figure 2.1.

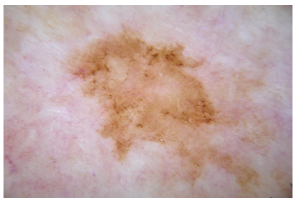

(a) Actinic Keratosis.

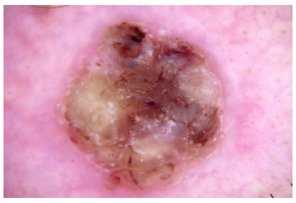

(b) Atopic Dermatitis.

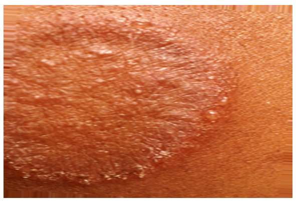

(c) Benign Keratosis.


(d) Dermatofibroma.


(e) Melanocytic Nevus.


(f) Melanoma.


(g) Squamous Cell Carcinoma.


(h) Tinea Ringworm Candidiasis.

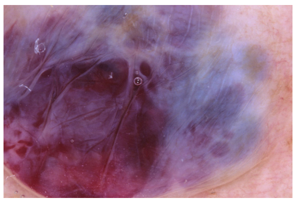

(i) Vascular Lesion.

Figure 2.1: Sample images from each class of skin disease classification dataset.

2.3 Methods

The overall methodology for skin disease classification is depicted in Figure 2.2.

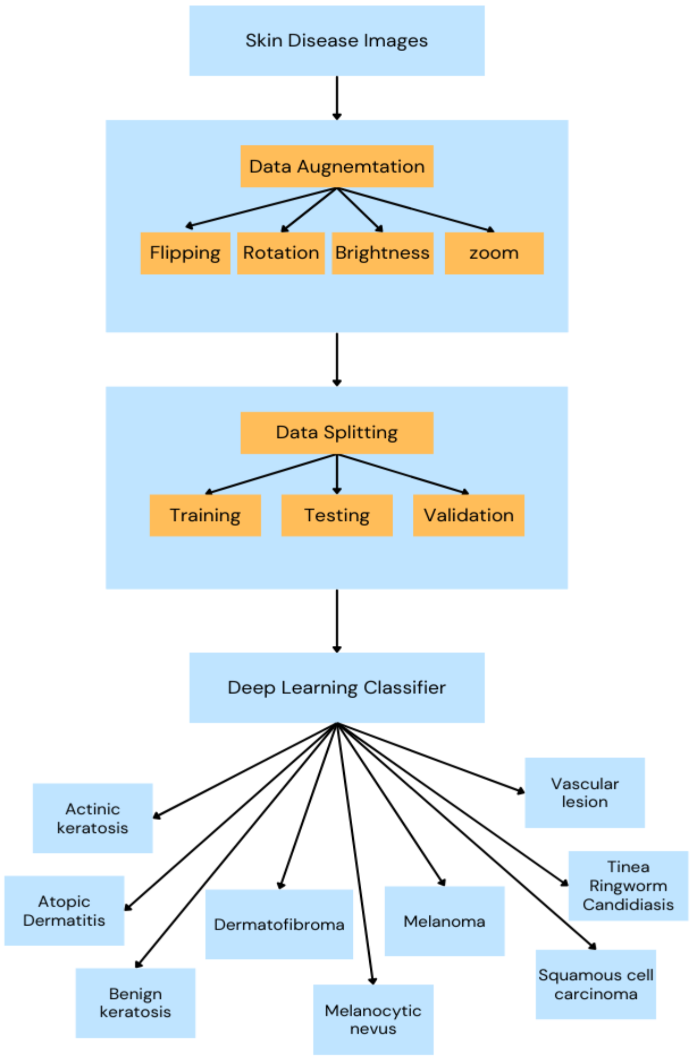


Figure 2.2: Block diagram of the methodology.

2.3.1 Data Preprocessing

Before passing our input image data to the deep learning classifiers, we need to take some preprocessing steps. Effective data preprocessing is crucial for achieving high performance in image classification tasks.

Necessary steps involved in data preprocessing are given below.

- *Image Resizing*: Resize all image to uniform size matching the size that is required as an input by the deep learning model.
- *Numeric conversion:* The neural network cannot directly take image as input, we need to convert the images into numeric values.Converting images

into arrays of pixel values. Python libraries such as OpenCV, PIL, NumPy and TensorFlow/Keras can be used to convert images to their numeric representation.

- *Normalization:* Adjusting the pixel values, usually in the range of [0.0, 1.0] or [-1.0, 1.0]. This can be done by dividing the pixel values by 255 (if the original range was 0-255) or normalization through mean and standard deviation of the dataset.
- *Data Augmentation:* Some techniques that should be applied towards the data set include the rotation, the flipping, the brightness, the zooming and searing in order to increase the size of training set artificially.
- *Shuffling:* Properly randomize the dataset to get an input of varied arrangements for the model while training, which is good for generalization.
- *Splitting:* Dividing the dataset into training, testing, and validation. The division is usually 70% to train, 15% to validate, and the remaining 15% to test.

2.3.2 Data Augmentation

Data Augmentation is a method applied to enhance the variety of training data without necessarily obtaining new samples. For image classification, it is used in producing new images which have been altered in some way without the need of importing new sample images. This means that in an attempt to feed the model with less training data, data augmentation enhances the generalization and robustness of models. It helps prevent overfitting, improves accuracy on less data, and results in better ability to deal with real life shifts. Thus, it allows improving the accuracy of the resulting models of machine learning and achieving their higher reliability without the need for new data acquisitions.

Augmentation is used on the selected dataset to increase the number of sample images per class. Initially, the skin disease classification dataset comprised 696 sample images. However, with the aid of data augmentation, the number of samples per class can be significantly increased. The new augmented dataset can be found at http://www.kaggle.com/datasets/jubinsaud/balanced-skin-disease-image-classification-dataset.

Image sample count per class, before and after data augmentation, is shown in Table 2.1.

Table 2.1: Sample count per class before and after augmentation.

Class	**Before augmentation**	**After Augmentation**
Actinic Keratosis	80	1137
Atopic Dermatitis	81	1136
Benign Keratosis	80	1143
Dermatofibroma	80	1142
Melanocytic Nevus	80	1129
Melanoma	80	1125
Squamous Cell Carcinoma	56	1132
Tinea Ringworm Candidiasis	80	810
Vascular Lesion	80	1134

Total sample count before augmentation is 696 images and after augmentation it consists of 9888 images.

Augmentations applied on the original dataset are shown in Figure 2.3.

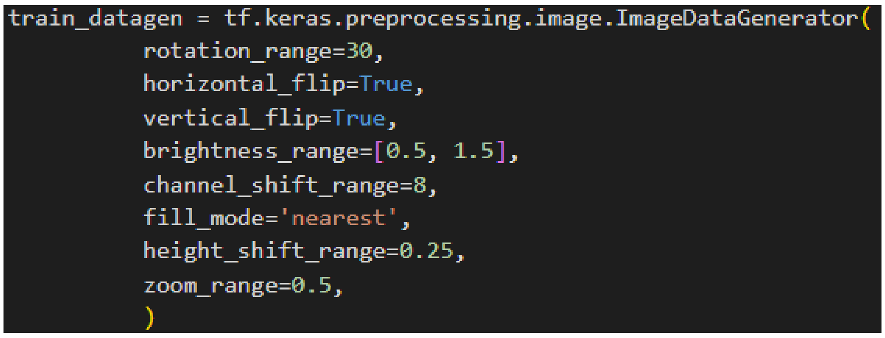

```
train_datagen = tf.keras.preprocessing.image.ImageDataGenerator(
          rotation_range=30,
          horizontal_flip=True,
          vertical_flip=True,
          brightness_range=[0.5, 1.5],
          channel_shift_range=8,
          fill_mode='nearest',
          height_shift_range=0.25,
          zoom_range=0.5,
          )
```

Figure 2.3: Augmentations applied.

2.3.3 Data Splitting

Separating your data into training, validation and testing datasets is a basic yet critical process of model building and evaluation in deep learning. This helps to make sure that, while the model is trained with one subgroup of the data, it is tested on other, different subgroups to see how well it predicts and how broadly applicable it is.

$$\begin{aligned}trainData, dummyData = train\_test\_split(&df, train\_size = 0.8,\\ &shuffle = True, random\_state = 42,\\ &stratify = strat)\end{aligned} \tag{2.1}$$

The line of code shown in Equation 2.1 is used to split the actual data store in "df". Since the *train_size* here is 0.8, so the data split is 80% for trainData and the rest 20% for dummyData. After this, the next line of code is executed.

$$\begin{aligned}validData, testData = train\_test\_split(&dummyData, train\_size = 0.5,\\ &shuffle = True, random\_state = 42,\\ &stratify = strat)\end{aligned} \tag{2.2}$$

The line of code shown in Equation 2.2 is used to split the data in dummyData into validData and testData. Since the *train_size* is 0.5, it indicates that dummyData has been equally split between validData and testData.

random_state controls the shuffling applied to data before splitting. *stratify* ensures that the class distribution in the training and testing sets is the same as in the dataset.

For this project work, the division of the dataset is 80% is used for training, 10% for validation and the remaining 10% for testing. After data splitting the trainData consists of 7910 images, validData consists of 989 images and testData consists of 989 images.

2.3.4 Deep Learning Image Classifiers

Neural network classifiers are complex artificial neural networks developed specifically to classify the input data into certain categories. These powerful models have an input layer, one or more hidden layers, and an output layer with several 'nodes' or 'neurons' in each layer. Convolutional neural network-based image classifiers have brought tremendous changes to the field of computer vision, since they make detailed identification and classification possible. These classifiers are multilayered, the layers aggregate sophisticated features, and they are not easily trained and demand large datasets and processing power. They are employed in areas such as medical diagnosis, self-driving cars, and even facial identification. However, they may be sensitive to training data quality and are known to easily overfit the data if the regularization techniques are not well implemented. This has greatly improved the performance of machines in comprehending the nature and content of the images.

2.4 Architectural Details of the Deep Learning Image Classifiers

There are sophisticated artificial neural networks known as deep learning classifiers that are capable of finding the features and patterns needed for classification of input data this is done out of a large database and are capable of sorting the input data into desired categories. They commonly have more than one level, hence the term "deep" allowing them to solve more complex classification problems across the broad fields of applications such as image, text, and speech recognition. For the task of classifying skin diseases, 7 different pre-trained deep learning classifiers viz., EfficientNetB7, EfficentNetB6, ResNet101, MobileNetV2, DenseNet121, VGG19. Out of which, the best performing classifier will be used to classify skin diseases in our mobile application.

2.4.1 EfficientNetB7

EfficientNetB7 is the biggest model in the EfficientNet family, which are the series of CNN developed by Tan and Le in the year 2019 [43]. To counter the issue of scaling, the EfficientNet series was realized with an aim of improving the efficiency of neural networks.

Compound scaling is the core component of EfficientNetB7, and this allowed the usage of much larger models. This method makes all aspects of width, depth, and the sizes of network layers proportional with the help of compound coefficient. Width scaling means to increase the number of channels in layers. Depth scaling means to increase the number of layers. Resolution scaling is to increase the input image resolution. Accordingly, Tan and Le [43] suggested that with an input image, the network has to possess more layers to widen its receptive field and more channels to capture rather detailed features.

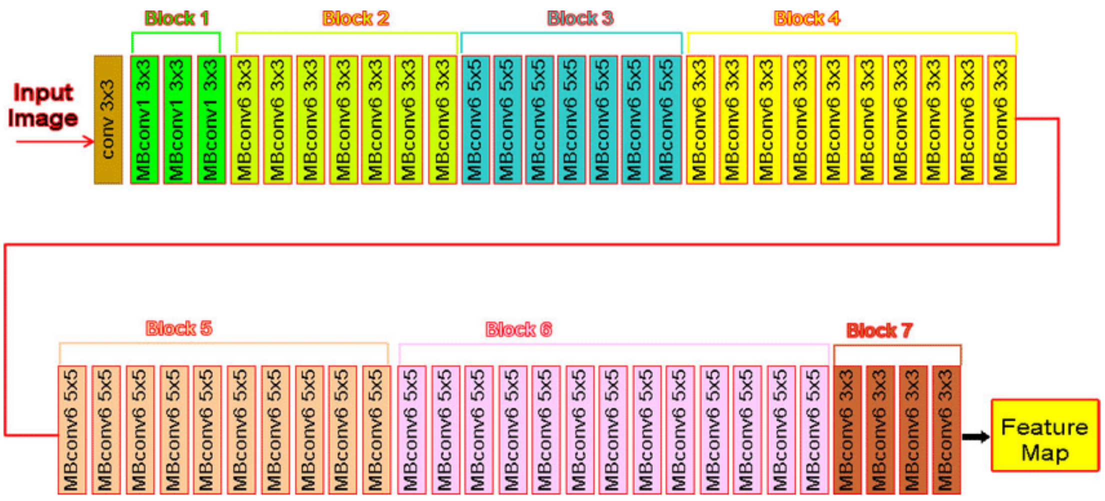


Figure 2.4: EfficientNetB7 architecture.

Figure 2.4 illustrates the EfficientNetB7 architecture. It is a deep convolutional neural network that is intended for image classification. The various components of the architecture are further discussed below:

1. *Input Image:* The image is inserted into the network after necessary data-preprocessing has been completed.

2. *Initial Convolution Layer:* The first layer to follow is the basic convolution layer, based on a filter of 3×3. It contains the implementation of the basic structure of the network that takes an input image and produces a feature map.

3. *Blocks:* The primary component of EfficientNetB7 is made out of numerous blocks (Block 1 to Block 7) comprising different MBConv stages.

   MBConv layers represent their layers as Mobile Inverted Bottleneck Convolution layers. There are one or more MBConv layers within each block, and the kernel sizes of these layers may be different within the same block.

- Block 1: Corresponds to different variants of MBConv with one of them being MBConv1 3×3.
- Block 2: Has MBConv6 3×3 layers.
- Block 3: Has MBConv6 5×5 layers.
- Block 4: Includes descending MBConv6 3×3 layers.
- Block 5: Composed of MBConv6 5×5 layers.
- Block 6: It has MBConv6 layers with 5×5 kernel size.
- Block 7: Had MBConv6 3×3 layers in some of its parts.

The figures next to MBConv (for instance, MBConv1, MBConv6) stand for the expansion ratio, which defines the number of channels expanded in the bottleneck layer. And the numbers after MBConv6 (for instance 3×3, 5×5) is the size of the convolutional kernel.

4. *Transition between blocks:* Whereas the input feature maps for each block are reformulated, the output feature maps act as the input to the next block. These transitions might include downsampling, which decreases the spatial dimensions of the feature maps while increasing the depth.

5. *Global Feature Map:* Following the progression of all the blocks, the final feature map is produced as it demonstrates prominence and abstract features from the input image.

6. *Classifier:* Subsequently, the feature map is followed by a global average pooling layer, in which case the spatial dimensions reduce to a single value per feature map. After this, it is succeeded by a fully connected (dense) layer or multiple dense layers that give the final classification probability of each class.

7. *Output:* The last layer which results is a Feature Map that can be used for classification The Features of the training set and the Features of the testing set will be the same for this type of Networks.

EfficientNetB7 has a depth of 55, width multiplier is 2.4 and a resolution of 600×600.

2.4.2 EfficientNetB6

EfficientNetB6 is a convolutional neural network for the classification of images belonging to the range of models($B0 - B7$) of the EfficientNet family [43]. It is one of the mini and mid-size models in a series of models, starting from B0 and ending with B7, mirroring the degree of increasing model size.

EfficientNetB6 is similar to EfficientNetB7, the key difference between them is that EfficientNetB7 is even bigger and its computational complexity is higher as well as the number of parameters. While B7 can achieve higher accuracy, B7 provides a better trade-off between performance and efficiency.

EfficientNetB6 has a depth of 50, width multiplier is 2.2 and a resolution of 528×528.

2.4.3 ResNet101

ResNet101, also knows as Residual Network-101, is a deep learning model which belongs to convolutional neural network with its application extending to image classification and object detection among other computer vision problems. It was proposed by Kaiming He et al. in the paper called "Deep Residual Learning for Image Recognition" in 2015 [44].

ResNet101 contains 101 layers, with a basic building block containing two 3x3 convolutional layers and a shortcut connection. This design enables it to obtain a very high accuracy on different tests than earlier networks such as VGG and Inception.

Figure 2.5 illustrates the architecture of Resnet101. Detailed discussion of various components is given below:

- *Input:* The first layer that is usually formed to accept the given data feed.

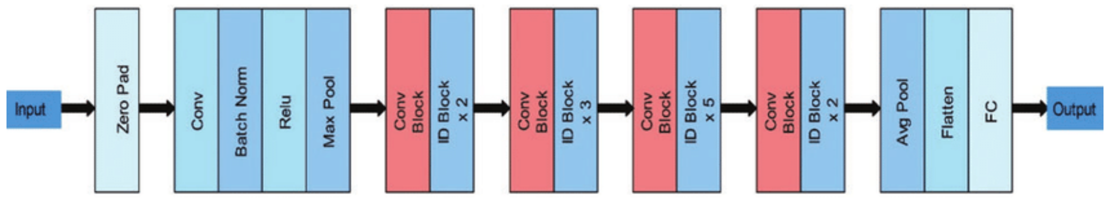


Figure 2.5: ResNet101 architecture.

- *Zero Pad:* This layer adds extra dimensions of zero to the data fed into it, typically to retain spatial configurations after convolutions.
- *Conv:* A layer in a CNN that convolves some input data in order to produce a new set of values.
- *Batch Norm:* Activates the BN layer of batch normalization to normalize the output of the convolution layer.
- *ReLU:* Used the ReLU(Rectified Linear Unit) activation function to bring in non-linearity into the network.
- *Max Pool:* A max-pooling layer of a CNN that will reduce the size of the feature maps by taking the maximum value over a window.
- *Conv Block:* A convolutional block that usually has one or more convolutional layers, batch normalization, and/or ReLU activation layers.
- *ID Block:* This is normally followed by the identity block, these being skip connections that directly add the input to the output of the specific convolutional block as a form of residual learning.
- *Conv Block x2:* Two layers of convolution carried out one after the other.
- *Conv Block x3:* Three BLOCK IN SEQUENCE 3D convolutional.
- *Conv Block x5:* These are five successive pairs of convolutional layers which contain convolutional layers and ReLU nonlinearities and pooling.
- *Conv Block x2:* Two more consecutive convolutional blocks, and why this is a good idea we should discuss in the next section.
- *Avg Pool:* An Average pooling layer that cuts the spatial dimension by coming up with an average value of a window.

- *Flatten:* A flatten layer that reforms an output of the pool layer to a vector for the next layer to process.
- *FC (Fully Connected):* A layer that provides a complete connection between the input neuron and all the neurons in the output layer, which is generally used in the course of classification.
- *Output:* The output layer that is the last layer of the network and gives the classification output.

Figure 2.2 image shows a common way of data passing through the ResNet101 network starting from the input and passing through convolutional layers, batch normalization, ReLU activation, pooling, and ending at fully connected layer that gives the classification result. This paper establishes that the identity and convolutional blocks with skip connections are the main architectural features of ResNet that aid in learning residual and enhancing the gradient flow during the training phase.

2.4.4 MobileNetV2

To support the mobile and embedded vision systems, an efficient network model, MobileNetV2 [45], is developed as a convolutional neural network (CNN). They derived it from the previous MobileNet model, which was also created by Google researchers. What is most interesting about this is model is the tune it manages to hit between model complexity and error rate makes it ideal for resource-scarce devices.

MobileNetV2 architecture consists of convolution layer, depthwise separable convolutions, inverted residuals, bottleneck design, linear bottlenecks, and squeeze-and-excitation (SE) blocks. These components assist to decrease the number of parameters and calculations without compromising the identification of intricate characteristics.

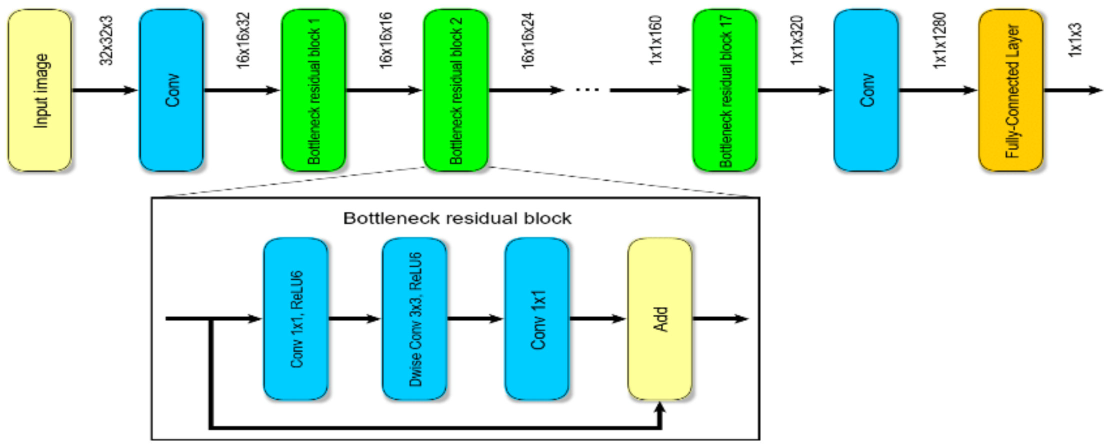


Figure 2.6: MobileNetV2 architecture.

Figure 2.6 illustrates the architecture of MobileNetV2. The components of the architecture are discussed below:

- *Depthwise Separable Convolution:* Depthwise separable convolution is a technique used in MobileNetV2 to reduce the computational cost of convolutions. It separates the standard convolution into two separate operations: depthwise convolution and pointwise convolution. This separation significantly reduces the number of computations required, making the model more efficient.

- *Inverted Residuals:* Inverted residuals are a key component of MobilenetV2 architecture that helps improve the model's accuracy. They introduce a bottleneck structure that expands the number of channels before applying depthwise separable convolutions. This expansion allows the model to capture more complex features and enhance its representation power.

- *Bottleneck Design:* The bottleneck design in MobileNetV2 further reduces the computational cost by using 1×1 convolutions to reduce the number of channels before applying depthwise separable convolutions. This design choice helps maintain a good balance between model size and accuracy.

- *Linear Bottlenecks:* MobileNetV2 proposes linear bottlenecks to replace the previous non-linear ones to help not lose information during the shaping of bottlenecks. Linearity is applied to avoid distorting the data by using non-linear activations which help the model to be more accurate, and capture the inputs' minor differences.

- *Squeeze-and-Excitation (SE) Blocks:* To improve the feature representation ability, the first SE blocks are introduced in this paper to attach to the MobileNetV2 model. These blocks are able to dynamically adjust the channel-wise feature responses for the model to learn important features and disregard less important ones.

2.4.5 DenseNet169

DenseNet169 belongs to the family of DenseNet in which each layer is connected with every other layer in a feed-forward network. This connectivity pattern contrasts with the convolutional neural networks (CNNs) where each output layer maps generate an input feature map for the next layer in line. DenseNet169, as the name suggests, contains 169 layers.

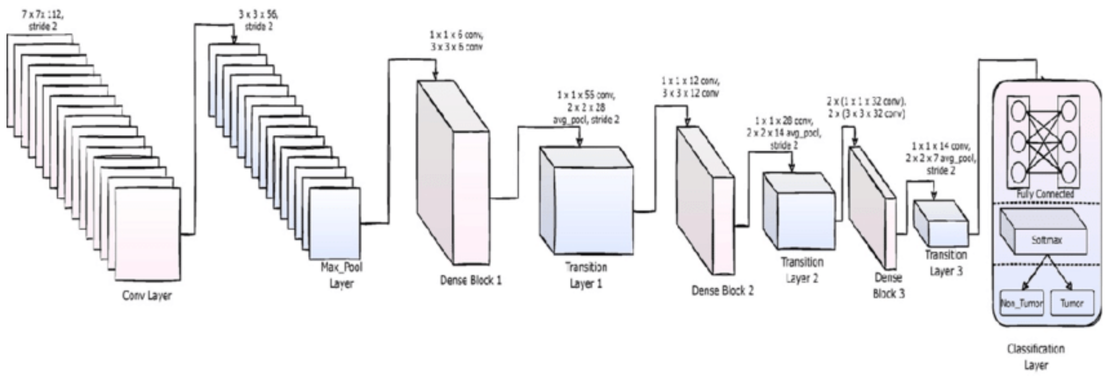

7 x 7x 112, stride 2
Conv Layer
3 x 3 x 56, stride 2
Max_Pool Layer
1 x 1 x 6 conv, 3 x 3 x 6 conv
Dense Block 1
1 x 1 x 56 conv, 2 x 2 x 28 avg_pool, stride 2
Transition Layer 1
1 x 1 x 12 conv, 3 x 3 x 12 conv
Dense Block 2
1 x 1 x 28 conv, 2 x 2 x 14 avg_pool, stride 2
Transition Layer 2
2 x (1 x 1 x 32 conv), 2 x (3 x 3 x 32 conv)
Dense Block 3
1 x 1 x 14 conv, 2 x 2 x 7 avg_pool, stride 2
Transition Layer 3
Fully Connected
Softmax
Non_Tumor
Tumor
Classification Layer

Figure 2.7: DenseNet architecture.

Figure 2.7 illustrates the architecture of DenseNet.

The architecture of DenseNet169 can be described in detail as follows:

- *Input Layer:* It accepts input images of size typically 224x224 pixels with three color channels, of RGB.

- *Initial Convolutional Layer:* Separate convolution with 7×7 kernel being used, followed by the batch normalization and rectified linear units' activation. This layer convolves the input images and produces the first set of features called feature maps.

- *Dense Blocks:* In DenseNet169 we have 4 dense blocks which are further combined with several dense layers. Each dense block is structured as follows:Each dense block is structured as follows:

- *Dense Layers:* The next one is a set of convolutional layers – usually containing 3x3 kernels – that are followed by batch normalization and ReLU activation. All layers in a given dense block are fully connected and pass input to each other layer in the block.

- *Bottleneck Layers:* This is used in many networks to reduce the number of channels fed into the next operation for efficiency, this layer makes use of 1x1 convolution followed by 3×3 convolution.

- *Transition Layers:* Between each pair of dense blocks there are transition layers implemented by average pooling for reducing the spatial dimensions and 1x1 convolution for reducing the number of feature maps. This is useful in mitigating the model complexity and the computational expenses.

- *Global Average Pooling:* The final dense block is followed by GlobalAveragePooling layer, in which the spatial dimensions of the feature maps for each channel are averaged to obtain a fixed sized feature vector.

- *Fully Connected Layer:* The next layer is denser and is followed by the softmax activation for the final classification. This layer gives the likelihood of the dataset for all the classes that are in it.

From the parameter's standpoint, DenseNet169 contains about 14.3 million parameters and, thus, can be considered relatively non-verbose when compared with other deep architectures like ResNet-101.

2.4.6 VGG19

VGG19 is a CNN architecture which is part of the VGG models family of architectures. It was introduced in the paper "Very Deep Convolutional Networks for Large-Scale Image Recognition" by Simonyan and Zisserman [46] paper in 2014 build from the previous model called VGG16 but with an advanced depth design.

VGG19 is made up of 16 Convolutional layers arranged into Convolutional layers, 3 Fully connected layers where 5 layers of Max pooling are used for down sampling.

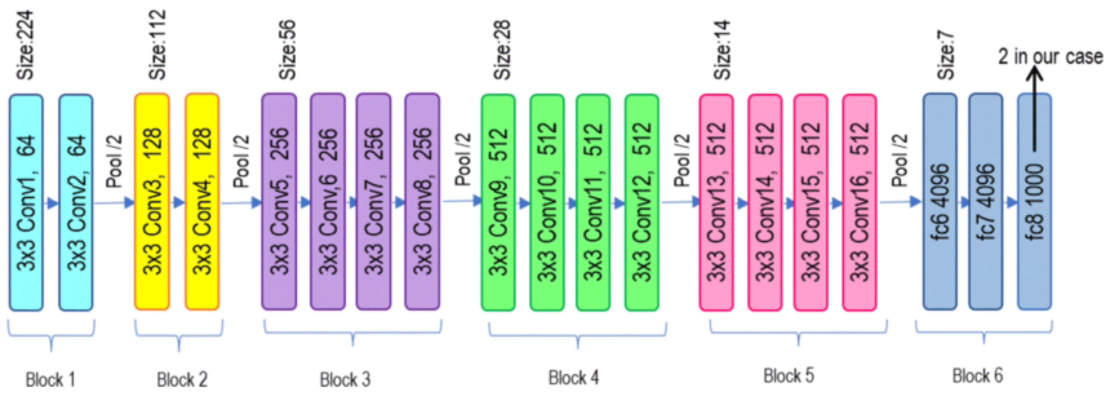


Figure 2.8: VGG19 architecture.

Figure 2.8 illustrates the architecture of VGG19. The components are discussed below:

- *Input Layer:* They feed forward the input images which are normally of size 224 x224 with colors in three channel band (RGB).
- *Convolutional Layers:* The convolutional layers present are 3×3 filters and a stride size of one pixel, this means that after convolution, the result is padded giving it the same size as the original input.
- *Convolutional Blocks:* The architecture is divided into blocks in which multiple convolutional layers are employed, after which the data is passed through the max pooling layer that performs the spatial resolution reduction.
- *Convolutional Layers:* The second layer known as convolution layer has sequential 3x3 convolutional layers with ReLU activation functions.
- *Max Pooling:* max pooling of the type ‘2×2’ with the stride of ‘2 pixels’ which effectively decreases the size of feature maps to half.
- *Fully Connected Layers:* Similar to other convolutional based networks, VGG19 has 3 fully connected layers after those convolutional blocks.

- *Flattening:* To be passed to the fully connected layers, the output of the last convolutional block is most often flattened into a vector.
- *Fully Connected (Dense) Layers:* The first three layers which are all fully connected layers are followed by a ReLU activation function and the final output layer.
- *Output Layer:* The last but not the least, we have a softmax activation layer that gives the final probability for each class present in the dataset.

2.5 Summary

In this chapter, we discussed related work, we also discussed the deep learning classifiers which we will be using in our project to classify skin diseases.

Chapter 3

Experimental Evaluation, Result, and Analysis

3.1 Introduction

In this chapter, the methods of deep learning for the classification of skin diseases are discussed. To start with, this paper presents brief background on the experiments involving the given dataset, preprocessing of data, and the choice of deep learning models. We then proceed with the description of the results of the experiments, as well as of the operating environment and hyper-parameter settings. In the last section, we present the results and a comparative study. By means of such an extensive analysis, we strive to provide an idea of how deep learning can contribute to skin disease diagnosis, improving both the precision and the openness of the procedure and also to find the best performing model for the task of image classification.

3.2 Experimental Evaluation

For classifying an image into a class, using a deep learning classifier model, the model first needs to be trained on labeled data of that class. After training, validating, and testing the model's performance, we can determine its usability, reliability, and efficiency. Various experimental evaluation matrices exist such as accuracy, precision, recall, and f1-score. Below, we will discuss the dataset used, and the evaluation matrices recorded after experimenting with various classifier models.

3.2.1 Evaluation Measures

A major practical aspect of model assessment is a set of quantitative performance criteria including Accuracy, Precision, Recall, and F1-score.

A brief introduction of these evaluation matrices for multi-class classification task is given below.

- *Accuracy:* The percentage of true positive instances and the true negative instances out of the total positive and negative instances. It is the general measure which shows the troublesomeness of this phenomenon on the whole. The mathematical equation for finding accuracy is shown in Equation 3.1.

$$\text{Accuracy} = \frac{\sum_{i=1}^{N} TP_i}{\sum_{i=1}^{N}(TP_i + FP_i + FN_i + TN_i)} \times 100 \tag{3.1}$$

- *Precision:* The ratio of the number of observations grouped under positive by the model and classified correctly to all the observations classified as positive by the model. It is a special type of metric that quantifies the accuracy or quality of the model in relation to the data you are analyzing. The mathematical equation for finding precision is shown in Equation 3.2.

$$\text{Precision}_i = \frac{TP_i}{TP_i + FP_i} \tag{3.2}$$

- *Recall (Sensitivity or True Positive Rate):* The ratio of actual positive prediction to all prediction within the actual class of a variable. Completeness is very important, and it is a measure of completion or when a certain job is fully done. The mathematical equation for finding recall is shown in Equation 3.3.

$$\text{Recall}_i = \frac{TP_i}{TP_i + FN_i} \tag{3.3}$$

- *F1-score:* An average of Precision divided by the average of Recall. It handles both the metrics together and is most advantageous when the class distribution is skewed. The mathematical equation for finding f1-score is shown in Equation 3.4.

$$\text{F1-Score}_i = 2 \cdot \frac{\text{Precision}_i \cdot \text{Recall}_i}{\text{Precision}_i + \text{Recall}_i} \tag{3.4}$$

- *Support:* It refers to the number of occurrences of each class in our dataset. The mathematical equation for finding support is shown in Equation 3.5.

$$\text{Support}_i = TP_i + FN_i \tag{3.5}$$

Where:

- TP_i (True Positive): Correctly identified positive samples for class i.
- TN_i (True Negative): Correctly identified negative samples for class i.
- FP_i (False Positive): Negative samples incorrectly identified as positive for class i.
- FN_i (False Negative): Positive samples incorrectly identified as negative for class i.

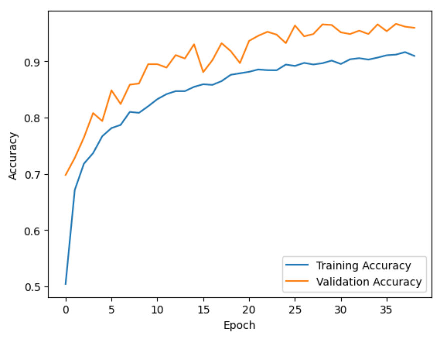
3.2.2 Accuracy Curve and Loss Curve

Accuracy curves and the loss curves are two basic plots that are used to assess and plot the performance of any model during the training phase.

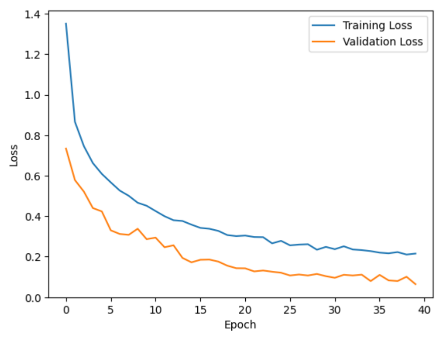
The accuracy curve depicts the overall rate of successful machine's predictions. It normally scales between 0 and 1 (or between 0% and 100%), with one being the best. Since it is customary to present both a training accuracy and a validation accuracy, the graph presented in the image is examples of accuracy curves where the accuracy grows with the training progress.

In contrast, the loss curve indicates the error or the difference of the model from the actual values. Examples of the loss functions include mean squared error for regression problems, and cross-entropy for classification ones. However, with loss, lower numbers are better because it quantifies the size of the error that is visible between the original input and the model's output. In the context of the given image, although the curves for loss are not included, it is customary to plot accuracy and loss together.

Examples of accuracy curve 3.1a and loss curve 3.1b are shown is Figure 3.2.



(a) Accuracy curve.



(b) Loss curve.

Figure 3.1: Example of Accuracy curve and Loss curve.

3.2.3 Confusion Matrix

A confusion matrix is a matrix that is widely used for the assessment of the performance of a classification model. They show the true positive (TP), false positive (FP), true negative (TN) and false negative (FN) for different classes successfully [47].

An example of confusion matrix is shown is Figure 3.2.

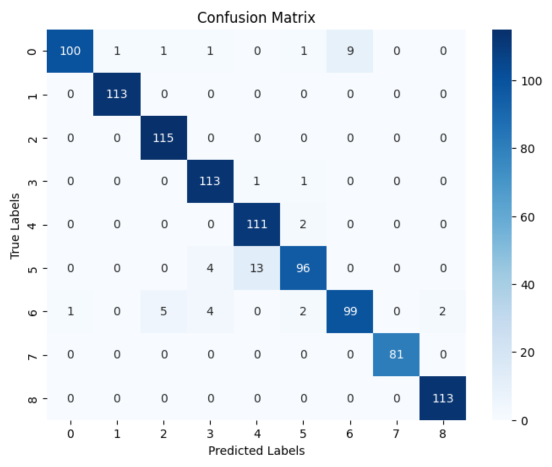


Figure 3.2: Example of Confusion matrix.

The confusion matrix in Figure 3.2 shows the performance of a classification model with 9 classes(labeled 0 to 8). The diagonal elements represents the correctly classified instances for each class, while off-diagonal elements show misclassifications. For instance, in Figure 3.2 is can be observed that class 5 has 96 correct and 17 incorrect classifications.

3.2.4 Operating Environment and Hyperparameter Settings

The operating environment for this report is Google Colab. Google Colab which stands for Google Colaboratory is a cloud-based, free tool that enable users to write and execute Python code through a browser. It offers a Jupyter notebook environment, on top of which you always have free access to GPUs and/ or TPUs depending on your needs expressed in multiple machine learning and data science applications. The service offers integration with such popular formats for data analysis as TensorFlow, PyTorch, and scikit-learn and enables additional packages' installation. Google Colab has proven to be very useful due to factors such as ease of use, processing speed, and capability to support cooperation among the users in their daily tasks in data analysis, machine learning, and artificial intelligence for students, researchers, and other individuals [48].

Hyperparameters in deep learning are parameters that define the structures of the model and the rules of training. They are defined prior to training and do not emerge during the process of learning from the data [14]. Some of them include: learning rate, number of layers, and batch size. Hyperparameters affect the model more than the variables and need to be optimized for better results [26].

Hyperparameters used for conducting experiments for this report are shown in

- *Learning rate:* Controls the step size during optimization. In this case, we have used a learning rate of 0.0001.
- *Batch size:* Number of training examples used in one iteration. In this case, we have used a batch of 1.

- *Number of epochs:* Number of times the model processes the entire dataset. For our experiments, we ran each model for 40 epochs.

- *Activation functions:* ReLU is an activation function that outputs the input if it's positive, otherwise zero, introducing non-linearity and mitigating the vanishing gradient problem.

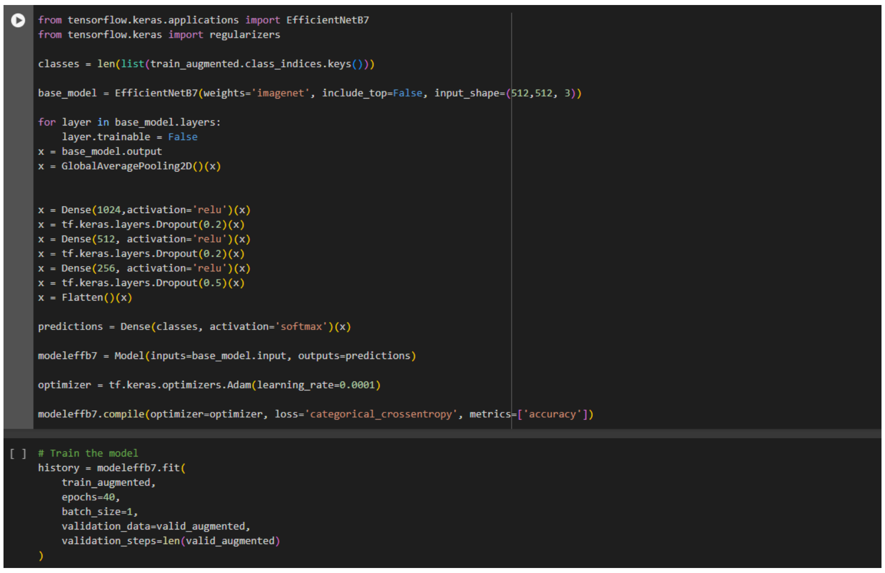
  Softmax, another activation function that converts logits into a probability distribution, used in the output layer of classification models.

- *Optimizer:* Adam, an optimizer algorithm for training deep learning models. It's popular for its ability to handle noisy data and achieve good results quickly, often requiring less hyperparameter tuning than other optimizers.

- *GlobalAveragePooling:* A method that brings down the spatial dimensions by performing a pooling operation.

- Dropout rate: A dropout technique that makes a percentage of neurons not participate in a node's training, which helps prevent overfitting. A dropout rate of 0.2 was used for experiments for this report.

The model architecture of EfficientNetB7 shown in Figure 3.3 tells us about the hyperparameters used in our experiments.



```python
from tensorflow.keras.applications import EfficientNetB7
from tensorflow.keras import regularizers

classes = len(list(train_augmented.class_indices.keys()))

base_model = EfficientNetB7(weights='imagenet', include_top=False, input_shape=(512,512, 3))

for layer in base_model.layers:
    layer.trainable = False
x = base_model.output
x = GlobalAveragePooling2D()(x)


x = Dense(1024,activation='relu')(x)
x = tf.keras.layers.Dropout(0.2)(x)
x = Dense(512, activation='relu')(x)
x = tf.keras.layers.Dropout(0.2)(x)
x = Dense(256, activation='relu')(x)
x = tf.keras.layers.Dropout(0.5)(x)
x = Flatten()(x)

predictions = Dense(classes, activation='softmax')(x)

modeleffb7 = Model(inputs=base_model.input, outputs=predictions)

optimizer = tf.keras.optimizers.Adam(learning_rate=0.0001)

modeleffb7.compile(optimizer=optimizer, loss='categorical_crossentropy', metrics=['accuracy'])
```

```python
[ ] # Train the model
history = modeleffb7.fit(
    train_augmented,
    epochs=40,
    batch_size=1,
    validation_data=valid_augmented,
    validation_steps=len(valid_augmented)
)
```

Figure 3.3: Model architecture of EfficientNetB7.

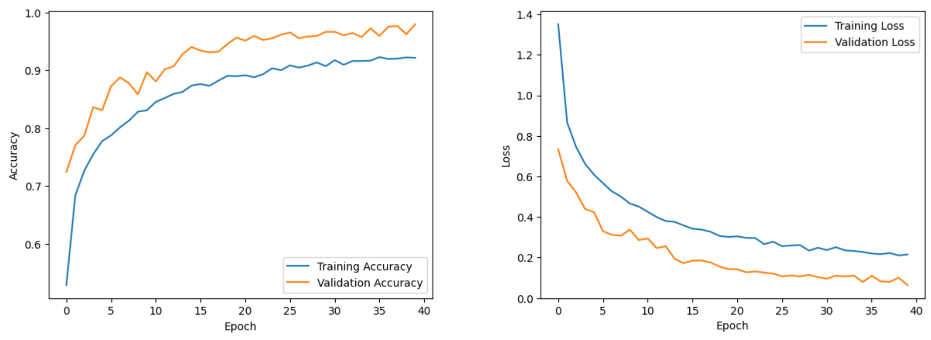
3.3 Experimental Results and Analysis

EfficientNetB7, EfficientNetB6, ResNet101, DenseNet169, MobileNetV2 and VGG19 models have been used to evaluate their performance in skin disease classification. The models were run for a duration of 40 epochs each, and their outcomes were observed and recorded.

3.3.1 Results from EfficientNetB7

After executing for 40 epochs, EfficientNetB7 achieved 92.16% as training Accuracy, 97.97% as validation accuracy and 97.17% as testing accuracy.

The accuracy curve and loss curve produced by EfficientNetB7 is shown in Figure 3.4.



(a) Accuracy curve. (b) Loss curve.

Figure 3.4: Accuracy curve and Loss curve produced by EfficientNetB7.

The confusion matrix produced by EfficeintNetB7 is shown is Figure 3.5.

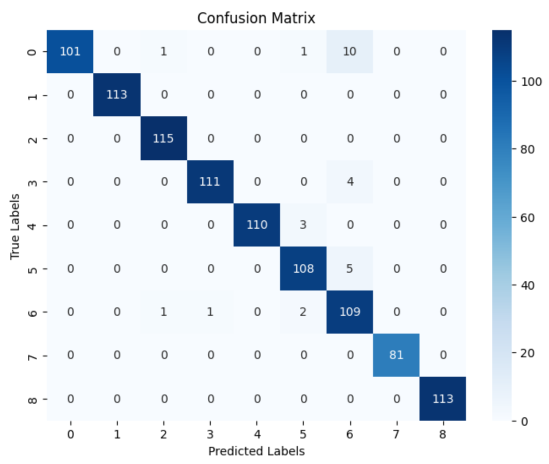
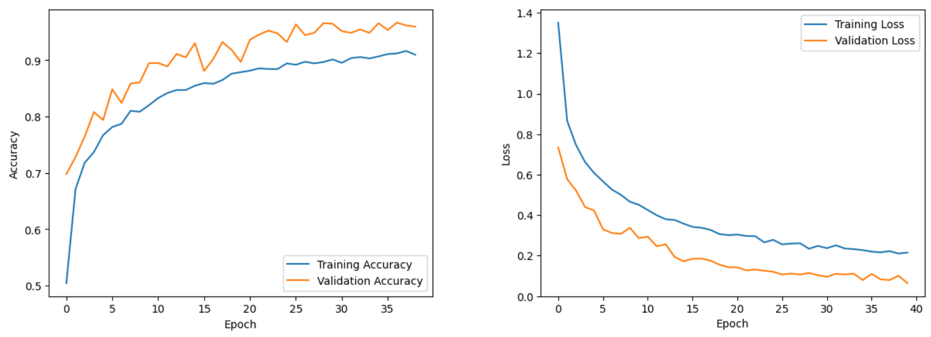


Figure 3.5: Confusion matrix produced by EfficientNetB7.

3.3.2 Results from EfficientNetB6

After executing for 40 epochs, EfficientNetB6 achieved 92.16% as training Accuracy, 97.97% as validation accuracy and 95.14% as testing accuracy.

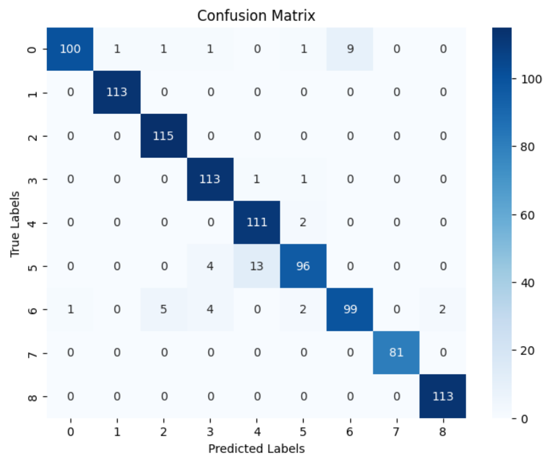
The accuracy curve and loss curve produced by EfficientNetB6 is shown in figure 3.6.



(a) Accuracy curve. (b) Loss curve.

Figure 3.6: Accuracy curve and Loss curve produced by EfficientNetB6.

The confusion matrix produced by EfficeintNetB6 is shown is Figure 3.7.



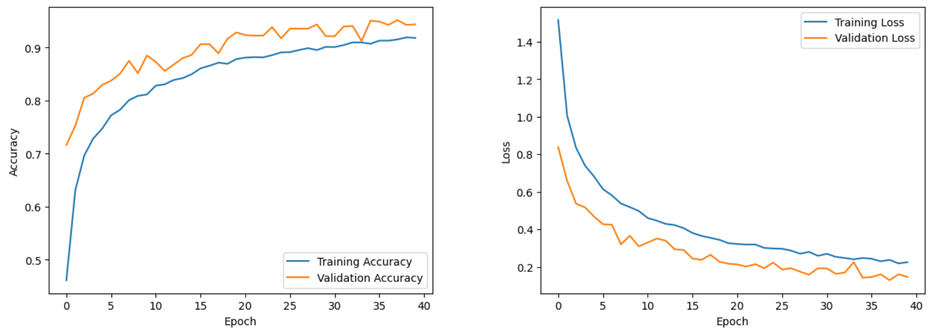
	0	1	2	3	4	5	6	7	8
0	100	1	1	1	0	1	9	0	0
1	0	113	0	0	0	0	0	0	0
2	0	0	115	0	0	0	0	0	0
3	0	0	0	113	1	1	0	0	0
4	0	0	0	0	111	2	0	0	0
5	0	0	0	4	13	96	0	0	0
6	1	0	5	4	0	2	99	0	2
7	0	0	0	0	0	0	0	81	0
8	0	0	0	0	0	0	0	0	113

Figure 3.7: Confusion matrix produced by EfficientNetB6.

3.3.3 Results from ResNet101

After executing for 40 epochs, ResNet101 achieved 91.78% as training Accuracy, 94.33% as validation accuracy and 95.75%

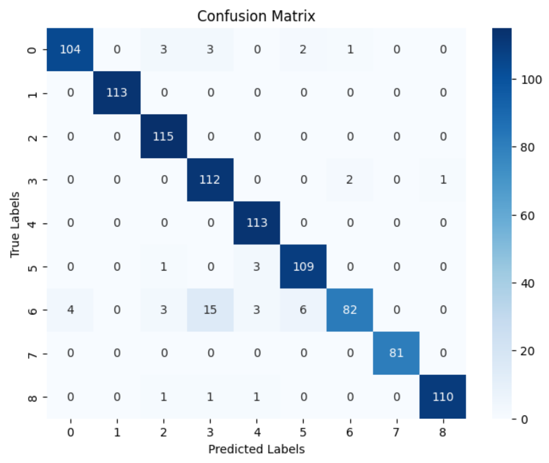
The accuracy curve and loss curve produced by ResNet101 is shown in Figure 3.8.



(a) Accuracy curve. (b) Loss curve.

Figure 3.8: Accuracy curve and Loss curve produced by ResNet101.

The confusion matrix produced by ResNet101 is shown is Figure 3.9.



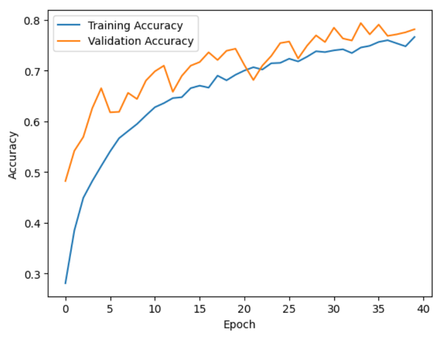
	0	1	2	3	4	5	6	7	8
0	104	0	3	3	0	2	1	0	0
1	0	113	0	0	0	0	0	0	0
2	0	0	115	0	0	0	0	0	0
3	0	0	0	112	0	0	2	0	1
4	0	0	0	0	113	0	0	0	0
5	0	0	1	0	3	109	0	0	0
6	4	0	3	15	3	6	82	0	0
7	0	0	0	0	0	0	0	81	0
8	0	0	1	1	1	0	0	0	110

Figure 3.9: Confusion matrix produced by ResNet101.

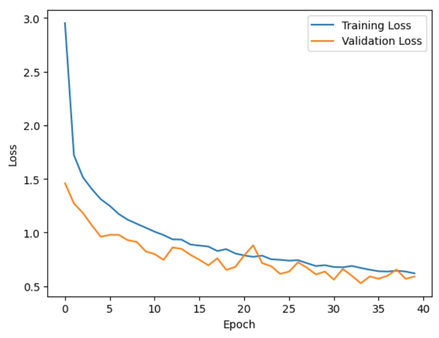
3.3.4 Results from DenseNet169

After executing for 40 epochs, DenseNet169 achieved 76.63% as training Accuracy, 79.17% as validation accuracy and 81.29% as testing accuracy.

The accuracy curve and loss curve produced by DenseNet169 is shown in Figure 3.10.



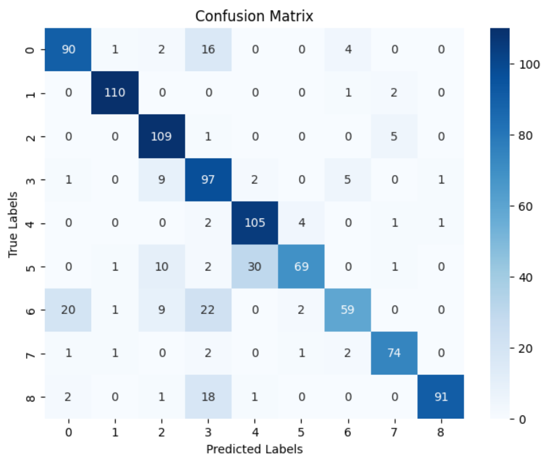
(a) Accuracy curve.



(b) Loss curve.

Figure 3.10: Accuracy curve and Loss curve produced by DenseNet169.

The confusion matrix produced by DenseNet169 is shown is Figure 3.11.



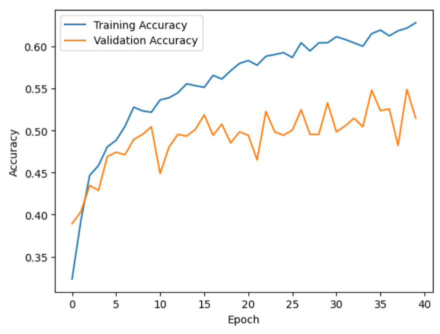
True \ Predicted	0	1	2	3	4	5	6	7	8
0	90	1	2	16	0	0	4	0	0
1	0	110	0	0	0	0	1	2	0
2	0	0	109	1	0	0	0	5	0
3	1	0	9	97	2	0	5	0	1
4	0	0	0	2	105	4	0	1	1
5	0	1	10	2	30	69	0	1	0
6	20	1	9	22	0	2	59	0	0
7	1	1	0	2	0	1	2	74	0
8	2	0	1	18	1	0	0	0	91

Figure 3.11: Confusion matrix produced by DenseNet169.

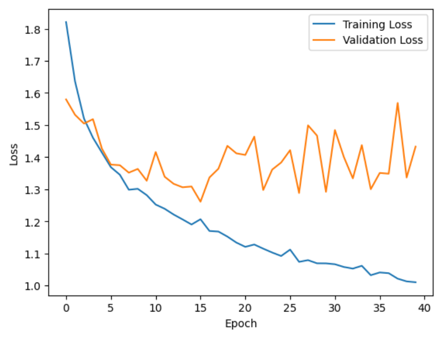
3.3.5 Results from MobileNetV2

After executing for 40 epochs, MobileNetV2 achieved 62.81% as training Accuracy, 51.46% as validation accuracy and 52.67% as testing accuracy.

The accuracy curve and loss curve produced by MobileNetV2 is shown in Figure 3.12.



(a) Accuracy curve.



(b) Loss curve.

Figure 3.12: Accuracy curve and Loss curve produced by MobileNetV2.

The confusion matrix produced by MobileNetV2 is shown is Figure 3.13.

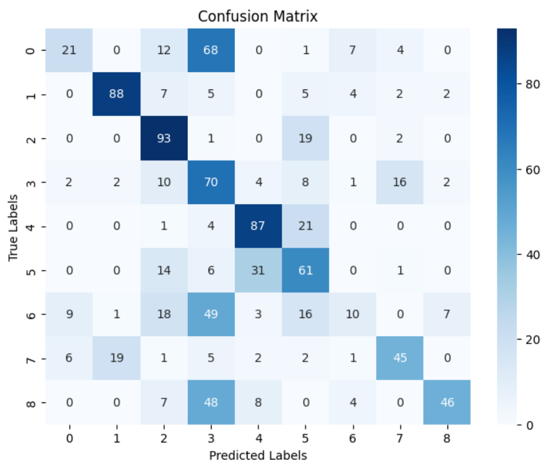
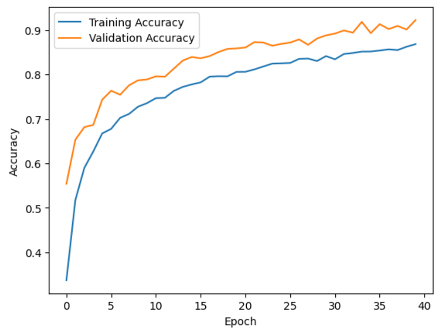


Figure 3.13: Confusion matrix produced by MobileNetV2.

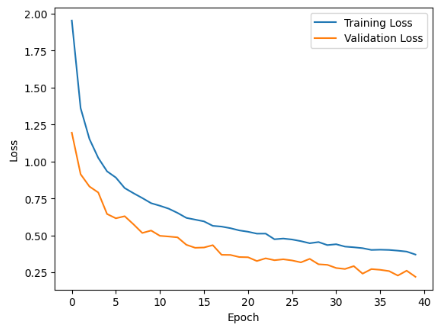
3.3.6 Results from VGG19

After executing for 40 epochs, VGG19 achieved 86.80% as training Accuracy, 92.21% as validation accuracy and 90.89% as testing accuracy.

The accuracy curve, and loss curve produced by VGG19 is shown in Figure 3.14.



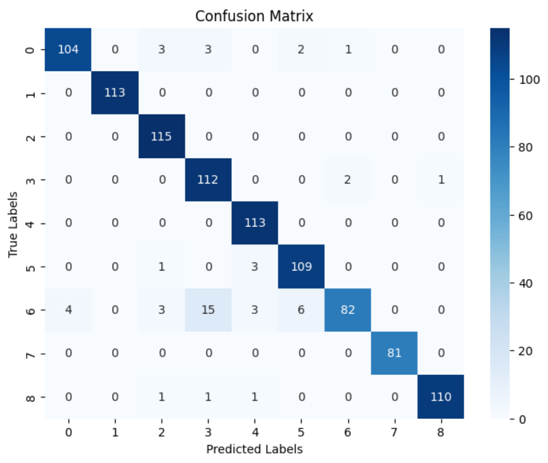
(a) Accuracy curve.



(b) Loss curve.

Figure 3.14: Accuracy curve and Loss curve produced by VGG19.

The confusion matrix of VGG19 is shown is Figure 3.15.



	0	1	2	3	4	5	6	7	8
0	104	0	3	3	0	2	1	0	0
1	0	113	0	0	0	0	0	0	0
2	0	0	115	0	0	0	0	0	0
3	0	0	0	112	0	0	2	0	1
4	0	0	0	0	113	0	0	0	0
5	0	0	1	0	3	109	0	0	0
6	4	0	3	15	3	6	82	0	0
7	0	0	0	0	0	0	0	81	0
8	0	0	1	1	1	0	0	0	110

Figure 3.15: Confusion matrix produced by VGG19.

3.3.7 Comparative Results

The training and validation accuracy results from the experiments' conduction on the dataset using training and validation data on deep learning classifier models(viz., EfficientNetB7, EfficientNetB6, ResNet101, DenseNet169, MobileNetV2, and VGG19). is shown in Table 3.1. In Table 3.1 the best performing model is highlighted in bold-italic font format.

Table 3.1: Accuracy results from the classifier models(viz., EfficientNetB7, EfficientNetB6, ResNet101, DenseNet169, MobileNetV2, and VGG19).

Classifier	**Training Accuracy (in %)**	**Validation Accuracy (in %)**
EfficientNetB7	***92.16***	***97.97***
EfficientNetB6	92.16	97.97
ResNet101	91.78	94.33
DenseNet169	76.63	79.13
MobileNetV2	62.81	51.46
VGG19	86.80	92.21

The test accuracy, and the average precision, recall, f1-score and support value of all the used models(viz., EfficientNetB7, EfficientNetB6, ResNet101, DenseNet169, MobileNetV2, and VGG19) is shown in Table 3.2. In Table 3.2 the best performing model is highlighted in bold-italic font format.

Table 3.2: Precision, recall, f1-score and support of models(viz., EfficientNetB7, EfficientNetB6, ResNet101, DenseNet169, MobileNetV2, and VGG19).

Classifier	**Testing Accuracy (in %)**	**Average**		
		Precision	**Recall**	**F1-score**
EfficientNetB7	***97.16***	***0.9740***	***0.9716***	***0.9720***
EfficientNetB6	95.14	0.9524	0.9514	0.9508
ResNet101	95.75	0.9592	0.9573	0.9573
DenseNet169	81.29	0.8319	0.8129	0.8099
MobileNetV2	52.67	0.5658	0.5267	0.5074
VGG19	90.89	0.9084	0.9089	0.9072

After experimenting and observing the results of the deep learning classifier model, it can be easily concluded that EfficientNetB7 has the best overall performance. It achieved training accuracy of 0.921 with 0.971 as testing accuracy and 0.979 as validation accuracy. EfficientNetB7 due to its optimized compound scaling of depth, width, and resolution is able to effectively learn complex patterns in the skin disease images and can classify them in different classes. ResNet101 also performed well, with a training accuracy of 0.959, testing accuracy of 0.957 and validation accuracy of 0.957.

3.3.8 Execution Time

The execution time of the various models used in this report is shown in Table 3.3.

Table 3.3: Execution time of the models(viz., EfficientNetB7, EfficientNetB6, ResNet101, DenseNet169, MobileNetV2, and VGG19).

Classifier	**Execution Time (hours)**
EfficientNetB7	2.47
EfficientNetB6	2.34
ResNet101	1.72
DenseNet169	1.45
MobileNetV2	1.25
VGG19	1.61

3.4 Confidence Interval Test

A confidence interval(CI) is an estimate of an interval within which the true value of one or several population parameters is expected to lie with a certain degree of confidence [49]. Confidence intervals are useful in research and data analysis because they provide more information than the mere point estimates, they provide a measure of the degree of precision of the estimate, allowing for more informed interpretations of results. It does not state the likelihood as to if the parameter will be found within the interval or not, but they measure the precision of the estimation procedure of a given interval in case the procedure is repeated in the future. Scholars and analysts apply confidence intervals in inferring the populations, comparing groups and presenting the results while considering the statistical accuracy.

The mathematical equation for finding CI is shown in Equation 3.6.

$$\text{CI} = error_s(h) \pm Z_N \sqrt{\frac{error_s(h)(1 - error_s(h))}{n}} \tag{3.6}$$

where $error_s(h)$ gives the sampling error experienced over a sample S constituted by n numbers of independently chosen examples and independently of discrete-valued hypothesis h. The CI constant at a confidence level of $p\%$ is denoted by Z_n, with specific values assigned for different confidence levels. For instance, for confidence levels for 90%, *95%*, and *99%* the corresponding values of Z_n are 1.64, 1.96, and 2.58 respectively.

Table 3.4 shows the summary of confidence interval for error rate at confidence levels of *90%*, *95%*, and *99%* produced by the models(viz., EfficientNetB7, EfficientNetB6, ResNet101, DenseNet169, MobileNetV2, and VGG19).

Table 3.4: Summary of confidence interval for error rate at confidence levels (CL) of *90%*, *95%*, and *99%* produced by the models(viz., EfficientNetB7, EfficientNetB6, ResNet101, DenseNet169, MobileNetV2, and VGG19).

Methods	**Error Rate**	**Support**	**Error Bound**		
			CL 90%	**CL 95%**	**CL 99%**
EfficientNetB7	0.0284	989	0.0086	0.0104	0.0136
EfficientNetB6	0.0486	989	0.1121	0.0134	0.0176
ResNet101	0.0425	989	0.0105	0.0125	0.0165
DenseNet169	0.1871	989	0.0203	0.0243	0.0319
MobileNetV2	0.4733	989	0.0260	0.0311	0.0409
VGG19	0.0911	989	0.0150	0.0179	0.0236

For Table 3.4, error rate for a model can calculated by subtracting the accuracy of that model by 1 i.e., error rate = 1 — accuracy of model, and support is the number of test samples in our dataset.

3.5 Summary

In this chapter, we see the methodology of the experiments conducted on the skin disease image classification dataset. We experimented with the few of the deep learning image classifier models. These results from the experiments helped us compare the performance of the used classifiers and also choose the best performing out of them. The confidence interval test of the model(viz., EfficientNetB7, EfficientNetB6, ResNet101, DenseNet169, MobileNetV2, and VGG19) is also shown.

Chapter 4

Conclusion and Future work

4.1 Conclusion

In this report, we have experimented and evaluated various deep learning classifiers for skin disease image classification. EfficientNetB7 demonstrated exceptional performance, achieving high accuracy and efficiency due to its optimized compound scaling of depth, width, and resolution. The model's advanced architecture allowed it to effectively learn complex patterns in the data while maintaining a manageable number of parameters. This performance underscores the potential of EfficientNetB7 as a powerful tool for automated skin disease diagnosis, providing reliable and swift preliminary assessments that can assist dermatologists in clinical practice.

4.2 Future Work

Future work of skin disease classification will focus on several key area mentioned below:

– Increasing the number of diseases the model can classify, by incorporating more skin disease classes into the dataset.

– Patient data such as history, demographics, and other clinical data can be incorporated for better diagnostic assistance.

– Sophisticated data augmentation can be applied to simulate real life variations.

– Further experiments with hyperparameter tuning can be done to fine tune classifier models.

– Hybrid models can be experimented with, for better feature extraction.

– Real life clinical trails can be conducted to validate performance against human dermatologists.

– Mobile apps for remote diagnostics can be built.

– Increase generalization of the model.

Bibliography

[1] B. Zhang, X. Zhou, Y. Luo, *et al.*, "Opportunities and challenges: Classification of skin disease based on deep learning," *Chinese Journal of Mechanical Engineering volume*, vol. 34, no. 112, pp. 1–14, 2021.

[2] D. T. Jamison, J. G. Breman, A. R. Measham, *et al.*, *Disease Control Priorities in Developing Countries*, D. T. Jamison, J. G. Breman, A. R. Measham, *et al.*, Eds. New York, USA: Oxford University Press, 2006.

[3] R. J. Hay, N. E. Johns, H. C. Williams, *et al.*, "The global burden of skin disease in 2010: An analysis of the prevalence and impact of skin conditions," *Journal of Investigative Dermatology*, vol. 134, no. 6, pp. 1527–1534, 2014.

[4] J. L. Bolognia, J. V. Schaffer, and L. Cerroni, *Dermatology.* Elsevier, 2018.

[5] J. G. Marks and J. J. Miller, *Lookingbill and Marks' Principles of Dermatology.* Elsevier, 2019.

[6] T. Burns, S. Breathnach, N. Cox, and C. Griffiths, *Rook's Textbook of Dermatology.* Wiley-Blackwell, 2010.

[7] N. R. Abbasi, H. M. Shaw, D. S. Rigel, *et al.*, "Early diagnosis of cutaneous melanoma," *JAMA*, vol. 292, no. 22, pp. 2771–2776, 2004.

[8] A. A. Marghoob, R. P. Usatine, and N. Jaimes, "Dermoscopy for the family physician," *American Family Physician*, vol. 88, no. 7, pp. 441–450, 2013.

[9] Y. Fujisawa *et al.*, "Deep-learning-based, computer-aided classifier developed with a small dataset of clinical images surpasses board-certified dermatologists in skin tumour diagnosis," *British Journal of Dermatology*, vol. 180, no. 2, pp. 373–381, 2019.

[10] A. Esteva *et al.*, “Dermatologist-level classification of skin cancer with deep neural networks,” *Nature*, vol. 542, no. 7639, pp. 115–118, 2017.

[11] A. S. Adamson and A. Smith, “Machine learning and health care disparities in dermatology,” *JAMA Dermatology*, vol. 154, no. 11, pp. 1247–1248, 2018.

[12] A. T. Young *et al.*, “Artificial intelligence and dermatology: Opportunities, challenges, and future directions,” *Seminars in Cutaneous Medicine and Surgery*, vol. 38, no. 1, E31–E37, 2019.

[13] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

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

[15] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[16] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, “Attention is all you need,” in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[17] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[18] I. Goodfellow, J. Pouget-Abadie, M. Mirza, *et al.*, “Generative adversarial nets,” in *Advances in Neural Information Processing Systems*, vol. 27, 2014.

[19] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT Press, 2018.

[20] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[21] S. J. Pan and Q. Yang, “A survey on transfer learning,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[22] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[23] C. Shorten and T. M. Khoshgoftaar, “A survey on image data augmentation for deep learning,” *Journal of Big Data*, vol. 6, no. 1, pp. 1–48, 2019.

[24] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[25] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[26] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, pp. 281–305, 2012.

[27] L.-s. Wei and Q. G., "Skin disease recognition method based on image color and texture features," *Computational and Mathematical Methods in Medicine*, vol. 2018, 2018.

[28] P. N. Srinivasu and J. G., "Classification of skin disease using deep learning neural networks with mobilenet v2 and lstm," *Sensors*, vol. 21, no. 8, 2021.

[29] J. Robinson, "Sun exposure, sun protection and vitamin d," *IEEE Journal of the American Medical Association*, vol. 294, no. 13, pp. 1541–1543, Oct. 2005.

[30] R. Sumithra and M. Suhil, "Segmentation and classification of skin lesions for disease diagnosis," in *Procedia Computer Science*, vol. 45, 2015, pp. 76–85.

[31] K. S. M., K. J. R., and K. G., "Skin cancer diagnostic using machine learning techniques-shearlet transform and naive bayes classifier," *International Journal of Engineering and Advanced Technology*, vol. 9, no. 2, pp. 3478–3480, 2019.

[32] Z. Wu and S. Zhang, "Studies on different cnn algorithms for face skin disease classification based on clinical images," *IEEE Access*, vol. 7, pp. 66 505–66 511, 2019.

[33] L. Y. C., J. S.-H., and W. H.-H., "Wonderm: Skin lesion classification with fine-tuned neural networks," 2018, pp. 178–196.

[34] Z. T. Fernando and P. T., "Docaid: Predictive healthcare analytics using naive bayes classification," in *International Conference on Advances in Computing*, 2016, pp. 67–75.

[35] A. Shrestha and A. Mahmood, “Improving genetic algorithm with fine tuned crossover and scaled architecture,” *Journal of Mathematics*, vol. 2016, pp. 401–584, 2016.

[36] E. Saber, S. Ruhul, and E. Daryl, “A new genetic algorithm for solving optimization problem,” *Engineering Applications of Artificial Intelligence*, vol. 27, pp. 57–69, 2013.

[37] W. Hsu, “Bayesian classification,” in *Encyclopedia of Database Systems*, L. Liu and M. T. Ozsu, Eds., New York, NY, USA: Springer, 2018.

[38] K. Roy *et al.*, “Skin disease detection based on different segmentation techniques,” in *Proceedings of the 2019 International Conference on Opto Electronics and Applied Optics (Optronix)*, Kolkata, India, 2019, pp. 1–5.

[39] X. Zhang *et al.*, “Towards improving diagnosis of skin diseases by combining deep neural network and human knowledge,” *BMC Medical Informatics and Decision Making*, vol. 18, no. 1, p. 59, Dec. 2018.

[40] F. Mahmood, H. Dabbagh, and A. A. Mohammed, “Comparative study on using chemical and natural admixtures (grape and mulberry extracts) for concrete,” *Case Studies in Construction Materials*, vol. 15, e00603, Dec. 2021.

[41] S. Kumar, “A quest for sustainium (sustainability premium): Review of sustainable bonds,” *Academy of Accounting and Financial Studies Journal*, vol. 26, no. 2, pp. 1–18, 2022.

[42] R. E. Shaju, *Skin disease classification image dataset*, `https://www.kaggle.com/datasets/riyaelizashaju/skin-disease-classification-image-dataset`.

[43] M. Tan and Q. V. Le, “Efficientnet: Rethinking model scaling for convolutional neural networks,” in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 97, PMLR, 2019, pp. 6105–6114.

[44] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *arXiv preprint arXiv:1512.03385*, Dec. 2015.

[45] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," *arXiv preprint arXiv:1801.04381*, 2018.

[46] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[47] T. Fawcett, "An introduction to roc analysis," *Pattern Recognition Letters*, vol. 27, no. 8, 2006.

[48] E. Bisong, "Google colaboratory," in *Building Machine Learning and Deep Learning Models on Google Cloud Platform*. Berkeley, CA: Apress, 2019, ch. 7.

[49] S. Greenland, S. J. Senn, K. J. Rothman, *et al.*, "Statistical tests, P values, confidence intervals, and power: A guide to misinterpretations," *European Journal of Epidemiology*, vol. 31, no. 4, pp. 337–350, 2016.