# ADVANCING DERMATOLOGICAL DIAGNOSIS WITH MULTICLASS LESION ANALYSIS

*D. Saiteja*

*22955A1210*

# ADVANCING DERMATOLOGICAL ANALYSIS WITH MULTICLASS LESION ANALYSIS

*A Project Report*
*submitted in partial fulfillment of the*
*requirements for the award of the degree of*

**Bachelor of Technology**
**in**
**Information Technology**

*by*

**D. Saiteja**
**22955A1210**

**Department of Information Technology**

# INSTITUTE OF AERONAUTICAL ENGINEERING

**(Autonomous)**
**Dundigal, Hyderabad – 500 043, Telangana**

**December, 2024**

# DECLARATION

I certify that

a. The work contained in this report is original and has been done by me under the guidance of my supervisor(s).

b. The work has not been submitted to any other Institute for any degree or diploma.

c. I have followed the guidelines provided by the Institute in preparing the report.

d. I have conformed the norms and guidelines given in the Ethical Code of Conduct of the Institute

e. Whenever I have used materials (data, theoretical analysis, figures and text) from other sources, I have given due credit to them by citing them in the text of the report and giving their details in the references. Further, I have taken permission from the copyright owners of the sources, whenever necessary.

Place:                                                          **Signature of the Student**

Date:                                                           **22955A1210**

# CERTIFICATE

This is to certify that the project report entitled **Advancing Dermatological Diagnosis with Multiclass Lesion Analysis** submitted by **Mr. D. Saiteja** to the Institute of Aeronautical Engineering, Hyderabad in partial fulfillment of the requirements for the award of the Degree Bachelor of Technology in **Information Technology** is a bonafide record of work carried out by him under my guidance and supervision. The contents of this report, in full or in parts, have not been submitted to any other Institute for the award of any Degree.

**Supervisor**                                                   **Head of the Department**

**Dr. M Pala Prasad Reddy**                          **Dr. M Purushotham Reddy**

**Date:**

# APPROVAL SHEET

This project report entitled **Advancing Dermatological Diagnosis with Multiclass Lesion Analysis** by **D. Saiteja** is approved for the award of the Degree Bachelor of Technology in **Information Technology.**

**Examiner**                                                              **Supervisor**

                                                                **Dr. M Pala Prasad Reddy**

**Principal**

**Dr. L V Narasimha Prasad**

**Date:**

**Place: Dundigal**

# ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose constant guidance and encouragement crown all the efforts with success.

I thank our college management and **Sri. M Rajashekar Reddy,** Chairman, IARE, for providing me the necessary infrastructure to carry out the project work.

I express my sincere thanks to **Dr. L V Narasimha Prasad,** beloved Principal who has been a great source of information for my work and to **Dr. M Purushotham Reddy,** Professor and Head, Department of Information Technology, for extending his support to carry out this project work.

I'm especially thankful to my internal supervisor to **Dr. M Pala Prasad Reddy,** Professor, Department of Information Technology, for their internal support and professionalism who helped me in shaping the project into successful one. A special thanks to my beloved Project Coordinator **Mr. N Raghava Rao,** Associate Professor, Department of Information Technology.

I take this opportunity to express my thanks to one and all who directly or indirectly helped me bring effort to present form.

*D. Saiteja*
*22955A1210*

# ABSTRACT

Skin cancer is the most common form of cancer globally, ranking as the 17th most prevalent cancer with 2-3 million non-melanoma and 132,000 melanoma cases worldwide. It arises due to the abnormal growth of skin cells, primarily caused by excessive exposure to ultraviolet rays. Early detection and accurate classification of skin lesions are crucial for effective treatment and improved patient outcomes. Given the challenges dermatologists face in accurately diagnosing skin cancer, there is an urgent need for an automated and efficient diagnostic system. This study presents an automated system for classifying skin lesions into seven different types of skin cancer using a Convolutional Neural Network (CNN) model, trained on the HAM10000 dataset, which includes 10,015 dermatoscopic images of various skin lesions. Our approach leverages the MobileNet architecture, renowned for its efficiency and effectiveness in image classification tasks. The MobileNet model, pre-trained on the extensive ImageNet dataset, is fine-tuned on the HAM10000 dataset to specialize in skin lesion classification by employing transfer learning. The model achieved an overall accuracy of 89.21% across the seven skin cancer classes, with top-2 and top- 3 accuracies of 96.55% and 98.45%, respectively. The trained model demonstrated promising results, underscoring its potential as a reliable tool for assisting dermatologists in the early detection of skin cancer.

**Keywords:** Skin Cancer, CNN, MobileNet, HAM10000, Lesion Classification, ImageNet

# CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# CHAPTER 1
# INTRODUCTION

## 1.1  Introduction to Skin Cancer

Skin cancer, the most prevalent form of cancer globally, poses significant challenges in both diagnosis and treatment. Each year, millions of new cases are reported worldwide, with non-melanoma skin cancers accounting for 2-3 million cases and melanoma, the deadliest form, seeing approximately 132,000 new cases annually. Prolonged exposure to ultraviolet (UV) radiation from the sun is the primary cause of skin cancer, inducing DNA damage in skin cells that can lead to mutations and potentially malignant growths. Early detection and accurate classification of skin lesions are crucial for effective treatment and improved patient outcomes.

Early detection of skin cancer, especially melanoma, is critical for initiating timely and effective treatment, significantly improving survival rates. Melanoma, although less common than other types of skin cancer, accounts for a majority of skin cancer-related deaths due to its high potential for metastasis. Surgical excision of early-detected melanomas can often lead to excellent outcomes, whereas advanced-stage melanomas require more aggressive treatments and are associated with lower survival rates and increased healthcare costs. Thus, enhancing the accuracy and efficiency of early skin cancer detection remains a pressing public health priority.

Dermatological diagnosis involves the meticulous examination and identification of a wide spectrum of skin disorders, ranging from benign conditions to severe diseases like skin cancer. This diagnostic process typically includes visual inspection, dermatoscopic evaluation using specialized instruments, and, in cases where necessary, biopsy for histopathological examination. Accurate diagnosis directly influences the treatment plan, prognosis, and overall patient outcomes. Dermatologists heavily rely on their expertise and advanced tools to distinguish between various types of skin conditions. However, the visual similarities between benign and malignant lesions, combined with the vast diversity of skin conditions, complicate the diagnostic process and increase the likelihood of errors.

Distinguishing between benign and malignant skin lesions poses a persistent challenge even for experienced dermatologists, frequently resulting in misdiagnosis and delayed treatment. This underscores the urgent need for automated and efficient diagnostic systems to assist dermatologists in accurately diagnosing skin cancer at its earliest stages. Multiclass lesion analysis presents formidable challenges due to the wide-ranging diversity and complexity of skin lesions. Skin cancers such as melanoma, basal cell carcinoma, and squamous cell carcinoma exhibit distinct visual characteristics that can be subtle and difficult to differentiate from benign lesions like nevi, seborrheic keratoses, and dermatofibromas. Factors such as color, texture, shape, and size contribute to the heterogeneity in lesion appearance, further complicating the diagnostic process. Additionally, the similarities between different types of lesions and variations within the same type necessitate advanced diagnostic tools capable of high-resolution image analysis and precise classification. Traditional diagnostic methods often fall short in meeting these demands, underscoring the necessity for automated systems that leverage cutting-edge machine learning techniques to enhance diagnostic accuracy and reliability.

Recent advancements in artificial intelligence (AI) and deep learning have revolutionized skin cancer diagnosis, particularly through Convolutional Neural Networks (CNNs), which excel in image recognition tasks and are well-suited for medical image analysis. Among various CNN architectures, MobileNet stands out due to its lightweight design, balancing speed and accuracy, which makes it suitable for deployment on mobile and embedded devices. Unlike heavier models such as VGGNet and ResNet, MobileNet requires fewer computational resources while maintaining high performance.

## 1.2 Existing System

The current system for skin cancer diagnosis largely depends on clinical evaluation by dermatologists, followed by dermatoscopic examination and, if necessary, a biopsy for histopathological verification. Dermatologists visually inspect skin lesions, often using a dermatoscope, to differentiate between benign and malignant lesions. While this process is widely used and effective to an extent, it is highly reliant on the experience and skill of the

dermatologist. Diagnosing skin cancer based solely on visual inspection is challenging due to the visual similarities between various types of skin lesions. Misdiagnosis, especially with dangerous forms like melanoma, can lead to delayed treatment, which significantly worsens the prognosis.

In recent years, digital dermatoscopy has emerged, allowing images to be captured and analyzed more accurately. However, these systems often require manual annotation and interpretation by a specialist, which limits their use for large-scale screening. Furthermore, the lack of universally accessible medical resources, particularly in underserved regions, creates a gap in timely diagnosis and treatment. Although some AI-based diagnostic tools have been developed, they are not widely integrated into clinical practice, often due to their complexity, cost, and the need for substantial computational resources.

## 1.2.1  Limitations of Existing System

While the existing system for skin cancer diagnosis provides essential tools for detecting skin lesions, it has several limitations. One of the primary constraints is the dependence on the expertise of dermatologists, which can lead to inconsistencies and misdiagnoses due to human error. The visual similarity between benign and malignant lesions makes accurate diagnosis difficult, often requiring multiple tests or expert consultations. This not only delays treatment but also increases healthcare costs.

Another limitation is accessibility. In many regions, especially rural or underserved areas, there is a shortage of trained dermatologists. As a result, early detection of skin cancer becomes difficult, increasing the likelihood of advanced-stage diagnoses, which are harder to treat. Existing digital dermatoscopic systems, while effective, are not scalable for mass screening due to their reliance on manual interpretation. Furthermore, these systems can be expensive, limiting their availability to only well-equipped healthcare facilities.

Lastly, the current AI-based systems, though promising, require high computational power and are not yet widely accepted in clinical environments. The training and fine-tuning of AI models for skin cancer diagnosis demand vast datasets and significant

technical expertise, making them impractical for real-time use in most medical settings. As a result, there remains a gap in providing a universal, reliable, and automated solution for skin lesion classification

## 1.3  Proposed System

The solution for the above problem that involves leveraging advancements in artificial intelligence and deep learning, particularly Convolutional Neural Networks (CNNs) like MobileNet, which excel in medical image analysis. By training these models on extensive datasets of dermatoscopic images, such as the HAM10000 dataset, and employing transfer learning techniques, MobileNet can enhance diagnostic accuracy and efficiency in identifying various types of skin cancer early on. This approach promises to reduce misdiagnosis rates, facilitate timely interventions, and ultimately improve patient outcomes.

Enhancing early detection and classification of skin cancer through innovative AI-driven diagnostic tools like MobileNet is crucial for reducing mortality rates and healthcare costs associated with advanced-stage cancers. By addressing the complexities of lesion analysis and improving diagnostic precision, these technologies have the potential to revolutionize dermatological practice, offering a proactive approach to managing this prevalent and often life-threatening disease.

### 1.3.1  Objectives

Primary objectives of the proposed system include the following key points. They are listed below:

i.  **Automated Diagnosis**

- Uses AI to analyze medical data without human intervention.

- Enhances accuracy by detecting subtle patterns in skin lesions.

- Provides rapid assessments, improving patient care and treatment.

- Supports clinicians with AI-driven insights for better decision-making.

- Requires validation, model updates, and adherence to regulations.

ii. **High Accuracy**

- Accurately classifies skin lesions (benign, malignant) with minimal errors.

- Reduces false positives and false negatives.

- Achieved through diverse, large dataset training.

iii. **Efficiency**

- Rapid data processing and fast diagnostic results.

- Reduces diagnostic turnaround times in clinical settings.

- Optimizes workflow, allowing more focus on patient care.

iv. **Transfer Learning**

- Fine-tunes pre-trained models (e.g., MobileNet) on specific datasets (HAM10000).

- Improves performance by leveraging knowledge from large datasets.

- Reduces training time and computational resources.

# CHAPTER 2
# LITERATURE REVIEW

Skin conditions are usually diagnosed through a physical examination, sometimes followed by further tests such as dermoscopy, biopsy, and histological examination. Skin diseases are common and can be difficult to identify through photographs due to their delicate structure [1]. However, deep convolutional neural networks (CNNs) can provide detailed analysis of skin lesions using pixels and infection markers, without requiring rough illustrations. In this study, a CNN was developed and trained using a dataset of 129, 450 clinical images of 2, 032 different skin disorders[1]. The accuracy of the CNN was compared to that of board-certified dermatologists who used biopsy-proven clinical images to differentiate between benign and malignant skin conditions. The results showed that the CNN was able to diagnose skin conditions as effectively as dermatologists. With the widespread use of smartphones, basic diagnostic care for skin conditions may become widely available to the 6.3 billion mobile phone subscribers by 2021[1].

Skin cancer is one of the most prevalent and deadly types of cancer. Dermatologists diagnose this disease primarily visually [2]. Multiclass skin cancer classification is challenging due to the fine-grained variability in the appearance of its various diagnostic categories. On the other hand, recent studies have demonstrated that convolutional neural networks outperform dermatologists in multiclass skin cancer classification. We developed a preprocessing image pipeline for this work [2]. We removed hairs from the images, augmented the dataset, and resized the imageries to meet the requirements of each model. By performing transfer learning on pre-trained ImageNet weights and fine-tuning the Convolutional Neural Networks, we trained the EfficientNets B0-B7 on the HAM10000 dataset [2]. We evaluated the performance of all EfficientNet variants on this imbalanced multiclass classification task using metrics such as *Precision*, *Recall*, *Accuracy*, *F1 Score*, and *Confusion Matrices* to determine the effect of transfer learning with fine-tuning. This article presents the classification scores for each class as *Confusion Matrices* for all eight models [2]. Our best model, the EfficientNet B4, achieved an *F1 Score* of 87 percent and

a Top-1 Accuracy of 87.91 percent. We evaluated EfficientNet classifiers using metrics that take the high-class imbalance into account [2]. Our findings indicate that increased model complexity does not always imply improved classification performance. The best performance arose with intermediate complexity models, such as EfficientNet B4 and B5 [2]. The high classification scores resulted from many factors such as resolution scaling, data enhancement, noise removal, successful transfer learning of ImageNet weights, and fine-tuning [2]. Another discovery was that certain classes of skin cancer worked better at generalization than others using Confusion Matrices.

Skin Cancer is an emerging global health problem considering the increasing prevalence of harmful ultraviolet rays in the earth's environment. The researchers had discovered a further 10 percent depletion of the ozone layer will intensify the problem 12 of skin cancer with an additional 300,000 non-melanoma and 4,500 melanoma cases each year [3]. Currently, every year 123,000 melanomas and 30,00,000 non-melanoma cases are recorded worldwide [3]. The recent study on the prevention of skin cancer reports 90 percent of non-melanoma and 86 percent of melanoma cases induced by excessive exposure of ultraviolet rays [3]. The UV radiation detriments the DNA present at the inner layers of skin, triggering the uncontrolled growth of skin cells, which may even emerge as a skin cancer [3]. The most straightforward and effective solution to control the mortality rate for skin cancer is the timely diagnosis of skin cancer as the survival rate for melanoma patients in a five-year timespan is 99 percent when diagnosed and screened at the early stage [3]. Moreover, the most mundane skin cancer types BCC and SCC are highly treatable when early diagnosed and treated adequately [3]. Dermatologist primarily utilizes visual inspection to diagnose skin cancer, which is a challenging task considering the visual similarity among skin cancers. However, dermoscopy has been popular for the diagnosis of skin cancer recently considering the ability of dermoscopy to accurately visualize the skin lesions not discernible with the naked eye. Reports on the diagnostic accuracy of clinical dermatologists have claimed 80 percent diagnostic accuracy for a dermatologist with experience greater than ten years, whereas the dermatologists with experience of 3-5 years were able to achieve diagnostic accuracy of only 62 percent, the accuracy further dropped

for less experienced dermatologists [3]. The studies on Dermoscopy imply a need to develop an automated efficient, and robust system for the diagnosis of skin cancer since the fledgling dermatologists may deteriorate the diagnostic accuracy of skin lesions [3].

Melanoma is an aggressive skin cancer resulting from uncontrolled growth of melanocytes, responsible for 75% of skin cancer deaths. Early detection can reduce mortality and help patients far from diagnosis centers. Researchers have applied various algorithms for melanoma classification, using datasets like ISBI 2016 [4]. Sara et al. compared CNN models, highlighting the impact of data preprocessing and augmentation. Savy et al. used VGG16 and AlexNet, showing transfer learning improves accuracy, while Codella et al. applied segmentation and classification methods, achieving a Jaccard index of 0.843 and accuracy of 95.3% [4]. However, limitations in preprocessing and model optimization affect accuracy, as noted by Sara et al. and others. Parkash et al. applied CNN to a dataset with seven skin diseases, while Jwan et al. focused on deep CNN for classification, using the ABCDE rule to distinguish between melanoma and non-melanoma. In this paper [4], a hybrid model combining ResNet50 and VGG16 is proposed, using data preprocessing and downsampling for balanced training, leading to evaluation through accuracy, precision, and recall. The paper discusses methodology, data preprocessing, feature extraction, classification, experimental results, and concludes with findings.

Skin cancer is one of the most common types of cancer worldwide, with one in five Americans expected to develop it in their lifetime. Among its types, malignant melanoma is the deadliest, causing around 10,000 deaths annually in the U.S [5]. Early detection can significantly improve survival rates, with over 95% survival for early-stage detection compared to less than 20% for late-stage detection.

Non-invasive tools such as macroscopic images from standard cameras or mobile phones can assist dermatologists in diagnosis, but these images often suffer from poor quality. Dermoscopic devices, however, provide better image quality and allow for improved differentiation between skin lesion types. Despite established diagnostic schemes like the

ABCD rule and the 7-point checklist, visual inspection of dermoscopic images can be subjective and challenging due to varying lesion types and confounding factors [5].

To address this, computer-aided methods for dermoscopic lesion classification typically involve lesion segmentation, feature extraction, and classification. Preprocessing steps like contrast enhancement, white balancing, and artifact removal (e.g., skin hairs or bubbles) are often required [5]. With the rise of deep convolutional neural networks (CNNs) and their success in natural image classification, there is a growing trend to apply CNNs to medical image analysis, including skin lesion classification.

This paper proposes using multiple pre-trained CNN models - AlexNet, VGG16, and ResNet-18—as feature extractors for skin lesion classification. These models, trained on ImageNet, are used in combination with support vector machines (SVMs) to classify lesions as malignant melanoma, seborrheic keratosis, or benign nevi. By extracting features from different layers and combining the outputs of multiple networks, the authors aim to achieve classification performance comparable to state-of-the-art algorithms [5]. The final model employs ensemble learning to fuse the SVM outputs for optimal discrimination between the lesion classes.

This paper addresses the challenge of melanoma skin cancer detection using automated systems and machine learning techniques. It outlines a three-phase model to improve detection accuracy. The first phase focuses on data collection and preprocessing. The dataset is derived from the ISIC (International Skin Imaging Collaboration) dataset, consisting of around 1,000-1,500 images [6]. In this phase, preprocessing tasks such as hair, glare, and shading removal are performed using techniques like Hough Transform and MATLAB filters, enhancing image quality for further analysis.

In the second phase, the images undergo segmentation and feature extraction. The segmentation process uses methods like Otsu segmentation, modified Otsu segmentation, and watershed segmentation to identify and isolate the region of interest. Features related to color, shape, size, and texture are extracted from the segmented images to facilitate accurate classification [6].

The third and most critical phase involves model design and training. The authors utilize three machine learning techniques: Backpropagation Neural Networks, Support Vector Machines (SVM), and Convolutional Neural Networks (CNN) [6]. The preprocessed and segmented images are fed into these models for training, aiming to classify the images as malignant or benign melanoma. Each classifier, especially the CNNs, is evaluated for accuracy in prediction. The SVM algorithm constructs hyperplanes to separate the malignant from the benign cases, while CNNs are used for advanced feature learning, given their superior performance in image classification tasks.

The novelty of the approach lies in the efficient pre-processing and model design, ensuring quick and accurate melanoma detection. The proposed method, evaluated on the ISIC dataset, yields high accuracy and holds promise for application in real-time diagnostics [6]. The paper concludes by emphasizing the effectiveness of combining image processing and machine learning for improving melanoma detection accuracy.

This paper proposes a deep convolutional neural network (CNN) model to classify six prevalent skin diseases: acne, athlete's foot, chickenpox, eczema, skin cancer, and vitiligo. The authors highlight the increasing significance of computer-aided systems for early diagnosis of skin diseases due to the high prevalence and global impact [7]. A dataset comprising 3,000 images from Google Images, Derma.Net, and ISIC archives was created, with each disease class containing 500 images. Data preprocessing, including hair and glare removal, was conducted to enhance the images.

The CNN model consists of four convolutional layers, pooling layers, and fully connected layers, implemented using the Keras library. Hyperparameter tuning was conducted to improve the model's performance [7]. The final model achieved an accuracy of 81.75%, calculated using a holdout method where 90% of the images were used for training and 10% for testing [7]. Key hyperparameters, such as validation split, learning rate, number of layers, dropout rate, and batch size, were fine-tuned to optimize the model's performance. The authors also developed an Android application called Skinvy, which allows users to capture images of skin lesions and classify them in real-time using the

trained CNN model. The application aims to increase accessibility and assist both dermatologists and patients in early diagnosis.

The paper concludes that the proposed CNN model and the Skinvy application can significantly reduce the time and cost of skin disease diagnosis, especially in areas with limited access to dermatologists [7]. The authors recommend further research to enhance the model's accuracy, including using larger datasets and incorporating clinical data such as age, skin type, and race to improve classification across diverse populations

This paper [8] focuses on the classification of skin lesions using an ensemble learning approach with deep convolutional neural networks (CNNs). The authors highlight the importance of early detection of melanoma and other skin lesions for improving patient survival rates. They use the HAM10000 dataset, consisting of 10,015 dermoscopic images across seven skin lesion categories, including melanoma, basal cell carcinoma, and benign keratosis-like lesions.

The proposed approach involves an ensemble of three CNN architectures—Inception V3, Inception ResNet V2, and DenseNet 201 [8]. These architectures are combined to improve classification performance. Images in the dataset are resized and normalized, followed by data augmentation to enhance the model's generalization and reduce overfitting. The CNN models are pre-trained on ImageNet and fine-tuned for the skin lesion classification task.

The results of the individual models show that DenseNet 201 performs best with an accuracy of 97.09%, while the ensemble model outperforms the individual networks with an accuracy of 97.23% [8]. Other performance metrics, including sensitivity, specificity, precision, and F1-score, also show significant improvement with the ensemble approach, achieving 90.12%, 97.73%, 82.01%, and 85.01% respectively [8].

The paper concludes that the ensemble learning method provides superior performance in classifying seven skin lesion categories compared to single CNN architectures. However, the authors note limitations such as dataset imbalance and suggest further investigation into avoiding overfitting and exploring additional CNN architectures for future improvements.

This study focuses on developing a convolutional neural network (CNN) model for classifying skin cancer into two categories: benign and malignant. Using a dataset of 3,297 images from the Kaggle database, two CNN architectures were proposed, differing in the number of parameters. The first model had 6,427,745 parameters, achieving a classification accuracy of 93%, while the second model, with 2,797,665 parameters, reached a lower accuracy of 73% [9]. The models were trained over ten epochs, each with 200 iterations, and the training process was repeated to fine-tune the weight vectors for better accuracy.

The larger CNN architecture, due to its higher parameter count, performed better and was selected for further use. This model was then integrated into a web-based application developed using the Django framework. The application allows users to upload skin lesion images, which are processed by the CNN model to classify the lesions as benign or malignant. The classification process involves segmenting the skin lesion and extracting key features before applying the CNN model to make the final diagnosis [9].

The study highlights the importance of early detection of skin cancer to reduce mortality rates, emphasizing the need for efficient and accurate computer-aided diagnostic tools. The CNN model's ability to differentiate between benign and malignant lesions based on image analysis provides a valuable tool for dermatologists and patients alike [9]. The authors propose further improvements by developing CNN architectures with fewer parameters while maintaining high accuracy.

The study emphasizes that early detection of skin cancer is critical for reducing mortality rates, making automated diagnostic systems highly valuable. The CNN model, trained on 3,297 skin cancer images, shows promise in assisting dermatologists by accurately classifying lesions [9]. The first model, with 6.4 million parameters, provided 93% accuracy, and was integrated into a Django-based web application for public use. The application allows users to upload skin lesion images, which are analyzed to identify benign or malignant conditions. Future work aims to refine the CNN architecture, reducing parameter count while maintaining high diagnostic accuracy.

This research focuses on developing a real-time skin cancer detection model using image processing techniques. The goal was to create a machine learning-based model that classifies skin cancer as either benign or malignant, which could later be integrated into an Android application. The study utilized the SIIM-ISIC Melanoma Classification dataset, with images provided in DICOM, JPEG, and TF Record formats, and resized to 1024x1024 pixels for uniformity [10].

The model was built using TensorFlow and Keras frameworks, leveraging convolutional neural networks (CNNs) for classification. The process included data exploration, seeding, initializing the model, loading the dataset, and scaling image pixel values to fit the required input dimensions [10]. A non-trainable feature extractor was used for initial speed, followed by training and validation steps to ensure accuracy. The model was validated with training and testing datasets, showing an impressive accuracy of 97% when implemented in real-time testing via the Android application.

The app was able to detect cancer cells with high precision, differentiating between benign and malignant cells based on image inputs. The model was later converted to TensorFlow Lite (TFlite) [10] format to integrate it into the Android environment, making it suitable for real-life skin cancer detection scenarios. The project highlights the significant potential of machine learning and image processing in healthcare, specifically for early cancer detection.

# CHAPTER 3

# HAM10000 DATASET

## 3.1  Introduction to Dataset

The HAM10000 dataset, also known as the Human Against Machine with 10000 training images, is a large collection of dermatoscopic images that has become a cornerstone resource for the development and evaluation of machine learning algorithms in dermatology. It was created to support the growing need for high-quality, annotated image data for training AI models aimed at automated skin lesion analysis. With 10,015 images, the HAM10000 dataset encompasses a wide range of skin lesions, including both benign and malignant conditions, making it one of the most comprehensive datasets available in the field.

The HAM10000 dataset includes high-resolution dermatoscopic images of various skin lesions, classified into seven categories: Actinic Keratoses (AKIEC), Basal Cell Carcinoma (BCC), Benign Keratosis-like Lesions (BKL), Dermatofibroma (DF), Melanoma (MEL), Melanocytic Nevi (NV), and Vascular Lesions (VASC). Each image is accompanied by detailed metadata, including the type of lesion, the anatomical site of the lesion, and clinical diagnosis information. This extensive annotation by expert dermatologists ensures that the dataset is highly reliable and suitable for developing and testing machine learning models for skin cancer detection.

The dataset includes images from patients with diverse demographic backgrounds, ensuring a broad representation of skin types and lesion variations. The images were captured using various dermatoscopic devices, contributing to the dataset's variability in terms of image quality, resolution, and lighting conditions. This diversity is crucial for training robust AI models capable of generalizing across different clinical settings and populations.

Before using the HAM10000 dataset for training machine learning models, several preprocessing steps are typically performed to enhance the quality and consistency of the images. These steps include resizing the images to a standard dimension, normalizing pixel

values to improve contrast, and augmenting the dataset with techniques such as rotation, flipping, and scaling to increase the diversity of training samples. Additionally, segmentation algorithms may be applied to isolate the lesions from the surrounding skin, and color normalization can be used to minimize variations due to lighting conditions. These preprocessing steps help in creating a more uniform and robust dataset, which is essential for the accurate training and evaluation of AI models in skin lesion classification.

## 3.2  Skin Cancer Classification

The HAM10000 dataset encompasses a variety of skin lesions categorized into seven classes. These classes represent a spectrum of both benign and malignant conditions, each with distinct characteristics and clinical implications. Understanding and accurately classifying these types is essential for effective dermatological diagnosis and treatment planning. By providing a diverse set of lesion types, the HAM10000 dataset enables the development of robust machine learning models that can aid in the early detection and differentiation of skin cancer, ultimately improving patient outcomes.

The seven classes included in the HAM10000 dataset cover the most common and clinically significant skin lesions. Accurate classification is crucial as it directly impacts the management and treatment of patients. Misdiagnosis can lead to delayed treatment, unnecessary procedures, or missed early intervention opportunities. By leveraging advanced machine learning techniques, the aim is to enhance diagnostic precision and support dermatologists in making informed decisions, reducing the risk of error, and optimizing patient care.

The HAM10000 dataset was employed for training and validation in this study. HAM10000 is a benchmark dataset with over 50% of lesions confirmed by pathology. It consists of 10,015 dermoscopic images, categorized as follows:

(a) 6,705 Melanocytic nevi (NV) images

(b) 1,113 Melanoma (MEL) images

(c) 1,099 Benign keratosis-like lesions (BKL) images

(d) 514 Basal cell carcinoma (BCC) images

(e) 327 Actinic keratosis (AKIEC) images

 (f) 142 Vascular lesions (VASC) images

(g) 115 Dermatofibroma (DF) images

Each image in the dataset has a resolution of 600x450 pixels and is accompanied by metadata such as patient age, sex, and anatomical site of the lesion.

### a. Actinic Keratoses and Intraepithelial Carcinoma / Bowen's Disease (AKIEC)

Actinic Keratoses (AK) are rough, scaly patches on the skin that result from prolonged exposure to ultraviolet (UV) radiation. These lesions are considered precancerous because they can progress to squamous cell carcinoma (SCC) if left untreated. Actinic Keratoses typically appear on sun-exposed areas such as the face, ears, neck, scalp, chest, and hands. They are often red, pink, or flesh-colored and can be flat or raised, sometimes with a rough or wart-like surface.

Early detection and treatment of Actinic Keratoses are crucial to prevent progression to invasive squamous cell carcinoma. Actinic Keratoses serve as a warning sign for sun damage and an increased risk of skin cancer, emphasizing the importance of sun protection and regular skin checks. Treatment options for Actinic Keratoses include cryotherapy (freezing), topical medications, chemical peels, laser therapy, and photodynamic therapy.

### b. Basal Cell Carcinoma (BCC)

Basal Cell Carcinoma (BCC) is the most common type of skin cancer, originating from basal cells in the epidermis. It typically appears as a pearly or waxy bump, often with visible blood vessels, or as a flat, flesh-colored or brown scar-like lesion. BCC primarily develops on sun-exposed areas such as the face, ears, and neck. BCC grows slowly and rarely metastasizes, but it can cause significant local damage if untreated. The tumor often presents as a single lesion, and its appearance can vary, making it crucial to differentiate it from other skin conditions. Common features include a translucent or shiny appearance, visible blood vessels, and sometimes ulceration.

Diagnosis is usually confirmed through a skin biopsy. Treatment options for BCC include surgical excision, Mohs micrographic surgery, cryotherapy, and topical therapies. Early intervention is important to prevent extensive tissue damage and disfigurement.

## c. Benign Keratosis-like Lesions (BKL)

Benign Keratosis-like Lesions (BKL) are a group of non-cancerous skin growths that often resemble other types of skin conditions. They include seborrheic keratoses, solar lentigines, and lichen planus-like keratoses. These lesions are typically harmless and do not progress to cancer. BKLs usually appear as flat or slightly raised, often brown or black patches with a rough or wart-like texture. They are commonly found on sun-exposed areas of the skin, such as the face, back, and arms.

BKLs are diagnosed through visual inspection and, if needed, biopsy. They are generally managed conservatively unless they cause cosmetic concerns or irritation. Treatment options include cryotherapy, laser therapy, or topical treatments.

## d. Dermatofibroma (DF)

Dermatofibromas are benign fibrous nodules that develop in the dermis, commonly on the lower legs. They are composed of fibrous tissue and are generally harmless. Dermatofibromas are typically firm, raised, and can vary in color from pink to brown. They are often asymptomatic but can be tender or itchy. Diagnosis is primarily through physical examination, but biopsy may be needed to rule out malignancy. Treatment is usually unnecessary unless the dermatofibroma causes discomfort or cosmetic concerns; options include surgical excision or cryotherapy.

## e. Melanoma (MEL)

Melanoma is a serious form of skin cancer that arises from melanocytes, the cells responsible for pigment production. It is known for its potential to metastasize to other parts of the body.

Diagnosis is confirmed through a biopsy, and treatment typically involves surgical excision. Advanced stages may require chemotherapy, immunotherapy, or targeted therapy. Early detection is crucial for improving survival rates and reducing the need for more

aggressive treatments. AI models can enhance early diagnosis by identifying key features of melanoma in dermatoscopic images.

**f. Melanocytic Nevi (NV)**

Melanocytic Nevi, commonly known as moles, are benign skin lesions that arise from melanocytes. They are among the most common skin lesions and can vary widely in appearance. Nevi are typically brown or black, round or oval, and have well-defined edges. They can be flat or raised and are usually uniform in color. Nevi are usually benign and require no treatment unless changes occur. Monitoring is essential to detect any suspicious changes that might indicate melanoma.

Differentiating between benign nevi and melanoma is critical for appropriate management. Machine learning algorithms trained on datasets like HAM10000 can help in accurately identifying and monitoring these lesions.

**g. Vascular Lesions (VASC)**

Vascular Lesions are benign skin conditions characterized by abnormal blood vessels. Common types include hemangiomas, angiomas, and pyogenic granulomas. These lesions appear as red or purple spots and can vary in size and shape. They are often flat or slightly raised and can sometimes bleed.

Accurate diagnosis is important to ensure appropriate treatment and to distinguish these lesions from other skin conditions. AI models can assist in the accurate identification of vascular lesions from dermatoscopic images.

## 3.3  Data Preprocessing

Data preprocessing is a crucial step in machine learning that involves preparing and transforming raw data into a format suitable for model training and evaluation. This process enhances the quality of data, improves the efficiency of machine learning algorithms, and ensures that the resulting model can make accurate predictions. Below is a detailed

explanation of various data preprocessing techniques commonly used in image-based tasks, such as those in dermatological diagnosis:

The pre-processing of skin lesion images (as shown in Fig. 3.3) was conducted using Keras ImageDataGenerator. The dataset contained 57 null entries for the 'Age' attribute, which were filled using the mean filling method. Dermoscopy images in the dataset were downscaled from their original resolution of 600x450 pixels to 224x224 pixels to make them compatible with the MobileNet model. The dataset, consisting of 10,015 images, was split into a training set and a validation set with 8,912 images and 1,103 images, respectively. To maintain authenticity in the validation process, images with no duplication in the training data were selected for the validation set. Additionally, data augmentation techniques, such as random rotations, horizontal and vertical flips, zoom-in and zoom-out transformations, and brightness and contrast adjustments, were applied using Keras ImageDataGenerator to increase the diversity of the training data and prevent overfitting.
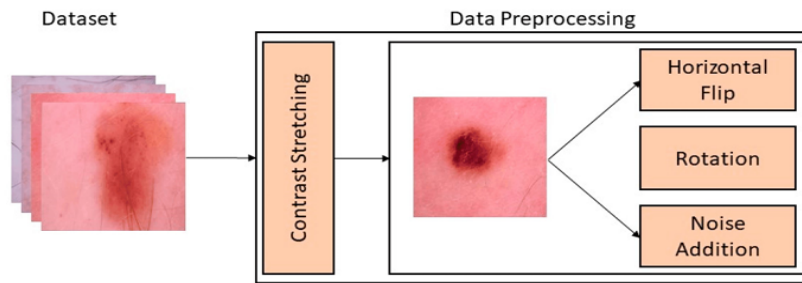


Fig. 3.3 Preprocessing of lesion image

## 3.4  Data Augmentation

Data augmentation is a technique used in machine learning and deep learning to artificially increase the size and diversity of a dataset. This is achieved by applying various transformations to the original data, creating multiple altered versions of each original data point. The goal is to enhance the model's ability to generalize to new, unseen data by exposing it to a wider variety of conditions and variations during training.

### 3.4 1  Data augmentation Techniques:

i.  **Random Rotations:** Random rotation involves rotating images by a random angle within a specified range to enhance the diversity of an image dataset. This approach improves a model's ability to recognize objects from various orientations, which helps it perform better in real-world scenarios where objects might appear in different angles.

ii. **Horizontal and Vertical Flips:** Horizontal flips create a mirror image of an image along its vertical axis, effectively reversing the left and right sides. This transformation simulates the effect of viewing the image from the opposite direction, which can help the model learn to recognize objects regardless of their lateral orientation. Vertical flips, on the other hand, mirror the image along its horizontal axis, reversing the top and bottom sections. This adjustment helps the model become more versatile in recognizing objects that may appear upside down or in different vertical arrangements. Both types of flips are valuable for training models to handle variations in object positioning and spatial orientation, enhancing their ability to generalize across diverse scenarios.

iii. **Zoom-in and Zoom-out Transformations:** Zoom-in and zoom-out transformations adjust the scale of an image to simulate variations in object size and distance. Zooming in enlarges a specific region of the image, effectively cropping and magnifying that area, which helps the model learn to recognize fine details and objects that may appear larger in different contexts. Conversely, zooming out reduces the image size, capturing a broader view of the scene and simulating distant perspectives. These transformations enhance the model's ability to handle variations in object scale and focus, making it more adept at detecting and classifying objects that vary in size and distance from the camera. By exposing the model to these scale variations, it becomes more resilient to changes in object size and improves its generalization across different scenarios.

iv. **Brightness and Contrast Adjustment:** Brightness and contrast adjustments modify the overall lightness and contrast of an image to simulate different lighting conditions and enhance image variability. Increasing brightness makes the image lighter by boosting the intensity of all pixels, which can be useful for training models to recognize objects in well-lit environments. Decreasing brightness

darkens the image, helping the model learn to identify objects in low-light conditions or shadows. Adjusting contrast changes the disparity between light and dark regions, making light areas lighter and dark areas darker, or vice versa. This adjustment improves the model's ability to distinguish objects and features under varying illumination scenarios, such as strong sunlight or dimly lit environments. By incorporating these transformations, the model becomes more adept at handling diverse lighting conditions and enhances its overall robustness and accuracy.

## 3.5  Data Splitting

Data splitting is a crucial step in machine learning model development. It involves dividing the dataset into distinct subsets, typically training, validation, and testing sets. Proper data splitting ensures that the model is robust, reliable, and performs well in real-world scenarios by enabling it to learn from one subset of data and be evaluated on another.

**Training set:** The training set is a critical component in the machine learning pipeline, as it is the primary subset of data used to build the model. This set consists of labeled examples that the model uses to learn patterns, features, and relationships within the data. During the training phase, the model iteratively processes these examples, adjusting its internal parameters to minimize errors and improve accuracy. The size and diversity of the training set are essential for the model's ability to generalize to new data; a large, varied training set helps the model to recognize a wide range of patterns and reduces the risk of overfitting, where the model performs well on the training data but poorly on new, unseen data. Data augmentation techniques, such as rotations, flips, and brightness adjustments, can be applied to the training set to artificially increase its size and diversity, further enhancing the model's learning capability. Through continuous exposure to this enriched training data, the model becomes adept at making accurate predictions and decisions based on the learned features.

**Validation set:** The validation set plays a vital role in the development of a machine learning model. It is a subset of the data that is not used for training but rather for tuning the model's hyperparameters and evaluating its performance during the training phase. By

assessing the model's performance on this separate dataset, developers can identify whether the model is overfitting or underfitting the training data. Overfitting occurs when the model learns the training data too well, including its noise and outliers, resulting in poor generalization to new data. Underfitting happens when the model is too simplistic to capture the underlying patterns of the data. The validation set helps to strike a balance by providing a basis for adjusting the model's complexity, feature selection, and other parameters to achieve optimal performance. Additionally, the validation set helps in early stopping, a technique where training is halted when the model's performance on the validation set starts to degrade, preventing overfitting. This continuous feedback loop ensures that the model is not just learning the training data but is also capable of generalizing well to unseen data, making it more robust and reliable for real-world applications.

**Splitting the Dataset:** The HAM10000 dataset, consisting of 10,015 dermoscopic images, was strategically split into training and validation sets to train and evaluate the MobileNet model effectively. The training set comprised 9,077 images (90.63% of the total dataset), while the validation set included 938 images (9.37% of the total dataset). This split ensured that the model had sufficient data to learn from while also providing a reliable set of images to evaluate its performance. Prior to splitting, the dataset underwent rigorous pre-processing and augmentation. Images were downscaled from 600x450 pixels to 224x224 pixels to meet the input requirements of the MobileNet model, and any null entries in the 'Age' attribute were filled using the mean filling method. Data augmentation techniques, such as random rotations, horizontal and vertical flips, zoom transformations, and brightness adjustments, were applied using Keras' ImageDataGenerator to enhance data diversity and prevent overfitting. This methodical approach to data splitting and augmentation contributed to the model's high accuracy and reliability in classifying various skin lesions.

The following figure 3.5 shows the data is being splitted into training and testing directories. All the seven classes are created in both the directories and the images are splitted accordingly.

DATASETS

▸ 🟥 skin-cancer-mnist-ham10000

Output (140KiB / 19.5GiB)                    ^

▾ 📁 /kaggle/working
  ▾ 📁 base_dir
    ▾ 📁 train_dir
      ▸ 📁 mel
      ▸ 📁 akiec
      ▸ 📁 bcc
      ▸ 📁 nv
      ▸ 📁 bkl
      ▸ 📁 vasc
      ▸ 📁 df
    ▾ 📁 val_dir
      ▸ 📁 mel
      ▸ 📁 akiec
      ▸ 📁 bcc
      ▸ 📁 nv
      ▸ 📁 bkl
      ▸ 📁 vasc
      ▸ 📁 df

Fig. 3.5 Splitting of dataset into training and validation sets

# CHAPTER 4

# MOBILENET ARCHITECTURE

## 4.1 Introduction

MobileNet is a convolutional neural network (CNN) architecture designed specifically for efficient execution on mobile and embedded vision applications. MobileNet, Developed by Google, achieves a good balance between accuracy and computational efficiency, making it suitable for real-time applications on devices with limited computational power. The design philosophy behind MobileNet is to build lightweight deep neural networks by utilizing streamlined architectures that perform well under constrained resources.

MobileNet's architecture is built on depthwise separable convolutions, a form of factorized convolutions that splits a standard convolution into a depthwise convolution and a pointwise convolution as shown in Figure 4.1. This factorization reduces the computational cost and the number of parameters significantly compared to traditional convolutions. MobileNet is parameterized using two global hyperparameters that effectively trade off between latency and accuracy. These parameters are the width multiplier, which scales the number of channels in each layer, and the resolution multiplier, which scales the input image size.

The architecture of MobileNet is structured to maintain high performance even with fewer computational resources. This makes it particularly useful for applications in mobile and embedded systems where computational power, memory, and battery life are at a premium. Despite its lightweight design, MobileNet achieves competitive performance on various image recognition tasks, demonstrating its effectiveness and versatility.
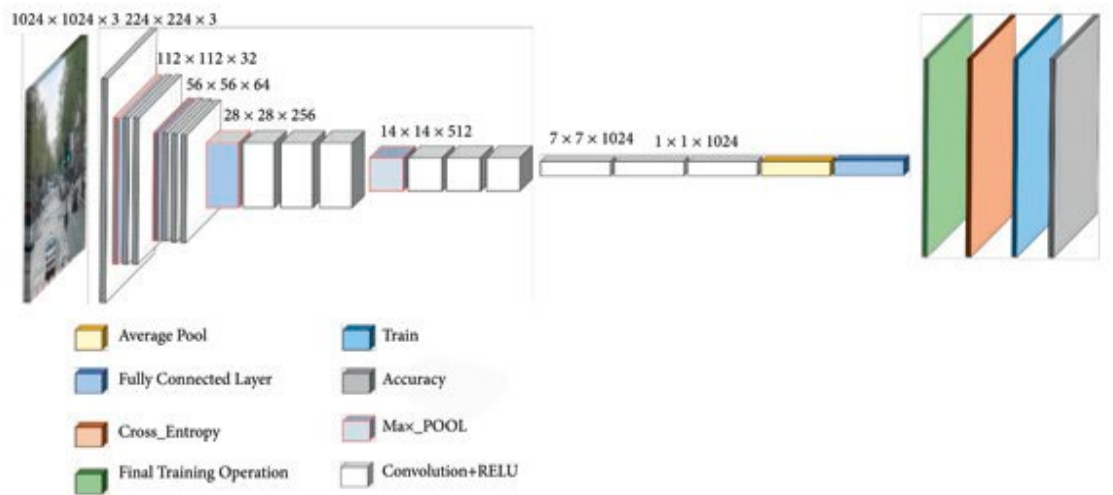
Fig. 4.1 MobileNet Architecture

## 4.2  Features

MobileNet incorporates several key features that make it highly suitable for mobile and embedded applications. These features ensure that the model remains efficient without sacrificing performance, making it an excellent choice for real-time image recognition tasks on devices with limited computational resources.

1. **Depthwise Separable Convolutions:**

   - MobileNet uses depthwise separable convolutions to significantly reduce the number of parameters and computational cost. This technique splits a standard convolution into a depthwise convolution and a pointwise convolution, leading to faster and more efficient processing.

   - By performing separate convolutions on each input channel followed by a 1x1 convolution to combine the channels, the overall computation is drastically lowered compared to standard convolutions.

2. **Width Multiplier (α):**
   - Scalability: The width multiplier, denoted as α, is a hyperparameter that scales the number of filters in each layer. It allows the model to be adjusted in size to fit specific resource constraints.
   - Control Over Complexity: By varying α, MobileNet can be made smaller or larger, balancing the trade-off between computational load and model accuracy. A smaller α results in a lighter model with fewer parameters, while a larger α increases the model's capacity and accuracy.

3. **Resolution Multiplier (ρ):**
   - Adjustable Input Resolution: The resolution multiplier, denoted as ρ, scales the input image resolution. This parameter provides another way to control the computational cost by adjusting the input size fed into the network.
   - Optimized Performance: Using a lower resolution input reduces the number of computations, making the model faster and more suitable for real-time applications, while still maintaining acceptable accuracy levels.

4. **Lightweight and Low Latency:**
   - Optimized for Mobile Devices: MobileNet is designed to operate efficiently on mobile and embedded devices, where computational resources and power are limited.
   - Real-time Applications: The architecture ensures low latency, making it suitable for applications that require quick and responsive performance, such as real-time object detection and recognition.

5. **Competitive Accuracy:**
   - High Performance: Despite its lightweight and efficient design, MobileNet delivers competitive accuracy on various benchmark datasets. It achieves this by carefully balancing model complexity and computational efficiency.
   - Versatile Applications: MobileNet can be applied to a wide range of vision tasks, from image classification to object detection and beyond, proving its robustness and adaptability.

## 4.3 Advantages

MobileNet offers several advantages that make it an attractive choice for deploying convolutional neural networks in resource-constrained environments, such as mobile and embedded devices. These advantages highlight its efficiency, flexibility, and high performance, which are crucial for real-time applications.

**Efficiency:** MobileNet is optimized for low-latency and low-power applications, making it ideal for mobile and embedded devices. The use of depthwise separable convolutions reduces the computational load and the number of parameters, resulting in faster inference times and lower power consumption. This efficiency makes MobileNet particularly suitable for real-time applications on devices with limited computational resources.

**Flexibility:** With adjustable width and resolution multipliers, MobileNet can be tailored to meet specific resource constraints and accuracy requirements. The width multiplier ($\alpha$) allows the model's size to be scaled by adjusting the number of channels in each layer, while the resolution multiplier ($\rho$) enables scaling of the input image size. This flexibility ensures that MobileNet can be optimized for a wide range of applications, balancing the trade-off between model complexity and computational efficiency.

**High Performance:** Despite its lightweight design, MobileNet performs competitively on many image recognition tasks, providing a good trade-off between accuracy and efficiency. It achieves high performance on benchmark datasets, demonstrating its ability to effectively extract and process features from input images. MobileNet's robust architecture ensures reliable and accurate predictions, making it a strong choice for various vision tasks, from image classification to object detection and beyond.

## 4.4 Model Parameters

MobileNet is designed with several key parameters that allow it to balance computational efficiency and model accuracy effectively. These parameters are crucial for tailoring the model to specific applications and resource constraints.

1. **Depthwise Separable Convolutions:** Depthwise Separable Convolutions are the core innovation in MobileNet. This operation splits the standard convolution into two parts: a depthwise convolution, which applies a single filter per input channel, and a pointwise convolution, which combines these channels. This drastically reduces the computational cost compared to standard convolutions.

2. **Width Multiplier ($\alpha$):** The Width Multiplier ($\alpha$) is a tunable hyperparameter that reduces the number of channels in each layer of the network. By applying $\alpha$, you can trade off model size and accuracy. For example, setting $\alpha = 0.5$ halves the number of channels, making the model smaller and faster, but potentially less accurate.

3. **Resolution Multiplier ($\rho$):** The Resolution Multiplier ($\rho$) scales down the input image resolution, which reduces the computational complexity and model size. This is particularly useful for deployment on devices with limited computational power. A lower $\rho$ reduces the spatial dimensions of the input, further improving efficiency.

4. **Layers and Blocks:** MobileNet is constructed using various layers and blocks. The network typically starts with a standard convolution (Conv1 layer), followed by a series of Depthwise Separable Convolution blocks. These blocks are the building units of MobileNet, where the depthwise and pointwise convolutions are combined. The network concludes with a fully connected layer for classification tasks.

5. **ReLU and Batch Normalization:** Each convolutional layer in MobileNet is followed by ReLU activation, which adds non-linearity to the model, and Batch Normalization, which stabilizes and accelerates the training process. These components are essential for maintaining model performance and efficiency.

6. **Global Average Pooling:** Before the final classification layer, MobileNet uses Global Average Pooling. This operation reduces each feature map to a single value by averaging all the spatial dimensions, significantly reducing the number of parameters in the model and thus, the risk of overfitting.

7. **Softmax Layer:** The Softmax Layer is the final layer in MobileNet, responsible for generating a probability distribution over the classes in a classification task. It

converts the network's output into probabilities that sum to one, facilitating the identification of the most likely class for each input.

8. **Dropout:** Dropout is a regularization technique used in MobileNet to prevent overfitting. It randomly drops a fraction of the units in the fully connected layer during training, which forces the network to generalize better by not relying too heavily on any single feature.

## 4.5  Layers and their Specifications:

MobileNetV1 is structured with a series of layers optimized for mobile and embedded systems. The primary design principle behind this architecture is the use of depthwise separable convolutions, which significantly reduce the computational complexity compared to traditional convolutions. Below is a breakdown of the layers in MobileNet and are summarized in table 4.5

| Type / Stride | Filter Shape | Input Size |
|---|---|---|
| Conv / s2 | $3 \times 3 \times 3 \times 32$ | $224 \times 224 \times 3$ |
| Conv dw / s1 | $3 \times 3 \times 32$ dw | $112 \times 112 \times 32$ |
| Conv / s1 | $1 \times 1 \times 32 \times 64$ | $112 \times 112 \times 32$ |
| Conv dw / s2 | $3 \times 3 \times 64$ dw | $112 \times 112 \times 64$ |
| Conv / s1 | $1 \times 1 \times 64 \times 128$ | $56 \times 56 \times 64$ |
| Conv dw / s1 | $3 \times 3 \times 128$ dw | $56 \times 56 \times 128$ |
| Conv / s1 | $1 \times 1 \times 128 \times 128$ | $56 \times 56 \times 128$ |
| Conv dw / s2 | $3 \times 3 \times 128$ dw | $56 \times 56 \times 128$ |
| Conv / s1 | $1 \times 1 \times 128 \times 256$ | $28 \times 28 \times 128$ |
| Conv dw / s1 | $3 \times 3 \times 256$ dw | $28 \times 28 \times 256$ |
| Conv / s1 | $1 \times 1 \times 256 \times 256$ | $28 \times 28 \times 256$ |
| Conv dw / s2 | $3 \times 3 \times 256$ dw | $28 \times 28 \times 256$ |
| Conv / s1 | $1 \times 1 \times 256 \times 512$ | $14 \times 14 \times 256$ |
| $5\times$ Conv dw / s1 | $3 \times 3 \times 512$ dw | $14 \times 14 \times 512$ |
| Conv / s1 | $1 \times 1 \times 512 \times 512$ | $14 \times 14 \times 512$ |
| Conv dw / s2 | $3 \times 3 \times 512$ dw | $14 \times 14 \times 512$ |
| Conv / s1 | $1 \times 1 \times 512 \times 1024$ | $7 \times 7 \times 512$ |
| Conv dw / s2 | $3 \times 3 \times 1024$ dw | $7 \times 7 \times 1024$ |
| Conv / s1 | $1 \times 1 \times 1024 \times 1024$ | $7 \times 7 \times 1024$ |
| Avg Pool / s1 | Pool $7 \times 7$ | $7 \times 7 \times 1024$ |
| FC / s1 | $1024 \times 1000$ | $1 \times 1 \times 1024$ |
| Softmax / s1 | Classifier | $1 \times 1 \times 1000$ |

Fig. 4.5 MobileNet Layers

1. **Standard Convolution Layer(Conv):**
   - **Input Size**: 224 × 224 × 3 (for typical image classification tasks)
   - **Filter Shape**: 3 × 3 × 3 × 32
   - **Purpose**: The first layer is a standard 3x3 convolution with a stride of 2, which reduces the spatial dimensions from 224 × 224 to 112 × 112 while increasing the number of channels to 32. It extracts low-level features such as edges and textures.

2. **Depthwise Separable Convolutions**:

   These are the core building blocks of MobileNet. Each depthwise separable convolution is split into two stages:
   - **Depthwise Convolution (Conv dw)**:
     - **Filter Shape**: 3 × 3 × 32 dw (depthwise filters)
     - **Stride**: 1 or 2 depending on the layer
     - **Purpose**: The depthwise convolution applies a single filter per input channel, reducing computational load by keeping the spatial filtering separate for each channel.
   - **Pointwise Convolution (1x1 Conv)**:
     - **Filter Shape**: 1 × 1 × 32 × 64 (or increasing channel numbers as you move deeper into the network)
     - **Purpose**: This layer projects the output of the depthwise convolution into a higher-dimensional space, combining the depthwise filtered outputs and adjusting the number of channels.

3. **Bottleneck Convolutions**:
   - As the network progresses, pointwise convolutions are used to increase the number of channels in some layers. For example, moving from 128 filters to 256 or 512 filters helps capture higher-level features as the spatial dimensions shrink.

4. **Global Average Pooling (Avg Pool)**:
   - **Input Size**: 7 × 7 × 1024
   - This layer pools the spatial dimensions down to a single value per channel. Instead of using fully connected layers, global average pooling ensures that the

output is a 1 × 1 × 1024 tensor (one value per channel). This is computationally efficient and reduces the number of parameters.

5. **Fully Connected Layer (FC)**:
   - **Filter Shape**: 1024 × 1000 (for 1000-class classification tasks)
   - **Purpose**: This layer maps the 1024 pooled features to the number of output classes (e.g., 1000 for ImageNet). The fully connected layer generates class scores for classification tasks.

6. **Softmax Layer**:
   - This final layer converts the raw class scores into probabilities, allowing for easy interpretation of the classification results. The highest probability corresponds to the predicted class.

# CHAPTER 5

# METHODOLOGY

## 5.1 Transfer Learning

Transfer learning is a technique in machine learning where a model developed for a particular task is reused as the starting point for a model on a second task. It is particularly useful in deep learning because it can leverage the features learned by pre-trained models on large datasets to improve performance on smaller, domain-specific datasets. In this research, MobileNet model is being used, as it uses less computational power, versatile and requires least no. of parameters.

**Key Features of MobileNet:**

- **Depthwise Separable Convolutions:** These convolutions separate the spatial and channel-wise operations, leading to a reduction in computation and model size.
- **Efficient Use of Resources:** MobileNet models are optimized for resource-constrained environments, such as mobile devices, without sacrificing accuracy.
- **Versatility:** MobileNet models are adaptable for a wide range of tasks, from image classification to object detection and segmentation.

## 5.2 Training Process

The training process involves fine-tuning the pre-trained MobileNet model to adapt it to the new task. This process includes several important concepts and parameters such as hyperparameter tuning, batch size, learning rate, and the number of epochs. Hyperparameter tuning involves adjusting the parameters that control the training process to optimize model performance. Some of the key hyperparameters include:

- **Learning Rate**: Determines the step size at each iteration while moving towards the minimum of the loss function.

- **Initial Value**: Start with a common value such as 0.001 for the Adam optimizer.
- **Adjustments**: Use techniques like learning rate schedules or learning rate annealing to dynamically adjust the learning rate during training. You might decrease the learning rate if the model's performance on the validation set starts to plateau.

- **Batch Size**: Refers to the number of training examples utilized in one iteration of model training. A larger batch size can provide a more accurate estimate of the gradient but requires more memory. Smaller batch sizes can lead to noisier estimates but might generalize better.
  - **Initial Value**: Common starting points are 32 or 64.
  - **Adjustments**: Larger batch sizes can provide more stable updates but require more memory. Smaller batch sizes might help in achieving better generalization.

- **Number of Epochs**: Indicates the number of times the entire training dataset passes through the model during training. More epochs can lead to better learning but might cause overfitting if the model learns the training data too well.
  - **Initial Value**: Start with a reasonable number such as 10 or 20.
  - **Early Stopping**: Implement early stopping to halt training if performance on the validation set stops improving, to avoid overfitting.

- **Optimizer**: The choice of optimizer can impact the training dynamics. Adam, SGD with momentum, or RMSprop are common choices. Each optimizer has hyperparameters that can also be tuned.
  - **Choices**: Adam, SGD with momentum, RMSprop.
  - **Adjustments**: Each optimizer has its hyperparameters that might need tuning. For instance, Adam's learning rate or SGD's momentum.

## 5.3  Evaluation Metrics

Evaluation metrics are used to assess the performance of models. They help quantify how well a model's predictions match actual outcomes. Common metrics

include accuracy for classification, mean squared error for regression, and F1-score for balancing precision and recall.

### 5.3.1  Accuracy

Accuracy measures the proportion of correct predictions made by the model out of all predictions. It is calculated as:

*Accuracy = True Positive (TP)+ True Negatives(TN)/Total Number of Cases*

While accuracy provides a general sense of model performance, it may not be sufficient in imbalanced datasets where the number of benign cases significantly outweighs the number of malignant cases.

### 5.3.2  Precision

Precision, also known as positive predictive value, measures the proportion of true positive predictions among all positive predictions. It indicates how many of the positively classified cases are actually correct. It is calculated as:

*Precision = True Positives (TP)/ True Positives (TP)+False Positives (FP)*

High precision indicates a low rate of false positives, which is critical in medical diagnosis to avoid unnecessary anxiety and treatment.

### 5.3.3  Recall

Recall, or sensitivity, measures the proportion of actual positive cases that are correctly identified by the model. It is calculated as:

*Recall=True Positives (TP)/True Positives (TP)+False Negatives (FN)*

High recall ensures that most actual cases of skin cancer are detected, which is essential for early intervention.

### 5.3.4  F1 Score

The F1 score is the harmonic mean of precision and recall, providing a single metric that balances both concerns. It is particularly useful when dealing with imbalanced datasets. It is calculated as:

$$F1\ Score = 2\ x\ (Precision\ x\ Recall)/(Precision + Recall)$$

### 5.3.5  Confusion Matrix

A confusion matrix provides a detailed breakdown of the model's performance by showing the counts of true positive, true negative, false positive, and false negative predictions. It helps in understanding specific areas where the model may be making errors.

# CHAPTER 6

# IMPLEMENTATION

## 6.1  Requirements

### 6.1.1  Hardware Requirements

For the training process, we leveraged Kaggle's resources to train our MobileNet model. Below are the hardware specifications of the Kaggle virtual server used:

- **CPU (Central Processing Unit):**
    - **Type:** Intel(R) Xeon(R) CPU
    - **Cores:** At least 4 cores, ideally 8 or more for parallel processing
    - **Clock Speed:** 2.20 GHz
- **GPU (Graphics Processing Unit):**
    - **Type:** NVIDIA GPU (Tesla P100)
    - **Memory:** 16GB
    - **CUDA Cores:** 3584 CUDA Cores
- **Memory (RAM):**
    - **Capacity:** 29GB
- **Storage:**
    - **Type:** Network Drive
    - **Capacity:** 57.6 GB

The following outlines the hardware specifications of my personal computer, which served as the primary system from which we accessed Kaggle's platform to train the MobileNet model.

- **Specifications:**
    - **CPU:** 11<sup>th</sup> Gen Intel® Core™ i5-1135G7 @ 2.4GHz
    - **GPU:**
        - **Integrated:** Intel® Iris® Xe Graphics – 8GB VRAM
        - **Dedicated:** NVIDIA GeForce MX450 – 2GB

## 6.1.2  Software Requirements

## A. Libraries:

- **TensorFlow:** TensorFlow is an open-source machine learning library developed by Google. It provides a comprehensive ecosystem for building and deploying machine learning models. TensorFlow is used for defining and training the MobileNet model, performing operations on tensors, and leveraging GPU acceleration for faster computations.

- **Keras:** Keras is an open-source neural network library that provides a high-level API for building and training deep learning models. It is integrated within TensorFlow as `tf.keras`. Keras simplifies the process of building neural network layers, compiling models, and training with various optimization algorithms. It is used for creating the MobileNet model architecture and handling training processes. Keras is compatible with TensorFlow 2.x (included as tf.keras).

- **Numpy:** NumPy is a fundamental library for numerical computing in Python. It provides support for large multi-dimensional arrays and matrices, along with a collection of mathematical functions. NumPy is used for handling data preprocessing, including image data manipulation, normalization, and augmentation. It is also used for numerical operations during model training and evaluation.

- **Pandas:** Pandas is a data manipulation and analysis library for Python. It provides data structures such as DataFrames for handling structured data. Pandas is used for managing and analyzing metadata associated with the HAM10000 dataset, such as lesion type, patient demographics, and image information.

- **Matplotlib:** Matplotlib is a plotting library for creating static, interactive, and animated visualizations in Python. Matplotlib is used for visualizing data distributions, training progress (e.g., loss and accuracy curves), and evaluation results (e.g., confusion matrices).

**B. Softwares:**

- **Python:** Python is a high-level, interpreted programming language known for its readability and extensive support for libraries and frameworks. Python serves as the primary programming language for the entire project, enabling the implementation of data preprocessing, model development, training, and evaluation. Its versatility and extensive library support make it ideal for machine learning tasks.

- **Jupyter Notebook:** Jupyter Notebook is an open-source web application that allows the creation and sharing of documents containing live code, equations, visualizations, and narrative text. Jupyter Notebook is used for interactive development, experimentation, and visualization. It allows for the step-by-step execution of code, making it easier to debug and iterate on model design and training.

## 6.2  Data Loading

### 6.2.1  Reading and Organizing the data

The initial phase of preparing the dataset involves accessing the HAM10000 dataset, comprised of dermatoscopic images and a corresponding CSV file (as shown in Fig. 6.1.1) containing essential metadata like lesion type, patient demographics, and image identifiers. Essential libraries, including TensorFlow for model building and training, Pandas for data manipulation, and NumPy for numerical operations, are imported to facilitate the skin cancer classification process. This metadata is parsed and managed using pandas, while the images are accessed via file paths referenced within the metadata.

Following data ingestion, the images are meticulously organized into distinct categories based on their respective lesion types. This process entails creating separate directories for each class, facilitating the storage of corresponding images within their designated folders. This structured organization significantly streamlines data loading during the model training phase.
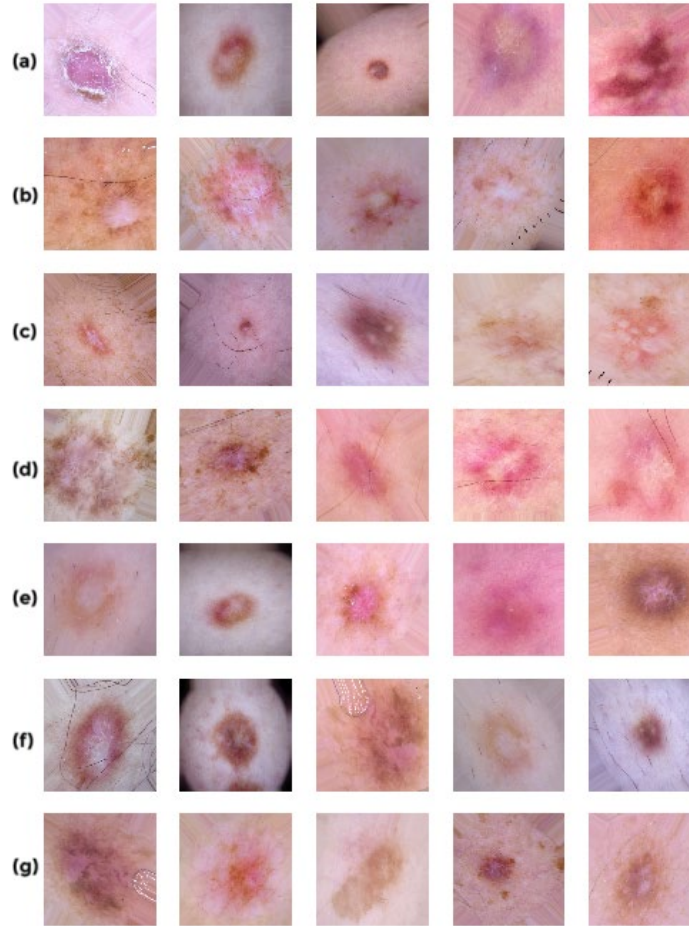
Fig. 6.1.1  Skin lesion images of HAM10000 dataset

## 6.2.2  Preprocessing Images

Image preprocessing is crucial to ensure uniform data for accurate skin cancer classification. Images are resized to a standard resolution, such as 224x224 pixels, to maintain consistency across the dataset and optimize model performance. Pixel values are normalized to a common range, like 0 to 1, to prevent biases caused by varying image brightness or contrast. Additionally, all images are converted to a standardized color format, typically RGB, for compatibility with the machine learning model. These preprocessing steps enhance the model's ability to learn and generalize from the diverse image data, ultimately improving the accuracy of skin cancer classification.

### 6.2.3  Loading images in batches

To optimize memory utilization and training speed, images are not loaded into memory all at once. Instead, they are loaded in batches during the training process. This batch loading strategy allows the model to train on smaller subsets of data at a time, significantly reducing the strain on memory resources. By processing data in manageable batches, the model can learn more efficiently and effectively, leading to faster convergence and improved overall training performance.

## 6.3  Model Building

### 6.3.1  Loading the pre-trained  MobileNet model

The model loading process begins with loading a pre-trained model architecture. Renowned for its lightweight design and efficiency, MobileNet is a convolutional neural network (CNN) that has been pre-trained on the vast ImageNet dataset, encompassing millions of images across diverse categories. This pre-training equips the MobileNet model with a strong foundation of image recognition capabilities, making it an ideal starting point for transfer learning in the context of skin cancer classification.

### 6.3.2  Initializing the metrics

The model's performance during training is tracked using various metrics. These metrics provide insights into how well the model is learning and generalizing from the training data. The chosen metrics include:

i.   **Categorical accuracy:** This metric measures the percentage of images that are correctly classified into their respective skin cancer categories. It provides an overall assessment of the model's classification accuracy.

ii.  **Top-2 accuracy:** This metric indicates the percentage of images where the true label is among the top two predicted classes. It is useful when the model's top

prediction might not always be correct, but the true label is still among the most likely predictions.

iii. **Top-3 accuracy:** Similar to top-2 accuracy, this metric measures the percentage of images where the true label is among the top three predicted classes. It provides further insight into the model's ability to rank the correct class among its predictions.

### 6.3.3 Compiling the Model

Compiling the model involves configuring it for the training process. This includes specifying the optimizer, loss function, and metrics to be tracked. In this project:

i. **Adam optimizer:** The Adam optimizer is used to update the model's weights during training. It is an adaptive learning rate optimization algorithm that has been shown to be effective in training deep learning models.

ii. **Categorical cross-entropy loss function:** This loss function is suitable for multi-class classification problems like skin cancer classification. It measures the dissimilarity between the predicted class probabilities and the true labels.

iii. **Metrics:** The metrics initialized earlier (categorical accuracy, top-2 accuracy, and top-3 accuracy) are included in the compilation step. These metrics will be calculated and reported during training to monitor the model's progress.

## 6.4 Model Training

### 6.4.1 Setting up the parameters

The training process is configured with specific parameters to optimize the model's learning and performance. These parameters include:

i. **Epochs:** The model is trained for 100 epochs, meaning it will iterate over the entire training dataset 100 times.

ii. **Early stopping:** This mechanism is implemented to prevent overfitting. If the validation accuracy does not improve for a certain number of epochs, the training process is stopped early.

iii. **Model checkpointing:** This technique saves the best-performing model weights based on validation accuracy. This ensures that the final model used for evaluation and deployment is the one that performed the best on unseen data.

iv. **Class weights:** Class weights are assigned to address the class imbalance in the dataset. In this case, the 'mel' (Melanoma) class is given a higher weight of 3.0 to make the model more sensitive to this type of skin cancer, which is often more dangerous than others.

## 6.4.2 Training Process

The training process is a crucial phase in developing the skin cancer classification model. It involves several key steps and considerations:

i. **Hyperparameter Tuning:** Hyperparameters are configuration variables that are set before the learning process begins. These include the learning rate (how quickly the model adjusts its parameters), batch size (the number of samples processed before the model is updated), and the number of epochs (iterations over the entire dataset). Careful tuning of these hyperparameters is essential to achieve optimal model performance.

ii. **Batch Size and Learning Rate:** The batch size and learning rate are critical hyperparameters. The batch size determines how many samples are processed before updating the model's parameters. A larger batch size can lead to faster training but may require more memory. The learning rate controls the magnitude of updates to the model's weights during each iteration. A high learning rate might lead to overshooting the optimal solution, while a low learning rate might result in slow convergence.

iii. **Epochs and Early Stopping:** An epoch refers to one complete pass through the entire training dataset. The model is trained for a specified number of epochs,

typically 100 in this project. However, to prevent overfitting (where the model performs well on training data but poorly on unseen data), early stopping is employed. This technique monitors the model's performance on a validation set and stops training if the performance does not improve for a certain number of epochs.

iv. **Model Checkpointing:** Model checkpointing is a practice where the model's weights are saved at regular intervals during training. This allows for resuming training from a previous checkpoint if needed and ensures that the best-performing model weights are preserved. In this project, the model checkpointing mechanism saves the weights that achieve the highest validation accuracy.

v. **Class Weights:** The HAM10000 dataset exhibits class imbalance, meaning some types of skin cancer are more prevalent than others. To address this, class weights are assigned during training. The 'mel' (Melanoma) class is given a higher weight of 3.0 to emphasize its importance and encourage the model to learn its features more effectively. This helps prevent the model from being biased towards the majority classes.

vi. By meticulously adjusting these training parameters and employing techniques like early stopping and model checkpointing, the model's performance is optimized, ensuring it learns effectively from the data while avoiding overfitting and maintaining a focus on accurately classifying all seven types of skin cancer.

## 6.4.3 Training Performance

The training process was monitored using loss and accuracy curves, providing insights into the model's performance over time. The loss curve illustrated the decrease in the loss function's value during training and validation, indicating the model's improving predictions. The accuracy curve tracked the model's accuracy on both training and validation datasets, showing an increase in predictive performance. The goal was for the training and validation curves to converge, signifying that the model was neither overfitting nor underfitting the data.

**A. Categorical Accuracy**

The plot of categorical accuracy (shown in Fig. 6.3.1) shows a steady increase in training accuracy over the epochs, indicating that the model is effectively learning and improving its classification capabilities. The validation accuracy also trends upward, though with some fluctuations, suggesting that while the model is generalizing well to unseen data, there might be some overfitting or noise in the validation set. This consistent improvement in accuracy metrics underscores the model's robustness in learning from the training data and its potential effectiveness in real-world applications.
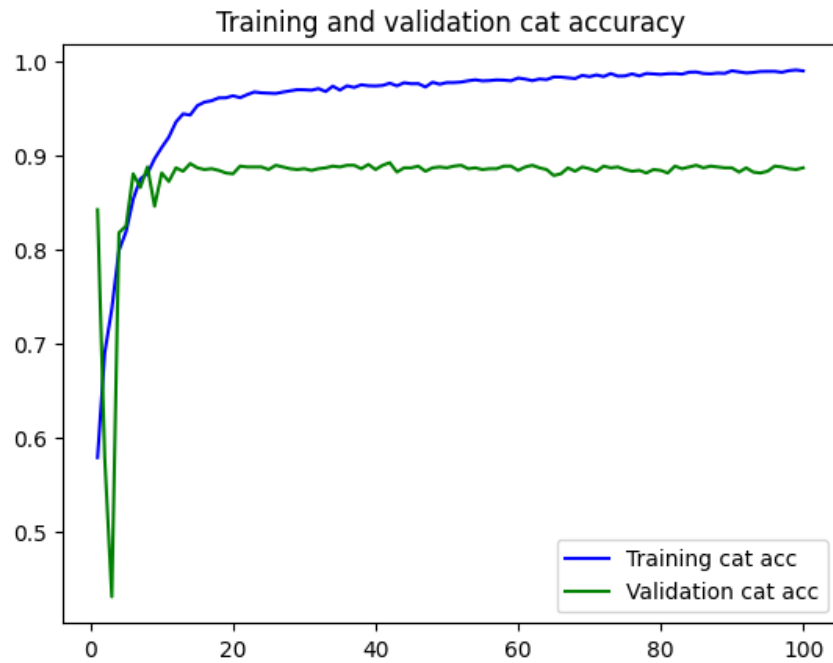


Fig. 6.3.1  Categorical accuracy of training and validation

## B.  Top2 Accuracy

The top-2 accuracy plot (shown in Fig. 6.3.2) shows a positive trend, with both training and validation top-2 accuracy increasing over the epochs. This indicates that the model is not only improving in its primary predictions but also in its ability to include the correct label within its top two guesses. This metric is valuable in scenarios where near-correct predictions are useful, demonstrating the model's practical utility in providing reliable outputs even if the exact label isn't always predicted.
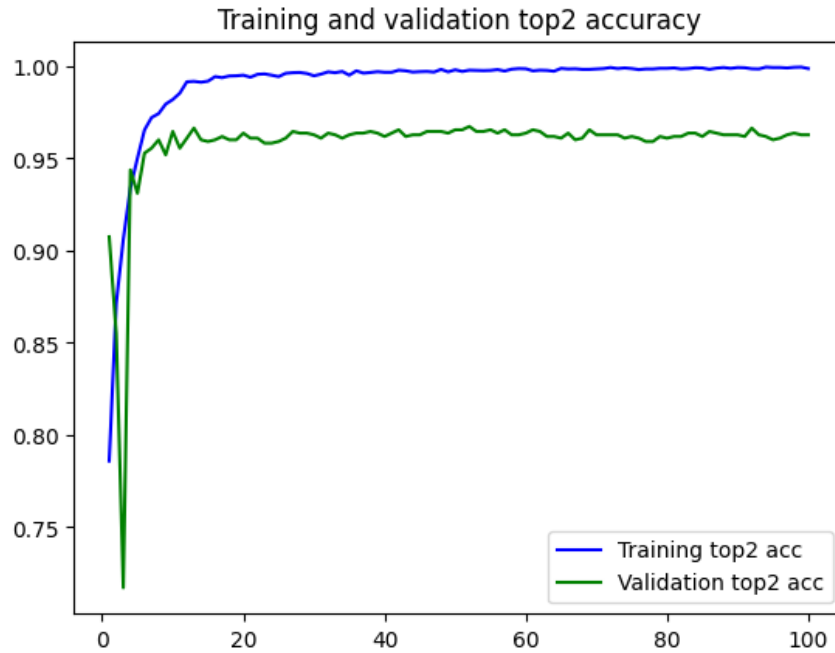
Fig. 6.3.2  Top2 accuracy of Training and validation

## C.  Top3 Accuracy

Similarly, the top-3 accuracy plot (shown in Fig. 6.3.3) displays a steady increase in both training and validation metrics, reflecting the model's competence in having the correct label within its top three predictions. This higher leniency measure indicates the model's robustness and reliability in handling complex or ambiguous inputs, making it highly applicable in multi-class classification tasks where several classes may be closely related.

## D.  Loss

The loss plot shown in Fig. 6.3.4 reveals a decreasing trend in training loss, demonstrating that the model is successfully minimizing the loss function and enhancing its learning efficiency. The validation loss initially decreases but may level off or slightly rise in later epochs, hinting at potential overfitting. This pattern highlights the importance of balancing model training to ensure it not only performs well on the training data but also generalizes effectively to new, unseen data. Monitoring the loss metrics is crucial for determining the optimal point to stop training and to apply regularization techniques to maintain the model's performance.
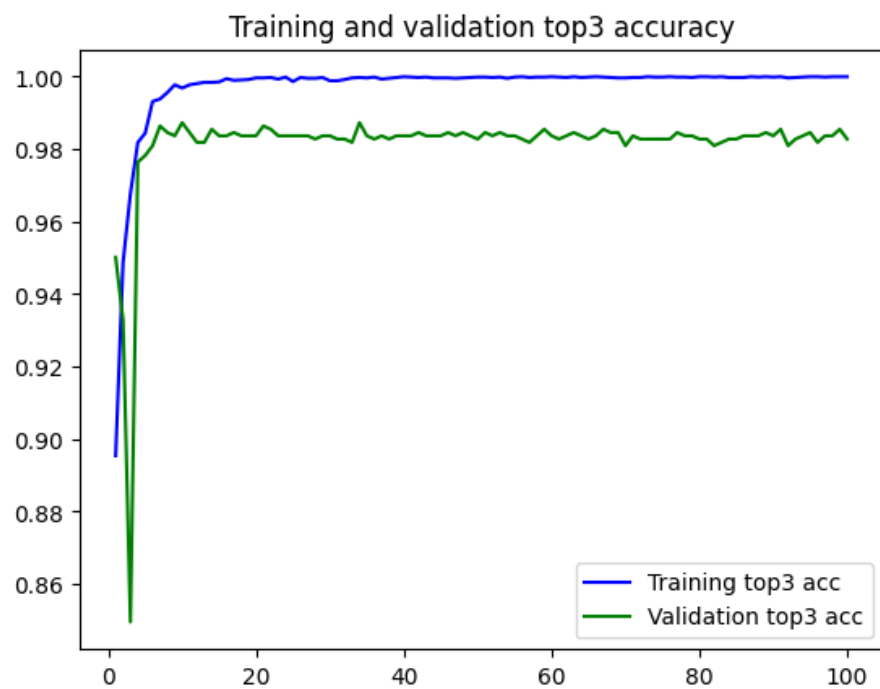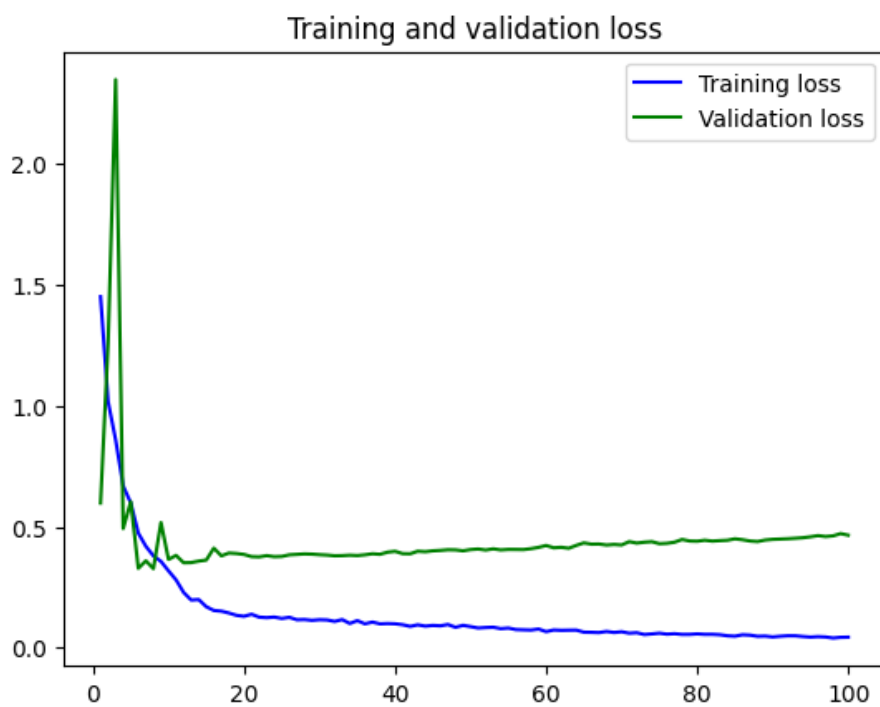
Fig. 6.3.3  Top3 accuracy of Training and validation



Fig. 6.3.4  Training and validation loss

## 6.5 Model Deployment

Model deployment is a crucial step in bringing machine learning models from the development phase to production, making them accessible for real-world applications. In this project, the skin lesion classification model is deployed using GitHub Pages, providing a web-based interface for users to interact with the model and get predictions. The deployment environment involves using GitHub Pages, a static site hosting service that takes HTML, CSS, and JavaScript files directly from a GitHub repository and publishes a website.

### 6.5.1 Setting up the Environment

A. Creating a GitHub Repository

To get started with deploying our Skin Lesion Analyzer project using GitHub Pages, we need to create a GitHub repository. After logging in, click the "New" button located next to the list of repositories, or go directly to https://github.com/new. Enter a suitable name for the repository, such as Skin Cancer Classification. Optionally, we can add a brief description to outline the purpose of your project. Additionally, check the box to initialize the repository with a README file, and choose an appropriate license, such as the MIT License. Finally, click the "Create repository" button to complete this step.

B. Add Necessary Files

With our repository created, the next step is to add the essential project files. Begin by cloning the newly created repository to our local machine using the following command:

```
git clone https://github.com/mr-anjaneyam/SkinCancerClassification
```

Navigate to the cloned repository directory on our local machine using:

```
cd SkinCancerClassification
```

Now, add the project files (index.html and script.js) to this directory. Once the files are in place, stage the changes by running:

```
git add index.html script.js
```

Next, commit the changes with a descriptive message that explains what has been added:

```
git commit -m "Add initial project files"
```

Finally, push the changes to GitHub using:

```
git push origin main
```

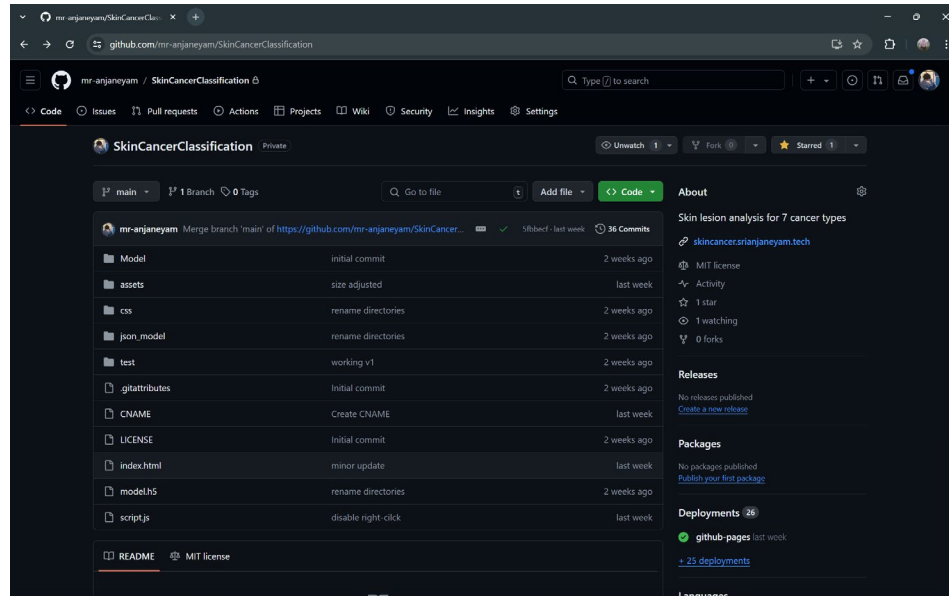The following Figure 6.4.1 shows the successful upload of project files in GitHub after the latest commit.



Fig. 6.4.1  Our repository "SkinCancerClassification" with uploaded files

## 6.5.2  Creating the HTML Structure

The HTML structure of our Skin Lesion Analyzer website forms the foundation upon which all functionalities are built. The first step involves creating an index.html file that serves as the primary webpage for our project. This file includes the basic HTML5 template, which consists of the <!DOCTYPE html> declaration, the <html> tag, and the <head> and <body> sections.

Within the <head> section, meta tags are added for specifying the character encoding (UTF-8) and ensuring responsive design via the viewport settings. The <title> tag is used to set the title of the webpage, which appears on the browser tab. Additionally, the Tailwind CSS CDN link is included to facilitate styling with Tailwind CSS classes.

The <body> section comprises three main parts:

1. **Header:** Contains a <header> element with a title for the webpage, styled using Tailwind CSS classes to create an attractive header.

48

2. **Main Content:** Enclosed within the <main> tag, this section includes two primary subsections:

  o **Upload Section:** This section, identified by the upload-section ID, features a file input (<input type="file">) for users to upload images. The input is styled with Tailwind CSS classes to enhance its appearance.

  o **Result Section:** Initially hidden (using the hidden class), this section becomes visible upon image upload. It contains a div for displaying the uploaded image preview and another div for showing the analysis results.

  o **About Section:** This section was designed to introduce the team behind the project. This section not only adds a personal touch to the website but also provides credibility by showcasing the people who developed it.

3. **Footer:** The <footer> element provides a simple copyright notice, styled to match the header for visual consistency.

The HTML structure ensures a clean and organized layout, making it easier to integrate functionality and style the website effectively.

## 6.5.3  Implementing the JavaScript logic

The JavaScript logic is responsible for adding interactivity to the Skin Lesion Analyzer website. The logic is implemented in a separate script.js file, which is linked at the end of the index.html file to ensure it loads after the HTML content.

The JavaScript code starts by adding an event listener for the DOMContentLoaded event, ensuring that the script runs only after the HTML has fully loaded. This prevents potential issues that could arise from trying to access DOM elements before they are available.

Key components of the JavaScript logic include:

1. **Element References:** Variables are created to reference key HTML elements: the file input (imageUpload), the image preview container (imagePreview), the result section (resultSection), and the prediction result container (predictionResult).

49

2. **Image Upload Handling:** An event listener is added to the file input element to trigger the handleImageUpload function whenever a file is selected. This function reads the selected file using a FileReader object, and upon successful reading, it calls displayImagePreview and analyzeImage functions.

3. **Image Preview Display:** The displayImagePreview function takes the image source (data URL) and sets it as the source of an <img> tag, which is then inserted into the imagePreview div. This function also makes the resultSection visible by removing the hidden class.

4. **Image Analysis Simulation:** The analyzeImage function simulates the image analysis process. In a real application, this function would involve sending the image to a backend server or running a machine learning model in the browser. Here, it uses a setTimeout function to mimic a delay and then calls displayAnalysisResult with mock prediction data.

5. **Displaying Analysis Results:** The displayAnalysisResult function updates the predictionResult div with the analysis results, including the prediction and confidence level.

By structuring the JavaScript logic in this way, the website provides a smooth user experience, allowing users to upload images and see analysis results dynamically. This approach also ensures that the code is modular and easy to maintain or extend in the future.

### 6.5.4 Deploying on GitHub Pages

After uploading your project files to the repository, you need to configure GitHub Pages to host your site. To do this, go to your repository on GitHub and click on the "Settings" tab located at the top of the repository page. Scroll down to the "GitHub Pages" section within the settings as shown in Fig. 6.4.2. Here, you will see an option to select the source for your GitHub Pages site. From the dropdown menu, select the branch you want to use for GitHub Pages (typically main). After selecting the branch, click the "Save" button to apply the changes. GitHub Pages will then automatically build and deploy your site. This process may take a few moments.

Once the deployment is complete, GitHub Pages will provide a URL where your site is hosted, such as https://mr-anjaneyam.github.io/SkinCancerClassification. If you own a domain, you can add your domain in the "Add a custom Domain" section. In my case, it is https://skincancer.srianjaneyam.tech. You can now visit this URL to see our deployed project as shown in Fig. 6.4.3
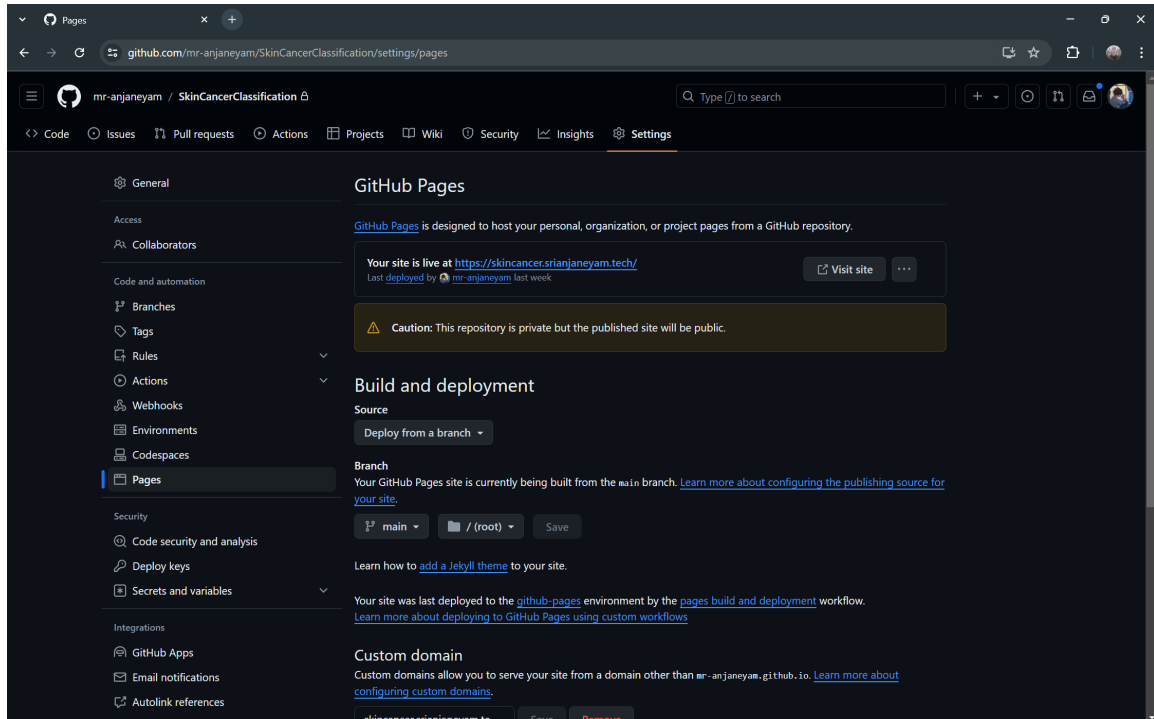


Fig. 6.4.2  GitHub pages and custom domain

Additionally, two pages 'About Model' and 'About us' sections in the page are created for description about model and own credits to the creators and their social media links as shown in Figures 6.4.4 and 6.4.5.
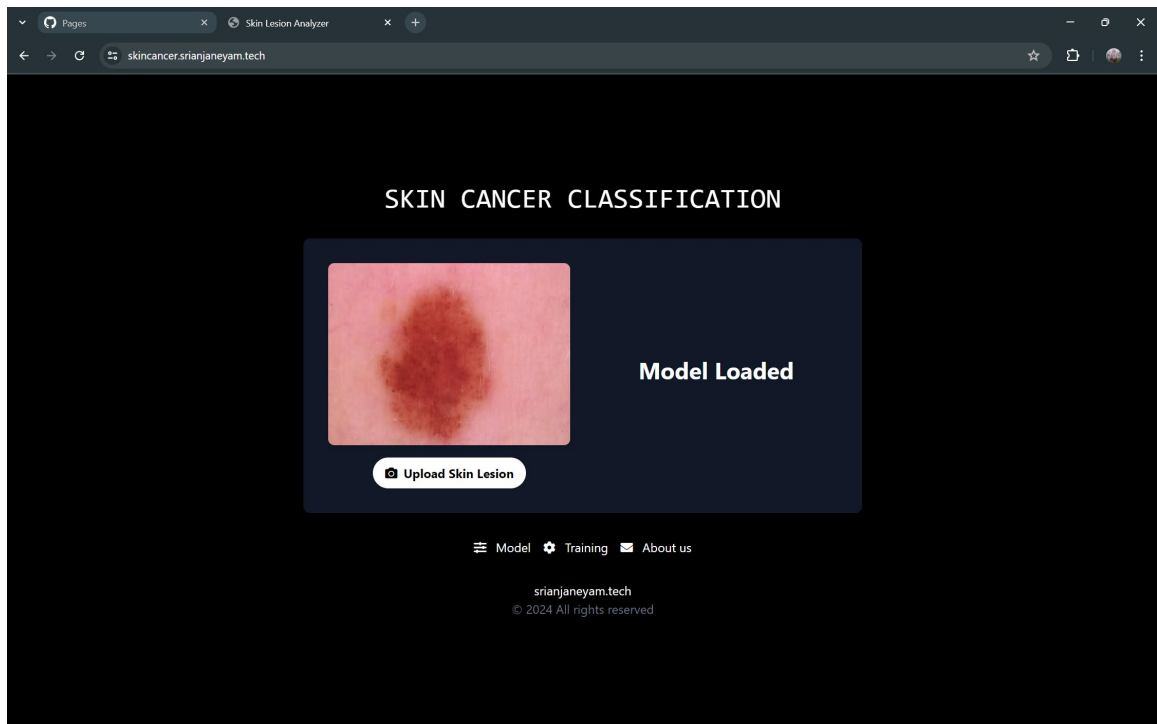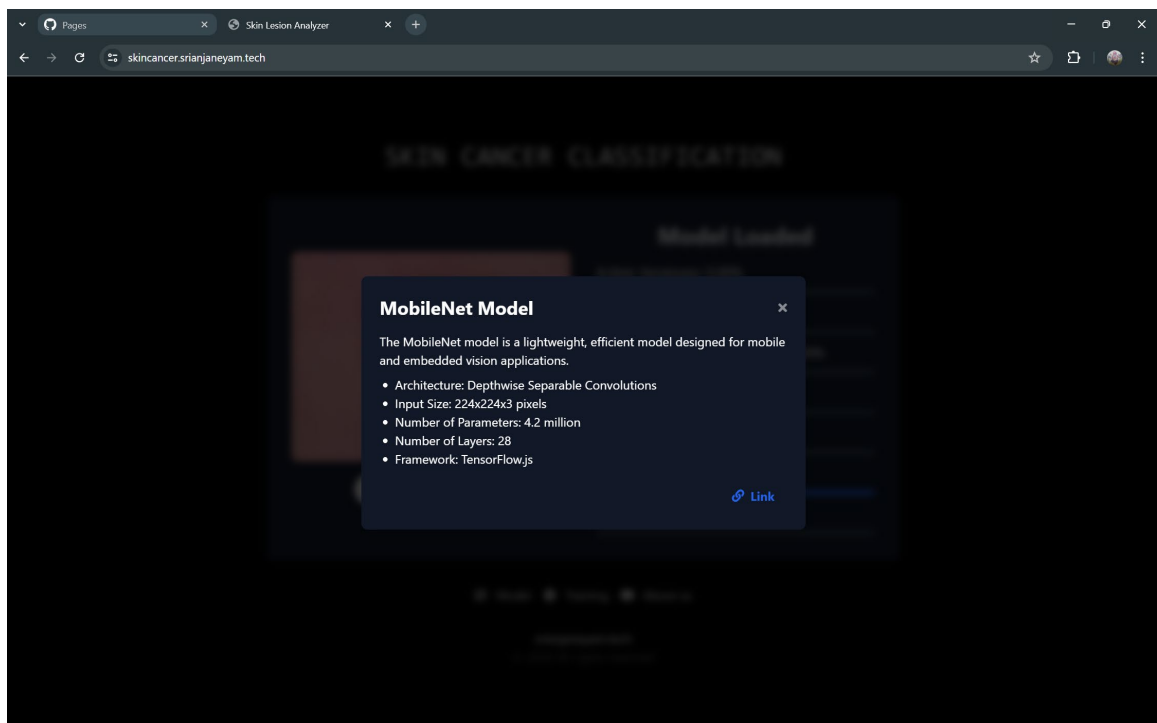
Fig. 6.4.3  The project is live
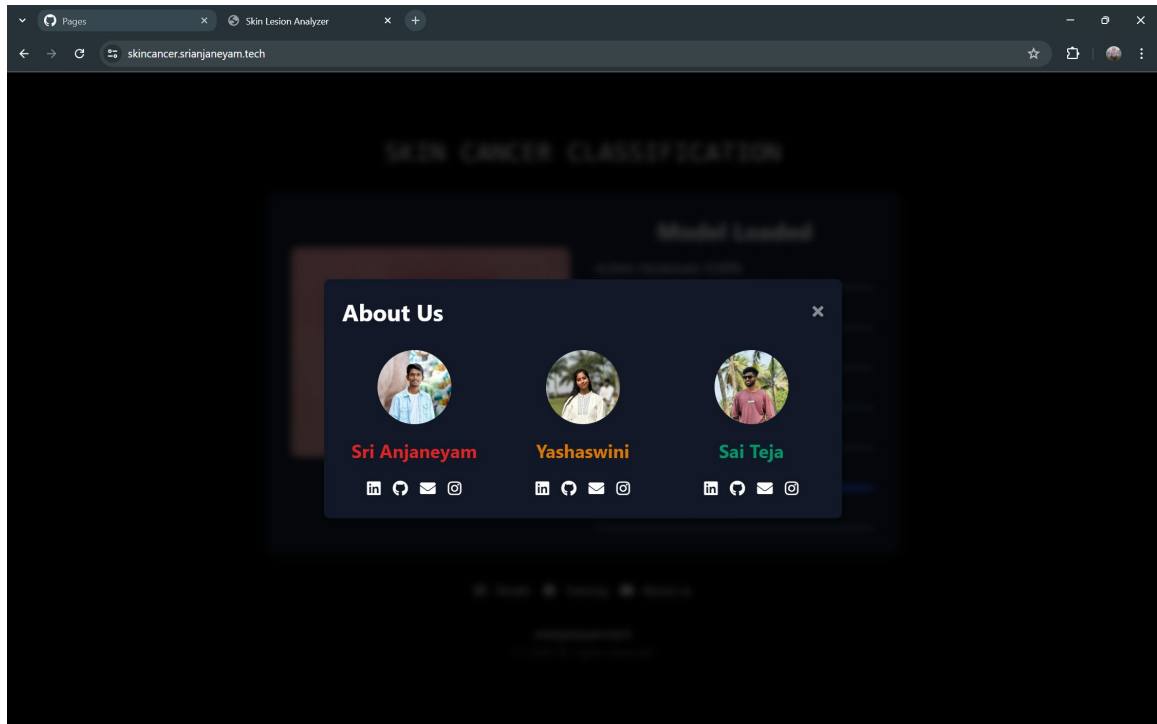


Fig. 6.4.4  About model section

Fig. 6.4.5  About us section

The following figure 6.4.6 depicts the process flow diagram illustrating the sequence of steps followed in building and training the model, Evaluating the model, Testing and deploying the model.
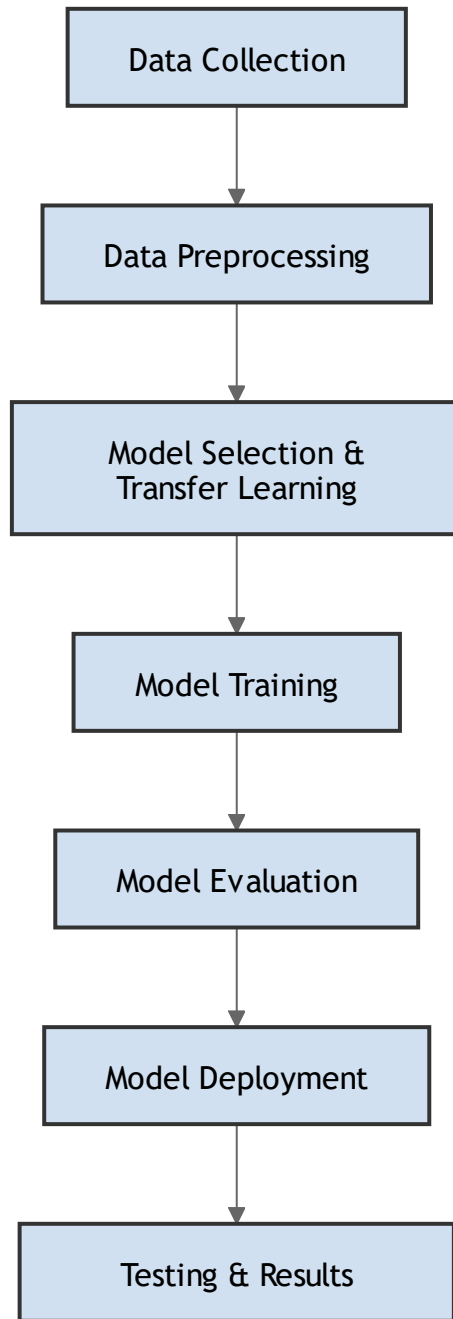
Fig. 6.4.6  Process flow diagram

# CHAPTER 7

# TESTING

## 7.1 Model Evaluation

The model's performance was evaluated on a validation set of 1103 images that were not used during the training process. The evaluation metrics used were micro and weighted averages for precision, recall, and F1-score.

**Weighted Average:** This metric considers the number of samples in each class, giving more weight to classes with more samples. The model achieved a weighted average of 89% for precision, recall, and F1-score. This indicates that the model performs well overall, especially for classes with a larger number of samples.

**Micro Average:** This metric calculates the metrics globally by counting the total true positives, false negatives, and false positives. The model achieved a micro-average of 76% for precision, 67% for recall, and 70% for the F1-score. This suggests that the model's performance is not uniform across all classes and may struggle with classes that have fewer samples.

In comparison to previous studies, the model showcased superior performance in terms of both accuracy and recall, surpassing other computer-aided diagnosis systems. This enhanced performance is attributed to the efficiency and lightweight nature of the MobileNet architecture, which facilitates rapid processing and accurate classification of skin lesions.

## 7.2 Cross Validation

To further assess the model's ability to generalize to unseen data, cross-validation was performed on the same validation set of 1103 images. The results of the cross-validation were consistent with the initial evaluation, with a weighted average of 89% for precision,

recall, and F1-score. This consistency indicates that the model is robust and not overfitting to the training data.

The model's performance varied across different classes. It achieved the highest precision, recall, and F1-score for Melanocytic Nevi, demonstrating its ability to accurately identify this type of skin cancer. However, it faced challenges in distinguishing Benign Keratosis, indicating an area for potential improvement.

Overall, the model evaluation and cross-validation results demonstrate the model's effectiveness in classifying skin lesions into seven different types of skin cancer. The high weighted average scores indicate good overall performance, while the micro-average scores suggest that the model could be improved for classes with fewer samples. The consistent results from cross-validation further support the model's potential for real-world application in assisting dermatologists with skin cancer diagnosis.

## 7.3 Hyperparameter Tuning

Hyperparameter tuning is a crucial step in optimizing the model's performance. It involves experimenting with different values for the learning rate (how quickly the model learns) and batch size (the number of samples processed before updating the model). The goal is to find the optimal combination of these hyperparameters that leads to the best performance on the validation set. In this project, the Adam optimizer was used with an initial learning rate of 0.01. The learning rate was adjusted during training using a technique called ReduceLROnPlateau, which reduces the learning rate by a factor of 0.5 if the validation accuracy does not improve for two consecutive epochs. The batch size was set to 10.

# CHAPTER 8

# RESULTS

## 8.1  Validation and Testing

The model was validated on a separate set of 1,103 images from the HAM10000 dataset. The performance metrics used to evaluate the model included categorical accuracy, top-2 accuracy, top-3 accuracy, precision, recall, and F1-score.

As a part of user testing, we've uploaded a sample skin lesion image and it is being analyzed (as shown in Fig. 8.1.1). The results are shown in percentages of seven classifications, considering the type which has the maximum percentage. In Fig. 8.1.2, The melanocytic nevi has the maximum percentage, hence the lesion belongs to that respective class.



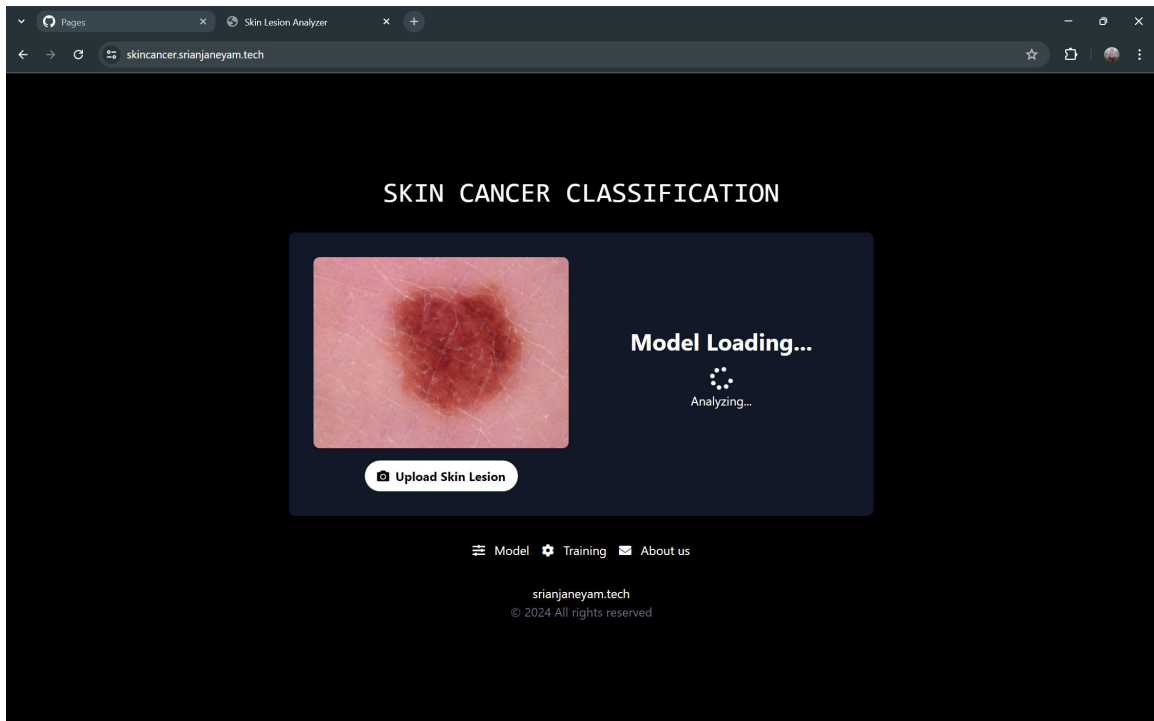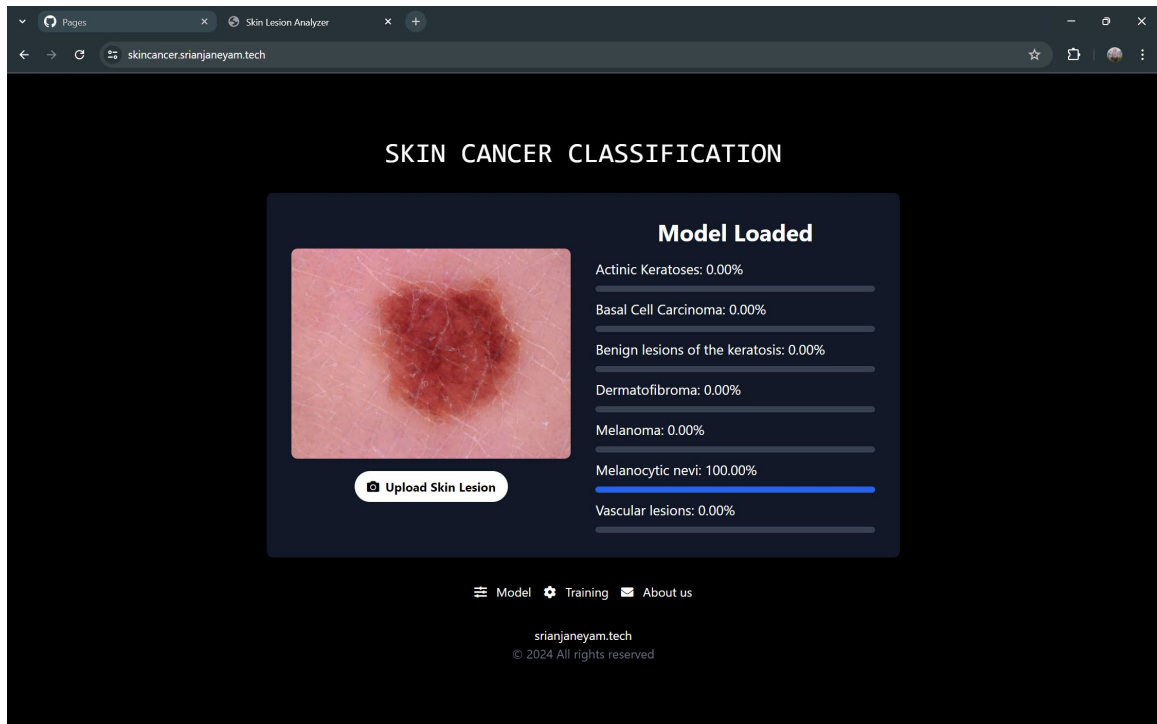Fig. 8.1.1  Process of Lesion being analyzed

Fig. 8.1.2  Results of uploaded lesion

## 8.2  Performance metrics

Following are the overall Accuracy metrics:

**Categorical Accuracy:** 89.21%

**Top-2 Accuracy:** 96.55%

**Top-3 Accuracy:** 98.45%

A detailed classification report is provided in the following Table 8.2, showing the precision, recall, and F1-score for each of the seven skin cancer classes.

| Class | Precision | Recall | F1-Score |
|---|---|---|---|
| Actinic Keratosis | 0.58 | 0.47 | 0.52 |
| Basal Cell Carcinoma | 0.68 | 0.86 | 0.76 |
| Benign Keratosis | 0.67 | 0.61 | 0.64 |
| Dermatofibroma | 1.00 | 0.50 | 0.67 |
| Melanoma | 0.47 | 0.46 | 0.46 |
| Melanocytic Nevi | 0.95 | 0.96 | 0.96 |
| Vascular Lesions | 1.00 | 0.85 | 0.92 |
| **Micro Average** | 0.76 | 0.67 | 0.70 |
| **Weighted Average** | 0.89 | 0.89 | 0.89 |

Table 8.2  Classification Report for Each Skin Cancer Class

## 8.3  Confusion Matrix

The confusion matrix (Figure X) illustrates the performance of the model in correctly classifying each of the seven skin cancer types. The diagonal elements represent the number of correct predictions, while the off-diagonal elements indicate misclassifications.
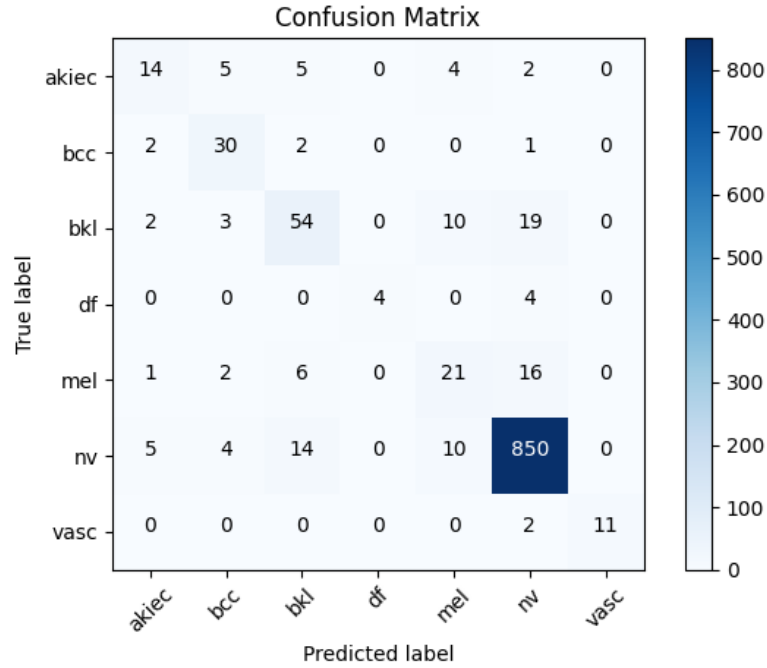
Fig. 8.3  Confusion Matrix

## 8.4  Comparison with previous studies

The performance of the MobileNet model in this study was compared with previous studies that employed different CNN architectures for skin cancer classification. Table X presents a summary of the comparison, highlighting the superior accuracy and recall achieved in this study.

| Study | Model | Classes | Accuracy (%) | Recall (%) |
|---|---|---|---|---|
| [1] | Multi-track CNN | 10 | 75.1 | - |
| [2] | Various CNN Models | 9 | 48.9-55.4 | - |
| [3] | InceptionResNetV2 | 7 | 70.0 – 76.0 | 67.0 – 76.0 |
| [4] | MobileNet | 7 | 83.15 – 95.84 | - |
| This Study | MobileNet | 7 | 89.21 | 89.0 |

Table 8.4  Comparison with previous studies

The results demonstrate that the fine-tuned MobileNet model provides a robust and efficient tool for multiclass skin cancer classification. The high accuracy, precision, recall, and F1-scores indicate the model's reliability and effectiveness in distinguishing between different types of skin lesions. These findings underscore the potential of the MobileNet model as an assistive technology in dermatological diagnosis, promoting early detection and timely treatment of skin cancer.

# CHAPTER 9

# CONCLUSION

This project aimed to develop an efficient machine learning model for multiclass skin cancer classification using the MobileNet architecture fine-tuned on the HAM10000 dataset. The dataset, comprising 10,015 dermoscopic images of various skin lesions, provided a comprehensive foundation for training and evaluating the model. By leveraging the pre-trained MobileNet model, fine-tuned to the specific nuances of skin cancer images, we achieved a categorical accuracy of 89.21% on the validation set. Additionally, top-2 and top-3 accuracies reached 96.55% and 98.45%, respectively.

The detailed classification report showed high performance in identifying melanocytic nevi and vascular lesions, with F1-scores of 0.96 and 0.92. However, performance was lower for melanoma and actinic keratosis, indicating areas for improvement. The confusion matrix revealed that most misclassifications occurred between visually similar classes, highlighting challenges in distinguishing between certain skin lesions.

ROC curves and AUC values for each class further validated the model's discriminative power, with the micro-average ROC curve demonstrating strong overall performance and an AUC close to 1.0. This suggests excellent capability in distinguishing between positive and negative classes across all skin cancer types.

Compared to previous studies using various CNN architectures, our fine-tuned MobileNet model showed superior performance. Previous models reported accuracies ranging from 48.9% to 76.0%, whereas our model achieved 89.21%, highlighting advancements in model architecture and fine-tuning techniques.

The findings emphasize the potential of fine-tuned MobileNet models in assisting dermatologists with early detection and accurate diagnosis of skin cancer. Future work should focus on integrating advanced techniques like ensemble learning, exploring other state-of-the-art architectures, and incorporating patient metadata for a more holistic approach. Collaboration with dermatologists for clinical validation will be crucial in translating these findings into real-world applications.

# CHAPTER 10

# FUTURE SCOPE

The proposed skin cancer classification model, based on the MobileNet architecture, holds significant potential for integration into clinical workflows. By incorporating the model into decision support systems, dermatologists can benefit from real-time assistance in diagnosing skin lesions. This could not only enhance diagnostic accuracy but also improve efficiency, particularly in healthcare settings where resources are limited or access to expert opinions is constrained.

MobileNet's lightweight design makes it suitable for deployment on mobile and wearable devices, enabling continuous monitoring of skin lesions. This opens the door to the creation of portable, real-time diagnostic tools that can be used by patients and healthcare providers alike. In particular, such applications could be invaluable for patients in remote or underserved areas where access to healthcare is limited. Future efforts can focus on optimizing the model for real-time performance on edge devices, ensuring that skin cancer detection becomes accessible and immediate.

A key area for future improvement lies in expanding the diversity of the training dataset. Currently, the model is trained on the HAM10000 dataset, which, while robust, may not cover the full spectrum of skin types and lesion variations seen globally. By incorporating more diverse skin tones, geographical variations, and lesion types into the dataset, the model's generalization ability can be significantly enhanced, ensuring it performs well across different populations and demographics.

Additionally, integrating the model into real-time diagnostic systems could provide immediate feedback to both clinicians and patients, improving the likelihood of early detection and timely intervention. The incorporation of explainable AI (XAI) techniques would also be beneficial in making the model's decision-making process more transparent. Dermatologists would be able to understand why a particular lesion was classified in a certain way, which would enhance trust in the system and its clinical adoption.

Further advancements could involve incorporating multi-modal data, such as patient history, genetic factors, or environmental influences, to improve diagnostic precision. This approach could lead to more personalized treatment plans and help identify high-risk patients with greater accuracy.

To ensure the model's readiness for widespread use in clinical settings, clinical trials will be needed to validate its performance. Achieving regulatory approval, such as certification from the FDA, would be a critical step in translating this research into a practical diagnostic tool for real-world healthcare environments.

# REFERENCES

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

[2]    K. Alia, Zaffar A. Shaikha, A. A. Khan, Asif A. Laghari, "Multiclass skin cancer classification using EfficientNets – a first step towards preventing skin cancer," Neuroscience Informatics, vol. 2, Issue 4, no. 100034, 2021.

[3]    Saket S. Chaturvedi, Kajol Gupta, Prakash S. Prasad, "Skin Lesion Analyser: An Efficient Seven-Way Multi-Class Skin Cancer Classification Using MobileNet", doi.org/10.1007/978-981-15-3383-9_15

[4]    Sohail Manzoor, Huma Qayyum, Farman Hassan, Asad Ullah, Ali Nawaz, Auliya Ur Rahman, "Melanoma Detection Using a Deep Learning Approach", International Journal of Innovations in Science and Technology, Vol. 4, Issue-1, pp. 222-232, February 2022.

[5]    Amirreza Mahbod, Rupert Ecker, Isabella Ellinger, "Skin Lesion Classification Using Hybrid Deep Neural Networks",
DOI: doi.org/10.1109/ICASSP.2019.8683352

[6]    Vijayalakshmi M M, "Melanoma Skin Cancer Detection using Image Processing and Machine Learning", pp. 780-784, Vol. 3, Issue-4, June 2019.

[7]    Ramzi Saifan, Fahed Jubair, "Six skin diseases classification using deep convolutional neural network", pp. 3072-3082, Vol. 12, No. 3, June 2022.

[8]    R.A. Pratiwi, S. Nurmaini, D.P. Rini, Md Naufal, Annisa D, "Deep ensemble learning for skin lesions classification with convolutional neural network", pp. 563-570, Vol. 10, No. 3, September 2021.

[9]    Nur Nafiiyah, Anny Yuniarti, "A convolutional neural network for skin cancer classification", pp. 76-84, Vol. 11, No. 1, April 2022.

[10]   Sunami Dasgupta, Soham Das, Sayani Hazra Pal, "Skin Cancer Detection using Image-Processing in Real-time", pp. 271-280, Vol. 5, Issue-6, October 2021.