Web-based System for Skin Anomaly Classification using Deep Learning Techniques

Submitted in partial fulfillment of the requirements for the degree of

Bachelor of Technology

In

Computer Science with specialization in Data Science

by

Riya Eliza Shaju

19BDS0061

Under the guidance of

Dr. Ilanthendral K,

SCOPE,

VIT, Vellore

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

May, 2023

DECLARATION

I hereby declare that the thesis entitled "Web-based system for Skin Anomaly Classification using Deep Learning Techniques" submitted by me, for the award of the degree of Bachelor of technology in Computer Science with specialization in Data Science to VIT is a record of bonafide work carried out by me under the supervision of Dr. Ilanthenral K.

I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree ordiploma in this institute or any other institute or university.

Place : Vellore

Date : 19/05/2023

Signature of the Candidate

CERTIFICATE

This is to certify that the thesis entitled "Web-based system for Skin Anomaly Classification using Deep Learning Techniques" submitted by Riya Eliza Shaju, 19BDS0061, VIT, for the award of the degree of *Bachelor of technology in Computer Science with specialization in Data Science* is a record of bonafide work carried out by him / her under my supervision during the period, 01. 07. 2022 to 30.04.2023, as per the VIT code of academic and research ethics.

The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute orany other institute or university. The thesis fulfills the requirements and regulations ofthe University and in my opinion meets the necessary standards for submission.

Place : Vellore

Date : 19/05/2023 **Signature of the Guide**

Internal Examiner **External Examiner**

Dr. Parveen Sultana

HOD, Department of Database Systems

ACKNOWLEDGEMENTS

It is my pleasure to express with deep sense of gratitude to Dr. Ilantenral K, Associate Professor Sr. Grade 2, SCOPE, for her constant guidance, continual encouragement, and understanding; more than all, he taught me patience in my endeavor. My association with him / her is not confined to academics only, but it is a great opportunity on my part of work with an intellectual and expert in the field of Data Science.

I would like to express my gratitude to Vellore Institute of Technology for providing with an environment to work in and for his inspiration during the tenure of the course.

In jubilant mood I express ingeniously my whole-hearted thanks to the Head f Department, Dr. Parveen Sultana, all teaching staff and members working as limbs of our university for their not-self-centered enthusiasm coupled with timely encouragements showered on me with zeal, which prompted the acquirement of the requisite knowledge to finalize my course study successfully. I would like to thank my parents for their support.

It is indeed a pleasure to thank my friends who persuaded and encouraged me to take up and complete this task. At last but not least, I express my gratitude and appreciation to all those who have helped me directly or indirectly toward the successful completion of this project.

Place: Vellore

Date: 19/05/2023

Riya Eliza Shaju

Executive Summary

For successful cancer prevention, skin conditions must be identified and treated as soon as possible, especially if they have the potential to become cancerous. However, a lack of knowledge, a lack of awareness, or the inconvenience of extended wait periods to see a dermatologist cause many people all over the world to ignore new skin developments. This delay could have very negative effects. A user-friendly system that can offer a crude diagnostic with the push of a button is suggested as a solution to this.

With Flask, HTML, and CSS, the system incorporates powerful deep learning methods—specifically DenseNet—into a web-based application. Modern computer vision techniques are used by the system to examine images that users upload of their skin anomalies via a web link. It offers statistical data about the anomaly, enabling consumers to determine whether expert consultation is necessary well in advance.

People can avoid spending hours waiting in hospital waiting rooms by completing this process in a matter of minutes. The method prioritizes more serious cases in emergency departments and optimizes healthcare resources by hastening the diagnostic and triaging process.

This straightforward technique seeks to encourage early identification and intervention, potentially saving lives, by utilizing technology to get around information gaps and time constraints. In our fast-paced modern society, it enables people to make knowledgeable decisions about their health and quickly seek the right medical care, which improves the efficiency and effectiveness of healthcare delivery.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

a) Theoretical Background:

The skin disease classification system aims to provide users with a convenient and efficient solution for identifying various skin diseases through the use of deep learning techniques and web technology. This section presents the theoretical background that forms the foundation of the project.

Deep learning, specifically Convolutional Neural Networks (CNN), has revolutionized the field of image recognition and classification. CNN models are particularly effective in analyzing visual data, making them well-suited for tasks such as skin disease classification. These models can automatically extract relevant features from input images, enabling accurate identification of different skin conditions.

Transfer learning is a widely adopted approach in deep learning that leverages pre-trained models to improve the performance and efficiency of new models. In the proposed system, transfer learning techniques are applied to the MobileNet and DenseNet models, which have been pre-trained on large-scale image datasets like ImageNet. By leveraging the knowledge gained from these datasets, the models can learn to recognize and classify skin diseases with high accuracy.

To train and validate the models, a large dataset of skin disease images is required. The proponents have collected 900 photographs of various skin diseases, but it should be noted that the dataset may be unbalanced, meaning that some classes may have fewer samples than others. This imbalance can impact the performance of the models, and therefore, the proponents have explored several sampling techniques and data preprocessing methods to address this issue and improve accuracy.

The implementation of the skin disease classification system utilizes web technologies such as HTML, CSS, and Flask framework. Flask provides a lightweight and efficient framework for developing web applications, enabling seamless integration of the deep learning models into a user-friendly web interface. This allows users to conveniently upload skin images through a web link and receive accurate results regarding the probability of four types of skin diseases, along with corresponding probability percentages.

The system not only aids in early detection of skin diseases but also provides users with information to gauge the severity of the condition and make informed decisions

regarding further medical consultations. By saving patients' waiting time at hospitals and prioritizing more severe cases, the system can help reduce congestion in emergency rooms and improve overall healthcare efficiency.

b) Motivation:

The primary cause of cancer in the majority of individuals worldwide is ignoring a new skin development that may be cancerous. Due to a lack of information, ignorance, or just not having the time to wait in long lines to see a dermatologist, this may be the case. This short ignorance can have grave consequences. In order to increase efficiency in detecting these disease, a user friendly system is proposed that is able to give a primitive diagnosis with the click of a button. The user is given statistics about the anomaly and can then gauge the need to meet a professional well in advance. This process take just a few minutes which intron saves the many hours spent at a waiting room in a hospital.

c) Aim of proposed work:

The study intends to shorten wait times at emergency rooms and make patients feel more comfortable when they notice an unidentified growth on their skin. The user can upload an image to a website, click a link to acquire information on the disease, its potential causes, and its severity, and can communicate with a chatbot about its symptoms and how soon it is best to see a doctor. Dense-net, a pre-trained model that is put on a cloud-based service, is the transfer learning technique used to train the model. The website, which is in step with today's fast-paced society, will aid users in determining the severity. Users can receive a precise primary diagnosis and are then told where to find an expert. By clearing emergency rooms, this product can reduce patient wait times while giving priority to more serious situations in hospitals.

d) Objective of proposed work:

1) To be able to classify skin diseases utilizing genetic information and input image in order to provide precise primary diagnoses and dramatically reduce hospital crowds.
2) To alleviate the initial anxiety and concern individuals often feel when they discover a new, unfamiliar rash or mole on their skin.
3) To forecast how serious the skin condition is and how soon they should see a medical professional.

LITERATURE SURVEY

a) Survey of Existing Models/Work:

Several studies published in Skin disease since the last decade. In a research paper titled “Derm-NN: Skin Diseases Detection Using Convolutional Neural Network" the authors by Tanzina Afroz Rimi, Nishat Sultana, and Md. Ferdouse Ahmed Foysal highlight the importance of the skin as a protective barrier and the prevalence of skin diseases worldwide. The authors emphasize the need for early detection and proper diagnosis to provide appropriate treatment and reduce suffering. They propose using CNN, a type of deep neural network, to classify different skin diseases based on images. The research specifically focuses on dermatitis hand, eczema hand, eczema subcute, lichen simplex, stasis dermatitis, and ulcers. The authors discuss the application of image processing techniques and machine learning in their approach. They train the CNN model using a dataset consisting of 500 images of various skin diseases, including those obtained from the Dermnet dataset and randomly collected from the internet.

The prototype classifier achieves 73% precision in classifying the skin diseases. The paper mentions related literature in the field, including studies on skin disease classification using deep learning algorithms and the application of CNN in skin disease diagnosis. The authors explain the methodologies used in their research, such as image processing, dataset collection, data augmentation, and data preparation. They describe the architecture of their CNN model, which includes convolutional, max-pooling, dropout, and dense layers. Overall, the paper proposes a CNN-based system, Derm-NN, for the detection of skin diseases. The authors demonstrate the effectiveness of their approach by achieving high precision in classifying various skin diseases. The research aims to provide a tool for early diagnosis and raise awareness about skin issues, enabling patients to access assistance remotely through their mobile phones or computer programs.

Another study titled “Automatic skin disease diagnosis using deep learning from clinical image and patient information” proposes a deep learning technique for smartphone-based automatic diagnosis of five common skin diseases. By utilizing clinical images and patient clinical information, the developed system achieves high accuracy, precision, recall, F1-score, and kappa score, with values of 97.5%, 97.7%, 97.7%, 97.5%, and 0.976, respectively. These results indicate that the system exhibits excellent diagnostic performance for the five skin diseases. The authors suggest that the

developed system has the potential to serve as a decision support tool for dermatologists, general practitioners, health practitioners in rural areas, and patients in the diagnosis of skin diseases. The study developed an automated diagnosis system for skin diseases using a pre-trained MobileNet-v2 model. The dataset consisted of 1,376 images collected from patients diagnosed with acne vulgaris, atopic dermatitis, lichen planus, onychomycosis, tinea capitis, as well as images from other less common skin diseases labeled as an unknown class. The images were captured and confirmed by expert dermato-venerologists and tropical dermatologists. Patient information, including age, gender, anatomical sites, and symptoms, was also collected. Before feeding the images into the deep learning network, preprocessing steps such as resizing the images to 224x224 pixels, applying a color constancy algorithm to remove color bias, and data augmentation techniques (rotation, flipping) were performed. The patient information was converted into a feature vector using one-hot encoding. The pre-trained MobileNet-v2 model, known for its performance improvement in mobile models, was repurposed for skin disease classification. It utilizes an inverted residual structure with bottleneck layers and depth-wise separable convolution to reduce computational complexity. The model outputs 1,280 image feature maps to the classifier. By applying transfer learning with the pre-trained MobileNet-v2 model, the proposed system achieves high accuracy, precision, recall, F1-score, and kappa score for skin disease classification. The study adds value by demonstrating the effectiveness of a deep learning-based automatic system using clinical images and patient information for diagnosing five common skin diseases.

The work of Aparna Iyer, Shraddha Iyer and Kshitija Hire titled “A Skin Disease detection system using CNN Deep Learning Algorithm” distinguished between normal skin and skin with abnormalities like pigmentation, rosacea, and acne while also offering advice on how to treat the latter. 5,000 images were gathered by the researchers from Dermnet.com and various outside sources. Using a pretrained MobileNetV2 model on more than a million ImageNet images, transfer learning was used. They created a fully connected layer and trained it to classify the skin into four separate groups. To train the model, a dataset of 5,000 photos was collected from Dermnet.com and other external sources. Transfer learning is employed in this study, where a pre-trained MobileNetV2 model, trained on over a million images from the ImageNet dataset, is utilized to extract visual features from the skin photos. These features are then fed into a fully connected layer, which is added to the model and trained using the collected dataset. By leveraging transfer learning and fine-tuning the fully connected layer, the model is able to categorize the skin images into one of the four categories: normal skin,

pigmentation, rosacea, and acne. The combination of pre-trained features and specific training on the collected dataset enhances the model's ability to accurately classify and provide appropriate treatment recommendations for different skin conditions. The study utilizes transfer learning with a pre-trained MobileNetV2 model to classify skin images into different categories, including normal skin and various skin anomalies. The trained model offers valuable insights by recommending suitable treatment options based on the identified skin condition.

The authors of the article “Convolutional Neural Network for Skin Lesion Classification Understanding the Fundamentals Through Hands-On Learning” have developed a hands-on activity that allows students or beginners to understand the inner workings of a Convolutional Neural Network (CNN) through interactive computer code. The activity provides a detailed description of the steps required to implement and fine-tune a CNN for classifying dermatological images. The examples provided in the activity involve the fine-tuning of a pre-trained ResNet-50 network using a public dataset of skin lesions with different diagnoses. The authors emphasize that the use of specialized toolboxes and libraries simplifies the coding process and makes it accessible to beginners. They believe that the hands-on activity, along with the accompanying description in the article, can serve as a valuable learning tool for students interested in gaining a basic understanding of CNNs.

It can also be used as a tutorial for beginners learning computer programming for building and optimizing CNNs. The activity allows users to execute the code with default parameters to visualize the output of each step. By visualizing the results, users can intuitively grasp the principles of CNNs. For example, plotting the feature maps obtained at different layers helps understand the impact of pooling, batch normalization, and activation layers on the features. Furthermore, the authors suggest that users can explore modifications to the dataset and hyperparameters to observe their effects on the network's performance. Examples could include changing learning rates or freezing specific layers to compare performance. The authors believe that incorporating interactive hands-on activities like this, which replicate complex approaches, can be a powerful strategy for developing problem-solving and analytical skills, particularly through group work in educational settings. They also highlight that making this technology more accessible to non-experts can foster collaboration between dermatologists and computer scientists, leading to improved image-based medical diagnosis.

The hands-on activity and its description in the article offer a valuable learning tool

for understanding CNNs in the context of dermatological image classification. It can be used by students or beginners to gain insights into CNN functioning and can also facilitate collaboration between dermatologists and computer scientists.

The article titled "A Smartphone-Based Skin Disease Classification Using MobileNet CNN" by Jessica Velasco et al describes the use of the MobileNet model with transfer learning to create a skin disease classification system for an Android application. The researchers collected a total of 3,406 images of 7 different skin diseases and explored different sampling methods and preprocessing techniques to improve the accuracy of the MobileNet model. They achieved accuracies ranging from 84.28% to 94.4% depending on the approach used. The article discusses the high prevalence of skin diseases in the Philippines and the potential for utilizing big data and image recognition technology to aid in diagnosis. It mentions previous studies that have used artificial neural networks and deep learning algorithms for skin disease detection. The methodology section explains the dataset used, which consists of images obtained from public dermatology repositories, a color photo atlas of dermatology, and manually taken images. The dataset includes images of acne, eczema, pityriasis rosea, psoriasis, tinea corporis, chickenpox, and vitiligo. The data was divided into training, testing, and validation sets. The researchers used the MobileNet model with transfer learning, where they removed the final classification layer, froze the other layers, and trained the last layer with their dataset. They performed preprocessing on the input images and used specific parameters for the model training. The article also describes the process of deploying the trained model on an Android application. The results and discussion section presents the accuracy and confusion matrices for different experiments conducted by the researchers. They achieved accuracies of 93.6%, 91.8%, and 94.4% using different techniques such as imbalanced dataset training, oversampling, and data augmentation.

In conclusion, these studies demonstrate important advances in the design of transfer learning, convolutional neural networks, data preparation, and augmentation methods for skin disease classification systems. They investigate several techniques, such as the use of multiple datasets, the integration of Android applications, open-source educational activities, and treatment recommendation systems. These developments raise the possibility of precise and effective skin disease detection and treatment.

b) Summary of gaps in existing work:

i) Limited dataset: The paper mentions using a dataset of 500 images for training and testing the model. However, this dataset size is relatively small for training a CNN, which could limit the generalizability of the proposed model. A larger and more diverse dataset would be beneficial for achieving better accuracy and robustness.

ii) Lack of comparison with existing methods: The paper does not provide a comparison of the proposed Derm-NN model with existing methods or state-of-the-art approaches for skin disease detection. Without benchmarking the performance of the proposed model against other methods, it is difficult to assess its effectiveness and superiority.

iii) Evaluation metrics: The paper mentions achieving 73% precision with the implemented system on the dermnet dataset. However, it does not provide comprehensive evaluation metrics such as accuracy, sensitivity, specificity, or area under the curve (AUC) to evaluate the overall performance of the model. Without these metrics, it is challenging to assess the true effectiveness of the proposed approach.

iv) Lack of detail on the CNN architecture: Although the paper briefly mentions the architecture of the CNN used in the study, it does not provide detailed information about the layers, parameters, or optimization techniques employed. This lack of information makes it difficult to replicate the experiment or understand the specific choices made in designing the CNN.

v) Limited scope of skin diseases: The paper focuses on detecting and classifying five specific skin diseases, namely eczema hand, eczema nummular, eczema subcute, lichen simplex, stasis dermatitis, and ulcers. While this is a valuable contribution, it limits the applicability of the proposed model to a narrow range of skin diseases. Including a broader range of skin conditions would enhance the usefulness and impact of the system.

vi) External validation: The paper does not mention external validation or testing of the proposed model on independent datasets or real-world clinical scenarios. External validation is crucial to assess the generalizability and real-world performance of the model. Without such validation, the practical utility of the proposed approach remains uncertain.

vii) Imbalanced dataset: Imbalanced datasets, where certain classes have

significantly fewer samples than others, can pose challenges in training and evaluating models. It may lead to biased predictions and lower accuracy for minority classes.

viii) Lack of diversity: Skin diseases can vary across different ethnicities and regions. If the dataset used in the papers does not adequately represent diverse populations, the model's performance may be limited to specific demographic groups.

ix) Overfitting: Overfitting occurs when a model performs well on the training data but fails to generalize to new, unseen data. It can happen if the model is too complex relative to the size of the dataset or if data augmentation techniques are not effectively applied.

x) Limited clinical validation: While deep learning models can achieve high accuracy in classifying skin diseases, the translation of these models into clinical practice requires rigorous validation and testing on real-world patient data. The absence of clinical validation can limit the practical applicability of the proposed models.

xi) Hardware and deployment constraints: In some cases, the practical deployment of deep learning models on mobile devices or resource-constrained environments may pose challenges. The papers may not thoroughly address the hardware requirements, model size, latency, or power consumption considerations.

In summary, the highlighted drawbacks include limited exploration of alternative sampling approaches and preprocessing methods, the lack of comprehensive validation and comparison in the interactive exercise, potential limitations in dealing with rare or complex skin diseases, the reliance on visual data in the smartphone-based diagnostic system, and the potential challenges related to internet connectivity in remote areas. These limitations should be considered for future research and development to enhance the effectiveness, reliability, and accessibility of skin disease classification systems.

OVERVIEW OF PROPOSED SYSTEM

a) Introduction:

The proposed approach for classifying skin illnesses intends to offer a quick and simple platform for detecting different skin conditions. A web application that enables users to upload photographs of their skin anomalies and get fast classification results was created using a combination of HTML, CSS, and Flask. To produce precise and trustworthy classification results, the system uses deep learning models such as Convolutional Neural Networks (CNN), MobileNet, and DenseNet.

In terms of prompt diagnosis and treatment, skin disorders present considerable hurdles. Dermatologists are frequently out of reach and patients frequently have long wait times, which delays treatment and could have significant implications. The suggested method tackles these problems by giving consumers an easy-to-use tool to get a quick understanding of their skin diseases.

i) Web app: The system is based on HTML, CSS, and Flask, utilizing these technologies to produce a user-friendly and attractive web interface. The many components of the user interface are organized using HTML, and CSS improves the look and feel to make using it more enjoyable. Flask, a micro web framework, enables effective data processing and model integration by facilitating smooth communication between the front-end and back-end components.

ii) Models: The system's deep learning model integration is its primary functional component. CNNs are renowned for their efficiency in image processing tasks, which makes them perfect for classifying skin conditions. To take use of its feature extraction capabilities, the MobileNet model, a well-known pre-trained deep learning architecture, is used. The DenseNet model is also used because it can more accurately classify data and capture complicated patterns.

iii) Dataset: An extensive dataset of skin illness photos is used to make sure the algorithm is reliable. The collection comprises a wide variety of skin defects that

represent frequent clinical practice problems. The related disease categories are identified and annotated on the photos, allowing supervised learning to train the deep learning models.

The suggested skin condition classification system seeks to close the gap between patients and dermatologists by utilizing cutting-edge deep learning algorithms and incorporating them into a user-friendly web application. It enables people to have a basic understanding of their skin diseases, promoting informed decision-making and perhaps lightening the load on medical services. The system has enormous promise for helping people with skin health issues in our fast-paced modern society in a timely and accessible manner.

b) Related Concepts:

i) Machine Learning (ML):

Machine Learning is a field of study that focuses on developing algorithms and models that allow computers to learn from data and make predictions or decisions without being explicitly programmed. It involves the construction of mathematical models that can analyze and interpret complex datasets, identify patterns, and make informed predictions or decisions. ML algorithms can be categorized into three main types: supervised learning, unsupervised learning, and reinforcement learning.

Supervised learning algorithms learn from labeled examples, where the input data is paired with corresponding output labels. These algorithms can then make predictions or classify new, unseen data based on the patterns they learned from the labeled examples. Unsupervised learning algorithms, on the other hand, analyze unlabeled data to identify patterns or structures within the dataset. They can cluster similar data points together or find underlying relationships between variables. Reinforcement learning algorithms learn by interacting with an environment and receiving feedback in the form of rewards or penalties. They aim to maximize the cumulative rewards by taking optimal actions in a given context.

ii) Deep Learning (DL):

Deep Learning is a subset of Machine Learning that focuses on using artificial neural networks with multiple layers, known as deep neural networks, to learn and extract high-level features from data. DL models are inspired by the

structure and functioning of the human brain, where each layer of neurons learns to recognize specific patterns or features. Deep learning architectures, such as Convolutional Neural Networks (CNNs) for image processing or Recurrent Neural Networks (RNNs) for sequential data, can automatically learn hierarchical representations from raw data. These models can handle large-scale datasets and extract intricate patterns, making them highly effective in tasks such as image and speech recognition, natural language processing, and generative modeling.

iii) Computer Vision (CV):

Computer Vision is a subfield of Artificial Intelligence that focuses on enabling computers to gain a high-level understanding of visual data, such as images or videos. It involves developing algorithms and techniques to extract meaningful information from visual inputs and make sense of the visual world. CV algorithms aim to replicate human visual perception and interpret images in a way that is similar to how humans do. This includes tasks such as image classification, object detection and tracking, image segmentation, and image synthesis.

Computer vision algorithms often utilize deep learning models, particularly Convolutional Neural Networks (CNNs), to process and analyze images. CNNs can automatically learn and extract features from raw image data, enabling tasks such as image recognition and object detection. CV has applications in various domains, including autonomous vehicles, surveillance systems, medical imaging, augmented reality, and robotics. It plays a crucial role in enabling machines to understand and interpret visual information, paving the way for advancements in fields like self-driving cars, medical diagnostics, and smart surveillance systems.

iv) Tensorflow:

TensorFlow is an open-source machine learning framework that has gained immense popularity due to its versatility, scalability, and extensive community support. Developed by Google Brain, TensorFlow provides a comprehensive ecosystem for developing and deploying machine learning models, making it a go-to choice for researchers and practitioners in the field. At its core, TensorFlow is a computational framework that allows users to define and execute complex mathematical computations efficiently. It utilizes a dataflow

graph model, where nodes represent mathematical operations and edges represent the flow of data between these operations. This graph-based approach enables efficient parallel execution, making TensorFlow capable of handling large-scale computations on various hardware architectures, including CPUs, GPUs, and specialized accelerators.

TensorFlow supports a wide range of machine learning tasks, including but not limited to deep learning, reinforcement learning, and natural language processing. It provides a rich set of APIs and tools that simplify the process of model development, training, and evaluation. The high-level TensorFlow APIs, such as Keras, offer an intuitive and user-friendly interface for building neural networks and other machine learning models, abstracting away the low-level details and reducing development time.

Another key feature of TensorFlow is its support for distributed computing. TensorFlow can seamlessly distribute computations across multiple devices or machines, enabling the training and inference of large-scale models on clusters or cloud platforms. This distributed nature of TensorFlow empowers researchers and organizations to leverage the power of parallel processing and tackle computationally intensive tasks efficiently.

Furthermore, TensorFlow has a vibrant and active community that contributes to its growth and development. The TensorFlow community regularly releases updates, bug fixes, and new features, ensuring the framework stays relevant and up to date with the latest advancements in machine learning. This community support also provides ample resources, tutorials, and pre-trained models that facilitate learning and implementation.

In recent years, TensorFlow has evolved beyond traditional machine learning and has expanded its capabilities to include areas like computer vision, natural language processing, and generative models. TensorFlow's integration with other popular libraries, such as TensorFlow Probability and TensorFlow Federated, further extends its functionality and makes it a comprehensive platform for a wide range of applications.

Overall, TensorFlow has revolutionized the field of machine learning by providing a powerful, flexible, and scalable framework for developing and deploying models. Its ability to handle complex computations, support distributed computing, and its rich ecosystem of tools and resources make it a top choice for both beginners and experienced practitioners in the field of

machine learning and artificial intelligence.

v) Keras:

Keras is a high-level neural network library written in Python that is built on top of the TensorFlow framework. It provides a user-friendly and intuitive interface for designing, training, and evaluating deep learning models. Keras aims to make deep learning accessible to both beginners and experienced practitioners by offering a simplified and streamlined API. One of the key features of Keras is its modular architecture. It allows users to easily build neural networks by stacking together different types of layers. These layers can be combined to create complex network architectures, including feedforward networks, recurrent neural networks (RNNs), and convolutional neural networks (CNNs). Keras provides a wide range of pre-defined layers with various activation functions, regularization techniques, and normalization methods, making it easy to construct models tailored to specific tasks.

Keras also simplifies the process of defining the model's objective function and selecting optimization algorithms. It provides a variety of loss functions and evaluation metrics, allowing users to customize their models based on their specific needs. Additionally, Keras supports popular optimization algorithms such as Stochastic Gradient Descent (SGD), Adam, and RMSprop, which can be easily configured with adjustable learning rates and momentum.

Another notable feature of Keras is its support for model training and evaluation. It offers a high-level API for training models on both CPU and GPU, allowing users to leverage the power of parallel processing to accelerate training times. Keras also provides convenient methods for data preprocessing, including data augmentation, which helps to prevent overfitting and improve model generalization.

Keras supports seamless integration with other Python libraries and frameworks, making it versatile and flexible for a wide range of applications. It can be easily combined with TensorFlow, allowing users to take advantage of TensorFlow's low-level capabilities when needed. Keras also supports interoperability with other deep learning libraries such as Theano and Microsoft Cognitive Toolkit (CNTK).

Keras is a powerful and user-friendly library that has democratized deep learning by simplifying the process of building, training, and evaluating neural

network models. Its modular architecture, extensive pre-defined layers, and optimization algorithms make it accessible to beginners while still providing flexibility for advanced users. With its seamless integration with TensorFlow and other libraries, Keras has become a popular choice among researchers and practitioners in the field of deep learning.

vi) Python Packages (for model training):

(1) `numpy`: A library for numerical computing in Python, providing efficient and convenient array operations for mathematical and scientific computations.

(2) `Adam`: An optimization algorithm in TensorFlow's Keras API that adapts the learning rate during training to efficiently update model weights and improve convergence.

(3) `np_utils`: A utility module in Keras that provides functions for working with categorical data, including one-hot encoding and decoding.

(4) `Activation`: A layer in Keras that applies a specific activation function to the output of a previous layer, introducing non-linearity in the neural network.

(5) `Dropout`: A regularization technique in Keras that randomly sets a fraction of input units to 0 during training, reducing overfitting and improving generalization.

(6) `Convolution2D`: A layer in Keras that performs 2D convolution on input data, extracting features using filters and producing a feature map.

(7) `GlobalAveragePooling2D`: A layer in Keras that performs global average pooling on 2D input data, reducing spatial dimensions and extracting global information.

(8) `Sequential`: A linear stack of layers in Keras, used for building deep learning models by adding layers sequentially.

(9) `tensorflow`: An open-source machine learning framework that provides a wide range of tools and libraries for building and training machine learning models.

(10) `tensorflow.keras.applications. mobilenet`: A module in TensorFlow's Keras API that provides pre-trained MobileNet models for image classification tasks.

(11) `os`: A module in Python that provides functions for interacting with the operating system, allowing operations such as file and directory manipulation.

(12) `Dense`: A layer in Keras that represents a fully connected layer, where each neuron is connected to every neuron in the previous and next layer, allowing for complex non-linear mappings.

(13) `Dropout`: A regularization technique in Keras that randomly sets a fraction of input units to 0 during training, reducing overfitting and improving generalization.

(14) `MaxPooling2D`: A layer in Keras that performs 2D max pooling on input data, reducing spatial dimensions and retaining the most important features.

(15) `Flatten`: A layer in Keras that flattens the multi-dimensional input into a 1D array, allowing it to be connected to fully connected layers.

(16) `Dense` layer is commonly used in the output layers of neural networks for classification or regression tasks, where it produces the final predictions based on the learned features.

(17) `Dropout` layer helps in preventing overfitting by randomly dropping a certain percentage of neuron units during training, forcing the network to learn redundant representations.

(18) `Conv2D` layer performs 2D convolution by sliding filters over the input data to extract important spatial features.

c) Framework, Architecture or Module for the proposed system:

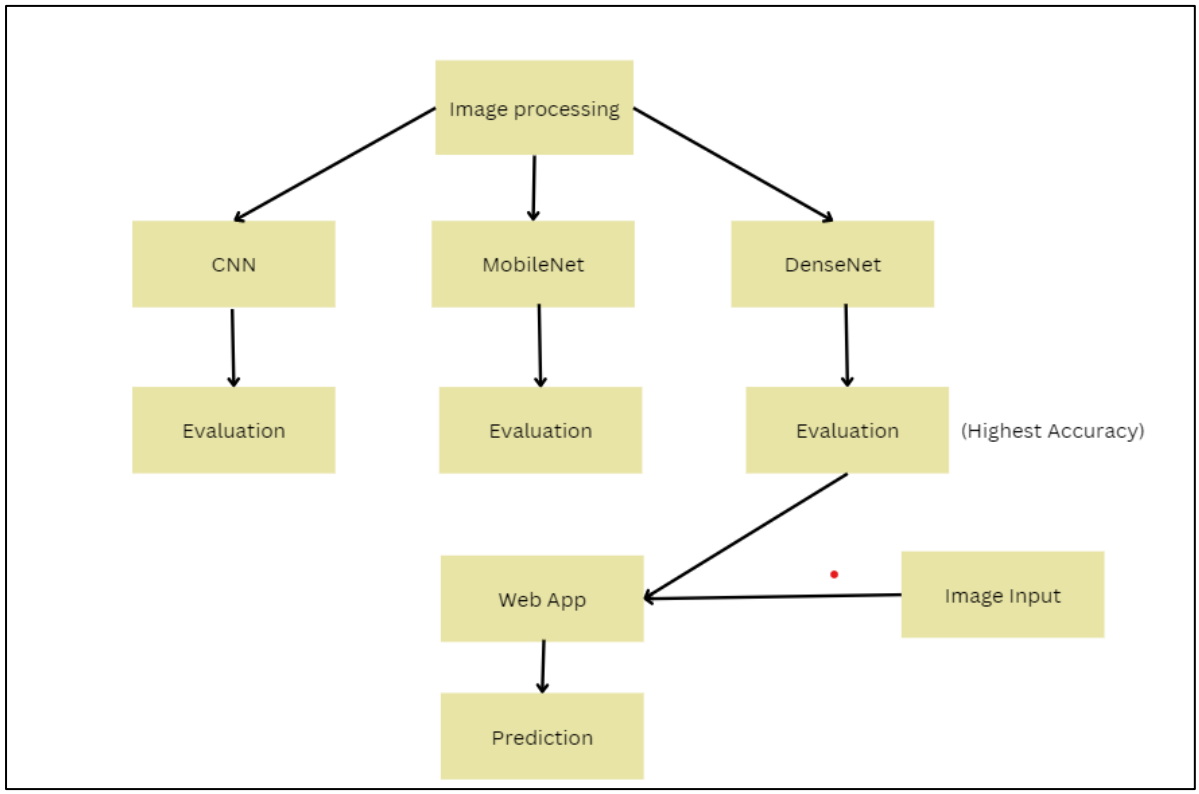


Fig.1: Overview of the proposed system

The suggested approach analyzes skin anomaly photographs in real time and provides the top 4 recommendations for what the anomaly may be with at least 80% accuracy. CNN, Mobile-net, and Dense-net are the three models that were trained and contrasted. The web app made with HTML and CSS and coupled with Flask uses the Dense-net model, which has the best performance with a training accuracy of 74%. The Flask program may then accept a user-submitted image and make predictions even when there is no internet connection.

d) Dataset creation:

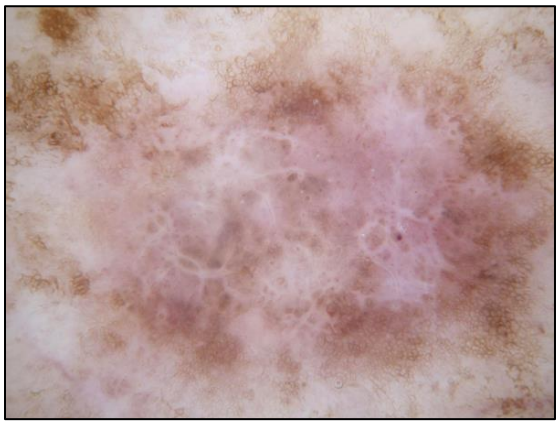
The dataset is self-created and loosely inspired by the ISIC skin disease dataset. The final dataset used to train the models contain images of the following skin anomalies:

i) Actinic keratosis: Actinic keratosis is a common skin condition characterized by rough, scaly patches on sun-exposed areas of the body. These patches, also known as solar keratosis, are usually small, red, and feel rough to the touch. Actinic keratosis is caused by cumulative sun damage over time and is considered a precancerous condition. If left untreated, it can develop into squamous cell carcinoma, a type of skin cancer. Treatment options for actinic keratosis include cryotherapy (freezing), topical medications, and photodynamic therapy.

ii) Atopic Dermatitis: Atopic dermatitis, commonly known as eczema, is a chronic inflammatory skin condition that affects both children and adults. It is characterized by dry, itchy, and inflamed skin patches that can become red, swollen, and cracked. Atopic dermatitis is believed to be caused by a combination of genetic and environmental factors, such as allergens and irritants. Management of atopic dermatitis involves avoiding triggers, moisturizing the skin, using topical corticosteroids or immunomodulators, and practicing good skincare habits.

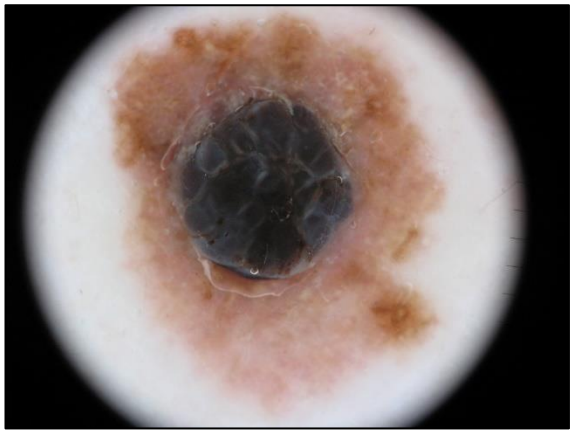
iii) Benign keratosis: Benign keratosis, also known as seborrheic keratosis, refers to non-cancerous growths that appear on the skin. These growths are usually brown, black, or tan and have a waxy, scaly texture. Benign keratoses are commonly found in middle-aged or older individuals and can occur anywhere on the body. While they are harmless and do not require treatment, they may be removed if they cause itching, irritation, or aesthetic concerns.

iv) Dermatofibroma: Dermatofibroma is a benign skin lesion that typically presents as a small, firm, reddish-brown bump on the skin. It is commonly found on the legs but can occur on other areas as well. Dermatofibromas are usually painless and do not require treatment unless they become symptomatic or cosmetically bothersome. While the exact cause is unknown, dermatofibromas may develop in response to minor skin injuries or insect bites. They are composed of fibrous tissue and may feel like a small pebble under the skin.



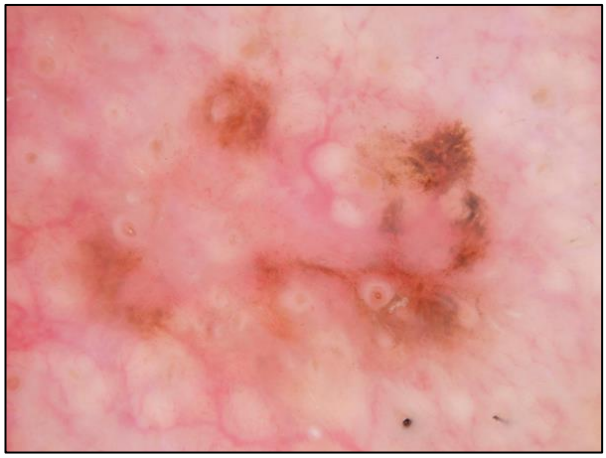
v) Melanocytic nevus: Melanocytic nevi, also known as moles, are common pigmented skin lesions. They can be flat or raised and range in color from brown to black. Melanocytic nevi are typically benign and appear during childhood or adolescence. However, some nevi may have atypical features or exhibit changes over time, requiring evaluation by a dermatologist to rule out melanoma, a type of skin cancer. Regular self-examination and monitoring of moles are important for detecting any suspicious changes and seeking medical attention if necessary.

vi) Melanoma: Melanoma is a type of skin cancer that develops in the cells that produce melanin, the pigment responsible for skin color. It often appears as a dark, irregularly shaped mole or lesion that may change in size, shape, or color over time. Melanoma can occur on any part of the body, including areas not exposed to the sun. It is the most dangerous form of skin cancer, as it has the potential to spread to other parts of the body. Early detection and prompt treatment are crucial for



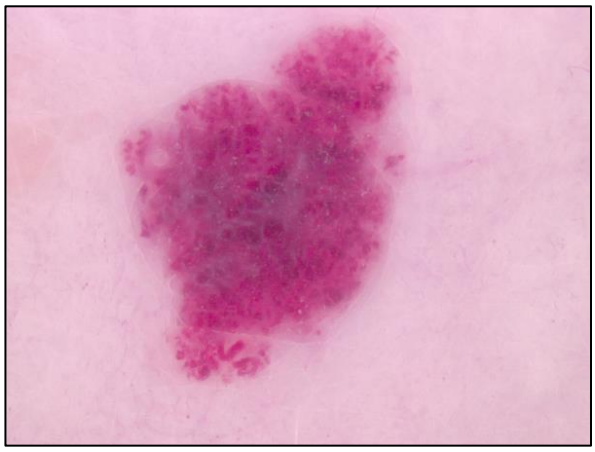
favorable outcomes. Treatment options for melanoma include surgical removal of the tumor, chemotherapy, radiation therapy, targeted therapy, and immunotherapy.

vii) Squamous cell carcinoma: Squamous cell carcinoma is a common type of skin cancer that arises from the squamous cells, which are found in the upper layers of the skin. It typically appears as a red, scaly, or crusted patch or bump that may bleed or develop a crust. Squamous cell carcinoma is often associated with prolonged sun exposure, but it can also develop on areas of the skin that are not frequently exposed to the sun. Treatment options for squamous cell carcinoma depend on the size, location, and stage of the cancer, but they may include surgical excision, radiation therapy, cryotherapy, and topical medications.



viii) Tinea (Ringworm) Candidiasis: Tinea, commonly known as ringworm, is a fungal infection that can affect the skin, nails, and scalp. It is characterized by circular or ring-shaped rashes that may be red, itchy, and scaly. Candidiasis, on the other hand, is a fungal infection caused by the Candida species, primarily Candida albicans. It can affect various parts of the body, including the skin, mouth, throat, and genital area. Both tinea and candidiasis thrive in warm and moist environments. Treatment options for these fungal infections include antifungal medications, topical creams or ointments, and maintaining good hygiene practices.

ix) Vascular Lesion: Vascular lesions are abnormalities in blood vessels that can affect the skin or other tissues. They can manifest as birthmarks, such as port-wine stains or hemangiomas, or acquired lesions like spider veins or cherry angiomas. Vascular lesions can vary in size, color, and appearance. While most vascular lesions are benign and do not require treatment, some may cause cosmetic concerns or pose a risk of bleeding or other complications. Treatment options for vascular lesions include laser therapy, sclerotherapy, surgical removal, or embolization, depending on the type and severity of the lesion. It is essential to consult a healthcare professional for accurate diagnosis and appropriate management of vascular lesions.



e) Models used:

i) Convolutional Neural Networks (CNN):

Convolutional Neural Networks (CNNs) have emerged as a groundbreaking approach in the field of deep learning and computer vision. Originally inspired by the structure and functioning of the visual cortex in humans, CNNs have revolutionized image recognition and analysis tasks. With their ability to automatically learn and extract intricate features from images, CNNs have become the cornerstone of many state-of-the-art computer vision applications.

At the core of a CNN are convolutional layers that perform local receptive field operations, capturing spatial dependencies and patterns in the input data. These layers are followed by pooling layers that downsample the feature maps, reducing computational complexity and providing translation invariance. The extracted features are then passed through fully connected layers, enabling the network to make high-level predictions.

One of the key strengths of CNNs lies in their ability to learn hierarchical representations of visual data. Through the use of multiple convolutional and

pooling layers, CNNs can capture low-level features such as edges and textures, gradually building up to more complex and abstract representations. This hierarchical feature learning enables CNNs to excel in tasks such as image classification, object detection, and segmentation.

Training a CNN involves a two-step process: forward propagation and backpropagation. During forward propagation, the input data is fed through the network, and the predicted outputs are compared to the ground truth labels. The resulting error is then backpropagated through the network, updating the weights and biases to minimize the error. This iterative process of forward and backward passes allows the CNN to learn and optimize its parameters.

The success of CNNs can be attributed to their ability to automatically learn relevant features from raw data, eliminating the need for manual feature engineering. This data-driven approach, combined with their hierarchical feature learning, has propelled CNNs to achieve remarkable results in various image-based tasks. From image classification to medical diagnosis and autonomous driving, CNNs have become a powerful tool in the realm of computer vision, pushing the boundaries of what is possible in visual recognition and analysis.

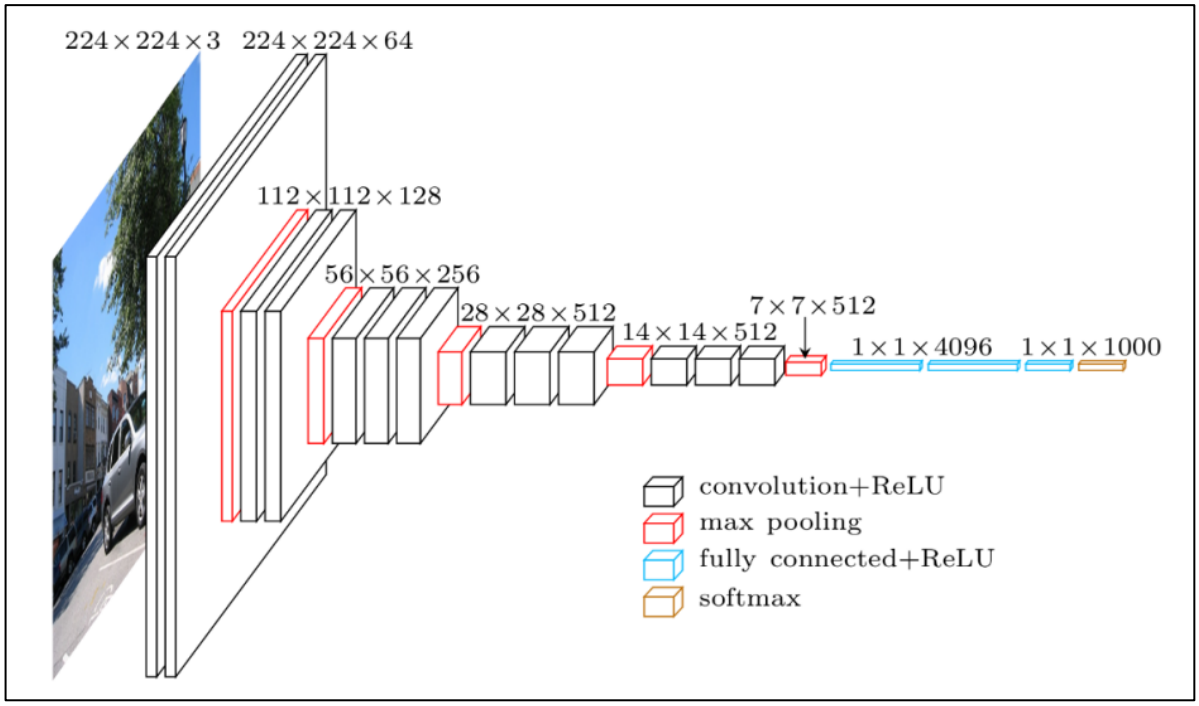


Fig.2: Overview of CNN architecture

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_1 (Conv2D)           (None, 238, 238, 8)       224

 conv2d_2 (Conv2D)           (None, 236, 236, 16)      1168

 max_pooling2d (MaxPooling2D  (None, 118, 118, 16)     0
 )

 dropout_2 (Dropout)         (None, 118, 118, 16)      0

 conv2d_3 (Conv2D)           (None, 116, 116, 32)      4640

 max_pooling2d_1 (MaxPooling  (None, 58, 58, 32)       0
 2D)

 conv2d_4 (Conv2D)           (None, 56, 56, 32)        9248

 max_pooling2d_2 (MaxPooling  (None, 28, 28, 32)       0
 2D)

 dropout_3 (Dropout)         (None, 28, 28, 32)        0

 flatten (Flatten)           (None, 25088)             0

 dense_1 (Dense)             (None, 64)                1605696

 dropout_4 (Dropout)         (None, 64)                0

 dense_2 (Dense)             (None, 9)                 585

=================================================================
Total params: 1,621,561
Trainable params: 1,621,561
Non-trainable params: 0
_________________________________________________________________
```

Fig.3: Layers of CNN architecture implemented (This model was used as baseline)

ii) MobileNet:

MobileNet is a highly efficient convolutional neural network (CNN) architecture specifically designed for mobile and embedded devices. It addresses the challenge of deploying deep learning models on resource-constrained platforms without compromising accuracy. MobileNet achieves this by employing depth-wise separable convolutions, which significantly reduce the computational complexity and model size while maintaining strong performance.

Unlike traditional CNN architectures that apply standard convolutions on the entire input volume, MobileNet separates the spatial and channel-wise convolutions into two distinct operations. In the depth-wise convolution step, MobileNet performs a separate convolutional operation for each input channel, applying a single filter to each channel independently. This reduces the number of parameters and operations required, resulting in computational efficiency.

Following the depth-wise convolutions, MobileNet utilizes point-wise convolutions to combine the information from different channels. Point-wise convolutions apply 1x1 convolutions to mix and transform the features spatially. This helps to capture complex relationships between channels while keeping the computational cost low.

MobileNet provides a trade-off between model size and accuracy by introducing a parameter called the "width multiplier." This parameter controls the number of filters in each layer, allowing users to scale the model according to their specific requirements. By adjusting the width multiplier, developers can balance the model's size and computational efficiency based on the available resources and performance needs.

The architecture of MobileNet can be customized to meet different constraints and applications. It offers a range of pre-trained models that have been trained on large-scale image datasets, such as ImageNet, enabling transfer learning and fast deployment in various computer vision tasks.

MobileNet has been widely adopted in mobile and embedded applications, including image classification, object detection, and semantic segmentation. Its lightweight design and efficient operations make it particularly suitable for real-time applications on devices with limited computational power and memory, such as smartphones, drones, and Internet of Things (IoT) devices.

Overall, MobileNet represents a significant advancement in deep learning architectures, enabling efficient deployment of CNN models on mobile and embedded platforms while maintaining competitive performance and accuracy. Its versatility and effectiveness have contributed to its widespread adoption and impact in the field of computer vision.

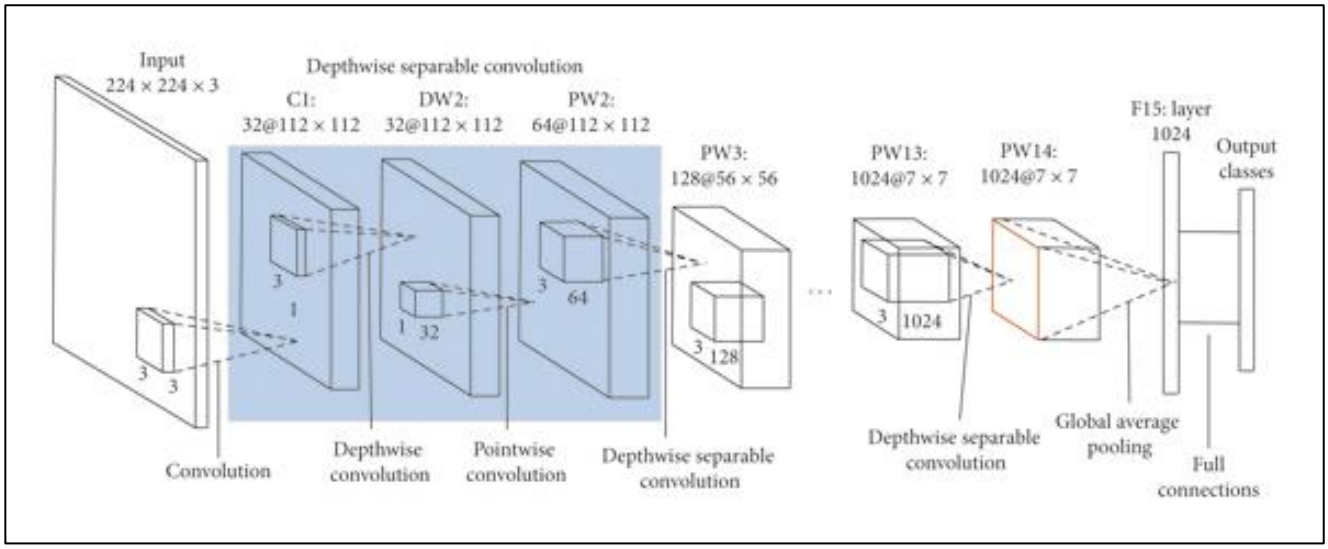


Fig.4: Overview of MobileNet architecture

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 mobilenet_1.00_224 (Functio  (None, 7, 7, 1024)       3228864
 nal)

 dropout_1 (Dropout)         (None, 7, 7, 1024)        0

 conv2d (Conv2D)             (None, 7, 7, 9)           9225

 activation (Activation)     (None, 7, 7, 9)           0

 global_average_pooling2d_1   (None, 9)                0
 (GlobalAveragePooling2D)

 activation_1 (Activation)   (None, 9)                 0

=================================================================
Total params: 3,238,089
Trainable params: 3,216,201
Non-trainable params: 21,888
_________________________________________________________________
```

Fig 5: MobileNet model used to compare performance

iii) Dense-Net:

DenseNet is a deep convolutional neural network (CNN) architecture that addresses the challenges of information flow and parameter efficiency. Unlike traditional CNNs, which pass information layer by layer, DenseNet establishes dense connections between layers, allowing direct connections between all layers within the network. This dense connectivity enhances gradient flow, promotes feature reuse, and mitigates the vanishing gradient problem, leading to improved training performance and feature representation.

The core building block of DenseNet is the "dense block." Within a dense block, each layer receives direct inputs from all preceding layers and passes its feature maps to all subsequent layers. This dense connectivity enables feature reuse, as each layer has access to the feature maps produced by all preceding layers, fostering richer feature representation and information propagation through the network.

To manage the increase in feature map dimensions within the dense block, DenseNet employs bottleneck layers. These bottleneck layers reduce the dimensionality of the feature maps using 1x1 convolutions before the subsequent 3x3 convolutions, reducing the computational cost while maintaining

representational capacity.

DenseNet's dense connectivity and bottleneck layers lead to a highly compact and parameter-efficient architecture. By eliminating the need for each layer to learn redundant features independently, DenseNet achieves parameter sharing, reducing the total number of parameters required compared to traditional CNN architectures. This parameter efficiency makes DenseNet particularly well-suited for scenarios with limited resources, such as mobile and embedded devices.

DenseNet has demonstrated impressive performance across various computer vision tasks, including image classification, object detection, and semantic segmentation. Its dense connectivity enables the network to capture fine-grained details and learn intricate patterns, leading to improved accuracy and robustness.

Additionally, DenseNet has also facilitated transfer learning by providing pre-trained models trained on large-scale image datasets such as ImageNet. These pre-trained models enable faster convergence and better generalization on specific tasks, making DenseNet an attractive choice for researchers and practitioners.

Overall, DenseNet's dense connectivity, bottleneck layers, and parameter efficiency have positioned it as a powerful architecture in the field of deep learning. Its ability to facilitate information flow, feature reuse, and parameter sharing has led to significant advancements in various computer vision applications, making DenseNet a valuable tool for both academic research and real-world deployment.

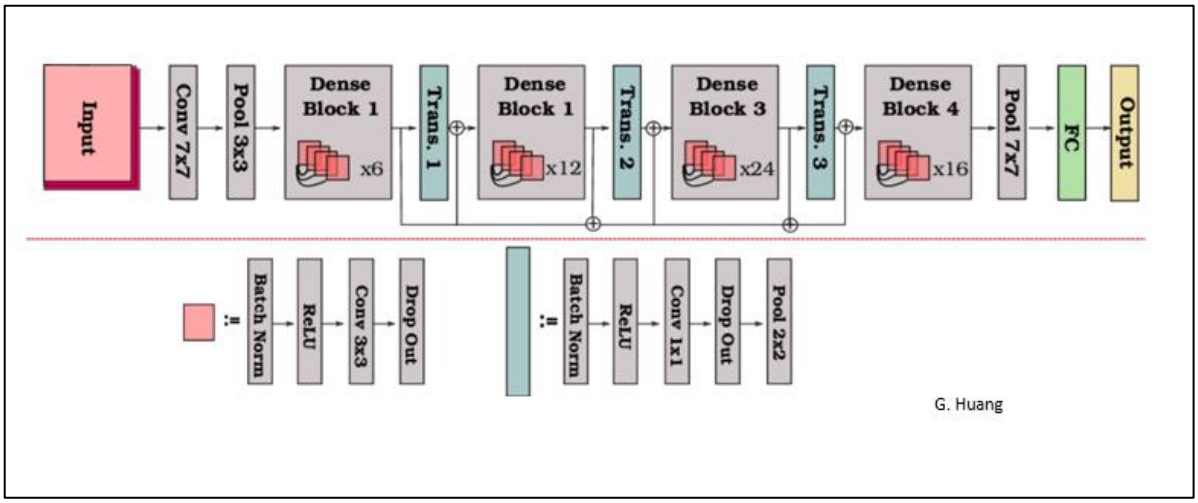


Fig.6: Overview of DenseNet model

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 densenet121 (Functional)    (None, 7, 7, 1024)        7037504

 global_average_pooling2d (G  (None, 1024)             0
 lobalAveragePooling2D)

 dropout (Dropout)           (None, 1024)              0

 dense (Dense)               (None, 9)                 9225

=================================================================
Total params: 7,046,729
Trainable params: 6,963,081
Non-trainable params: 83,648
_________________________________________________________________
```

Fig 7: DenseNet model used. This model had the highest performance

f) Proposed System Model (ER Diagram):

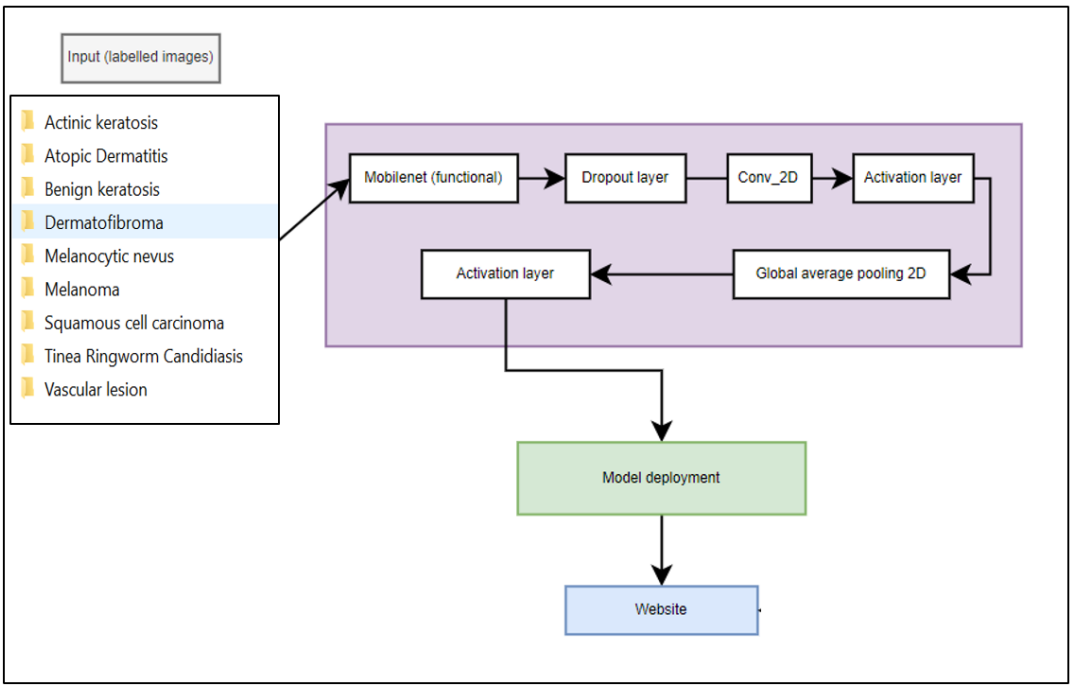
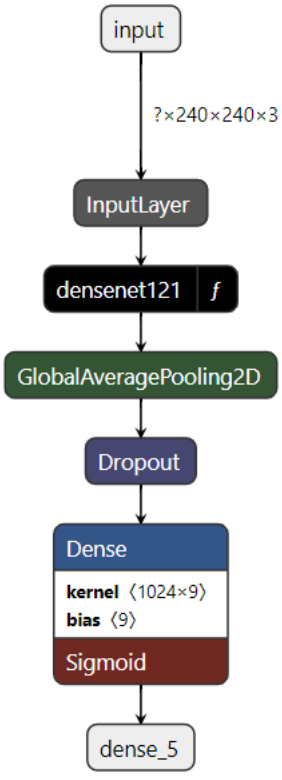


Fig.8: ER diagram of the proposed model

g) Model Architecture (Dense-Net):




```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 densenet121 (Functional)    (None, 7, 7, 1024)        7037504

 global_average_pooling2d (G  (None, 1024)             0
 lobalAveragePooling2D)

 dropout (Dropout)           (None, 1024)              0

 dense (Dense)               (None, 9)                 9225

=================================================================
Total params: 7,046,729
Trainable params: 6,963,081
Non-trainable params: 83,648
_________________________________________________________________
```

Fig.9&10: DenseNet layers implemented

h) Web-app:

The front-end of the web-app was created using HTML and CSS, the backend using Python and integration using flask.

i) HTML (Hypertext Markup Language): HTML, which stands for Hypertext Markup Language, is a standard markup language used for creating web pages and applications. It provides the structure and semantics for content on the internet. HTML uses tags to define elements and their characteristics, allowing browsers to interpret and display the content correctly. HTML is the backbone of the web, as it defines the structure of a webpage. It includes various elements such as headings, paragraphs, lists, images, links, tables, forms, and more. These elements allow developers to structure and organize content, making it more readable and accessible to users. With HTML, developers can create a hierarchical structure by using different tags to indicate the relationship between elements. For example, the <h1> tag represents the main heading, while <h2> and <h3> tags represent subheadings. This structure helps search engines and screen readers understand the content and navigate through it. HTML also supports the use of attributes, which provide additional information about elements. Attributes can be used to define the appearance, behavior, or functionality of elements. For instance, the "src" attribute in the <img> tag specifies the source URL of an image, while the "href" attribute in the <a> tag defines the hyperlink destination. Furthermore, HTML supports the concept of forms, allowing users to input data and interact with web pages. Form elements such as text fields, checkboxes, radio buttons, dropdown menus, and submit buttons can be used to collect user information, process data, and send it to a server for further processing. HTML is a versatile language that can be combined with other technologies like CSS (Cascading Style Sheets) and JavaScript to enhance the visual appearance and functionality of web pages. CSS is used to style and layout HTML elements, while JavaScript adds interactivity and dynamic behavior.

ii) CSS (Cascading Style Sheets): CSS, which stands for Cascading Style Sheets, is a style sheet language used to describe the presentation and formatting of HTML documents. It provides a way to separate the content and structure of a webpage from its visual design, allowing developers to create visually appealing and consistent web pages. CSS works by applying styles to HTML elements through

selectors and declarations. Selectors target specific HTML elements, classes, IDs, or other attributes, while declarations define the styles to be applied, such as color, font size, margin, padding, and more. One of the key benefits of CSS is its ability to control the layout and positioning of elements on a webpage. With CSS, developers can create responsive designs that adapt to different screen sizes and devices, providing an optimal viewing experience for users on desktops, tablets, and mobile devices. CSS also enables the creation of visually engaging designs through the use of backgrounds, borders, shadows, gradients, and animations. It allows for precise control over typography, including font styles, sizes, line heights, and spacing, ensuring readability and aesthetic appeal. Moreover, CSS supports the concept of cascading, which means that styles can be applied in a hierarchical manner. This allows for the easy management of styles across multiple web pages, as changes made to a single CSS file can be reflected throughout the entire website. CSS is continuously evolving, with new features and capabilities being added regularly. It is supported by all modern web browsers and has become an essential tool for web development. With CSS, developers can transform plain HTML documents into visually stunning and user-friendly web pages, enhancing the overall user experience.

iii) Flask: Flask is a lightweight and flexible web framework for Python that provides a simple yet powerful way to build web applications. It follows the model-view-controller (MVC) architectural pattern, allowing developers to separate the application logic from the presentation layer. One of the main advantages of Flask is its simplicity. It has a minimalistic design and does not impose any specific project structure or dependencies. This makes it easy to get started and allows developers to have more control over their application. Flask provides a wide range of features and functionalities to build web applications. It has built-in support for routing, allowing developers to define URL routes and associate them with specific view functions. It also supports template rendering, allowing the dynamic generation of HTML pages by combining static templates with dynamic data. Another key feature of Flask is its extensibility. It supports the use of extensions to add additional functionalities to the framework. There are numerous Flask extensions available for tasks such as database integration, user authentication, form validation, and more. This modular approach allows developers to tailor their application to their specific needs. Flask also includes a built-in development server, which makes it easy to test and debug applications locally. Additionally, it supports seamless integration with other popular

technologies such as SQLAlchemy for database management and Jinja2 for template rendering. Furthermore, Flask promotes the use of best practices in web development, such as RESTful principles and the use of HTTP methods for different operations.

iv) Python packages used (in the Web-app):

(1) The `os` module in Python provides functions for interacting with the operating system. It allows you to perform tasks such as accessing files and directories, manipulating paths, and executing system commands.

(2) The `uuid` module provides a unique identifier generation functionality. It allows you to generate universally unique identifiers (UUIDs) that can be used for various purposes, such as assigning unique names to files or tracking objects in a distributed system.

(3) Flask is a web framework for Python that provides tools and libraries for building web applications. It simplifies the process of handling HTTP requests and responses, routing URLs, managing sessions, and rendering templates.

(4) The `urllib` module is used for handling URLs in Python. It provides functions for making HTTP requests, parsing URLs, encoding and decoding data, and performing various URL-related operations.

(5) The `PIL` (Python Imaging Library) module is a powerful library for opening, manipulating, and saving many different image file formats. It provides functions for resizing, cropping, rotating, and applying various image processing operations.

(6) TensorFlow is a popular machine learning framework, and Keras is an open-source neural network library. The `keras` module in TensorFlow provides high-level APIs for building and training deep learning models.

(7) The `load_model` function from the `keras.models` module is used to load a

pre-trained deep learning model saved in the Keras format. It allows you to load and use a trained model for making predictions or further training.

(8) The `load_img` and `img_to_array` functions from the `keras.utils` module are used for loading images and converting them into numerical arrays, which can be used as input for deep learning models.

(9) `render_template` is used for rendering HTML templates. `request` is used for accessing HTTP request data. `send_file` is used for sending files in the HTTP response.

PROPOSED SYSTEM

a) Introduction:

Users of the proposed system can upload pictures of the skin irregularities they have via the web interface. The trained deep learning models are used to process the submitted photos in order to extract pertinent features and categorize the skin condition into particular disease groups. The users are then shown the classification findings, giving them their first impressions of the degree and type of their skin problems. Detailed below are the steps involved in creating the system.

i) Data Collection:

The dataset is a mix of the ISIC dataset and multiple skin cancer datasets found on Kaggle. Finally after narrowing down 9 visibly distinct diseases, the images were manually handpicked and verified, then added to folders named after the diseases it represents. An approximate of 100 images per disease were selected and then this folder was split into training and validation sets of 80:20 split.

ii) Model training:

The images which are categorized into different folders with respect to the disease they are associated to, are split into train and validation sets (80:20). The image preprocessing consists of dimensionality reduction, resizing, converting to RGB, creating a numpy array mapped to the disease name and finally normalizing the numpy array by dividing it by 255 (RGB ranges from 0 to 255). Next the models and their layers are defined (i.e; CNN, MobileNet, DenseNet), fit and their history of each epoch run is saved to a variable called “history”. Train accuracy, train loss, validation accuracy and validation loss are determined from history and plotted to visualize model performance. The models are then evaluated on the test data and a confusion matrix is plotted. These plots give an accurate interpretation on which model is to be picked for deployment and that particular model is saved.

iii) Web App:

The saved model is called into the python file for deployment called “app.py”. The backend connectivity is done using Flask and front end is developed using

HTML and CSS. “index.html” consists of all elements for the landing page which is a button for image upload. This function throws an error when any file other than jpg, jpeg, png, jfif is uploaded. “success.html” is the prediction page and shows a table with the models top 4 predictions, with probability percentage and the uploaded image. The CSS files are templates that determine fonts and styles for the HTML pages. “app.py” has the main function and Flask combines the backend and frontend. This file also has the prediction function and returns an array of the names of diseases and their probabilities.

b) Requirement Analysis:

i) Functional Requirements

(1) Product Perspective:

- For the system to make web-based data processing and transmission easier, Flask framework integration is required.
- The MobileNet or DenseNet deep learning models should be used by the system to classify skin diseases.
- For the system to produce an easy-to-use and aesthetically pleasing online interface, HTML and CSS should be used.

(2) Product Features:

- The system should allow users to upload images of skin anomalies for classification.
- The system should process the uploaded images using the deep learning models to extract features and classify the skin condition.
- The system should display the classification results to the users, indicating the probability and type of skin disease.

(3) User Characteristics:

- The system should be designed to cater to users with minimal technical knowledge or expertise in dermatology.
- Users should have access to a device with a web browser and an internet connection to utilize the system.

(4) Assumptions and Dependencies:

- The system assumes that users will provide high-quality images that

clearly depict the skin anomalies for accurate classification.

- The system relies on the availability of a comprehensive dataset of skin disease images for training the deep learning models.

(5) Domain Requirements:

- The system should accurately classify common skin diseases, such as dermatitis, eczema, pigmentation disorders, and acne.
- The system should provide recommendations for treating specific skin conditions based on the classification results.

(6) User Requirements:

- Users should be able to easily navigate the web interface and upload their skin anomaly images.
- Users should receive classification results promptly and in a clear and understandable format.
- Users should have confidence in the reliability and accuracy of the classification results.

ii) Non Functional Requirements

(1) Product Requirements:

(a) Efficiency (Time and Space): The system should be capable of processing and classifying skin disease images efficiently, providing quick results to users. It should optimize computational resources and minimize the time required for image analysis and classification.

(b) Reliability: The system should demonstrate high reliability by producing consistent and accurate classification results. It should minimize false positives and false negatives, ensuring trustworthy information for users.

(c) Portability: The system should be designed to be portable across different platforms and devices, allowing users to access it from various web browsers and operating systems.

(d) Usability: The system should have a user-friendly interface that is easy to navigate and understand. It should provide clear instructions for image uploading and display the classification results in a visually appealing and comprehensible manner.

(2) Organizational Requirements:

(a) Implementation Requirements (deployment):

(i) The system should be deployed on a reliable and secure web hosting platform to ensure uninterrupted availability.

(ii) It should be compatible with commonly used web servers such as Apache or Nginx.

(iii) The deployment process should include proper configuration of the web server, database, and other necessary components.

(iv) The system should be deployed in a scalable manner to accommodate increasing user demand without compromising performance.

(b) Engineering Standard Requirements:

(i) The system should adhere to established engineering standards and best practices during its development.

(ii) It should follow coding conventions, code documentation, and version control practices to maintain code quality and facilitate collaboration among developers.

(iii) The system should undergo rigorous testing and quality assurance processes to identify and address any defects or vulnerabilities.

(iv) It should comply with relevant software engineering standards, such as the use of modular and reusable code, to promote maintainability and future enhancements.

(v) The system should consider security standards and protocols to protect user data and prevent unauthorized access.

(3) Operational Requirements:

(a) Economic:

- The system is cost-effective in terms of development, maintenance, and operation.
- The deployment of the system does not impose significant financial burdens on the organization or end-users.

- It provides a cost-efficient alternative to traditional methods of diagnosing skin diseases, such as in-person consultations with dermatologists.

(b) Environmental:

- The system does not have any direct negative impact on the environment.
- It adheres to relevant environmental regulations and standards during its development and operation.

(c) Social:

- The system contributes to the well-being and convenience of individuals seeking skin disease diagnosis.
- It promotes accessibility and inclusivity, allowing users from diverse backgrounds to benefit from its services.

(d) Political:

- The system complies with all relevant political regulations and policies governing healthcare services, data privacy, and user consent.

(e) Ethical:

- The system prioritizes user privacy and confidentiality, ensuring that user data and images are protected.
- It should adheres ethical guidelines and principles related to the responsible use of AI and sensitive health information.

(f) Health and Safety:

- The system does not pose any health risks to users.
- It provides accurate and reliable classification results to prevent misdiagnosis or delayed treatment of skin diseases.

(g) Sustainability:

- The system is designed and implemented in a manner that supports long-term sustainability.

- It is scalable and adaptable to accommodate future updates, improvements, and increasing user demands.

(h) Legality:

- The system complies with applicable legal requirements, including but not limited to data protection, intellectual property rights, and healthcare regulations.

(i) Inspectability:

- The system has mechanisms in place to ensure transparency and allow for inspection by relevant authorities or auditors.
- It maintains proper documentation and logging of user interactions and system processes for auditing purposes.

iii) System Requirements

(1) H/W Requirements(details about Application Specific Hardware)

(a) The system is accessible through standard hardware devices such as laptops, desktop computers, smartphones, and tablets.

(b) The hardware should have sufficient processing power and memory capacity to support the execution of the deep learning models and web application.

(2) S/W Requirements(details about Application Specific Software)

(a) The system should be compatible with commonly used web browsers, such as Google Chrome, Mozilla Firefox, and Safari.

(b) The web application is developed using HTML, CSS, and Flask to ensure cross-platform compatibility.

(c) The system has incorporated the DenseNet deep learning model, requiring software frameworks such as TensorFlow or PyTorch for model implementation.

(d) The system may require additional software libraries and dependencies to enable image processing, data augmentation, and model evaluation.

MODEL COMPARISON

1. Training speeds:

a. DenseNet: Avg-850s/step

```
Epoch 1/5
18/18 [==============================] - 1073s 57s/step - loss: 0.8658 - accuracy: 0.1781 - val_loss: 0.7085 - val_accuracy: 0.
1929
Epoch 2/5
18/18 [==============================] - 942s 52s/step - loss: 0.6246 - accuracy: 0.3633 - val_loss: 0.6708 - val_accuracy: 0.2
000
Epoch 3/5
18/18 [==============================] - 851s 47s/step - loss: 0.4862 - accuracy: 0.4838 - val_loss: 0.6658 - val_accuracy: 0.2
071
Epoch 4/5
18/18 [==============================] - 896s 49s/step - loss: 0.3750 - accuracy: 0.6133 - val_loss: 0.6484 - val_accuracy: 0.2
357
Epoch 5/5
18/18 [==============================] - 911s 51s/step - loss: 0.2915 - accuracy: 0.7446 - val_loss: 0.6098 - val_accuracy: 0.1
929
```

b. MobileNet: Avg-150s/step

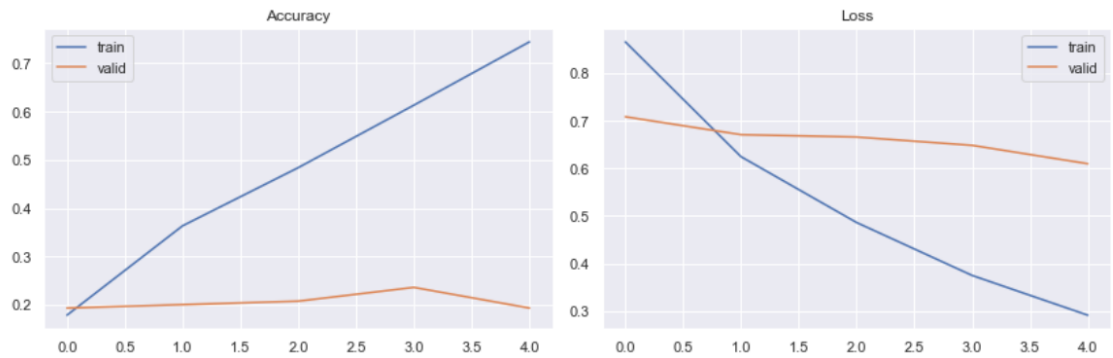
```
Epoch 1/5
16/16 [==============================] - 191s 11s/step - loss: 2.0261 - accuracy: 0.3080 - val_loss: 2.9103 - val_accuracy: 0.1
196
Epoch 2/5
16/16 [==============================] - 153s 10s/step - loss: 0.7432 - accuracy: 0.7967 - val_loss: 3.5524 - val_accuracy: 0.0
000e+00
Epoch 3/5
16/16 [==============================] - 157s 10s/step - loss: 0.3405 - accuracy: 0.9158 - val_loss: 3.3401 - val_accuracy: 0.0
000e+00
Epoch 4/5
16/16 [==============================] - 151s 9s/step - loss: 0.1954 - accuracy: 0.9630 - val_loss: 3.3552 - val_accuracy: 0.00
48
Epoch 5/5
16/16 [==============================] - 136s 9s/step - loss: 0.1034 - accuracy: 0.9815 - val_loss: 3.4380 - val_accuracy: 0.00
96
```

c. CNN: Avg-58s/step

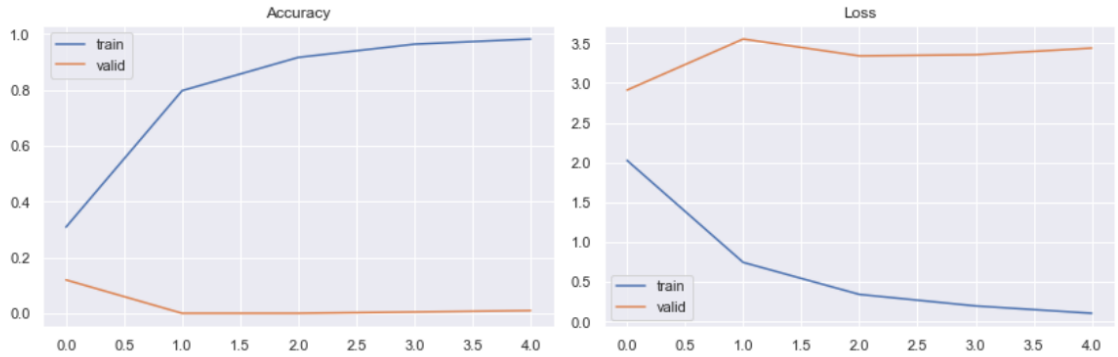
```
Epoch 1/5
17/17 [==============================] - 61s 3s/step - loss: 2.1070 - accuracy: 0.1360 - val_loss: 2.7620 - val_accuracy: 0.000
0e+00
Epoch 2/5
17/17 [==============================] - 56s 3s/step - loss: 2.0200 - accuracy: 0.1724 - val_loss: 2.7889 - val_accuracy: 0.000
0e+00
Epoch 3/5
17/17 [==============================] - 56s 3s/step - loss: 2.0314 - accuracy: 0.1858 - val_loss: 2.6704 - val_accuracy: 0.000
0e+00
Epoch 4/5
17/17 [==============================] - 62s 4s/step - loss: 1.9942 - accuracy: 0.2050 - val_loss: 2.8081 - val_accuracy: 0.000
0e+00
Epoch 5/5
17/17 [==============================] - 58s 3s/step - loss: 1.9680 - accuracy: 0.2011 - val_loss: 2.8076 - val_accuracy: 0.000
0e+00
```

2. Accuracy and loss curves:

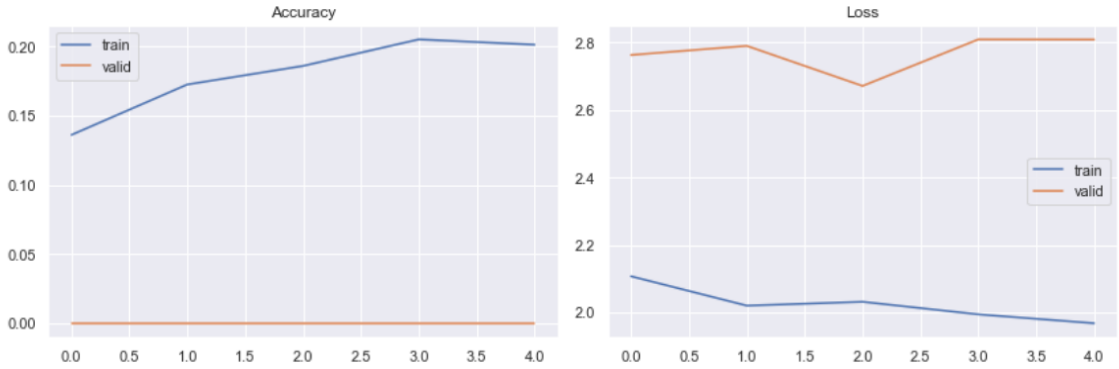
a. DenseNet



b. MobileNet

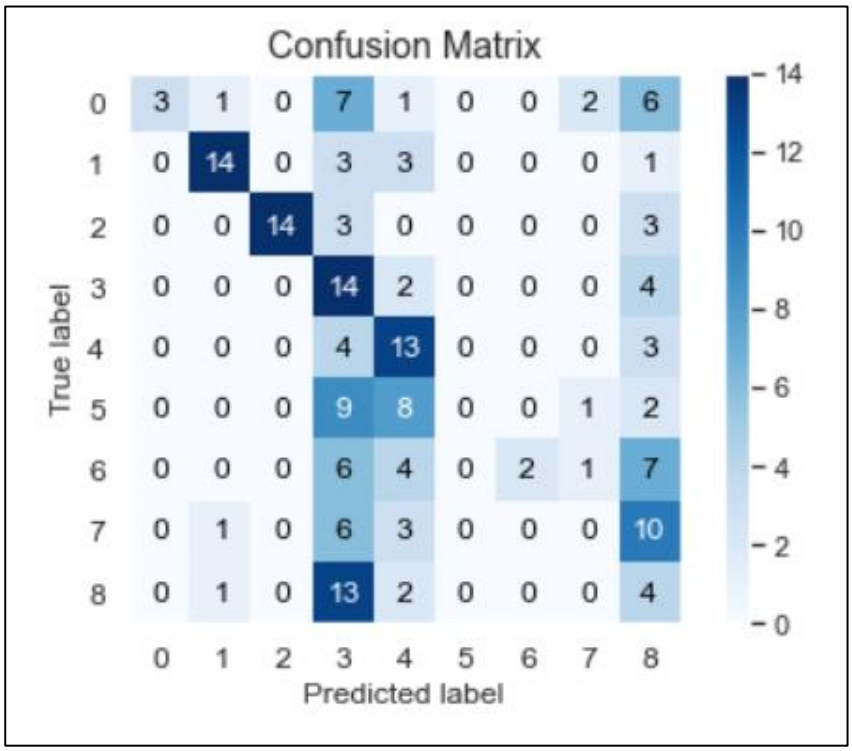


c. CNN

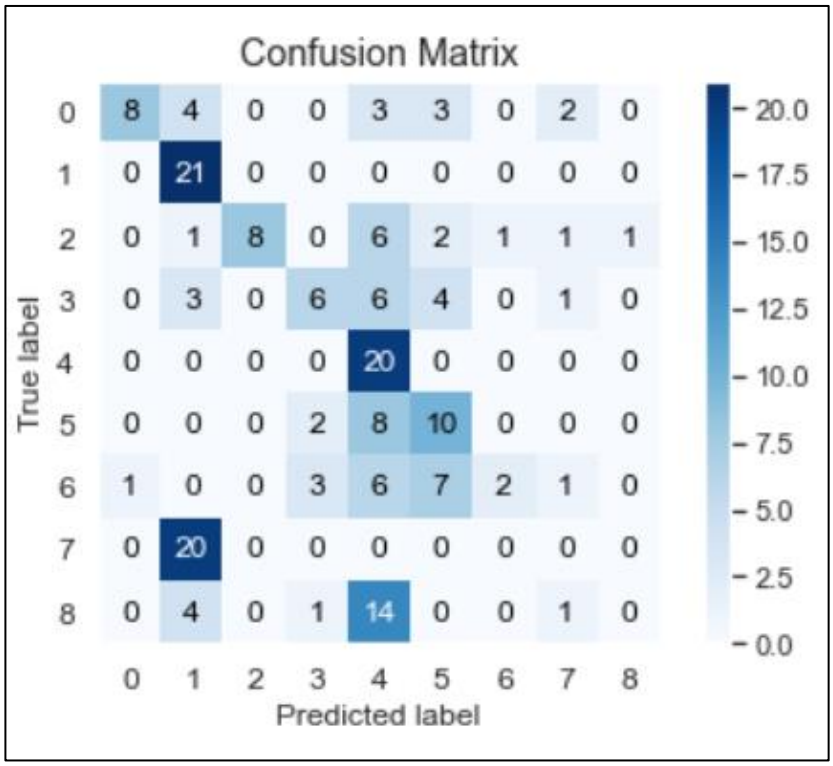


3. Confusion matrix (Testing with sample set):

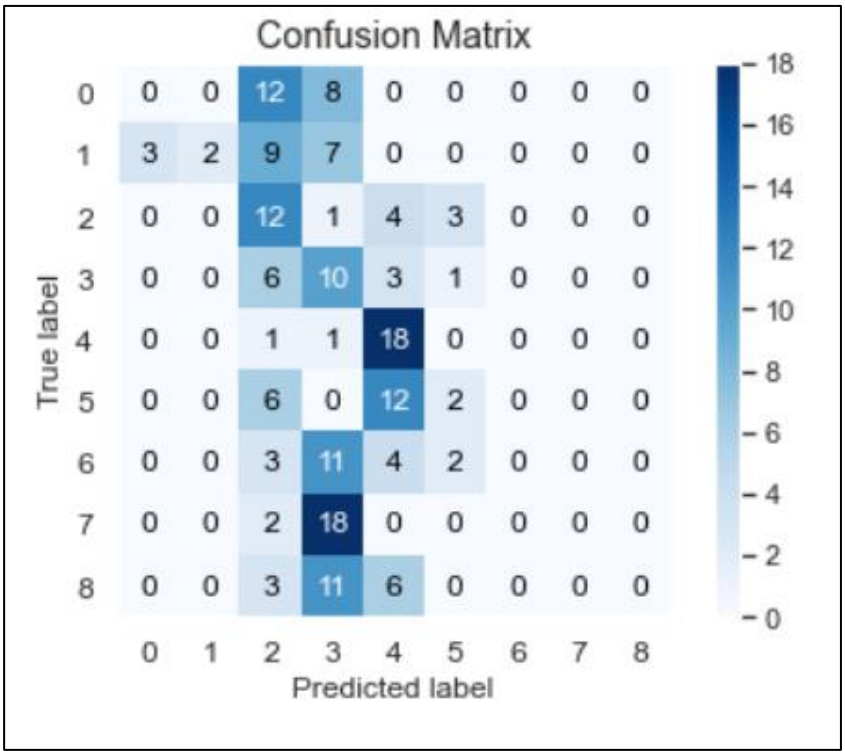
a. DenseNet



b. MobileNet



c. CNN



RESULTS

- Model Performance:

Models	Training Accuracy	Testing Accuracy	Training Loss	Testing Loss	Avg time to train/step (s/step)
CNN	**0.31**	**0.28**	**1.806**	**2.10**	**58**
MobileNet	**0.99**	**0.45**	**0.06**	**1.82**	**150**
DenseNet	**0.73**	**0.36**	**0.25**	**0.45**	**850**

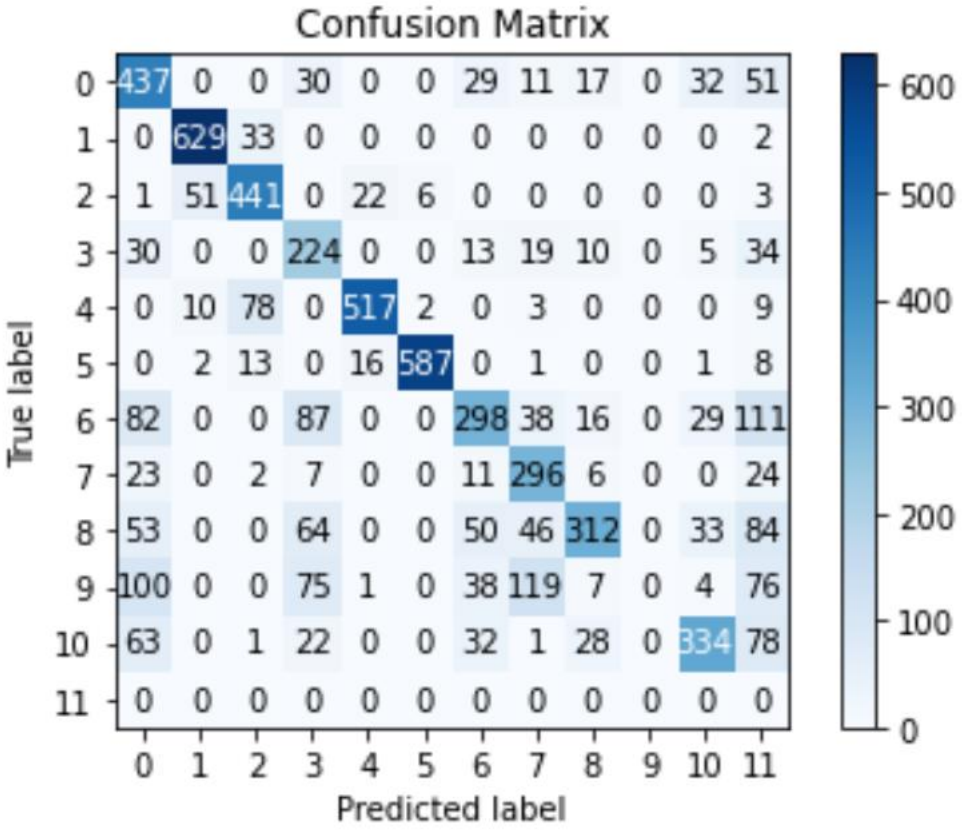


Fig.11: The confusion matrix showing predictions using the test images

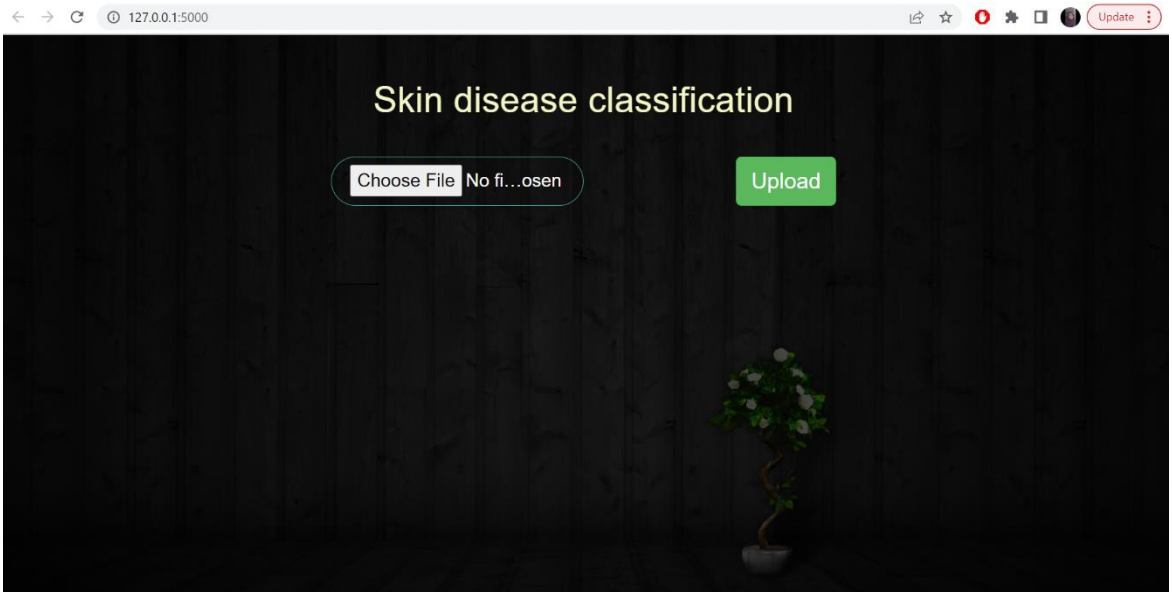
The DenseNet model predicts with an accuracy of 67.7%. The total images used for testing was 6021 out of which 4075 were predicted correctly. The model was expected to accurately classify the images into 11 diseases which were labelled 0-10. If the model identified the images as “NONE” of the trained labels, it would classify it into the “11”th

label. The model being trained only on images is unable to classify with more than 70% accuracy due to the very close visual similarity of skin anomalies. In order to improve the model, text prompts must be included.

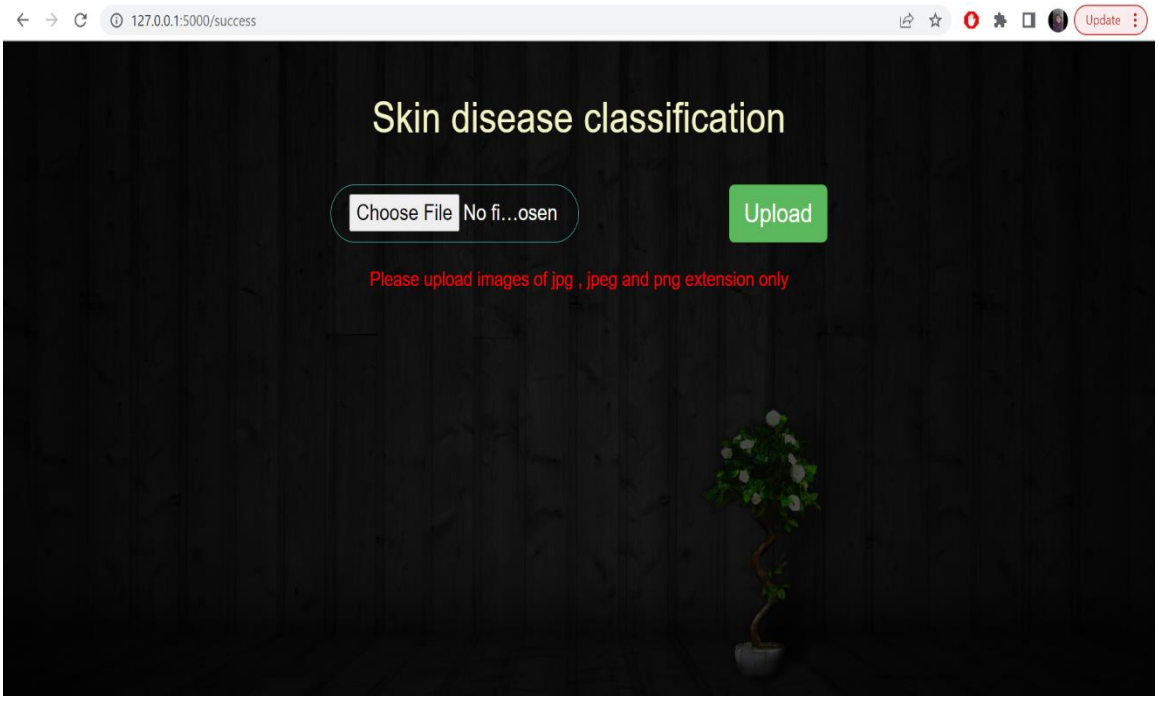
➢ Web page results (Test cases):

The web app is expected to show top 4 predictions with relation to the uploaded image.

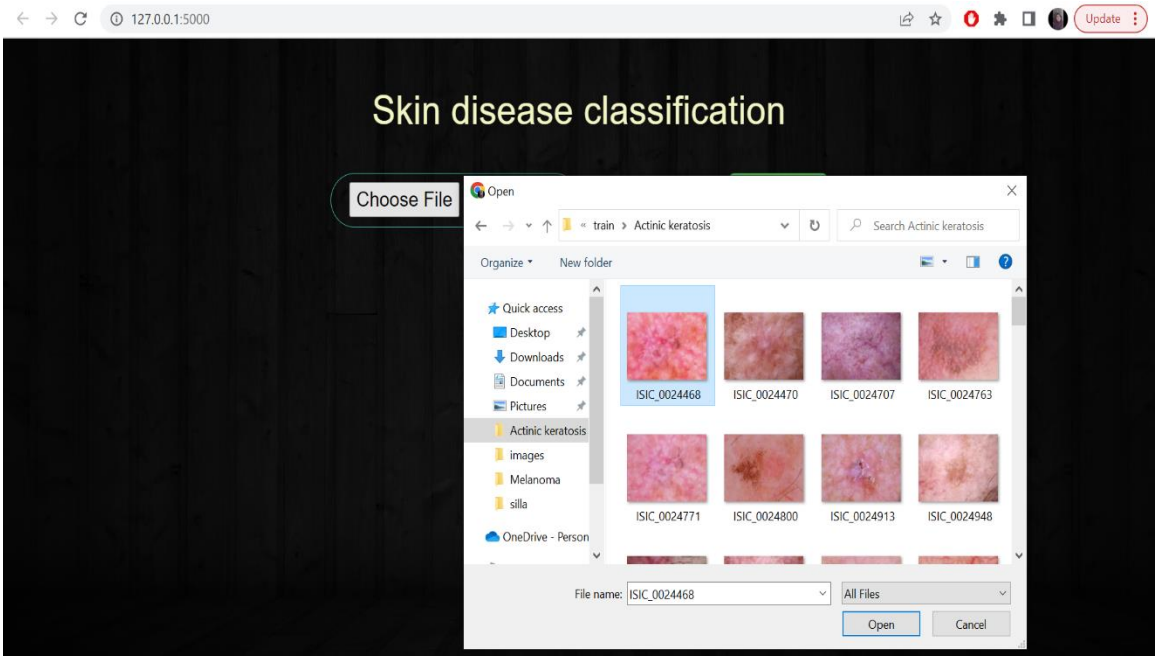
- Landing page:



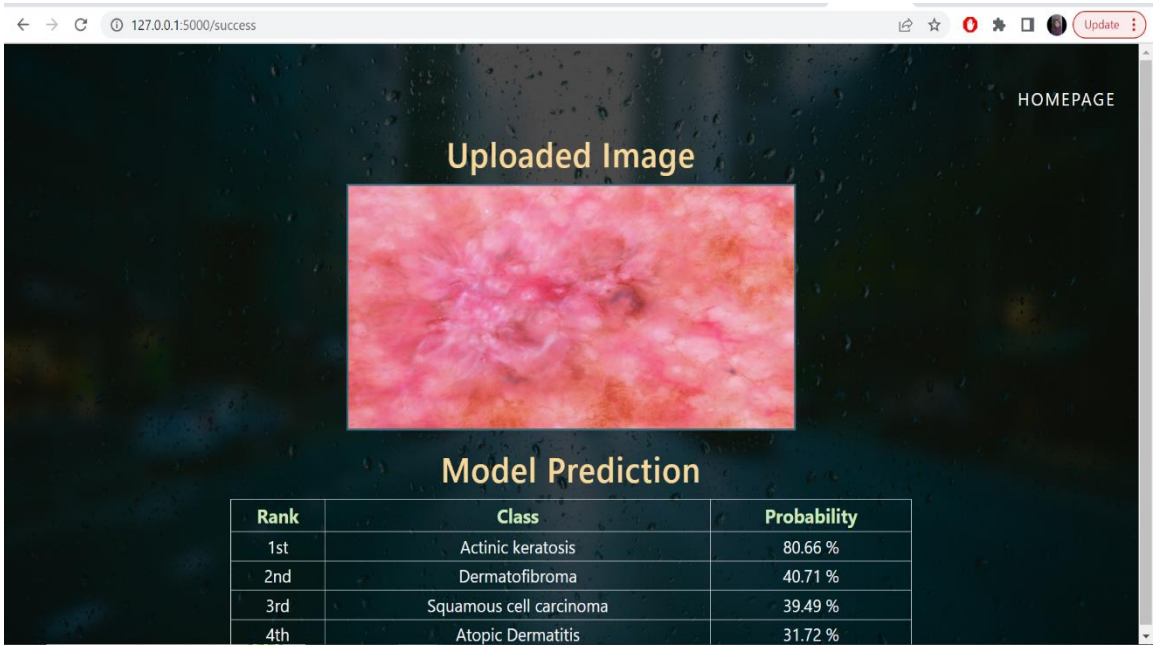
- In case the upload is not an image:



- Upload image:



- Prediction:



DRAWBACKS AND FUTURE ENHANCEMENTS

- Drawbacks:

  a. Heavy trained Model:

  One drawback of the project is the utilization of a heavy trained model, which can negatively impact the performance and efficiency of the system. The model's size can lead to longer processing times and increased memory requirements, making it less suitable for deployment on resource-constrained devices.

  b. Optimization Needed:

  As mentioned, the pre-trained model should be optimized to reduce its size and improve overall performance. Optimization techniques such as model compression, quantization, or pruning can be explored to minimize the memory footprint without significant loss in accuracy.

  c. Lack of Webpage Integration:

  The paper mentions the need for a webpage connected to the internet for the application, but the details and implementation of this component are not discussed. Future enhancements should include a detailed description of how the webpage is developed, including user interface design and interaction.

- Future Enhancements:

  a. Use a Light Pre-trained Model:

  To address the drawback of model size, future enhancements could involve exploring and implementing lighter pre-trained models that are specifically designed for deployment on mobile or web platforms. These models are optimized for efficiency and can provide comparable accuracy with reduced resource requirements.

  b. Increased Accuracy with Text Prompts: Incorporating text prompts along with image inputs can enhance the accuracy of the system. By leveraging natural language processing (NLP) techniques, the system can analyze textual descriptions

provided by the users to further refine the diagnosis and provide more accurate results.

c. Training Model with Text/NLP:

In addition to image classification, training the model with text or NLP techniques can expand the system's capabilities. This enhancement can enable the system to process textual information related to symptoms, medical history, or additional context, thereby providing more comprehensive and personalized diagnostic insights.

Overall, addressing the drawbacks and incorporating these future enhancements can lead to an optimized, more accurate, and versatile skin disease classification system, better suited for deployment on various devices and capable of providing more personalized and precise diagnostic recommendations.

REFERENCES

1. J. Velasco, “A smartphone-based skin disease classification using MobileNet CNN,” *International Journal of Advanced Trends in Computer Science and Engineering*, pp. 2632–2637, 2019.
2. M. Cullell-Dalmau, S. Noé, M. Otero-Viñas, I. Meić, and C. Manzo, “Convolutional neural network for skin lesion classification: Understanding the fundamentals through hands-on learning,” *Frontiers in Medicine*, vol. 8, 2021.
3. Iyer, Aparna, Shraddha Iyer, and Kshitija Hire. "A Skin Disease Detection System Using CNN Deep Learning Algorithm." In *Soft Computing and Signal Processing*, pp. 191-201. Springer, Singapore, 2021.
4. K. A. Muhaba, K. Dese, T. M. Aga, F. T. Zewdu, and G. L. Simegn, “Automatic skin disease diagnosis using deep learning from clinical image and patient information,” *Skin Health and Disease*, vol. 2, no. 1, 2021.
5. Rimi, T.A., Sultana, N. and Foysal, M.F.A., 2020, May. Derm-NN: skin diseases detection using convolutional neural network. In *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)* (pp. 1205-1209). IEEE.
6. Keras Team. “DenseNet.” *Keras: Deep Learning for Humans*, 20 Feb. 2023, https://keras.io/api/applications/densenet/.
7. Keras Team. *Keras Documentation: MobileNet, MobileNetV2, and MobileNetV3*. keras.io/api/applications/mobilenet.
8. “Coursera | Online Courses and Credentials From Top Educators. Join for Free | Coursera.” *Coursera*, www.coursera.org/learn/image-understanding-tensorflow-gcp/home/week/1.
9. “Coursera | Online Courses and Credentials From Top Educators. Join for Free | Coursera.” *Coursera*, www.coursera.org/learn/feature-engineering/home/week/1.
10. Python Simplified. “Simple Web App With Flask and Heroku - Python GUI for Beginners.” *YouTube*, 25 Sept. 2021, www.youtube.com/watch?v=6plVs_ytIH8.
11. “Skin Lesion Images for Melanoma Classification.” *Kaggle*, 28 May 2020, www.kaggle.com/datasets/andrewmvd/isic-2019.