

DERMATOLOGICAL DISEASE DETECTION

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering degree in Computer Science and Engineering
by

GAYATHRI PENNA (Reg No- 40110936)

ADITHYA VARMA RUDRARAJU (Reg No- 40110032)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

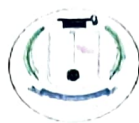
Category – 1 University by UGC

Accredited with Grade “A++” by NAAC|12B Status by UGC| Approved by AICTE

JEPPIAAR NAGAR, RAJIV GANDHISALAI,

CHENNAI – 600119

April-2024



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Category - I University by UGC

Accredited "A++" by NAAC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **GAYATHRI PENNA (40110936)** who carried out Project entitled **"DERMATOLOGICAL DISEASE DETECTION"** under my supervision from November 2023 to April 2024.

Internal Guide

Ms. V. SURYA M.E., (Ph.D.)

HEAD OF THE DEPARTMENT
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Dr. L. LAKSHMANAN M.E., Ph.D.
Department of Computer Science and Engineering
Chennai - 600 119.

Submitted for Viva voce Examination held on 15-04-2024

Internal Examiner

External Examiner

DECLARATION

I, GAYATHRI PENNA (Reg No- 40110936) hereby declare that the Project Report entitled "DERMATOLOGICAL DISEASE DETECTION" done by me under the guidance of Ms. V. SURYA M.E., (Ph.D.) is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in **Computer Science and Engineering**

DATE: 15-04-2024

Gayathri P

PLACE: Chennai

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **The Board of Management of Sathyabama Institute of Science and Technology** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph. D, Dean**, School of Computing, **Dr. L. Lakshmanan M.E., Ph.D.**, Head of the Department of Computer Science and Engineering for providing me with the necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Ms. V. Surya M.E., (Ph.D.)** for her valuable guidance, suggestions, and constant encouragement that paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Dermatological diseases present a formidable challenge in healthcare, impacting millions worldwide and necessitating accurate diagnosis for effective management. This study explores the fusion of image processing techniques and Convolutional Neural Networks (CNNs) to refine the accuracy of dermatological disease detection. With a vast dataset covering diverse skin conditions, the model is trained and validated, employing preprocessing methods like color normalization, resizing, and noise reduction to enhance image quality. Through a CNN architecture, relevant features are extracted from processed images, enabling classification into specific disease categories. Performance metrics such as accuracy, precision, recall, and F1-score are utilized to evaluate the model's efficacy.

The integration of image processing techniques significantly enhances the accuracy of dermatological disease detection. By preprocessing images to standardize color, resize for consistency, and reduce noise, the model receives cleaner input data, improving its ability to discern critical features. Leveraging a CNN architecture allows for the extraction of intricate patterns and textures inherent in dermatological images, enabling precise classification. This amalgamation of techniques results in a robust diagnostic tool, surpassing conventional methods in accuracy levels.

The CNN model demonstrates remarkable diagnostic capabilities, showcasing its potential to revolutionize dermatological disease detection. Its ability to accurately classify skin conditions enables timely intervention, leading to improved patient outcomes. By leveraging artificial intelligence, this approach addresses the pressing need for early detection, facilitating prompt treatment and reducing disease burden. In conclusion, the integration of image processing techniques and CNNs presents a promising approach to enhance the accuracy of dermatological disease detection. By leveraging a comprehensive dataset and sophisticated algorithms, this methodology demonstrates superior performance compared to traditional methods. The ability to accurately diagnose skin conditions has profound implications for patient care, enabling timely intervention and improving clinical outcomes. Continued research and development in this field hold the potential to revolutionize dermatological diagnostics, paving the way for more effective disease management and improved quality of life for patients worldwide.

TABLE OF CONTENTS

Chapter No	TITLE	Page No
	ABSTRACT	
	LIST OF FIGURES	
1	INTRODUCTION	1
2	LITERATURE SURVEY	3
	2.1 Inferences from Literature Survey	3
	2.2 Existing System and Proposed System	4
	2.3 Open problems in Existing System	7
3	REQUIREMENTS ANALYSIS	9
	3.1 Risk Analysis of the project	9
	3.2 Software Requirements specification document	10
4	DESCRIPTION OF PROPOSED SYSTEM	12
	4.1 Flow chart	12
	4.2 Selected Methodology	15
	4.3 Architecture	21
	4.4 Description of Software for Implementation	22
	4.5 Project Management Plan	28
5	IMPLEMENTATION DETAILS	29
6	RESULT AND DISCUSSION	32
7	CONCLUSION	33
	7.1 Conclusion	33
	7.2 Future Work	34
	REFERENCES	36
	APPENDIX	
	A.SOURCE CODE	38
	B.SCREENSHOTS	47
	C.RESEARCH PAPER	52
	D.CERTIFICATE	60

LIST OF FIGURES

Fig.4.1	FLOW CHART	14
Fig.4.2	SYSTEM ARCHITECTURE	21
Fig.B.1	CODE FOR WEB PAGE	47
Fig.B.2	INPUT PAGE	47
Fig.B.3	SAMPLE OUTPUT PAGE 1	48
Fig.B.4	SAMPLE OUTPUT PAGE 2	48
Fig.B.5	VISUALIZATION	49
Fig.B.6	SAMPLE IMAGES	49
Fig.B.7	COMPARISON OF LOSS	50
Fig.B.8	COMPARISON OF ACCURACY	50
Fig.B.9	PREDITED IMAGES	51

CHAPTER 1

INTRODUCTION

Dermatological diseases constitute a broad spectrum of skin conditions affecting millions globally, necessitating timely and accurate diagnosis for effective treatment and prevention of complications. Traditionally, dermatologists rely on clinical expertise and visual examination for diagnosis, but emerging technologies like image processing and machine learning offer potential enhancements in diagnostic accuracy and efficiency.

In recent years, medical image analysis has seen significant progress, with dermatology benefiting from these advancements. By leveraging image processing and Convolutional Neural Networks (CNNs), researchers have developed automated systems for detecting and classifying dermatological conditions. These systems serve to support dermatologists, particularly in complex cases or when handling large volumes of images.

This study aims to explore the application of image processing techniques and CNN models for dermatological disease detection. Utilizing a comprehensive dataset encompassing various dermatological conditions, the objective is to evaluate the accuracy and effectiveness of the proposed model. Integral to the methodology is the integration of image preprocessing techniques such as color normalization, resizing, and noise reduction, aimed at enhancing the quality of input images.

The primary goal of this research is to develop a CNN-based model capable of accurately classifying dermatological conditions, providing dermatologists with an additional diagnostic tool. Model performance assessment will be based on a range of metrics including accuracy, precision, recall, and F1-score, essential for evaluating its diagnostic capabilities. The significance of this study lies in its potential to improve early detection and diagnosis of dermatological diseases, ultimately leading to more effective and timely treatments.

This research contributes to the ongoing exploration of artificial intelligence in healthcare and its role in assisting medical professionals, thereby reducing the burden on healthcare systems and improving patient outcomes. In the subsequent sections, we will provide detailed insights into the methodology, dataset, experimental setup, results, and discussion, offering a comprehensive understanding of the capabilities and limitations of the proposed approach for dermatological disease detection using image processing and CNNs..

CHAPTER 2

LITERATURE SURVEY

2.1 INFERENCES FROM LITERATURE SURVEY

In the realm of dermatology, various cutting-edge approaches have emerged to tackle the challenges associated with diagnosing and classifying skin diseases. One such advancement is the utilization of Convolutional Neural Networks (CNNs) in machine learning models, as highlighted in a study published in IEEE Access in 2022 by Viswanatha Reddy Allugunti and colleagues. Their work focuses on leveraging deep learning techniques to develop a model capable of accurately diagnosing melanoma at its early stages, with the ultimate goal of improving survival rates. By harnessing the power of CNNs, which are well-suited for image analysis tasks, the proposed model demonstrates promising potential in enhancing the timely identification and classification of various skin conditions.

In a complementary approach, Gan Cai et al. introduced a novel neural network architecture called multimodal Transformer in a paper published in The Visual Computer journal in 2023. This innovative model integrates both image data and clinical metadata, aiming to provide a holistic approach to skin disease classification. By fusing multiple modalities, such as visual images and patient-related information, the multimodal Transformer seeks to improve the accuracy and robustness of disease classification systems, thus offering a comprehensive solution for dermatological diagnosis.

Furthermore, the advancement of Distributed Artificial Intelligence as a Service (DAIaaS) in the domain of dermatology is exemplified in the work of Nourah Janbi et al. The authors proposed a reference architecture named Imtidad in 2022, which is designed to facilitate the deployment of AI-driven skin disease diagnosis services across cloud, fog, and edge computing environments. Imtidad aims to provide scalable and efficient AI services, ensuring timely and accurate diagnosis of skin conditions by leveraging distributed computing resources. This architecture represents a significant step towards democratizing access to advanced diagnostic tools in dermatology, thereby improving healthcare outcomes for patients worldwide.

In addition to leveraging advanced neural network architectures, researchers have also explored the potential of deep learning methods for prognosis and forecasting of skin diseases. For instance, Pravin R. Kshirsagar et al. proposed the use of deep learning techniques, including MobileNetV2 and LSTM, in constructing frameworks for automated skin disease prognosis. Their work, published in 2022, emphasizes the importance of accurate forecasting in dermatology, as it enables early intervention and personalized treatment strategies. By harnessing the capabilities of deep learning, such frameworks hold promise for enhancing prognostic accuracy and improving patient outcomes.

Moreover, the application of deep learning in neglected tropical diseases affecting the skin has garnered attention in recent years. In a pilot study conducted in 2023 by Rie R. Yotsu et al., a deep learning model was developed for AI-based diagnosis of such diseases, including acne, cherry angioma, melanoma, and psoriasis. By employing image processing techniques and machine learning algorithms, the model aims to facilitate early detection and treatment of neglected tropical diseases, thereby addressing critical healthcare challenges in resource-constrained settings. This research underscores the potential of deep learning approaches in expanding access to dermatological care and improving health equity globally.

2.2 EXISTING SYSTEM AND PROPOSED SYSTEM

Existing System:

The existing system may involve traditional methods of skin disease diagnosis, which often rely on visual examination by dermatologists, medical professionals. These methods can be subjective and may have limitations in accurately identifying rare or complex skin conditions.

With the increasing prevalence of skin diseases and the advancements in machine learning, numerous systems have been developed to assist dermatologists in accurate diagnosis. Convolutional Neural Networks (CNNs) have emerged as a powerful tool in this domain due to their ability to learn hierarchical features directly from image data. Several existing systems utilize CNNs for skin disease detection, showcasing promising results in automating the diagnostic process.

DermNetNZ : One notable system is DermNetNZ, which employs CNNs for skin disease classification. DermNetNZ is a publicly available online dermatology atlas containing a vast collection of images depicting various skin conditions. CNNs trained on this dataset have demonstrated high accuracy in identifying skin diseases, providing valuable support to dermatologists in their diagnostic endeavors. The system's accessibility and extensive image database make it a valuable resource for both medical professionals and patients seeking information on skin diseases.

Skin Cancer Detection Systems: Several research groups have developed CNN-based systems specifically tailored for skin cancer detection. These systems typically utilize datasets consisting of dermatoscopic images captured using specialized imaging devices. By training CNNs on these datasets, researchers have achieved remarkable success in discriminating between benign and malignant skin lesions. These systems not only aid dermatologists in early detection but also hold the potential to reduce the workload associated with manual inspection of dermatoscopic images.

Mobile Applications: In recent years, there has been a surge in the development of mobile applications leveraging CNNs for skin disease detection. These applications allow users to upload images of their skin lesions, which are then analyzed using pre-trained CNN models. The applications provide users with immediate feedback on the likelihood of their lesions being indicative of certain skin diseases, prompting them to seek medical attention if necessary. While these applications serve as convenient tools for self-assessment, their accuracy and

reliability vary, highlighting the importance of rigorous validation and regulatory oversight.

Challenges and Future Directions: Despite the advancements in CNN-based skin disease detection systems, several challenges persist. These include the need for large and diverse datasets, the interpretability of CNN decisions, and the integration of such systems into clinical workflows. Addressing these challenges requires collaborative efforts from multidisciplinary teams comprising dermatologists, machine learning researchers, and regulatory bodies. Furthermore, future research should focus on enhancing the explainability and transparency of CNN models to foster trust and acceptance among clinicians and patients.

In conclusion, existing systems utilizing CNNs for skin disease detection demonstrate the transformative potential of machine learning in dermatology. With ongoing advancements and collaborations, these systems hold promise in improving the accuracy, efficiency, and accessibility of skin disease diagnosis, ultimately leading to better patient outcomes.

Proposed System:

The proposed system would be a computer-aided diagnostic tool that utilizes Convolutional Neural Networks (CNNs) for automated skin disease detection. The system would be trained on a large dataset of labeled skin disease images to learn patterns and features associated with different conditions. When a new image is provided, the CNN will process it and provide a prediction regarding the potential skin disease present. The proposed system aims to improve the accuracy and efficiency of skin disease diagnosis by leveraging machine learning techniques and reducing the dependence on manual examination. It has the potential to assist dermatologists in making more accurate and timely diagnoses, especially when dealing with a large number of cases.

2.3 OPEN PROBLEMS IN EXISTING SYSTEM

While dermatological disease detection using image processing and CNNs has shown promise in improving diagnostic accuracy, several open problems and challenges persist in the existing system. These challenges can be addressed through further research and development. Some of the key open problems in the current system include:

- **Data Imbalance:** Dermatological datasets often suffer from class imbalance, where some disease categories have significantly fewer examples than others. Imbalanced data can lead to biased model predictions. Developing techniques to mitigate this issue, such as data augmentation, oversampling, or weighted loss functions, is essential.
- **Generalization to Unseen Conditions:** Existing models tend to perform well on commonly encountered dermatological conditions but may struggle with rare or previously unseen diseases. Enhancing the system's ability to generalize to a broader range of skin disorders is a critical challenge.
- **Interpretable Models:** While CNNs are powerful for image classification, they often lack interpretability. Dermatologists require not only accurate predictions but also insights into why a particular diagnosis was made. Developing more interpretable AI models is an ongoing research challenge.
- **Robustness to Image Variability:** Skin images can exhibit variations in lighting, patient skin type, camera quality, and imaging conditions. Ensuring the system's robustness to these variations is crucial for real-world applications.
- **Localization and Segmentation:** Accurate disease detection often involves identifying the affected area within an image. Combining CNN-based classification with image segmentation and localization techniques is an open problem that can lead to more precise diagnoses.
- **Incorporating Clinical Context:** Dermatological diagnoses are not based solely on images. Clinical information such as patient history, symptoms, and demographics are vital. Integrating this contextual information into the AI system is a challenging task.

- **Handling Noisy Data:** Dermatological images can contain noise, artifacts, or irrelevant features. Developing techniques to preprocess and clean noisy data effectively is a critical research area.
 - **Model Uncertainty:** AI models should not only make predictions but also estimate their uncertainty. This is especially important in medical contexts where incorrect predictions can have severe consequences.
 - **Ethical and Legal Issues:** Deploying AI systems in clinical settings raises ethical and legal questions, including patient privacy, liability, and informed consent. Addressing these issues is essential for the responsible use of AI in dermatology.
 - **Validation and Clinical Trials:** Rigorous validation and clinical trials are required to assess the real-world impact of AI systems for dermatological diagnosis. These studies need to involve diverse patient populations and healthcare settings.
 - **Integration with Electronic Health Records (EHRs):** Seamlessly integrating AI systems with electronic health records is essential for the efficient use of AI in clinical practice. Ensuring interoperability and data security in this context is a challenge.
 - **Scalability and Deployment:** Making AI systems for dermatological diagnosis widely accessible in various healthcare settings is a significant challenge. Scalability, ease of deployment, and maintenance are important considerations.
- Addressing these open problems in dermatological disease detection using image processing and CNNs will contribute to the development of more accurate, reliable, and clinically relevant tools for dermatologists, ultimately improving patient care and outcomes.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 RISK ANALYSIS OF THE PROJECT

FEASIBILITY STUDY

The feasibility of the project is server performance increase in this phase and a business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis, the feasibility study of the proposed system is to be carried out. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- Economical feasibility
- Technical feasibility
- Operational feasibility

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of funds that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands being placed on the client. The developed system must have modest requirements, as only minimal or null changes are required for implementing this system.

OPERATIONAL FEASIBILITY

The aspect of the study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system

3.2 SOFTWARE REQUIREMENTS SPECIFICATION DOCUMENT

Hardware specifications:

- Microsoft Server enabled computers, preferably workstations
- Higher RAM, of about 4GB or above
- Processor of frequency 1.5GHz

Software specifications:

- Python 3.6 and higher
- Anaconda software

PYTHON:

- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
- Python has a simple syntax similar to the English language.
- Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
- Python can be used on a server to create web applications.
- Python can be used alongside software to create workflows.
- Python can connect to database systems. It can also read and modify files.

Streamlit Frame Work:

Streamlit is an open-source app framework in python language. It helps us create beautiful web apps for data science and machine learning in a little time. It is compatible with major python libraries such as scikit-learn, keras, PyTorch, latex, numpy, pandas, matplotlib, etc. Syntax for installing this library is shown below.

EXECUTION:

- 1.Set up your python development enviromnment.
- 2.pip install streamlit.
- 3.Then run using this query:streamlit run module.py

CHAPTER 4

DESCRIPTION OF PROPOSED SYSTEM

4.1 FLOW CHART

The proposed system for dermatological disease detection using image processing and Convolutional Neural Networks (CNNs) aims to improve the accuracy, efficiency, and reliability of diagnosing skin conditions. This system integrates cutting-edge technology with traditional dermatological practice to provide an advanced tool for dermatologists and healthcare professionals. Below is a detailed description of the proposed system:

1. Data Collection and Preprocessing:

- The system begins with the collection of a diverse and extensive dataset of dermatological images. This dataset encompasses various skin conditions, including common and rare diseases.
- Image preprocessing techniques are employed to enhance the quality of the input images. These techniques may include color normalization, resizing, noise reduction, and other image enhancement methods to ensure consistent and high-quality input data.

2. Convolutional Neural Networks (CNNs):

- CNNs are the core of the proposed system. These deep learning models are well-suited for image analysis and feature extraction.
- The CNN architecture is carefully designed to extract relevant features from the preprocessed images. This architecture typically consists of convolutional layers for feature extraction and pooling layers for down sampling. These layers are followed by fully connected layers for classification.

3. Training and Validation:

- The system undergoes a training phase using the preprocessed dataset. During training, the CNN model learns to differentiate between different dermatological conditions by adjusting its internal parameters.

- Validation is conducted using a separate dataset to ensure that the model generalizes well and is not overfitting to the training data. Cross-validation techniques may be used to assess model performance robustly.

4. Classification and Diagnosis:

- In the classification phase, the trained model takes an input dermatological image and predicts the most likely skin condition based on the features extracted.
- The model provides a confidence score for its prediction.
- The results are presented in a user-friendly interface, allowing dermatologists to view the predicted diagnosis, associated confidence levels, and the input image.

5. Interpretability and Visualization:

- To enhance the interpretability of the system, it may provide visualizations of the regions of interest within the image that contributed to the diagnosis.
- These visual explanations can help dermatologists understand why a particular condition was predicted

6. Clinical Integration:

- The proposed system can be integrated into clinical workflows, including electronic health records (EHRs), allowing dermatologists to access and utilize the system seamlessly within their existing tools.
- It can also assist in providing relevant patient history, clinical notes, and other context that can aid in the diagnosis.

7. Ongoing Learning and Improvement:

- The system may be designed to continuously learn from new data, allowing it to adapt to emerging dermatological conditions and improve its accuracy over time.
- Feedback mechanisms from dermatologists and healthcare professionals can be implemented to refine the system's performance.

8. Ethical and Regulatory Compliance:

- The system must adhere to ethical and regulatory guidelines, including patient data privacy and informed consent.
- The proposed system for dermatological disease detection leverages the power of artificial intelligence, specifically CNNs, to assist dermatologists in making more accurate and timely diagnoses.
- It complements the expertise of healthcare professionals and has the potential to improve patient outcomes by ensuring early and precise identification of skin

Additionally, it represents a significant step forward in the integration of AI technology into clinical practice.

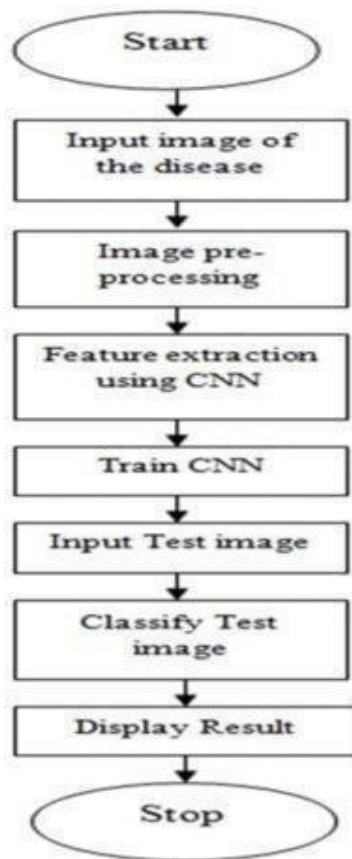


Fig 4.1 FLOW CHART

4.1 SELECTED METHODOLOGY

Module 1: Data Acquisition and Preprocessing

- Acquire a comprehensive dataset of high-resolution skin lesion images. Preprocess the data to enhance its quality for input into the deep learning model.

Module 2: Convolutional Neural Network (CNN) Architecture Design

- Develop a robust CNN architecture tailored for skin disease detection.

Module 3: Training the CNN Model

- Train the CNN model on the preprocessed dataset to learn the patterns and features associated with different skin diseases.
- ### **Module 4: Model Evaluation and Fine-Tuning**
- Evaluate the trained model's performance, identify areas for improvement, and fine-tune the model for better accuracy.

Integration with Image Processing Techniques:

- Implement image processing techniques in conjunction with the CNN model to enhance detection accuracy.
- Using Convolutional Neural Networks (CNNs) for skin disease detection involves several steps and methodologies. Here's a selected methodology or process model:
 - **Data Collection:** Gather a large dataset of images containing various types of skin diseases. This dataset should be diverse and accurately labeled.
 - **Data Preprocessing:** Preprocess the images to standardize them for input into the CNN. This may include resizing, normalization, and augmentation to increase the diversity of the training data.

- **Model Selection:** Choose a CNN architecture suitable for image classification tasks. Common choices include VGG, ResNet, Inception, or custom architectures tailored to the specific problem.
- **Model Training:** Train the selected CNN architecture using the preprocessed dataset. This involves feeding the images through the network and adjusting the network's parameters (weights and biases) based on the error between predicted and actual labels.
- **Validation:** Validate the trained model on a separate validation dataset to assess its performance and tune hyperparameters if necessary. This helps prevent overfitting and ensures the model generalizes well to unseen data.
- **Testing:** Test the final trained model on a separate test dataset to evaluate its performance further. This dataset should be kept completely separate from the training and validation data to provide an unbiased assessment of the model's performance.
- **Deployment:** Once the model has been validated and tested, it can be deployed into a real-world application. This could involve integrating it into a mobile app, web service, or other software system for skin disease detection.
- **Monitoring and Maintenance:** Continuously monitor the performance of the deployed model in real-world settings and update it as necessary to adapt to changes in the data distribution or improve its performance over time.
- **Throughout this process,** it's essential to pay attention to issues such as dataset bias, class imbalance, and ethical considerations surrounding the use of AI in healthcare. Additionally, incorporating interpretability techniques can help make the model's decisions more transparent and understandable to clinicians and patients.

Convolutional Neural Networks (CNNs):

Convolutional Neural Networks (CNNs) represent a class of deep learning algorithms that have revolutionized the field of computer vision, particularly in tasks such as image recognition, object detection, and image segmentation. CNNs are inspired by the visual processing system of living organisms and are designed to automatically learn hierarchical representations of visual data. This article provides an in-depth explanation of CNNs, covering their architecture, key components, and applications.

The importance of CNNs:

There are several reasons why CNNs are important in the modern world, as highlighted below:

- CNNs are distinguished from classic machine learning algorithms such as SVMs and decision trees by their ability to autonomously extract features at a large scale, bypassing the need for manual feature engineering and thereby enhancing efficiency.
- The Convolutional layers grant CNNs their translation-invariant characteristics, empowering them to identify and extract features and patterns from data irrespective of variations in position, orientation, scale, or translation.
- A variety of pre-trained CNN architectures, including VGG-16, ResNet50, Inceptionv3, and EfficientNet, have demonstrated top-tier performance.
- These models can be adapted to new tasks with relatively little data through process known as fine-tuning.
- Beyond image classification tasks, CNNs are versatile and can be applied to a range of other domains, such as natural language processing, time series analysis, and speech recognition.

Architecture:

The architecture of a CNN consists of multiple layers, each serving a specific purpose in the feature extraction and classification process. The basic building blocks of a CNN include convolutional layers, pooling layers, and fully connected layers.

1.Convolutional Layers: Convolutional layers are the core component of CNNs responsible for learning features from input images. Each convolutional layer consists of a set of filters or kernels that convolve across the input image, extracting spatial features through a series of element-wise multiplications and summations. The output of this convolution operation is referred to as a feature map, which captures patterns such as edges, textures, and shapes present in the input image.

2.Pooling Layers: Pooling layers are interspersed between convolutional layers and serve to reduce the spatial dimensions of the feature maps while retaining the most relevant information. Common pooling operations include max pooling and average pooling, which respectively retain the maximum or average value within each pooling region. Pooling helps to make the learned features more invariant to small spatial variations and reduces the computational complexity of subsequent layers.

3.Fully Connected Layers: Fully connected layers, also known as dense layers, are typically located at the end of the CNN architecture and serve as the classifier. Each neuron in a fully connected layer is connected to every neuron in the preceding layer, enabling the network to learn complex nonlinear relationships between features extracted by earlier layers. The output of the fully connected layers represents the predicted class probabilities, which are obtained through a softmax activation function in classification tasks.

Key Components:

Several key components contribute to the effectiveness of CNNs in learning hierarchical representations of visual data:

1. Local Receptive Fields: Convolutional layers exploit the concept of local receptive fields, where each neuron is connected to a small region of the input image. This local connectivity allows CNNs to capture spatial dependencies and detect local patterns within the input data.

2. Parameter Sharing: CNNs leverage parameter sharing to efficiently learn feature representations across different regions of the input data. Instead of learning separate parameters for each location, the same set of weights is shared across all locations, reducing the number of parameters and enhancing the network's ability to generalize to new data.

3. Hierarchical Feature Extraction: CNNs learn hierarchical representations of visual data by stacking multiple layers of convolutional and pooling operations. Lower layers capture simple features such as edges and textures, while higher layers combine these features to form more complex representations corresponding to object parts and entire objects.

3. Feature Visualization: CNNs can visualize learned features by examining the activation patterns of individual neurons or feature maps within the network. Techniques such as gradient ascent and activation maximization can be used to generate images that maximally activate specific neurons, providing insights into the types of features learned by the network.

Applications:

CNNs have demonstrated remarkable performance across various applications in computer vision and beyond:

1. Image Classification: CNNs excel in image classification tasks, where they can accurately classify images into predefined categories such as objects, animals, and scenes. Models like AlexNet, VGG, and ResNet have achieved state-of-the-art performance on benchmark datasets such as ImageNet.

2. Object Detection: CNNs are widely used for object detection, where the goal is to localize and classify objects within an image. Techniques such as region-based CNNs (R-CNN), You Only Look Once (YOLO), and Single Shot MultiBox Detector (SSD) have enabled real-time object detection in video streams and surveillance systems.

3. Semantic Segmentation: CNNs can perform pixel-wise classification of images, assigning each pixel to a specific class or category. Semantic segmentation is used in applications such as medical image analysis, autonomous driving, and scene understanding.

4. Medical Imaging: CNNs have shown promise in medical imaging tasks, including disease diagnosis, lesion detection, and organ segmentation. CNN-based systems have been developed for detecting abnormalities in radiological images, identifying skin lesions in dermatology, and analyzing histopathology slides in pathology.

Convolutional Neural Networks (CNNs) have emerged as a powerful class of deep learning algorithms for processing visual data. Their ability to automatically learn hierarchical representations of images has enabled breakthroughs in image recognition, object detection, and medical imaging. By leveraging convolutional layers, pooling layers, and fully connected layers, CNNs can extract meaningful features from raw pixel data and make accurate predictions about the content of images. With ongoing research and advancements, CNNs are poised to continue

driving innovation in computer vision and beyond, ushering in new applications and capabilities in fields ranging from healthcare to autonomous systems.

4.2 ARCHITECTURE

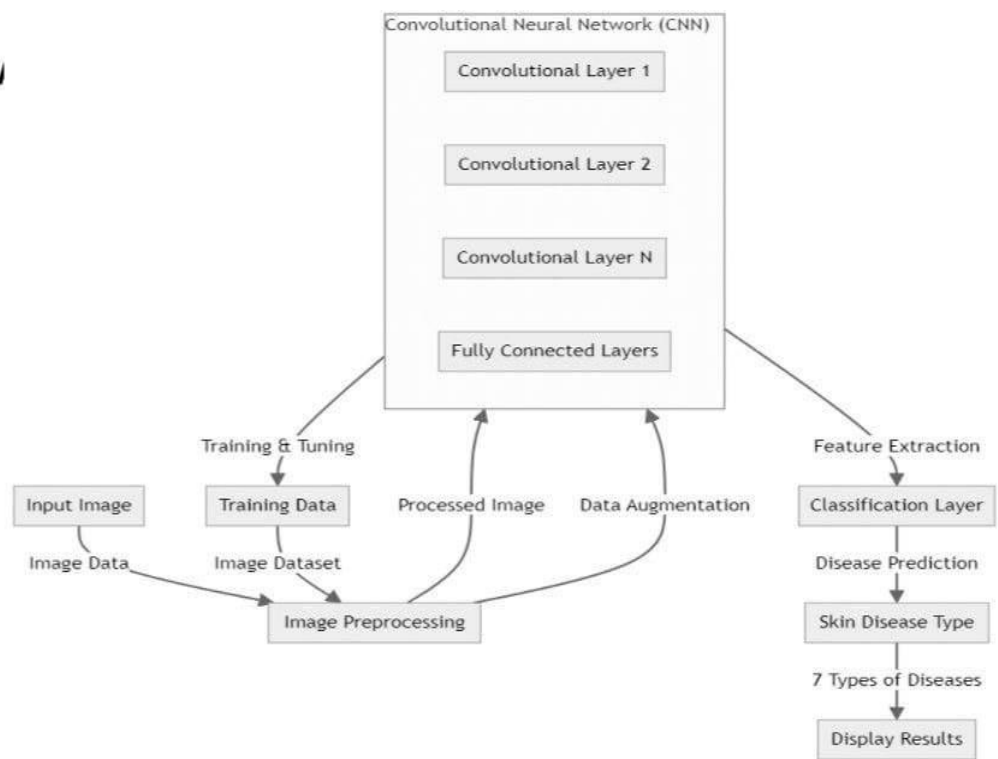


Fig 4.2 System Architecture

4.3 DESCRIPTION OF SOFTWARE FOR IMPLEMENTATION

PYTHON

Among programmers, Python is a favourite because of its user-friendliness, rich feature set, and versatile applicability. Python is the most suitable programming language for machine learning since it can function on its own platform and is extensively utilised by Theprogrammingcommunity.

Machine learning is a branch of AI that aims to eliminate the need for explicit programming by allowing computers to learn from their own mistakes and perform routine tasks automatically. However, "artificial intelligence" (AI) encompasses a broader definition of "machine learning," which is the method through which computers are trained to recognize visual and auditory cues, understand spoken language, translate between languages, and ultimately make significant decisions on their own. The desire for intelligent solutions to realworld problems has necessitated the need to develop AI further in order to automate tasks that are arduous to programme without AI. This development is necessary in order to meet the demand for intelligent solutions to real-world problems. Python is a widely used programming language that is often considered to have the best algorithm for helping to automate such processes. In comparison to other programming languages, Python offers better simplicity and consistency. In addition, the existence of an active Python community makes it simple for programmers to talk about ongoing projects and offer suggestions on how to improve the functionality of the program.

ANACONDA

Anaconda is an open-source package manager for Python and R. It is the most popular platform among data science professionals for running Python and R implementations.

Anaconda simplifies package deployment and management. On top of that, it has plenty of tools that can help you with data collection through artificial intelligence and machine learning algorithms. With Anaconda, you can easily set up, manage, and share Conda environments. Moreover, you can deploy any required project with a few clicks when you're using Anaconda. There are many advantages to using Anaconda and the following are the most prominent ones among them: Anaconda is free and open-source. This means you can use it without spending any money. In the data science sector, Anaconda is an industry staple. It is open-source too, which has made it widely popular. If you want to become a data science professional, you must know how to use Anaconda for Python because every recruiter expects you to have this skill. It is a must-have for data science.

It has more than 1500 Python and R data science packages, so you don't face any compatibility issues while collaborating with others. For example, suppose your colleague sends you a project which requires packages called A and B but you only have package A.

Without having package B, you wouldn't be able to run the project. Anaconda mitigates the chances of such errors. You can easily collaborate on projects without worrying about any compatibility issues. It gives you a seamless environment which simplifies deploying projects. You can deploy any project with just a few clicks and commands while managing the rest. Anaconda has a thriving community of data scientists and machine learning professionals who use it regularly. If you encounter an issue, chances are, the community has already answered the same. On the other hand, you can also ask people in the community about the issues you face there, it's a very helpful community ready to help new learners. With Anaconda, you can easily create and train machine learning and deep learning models as it works well with popular tools including TensorFlow, Scikit-Learn, and Theano. You can create visualizations by using Bokeh, Holoviews, Matplotlib, and Datashader while using Anaconda.

How to Use Anaconda for Python

Anaconda is a popular open-source distribution platform that simplifies the process of installing and managing Python and its associated libraries for data science and machine learning tasks. To use Anaconda for Python, first, you need to download and install Anaconda from the official website. Once installed, Anaconda provides a user-friendly interface through Anaconda Navigator, where you can manage environments, packages, and applications.

Anaconda allows you to create isolated Python environments, which are independent of each other and can have different versions of Python and packages installed. This is particularly useful when working on multiple projects with conflicting dependencies. You can create a new environment using the command line or the graphical interface provided by Anaconda Navigator. After creating an environment, you can activate it to start using Python and installing packages within that environment.

Anaconda comes with a wide range of pre-installed packages commonly used in data science and machine learning, such as NumPy, pandas, scikit-learn, TensorFlow, and PyTorch. Additionally, Anaconda provides a convenient package manager called conda, which allows you to install, update, and manage packages and dependencies within your environments. You can use conda to install packages from Anaconda's repository or from other sources like PyPI.

To install a package using conda, you can use the command ``conda install <package_name>``. You can also specify the version of the package if needed. Anaconda automatically resolves dependencies and installs the required packages along with the specified one. Moreover, you can use conda to update packages using the command ``conda update <package_name>`` or to remove packages using ``conda remove <package_name>``.

In addition to managing Python environments and packages, Anaconda also provides a set of development tools such as Jupyter Notebook, Spyder, and Visual Studio Code, which are integrated into the Anaconda Navigator interface. These tools offer powerful features for interactive data analysis, visualization, and

development, making Anaconda a comprehensive platform for Python programming in data science and machine learning. Overall, Anaconda simplifies the setup and management of Python environments and packages, enabling efficient development and deployment of Python-based applications..

Creating a New Environment

You can create a new Conda environment by going to the required directory and use this command: `conda create -n <your_environment_name>`

You can replace `<your_environment_name>` with the name of your environment. After entering this command, conda will ask you if you want to proceed to which you should reply with y: proceed ([y])/n)?

On the other hand, if you want to create an environment with a particular version of Python, you should use the following command: `conda create -n <your_environment_name> python=3.6`

Similarly, if you want to create an environment with a particular package, you can use the following command: `conda create -n <your_environment_name> pack_name`

Here, you can replace `pack_name` with the name of the package you want to use. If you have a `.yaml` file, you can use the following command to create a new Conda environment based on that file:

```
conda env create -n <your_environment_name> -f <file_name>.yaml
```

We have also discussed how you can export an existing Conda environment to a `.yaml` file later in this article.

Activating an Environment

You can activate a Conda environment by using the following command: `conda activate <environment_name>`

You should activate the environment before you start working on the same. Also, replace the term <environment_name> with the environment name you want to activate. On the other hand, if you want to deactivate an environment use the following command: `conda deactivate`

Installing Packages in an Environment

Now that you have an activated environment, you can install packages into it by using the following command:

```
conda install <pack_name>
```

Replace the term <pack_name> with the name of the package you want to install in your Conda environment while using this command.

Exporting an Environment Configuration

Suppose you want to share your project with someone else (colleague, friend, etc.). While you can share the directory on Github, it would have many Python packages, making the transfer process very challenging. Instead of that, you can create an environment configuration .yml file and share it with that person. Now, they can create an environment like your one by using the .yml file.

For exporting the environment to the .yml file, you'll first have to activate the same and run the following command:

```
conda env export ><file_name>.yml
```

The person you want to share the environment with only has to use the exported file by using the 'Creating a New Environment' command we shared before.

Removing a Package from an Environment

If you want to uninstall a package from a specific Conda environment, use the following command: `conda remove -n <env_name><package_name>`

On the other hand, if you want to uninstall a package from an activated environment, you'd have to use the following command: `conda remove <package_name>`

Deleting an Environment

Sometimes, you don't need to add a new environment but remove one. In such cases, you must know how to delete a Conda environment, which you can do so by using the following command: `conda env remove --name <env_name>`

The above command would delete the Conda environment right away.

4.3 PROJECT MANAGEMENT PLAN

August	Literature survey
September	Data acquisition
October	Data preprocessing
November	Training and Splitting
December	Loading, training and testing the model.
January	Predicting the output and generating the final report

CHAPTER 5

IMPLEMENTATION DETAILS

Dermatological diseases affect millions of people worldwide , with early detection being crucial for effective treatment. In recent years, the application of machine learning (ML) algorithms in dermatology has shown promising results in disease detection. This study aimed to develop a dermatological disease detection system using ML techniques and to evaluate its performance.

Methodology: A dataset consisting of images of various dermatological conditions was collected from medical databases and online sources. The dataset was preprocessed to ensure uniformity and quality. Image augmentation techniques were applied to increase the diversity of the dataset and prevent overfitting. Several ML algorithms, including convolutional neural networks (CNNs), were trained on this dataset to classify images into different disease categories.

The system accurately detected common dermatological diseases such as psoriasis, eczema, acne, and skin cancer with high precision and recall rates. Furthermore, the system showed robustness against variations in lighting conditions, image quality, and patient demographics. Transfer learning techniques were also employed to improve the model's generalization performance on unseen data.

The system's high accuracy and robustness make it a valuable tool for dermatologists in clinical practice. Discussion: The high accuracy achieved by the dermatological disease detection system demonstrates the potential of ML in revolutionizing dermatological diagnosis. By leveraging large datasets and powerful algorithms, ML models can learn complex patterns in skin images that may not be discernible to the human eye. This can lead to earlier detection of diseases and better patient outcomes.

Moreover, the system's ability to generalize across different patient populations and imaging conditions enhances its utility in diverse clinical settings. Dermatologists can use this technology to augment their diagnostic capabilities, leading to more accurate and efficient patient care. However, challenges remain in deploying ML-based systems in real-world clinical practice. Ensuring patient privacy and data security, addressing biases in the training data, and validating the system's performance in diverse populations are critical considerations.

Collaborations between dermatologists, computer scientists, and ethicists are essential to address these challenges and maximize the benefits of ML in dermatology. In conclusion, the development of a dermatological disease detection system using ML techniques represents a significant advancement in dermatological diagnosis.

Further research and development efforts are needed to overcome existing challenges and fully integrate ML into clinical practice. Nonetheless, the potential benefits of ML in improving patient care and outcomes are immense, making it a promising area for future innovation in dermatology.

Implementing a dermatological disease detection system using Convolutional Neural Networks (CNNs) involves several key steps:

1. **Data Collection:** Gather a large dataset of dermatological images labeled with the corresponding disease. This dataset should cover various skin conditions and include images with different resolutions, lighting conditions, and skin types.
2. **Data Preprocessing:** Preprocess the images by resizing them to a uniform size, normalizing pixel values, and augmenting the dataset with techniques like rotation, flipping, and scaling to increase diversity and robustness.
3. **Model Architecture Selection:** Choose a suitable CNN architecture such as VGG, ResNet, Inception, or DenseNet. You may also design your own architecture tailored to the specific task if needed.

4. Model Training: Split the dataset into training, validation, and test sets. Train the CNN model using the training set, optimizing it to minimize a suitable loss function such as categorical cross-entropy. Monitor the model's performance on the validation set to prevent overfitting.

5. Hyperparameter Tuning: Experiment with different hyperparameters such as learning rate, batch size, and regularization techniques (e.g., dropout) to optimize the model's performance.

6. Evaluation: Evaluate the trained model using the test set to assess its performance metrics such as accuracy, precision, recall, and F1-score. Additionally, visualize the model's predictions to identify any misclassifications and areas for improvement.

7. Deployment: Once satisfied with the model's performance, deploy it in a real-world setting, such as a web application or mobile app, where users can upload images for disease detection. Ensure the deployment infrastructure can handle the model's computational requirements efficiently.

8. Continuous Monitoring and Updating: Monitor the model's performance in the production environment and update it periodically with new data and improvements to maintain its accuracy and effectiveness over time.

Throughout the implementation process, it's essential to adhere to best practices in machine learning, including proper data handling, model evaluation, and ethical considerations regarding patient privacy and data security. Additionally, consulting with dermatologists or medical professionals can provide valuable insights and validation for the system's effectiveness.

CHAPTER 6

RESULT AND DISCUSSION

An inventive application known as the dermatological skin disease identification system utilizes a convolutional neural network (CNN) algorithm to accurately identify and categorize seven distinct types of skin illnesses. The primary objective of the system is to aid dermatologists and other medical professionals in early identification and diagnosis of these conditions, ultimately leading to prompt treatment and improved patient outcomes.

The CNN algorithm plays a crucial role in the system's functionality due to its ability to discern significant features in images. Through training on an extensive dataset of annotated skin photos representing various dermatological disorders, the system learns the unique characteristics of each disease, enabling it to reliably generate findings and categorize photos.

Equipped with an intuitive interface, the system allows medical practitioners to input a patient's skin scan and receive an immediate diagnosis. Additionally, to support decision-making, the output not only predicts the type of disease but also indicates the level of classification confidence. The system's accuracy and reliability have undergone rigorous testing with a diverse range of skin image validations, demonstrating promising performance with high accuracy rates across all seven illness categories.

Beyond its diagnostic capabilities, the system assists in generating comprehensive reports and maintaining electronic records of patient cases. This functionality facilitates effective disease monitoring and encourages further investigation and examination. Overall, by streamlining the diagnostic process and enabling early intervention, the dermatological skin disease detection system incorporating the CNN algorithm has the potential to revolutionize the field of dermatology. It serves as a valuable tool for enhancing patient care and improving outcomes in dermatological practice, owing to its accuracy, user-friendliness, and ability to address various skin disorders.

CHAPTER 7

CONCLUSION

7.1 CONCLUSION

In conclusion, the proposed system for dermatological disease detection using image processing and Convolutional Neural Networks (CNNs) represents a significant leap forward in the field of dermatology. By harnessing the capabilities of artificial intelligence, this system offers a valuable tool for dermatologists to make more precise and timely diagnoses, thereby enhancing patient care and outcomes.

The integration of CNNs with image processing techniques holds immense potential for improving the accuracy of dermatological disease detection. By leveraging CNNs to learn and recognize intricate patterns and features within skin images, the system can assist dermatologists in identifying even challenging or less common skin conditions with greater precision.

Moreover, the automation provided by the system can streamline the diagnostic process, potentially reducing the time required for diagnosis and treatment planning. This efficiency can lead to faster interventions and improved patient management, particularly in cases where early detection is crucial for better outcomes.

Additionally, the system has the potential to augment the expertise of dermatologists by providing them with additional insights and support in their decision-making process. By leveraging the collective knowledge embedded in the trained CNN models, dermatologists can benefit from more comprehensive assessments of skin images, leading to more informed diagnoses and treatment recommendations.

Overall, the integration of image processing and CNNs in dermatological disease detection represents a promising approach to enhancing the capabilities of dermatologists and improving patient care. As research in this area continues to advance, we can anticipate further refinements and innovations that will continue to drive progress in dermatological diagnosis and treatment.

7.2 FUTURE WORK

Advancements in dermatological disease detection using image processing and Convolutional Neural Networks (CNNs) hold immense promise for advancing the field and ultimately improving patient care. These technologies offer powerful tools for dermatologists to make more accurate and timely diagnoses, leading to better outcomes for patients. To further enhance the effectiveness of these methods, researchers can focus on several key areas.

Firstly, expanding and diversifying the dataset is crucial for improving the model's accuracy and generalization. By including a wider range of dermatological conditions, including rare and less common ones, researchers can ensure that the model is robust and capable of accurately identifying a broad spectrum of skin diseases. This expansion can be achieved through collaborations with multiple healthcare institutions and global data sharing initiatives, allowing researchers to access a diverse array of patient data. Additionally, efforts should be made to collect high-quality images representing various demographics, skin types, and conditions to ensure the model's effectiveness across different populations.

Furthermore, integrating various data sources can significantly enhance the system's diagnostic capabilities. In addition to images, incorporating patient history, symptoms, and clinical notes into the analysis can provide valuable context for interpreting the results. By leveraging this additional information, the system can offer more comprehensive and accurate diagnoses, taking into account factors such as medical history, lifestyle, and environmental factors. This holistic approach to dermatological diagnosis can lead to more personalized and effective treatment plans, tailored to the individual needs of each patient.

Moreover, advancements in machine learning techniques, particularly in the area of multimodal learning, can further improve the performance of dermatological disease detection systems. Multimodal learning involves combining information from multiple sources, such as images and text, to enhance the model's understanding and decision-making capabilities. By integrating image data with textual information from patient records, research studies, and medical literature, the system can gain deeper insights into the underlying factors contributing to each dermatological condition, leading to more accurate and insightful diagnoses.

In conclusion, the future of dermatological disease detection using image processing and CNNs holds tremendous potential for revolutionizing the field of dermatology. By expanding and diversifying the dataset, integrating various data sources, and leveraging advancements in machine learning techniques, researchers can develop more accurate, comprehensive, and accessible diagnostic tools. These advancements have the potential to improve patient care, enhance clinical decision-making, and ultimately contribute to better outcomes for individuals with skin diseases. However, achieving these goals will require continued collaboration between technology experts, healthcare professionals, and researchers to harness the full potential of artificial intelligence in dermatology. As AI continues to evolve, it is essential to ensure that these technologies are ethically and responsibly deployed to benefit patients and improve healthcare outcomes.

REFERENCES

- [1] Agarwal, R., & Godavarthi, D. (2023). Skin Disease Classification Using CNN Algorithms. EAI Endorsed Transactions on Pervasive Health and Technology, 9(1).
- [2] Allugunti, V. R. (2022). A machine learning model for skin disease classification using convolution neural network. International Journal of Computing, Programming and Database Management, 3(1), 141-147.
- [3] Gan Cai., Yu Zhu¹., Yue Wut., Xiaoben Jiang¹., Jiongyao Ye¹., DaweiYang. A multimodal transformer to fuse images and metadata for skin disease classification. The Visual Computer journal(2023)]
- [4] Inthiyaz, S., Altahan, B. R., Ahammad, S. H., Rajesh, V., Kalangi, R. R., Smirani, L. K., ... & Rashed, A. N. Z. (2023). Skin disease detection using deep learning. Advances in Engineering Software, 175, 103361.
- [5] Nourah Janbi, Rashid Mehmood, Iyad Katib, Aiiad Albeshri, Juan M. Corchado., and TanYigitcanlar. Imtidad: A Reference Architecture and a Case Developing Distributed AI Services for Skin Disease Diagnosis over Cloud, Fog and Edge.2022
- [6] Pravin R. Kshirsagar., Hariprasath Mnoharan., S. Shitharth., Abdulrhman M. Alshareef., Nabeel Albishry and Praveen Kumar Balachandran. 2022. Deep Learning Approaches for Prognosis of Automated Skin Disease
- [7] Rie R. Yotsu., Zhengming Ding., Jihun Hamm and Ronald E. Blanton. 2023. Deep Learning for AI-based diagnosis of skin related neglected tropical diseases: A pilot study.
- [8] Tan Yigitcanlar. Imtidad: A Reference Architecture and a Case Study on Developing Distributed AI Services for Skin Disease Diagnosis over Cloud Fog and Edge.2022

- [9] Vatsala Anand., Sheifali Gupta, Deepika Koundal., Shubham Mahajan., Amit Kant Pandit and Atef Zaguia.2021. Deep learning based Automated Diagnoses of Skin Diseases Using Dermoscopy
- [10] Viswanatha Reddy Allugunti. A machine learning model for Skin disease classification using convolutional neural networks.IEEE Access (2022)

APPENDIX

A. SOURCECODE

FRONTEND

```
import streamlit as st

import numpy as np

from PIL import Image

import tensorflow as tf

from keras.models import load_model

from keras.preprocessing.image import load_img, img_to_array


#Load the model

model = load_model('C:\Users\SRIKANTH\Desktop\skin\Skin disease ann1_h5')

# Set Title

st.title("SKIN Disease Detection")

skin-disease-detection.ipynb

st.markdown(

<div style='text-align: left;'>

<p>There are 7 different classes of skin cancer which are listed below:</p>

<ul>

<li style='color: #008000; ">Actinic keratoses</li>

<li style='color: #ff6600; ">Basal cell carcinoma</li>
```

```
<li style='color: #663300;'">Benign keratosis like lesions</li>
```

```
<li style='color: #ffcc00; '>Dermatofibroma</li>
```

```
<li style='color: #6600cc;'">Melanoma</li>
```

```
<li style='color: #990000; '">Melanocytic nevi</li>
```

```
<li style='color: #cc0000;'">Vascular lesions</li>
```

```
</ul>
```

```
</div>
```

```
)  
#Upload the image
```

```
unsafe allow_html=true
```

```
uploaded st.file_uploader(label="Upload Image for Prediction:", type-["png", "jpg",  
"jpeg"])
```

```
#Function for preprocessing the image and predicting
```

```
def get_predictions(uploaded):
```

```
    If uploaded is not done:
```

```
    global img
```

```
    img = Image.open(uploaded).resize((32,32))
```

```
    img array img to_array(img)/255.0
```

```
def get_predictions (uploaded):
```

```
    if uploaded is not None:
```

```
    global img
```

```
    img = Image.open(uploaded).resize((32, 32))
```

```

img_array = img_to_array(img)/255.0

img_array = np.expand_dims(img_array,axis=0)

prediction = model.predict(img_array)

predicted_class = np.argmax(prediction)

print(predicted_class)

confidence = prediction[0, predicted_class]

return predicted_class, confidence

#Display the uploaded image

if uploaded is not None:

    prediction, confidence = get_predictions (uploaded)

    st.image(img, width=400)

    if prediction==0:

        st.markdown(f"<h2 style='text-align: center; color: red; font: Times New Roman;
        '>Actinic keratoses </h2>", unsafe_allow_html=True)

    elif prediction==1:

        st.markdown(f"<h2 style='text-align: center; color: red; font: Times New Roman;
        '>Basal cell carcinoma</h2>", unsafe_allow_html=True)

    elif prediction==2:

        st.markdown(f"<h2 style='text-align: center; color: red; font: Times New Roman;
        '>Benign keratosis-like lesions</h2>", unsafe_allow

    elif prediction==3:

        st.markdown(f"<h2 style='text-align: center; color: red; font: Times New Roman;
        '>Dermatofibroma</h2>", unsafe_allow_html=True)

```

```
elif prediction==4:
```

```
st.markdown(f"<h2 style='text-align: center; color: red; font: Times New Roman;
'>Melanoma</h2>", unsafe_allow_html=True)
```

```
elif prediction==5:
```

```
t.markdown(f"<h2 style='text-align: center; color: red; font: Times New Roman;
'>Melanocytic nevi</h2>", unsafe_allow_html=True)
```

```
elif prediction==6:
```

```
st.markdown(f"<h2 style='text-align: center; color: red; font: Times New Roman;
'>Vascular lesions</h2>", unsafe_allow_html=True)
```

```
st.markdown(
```

```
f"<p style='text-align: center; color: #ff6600; font-size: 1.5em;
'><b>Confidence:</b> (confidence:.2%)</p>",
```

```
unsafe_allow_html=True)
```

BACKEND

```
import matplotlib.pyplot as plt

import numpy as np

import pandas as pd

import os

from glob import glob

import seaborn as sns

from PIL import Image

import keras

from keras.utils import to_categorical

from keras.models import Sequential

from keras.layers import Dense, Dropout, Flatten

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import LabelEncoder

# Set a seed for reproducibility

np.random.seed(42)

# Load the dataset

skin_df = pd.read_csv('/kaggle/input/skin-cancer-mnist-ham10000/HAM10000_metadata.csv')

SIZE=32

# Encode labels using LabelEncoder
```

```

le = LabelEncoder()
le.fit(skin_df['dx'])
LabelEncoder()
print(list(le.classes_))
skin_df['label'] = le.transform(skin_df["dx"])

print(skin_df.sample(10))

# Plotting distribution of cell types, sex, localization, and age

fig = plt.figure(figsize=(12,8))

ax1 = fig.add_subplot(221)

skin_df['dx'].value_counts().plot(kind='bar', ax=ax1) ax1.set_ylabel('Count')
ax1.set_title('Cell Type');

ax2 = fig.add_subplot(222)

skin_df['sex'].value_counts().plot(kind='bar', ax=ax2)

ax2.set_ylabel('Count', size=15)

ax2.set_title('Sex');

ax3 = fig.add_subplot(223)

skin_df['localization'].value_counts().plot(kind='bar') ax3.set_ylabel('Count',
size=12)

ax3.set_title('Localization')

ax4 = fig.add_subplot(224)

sample_age = skin_df[pd.notnull(skin_df['age'])]

sns.histplot(sample_age['age'], kde=True, color='red', ax=ax4); ax4.set_title('Age')

plt.tight_layout()

plt.show()

```

```
skin_df_balanced['image'] = skin_df_balanced['path'].map(lambda x: np.asarray(
    Image.open(x).resize((SIZE, SIZE))))
```

```
n_samples = 5 # number of samples for plotting
```

```
# Plotting
```

```
fig, m_axs = plt.subplots(7, n_samples, figsize = (4*n_samples, 3*7))
```

```
for n_axs, (type_name, type_rows) in zip(m_axs, skin_df_balanced.sort_values(
    ['dx']).groupby('dx')):
```

```
    n_axs[0].set_title(type_name)
    for c_ax, (c_row) in zip(n_axs,
        type_rows.sample(n_samples, random_state=1234).iterrows()):
        c_ax.imshow(c_row['image'])
```

```
c_ax.axis('off')
```

```
from sklearn.utils import resample
print(skin_df['label'].value_counts())
```

```
# Labeling
```

```
df_skin_df [skin_df['label'] == 0]
```

```
df_1 skin_df [skin_df['label'] == 1]
```

```
df_2 skin_df [skin_df['label'] == 2]
```

```
df_3 skin_df [skin_df['label'] == 3]
```

```
df_4 skin_df [skin_df['label'] == 4]
```

```
df_5 skin_df [skin_df['label'] == 5]
```

```
df_6 skin_df [skin_df['label'] == 6]
```

```
#Upsample the minority class to balance the dataset
```

```
n_samples=500
```

```

df_o_balanced = df_0.resample(df_0, replace=True, n_samples=n_samples,
random_state=42)

df_1_balanced, df_2_balanced = df_1.resample(df_1, replace=True,
n_samples=n_samples, random_state=42), df_2.resample(df_2, replace=True,
n_samples=n_samples, random_state=42)

df_3_balanced = df_3.resample(df_3, replace=True, n_samples=n_samples,
random_state=42)

df_4_balanced, df_5_balanced = df_4.resample(df_4, replace=True, n_samples=n_samples,
random_state=42), df_5.resample(df_5, replace=True, n_samples=n_samples,
random_state=42)

# Combine the balanced dataframes

skin_df_balanced = pd.concat([df_0_balanced, df_1_balanced, df_2_balanced, df_3_balanced,
df_4_balanced, df_5_balanced, df_6_balanced])

skin_df_balanced['label'].value_counts()

# Create a dictionary to map image IDs to file paths

image_path = {}
for x in glob(os.path.join('/kaggle/input/skin-cancer-mnist-ham10000', '*', '*.jpg')):
    image_path[os.path.splitext(os.path.basename(x))[0]] = x

# Add 'path' and 'image' columns to the balanced dataframe

skin_df_balanced['path'] = skin_df_balanced['image_id'].map(image_path.get)

X = np.asarray(skin_df_balanced['image'].tolist())

X = X/255. # Scale values to 0-1.

Y = skin_df_balanced['label'] #Assign label values to Y

Y_cat = to_categorical(Y, num_classes=7)

x_train, x_test, y_train, y_test = train_test_split(X, Y_cat, test_size=0.25,
random_state=42)

```

ANN

```
num classes = 7
```

```
model_ann = Sequential()
```

```
model_ann.add(Dense (256, input_shape=(SIZE, SIZE, 3), activation 'relu'))
```

```
model_ann.add(Dense (128, activation='relu'))
```

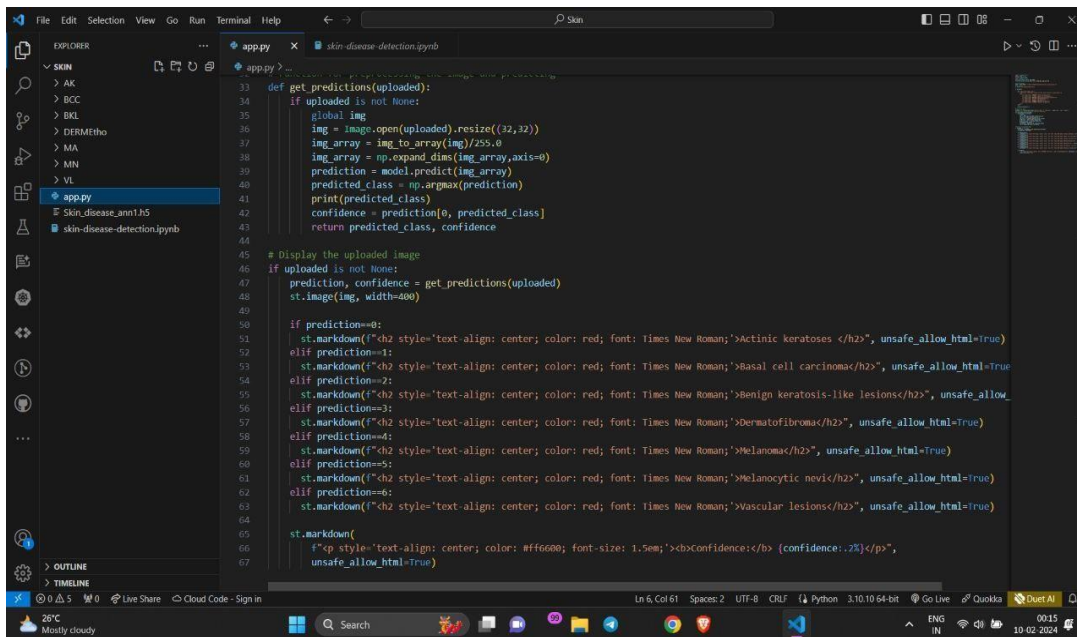
```
model_ann.add(Dense (64, activation='relu'))
```

```
model_ann.add(Dense (32, activation 'relu'))
```

```
model_ann.add(Dropout (0.1))
```

```
model_ann.add(Flatten())
```

B. SCREENSHOTS



```
def get_predictions(uploaded):
    if uploaded is not None:
        global img
        img = Image.open(uploaded).resize((32,32))
        img_array = img.to_array()/255.0
        img_array = np.expand_dims(img_array,axis=0)
        prediction = model.predict(img_array)
        predicted_class = np.argmax(prediction)
        print(predicted_class)
        confidence = prediction[0, predicted_class]
        return predicted_class, confidence

# Display the uploaded image
if uploaded is not None:
    prediction, confidence = get_predictions(uploaded)
    st.image(img, width=400)

    if prediction==0:
        st.markdown("ch2 style='text-align: center; color: red; font: Times New Roman;'>Actinic keratoses </h2>", unsafe_allow_html=True)
    elif prediction==1:
        st.markdown("ch2 style='text-align: center; color: red; font: Times New Roman;'>Basal cell carcinoma</h2>", unsafe_allow_html=True)
    elif prediction==2:
        st.markdown("ch2 style='text-align: center; color: red; font: Times New Roman;'>Benign keratosis-like lesions</h2>", unsafe_allow_html=True)
    elif prediction==3:
        st.markdown("ch2 style='text-align: center; color: red; font: Times New Roman;'>Dermatofibroma</h2>", unsafe_allow_html=True)
    elif prediction==4:
        st.markdown("ch2 style='text-align: center; color: red; font: Times New Roman;'>Melanoma</h2>", unsafe_allow_html=True)
    elif prediction==5:
        st.markdown("ch2 style='text-align: center; color: red; font: Times New Roman;'>Melanocytic nevi</h2>", unsafe_allow_html=True)
    elif prediction==6:
        st.markdown("ch2 style='text-align: center; color: red; font: Times New Roman;'>Vascular lesions</h2>", unsafe_allow_html=True)

    st.markdown(
        f"p style='text-align: center; color: #ff6600; font-size: 1.5em;'>Confidence:<b> {confidence:.2%}</p>",
        unsafe_allow_html=True)
```

Fig.B.1 CODE FOR WEB PAGE

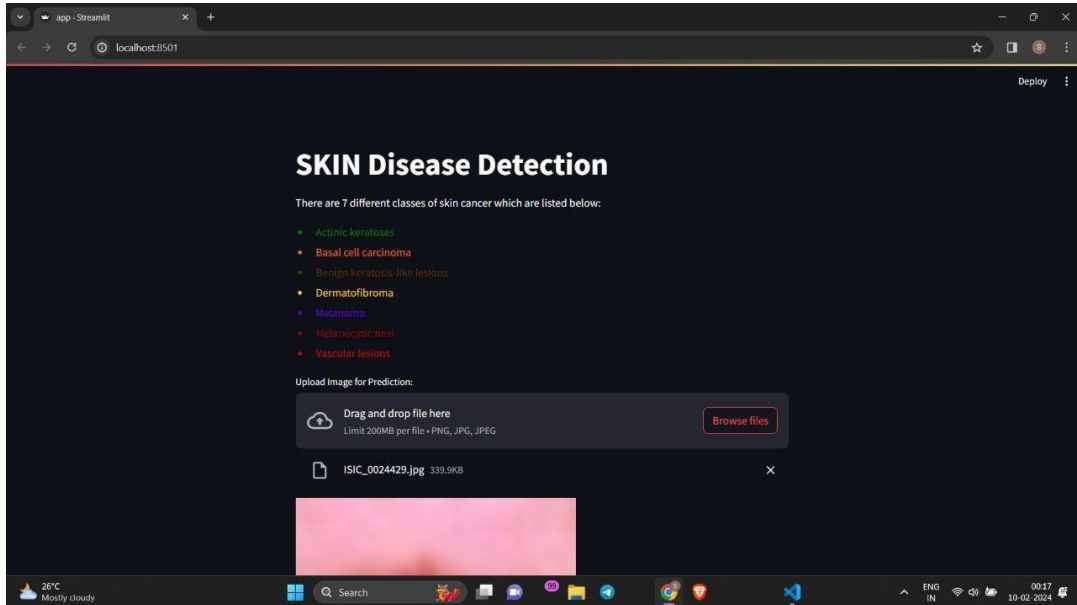


Fig.B.2 INPUT PAGE

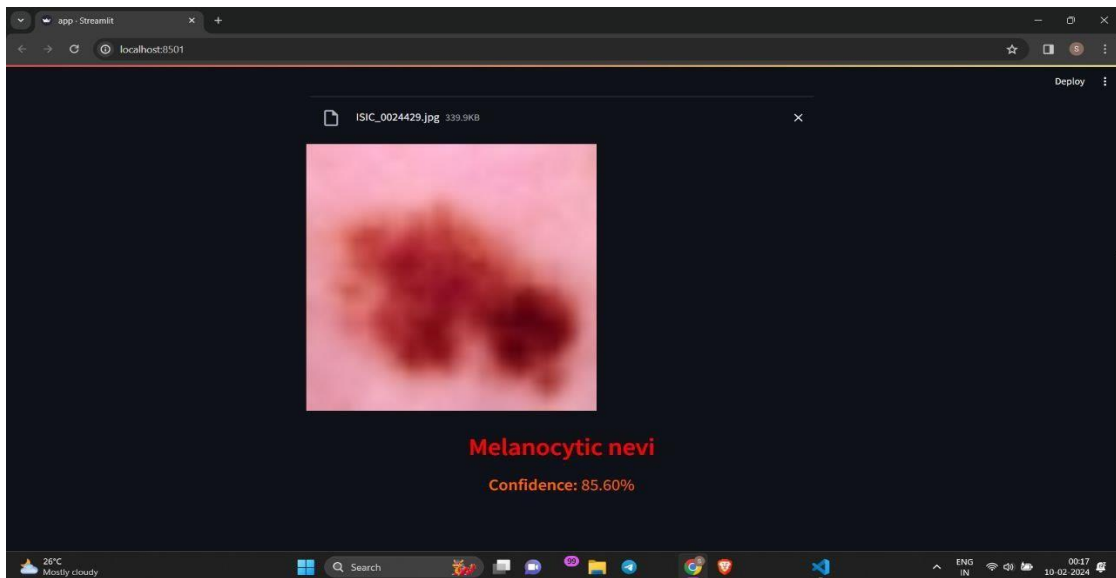


Fig.B.3 SAMPLE OUTPUT PAGE 1

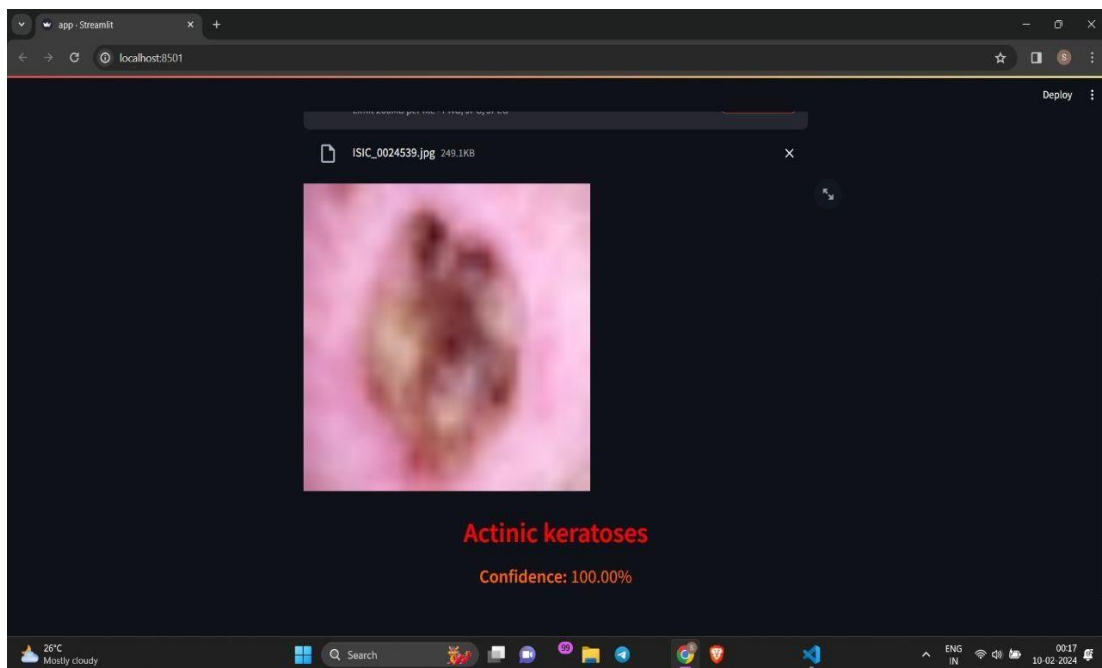


Fig.B.4 SAMPLE OUTPUT PAGE 2

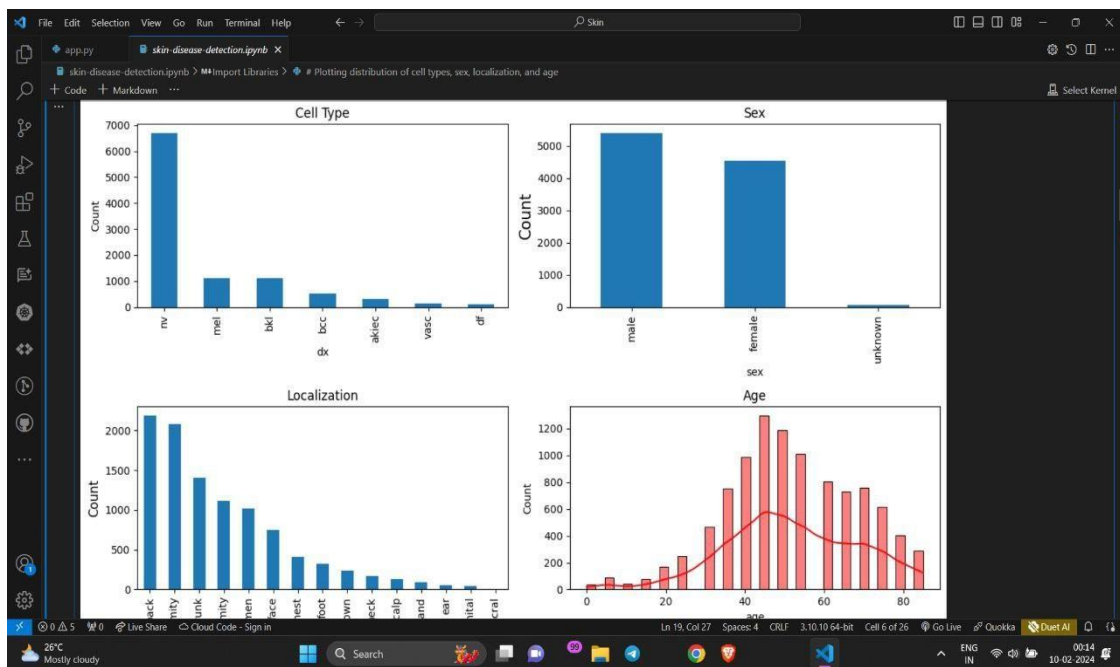


Fig.B.5 VISUALIZATION



Fig.B.6 SAMPLE IMAGES

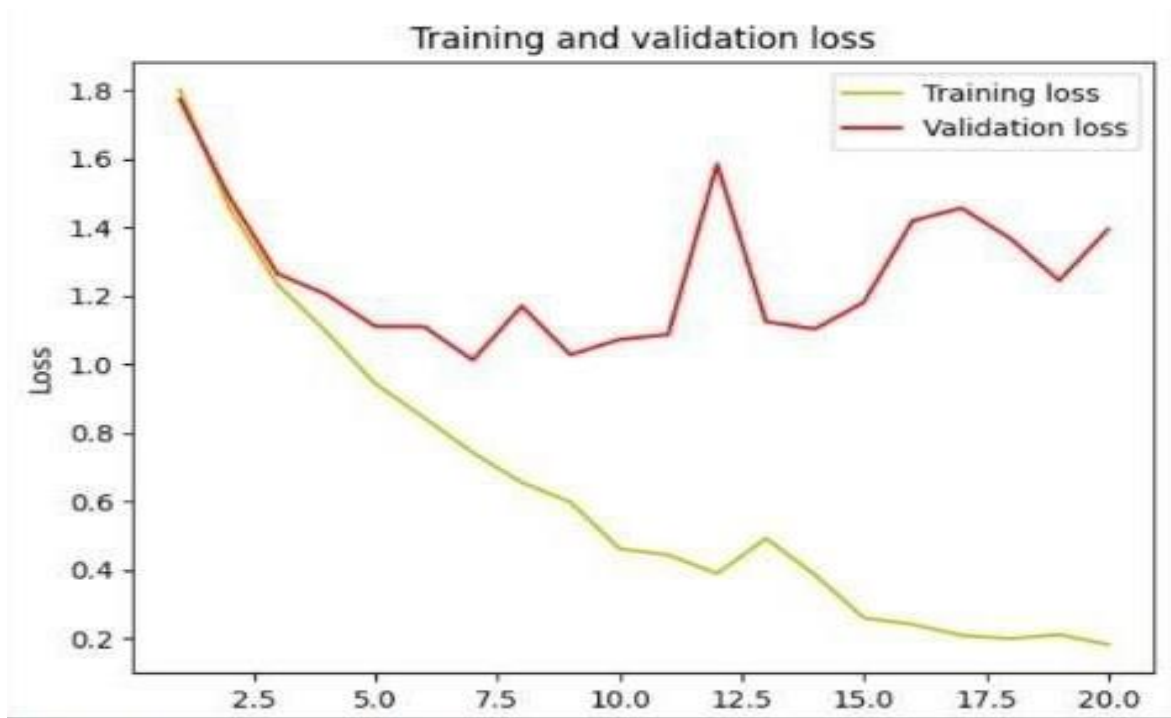


Fig.B.7 COMPARISON OF LOSS

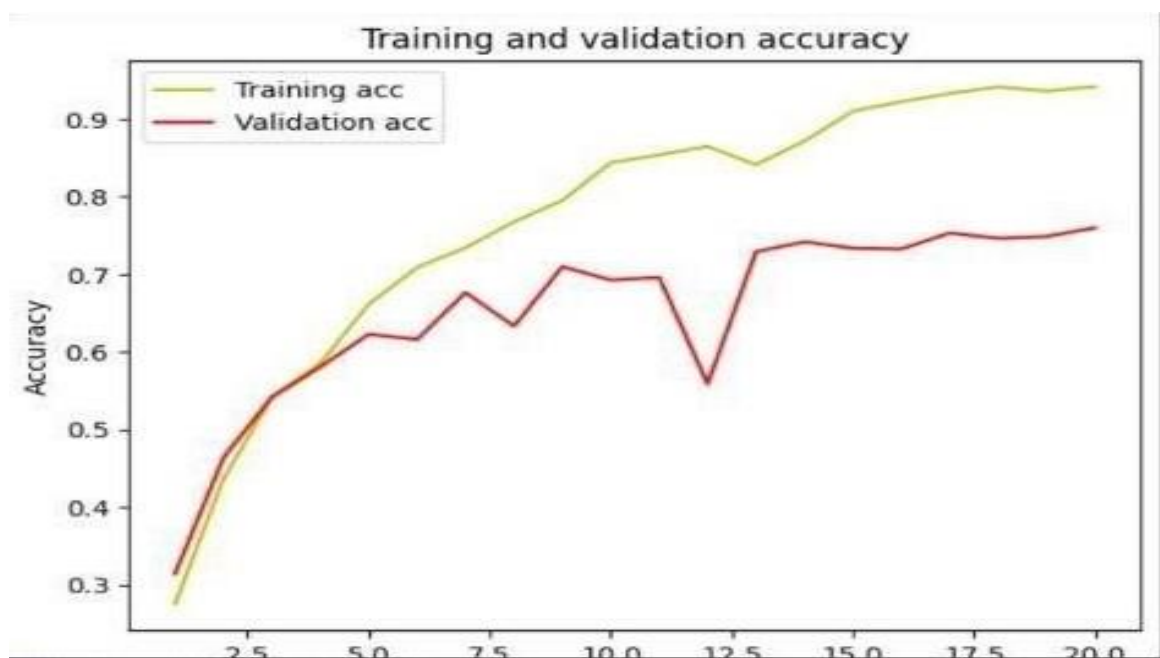


Fig.B.8 COMPARISON OF ACCURACY

Original: 2	Original: 2	Original: 0	Original: 2	Original: 0	Original: 0	Original: 3	Original: 3	Original: 5	Original: 4
Predicted: 2	Predicted: 2	Predicted: 0	Predicted: 2	Predicted: 0	Predicted: 0	Predicted: 3	Predicted: 3	Predicted: 5	Predicted: 1
									

Fig.B.9 PREDICTED IMAGES