A Project Report

on

SKIN CANCER IDENTIFICATION USING DEEP LEARNING TECHNIQUE

Submitted in partial fulfillment of the requirements

for the award of degree of

BACHELOR OF TECHNOLOGY

in

Information Technology

by

Jatavath Anitha (21WH1A1267)

Kondapalli Padma Sree (21WH1A1270)

Chittimelli Siri Meghana (21WH1A12A2)

Trasula Manaswini (21WH1A12A5)

Under the esteemed guidance of

Dr.B.Srinivasulu

Assistant Professor

Department of Information Technology

BVRIT HYDERABAD College of Engineering for Women

Rajiv Gandhi Nagar, Nizampet Road, Bachupally, Hyderabad – 500090

(Affiliated to Jawaharlal Nehru Technological University, Hyderabad)

(NAAC ‘A’ Grade & NBA Accredited- ECE, EEE, CSE & IT)

May, 2025

DECLARATION

We hereby declare that the work presented in this project entitled **“Skin Cancer Identification Using Deep Learning Technique”** submitted towards completion of in IV year II sem of B.Tech IT at “BVRIT HYDERABAD College of Engineering for Women”, Hyderabad is an authentic record of our original work carried out under the esteemed guidance of Dr. B. Srinivasulu, Assistant Professor, Department of Information Technology.

Jatavath Anitha (21WH1A1267

Kondapalli Padma Sree (21WH1A1270)

Chittimelli Siri Meghana (21WH1A12A2)

Trasula Manaswini (21WH1A12A5)

BVRIT HYDERABAD

College of Engineering for Women

Rajiv Gandhi Nagar, Nizampet Road, Bachupally, Hyderabad – 500090

(Affiliated to Jawaharlal Nehru Technological University Hyderabad)

(NAAC ‘A’ Grade & NBA Accredited- ECE, EEE, CSE & IT)

CERTIFICATE

This is to certify that the Project report on **“Skin Cancer Identification Using Deep Learning Technique”** is a bonafide work carried out by **J.Anitha (21WH1A1267), K.Padma Sree (21WH1A1270), CH.Siri Meghana (21WH1A12A2)** and **T.Manaswini (21WH1A12A5)** in the partial fulfillment for the award of B.Tech degree in **Information Technology , BVRIT HYDERABAD College of Engineering for Women, Bachupally, Hyderabad** affiliated to Jawaharlal Nehru Technological University, Hyderabad under my guidance and supervision.

The results embodied in the project work have not been submitted to any other university or institute for the award of any degree or diploma.

Internal Guide	**Head of the Department**
Dr.B.Srinivasulu	**Dr.Aruna Rao S L**
Assistant Professor	**Professor & HoD**
Department of IT	**Department of IT**

External Examiner

ACKNOWLEDGEMENT

We would like to express our profound gratitude and thanks to **Dr. K. V. N. Sunitha, Principal, BVRIT HYDERABAD College of Engineering for Women** for providing the working facilities in the college.

Our sincere thanks and gratitude to **Dr. Aruna Rao S L, Professor & Head, Department of IT, BVRIT HYDERABAD College of Engineering for Women** for all the timely support, constant guidance and valuable suggestions during the period of our project.

We are extremely thankful and indebted to our internal guide, **Dr. B. Srinivasulu, Assistant Professor, Department of IT, BVRIT HYDERABAD College of Engineering for Women** for his constant guidance, encouragement and moral support throughout the project.

Finally, we would also like to thank our Project Coordinators **Dr. P. Kayal, Professor and Ms. K. Bharathi, Associate Professor**, all the faculty and staff of Department of IT who helped us directly or indirectly, parents and friends for their cooperation in completing the project work.

Jatavath Anitha(21WH1A1267)

Kondapalli Padma Sree(21WH1A1270)

Chittimelli Siri Meghana(21WH1A12A2)

Trasula Manaswini(21WH1A12A5)

ABSTRACT

Skin cancer, especially Melanoma, remains one of the most serious and life-threatening diseases worldwide, highlighting the need for timely diagnosis in order to develop better treatment plans and improve the chances of survival. Conventional diagnostic approaches, based on manual inspection and biopsy, frequently face issues of subjectivity, high expenses and a shortage of qualified experts, resulting in variable outcomes and late diagnoses. In recent years, Convolutional Neural Networks (CNNs) have proved useful for medical images classification by automatically extracting features and classifying images for skin cancer detection.To further promote this field, in this study we perform an extensive comparative analysis with strong CNN architectures such as ResNet-50, VGG16, EfficientNet, and EfficientNetB3. Using three different datasets, this work investigates and compares these models based on accuracy, interpretability, and computational efficiency. Moreover, Using data augmentation helps make your dataset more varied and fairness evaluation guarantees equitable predictions among different demographics.The added benefits of interpretability from this method include improved detection performance along with improving the acceptance of the model by clinicians to trust and comprehend the model output. The proposed approach is scalable and cost-effective and has the potential to reduce melanoma mortality rates significantly through earlier and more precise diagnosis, notably in resource limited settings.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVATIONS

Abbrevation	**Meaning**
CNN	Convolutional Neural Network
VGG	Visual Geometry Group
ResNet	Residual Neural Network
HAM10000	Human Against Machine with 10000 training images
ISIC	International Skin Imaging Collaboration
SVM	Support Vector Machine
MRI	Magnetic Resonance Imaging
ML	Machine Learning
AI	Artificial Intelligence
API	Application Programming Interface
IDE	Integrated Development Environment

CONTENTS

1. INTRODUCTION

Skin cancer is one of the most prevalent and serious forms of cancer globally. Among all types, melanoma poses the greatest threat due to its high potential for rapid spread and fatality. Detecting melanoma and other skin cancers at an early stage significantly increases the chances of successful treatment. Traditional diagnostic procedures involve visual inspection by dermatologists followed by biopsy, which often suffer from subjectivity, high costs, and time delays. However, these techniques often suffer from several limitations including high cost, subjectivity in interpretation, long diagnostic cycles, and dependence on highly skilled medical professionals. These constraints are especially challenging in remote or resource-limited settings, where access to dermatologists and specialized equipment is restricted. As a result, many skin cancer cases are either undiagnosed or identified too late, leading to lower survival rates and higher treatment costs.

In recent years, the deep learning techniques into the field of medical image analysis has demonstrated promising potential to overcome these challenges. Among various deep learning techniques, Convolutional Neural Networks (CNNs) are widely used for image classification tasks and have demonstrated high accuracy in identifying skin lesions from dermatoscopic images. CNNs eliminate the need for manual feature extraction by learning patterns directly from the image data. They can classify skin lesions as benign or malignant based on variations in color, texture, and structure. Several CNN architectures like ResNet-50, VGG-16, EfficientNet-B0, and EfficientNet-B3 are commonly employed to tackle such classification problems due to their ability to generalize across different image datasets.

Skin cancer diagnosis using CNNs involves multiple steps starting from data collection, preprocessing, feature extraction, training, and evaluation. High-resolution dermatoscopic image datasets such as HAM10000, ISIC, and the Melanoma Skin Cancer dataset are commonly used for training deep learning models. These datasets contain a large number of labeled skin lesion images categorized into different types including melanoma, basal cell carcinoma, and squamous cell carcinoma, making them suitable for training and evaluating deep learning models. Each image is labeled as benign or malignant, allowing the model to learn the distinguishing characteristics of each class.

Before feeding the data into the neural network, pre-processing is essential to improve model performance and generalization. Images are resized to a uniform resolution, normalized, and augmented through techniques like rotation, flipping, scaling, and brightness adjustments. Data augmentation introduces variability, reduces overfitting, and ensures that the model does not learn biased patterns. Class imbalance, a common issue in medical datasets, is addressed by augmenting minority class images to maintain an even distribution of classes during training. This is essential for developing a clinically trustworthy and ethically sound diagnostic tool that can be reliably used in diverse populations.

EfficientNet models are designed to achieve high accuracy with fewer parameters by balancing the network's depth, width, and resolution. EfficientNet-B3 uses compound scaling technique, which efficiently balances network depth, width, and resolution to optimize both accuracy and computational cost. This makes it particularly well-suited for deployment in real-time or resource-constrained environments such as rural clinics and mobile health applications. Meanwhile, ResNet-50 offers the advantage of deeper learning via residual connections, mitigating the vanishing gradient problem and allowing for more effective training. VGG16, though simpler in design, provides strong baseline performance and is known for its consistent results across various image recognition tasks. Comparative evaluation of these models helps in identifying which architecture best suits the task of skin cancer detection under various constraints.

Performance evaluation is carried out using metrics such as accuracy, precision, recall, F1-score, and confusion matrices. These metrics provide detailed insights into how well the model distinguishes between benign and malignant cases. Training and validation accuracy curves, along with loss plots, help assess the convergence behavior and overfitting tendencies of the models. Batch processing time and resource utilization are also analyzed to determine the efficiency of each architecture.

Deep learning-based classification of skin lesions aims to assist dermatologists in early and reliable diagnosis. An automated system built using optimized CNN models can significantly reduce the burden on healthcare systems, improve access to diagnostic tools in remote areas, and deliver faster and more consistent results. By leveraging large-scale dermatoscopic datasets and efficient model architectures, it becomes possible to create scalable, accurate, and cost-effective skin cancer detection systems suitable for real-world deployment.

1.1. Objective

i) To design and implement a deep learning-based system that automates the identification and classification of skin lesions from dermatoscopic images.

ii) To utilize multiple convolutional neural network architectures such as ResNet-50, VGG-16, EfficientNet-B0, and EfficientNet-B3 for accurate classification of skin cancers into benign and malignant categories.

iii) To analyze and compare the performance of these models across various datasets, including HAM10000, ISIC, and the Melanoma Skin Cancer dataset, in terms of accuracy, speed, and computational efficiency.

iv) To apply image preprocessing techniques like resizing, normalization, and data augmentation to enhance model generalization and reduce overfitting.

v) To evaluate the effectiveness of the models using metrics such as precision, recall, F1-score, and confusion matrices, ensuring fairness and robustness in predictions across different types of skin cancer.

vi) To develop a scalable, interpretable, and low-latency solution suitable for deployment in real-time clinical settings and resource-limited environments.

1.2. Problem Definition

The skin cancer identification system should accurately classify skin lesions regardless of variations in skin tone, lighting conditions, image quality, or lesion size, using deep learning models trained on dermatoscopic images. The system must be capable of distinguishing between benign and malignant lesions across different types of skin cancer including melanoma, basal cell carcinoma, and squamous cell carcinoma. The classification process involves extracting meaningful features from medical images using convolutional neural networks such as ResNet-50, VGG-16, EfficientNet-B0, and EfficientNet-B3. The image datasets used in this project include HAM10000, ISIC, and the Melanoma Skin Cancer dataset, each offering high-resolution and diverse samples. The system should maintain fairness and interpretability in predictions, ensuring reliable outcomes across all demographic groups. It must also operate efficiently in real-time, making it suitable for deployment in clinical environments and resource-constrained settings. Through the use of data augmentation and preprocessing techniques, the model should continuously improve in accuracy and generalization. The goal is to build an end-to-end scalable diagnostic tool that leverages deep learning for early and precise detection of skin cancer, ultimately supporting clinicians and improving patient outcomes.

1.3. Modules

1) Data Collection and Dataset Handling
2) Data Pre-processing
3) Data Augmentation and Class Balancing
4) Model Building and Compilation
5) Model Training and Optimization
6) Model Evaluation and Comparison
7) Result Visualization and Interpretation

2. LITERATURE SURVEY

S. S. G. K. Gautam and A. Singh [1] The purpose of this study is to evaluate the effectiveness of deep learning models in diagnosing skin cancer using a scalable, high-performing architecture. The authors use the EfficientNet model, which is known for its parameter efficiency and high accuracy, to identify different skin lesion types including melanoma, basal cell carcinoma, and benign keratoses. They utilize the HAM10000 dataset, which contains high-resolution dermatoscopic images, and apply preprocessing techniques such as noise filtering, albumentation, and normalization to enhance image quality and model trainability. Additionally, fairness metrics are introduced to ensure that the model's performance is unbiased across various demographic groups. Federated learning is also used to maintain privacy while enabling collaborative training on distributed datasets. Their results show notable improvements in classification accuracy, robustness, and computational efficiency, establishing the model as a cost-effective and scalable diagnostic tool.

N. E.-S. M. M. Diab, Amal G.Fayez [2] This research presents a self-contained automated framework for skin cancer diagnosis using Convolutional Neural Networks (CNNs). The primary goal is to accurately differentiate between benign and malignant skin lesions. The framework begins with advanced preprocessing steps such as noise reduction and normalization to improve image clarity. CNNs are then used for feature extraction, followed by an SVM classifier for final lesion classification. The approach is validated using the ISIC dataset, where it achieves accuracy rates exceeding 90%. The study further explores the impact of hyperparameter tuning and transfer learning, showing that pre-trained CNN models can significantly enhance classification performance. The authors highlight the clinical relevance of such automated systems, especially in aiding dermatologists with early detection and decision-making.

S. T. S. Sharma, K. Guleria and S. Kumar [3] Although focused on Alzheimer's disease, this study illustrates the potential of CNNs for medical image classification. Using MRI scans, the authors employ a CNN architecture based on VGG16 for early diagnosis. The model extracts complex patterns from MRI images, which are difficult to interpret manually. Despite being trained on a relatively small dataset, the model achieves high accuracy, thanks to the pre-trained feature extractor. The study emphasizes the adaptability of CNNs for different medical imaging tasks and reinforces the importance of deep learning in improving diagnostic speed and precision. These findings underline the general applicability of CNNs in healthcare beyond dermatology.

M. I. K. M. M. A. Hasan, Mohammed Rakeibulfatemi [4] This paper compares several CNN architectures, including ResNet50, VGG16, and custom networks, for skin cancer detection using a dataset of 6,594 dermatoscopic images. The study explores the influence of model depth on classification performance, noting that deeper models like ResNet50 outperform shallower ones in feature extraction and accuracy. Data preprocessing techniques such as resizing, normalization, and augmentation are shown to significantly enhance model robustness. The research also discusses the trade-off between computational cost and model accuracy, concluding that while deeper networks yield better results, they require greater computational resources. Overall, the paper highlights the importance of balancing performance and efficiency for real-world deployment.

M. I. e. a. e. a. M. Dildar, S. Akram [5] In this study, the authors compare the performance of linear regression methods with CNN-based deep learning approaches for melanoma detection. They use Apple's Core ML tools to develop a mobile-friendly real-time application. Their findings reveal that traditional linear regression models are not capable of capturing the intricate patterns in dermatoscopic images, which are essential for distinguishing between benign and malignant lesions. In contrast, the CNN model significantly reduces false negatives and generalizes well across various image types. The integration of Core ML further demonstrates the potential for deploying CNN models in portable clinical settings, making it a promising solution for rapid and reliable diagnosis.

E. Y. H. Sujaini, Herryramadhan [6] This research emphasizes the urgent need for early detection of skin cancer and explores how CNNs can be utilized to improve diagnostic accuracy and reliability.

The model is designed to address challenges such as variability in lesion appearance, differences in skin tone, and demographic diversity. By leveraging a deep learning approach, the system effectively identifies and classifies dermatoscopic images as benign or malignant. The model's robustness across varied inputs makes it suitable for practical use in clinical and teledermatology environments. The study reinforces the idea that deep learning can significantly reduce diagnostic delays and improve treatment outcomes.

S. Alshourbaji, Ibrahimarabia [7] This statistical report, prepared in collaboration with the National Cancer Registration and Analysis Service (NCRAS), presents comprehensive data on the incidence of melanoma in the UK. The report highlights melanoma as one of the fastest-growing cancers, with increased incidence linked to excessive ultraviolet (UV) exposure from sunlight and artificial sources like tanning beds. The analysis shows age and gender-based disparities, noting higher cases among older adults and a rising trend in younger populations. It stresses the importance of early detection in improving survival rates and suggests public health interventions such as awareness campaigns and technological solutions, including AI, to mitigate risk and enhance diagnosis.

3. SYSTEM DESIGN

We train and evaluate our architecture on six datasets: three general skin cancer datasets—Melan oma, Basal Cell Carcinoma, and Squamous Cell Carcinoma—and three melanoma-specific data-sets HAM10000, ISIC, and the Melanoma Skin Cancer dataset. The general datasets consist of high-resolution dermoscopic images labeled as benign or malignant. The Melanoma dataset includes 10,605 images with an 80/20 train-test split, the Basal Cell Carcinoma dataset comprises 1,249 images with a 90/10 split, and the Squamous Cell Carcinoma dataset contains 903 images, also split 90/10. These datasets provide a diverse foundation for evaluating model performance across different cancer types.

For more focused melanoma detection, we used three additional datasets. The HAM10000 dataset contains 10,605 dermoscopic images (80% training, 20% testing), the Melanoma Skin Cancer dataset has 13,880 images (60% training, 40% testing), and the ISIC dataset consists of 3,279 images (80/20 split). These datasets enable more detailed analysis of CNN performance on melanoma-specific cases. We evaluated four deep learning models—ResNet-50, VGG16, EfficientNet-B0, and EfficientNet-B3—across all six datasets. Each model was trained to classify lesions into either benign or malignant categories. EfficientNet-B3 was included due to its high accuracy with lower computational cost, making it ideal for small and resource-constrained datasets. ResNet-50 was chosen for its residual learning framework that mitigates the vanishing gradient problem, while VGG16 was selected for its efficient feature extraction, particularly effective in segmenting lesions with asymmetry, irregular borders, and color variations.

All models were implemented using TensorFlow and Keras, with optimization strategies such as learning rate scheduling, early stopping, and dropout applied during training to prevent overfitting. The datasets were split into balanced training, validation, and test sets to ensure fair evaluation. Data augmentation techniques—including rotations, scaling, and lighting adjustments—were applied to increase training variance and improve model generalization. This comprehensive system allows for comparative evaluation of different model-dataset combinations, helping identify the most effective architecture for accurate and efficient skin cancer detection. Ultimately, our work supports the development of AI-based clinical tools that can aid in early diagnosis and improve survival rates, especially in under-resourced settings.

3.1 Architecture

The proposed system for skin cancer detection begins with the input of dermoscopic images, sourced from datasets stored on platforms like Google Drive. These images undergo preprocessing and normalization to ensure uniformity in size and pixel values. The processed images are then passed through deep learning models including ResNet-50, VGG16, and two variants of EfficientNet—B0 and B3—each chosen for their strengths in accuracy and computational efficiency. After feature extraction, the system proceeds to a training and post-processing phase, where the models are further refined using techniques like fine-tuning to enhance predictive accuracy. The fine-tuned models then perform classification to determine whether a given skin lesion is benign or malignant. This pipeline is designed to deliver high diagnostic accuracy while remaining efficient and scalable for use in real-world medical applications.

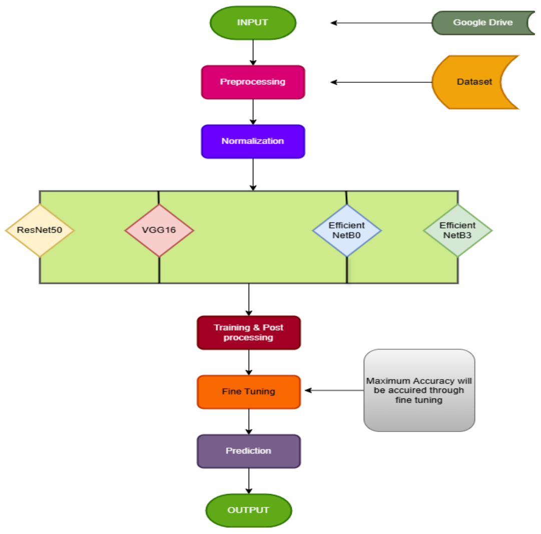

Figure 3.1.1: Project Flow

3.1.1 Dataset

The datasets employed in our project comprise dermo scopic images of skin lesions categorized into two categories, Benign and Malignant. First, we have used three datasets corresponding to three diverse skin cancer: Melanoma, Basal Cell Carcinoma, and Squamous Cell Carcinoma. Melanoma dataset have 10,605 images, with an 80% training and 20% testing split. The basal cell carcinoma dataset contains 1,249 images, partitioned into 90% for training and 10% for test ing.Squamous Cell Carcinoma dataset comprises 903 images, with 90% used for training and 10% for testing.Table 3.1.1.1 en capsulate these datasets.Evaluation also used three additional datasets: namely, HAM10000, Melanoma Skin Cancer, and ISIC. As summarized in Table 3.1.1.2, the HAM10000 dataset includes an extensive collection of 10,605 images that were split into 80% for training and 20% for testing. Melanoma Skin Cancer dataset contains 13,880 images, further divided into 60% training data and 40% testing. The ISIC dataset comprises 3,279 images, divided into 80% training and 20% testing. These datasets provide varying yet high-quality images that are essential in training deep learning models to find visual patterns in skin lesions. Figure 3.1.1.1 offers an overview of skin lesion images, providing visual insights into the dataset composition. With the proportions tuned to allow the models to evaluate unseen images, it could finally be ascertained how well or fairly a particular model performed, giving way to comparative analysis across multiple datasets.

Table 3.1.1.1: Summary of Skin Cancer Datasets

S.No	Dataset Name	No.of samples
1.	Melanoma	10605
2.	Basal Cell Carcinoma	1249
3.	Squamous Cell Carcinoma	903

Table 3.1.1.2: Summary of Melanoma Datasets

S.No	Dataset Name	No.of samples
1.	HAMM10000	10605
2.	Melanoma Skin Cancer	13880
3.	ISIC Skin Cancer	3279

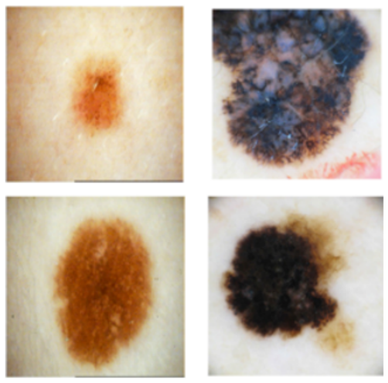

Figure 3.1.1.1: (a) Skin images of Benign (b) Skin images of Malignant

3.1.2 Data Preprocessing

The proposed system incorporates essential data preprocessing steps to make ready the skin cancer images for training the ResNet-50, VGG-16, EfficientNet-B0 and EfficientNet-B3 models. These steps ensure that the input data is structured, informative, and optimized for machine learning. The key data preprocessing steps are as follows:

- **Image Loading and Conversion:** Importing skin cancer images from melanoma cancerdataset. The image file paths and labels are read and or-ganized via Python libraries such as os, glob, and pandas. We load up the images to raw form and transform theminto arrays to be processed.

- **Dataset Shuffling:** Using pandas, the dataset is shuffled. repeat the process but take a random sample of instances during training to minimize bias in the data distribution.

- **Data Visualization:** To check for any imbalance in the dataset, class distribution analysis is done through seaborn bar plots Furthermore, certain images are visualized according to their labels to validate if the dataset is correct.

- **Dataset Splitting:** Here we used an 80-20 train test split function from sklearn. This also trains the model on diverse data and evaluates it on unseen images in Figure 3.1.2.1.
- **Image Resizing and Preprocessing:**All images are rescaled to a standard definition of 224×224 pixels to be compliant with the input specifications of prototype. Model performance and convergence are maximized through normalizing the pixel values (uses the preprocess input function of Tensor- Flow's module).
- **Data Augmentation:** We apply data augmentation using ImageDataGenerator to help tackle any overfitting and add diversity. By using augmentation techniques like flipping, the application of rotations,and image scaling, the model is presented with much more variety of images.
- **Class Balancing:** Class balancing is handled implicitly via the augmentation process, which guarantees sufficient frequency of minority classes in the dataset.
- **Batch Generation:** We use ImageDataGenerator to generate training, validation and the testing data generators. These generators augment images in batches with on-the-fly modification during training to create the training process more productive and effective.

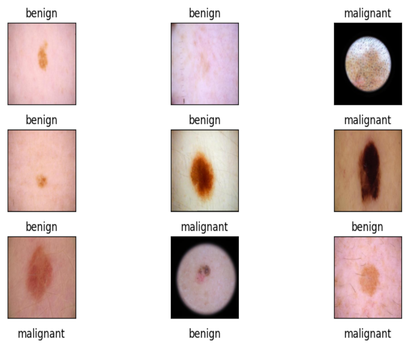

Figure 3.1.2.1: Splitting Data

3.2 Technologies

3.2.1 Deep Learning Deep learning (also known as deep structured learning or hierarchical learning) is part of a broader family of machine learning methods based on learning data representations, as opposed to task-specific algorithms. Learning can be supervised, semi-supervised or unsupervised. Deep learning architectures such as deep neural networks, deep belief networks and convolutional neural networks have been applied to fields including computer vision, medical diagnostics, natural language processing, and bioinformatics. In this project, deep learning enables the automated feature extraction and classification of skin lesion images to distinguish between benign and malignant categories with high accuracy.

3.2.2 Convolutional Neural Network

A Convolutional Neural Network (CNN or ConvNet) is a class of deep, feed-forward artificial neural networks, most commonly applied to analyzing visual imagery. CNNs are known for their ability to detect patterns in images using convolutional layers, pooling layers, and fully connected layers. These networks require minimal preprocessing and are highly effective for image classification tasks. In this project, CNN-based models such as ResNet-50, VGG16, EfficientNet-B0, and EfficientNet-B3 were used for the classification of skin cancer from dermoscopic images.

3.2.3 TensorFlow

TensorFlow is an end-to-end open-source platform for machine learning developed by the Google Brain team. It provides a comprehensive and flexible ecosystem of tools, libraries, and community resources that support the development and deployment of ML-powered applications. In this project, TensorFlow served as the foundational framework to build, train, and deploy deep learning models, enabling efficient computation and scalable experimentation.

3.2.4 Keras

Keras is an open-source high-level neural networks API, written in Python and capable of running on top of TensorFlow. It was developed by François Chollet and is designed to be user-friendly, modular, and extensible, making it suitable for fast prototyping and experimentation. In this project, Keras was used to design, train, and fine-tune CNN architectures, including EfficientNet-B0 and EfficientNet-B3, which were central to achieving high accuracy in skin cancer classification.

3.2.5 Pandas and NumPy

Pandas is a data manipulation and analysis library for Python that provides data structures like DataFrames, enabling efficient handling of structured datasets. NumPy is a fundamental package for scientific computing with Python, offering support for large multi-dimensional arrays and matrices. Together, they were used in this project to manage image paths, organize labels, and perform numerical operations like reshaping and normalization of input data.

3.2.6 Matplotlib and Seaborn

Matplotlib is a plotting library for Python that enables the generation of static, interactive, and animated visualizations. Seaborn is built on top of Matplotlib and provides a high-level interface for drawing attractive and informative statistical graphics. These tools were used in this project to visualize dataset distributions, training metrics such as accuracy and loss, and sample outputs.

3.2.7 ImageDataGenerator

ImageDataGenerator is a Keras class that allows real-time data augmentation during model training. It was used to apply transformations such as rotations, flips, zooms, and shifts to the training images, increasing dataset diversity and helping prevent model overfitting.

3.3 Software and Hardware Requirements

Hardware

i) Operating System: Windows 10

ii) Processor - Intel Core i7

iii) Memory(RAM) - 8 GB

iv) Hard disk: 1TB

Software

i) Python 3.5 and later versions

ii) IDE: Visual Studio(VS Code)

4. METHODOLOGY

4.1 Algorithm

The algorithm has been developed using some deep learning models to correctlydetect and arrange skin lesions on high- resolution images as benign or malignant. First, dermoscopic images are gathered and preprocessed to be compatible with themodels selected. This includes resizing, normalization, and augmentation techniques thatcombat class imbalance and help with model robustness. The processed imagesare then fed to different deep learning architectures which were fine-tuned for skin cancer detection and classification.

- **EfficientNet-B0:** EfficientNet-B0 is a scalable and efficient convolutional neural network (CNN) designed for resource-constrained environments like mobile and embedded devices. It achieves a balance between high performance and low computational cost by leveraging a compound scaling method to uniformly scale network depth, width, and resolution.

  The model takes an input image of size 224x224 with three channels. It begins with a 3x3 convolution layer to reduce spatial dimensions and extract initial feature maps. This is followed by Mobile Inverted Bottleneck Convolution (MBConv) blocks, which use depthwise separable convolutions to reduce computational complexity. Each MBConv block includes Squeeze-and-Excitation (SE) modules for recalibrating channel-wise features and Swish activation for improved non-linearity.

  The architecture progressively reduces spatial dimensions while increasing feature depth. For example, Block 2 reduces dimensions from 112x112x16 to 56x56x24, and Block 4 transitions from 28x28x64 to 14x14x128. The final MBConv layers produce a compact feature map, which is passed through a global average pooling (GAP) layer to summarize features.

  The model concludes with a fully connected (FC) layer and a softmax layer to classify images into predefined categories. EfficientNet-B0's design makes it suitable for real-time applications, offering high accuracy and computational efficiency for deployment in resource-constrained environments.

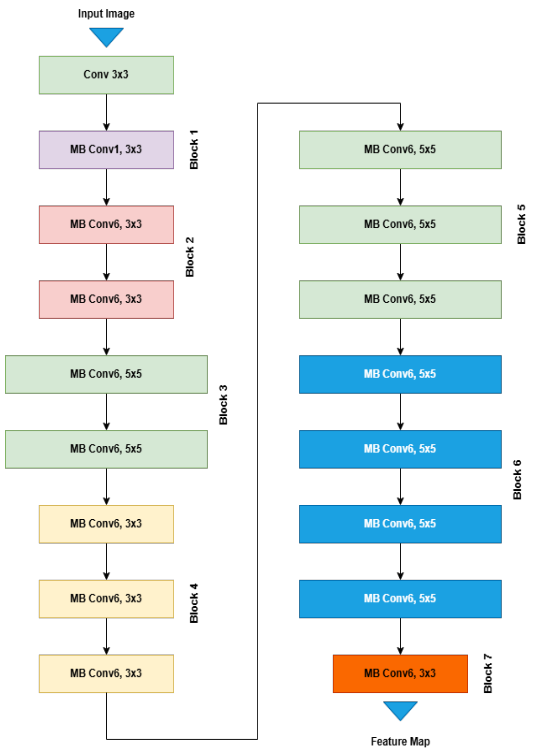

Figure 4.1.1: EfficientNet-B0 Architecture

- **EfficientNet-B3:** EfficientNet-B3 is a scalable and efficient convolutional neural network (CNN) designed for resource-constrained environments like mobile and embedded devices. It balances network depth, width, and resolution using a compound scaling method, achieving high performance with minimal computational complexity.

  The model takes a 300x300x3 input image and begins with a 3x3 convolution layer that reduces the spatial dimensions to 112x112x40 while extracting initial features. It then uses a series of Mobile Inverted Bottleneck Convolution (MBConv) blocks. These blocks combine depthwise separable convolutions for reduced computation, Squeeze-and-Excitation (SE) modules for channel-wise recalibration, and Swish activation for improved non-linearity.

  The architecture progressively reduces spatial dimensions while increasing feature depth. Key stages include an MBConv1 block reducing dimensions to 56x56x32, followed by MBConv6 blocks reducing dimensions to 28x28x48, 14x14x96, and eventually 7x7x384. A final 1x1 convolution and global average pooling (GAP) layer reduce the spatial size to 1x1, followed by a fully connected (FC) layer producing a 1536-dimensional feature vector. The classifier then generates class probabilities.

EfficientNet-B3's efficient design makes it ideal for real-time applications, offering high accuracy and computational efficiency for tasks like image classification and object detection.

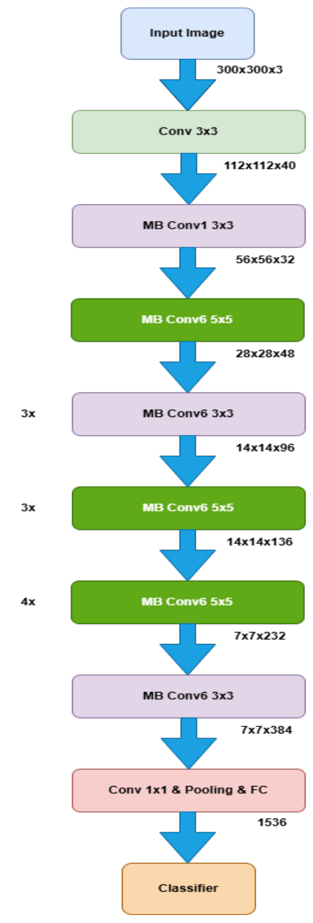

Figure 4.1.2: EfficientNet-B3 Architecture

- **VGG16:** VGG (Visual Geometry Group) is a convolutional neural network designed for image classification, known for its simplicity and effective use of small 3x3 convolutional kernels.

  The network processes input images through stages of convolutional layers followed by max-pooling. The first stage includes two 3x3 convolutional layers with 64 filters, followed by max-pooling. The second stage uses two 3x3 convolutions with 128 filters and max-pooling. The third stage includes three 3x3 convolutions with 256 filters, and the fourth and fifth stages have three 3x3 convolutions with 512 filters each, all followed by max-pooling.

After feature extraction, two fully connected layers with 4096 neurons process the features. A final fully connected layer with 100 neurons generates class probabilities via a softmax layer.

VGG's structured use of small filters and deep layers makes it highly effective for feature extraction and widely used in transfer learning and computer vision tasks.

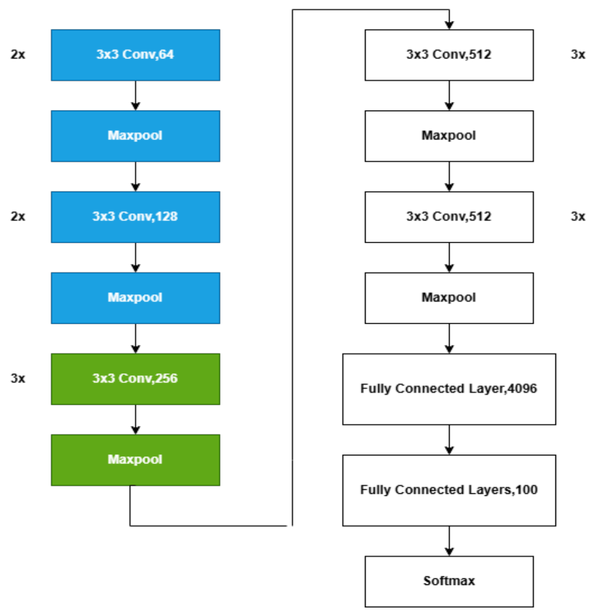

Figure 4.1.3: VGG16 Architecture

- **ResNet-50:** ResNet (Residual Neural Network) is a deep convolutional neural network designed to address the vanishing gradient problem in deep networks. It introduces residual learning through shortcut connections that skip one or more layers, enabling efficient gradient flow and allowing very deep networks to train effectively without performance degradation.

  The architecture starts with a 224x224x3 input image, processed through an initial 7x7 convolution layer with a stride of 2, followed by batch normalization, activation, and 3x3 max-pooling.

These layers reduce spatial dimensions and extract basic features. ResNet's core consists of two block types: convolutional blocks and identity blocks. Convolutional blocks downsample spatial dimensions while increasing feature depth using three layers of convolutions (1x1, 3x3, 1x1) and shortcut connections for residual learning. Identity blocks maintain spatial dimensions and refine features without downsampling.

The architecture stacks these blocks hierarchically, with configurations like two identity blocks followed by three and five identity blocks in subsequent stages. This progressive design captures both low-level and high-level features for effective feature representation. After feature extraction, global average pooling reduces spatial dimensions, and a fully connected layer produces the final feature vector, followed by a softmax layer for classification.

ResNet's use of residual connections revolutionized deep learning by enabling the training of very deep networks. Its efficiency and accuracy make it widely used in tasks like image classification, object detection, and medical image analysis.

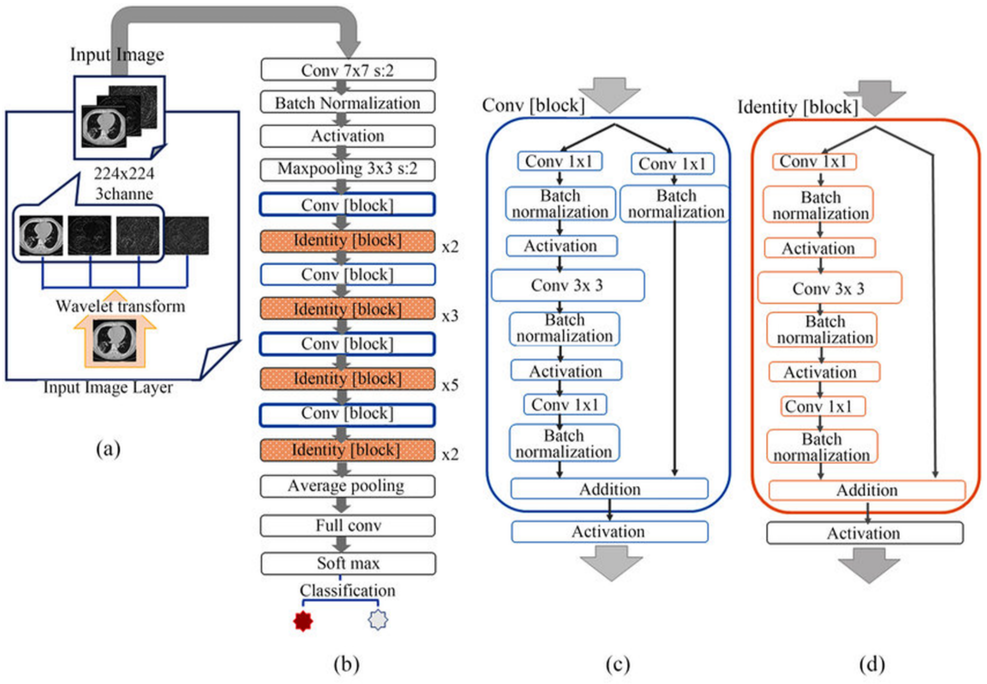

Figure 4.1.4: ResNet-50 Architecture

5. IMPLEMENTATION

The implementation phase of the project is where the detailed design is actually transformed into working code. Aim of the phase is to identify skin cancer using deep learning techniquies :

5.1 Code

```
# EfficientNet - B3


import os, glob
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import train_test_split
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Model, Sequential , load_model
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense,
Dropout, Input
from tensorflow.keras.callbacks import EarlyStopping
from tensorflow.keras.applications import EfficientNetB3
from tensorflow.keras.applications.efficientnet import preprocess_input
from sklearn.metrics import classification_report
import numpy as np
import cv2
import time


# Load and preprocess data
file_path = r'C:\Users\kpadm\OneDrive\Desktop\Projects\Major Project\myproject
\melanoma_cancer_dataset\train'
```

```
name_class = os.listdir(file_path)
filepaths = list(glob.glob(file_path + '/**/*.*'))
labels = list(map(lambda x: os.path.split(os.path.split(x)[0])[1], filepaths))
filepath = pd.Series(filepaths, name='Filepath').astype(str)
labels = pd.Series(labels, name='label')
data = pd.concat([filepath, labels], axis=1)
data = data.sample(frac=1).reset_index(drop=True)

# Data visualization
counts = data.label.value_counts()
sns.barplot(x=counts.index, y=counts)
plt.xlabel('Type')
plt.xticks(rotation=90)
plt.show()

# Train-test split
train, test = train_test_split(data, test_size=0.25, random_state=42)
fig, axes = plt.subplots(nrows=5, ncols=3, figsize=(10, 8), subplot_kw={'xti
cks': [], 'yticks': []})
for i, ax in enumerate(axes.flat):
    ax.imshow(plt.imread(data.Filepath[i]))
    ax.set_title(data.label[i])
plt.tight_layout()
plt.show()

# Data generators
train_datagen = ImageDataGenerator(preprocessing_function=preprocess_input)
```

```
test_datagen = ImageDataGenerator(preprocessing_function=preprocess_input)

train_gen = train_datagen.flow_from_dataframe(
    dataframe=train,
    x_col='Filepath',
    y_col='label',
    target_size=(300, 300),
    class_mode='categorical',
    batch_size=32,
    shuffle=True,
    seed=42
)

valid_gen = train_datagen.flow_from_dataframe(
    dataframe=test,
    x_col='Filepath',
    y_col='label',
    target_size=(300, 300),
    class_mode='categorical',
    batch_size=32,
    shuffle=False,
    seed=42
)

test_gen = train_datagen.flow_from_dataframe(
    dataframe=test,
    x_col='Filepath',
    y_col='label',
    target_size=(300, 300),
    class_mode='categorical',
    batch_size=32,
    shuffle=False,
)

Found 7203 validated image filenames belonging to 2 classes.
Found 2402 validated image filenames belonging to 2 classes.
Found 2402 validated image filenames belonging to 2 classes.

# Load EfficientNetB3 for feature extraction
efficientnet = EfficientNetB3(
    input_shape=(300, 300, 3),
    include_top=False,
    weights='imagenet',
    pooling='avg'
)
efficientnet.trainable = False

# Build custom CNN model
model = Sequential([
    efficientnet,
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(len(data['label'].unique()), activation='softmax')
])

model.compile(
    optimizer='adam',
    loss='categorical_crossentropy',
    metrics=['accuracy']
)

# Train model
my_callbacks = [EarlyStopping(monitor='val_accuracy', patience=2)]

history = model.fit(
    train_gen,
    validation_data=valid_gen,
    epochs=5,
    callbacks=my_callbacks
)

c:\Users\kpadm\anaconda3\Lib\site-packages\keras\src\trainers\data_adapters
\py_dataset_adapter.py:121: UserWarning: Your `PyDataset` class should call
super().__init__(**kwargs)` in its constructor. `**kwargs` can include `work
ers`, `use_multiprocessing`, `max_queue_size`. Do not pass these arguments to
`fit()`, as they will be ignored.
  self._warn_if_super_not_called()
Epoch 1/5
226/226  770s 3s/step - accuracy: 0.8226 - loss: 0.3723 - val_accuracy: 0.9084
```

```
- val_loss: 0.2219
Epoch 2/5
226/226  706s 3s/step - accuracy: 0.9033 - loss: 0.2362 - val_accuracy: 0.9092
- val_loss: 0.1812
Epoch 3/5
226/226  699s 3s/step - accuracy: 0.9114 - loss: 0.2210 - val_accuracy: 0.9104
- val_loss: 0.1722
Epoch 4/5
226/226  673s 3s/step - accuracy: 0.9258 - loss: 0.1988 - val_accuracy: 0.9145
- val_loss: 0.1695
Epoch 5/5
226/226  720s 3s/step - accuracy: 0.9179 - loss: 0.2038 - val_accuracy: 0.9242
- val_loss: 0.1684

# Save model
model.save("model_efficientnet_b3.h5")

WARNING:absl:You are saving your model as an HDF5 file via ‘model.save()‘ or
‘keras.saving.save_model(model)‘. This file format is considered legacy. We rec
ommend using instead the native Keras format, e.g. ‘model.save(’my_model.keras’)
‘ or‘keras.saving.save_model(model, ’my_model.keras’)‘.

# Evaluate model
results = model.evaluate(test_gen, verbose=0)
print("    Test Loss: {:.5f}".format(results[0]))
print("Test Accuracy: {:.2f}%".format(results[1] * 100))
```

```
c:\Users\kpadm\anaconda3\Lib\sitepackages\keras\src\trainers\data_adapters
\py_dataset_adapter.py:121: UserWarning: Your ‘PyDataset‘ class should call
‘super().__init__(**kwargs)‘ in its constructor. ‘**kwargs‘ can include ‘work
ers‘, ‘use_multiprocessing‘, ‘max_queue_size‘. Do not pass these arguments to
‘fit()‘, as they will be ignored.
  self._warn_if_super_not_called()
    Test Loss: 0.16840
Test Accuracy: 92.42%

# Predict on test data
import numpy as np
pred = model.predict(test_gen)
pred = np.argmax(pred, axis=1)

76/76  199s 3s/step

# Decode predictions
labels = train_gen.class_indices
labels = dict((v, k) for k, v in labels.items())
pred = [labels[k] for k in pred]
y_test = list(test.label)
print(classification_report(y_test, pred))

              precision    recall  f1-score   support

      benign       0.92      0.93      0.92      1245
   malignant       0.92      0.91      0.92      1157
```

```
    accuracy                           0.92      2402
   macro avg       0.92      0.92      0.92      2402
weighted avg       0.92      0.92      0.92      2402

# Visualization of predictions
fig, axes = plt.subplots(nrows=5, ncols=2, figsize=(12, 8), subplot_kw={'xti
cks': [], 'yticks': []})
for i, ax in enumerate(axes.flat):
    ax.imshow(plt.imread(test.Filepath.iloc[i]))
    ax.set_title(f"True: {test.label.iloc[i]}\nPredicted: {pred[i]}")
plt.tight_layout()
plt.show()

# Load and use the model
loaded_model = load_model("model_efficientnet_b3.h5")

WARNING:absl:Compiled the loaded model, but the compiled metrics have yet to
be built. `model.compile_metrics` will be empty until you train or evaluate the
model.

# Test single image prediction
img_path = r'C:\Users\kpadm\OneDrive\Desktop\Projects\Major Project\myproject\m
elanoma_cancer_dataset\test\benign\melanoma_9627.jpg'
img = cv2.imread(img_path)
img = cv2.resize(img, (300,300)) # Resize to EfficientNetB3 dimensions
x=np.expand_dims(img, axis=0)
x = preprocess_input(x)
```

```
result = loaded_model.predict(x)

1/1  5s 5s/step

# Display result
p = list((result * 100).astype('int'))
pp = list(p[0])
index = pp.index(max(pp))
name_class = ['Benign', 'Malignant']
print(f"Prediction: {name_class[index]} with confidence {max(pp)}%")
plt.title(name_class[index])
plt.imshow(img)

Prediction: Benign with confidence 95%

# Test single image prediction
img_path = r'C:\Users\kpadm\OneDrive\Desktop\Projects\Major Project\myproject\m
elanoma_cancer_dataset\test\malignant\melanoma_10108.jpg'
img = cv2.imread(img_path)
img = cv2.resize(img, (300,300)) # Resize to EfficientNetB3 dimensions
x=np.expand_dims(img, axis=0)
x = preprocess_input(x)
result = loaded_model.predict(x)

1/1  0s 192ms/step

# Display result
```

```
p = list((result * 100).astype('int'))
pp = list(p[0])
index = pp.index(max(pp))
name_class = ['Benign', 'Malignant']
print(f"Prediction: {name_class[index]} with confidence {max(pp)}%")
plt.title(name_class[index])
plt.imshow(img)

Prediction: Benign with confidence 89%

import time

# Function to measure execution time
def measure_time(model, data_gen):
    start_time = time.time()
    results = model.predict(data_gen)
    end_time = time.time()
    total_time = end_time - start_time
    return total_time, results

# Measure the time complexity for the test data
time_taken, _ = measure_time(model, test_gen)

# Output the result
print(f"Time taken for prediction on test data: {time_taken:.5f} seconds")
print(f"Time per batch: {time_taken / len(test_gen):.5f} seconds")
```

```
76/76  182s 2s/step
Time taken for prediction on test data: 28.06975 seconds
Time per batch: 0.44579 seconds

# ResNet-50
# import system libs
import os
import itertools

# import data handling tools
import numpy as np
import pandas as pd
import seaborn as sns
sns.set_style('darkgrid')
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, classification_report

# import Deep learning Libraries
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.models import Sequential
from tensorflow.keras.optimizers import Adam, Adamax
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense,
Dropout, GlobalAveragePooling2D, BatchNormalization
from tensorflow.keras.models import Model
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau
from keras.applications.resnet50 import ResNet50
from keras.applications import VGG16

epochs = 15   # number of all epochs in training
ResNet50_history = ResNet50_model.fit(train_gen, epochs= epochs, verbose= 1,
validation_data= valid_gen, shuffle= False)

Epoch 1/15
1201/1201  939s 782ms/step - accuracy: 0.8871 - loss: 0.2834 - val_accuracy:
0.7400 - val_loss: 0.6812
Epoch 2/15
1201/1201  1246s 1s/step - accuracy: 0.8880 - loss: 0.2750 - val_accuracy:
0.7700 - val_loss: 0.4998
Epoch 3/15
1201/1201  1225s 1s/step - accuracy: 0.8852 - loss: 0.2826 - val_accuracy:
0.8680 - val_loss: 0.3515
Epoch 4/15
1201/1201  1288s 1s/step - accuracy: 0.8889 - loss: 0.2726 - val_accuracy:
0.8860 - val_loss: 0.3205
Epoch 5/15
1201/1201  1244s 1s/step - accuracy: 0.8883 - loss: 0.2817 - val_accuracy:
0.8360 - val_loss: 0.4649
Epoch 6/15
1201/1201  1202s 1s/step - accuracy: 0.8987 - loss: 0.2517 - val_accuracy:
0.9080 - val_loss: 0.2551
Epoch 7/15
```

```
1201/1201  1114s 928ms/step - accuracy: 0.9004 - loss: 0.2420 - val_accuracy:
0.9080 - val_loss: 0.2399
Epoch 8/15
1201/1201  831s 692ms/step - accuracy: 0.9008 - loss: 0.2489 - val_accuracy:
0.9100 - val_loss: 0.2234
Epoch 9/15
1201/1201  911s 758ms/step - accuracy: 0.8957 - loss: 0.2554 - val_accuracy:
0.8680 - val_loss: 0.2879
Epoch 10/15
1201/1201  681s 567ms/step - accuracy: 0.9094 - loss: 0.2302 - val_accuracy:
0.6900 - val_loss: 0.6818
Epoch 11/15
1201/1201  662s 551ms/step - accuracy: 0.9025 - loss: 0.2451 - val_accuracy:
0.8380 - val_loss: 0.3850
Epoch 12/15
1201/1201  642s 535ms/step - accuracy: 0.9062 - loss: 0.2434 - val_accuracy:
0.8920 - val_loss: 0.2597
Epoch 13/15
...
Epoch 14/15
1201/1201  655s 545ms/step - accuracy: 0.9092 - loss: 0.2220 - val_accuracy:
0.7780 - val_loss: 0.6084
Epoch 15/15
1201/1201  640s 532ms/step - accuracy: 0.9014 - loss: 0.2357 - val_accuracy:
0.9104 - val_loss: 0.2351
```

```
# VGG16
# get the pre-trained model (VGG16)
base_model = VGG16(weights='imagenet', include_top=False, input_shape = img_sh
ape, pooling= 'max')
# freeze four convolution blocks
for layer in base_model.layers[:15]:
    layer.trainable = False
# fine-tune VGG16 (Adding some custom layers on top)
x = base_model.output
x = Flatten()(x)
x = Dense(512, activation = 'relu')(x)
x = Dropout(0.2)(x)   # Dropout layer to prevent overfitting
x = Dense(256, activation = 'relu')(x)
x = Dense(128, activation = 'relu')(x)
x = Dense(32, activation = 'relu')(x)
predictions = Dense(class_counts, activation = "sigmoid")(x)  # output layer
with softmax activation
# the model
VGG16_model = Model(inputs = base_model.input, outputs = predictions)

epochs = 7   # number of all epochs in training
VGG16_history = VGG16_model.fit(train_gen, epochs= epochs, verbose= 1, validat
ion_data= valid_gen, shuffle= False)

Epoch 1/7
1201/1201  2092s 2s/step - accuracy: 0.8160 - loss: 0.3810 - val_accuracy: 0.90
00 - val_loss: 0.2454
```

```
Epoch 2/7
1201/1201  9380s 8s/step - accuracy: 0.9218 - loss: 0.2066 - val_accuracy: 0.92
60 - val_loss: 0.1954
Epoch 3/7
1201/1201  1940s 2s/step - accuracy: 0.9414 - loss: 0.1597 - val_accuracy: 0.93
20 - val_loss: 0.1927
Epoch 4/7
1201/1201  1917s 2s/step - accuracy: 0.9474 - loss: 0.1377 - val_accuracy: 0.91
00 - val_loss: 0.2161
Epoch 5/7
1201/1201  3247s 3s/step - accuracy: 0.9584 - loss: 0.1165 - val_accuracy: 0.92
40 - val_loss: 0.2138
Epoch 6/7
1201/1201  2065s 2s/step - accuracy: 0.9688 - loss: 0.0945 - val_accuracy: 0.92
20 - val_loss: 0.2147
Epoch 7/7
1201/1201  1944s 2s/step - accuracy: 0.9724 - loss: 0.0872 - val_accuracy: 0.92
40 - val_loss: 0.2743
```

6. RESULTS & DISCUSSIONS

Table 6.1 presents a comparative analysis of ResNet-50, VGG-16, EfficientNet-B0, and Efficient Net-B3 across Melanoma, Basal Cell Carcinoma, and Squamous Cell Carcinoma. When looking at skin cancer detection, models like EfficientNet-B3 stand out as the most accurate. It reaches about 92.42 % accuracy for Melanoma, 91.29 % for Basal Cell Carcinoma, and 93.57 % for Squamous Cell Carcinoma. Plus, it's quick to run, saving time and computing power. VGG16 does pretty well too, hitting 92.40 % for Melanoma, but it takes a lot longer to process. ResNet-50 and EfficientNet-B0 hit a good balance—they're pretty accurate and fairly efficient. Overall, these results show that each model has its strengths. But if you're after the best mix of high accuracy and low computational cost, EfficientNet-B3 is the best pick for classifying skin cancers.

Table 6.1: Comparison Table of Types of Skin Cancers

S.No	**Model**	**Melanoma**		**Basal Cell Carcinoma**		**Squamous Cell Carcinoma**	
		Accuracy	Time(in secs)	Accuracy	Time(in secs)	Accuracy	Time(in secs)
1	ResNet-50	91.04 %	57.048	87.80 %	35.16	85.38 %	27.22
2	VGG-16	92.40 %	156.62	86.41 %	90.86	89.16 %	50.662
3	EfficientNet-B0	91.42 %	36.29	89.90 %	24.351	90.70 %	13.32
4	EfficientNet-B3	92.42 %	28.069	91.29 %	8.908	93.57 %	6.04

Table 6.2 provides a side-by-side look at how different models—ResNet-50, VGG-16, EfficientNetB0, and EfficientNetB3—perform on three datasets: ISIC, Skin Cancer, and HAM10000. Overall, VGG-16 and EfficientNetB3 stand out, hitting the highest accuracy rates on HAM10000 with 92.40 % and 92.42 %, respectively. When it comes to the Skin Cancer dataset, EfficientNetB0 and EfficientNetB3 also do really well, reaching 89.19 % and 88.69 %. ResNet-50 and VGG-16 aren't far behind, both showing steady results around 88.5 %. For the ISIC dataset, VGG-16 leads with an accuracy of 88.48 %, with EfficientNetB3 close behind at 86.56 %. These results show that different models perform better on different datasets, emphasizing that there's no one-size-fits-all solution.

Table 6.2: Comparison Table of Melanoma datasets

S.No	Model	ISIC Skin Cancer		Melanoma Skin Cancer		HAM10000	
		Accuracy	Time(in secs)	Accuracy	Time(in secs)	Accuracy	Time(in secs)
1	ResNet-50	80.00%	193.70	88.55%	900.84	91.04%	57.048
2	VGG-16	88.48%	211.09	88.38%	288.50	92.40%	156.62
3	EfficientNet-B0	84.59%	59.11	89.19%	144.92	91.42%	36.29
4	EfficientNet-B3	86.56%	58.77	88.69%	147.44	92.42%	28.069

Table 6.3 and Table 6.4 present a comparative analysis of epochs per batch time for different skin cancer and melanoma-specific datasets across four deep learning models—ResNet-50, VGG-16, EfficientNet-B0, and EfficientNet-B3. In Table III, EfficientNet-B3 demonstrates the lowest batch time across all skin cancer types, particularly in basal cell carcinoma (0.989 sec) and squamous cell carcinoma (1.007 sec), while VGG-16 records the highest times, indicating higher computational complexity. Similarly, in Table IV, EfficientNet-B3 again shows the lowest batch time across melanoma datasets, achieving 0.445 sec for HAM10000 and 1.399 sec for ISIC. ResNet-50 and EfficientNet-B0 also exhibit competitive performance, whereas VGG-16 remains the slowest across datasets. These results highlight the efficiency of EfficientNet models in training time optimization, making them suitable for real-time applications.

Table 6.3: Epochs per Batch Time Comparison Across Different Skin Cancer Datasets

S.No	Model	Melanoma	Basal Cell Carcinoma	Squamous Cell Carcinoma
1	ResNet-50	0.905	3.907	3.265
2	VGG-16	2.060	10.096	8.44
3	EfficientNet-B0	0.477	2.705	2.221
4	EfficientNet-B3	0.445	0.989	1.007

Table 6.4: Epochs per Batch Time Comparison Across Different Melanoma Datasets

S.No	Model	ISIC Skin Cancer	Melanoma Skin Cancer	HAM10000
1	ResNet-50	1.399	9.68	0.905
2	VGG-16	5.025	2.45	2.060
3	EfficientNet-B0	2.814	1.558	0.477
4	EfficientNet-B3	1.399	1.585	0.445

Figure 6.1 shows the training and validation accuracy and loss for the EfficientNet-B3 model on the melanoma dataset. Accuracy continuously improves, reaching about 0.94, while loss decreases consistently. The lower validation loss suggests good generalization with minimal overfitting.

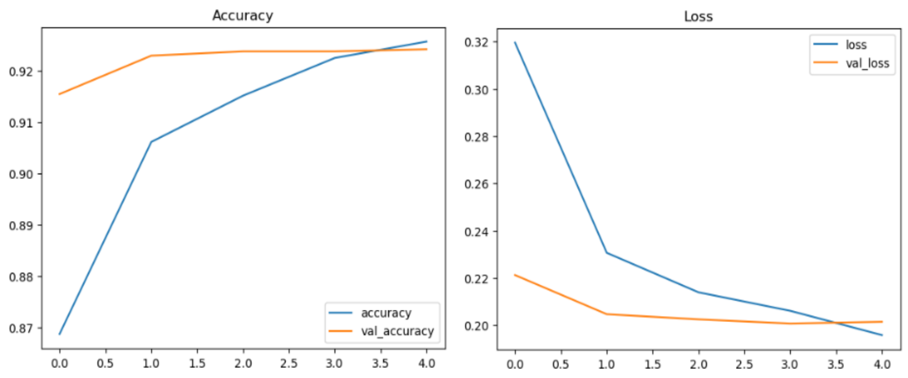

Figure 6.1: Accuracy and Loss for Melanoma using Efficient NetB3 Model

Figure 6.2 shows the training and validation accuracy and loss for the EfficientNet-B3 model on the basal cell carcinoma dataset. Accuracy continuously improves, reaching about 0.91, while loss shows an overall decreasing trend. The variations indicate some inconsistency, but the model maintains good performance.

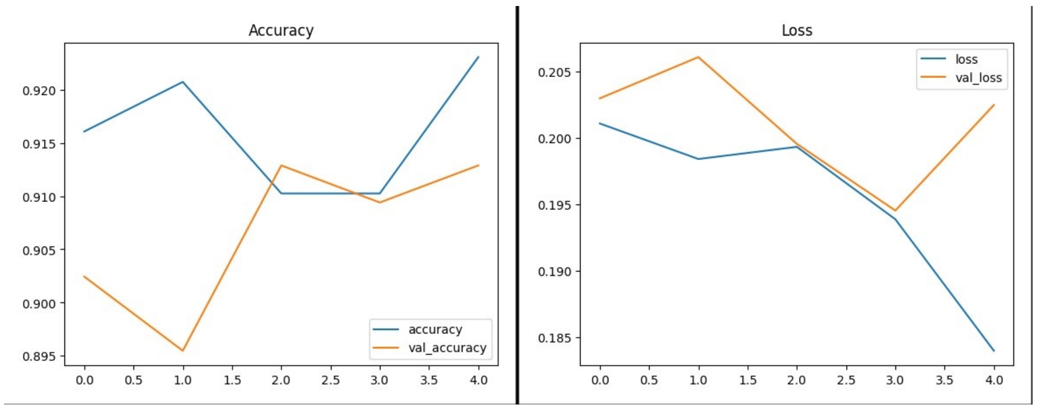

Figure 6.2: Accuracy and Loss for Basal Cell Carcinoma using EfficientNet-B3 Model

Figure 6.3 presents the training and validation accuracy and loss for the EfficientNet-B3 model on the squamous cell carcinoma dataset. Accuracy steadily increases, reaching around 0.93, while loss consistently decreases, indicating effective learning and good model generalization.

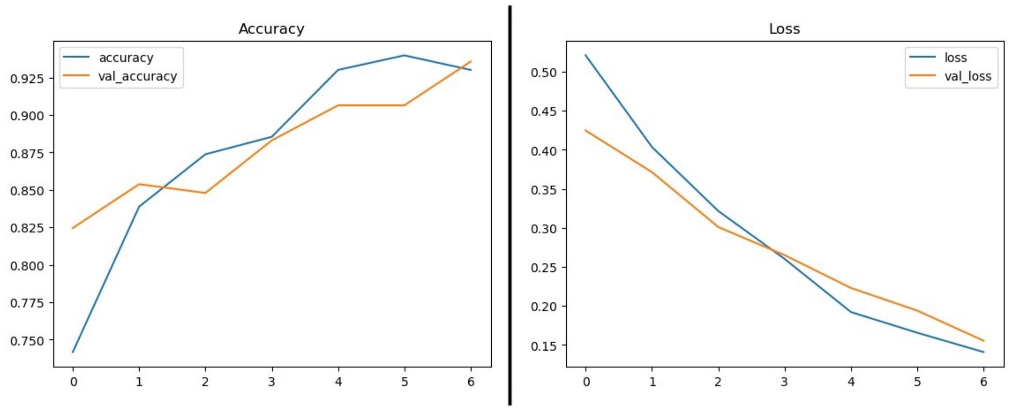

Figure 6.3: Accuracy and Loss for Squamous Cell Carcinoma using EfficientNet-B3 Model

Precision calculates the ratio of true positive predictions among all the positive predictions generated by the model,highlighting its capability to minimize false positives. Recall, alias sensitivity, evaluates the ratio of true positive predic- tions among all actual positive instances, reflecting the model's ability to identify relevant samples. The F1-score is the harmonic mean of precision and recall, balanc- ing their trade-off and proving especially useful for imbalanced datasets. Support represents the number of actual specimen for every type in dataset, providing insight into the data distribution across different classes.

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

Here, TP denotes True Positives, FP denotes False Positives, TN True Negativies and FN is False Negativies.

$$F1{-}Score = 2 \times \frac{Precision \cdot Recall}{Precision + Recall} \tag{3}$$

Table 6.5 displays EfficientNetB0 model's effectiveness on skin lesions. For benign samples, it secured a precision of 0.88, recall of 0.97, and F1-score of 0.92, while for malignant samples, precision was 0.96, recall 0.86, and F1-score 0.91. With an overall accuracy of 0.91 and consistent macro and weighted averages, the model demonstrates balanced performance.

Table 6.5: Classification Report for HAM10000 using EfficientNet-B0

Class	**Precision**	**Recall**	**F1-Score**	**Support**
Benign	0.88	0.97	0.92	1231
Malignant	0.96	0.86	0.91	1171

Table 6.6 shows the EfficientNetB3 model's performance on skin lesions. For benign samples, it secured a precision of 0.89, recall of 0.97, and F1-score of 0.93, while for malignant samples, precision was 0.97, recall 0.87, and F1-score 0.92. With an overall accuracy of 0.92 and consistent macro and weighted averages, the model demonstrates balanced performance.

Table 6.6: Classification Report for HAM10000 using EfficientNet-B3

Class	**Precision**	**Recall**	**F1-Score**	**Support**
Benign	0.89	0.97	0.93	1270
Malignant	0.97	0.87	0.92	1132

Figures 6.4 and 6.5 depict the confusion matrices for the ResNet-50 and VGG-16 models, respectively. The ResNet-50 model performs exceptionally well for benign cases, with 259 true positives and only 1 false negative. However, for malignant cases, it achieves 198 true positives but struggles with 42 false negatives, indicating a challenge in correctly identifying some malignant cases. Similarly, the VGG-16 model records 241 true positives and 19 false negatives for benign cases, while for malignant cases, it achieves 204 true positives but misclassifies 36 instances as false negatives. Although both models demonstrate strong overall performance, they show a tendency to misclassify some malignant cases, highlighting the need for further optimization.

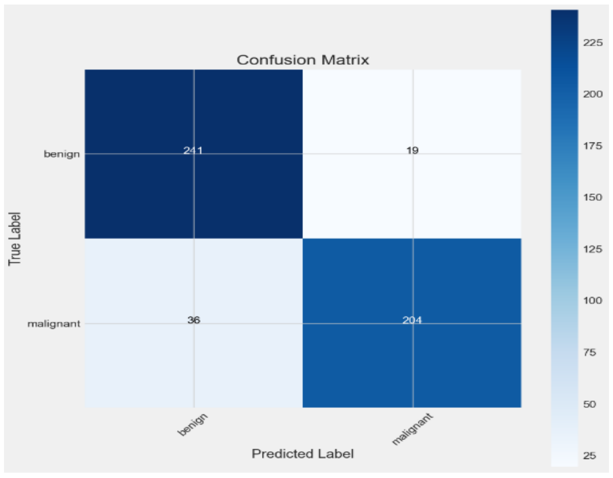

Figure 6.4: Confusion Matrix for HAM10000 using ResNet-50

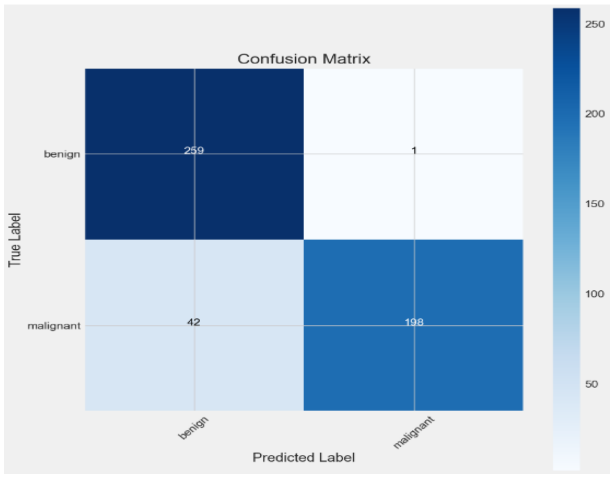

Figure 6.5: Confusion Matrix for HAM10000 using VGG-16

Among all datasets, HAM10000 proves to be the best, yielding the highest accuracies across all models. EfficientNet-B3 emerges as the top-performing model on HAM10000, achieving the highest accuracy of 92.42 %. While VGG-16 also performs well with 92.40 % accuracy, EfficientNet-B3 is preferred due to its efficiency in terms of computational cost and training time. Unlike VGG-16, which has a higher parameter count and requires longer training time, EfficientNet-B3 utilizes a more optimized architecture that balances accuracy and efficiency, making it the superior choice for HAM10000.

7. CONCLUSIONS & FUTURE SCOPE

We developed a deep learning-based framework using CNN architectures such as ResNet-50, VGG16, EfficientNet, and EfficientNet-B3 to address challenges in skin cancer detection, especially melanoma. Our work also includes classification of Basal Cell Carcinoma and Squamous Cell Carcinoma, evaluated across diverse datasets, with a refined focus on melanoma-specific data. Using techniques like data augmentation, normalization, and fairness evaluation, we ensured robustness and equity in predictions. Among the models, EfficientNet-B3 stood out with high accuracy and low computational cost, making it well-suited for real-time, cost-effective diagnosis. The overall system demonstrates strong potential to assist in early detection of skin cancer, particularly in resource-limited settings.

Looking forward, this framework can be further enhanced by integrating it into clinical workflows to provide dermatologists with real-time diagnostic assistance and decision support. Deploying the system as a lightweight mobile or cloud-based application can expand access to advanced diagnostic tools in rural and underserved areas, improving global healthcare equity. Future improvements may include implementing explainable AI (XAI) techniques to improve model interpretability and increase clinician trust in automated predictions. Additionally, extending the model to perform multi-class classification for a broader range of skin diseases would expand its clinical value. Real-time image analysis through smartphone cameras or wearable devices can also be explored, enabling quick, on-the-go screening for patients. Continuous learning from real-world feedback and larger datasets can further refine model accuracy and adaptability, making this a more comprehensive and reliable solution for wide-scale deployment.

REFERENCES

[1] S. S. G. K. Gautam and A. Singh, “Skin cancer identification using deep learning technique,” pp. 553–559, 2024.

[2] N. E.-S. M. M. Diab, Amal G.Fayez, “Accurate skin cancer diagnosis based on convolutional neural networks,” Indonesian Journal Of Electrical Engineering And Computer Science, vol. 25, no. 3, 2022.

[3] S. T. S. Sharma, K. Guleria and S. Kumar, “A deep learning based convolutional neural network model with vgg16 feature extractor for the detection of alzheimer disease using mri scans,” vol. 24, Dec. 2022.

[4] M. I. K. M. M. A. Hasan, Mohammed Rakeibulfatemi, “Comparative analysis of skin cancer (benign vs. malignant) detection using convolutional neural networks,” Journal Of Healthcare Engineering, vol. 2021, 2021.

[5] M. I. e. a. e. a. M. Dildar, S. Akram, “Skin cancer detection: a review using deep learning techniques,” International Journal of Environmental Research and Public Health, vol. 18, no. 10, p. 1–22, 2021.

[6] E. Y. H. Sujaini, Herryramadhan, “Comparing the performance of linear regression versus deep learning on detecting melanoma skin cancer using apple core ml,” Bulletin Of Electrical Engineering And Informatics, vol. 10, no. 6, 2021.

[7] S. Alshourbaji, Ibrahimarabia, “Early detection of skin cancer using deep learning approach,” vol. 20, no. 5, 2021.

[8] “National cancer registration and analysis service (public health england),” 2019.

[9] A. V. S. Adegun, “Deep learning-based system for automatic melanoma detection,” IEEE Access 2019, vol. 8, p. 7160–7172, 2019. [10] S. Kalouche, “Vision-based classification of skin cancer using deep learning,” Stanford’s Machinelearning Course (CS 229), 2016.

[10] S. Kalouche, “Vision-based classification of skin cancer using deep learning,” Stanford’s Machine learning Course (CS 229), 2016.

[11] J. R. Santy A, “Segmentation methods for computer aided melanoma detection,” Global Conference On Communication Technologies, 2015.

[12] B.-M. S. T. C. M. P. Zeljkovic V, Druzgalski C, “Supplemental melanoma diagnosis for darker skin complexion gradients,” Pan American Health Care Exchanges, 2015.

[13] A. N. Yasir R, Rahman A, “Dermatological disease detection using image processing and artificial neural network,” 2014.

[14] F. A. A. A. Y. H. Arifin S, Kibria G, “Dermatological disease diagnosis using color skin images.,” 2012.