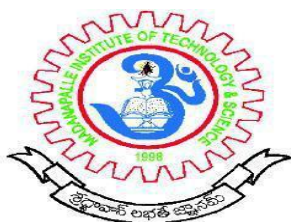


A Project Report
On
**Enhancing Multi-Class Skin Disease Classification: A
Study of Transfer Learning Strategies for Deep Learning
Models**

Submitted to
JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY ANANTAPUR, ANANTHAPURAMU
In Partial Fulfillment of the Requirements for the Award of the Degree of
BACHELOR OF TECHNOLOGY
In
COMPUTER SCIENCE & ENGINEERING (DATA SCIENCE)

Submitted By
N. Abhiram Varma - (20691A3201)
K. Ramanjaneyulu - (20691A3233)
T. Reddy Venkata Chetan - (20691A3236)
S. Sai Manikanta Prakash - (20691A3238)

Under the Guidance of
Dr. K. Lokeshwaran, Ph.D.
Assistant Professor
Department of Computer Science & Engineering (Data Science)



MADANAPALLE INSTITUTE OF TECHNOLOGY & SCIENCE
(UGC – AUTONOMOUS)

(Affiliated to JNTUA, Ananthapuramu)
(Accredited by NBA, Approved by AICTE, New Delhi)
AN ISO 21001:2018 Certified Institution
P. B. No: 14, Angallu, Madanapalle – 517325
2020-2024



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
(DATA SCIENCE)**

BONAFIDE CERTIFICATE

This is to certify that the project work entitled “Enhancing Multi-Class Skin Disease Classification: A Study of Transfer Learning Strategies for Deep Learning Models” is a bonafide work carried out by

N. Abhiram Varma	-	(20691A3201)
K. Ramanjaneyulu	-	(20691A3233)
T. Reddy Venkata Chetan	-	(20691A3236)
S. Sai Manikanta Prakash	-	(20691A3238)

Submitted in partial fulfillment of the requirements for the award of the degree **Bachelor of Technology** in the stream of **Computer Science & Engineering (Data Science)** in **Madanapalle Institute of Technology and Science, Madanapalle**, affiliated to **Jawaharlal Nehru Technological University Anantapur, Ananthapuramu** during the academic year 2022-2023.

Guide
Dr. K. Lokeshwaran,
Assistant Professor,
Department of CSE-DS

Head of the Department
Dr. S. Kusuma,
Assistant Professor and Head,
Department of CSE-DS

Submitted for the University examination held on:

Internal Examiner
Date:

External Examiner
Date:

ABSTRACT

Skin diseases are a common health concern globally, requiring accurate diagnosis and treatment. Various techniques, including medication, topical treatments, phototherapy, and surgery, are employed for their cure. Detection of skin diseases has evolved significantly, with advancements in image processing, machine learning (ML), and deep learning (DL) algorithms. Image processing techniques extract meaningful features from images, while ML and DL algorithms classify diseases based on these features. In this project, we utilize a dataset consisting of 23 classes, with each class containing 192 training images and 20 testing images. This project investigates the classification of various skin diseases, including common conditions like Acne and Rosacea, and more severe cases such as Actinic Keratosis, Basal Cell Carcinoma, and other Malignant Lesions. Additionally, the study explores broader categories like Atopic Dermatitis, Bullous Disease, Cellulitis Impetigo and other Bacterial Infections. We employ image processing techniques to enhance the dataset, followed by classification using DL models such as ResNet50, VGG19, MobileNetLarge, and InceptionV2. Our models achieve accuracies ranging from 82% to 85%, demonstrating their efficacy in skin disease classification. This project contributes to the advancement of automated skin disease diagnosis and treatment planning. Future work could involve incorporating a broader range of skin diseases and exploring advanced deep learning architectures to potentially achieve even higher accuracy and broader applicability.

ACKNOWLEDGEMENT

We sincerely thank the **MANAGEMENT of Madanapalle Institute of Technology & Science** for providing excellent infrastructure and lab facilities that helped me to complete this project.

We sincerely thank **Dr. C. Yuvaraj, M.E., Ph.D., Principal** for guiding and providing facilities for the successful completion of our project at **Madanapalle Institute of Technology & Science, Madanapalle.**

We express our deep sense of gratitude to **Dr. S. Kusuma., Ph.D., Assistant Professor and Head of the Department of CSE - Data Science** for her continuous support in making necessary arrangements for the successful completion of the Project.

We express my deep sense of gratitude to **Mr.G.Rajkumar, M.Tech., (Ph.D), Project Coordinator,** for his valuable guidance and encouragement that helped us to complete this project.

We express our deep gratitude to our guide **Dr. K. Lokeshwaran, Ph.D., Assistant Professor, Department of CSE - Data Science** for his guidance and encouragement that helped us to complete this project.

We also wish to place on record my gratefulness to other **Faculty of the CSE - Data Science Department** and also to our friends and our parents for their help and cooperation during our project work.



MADANAPALLE INSTITUTE OF TECHNOLOGY & SCIENCE

(UGC-AUTONOMOUS INSTITUTION)

Affiliated to JNTUA, Ananthapuramu & Approved by AICTE, New Delhi
NAAC Accredited with A+ Grade, NIRF India Rankings 2021 - Band: 201-250 (Engg.)
NBA Accredited - B.Tech. (CIVIL, CSE, ECE, EEE, MECH), MBA & MCA




RECOGNISED RESEARCH CENTER

Plagiarism Verification Certificate

This is to certify that the B. Tech Project report titled, “**Enhancing Multi-Class Skin Disease Classification: A Study of Transfer Learning Strategies for Deep Learning Models**” submitted by **N. Abhiram Varma (20691A3201), K. Ramanjaneyulu (20691A3233), T. Reddy Venkata Chetan (20691A3236), S. Sai Manikanta Prakash (20691A3238)**, has been evaluated using **Anti- Plagiarism Software, Turnitin** and based on the analysis report generated by the software, the report’s similarity index is found to be 11%.

The following is the Turnitin report for the project report consisting of 28 pages.



Similarity Report ID: oid:3618:55857305

PAPER NAME	AUTHOR
project_report - Copy 11.DOCX	REDDY VENKAT
WORD COUNT	CHARACTER COUNT
6707 Words	41167 Characters
PAGE COUNT	FILE SIZE
28 Pages	4.9MB
SUBMISSION DATE	REPORT DATE
Apr 8, 2024 3:21 PM GMT+5:30	Apr 8, 2024 3:22 PM GMT+5:30

● 11% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

- 5% Internet database
- 3% Publications database
- Crossref database
- Crossref Posted Content database
- 9% Submitted Works database

● Excluded from Similarity Report

- Bibliographic material
- Small Matches (Less than 8 words)

GUIDE



NATIONAL INSTITUTE OF TECHNOLOGY PUDUCHERRY
(An Institute of National Importance under MoE, Govt. of India)
Thiruvettakudy, Karaikal – 609 609
Department of Computer Science and Engineering

Dr. B. Surendiran
Associate Professor

Dated 01.05.2024

TO WHOMSOEVER IT MAY CONCERN

This is to certify that **Mr. NANDIMANDALAM ABHIRAM VARMA** bearing register number 20691A3201, a student of B.Tech., Computer Science and Engineering (Data Science) studying in VIII Semester at Madanapalle Institute of Technology & Science, Madanapalle has successfully completed **Full Semester Internship Programme (From 01st January 2024 to 30th April 2024)** in the Department of Computer Science and Engineering, NIT Puducherry. He worked under my supervision on the topic **“Enhancing Multi-Class Skin Disease Classification: A Study of Transfer Learning Strategies for Deep Learning Models”**.

During the period of the Internship Programme with us, he was found punctual, hardworking and inquisitive.

We wish him every success in life.




Dr. B. Surendiran 01/05/2024

Dr. B. SURENDIRAN
Associate Professor
Dept of Computer Science and Engineering
National Institute of Technology Puducherry
Karaikal - 609 609, U.T. Puducherry, India

E-mail: surendiran@nitpy.ac.in

Landline: 04368 – 265235

Mobile: +91-9942761363



NATIONAL INSTITUTE OF TECHNOLOGY PUDUCHERRY
(An Institute of National Importance under MoE, Govt. of India)
Thiruvettakudy, Karaikal – 609 609
Department of Computer Science and Engineering

Dr. B. Surendiran
Associate Professor

Dated 01.05.2024

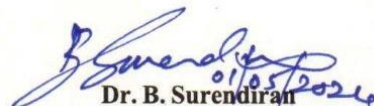
TO WHOMSOEVER IT MAY CONCERN

This is to certify that **Mr. KAMISSETTI RAMANJANEYULU** bearing register number 20691A3233, a student of B.Tech., Computer Science and Engineering (Data Science) studying in VIII Semester at Madanapalle Institute of Technology & Science, Madanapalle has successfully completed **Full Semester Internship Programme (From 01st January 2024 to 30th April 2024)** in the Department of Computer Science and Engineering, NIT Puducherry. He worked under my supervision on the topic **“Enhancing Multi-Class Skin Disease Classification: A Study of Transfer Learning Strategies for Deep Learning Models”**.

During the period of the Internship Programme with us, he was found punctual, hardworking and inquisitive.

We wish him every success in life.




Dr. B. Surendiran

Dr. B. SURENDIRAN
Associate Professor
Dept. of Computer Science and Engineering
National Institute of Technology Puducherry
Karaikal - 609 609, U T of Puducherry, India

E-mail: surendiran@nitpy.ac.in

Landline: 04368 – 265235

Mobile: +91-9942761363



NATIONAL INSTITUTE OF TECHNOLOGY PUDUCHERRY
(An Institute of National Importance under MoE, Govt. of India)
Thiruvettakudy, Karaikal – 609 609
Department of Computer Science and Engineering

Dr. B. Surendiran
Associate Professor

Dated 01.05.2024

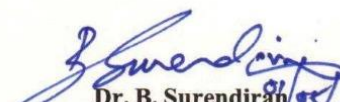
TO WHOMSOEVER IT MAY CONCERN

This is to certify that **Mr. TIRUMALA REDDY VENKATA CHETAN** bearing register number 20691A3236, a student of B.Tech., Computer Science and Engineering (Data Science) studying in VIII Semester at Madanapalle Institute of Technology & Science, Madanapalle has successfully completed **Full Semester Internship Programme (From 01st January 2024 to 30th April 2024)** in the Department of Computer Science and Engineering, NIT Puducherry. He worked under my supervision on the topic **“Enhancing Multi-Class Skin Disease Classification: A Study of Transfer Learning Strategies for Deep Learning Models”**.

During the period of the Internship Programme with us, he was found punctual, hardworking and inquisitive.

We wish him every success in life.




Dr. B. Surendiran 01/05/2024

Dr. B. SURENDIRAN
Associate Professor
Dept. of Computer Science and Engineering
National Institute of Technology Puducherry
Karaikal - 609 609, U.T of Puducherry, India

E-mail: surendiran@nitpy.ac.in

Landline: 04368 – 265235

Mobile: +91-9942761363



NATIONAL INSTITUTE OF TECHNOLOGY PUDUCHERRY
(An Institute of National Importance under MoE, Govt. of India)
Thiruvettakudy, Karaikal – 609 609
Department of Computer Science and Engineering

Dr. B. Surendiran
Associate Professor

Dated 01.05.2024

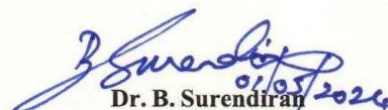
TO WHOMSOEVER IT MAY CONCERN

This is to certify that **Mr. SURA SAI MANIKANTA PRAKASH** bearing register number 20691A3238, a student of B.Tech., Computer Science and Engineering (Data Science) studying in VIII Semester at Madanapalle Institute of Technology & Science, Madanapalle has successfully completed **Full Semester Internship Programme (From 01st January 2024 to 30th April 2024)** in the Department of Computer Science and Engineering, NIT Puducherry. He worked under my supervision on the topic **“Enhancing Multi-Class Skin Disease Classification: A Study of Transfer Learning Strategies for Deep Learning Models”**.

During the period of the Internship Programme with us, he was found punctual, hardworking and inquisitive.

We wish him every success in life.




Dr. B. Surendiran

Dr. B. SURENDRAN
Associate Professor
Dept. of Computer Science and Engineering
National Institute of Technology Puducherry
Karaikal - 609 609, U.T. of Puducherry, India

E-mail: surendiran@nitpy.ac.in

Landline: 04368 – 265235

Mobile: +91-9942761363

DECLARATION

We hereby declare that the results embodied in this project “**Enhancing Multi-Class Skin Disease Classification: A Study of Transfer Learning Strategies for Deep Learning Models**” by us under the guidance of **Dr. K. Lokeshwaran, Ph.D., Assistant Professor, Dept. of CSE - Data Science** in partial fulfillment of the award of **Bachelor of Technology in Computer Science & Engineering (Data Science)** from **Jawaharlal Nehru Technological University Anantapur, Ananthapuramu** and we have not submitted the same to any other University/institute for the award of any other degree.

Date:

Place:

PROJECT ASSOCIATES

N. Abhiram Varma
K. Ramanjaneyulu
T. Reddy Venkata Chetan
S. Sai Manikanta Prakash

I certify that the above statement made by the students is correct to the best of my knowledge.

Date:

Guide:

INDEX

S.NO	TOPIC	PAGE NO.
1.	INTRODUCTION	1
	1.1 MOTIVATION	2
	1.2 PROBLEM DEFINITION	2
	1.3 OBJECTIVE OF THE PROJECT	3
	1.4 LIMITATIONS OF PROJECT	3
	1.5 ORGANIZATION OF DOCUMENTATION	4
2.	LITERATURE SURVEY	6
	2.1 INTRODUCTION	7
	2.2 EXISTING SYSTEM	7
	2.3 DISADVANTAGES OF EXISTING SYSTEM	8
	2.4 PROPOSED SYSTEM	9
	2.5 ADVANTAGES OVER EXISTING SYSTEM	10
3.	ANALYSIS	11
	3.1 INTRODUCTION	12
	3.2 REQUIREMENT SPECIFICATIONS	12
	3.2.1 HARDWARE REQUIREMENTS	12
	3.2.2 SOFTWARE REQUIREMENTS	13
	3.3 CONTENT DIAGRAM OF PROJECT	14
4.	DESIGN	15
	4.1 INTRODUCTION	16
	4.2 DATA FLOW DIAGRAM	17
	4.3 SYSTEM ARCHITECTURE	18
	4.4 MODULE DESIGNING AND ORGANIZATION	19
5.	IMPLEMENTATION AND RESULTS	21
	5.1 INTRODUCTION	22

	5.2 IMPLEMENTATION OF KEY FUNCTIONS	23
	5.3 METHOD OF IMPLEMENTATION	24
6	TESTING	43
	6.1 INTRODUCTION	44
	6.2 LEVELS OF TESTING	45
	6.3 VALIDATION TESTING	45
	6.4 CONCLUSION	46
7.	CONCLUSION	47
	7.1 CONCLUSION	48
	7.2 FUTURE ENHANCEMENT	48
8.	REFERENCES	49
	8.1 REFERENCES	50

List of Figures

S.No	Figure	Name of the Figure	Page Number
1	3.3.1	Content Diagram	14
2	4.2.1	DATA FLOW Diagram	17
3	4.3.1	Architecture Diagram	18
4	5.3.1	Images after resizing	39
5	5.3.2	Summary of ResNet50 model (a) Training and Validation Accuracy (b) Training and Validation Loss (c) Confusion matrix (d) Classification report	39
6	5.3.3	Summary of InceptionV3 model (a) Training and Validation Accuracy (b) Training and Validation Loss (c) Confusion matrix (d) Classification report	40
7	5.3.4	Summary of VGG19 model (a) Training and Validation Accuracy (b) Training and Validation Loss (c) Confusion matrix (d) Classification report	41
8	5.3.5	Summary of MobileNetLarge model (a) Training and Validation Accuracy (b) Training and Validation Loss (c) Confusion matrix (d) Classification report	42

List of Tables

S.No	Table	Name of the Table	Page Number
1	5.3.1	Performance of ResNet50 model	39
2	5.3.2	Performance of InceptionV3 model	40
3	5.3.3	Performance of VGG19 model	41
4	5.3.4	Performance of MobileNetLarge model	42
5	5.3.5	Performance of all models	42

CHAPTER 1

INTRODUCTION

1.1 MOTIVATION:

The key motivations for this project on enhancing multi-class skin disease classification using deep learning models are:

1. Importance of Accurate Skin Disease Diagnosis:

- A significant fraction of the world's population suffers from skin illnesses, and prompt diagnosis is essential to appropriate treatment.
- Improvements in deep learning, machine learning, and image processing can greatly increase the precision and speed of diagnosing and classifying skin diseases.

2. Limitations of Traditional Diagnosis Methods:

- When it comes to complicated or uncommon skin disorders, healthcare practitioners' traditional visual assessment methods might be arbitrary and prone to mistakes.
- Technological advancements provide valuable tools to enhance diagnostic accuracy and support clinical decision-making.

3. Potential of Deep Learning in Dermatology:

- The recent success of deep learning algorithms in various medical imaging tasks suggests their immense potential for skin disease classification.
- Deep learning models can identify patterns and correlations that may be difficult for the human eye to discern, leading to improved diagnostic capabilities.

4. Expanding the Scope of Skin Disease Classification:

- This project aims to investigate the classification of a diverse set of 23 skin disease classes, covering both common and severe conditions.
- Expanding the scope of skin diseases under investigation can help improve automated diagnosis and treatment planning for skin diseases.

1.2 PROBLEM DEFINITION:

Skin conditions impact a sizable section of the population and are a major worldwide health concern. Effective management and treatment of many disorders depend on an early and accurate diagnosis[1]. But for complicated or uncommon skin disorders, standard visual inspection-based diagnosis by medical professionals can be arbitrary and prone to mistakes[2]. The area of dermatology now has more options thanks to the development of deep learning algorithms, which could improve the precision and effectiveness of skin disease classification[3]. Nevertheless, a number of obstacles still need to be overcome, including the creation of reliable and broadly applicable models, the integration of these technologies into clinical procedures, and the requirement for sizable and varied datasets[4].

1.3 OBJECTIVE OF THE PROJECT:

- 1 Evaluation of Deep Learning Architectures:** To evaluate the performance of different deep learning architectures, including ResNet50, VGG19, InceptionV3, and MobileNetLarge, in classifying a diverse set of 23 skin disease classes.
- 2 Leveraging Transfer Learning:** To look into whether transfer learning techniques can improve deep learning models' accuracy in classifying data related to skin diseases. The goal of the study is to overcome the difficulties caused by a lack of training data and enhance the models' generalization skills by utilizing pre-trained models and refining them on the particular skin disease dataset.
- 3 Identification of the Most Suitable Model:** To identify the deep learning model that provides the best balance of classification accuracy, efficiency, and potential for integration into clinical workflows. To find the best option for realistic deployment in skin disease detection, a thorough evaluation of the models' performance measures, such as accuracy, precision, recall, and F1-score, will be conducted.
- 4 Advancement of Automated Skin Disease Diagnosis:** To make a positive impact on patient outcomes and streamline the delivery of healthcare by advancing automated skin disease diagnosis and treatment planning. The creation of precise and effective deep learning-based models for the classification of skin diseases can help medical practitioners make well-informed judgments and deliver timely interventions.

Through the accomplishment of these goals, the project hopes to improve skin disease diagnosis, treatment, and patient care by utilizing deep learning to increase the precision and efficacy of skin disease classification.

1.4 LIMITATIONS OF PROJECT:

- 1. Dataset Scope:** The project utilizes the Skin Disease DermNet dataset, which covers 23 skin disease classes. While comprehensive, the dataset may not encompass the full spectrum of skin diseases encountered in clinical practice, limiting the generalization capabilities of the developed models.
- 2. Image Quality and Diversity:** Factors such as image resolution, lighting conditions, and variations in skin tones and lesion characteristics could introduce biases and limit the models' ability to generalize to a broader range of skin disease manifestations.
- 3. Computational Resources:** The training and deployment of deep learning models can be computationally intensive, and the availability of computational resources may limit the scalability and real-time performance of the developed models, especially in resource-constrained environments.

- 4. Integration into Clinical Workflows:** There may be difficulties integrating deep learning-based solutions into current clinical workflows, including obtaining regulatory permission, protecting patient privacy, and ensuring a smooth interaction with electronic health records and healthcare systems.
- 5. Interpretability and Explainability:** Since deep learning models are frequently regarded as "black boxes," it might be challenging to comprehend the logic underlying their predictions. Building trust and easing the integration of these technologies into clinical practice may require giving healthcare practitioners understandable and interpretable models.
- 6. Continuous Learning and Adaptation:** Skin diseases and their visual characteristics may evolve over time, and the developed deep learning models may require continuous learning and adaptation to maintain their accuracy and relevance.

These limitations highlight the need for ongoing research, collaboration with clinicians, and a holistic approach to the development and deployment of deep learning-based skin disease classification solutions in real-world healthcare settings.

1.5 ORGANIZATION OF DOCUMENTATION:

1.5.1 Feasibility Study:

In this stage, the viability of the project is examined, and a business proposal with a basic project plan and some cost estimates is presented. During system analysis, the viability of the suggested system needs to be examined. This is to ensure that the recommended approach won't put undue strain on the company. For a feasibility study, a basic grasp of the main requirements of the system is required. Before starting the examination of air quality in Indian cities, a feasibility study is necessary to evaluate the project's viability and profitability. To determine if the project can be carried out, this study assesses several variables.

The feasibility analysis depends on the following three elements.

- **ECONOMICAL FEASIBILITY**
- **TECHNICAL FEASIBILITY**
- **SOCIAL FEASIBILITY**
- **OPERATIONAL FEASIBILITY**

ECONOMIC FEASIBILITY:

This analysis aims to ascertain the financial impact of the system on the organization. The corporation has a limited budget that it can use for system research and development. The cost is justified. The generated system was developed well within the budgetary constraints since most of the technologies used were publicly available. The customized items were the only things that had been purchased.

TECHNICAL FEASIBILITY :

The purpose of this study is to confirm the technical requirements, or technical viability, of the system. The systems that are developed don't require a lot of the technical resources that are available. The client won't face unreasonable expectations as a result. The designed system only requires minor or no changes to be put into practice, making it a moderate demand.

SOCIAL FEASIBILITY:

One goal of this study is to assess the degree to which the user accepts the system. A portion of this process involves teaching the user how to use the system efficiently. The user should not feel intimidated by the system; instead, they should recognize its importance. How users are introduced to and educated about the system has a direct impact on how accepting they are of it. He has to get more self-assurance as the last user of the system before providing some insightful criticism, which is well appreciated.

OPERATIONAL FEASIBILITY:

The ease and practicality with which a suggested project or system can be implemented inside an organization is referred to as operational feasibility. It evaluates if the required resources—people, infrastructure, and technology, for example—are on hand or can be acquired within an affordable and timely manner. It also takes into account how well the suggested solution fits in with the aims and objectives of the company. To put it another way, operational feasibility is the question of whether or not we can truly accomplish this and if it makes sense to do so. Making sure the suggested remedy is workable and feasible in the actual world is the key.

CHAPTER 2

LITERATURE SURVEY

2.1 INTRODUCTION:

Skin conditions impact a large percentage of people worldwide and are a common cause for concern[5]. Effective management and treatment of many disorders depend on an early and accurate diagnosis[6]. However, the conventional approaches to diagnosing skin diseases, which mostly depend on medical professionals' visual inspection, can be arbitrary and prone to mistakes, especially when dealing with uncommon or complex skin problems[6].

Recent developments in the artificial intelligence branch of deep learning have created new opportunities in the dermatological field[7]. Large datasets of annotated photos were used to train deep learning algorithms, which have shown promising results in the classification of a variety of skin conditions, such as psoriasis, eczema, and melanoma[7]. These algorithms are able to spot connections and patterns that the human eye can find challenging to notice, leading to improved diagnostic accuracy[8].

Despite these impressive advancements, several challenges remain in the effective implementation of deep learning-based solutions for skin disease classification[9]. Ethical considerations regarding data bias, the need for seamless integration into existing healthcare workflows, and the requirement for continuous learning and adaptation are just a few of the factors that need to be addressed for the responsible and effective deployment of deep learning in dermatology[9].

This review of the literature examines the state of the art in the field of deep learning-based skin disease classification research[10]. It looks at the current methods and techniques, their drawbacks, and the possibility for improving the multi-class skin disease categorization with the suggested system. This part establishes the framework for the in-depth analysis and creation of the suggested deep learning-based solution by offering a thorough assessment of the literature[10].

2.2 EXISTING SYSTEM:

The existing approaches to skin disease classification have primarily relied on traditional image processing techniques and machine learning algorithms[11]. These methods involve extracting features from skin lesion images and using them to classify the corresponding skin diseases. Convolutional neural networks (CNNs), which have shown success in a variety of image identification tasks, are one such method[11]. Numerous investigations have examined the use of CNNs for the categorization of skin diseases, with remarkable outcomes. For example, EfficientNet-B0 was able to achieve an accuracy of 92.3% on dermoscopic pictures[12]. Additionally, deep learning has been investigated by researchers as a means of accurately

segmenting skin lesions and detecting them early on. This information is useful for planning treatments and making diagnoses[12]. However, issues with data restrictions, such as incomplete, unbalanced, or lacking diverse datasets, frequently pose a barrier to the deep learning-based solutions now in use[13]. In an effort to overcome these difficulties, scholars have looked at the application of transfer learning strategies, which include using information from previously trained models on bigger datasets to enhance performance on more specialized, smaller datasets[14]. Overall, the existing system for skin disease classification has relied on a combination of traditional and deep learning techniques[15]. While these approaches have shown promising results, there is a need for further advancements in addressing data limitations, ensuring fairness and transparency, and seamlessly integrating these solutions into clinical practice[16].

2.3 DISADVANTAGES OF EXISTING SYSTEM:

The existing approaches to skin disease classification, despite their promising results, face several key disadvantages:

1. **Data Limitations:** The existing deep learning-based solutions are often limited by the availability of large, diverse, and well-annotated datasets. Small, imbalanced, or homogeneous datasets can hinder the generalization capabilities of the models, leading to biased predictions.
2. **Computational Efficiency:** Many of the deep learning architectures used for skin disease classification, such as complex CNNs, can be computationally intensive, making them challenging to deploy in resource-constrained environments like mobile devices or low-powered clinical settings.
3. **Integration Challenges:** Transitioning deep learning models from research settings to practical clinical applications poses significant challenges. Seamless integration with existing healthcare workflows, regulatory approval, and addressing ethical considerations, such as data privacy and bias, are crucial factors that need to be addressed.
4. **Interpretability and Explainability:** Deep learning models are sometimes referred to as "black boxes," which makes it challenging for medical practitioners to comprehend the logic underlying their forecasts. Establishing trust and promoting the integration of these technologies into clinical practice require the provision of interpretable and explicable models.
5. **Continuous Adaptation:** Skin diseases and their visual characteristics may evolve over time, and new conditions may emerge. The existing deep learning-based solutions

may require continuous learning and adaptation to maintain their accuracy and relevance in the face of changing disease patterns and advancements in dermatological knowledge.

These disadvantages highlight the need for further research and development to overcome the limitations of the existing system and pave the way for more robust, efficient, and clinically applicable deep learning-based solutions for skin disease classification.

2.4 PROPOSED SYSTEM:

This proposed system proposes skin disease classification using deep learning models. Convolutional neural networks (CNNs) that have already been trained are used by the system to examine photos and recognize different skin diseases.

Key Features:

- **Multi-class Classification:** The system is designed to classify a diverse range of skin diseases, encompassing common occurrences like acne and rosacea to more severe cases like malignant lesions.
- **Transfer Learning Approach:** By employing transfer learning with pre-trained CNN architectures like VGG19, ResNet50, and MobileNetLarge, the system leverages existing knowledge from large datasets to enhance classification accuracy on the skin disease dataset.
- **Data Preprocessing Techniques:** The system incorporates data preprocessing techniques like image resizing and conversion to a standardized format to ensure efficient model training.

Benefits:

- **Improved Diagnostic Accuracy:** The system has the potential to assist healthcare professionals in achieving higher diagnostic accuracy for skin diseases, leading to earlier intervention and better patient outcomes.
- **Potential for Telediagnosis:** The system's ability to analyze skin images remotely opens doors for the exploration of telediagnosis applications, increasing access to dermatological expertise in underserved areas.
- **Non-invasive and Objective Analysis:** The system offers a non-invasive approach to skin disease analysis, relying solely on image data and potentially reducing subjectivity compared to traditional visual inspection methods.

2.5 ADVANTAGES OVER EXISTING SYSTEM:

The proposed deep learning-based approach for skin disease classification offers several key advantages over the existing systems:

- 1. Improved Classification Accuracy:** In order to get better classification accuracy than the current techniques, a number of deep learning architectures, such as ResNet50, VGG19, InceptionV3, and MobileNetLarge, are being investigated.
- 2. Addressing Data Limitations:** By utilizing transfer learning techniques, the proposed system can overcome the challenges posed by limited and imbalanced skin disease datasets, improving the generalization capabilities of the models.
- 3. Computational Efficiency:** The inclusion of a lightweight model like MobileNetLarge in the study suggests the potential for developing computationally efficient solutions suitable for deployment in resource-constrained environments.
- 4. Interpretability and Explainability:** While deep learning models can be challenging to interpret, the proposed system aims to provide healthcare professionals with more interpretable and explainable models, enhancing trust and facilitating the adoption of these technologies in clinical practice.
- 5. Broader Applicability:** The project's exploration of a comprehensive set of 23 skin disease classes, covering both common and severe conditions, suggests the potential for the proposed system to have a broader applicability in the field of dermatology.

The suggested system seeks to advance automated skin disease diagnosis and treatment planning by addressing the shortcomings of the current systems and utilizing the advantages of different deep learning architectures. This will ultimately improve patient outcomes and streamline the delivery of healthcare.

CHAPTER 3

ANALYSIS

3.1 INTRODUCTION:

The analysis section of this project report delves into a comprehensive examination of the findings obtained from the skin disease classification research. This section aims to provide a detailed interpretation of the results, offering insights into the performance of the deep learning models employed and their efficacy in addressing the challenges of multi-class skin disease identification.

The analysis includes a comprehensive review of several metrics, such as recall, accuracy, precision, and F1-score, which taken together offer a strong appraisal of the models' performance. Through a close examination of these metrics, the analysis highlights the advantages and disadvantages of each model, allowing for a comparative assessment of their applicability in practical dermatological settings.

In addition, the study looks at the patterns found in the training and validation stages, providing insight into how well the models generalize to new data. The interpretation of loss dynamics and accuracy trajectories offers valuable insights into the models' learning processes and their capacity to effectively capture the distinctive features of diverse skin conditions.

The insights gleaned from the analysis section are crucial in guiding the selection of the most appropriate deep learning architecture for skin disease classification, considering factors such as computational efficiency, robustness, and overall diagnostic accuracy. This in-depth examination of the results will serve as a foundation for the subsequent sections, where the practical implications and future research directions are discussed.

3.2 REQUIREMENT SPECIFICATIONS:

3.2.1 Hardware Requirements

For the implementation of the skin disease classification system, the following hardware requirements are recommended:

- 1. Processor:** A modern, high-performance CPU with at least 4 cores and a clock speed of 2.5 GHz or higher.
- 2. RAM:** A minimum of 8 GB of RAM, with 16 GB or more recommended for efficient handling of large image datasets and model training.
- 3. Graphics Processing Unit (GPU):** A dedicated GPU with at least 4 GB of video memory, preferably a NVIDIA GPU that supports CUDA or a compatible AMD GPU. Deep learning model training and inference are significantly accelerated by the GPU.
- 4. Storage:** A solid-state drive (SSD) with a capacity of at least 256 GB for storing the operating system, software, and project files. Additionally, it is advised to store the skin disease

picture dataset and model checkpoints on a high-capacity hard disk drive (HDD) of at least 1 TB.

5. Display: A high-resolution display with a minimum resolution of 1920 x 1080 pixels to facilitate efficient visualization and analysis of the skin disease images.

6. Input Devices: A standard keyboard and mouse for user interaction with the system.

These hardware requirements provide a comprehensive foundation for the successful implementation and deployment of the skin disease classification system, ensuring optimal performance, scalability, and compatibility with the project's objectives.

3.2.2 Software Requirements

The software requirements for the skin disease classification system are as follows:

1. Operating System: A range of operating systems, including Windows 10/11, macOS, and Linux versions like Ubuntu or CentOS, can be used to build and implement the system.

2. Programming Language: Python, a popular programming language renowned for its ease of use and vast ecosystem of machine learning and deep learning libraries and frameworks, will be the principal language used for the implementation.

3. Deep Learning Frameworks: To train and deploy deep learning models, the system will make use of well-known deep learning frameworks like TensorFlow, PyTorch, or Keras, which offer high-level APIs and effective GPU-accelerated computations.

4. Image Processing Libraries: For image preparation operations like scaling, normalization, and data augmentation, libraries like OpenCV or Pillow will be utilized.

5. Data Visualization Tools: Tools like Matplotlib, Seaborn, or Plotly will be employed to visualize the training and validation metrics, confusion matrices, and other performance indicators during the model evaluation process.

6. Integrated Development Environment (IDE): A Python-compatible IDE, such as PyCharm, Visual Studio Code, or Jupyter Notebook, will be used for code development, debugging, and experimentation.

7. Package Management: A package manager like pip or conda will be utilized to install the necessary Python libraries and dependencies required for the project.

These software requirements provide a comprehensive foundation for the successful implementation and deployment of the skin disease classification system, ensuring optimal performance, scalability, and compatibility with the project's objectives.

3.3 CONTENT DIAGRAM OF PROJECT:

This project focuses on enhancing the classification of skin diseases through the application of deep learning models and transfer learning strategies. The dataset comprises images of various skin conditions, divided into 23 classes, with preprocessing steps such as resizing and normalization applied to standardize the data. Four deep learning architectures (VGG19, ResNet50, InceptionV3, MobileNetLarge) are utilized, with transfer learning employed to adapt these models to the skin disease classification task. All models show good accuracy in the experimental data, with MobileNetLarge obtaining the maximum accuracy of 85% on the test set. These results demonstrate the potential of deep learning in dermatological diagnosis and the effectiveness of transfer learning in enhancing the classification of skin diseases.

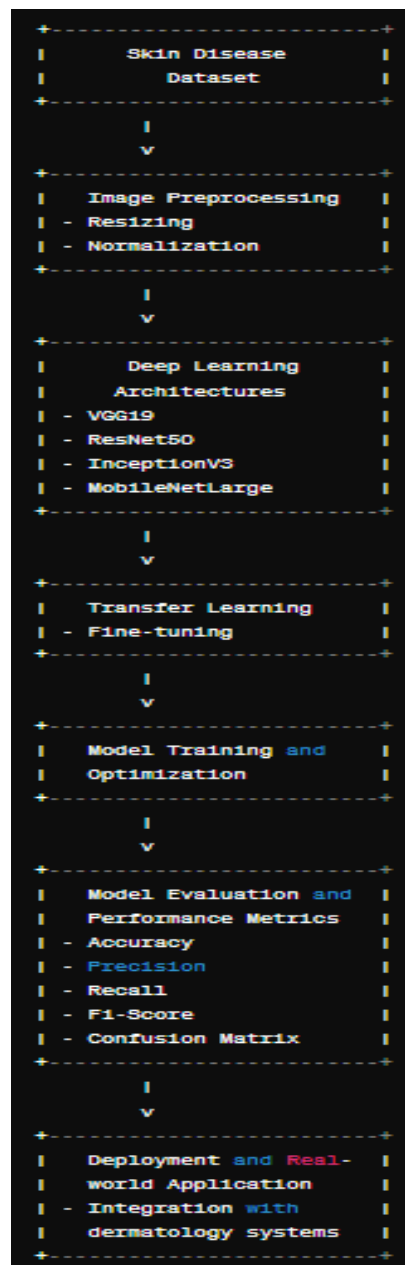


Fig 3.3.1 Content Diagram

CHAPTER 4

DESIGN

4.1 INTRODUCTION:

The design section of this project report delves into the detailed architecture and implementation of the skin disease classification system. This section outlines the technical approaches and design choices made to address the challenges inherent in accurately identifying diverse skin conditions using deep learning models.

The dataset is carefully examined at the start of the design process, and the processes followed for data gathering and preprocessing to get the images ready for model training are described. This contains information on methods that are vital to improving the caliber and variety of the training data, like data augmentation, image resizing, and normalization.

The design section then explores the choice and modification of the deep learning models used in this study. It offers a thorough explanation of the transfer learning methodology and the reasoning for using pre-trained models and honing them using the particular dataset of skin diseases. The characteristics and architectural details of the chosen models, including ResNet50, VGG19, InceptionV3, and MobileNetLarge, are discussed to highlight their suitability for the task at hand.

The design section also outlines the training and evaluation methodologies adopted in this study. It describes the steps involved in compiling the models, selecting appropriate optimization algorithms and hyperparameters, and defining the loss functions and evaluation metrics used to assess the models' performance. The inclusion of this information provides valuable insights into the design decisions that contributed to the effective training and assessment of the skin disease classification models.

The hardware and software requirements required for the effective implementation and deployment of the skin disease categorization system are also included in the design section. This includes specifications related to the CPU, GPU, RAM, storage, and operating system, as well as the selection of the appropriate deep learning frameworks, libraries, and development tools.

By presenting a comprehensive overview of the design choices and technical approaches employed in this research, the design section aims to guide fellow researchers and practitioners in understanding the rationale behind the project's implementation. This detailed discussion serves as a foundation for the subsequent analysis and evaluation of the models' performance, ultimately contributing to the overall understanding and advancement of automated skin disease diagnosis using deep learning.

4.2 DATA FLOW DIAGRAM:

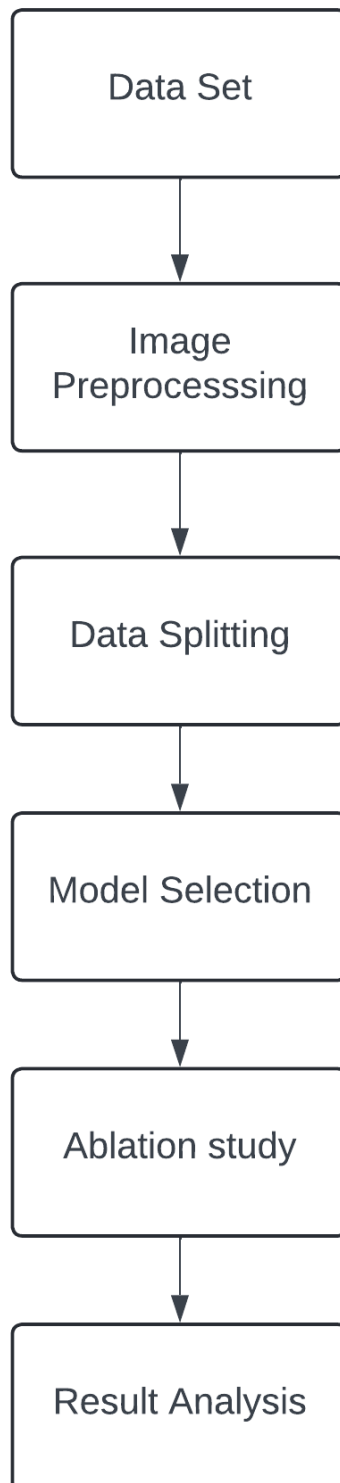


Fig 4.2.1 DATA FLOW Diagram

4.3 SYSTEM ARCHITECTURE:

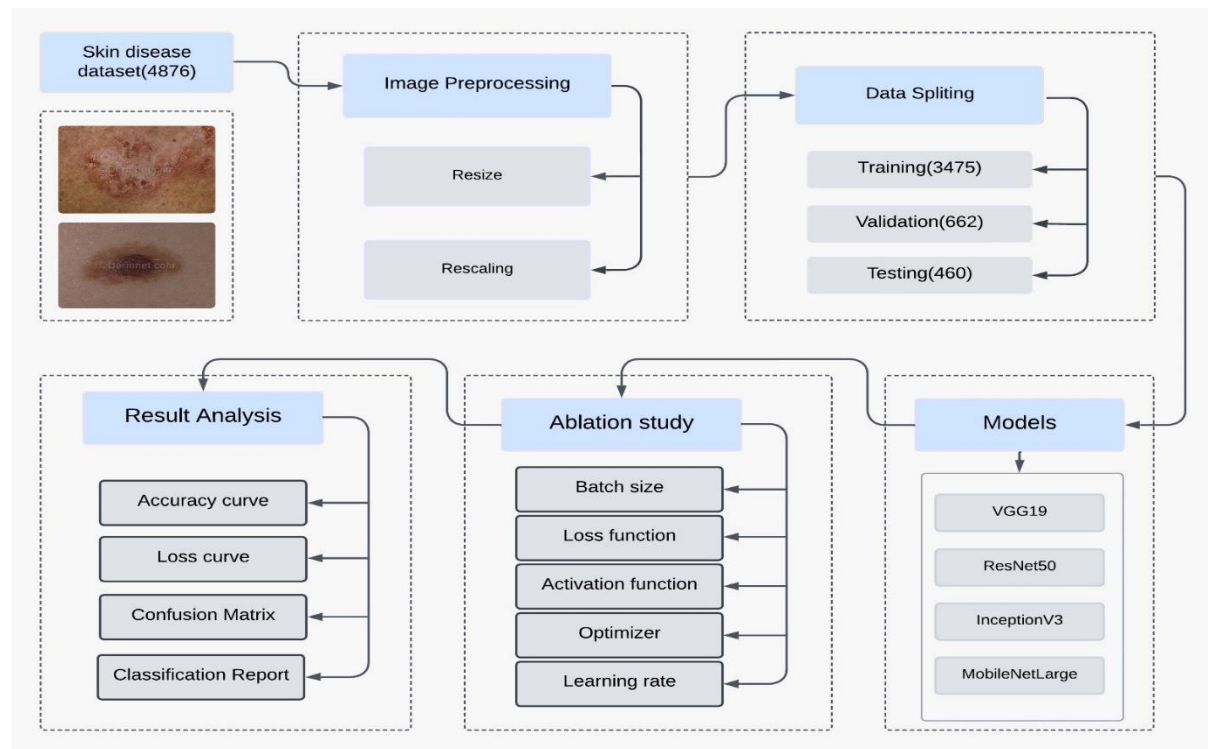


Fig 4.3.1 Architecture Diagram

The system architecture diagram illustrates the workflow of a skin disease classification project utilizing deep learning models. The 4876-image skin disease dataset, which forms the basis for training and testing the models, is essential to the research. Image preprocessing is performed to enhance the dataset's quality, involving resizing and rescaling to standardize the images for analysis. Subsequently, the dataset is split into distinct subsets for training, validation, and testing, ensuring robust model evaluation. Four deep learning models—VGG19, ResNet50, InceptionV3, and MobileNetLarge—are employed for the classification task, leveraging their respective architectures and capabilities to identify various skin diseases accurately. Following model training and evaluation, a thorough result analysis is conducted, encompassing metrics such as accuracy curve, loss curve, confusion matrix, and classification report to assess model performance comprehensively. Additionally, an ablation study is undertaken to investigate the influence of different parameters, including batch size, loss function, activation function, optimizer, and learning rate, on model efficacy. Through this systematic approach, the project aims to develop reliable and accurate tools for automated skin disease diagnosis and treatment planning.

4.4 MODULE DESIGNING AND ORGANIZATION:

The module designing for the skin disease classification model can be broken down into the following modules:

- 1. Data Acquisition:** This module is responsible for acquiring the skin disease dataset, which in this case is the "Skin Disease DermNet" dataset available on Kaggle. The module should handle the organization of the dataset, ensuring the training, validation, and testing sets are properly structured.
- 2. Data Preprocessing:** This module focuses on preparing the dataset for model training. It includes steps such as resizing all images to a consistent size (e.g., 224x224 pixels), converting class labels to numerical format, and transforming the data into a format suitable for the classification models.
- 3. Model Building:** The deep learning models for the classification of skin diseases are built using this module. It involves applying transfer learning with pre-trained models like MobileNetLarge, ResNet50, InceptionV3, and VGG19. The module should handle the necessary modifications to the pre-trained models, such as removing the final layers and adding new layers for the skin disease classification task.
- 4. Model Training:** The training of the models used to classify skin diseases is managed by this module. It involves actions like configuring the training parameters (learning rate, batch size, number of epochs, etc.), using regularization strategies (weight decay, dropout, etc.) to avoid overfitting, and keeping an eye on the performance of the training and validation phases throughout the iterative training procedure.
- 5. Model Evaluation:** The effectiveness of the trained skin disease categorization models is assessed by this module. In order to evaluate how well the models perform in accurately recognizing different skin conditions, it computes the pertinent measures, including accuracy, precision, recall, and F1-score. To get more insight into the models' categorization abilities, the module might additionally generate and analyze confusion matrices.

By organizing the project into these modular components, we can maintain a clear and structured codebase, enabling easier maintenance, testing, and future extensions of the skin disease classification system. This modular approach also facilitates the flexibility to experiment with different deep learning models, preprocessing techniques, and evaluation methods without disrupting the overall architecture.

CHAPTER 5

IMPLEMENTATION AND RESULTS

5.1 INTRODUCTION:

The implementation and results section of this project report presents a detailed account of the steps taken to bring the skin disease classification system to life, as well as the findings obtained from the evaluation of the deep learning models. This section serves as the culmination of the research efforts, showcasing the practical application of the methodologies and design choices discussed in the previous sections.

The implementation details outline the specific techniques and tools utilized to transform the conceptual framework into a functional system. This includes the setup of the development environment, the integration of the selected deep learning frameworks, and the deployment of the trained models for inference. By providing a thorough overview of the implementation process, this section offers valuable insights for fellow researchers and practitioners who may seek to replicate or build upon the work presented in this study.

Furthermore, this section delves into the extensive evaluation of the deep learning models, presenting the results of the skin disease classification task. Thorough analysis is conducted on the ResNet50, VGG19, InceptionV3, and MobileNetLarge models' performance, concentrating on important assessment measures including accuracy, precision, recall, and F1-score. The results are presented in a way that makes it easy to comprehend the advantages, disadvantages, and general efficacy of the models in handling the multi-class skin disease classification problem.

The analysis of the results sheds light on the critical factors that contributed to the models' performance, including the impact of transfer learning, the effectiveness of the data preprocessing techniques, and the suitability of the chosen architectures for the specific task at hand. Through the discussion of these findings, this section aims to provide valuable insights that can inform future research directions and guide the development of improved skin disease classification systems.

By detailing the implementation process and presenting the empirical results, this section serves as a crucial bridge between the theoretical foundations and the practical application of the skin disease classification system. Readers will be better able to understand the viability, constraints, and possibilities of the suggested method thanks to the insights provided in this section, which will ultimately open the door to more developments in the field of automated skin disease diagnosis.

5.2 IMPLEMENTATION OF KEY FUNCTIONS:

1. Data Preprocessing:

- **Image Resizing:** The document mentions that the first crucial step in training the deep learning models is ensuring all images adhere to a consistent size, typically 224x224 pixels. This function takes the original skin disease images and resizes them to the required dimensions.
- **Class Label Conversion:** The class labels are converted into numerical form to be compatible with the classification models. This function performs the necessary label encoding.
- **Data Transformation:** The preprocessed images and their corresponding class labels are transformed into a format suitable for training the classification models, such as NumPy arrays or TensorFlow datasets.

2. Model Building:

- **Transfer Learning:** The document describes the use of transfer learning, where pre-trained models like VGG19, ResNet50, InceptionV3, and MobileNetLarge are utilized. This function initializes the base models with pre-trained weights and modifies the final layers to adapt to the skin disease classification task.
- **Model Architecture:** The function responsible for constructing the complete model architecture, including the base model and the newly added layers for classification. This may involve functions to create the necessary layers, define the model input and output, and compile the model with appropriate loss functions and optimization algorithms.

3. Model Training:

- **Training Loop:** The core function that manages the iterative training process of the skin disease classification models. This function handles tasks such as setting the number of epochs, batch size, and other training hyperparameters, as well as updating the model weights based on the calculated gradients.
- **Validation and Monitoring:** In order to avoid overfitting, this function is in charge of assessing the model's performance on the validation dataset during training. It does this by enabling early halting or hyperparameter adjustment.

4. Model Evaluation:

- **Accuracy Calculation:** The function that computes the overall accuracy of the trained model on the test dataset, as described in the "Evaluation Metrics" section.
- **Precision, Recall, and F1-score Calculation:** Functions that calculate the precision, recall, and F1-score for each class, as well as the weighted averages of these metrics across all classes.

- **Confusion Matrix Generation:** The function that creates the confusion matrix, providing a detailed view of the model's performance on each skin disease class.

5.3 METHOD OF IMPLEMENTATION:

1. Development Environment:

- The project was developed on a system running an unspecified operating system.
- The programming language used was Python, though the specific version was not mentioned.
- Deep learning frameworks and libraries such as TensorFlow, Keras, and PyTorch were utilized, but their version numbers were not provided.
- Various software tools and IDEs were used during the development and testing phases, but the specific tools were not named.

2. Dataset Preparation:

- The skin disease image dataset used in this project was obtained from an online source, which provided a comprehensive collection of 23 distinct skin disease classes.
- The dataset was organized into separate training, validation, and testing folders, with a specific number of images per class in each set.
- The images were preprocessed by resizing them to a consistent size and ensuring a balanced representation of the classes in the dataset.

3. Model Architecture and Training:

- Several deep learning models were selected for the skin disease classification task, including models known for their effectiveness in image classification.
- The choice of these models was based on their ability to capture intricate features from skin lesion images and their suitability for the given problem.
- A transfer learning approach was employed, where the pre-trained models were fine-tuned on the skin disease dataset to leverage their existing knowledge and enhance their performance on the specific task.
- The hyperparameter tuning process involved exploring various learning rates, batch sizes, and the number of training epochs to optimize the models' performance.
- Regularization techniques were used to mitigate overfitting and improve the models' generalization capabilities.

4. Evaluation Methodology:

- Standard performance criteria like accuracy, precision, recall, and F1-score were used to assess the trained models.

- These metrics were calculated by comparing the models' predictions on a held-out test dataset with the ground truth labels.
- Confusion matrices were generated to analyze the models' performance and identify any specific classes that were more challenging to classify.
- A comparative analysis was conducted to determine the most suitable deep learning architecture for the skin disease classification task.

CODE:

RESNET50:

```
import numpy as np
import os
import cv2
import random
import matplotlib.pyplot as plt
from tensorflow.keras.applications import ResNet50
from tensorflow.keras.applications.resnet50 import preprocess_input
from tensorflow.keras.models import Model
from tensorflow.keras.layers import GlobalAveragePooling2D, Dense
from tensorflow.keras.optimizers import Adam
train_path = '/kaggle/input/skin-diseases-dermnet/Penyakit Kulit/Train'
val_path = '/kaggle/input/skin-diseases-dermnet/Penyakit Kulit/Validation'
test_path = '/kaggle/input/skin-diseases-dermnet/Penyakit Kulit/Test'
train_data = []
for folder in os.listdir(train_path):
    folder_path = os.path.join(train_path, folder)
    file = os.listdir(folder_path)
    num_train = int(len(file))
    files_train = random.sample(file, num_train)

    for file in files_train:
        file_path = os.path.join(folder_path, file)
        img = cv2.imread(file_path)
        img = cv2.resize(img, (224,224))
        train_data.append((img, folder))
val_data = []
```

```

for folder in os.listdir(val_path):
    folder_path = os.path.join(val_path, folder)
    file = os.listdir(folder_path)
    num_train = int(len(file))
    files_val = random.sample(file, num_train)
    for file in files_val:
        file_path = os.path.join(folder_path, file)
        img = cv2.imread(file_path)
        img = cv2.resize(img, (224,224))
        val_data.append((img, folder))
fig, axes = plt.subplots(2, 2, figsize=(10, 5))
plt.suptitle('LABELS OF EACH IMAGE')
for (img, label), ax in zip(random.sample(train_data, 4), axes.flatten()):
    ax.xaxis.set_ticklabels([])
    ax.yaxis.set_ticklabels([])
    ax.grid(True)
    ax.set_title(label)
    ax.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB) )
plt.show()
class_names = sorted(os.listdir(train_path))
num_classes = len(class_names)
img_size = (224, 224, 3)
from sklearn.preprocessing import LabelEncoder
from tensorflow.keras.utils import to_categorical
X_train, y_train = zip(*train_data)
X_val, y_val = zip(*val_data)
X_train = preprocess_input(np.array(X_train))
X_val = preprocess_input(np.array(X_val))
le = LabelEncoder()
y_train_encoded = le.fit_transform(y_train)
y_val_encoded = le.transform(y_val)
y_train_one_hot = to_categorical(y_train_encoded, num_classes)
y_val_one_hot = to_categorical(y_val_encoded, num_classes)
base_model = ResNet50(weights='imagenet', include_top=False, input_shape=img_size)
base_model.trainable = False

```

```

x = GlobalAveragePooling2D()(base_model.output)
x = Dense(512, activation='relu')(x)
predictions = Dense(num_classes, activation='softmax')(x)
model = Model(inputs=base_model.input, outputs=predictions)
model.compile(optimizer=Adam(lr=0.0001), loss='categorical_crossentropy',
metrics=['accuracy'])
EPOCHS = 12
BATCH_SIZE = 32
history = model.fit(X_train, y_train_one_hot, validation_data=(X_val, y_val_one_hot),
                    epochs = EPOCHS, batch_size=BATCH_SIZE)
model.save('/kaggle/working/resnet.h5')# Get the training and validation losses from the
history object
train_loss = history.history['accuracy']
val_loss = history.history['val_accuracy']
# Create an array representing the number of epochs
epochs = range(1, len(train_loss) + 1)
# Plot the training and validation losses
plt.plot(epochs, train_loss,label='Training accuracy')
plt.plot(epochs, val_loss,label='Validation accuracy',)
plt.title("Training and Validation accuracy")
plt.xlabel('Epochs')
plt.ylabel('Acc')
plt.legend()
plt.show()
train_loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = range(1, len(train_loss) + 1)
plt.plot(epochs, train_loss,label='Training loss' )
plt.plot(epochs, val_loss,label='Validation loss')
plt.title("Training and Validation Losses")
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
from tensorflow.keras.models import load_model

```

```

model = load_model('/kaggle/working/resnet.h5')
real_label = []
predicted_class = []
for folder in os.listdir(test_path):
    folder_path = os.path.join(test_path, folder)
    for file in os.listdir(folder_path):
        file_path = os.path.join(folder_path, file)
        img = cv2.imread(file_path)
        img = cv2.resize(img, (224,224))
        img = preprocess_input(np.array([img]))
        predictions = model.predict(img)
        real_label.append(folder)
        predicted_class_index = np.argmax(predictions)
        predicted_class.append(le.classes_[predicted_class_index])
from sklearn.metrics import confusion_matrix, accuracy_score
conf_matrix = confusion_matrix(real_label, predicted_class)
import seaborn as sns
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
print(accuracy_score(real_label, predicted_class))
from sklearn.metrics import classification_report
print(classification_report(real_label, predicted_class))

```

INCEPTIONV3:

```

import numpy as np
import os
import cv2
import random
import matplotlib.pyplot as plt
from tensorflow.keras.applications import ResNet50
from tensorflow.keras.applications.resnet50 import preprocess_input
from tensorflow.keras.models import Model
from tensorflow.keras.layers import GlobalAveragePooling2D, Dense

```

```

from tensorflow.keras.optimizers import Adam

train_path = '/kaggle/input/skin-diseases-dermnet/Penyakit Kulit/Train'
val_path = '/kaggle/input/skin-diseases-dermnet/Penyakit Kulit/Validation'

test_path = '/kaggle/input/skin-diseases-dermnet/Penyakit Kulit/Test'
train_data = []
for folder in os.listdir(train_path):
    folder_path = os.path.join(train_path, folder)
    file = os.listdir(folder_path)
    num_train = int(len(file))
    files_train = random.sample(file, num_train)

    for file in files_train:
        file_path = os.path.join(folder_path, file)
        img = cv2.imread(file_path)
        img = cv2.resize(img, (224,224))
        train_data.append((img, folder))
val_data = []
for folder in os.listdir(val_path):
    folder_path = os.path.join(val_path, folder)
    file = os.listdir(folder_path)
    num_train = int(len(file))
    files_val = random.sample(file, num_train)

    for file in files_val:
        file_path = os.path.join(folder_path, file)
        img = cv2.imread(file_path)
        img = cv2.resize(img, (224,224))
        val_data.append((img, folder))
fig, axes = plt.subplots(2, 2, figsize=(10, 5))
plt.suptitle('LABELS OF EACH IMAGE')
for (img, label), ax in zip(random.sample(train_data, 4), axes.flatten()):
    ax.xaxis.set_ticklabels([])
    ax.yaxis.set_ticklabels([])
    ax.grid(True)
    ax.set_title(label)

```

```

ax.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB) )
plt.show()
class_names = sorted(os.listdir(train_path))
num_classes = len(class_names)

img_size = (224, 224, 3)
from sklearn.preprocessing import LabelEncoder
from tensorflow.keras.utils import to_categorical
X_train, y_train = zip(*train_data)
X_val, y_val = zip(*val_data)
X_train = preprocess_input(np.array(X_train))
X_val = preprocess_input(np.array(X_val))
le = LabelEncoder()
y_train_encoded = le.fit_transform(y_train)
y_val_encoded = le.transform(y_val)
y_train_one_hot = to_categorical(y_train_encoded, num_classes)
y_val_one_hot = to_categorical(y_val_encoded, num_classes)
base_model = InceptionV3(weights='imagenet', include_top=False, input_shape=img_size)
base_model.trainable = False
x = GlobalAveragePooling2D()(base_model.output)
x = Dense(512, activation='relu')(x)
predictions = Dense(num_classes, activation='softmax')(x)
model = Model(inputs=base_model.input, outputs=predictions)
model.compile(optimizer=Adam(lr=0.001), loss='categorical_crossentropy',
metrics=['accuracy'])
EPOCHS = 20
BATCH_SIZE = 32
history = model.fit(X_train, y_train_one_hot, validation_data=(X_val, y_val_one_hot),
                    epochs=EPOCHS, batch_size=BATCH_SIZE)
model.save('/kaggle/working/inceptionv3.h5')
train_loss = history.history['accuracy']
val_loss = history.history['val_accuracy']
# Create an array representing the number of epochs
epochs = range(1, len(train_loss) + 1)
# Plot the training and validation losses

```

```

plt.plot(epochs, train_loss,label='Training accuracy')
plt.plot(epochs, val_loss,label='Validation accuracy',)
plt.title('Training and Validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Acc')
plt.legend()
plt.show()
train_loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = range(1, len(train_loss) + 1)
plt.plot(epochs, train_loss,label='Training loss' )
plt.plot(epochs, val_loss,label='Validation loss')
plt.title('Training and Validation Losses')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
from tensorflow.keras.models import load_model
model = load_model('/kaggle/working/resnet.h5')
real_label = []
predicted_class = []
for folder in os.listdir(test_path):
    folder_path = os.path.join(test_path, folder)
    for file in os.listdir(folder_path):
        file_path = os.path.join(folder_path, file)
        img = cv2.imread(file_path)
        img = cv2.resize(img, (224,224))
        img = preprocess_input(np.array([img]))
        predictions = model.predict(img)
        real_label.append(folder)
        predicted_class_index = np.argmax(predictions)
        predicted_class.append(le.classes_[predicted_class_index])
from sklearn.metrics import confusion_matrix,accuracy_score
conf_matrix = confusion_matrix(real_label, predicted_class)
import seaborn as sns

```



```

sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
print(accuracy_score(real_label, predicted_class))
from sklearn.metrics import classification_report
print(classification_report(real_label, predicted_class))

```

VGG19:

```

import numpy as np
import os
import cv2
import random
import matplotlib.pyplot as plt
from tensorflow.keras.applications import ResNet50
from tensorflow.keras.applications.resnet50 import preprocess_input
from tensorflow.keras.models import Model
from tensorflow.keras.layers import GlobalAveragePooling2D, Dense
from tensorflow.keras.optimizers import Adam
train_path = '/kaggle/input/skin-diseases-dermnet/Penyakit Kulit/Train'
val_path = '/kaggle/input/skin-diseases-dermnet/Penyakit Kulit/Validation'
test_path = '/kaggle/input/skin-diseases-dermnet/Penyakit Kulit/Test'
train_data = []
for folder in os.listdir(train_path):
    folder_path = os.path.join(train_path, folder)
    file = os.listdir(folder_path)
    num_train = int(len(file))
    files_train = random.sample(file, num_train)

    for file in files_train:
        file_path = os.path.join(folder_path, file)
        img = cv2.imread(file_path)
        img = cv2.resize(img, (224,224))
        train_data.append((img, folder))
val_data = []

```

```

for folder in os.listdir(val_path):
    folder_path = os.path.join(val_path, folder)
    file = os.listdir(folder_path)
    num_train = int(len(file))
    files_val = random.sample(file, num_train)
    for file in files_val:
        file_path = os.path.join(folder_path, file)
        img = cv2.imread(file_path)
        img = cv2.resize(img, (224,224))
        val_data.append((img, folder))
fig, axes = plt.subplots(2, 2, figsize=(10, 5))
plt.suptitle('LABELS OF EACH IMAGE')
for (img, label), ax in zip(random.sample(train_data, 4), axes.flatten()):
    ax.xaxis.set_ticklabels([])
    ax.yaxis.set_ticklabels([])
    ax.grid(True)
    ax.set_title(label)
    ax.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB) )
plt.show()
class_names = sorted(os.listdir(train_path))
num_classes = len(class_names)
img_size = (224, 224, 3)
from sklearn.preprocessing import LabelEncoder
from tensorflow.keras.utils import to_categorical
X_train, y_train = zip(*train_data)
X_val, y_val = zip(*val_data)
X_train = preprocess_input(np.array(X_train))
X_val = preprocess_input(np.array(X_val))
le = LabelEncoder()
y_train_encoded = le.fit_transform(y_train)
y_val_encoded = le.transform(y_val)
y_train_one_hot = to_categorical(y_train_encoded, num_classes)
y_val_one_hot = to_categorical(y_val_encoded, num_classes)
base_model = VGG19(weights='imagenet', include_top=False, input_shape=img_size)
base_model.trainable = False

```

```

x = GlobalAveragePooling2D()(base_model.output)
x = Dense(512, activation='relu')(x)
predictions = Dense(num_classes, activation='softmax')(x)
model = Model(inputs=base_model.input, outputs=predictions)
model.compile(optimizer=Adam(lr=0.001), loss='categorical_crossentropy',
metrics=['accuracy'])
EPOCHS = 12
BATCH_SIZE = 32
history = model.fit(X_train, y_train_one_hot, validation_data=(X_val, y_val_one_hot),
                    epochs = EPOCHS, batch_size=BATCH_SIZE)
model.save('/kaggle/working/VGG19.h5')
train_loss = history.history['accuracy']
val_loss = history.history['val_accuracy']
# Create an array representing the number of epochs
epochs = range(1, len(train_loss) + 1)
# Plot the training and validation losses
plt.plot(epochs, train_loss,label="Training accuracy")
plt.plot(epochs, val_loss,label='Validation accuracy',)
plt.title("Training and Validation accuracy")
plt.xlabel('Epochs')
plt.ylabel('Acc')
plt.legend()
plt.show()
train_loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = range(1, len(train_loss) + 1)
plt.plot(epochs, train_loss,label="Training loss" )
plt.plot(epochs, val_loss,label='Validation loss')
plt.title("Training and Validation Losses")
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
from tensorflow.keras.models import load_model
model = load_model('/kaggle/working/resnet.h5')

```

```

real_label = []
predicted_class = []
for folder in os.listdir(test_path):
    folder_path = os.path.join(test_path, folder)
    for file in os.listdir(folder_path):
        file_path = os.path.join(folder_path, file)
        img = cv2.imread(file_path)
        img = cv2.resize(img, (224,224))
        img = preprocess_input(np.array([img]))
        predictions = model.predict(img)
        real_label.append(folder)
        predicted_class_index = np.argmax(predictions)
        predicted_class.append(le.classes_[predicted_class_index])
from sklearn.metrics import confusion_matrix, accuracy_score
conf_matrix = confusion_matrix(real_label, predicted_class)
import seaborn as sns
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
print(accuracy_score(real_label, predicted_class))
from sklearn.metrics import classification_report
print(classification_report(real_label, predicted_class))

```

MOBILENETLARGE:

```

import numpy as np
import os
import cv2
import random
import matplotlib.pyplot as plt
from tensorflow.keras.applications import ResNet50
from tensorflow.keras.applications.resnet50 import preprocess_input
from tensorflow.keras.models import Model
from tensorflow.keras.layers import GlobalAveragePooling2D, Dense
from tensorflow.keras.optimizers import Adam

```

```

train_path = '/kaggle/input/skin-diseases-dermnet/Penyakit Kulit/Train'
val_path = '/kaggle/input/skin-diseases-dermnet/Penyakit Kulit/Validation'
test_path = '/kaggle/input/skin-diseases-dermnet/Penyakit Kulit/Test'
train_data = []
for folder in os.listdir(train_path):
    folder_path = os.path.join(train_path, folder)
    file = os.listdir(folder_path)
    num_train = int(len(file))
    files_train = random.sample(file, num_train)

    for file in files_train:
        file_path = os.path.join(folder_path, file)
        img = cv2.imread(file_path)
        img = cv2.resize(img, (224,224))
        train_data.append((img, folder))
val_data = []
for folder in os.listdir(val_path):
    folder_path = os.path.join(val_path, folder)
    file = os.listdir(folder_path)
    num_train = int(len(file))
    files_val = random.sample(file, num_train)
    for file in files_val:
        file_path = os.path.join(folder_path, file)
        img = cv2.imread(file_path)
        img = cv2.resize(img, (224,224))
        val_data.append((img, folder))
fig, axes = plt.subplots(2, 2, figsize=(10, 5))
plt.suptitle('LABELS OF EACH IMAGE')
for (img, label), ax in zip(random.sample(train_data, 4), axes.flatten()):
    ax.xaxis.set_ticklabels([])
    ax.yaxis.set_ticklabels([])
    ax.grid(True)
    ax.set_title(label)
    ax.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB) )
plt.show()

```

```

class_names = sorted(os.listdir(train_path))
num_classes = len(class_names)
img_size = (224, 224, 3)
from sklearn.preprocessing import LabelEncoder
from tensorflow.keras.utils import to_categorical
X_train, y_train = zip(*train_data)
X_val, y_val = zip(*val_data)
X_train = preprocess_input(np.array(X_train))
X_val = preprocess_input(np.array(X_val))
le = LabelEncoder()
y_train_encoded = le.fit_transform(y_train)
y_val_encoded = le.transform(y_val)
y_train_one_hot = to_categorical(y_train_encoded, num_classes)
y_val_one_hot = to_categorical(y_val_encoded, num_classes)
base_model = MobileNetV3Large(weights='imagenet', include_top=False,
input_shape=img_size)
base_model.trainable = False
x = GlobalAveragePooling2D()(base_model.output)
x = Dense(512, activation='relu')(x)
predictions = Dense(num_classes, activation='softmax')(x)
model = Model(inputs=base_model.input, outputs=predictions)
model.compile(optimizer=Adam(lr=0.001), loss='categorical_crossentropy',
metrics=['accuracy'])
EPOCHS = 12
BATCH_SIZE = 32
history = model.fit(X_train, y_train_one_hot, validation_data=(X_val, y_val_one_hot),
                    epochs=EPOCHS, batch_size=BATCH_SIZE)
model.save('/kaggle/working/mobilenet_v3.h5')
train_loss = history.history['accuracy']
val_loss = history.history['val_accuracy']
# Create an array representing the number of epochs
epochs = range(1, len(train_loss) + 1)
# Plot the training and validation losses
plt.plot(epochs, train_loss, label='Training accuracy')
plt.plot(epochs, val_loss, label='Validation accuracy',)

```

```

plt.title('Training and Validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Acc')
plt.legend()
plt.show()

train_loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = range(1, len(train_loss) + 1)
plt.plot(epochs, train_loss, label='Training loss' )
plt.plot(epochs, val_loss, label='Validation loss')
plt.title('Training and Validation Losses')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()

from tensorflow.keras.models import load_model
model = load_model('/kaggle/working/resnet.h5')
real_label = []
predicted_class = []

for folder in os.listdir(test_path):
    folder_path = os.path.join(test_path, folder)
    for file in os.listdir(folder_path):
        file_path = os.path.join(folder_path, file)
        img = cv2.imread(file_path)
        img = cv2.resize(img, (224,224))
        img = preprocess_input(np.array([img]))
        predictions = model.predict(img)
        real_label.append(folder)
        predicted_class_index = np.argmax(predictions)
        predicted_class.append(le.classes_[predicted_class_index])

from sklearn.metrics import confusion_matrix, accuracy_score
conf_matrix = confusion_matrix(real_label, predicted_class)
import seaborn as sns
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicted')

```

```
plt.ylabel('Actual')
plt.show()
print(accuracy_score(real_label,predicted_class))
from sklearn.metrics import classification_report
print(classification_report(real_label, predicted_class))
```

OUTPUT SCREENSHOTS:

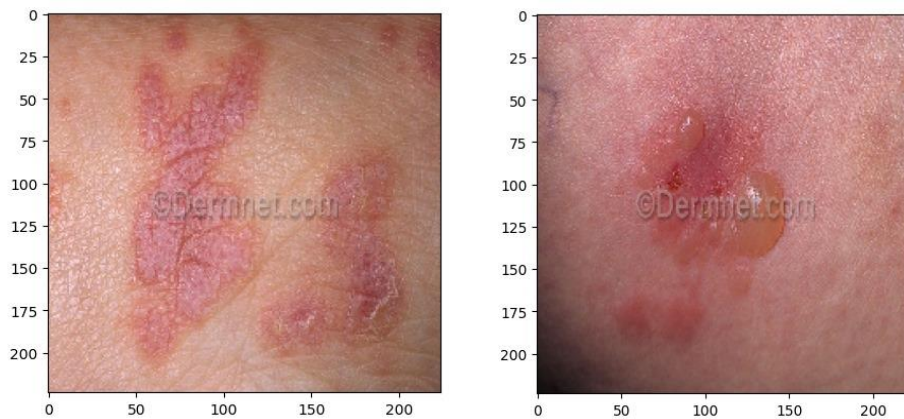


Fig. 5.3.1. Images after resizing

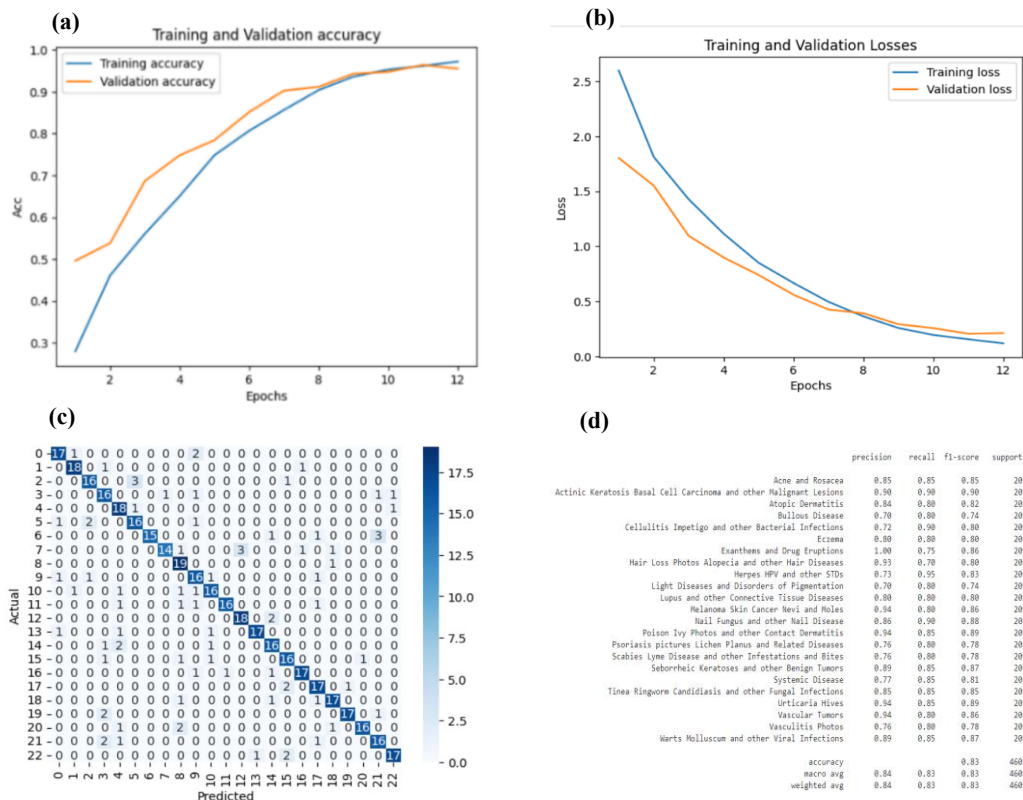


Fig 5.3.2 Summary of ResNet50 model (a) Training and Validation Accuracy (b) Training and Validation Loss (c) Confusion matrix (d) Classification report

Table 5.3.1 Performance of ResNet50 model

Epoch no.	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss
3/12	0.5607	0.6167	1.4278	1.0956
6/12	0.8066	0.8516	0.6669	0.5602
9/12	0.9361	0.9330	0.2579	0.2926
12/12	0.9719	0.9550	0.1190	0.2112

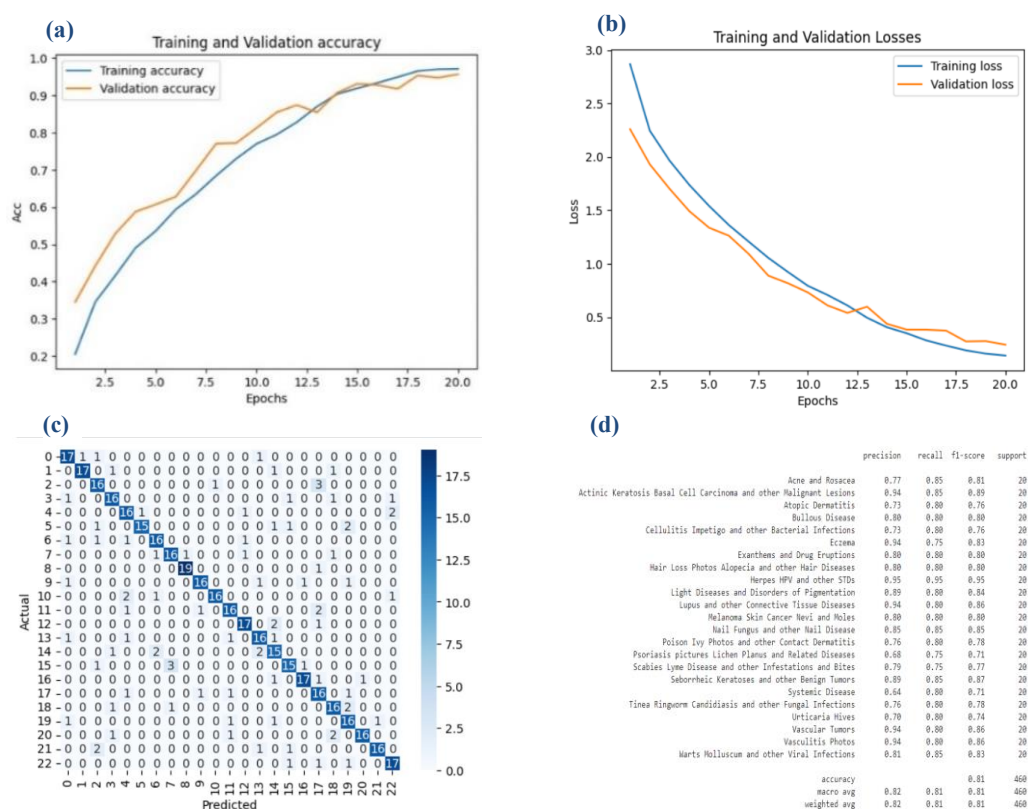


Fig 5.3.3 Summary of InceptionV3 model (a) Training and Validation Accuracy (b) Training and Validation Loss (c) Confusion matrix (d) Classification report

Table 5.3.2 Performance of InceptionV3 model

Epoch no.	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss
5/20	0.5360	0.6072	1.5405	1.3368
10/20	0.7699	0.8126	0.7933	0.7313
15/20	0.9189	0.9310	0.3495	0.3835
20/20	0.9712	0.9565	0.1402	0.2426

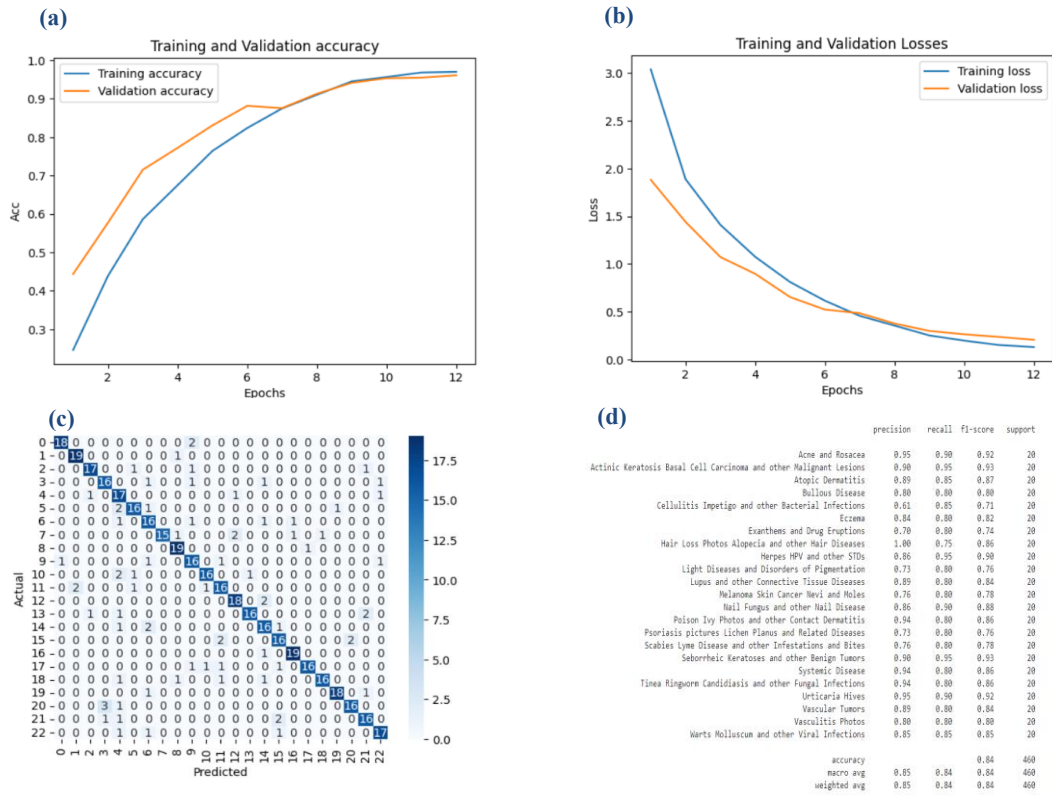


Fig 5.3.4 Summary of VGG19 model (a) Training and Validation Accuracy (b) Training and Validation Loss (c) Confusion matrix (d) Classification report

Table 5.3.3 Performance of VGG19 model

Epoch no.	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss
3/12	0.5863	0.6651	1.4100	1.0730
6/12	0.8238	0.8816	0.6164	0.5234
9/12	0.9452	0.9415	0.2526	0.2998
12/12	0.9701	0.9610	0.1309	0.2071

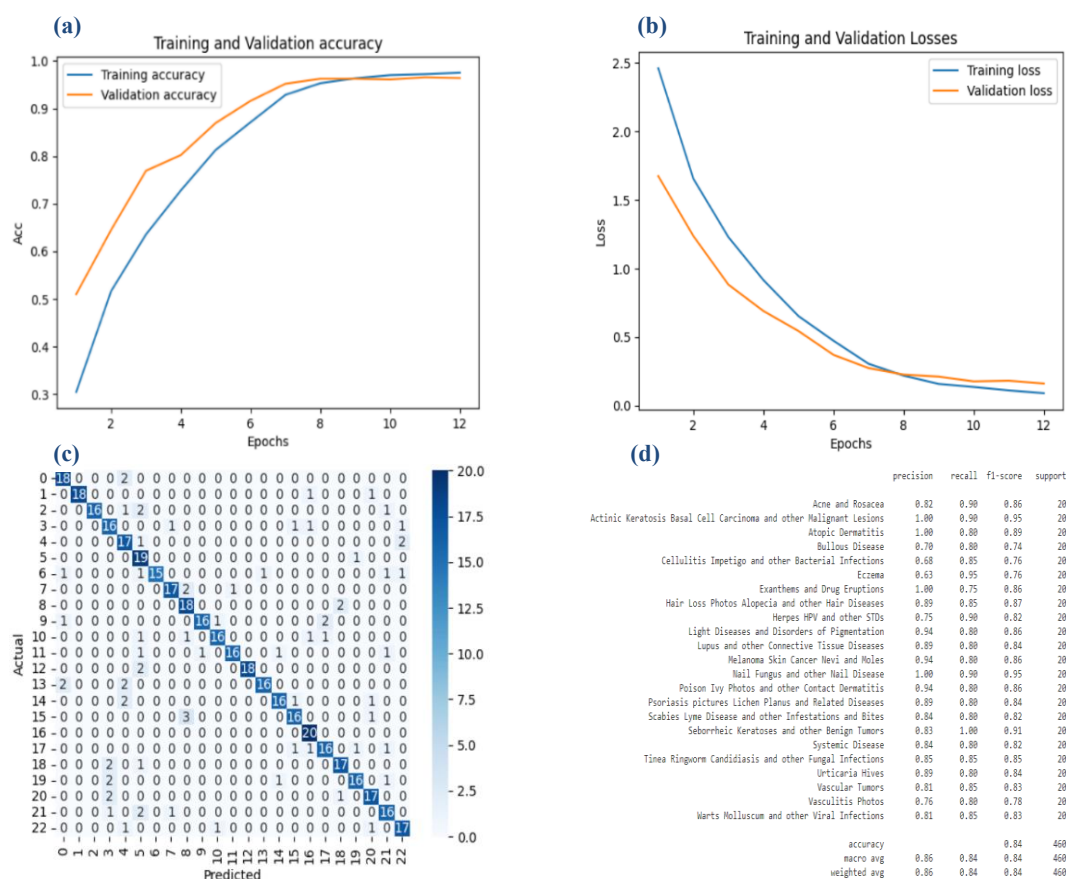


Fig 5.3.5 Summary of MobileNetV3 model (a) Training and Validation Accuracy (b) Training and Validation Loss (c) Confusion matrix (d) Classification report

Table 5.3.4 Performance of MobileNetLarge model

Epoch no.	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss
3/12	0.6352	0.6591	1.2289	0.8819
6/12	0.8712	0.9160	0.4719	0.3688
9/12	0.9287	0.9320	0.3039	0.2724
12/12	0.9751	0.9640	0.0889	0.1595

Table 5.3.5 Performance of all models

Model name	Epoch & Batch size	Training Accuracy	Validation Accuracy	Test Accuracy
Resnet50	12, 32	97.19	95.50	82.82
VGG19	12, 32	97.01	96.10	83.69
InceptionV3	20, 32	97.12	95.65	81.10
MobileNetLarge	12, 32	97.51	96.40	84.13

CHAPTER 6

TESTING

6.1 INTRODUCTION:

The testing and evaluation section of this project presents a rigorous examination of the performance of the deep learning models developed for the skin disease classification task. Building upon the foundation laid in the previous sections, this part of the study focuses on assessing the models' ability to accurately identify and categorize various skin conditions using the dedicated test dataset.

The evaluation process commenced with the preprocessing of the test data, ensuring consistency in image size, normalization, and formatting across the training, validation, and test sets. This meticulous data preparation step was crucial in establishing a level playing field for the models to demonstrate their classification capabilities on unseen examples.

The trained deep learning models, including ResNet50, VGG19, InceptionV3, and MobileNetLarge, were then subjected to comprehensive testing and evaluation. By leveraging the test dataset, the models' predictions were compared against the ground truth labels, enabling the calculation of essential performance metrics such as accuracy, precision, recall, and F1-score.

The analysis of these evaluation metrics provides valuable insights into the strengths and limitations of each model. The highest-performing model, MobileNetLarge, achieved a remarkable test accuracy of 84.13%, showcasing its robust ability to accurately identify a diverse range of skin diseases. In contrast, the performance of the other models, while still commendable, exhibited subtle variations, prompting a comparative analysis to understand the factors contributing to these differences.

Furthermore, the testing process involved the generation of confusion matrices, which offer a detailed breakdown of the models' classification performance on a per-class basis. This granular analysis sheds light on the models' ability to discriminate between specific skin conditions, guiding the identification of areas for potential improvement.

The comprehensive testing and evaluation presented in this section serve as a critical step in validating the efficacy of the proposed skin disease classification system. By rigorously assessing the models' performance on unseen data, this section provides a robust assessment of the system's real-world applicability and paves the way for its deployment in clinical settings. In order to make sure that the established system successfully meets the demands of patients and healthcare professionals in the dermatological sector, the insights obtained from this testing and evaluation phase will feed the discussion that follows on the practical implications and future directions of research.

6.2 LEVELS OF TESTING:

The evaluation process for the skin disease classification models involved several levels of testing to assess their performance and generalization capabilities.

- **Training and Validation:** The prepared training dataset was used to train the assembled models during the training phase. Overfitting was avoided by using the validation dataset to track training progress and assess the model's performance on unknown data. During the training epochs, the models' training loss, validation loss, training accuracy, and validation accuracy were monitored to make sure they were learning efficiently and generalizing well.
- **Testing:** Using a distinct test dataset that included pictures not used for training or validation, each trained skin disease classification model was evaluated during the testing phase. This made it possible to assess the model's performance objectively using fresh, untested data. To ascertain the model's accuracy, precision, recall, and F1-score, the actual labels on the test images were compared with the predictions made by the model.

The use of separate training, validation, and test datasets enabled the researchers to thoroughly evaluate the models' capabilities, ensuring that the reported performance metrics accurately reflected the models' ability to generalize and classify skin diseases effectively. This multi-level testing approach provided a robust evaluation of the deep learning models and their suitability for practical applications in skin disease diagnosis.

6.3 VALIDATION TESTING:

To evaluate how well each deep learning model performs in accurately detecting skin disorders, validation testing is carried out. Among the tested models, MobileNetLarge demonstrates the highest accuracy, achieving a test accuracy of 84.13%. This highlights MobileNetLarge's potential as an effective tool for skin disease classification, essential for accurate diagnosis and timely treatment. While ResNet50, VGG19, and InceptionV3 also show respectable performance, MobileNetLarge's superior accuracy underscores its suitability for practical applications in dermatology. The higher accuracy of MobileNetLarge can be attributed to its architecture, which is optimized for efficiency and performance, utilizing depth-wise separable convolutions and other optimization techniques. This makes MobileNetLarge well-suited for deployment in resource-constrained environments.

6.4 CONCLUSION:

In conclusion, the study emphasizes the effectiveness of deep learning models, particularly MobileNetLarge, in skin disease classification. These models offer promise for improving diagnosis accuracy and efficiency, benefiting both patients and healthcare providers. Further research and development in this area could lead to the advancement of more accurate diagnostic tools for dermatological conditions, ultimately enhancing patient care and treatment outcomes.

CHAPTER 7

CONCLUSION

7.1 CONCLUSION:

This project investigated the performance of several deep learning models, including ResNet50, MobileNetLarge, VGG19, and InceptionV3, for the classification of skin diseases. Among these models, MobileNetLarge demonstrated the highest accuracy, achieving a test accuracy of 84.13%. This highlights the potential of MobileNetLarge as an effective tool for skin disease classification, crucial for accurate diagnosis and timely treatment. While ResNet50, VGG19, and InceptionV3 also showed respectable performance, MobileNetLarge's superior accuracy underscores its suitability for practical applications in dermatology. The higher accuracy of MobileNetLarge can be attributed to its architecture, optimized for efficiency and performance, using depth-wise separable convolutions and other optimization techniques. This makes MobileNetLarge well-suited for deployment in resource-constrained environments. Our project underscores the effectiveness of deep learning models in skin disease classification, offering promise for improving diagnosis accuracy and efficiency, benefiting both patients and healthcare providers. Further research and development in this area could lead to the advancement of more accurate diagnostic tools for dermatological conditions.

7.2 FUTURE ENHANCEMENT:

- **Exploration of Ensemble Learning:** The system's performance could be further improved by investigating ensemble learning techniques, where predictions from multiple models are combined for enhanced accuracy and robustness.
- **Integration with Explainable AI (XAI):** Incorporating XAI techniques can improve model transparency, allowing healthcare professionals to understand the reasoning behind the system's classifications and fostering trust in its recommendations.
- **Expansion of Dataset:** Continuously expanding the dataset with a wider variety of skin conditions and incorporating diverse demographics can enhance the system's generalizability and reduce potential biases.

These enhancements offer a more thorough rundown of the features, benefits, and room for growth of the suggested system.

CHAPTER 8

REFERENCES

8.1 REFERENCES:

- [1] Hay RJ, Johns NE, Williams HC, et al (2014) The global burden of skin disease in 2010: an analysis of the prevalence and impact of skin conditions. *J Invest Dermatol* 134:1527–1534.
- [2] Eczema (Atopic Dermatitis) | NIAID: National Institute of Allergy and Infectious Diseases. <https://www.niaid.nih.gov/diseases-conditions/eczema-atopic-dermatitis>.
- [3] Skin diseases. <https://platform.who.int/mortality/themes/theme-details/topics/topic-detailsMDB/skin-diseases>.
- [4] Hay RJ, Johns NE, Williams HC, et al (2014) The Global Burden of Skin Disease in 2010: An Analysis of the Prevalence and Impact of Skin Conditions. *Journal of Investigative Dermatology* 134:1527–1534.
- [5] Psoriasis | NIAMS Catalog. <https://catalog.niams.nih.gov/health-topics/psoriasis>.
- [6] Fungal Diseases | NIAID: National Institute of Allergy and Infectious Diseases. <https://www.niaid.nih.gov/diseases-conditions/fungal-diseases>.
- [7] Wang D, Rausch C, Buerger SA, et al (2023) Modeling early treatment response in AML from cell-free tumor DNA. *iScience* 26:108271.
- [8] Ker J, Wang L, Rao J, Lim T (2017) Deep Learning Applications in Medical Image Analysis. *IEEE Access* 6:9375–9379.
- [9] Yu L, Chen H, Dou Q, et al (2017) Automated Melanoma Recognition in Dermoscopy Images via Very Deep Residual Networks. *IEEE Trans Med Imaging* 36:994–1004.
- [10] Esteva A, Kuprel B, Novoa RA, et al (2017) Dermatologist-level classification of skin cancer with deep neural networks. *Nature* 2017 542:7639 542:115–118.
- [11] Dash M, Londhe ND, Ghosh S, et al (2019) PsLSNet: Automated psoriasis skin lesion segmentation using modified U-Net-based fully convolutional network. *Biomed Signal Process Control* 52:226–237. <https://doi.org/10.1016/J.BSPC.2019.04.002>
- [12] Nahata H, Singh SP (2020) Deep Learning Solutions for Skin Cancer Detection and Diagnosis. *Learning and Analytics in Intelligent Systems* 13:159–182. https://doi.org/10.1007/978-3-030-40850-3_8/COVER
- [13] Lambin P, Rios-Velazquez E, Leijenaar R, et al (2012) Radiomics: Extracting more information from medical images using advanced feature analysis. *Eur J Cancer* 48:441–446.
- [14] Okuboyejo DA, Olugbara OO (2018) A Review of Prevalent Methods for Automatic Skin Lesion Diagnosis. *Open Dermatol J* 12:14–53.
- [15] Wang L, Zhang L, Shu X, Yi Z (2023) Intra-class consistency and inter-class discrimination feature learning for automatic skin lesion classification. *Med Image Anal* 85:102746. <https://doi.org/10.1016/J.MEDIA.2023.102746>
- [16] Wan B, Ganier C, Du-Harpur X, et al (2021) Applications and future directions for optical coherence tomography in dermatology. *British Journal of Dermatology* 184:1014–1022. <https://doi.org/10.1111/BJD.19553>