The Egyptian E-Learning University

Faculty of Computers and Information Technology

Skin Diseases Detection

Graduation Project Documentation

Supervised by:

Dr. Yasser Abd El Hamid

Eng. Yasmine Mahmoud

Team members:

Raed Mohamed Badr	20-00898
Banoub Nagy Edwar	20-01025
Sherif Mohamed Waheed	20-01153
Mohamed Kamal Abdelaziz	20-01080
Rawda Abdelhadi Alkashef	20-00774
Rowan Abdelkader Mohamed	20-01472
Maryam Mohamed Abdelsamad	20-00120

Acknowledgement

We extend our heartfelt appreciation to Dr. Yasser Abd El Hamid. His guidance and mentorship have been instrumental in the successful completion of this graduation project. His expertise, constructive feedback, and dedication to fostering academic growth have greatly influenced the development of this work. We are grateful for the opportunities to learn and collaborate under his insightful supervision.

We would also like to express our gratitude to Eng. Yasmine Mahmoud, who served as a co-supervisor for this project. Eng. Yasmine's technical acumen and collaborative spirit have significantly enriched the project's content and outcomes. Her support and encouragement have been invaluable, contributing to the overall success of this endeavor.

We extend our sincere thanks to both Dr. Yasser Abd El Hamid and Eng. Yasmine Mahmoud for their crucial roles in guiding and shaping this graduation project.

Abstract

Skin diseases have a profound impact on individuals' well-being and health, being pervasive across all age groups. As some of the most prevalent health issues globally, they pose a substantial burden, encompassing psychological, social, and economic challenges for patients, their families, and society at large. This intricate concept of dermatological burden underscores the urgency for innovative approaches in disease identification and management.

By harnessing the power of deep learning, which excels at capturing complex patterns, this paper discusses two algorithms ResNet-152 and DenseNet-121. The chosen algorithm operates on ISIC 2019 dataset to detect a nuanced spectrum of skin diseases. The investigation aims to contribute to a more precise and efficient means of identification, potentially alleviating the multifaceted challenges posed by dermatological conditions. The study's scope encompasses the exploration of eight distinct skin conditions, reflecting a comprehensive analysis within the domain of dermatology.

Table of Contents

List of abbreviations

Abbreviations	Meaning
HAM 10000	Human Against Machine with 10000
ISIC 2019	International Skin Imaging Collaboration 2019
PAD-UFES-20	Patient and Dermoscopic Images - Universidade Federal do Espírito Santo - 2020
BCN 20000	Barcelona 20,000 dermoscopic images of skin lesions
Derm7pt	Dermatology 7-point checklist
ISIC 2017	International Skin Imaging Collaboration 2017
DermNet	Dermatology Network or Dermatology on the Net
PH2	Pedro Hispano Hospital
AI	Artificial Intelligence
DL	Deep Learning
CNN	Convolutional Neural Network
ResNet	Residual Neural Network
DenseNet	Densely Connected Convolutional Network
LSTM	Long Short-Term Memory
VGG	Visual Geometry Group
Cubic SVM	Cubic Support Vector Machine

List of figures

List of tables

Introduction

1.1. Problem overview

Skin diseases, prevalent across age groups, profoundly impact well-being globally, posing multifaceted challenges. Beyond physical discomfort, individuals navigating dermatological issues face mental health struggles, societal stigmatization, and economic burdens. Accessibility to timely and accurate medical assistance compounds the problem, with barriers like geographical limitations, lack of local expertise, and personal concerns hindering prompt examination. This delay not only extends suffering but also amplifies psychological distress as individuals seek reassurance about their health. Addressing these accessibility challenges is crucial for developing innovative solutions in skin disease identification. Additionally, skin conditions are estimated to affect 1.8 billion people worldwide, with skin infections being the commonest cause of disease in tropical and resource-poor settings [1].

1.2. Aim and goal

The objective is tackling the existing challenges in the detection of skin diseases by Leveraging the capabilities of deep learning, the project aims to provide patients with a user-friendly platform that enables them to independently initiate the identification process for various skin conditions. By integrating sophisticated algorithms and intuitive interfaces, our goal is to empower individuals to take a proactive approach towards their dermatological health. This solution seeks to bridge the gap in accessibility to timely medical assistance, particularly in areas with geographical limitations or a lack of local expertise. Through this project, we aspire to ease the way individuals engage with their skin health, fostering a sense of autonomy and timely intervention.

1.3. Documentation overview

In our pursuit to contribute to the nexus of dermatology and deep learning, this documentation embarks on a focused exploration, undertaking a comparative analysis of key skin disease datasets and deep learning algorithms. By delving into the intricacies of these datasets, we aim to uncover not only their structural nuances but also the unique contributions they make to advancing our understanding of dermatological conditions.

The datasets under scrutiny HAM10000, ISIC 2019, PAD-UFES-20, BCN20000, Derm7pt, ISIC 2017, DermNet, PH2, and Dermofit - each offer a distinct lens through which we can view skin diseases. We traverse through the year of publication and authors, acknowledging the diverse origins of these datasets. Their structures, from the organization of images to metadata management, are meticulously outlined, providing a foundation for understanding their utility.

Within this visual tapestry, we explore the characteristics of the images, ranging from resolution to capturing techniques. The datasets predominantly feature colorful JPEG images captured through various methods, adding a layer of complexity to the classification task. Encompassing a spectrum of skin diseases, these datasets contribute to a simple understanding of dermatological conditions, embracing both non-cancerous which is our goal and cancerous afflictions.

Yet, our exploration doesn't shy away from challenges. We critically examine issues within each dataset, ensuring transparency about their limitations. Solutions proposed aim to enhance the quality and reliability of the datasets, providing a roadmap for optimizing their utility in decision-making processes and applications reliant on accurate data.

At the end of the section on datasets, we put together a table that contains a comparison between all the types of data that we discussed in this section, in addition to the best types of datasets that are likely to be used in our project.

Transitioning to the realm of deep learning models, our focus extends to ResNet-152, ResNet101, Inception-v3, DenseNet-121, DenseNet-201, MobileNet v1, MobileNet v2 & LSTM, VGGNet, and VGG16 - algorithms that have demonstrated efficacy in skin disease classification. Through a structured lens, we explore their origins, dataset usage, preprocessing techniques, structures, layers, model-building processes, and result metrics, ensuring a comprehensive understanding, and we offer observations about the pros and cons of employing these skin disease classification algorithms.

At the end of the section on algorithms, we put together a table that contains a comparison between all the research papers that we discussed in this section, in addition to the best types of algorithms that are likely to be used in our project.

Datasets

Skin disease recognition and classification using deep learning techniques rely on the availability and quality of skin disease datasets, which are collections of skin images with labels or annotations. However, the existing datasets for skin disease research are often incomplete, inconsistent, or imbalanced. In this documentation, we provide a comprehensive overview of the current state-of-the-art skin disease datasets and at the end of the datasets chapter, depended on our comparison we will determine what is best collection of datasets to be used in our project.

2.1. Introduction

we rely on nine points to detail each dataset:

Overview: Offering a comprehensive look into skin disease datasets. Each dataset's purpose, structure, and characteristics are succinctly outlined.

Goal: The primary objective behind assembling these datasets

Year of publication/Author: The datasets discussed here span a range of publication years and authors, showcasing contributions from multiple researchers and institutions.

Dataset structure: Each dataset exhibits its unique structure, ranging from the arrangement of images into distinct train, test, and validation sets to the organization of metadata to track the images.

Image characteristics: Across these datasets, the images vary in terms of resolution, format, and capturing techniques. They predominantly comprise colorful JPEG images captured through dermoscopic examinations, skin surface microscopy, and smartphone devices.

Diseases characteristics: The datasets cover a spectrum of skin diseases, including both cancerous and non-cancerous conditions.

Problems: we aim to elucidate and articulate the challenges and issues inherent in each dataset and shortcomings, anomalies, or discrepancies within the data.

Consequences: offers insights into how each issue may impact the quality, reliability, interpretability, decision-making processes, or any applications reliant on accurate and consistent data.

Solutions: proposing strategies and practical and actionable steps that can be taken to enhance the quality and reliability of the datasets to make informed decisions.

2.2. HAM 10000

The HAM10000 dataset [2] is a comprehensive collection of dermatoscopic images aimed at facilitating research in automated diagnosis of pigmented skin lesions. It contains 10,015 dermatoscopic images representing various diagnostic categories related to pigmented lesions.

This dataset aims to facilitate the training of neural networks for automated diagnosis by employing cleaning methods, ensuring comprehensive representation of diagnostic categories, and validating cases through pathology, expert consensus, or other confirmatory techniques.

The dataset was published on 2018-06-04 by Tschandl, Philipp (Medical University of Vienna).

Dataset does not specify divisions into distinct train, test, and valid sets. Instead, it comprises a collection of 10,015 dermatoscopic images.
The images are stored in two separate folders:
HAM10000_images_part1.zip with 5000 images and HAM10000_images_part2.zip with 5015 Images.
HAM10000_metadata is a csv file which contains lesion_id column to track lesions images and its disease type [3].

The images are colorful, resolution is 600*450, and format is JPEG. collected using technique involving skin surface microscopy from Medical University of Vienna.

It includes seven different skin diseases, including five non-cancerous diseases and two cancerous diseases:
Non-Cancerous:
Actinic Keratoses - 327 images,
Melanocytic Nevi - 6705 images,
Vascular Lesions - 142 images.
Benign Keratosis - 1099 images,
Dermatofibroma - 115 images,
Cancerous:
Melanoma - 1113 images,
Basal Cell Carcinoma - 514 images

There are two problems in HAM1000:
a) Dataset does not specify divisions into distinct train, test, and valid sets.
b) The number of samples in each class varies widely.

The consequences of the two problems:
a) Overfitting which leads to poor generalization.
b) biased predictions where the model tends to predict the majority class, ignoring the minority classes.

So, we can solve these problems using some strategies and techniques like
a) Split data into 80% train / 10% test / 10% validate.
b) Data Augmentation to Enhance dataset diversity.

2.3. ISIC 2019

The ISIC 2019 dataset [4] comprises 25,331 dermatoscopic images spanning eight skin lesion categories. It includes varied sizes of 600 × 450 and 1024 × 1024 but exhibits an unbalanced distribution across labels. Evaluated via balanced multi-class accuracy (recall), it aids in skin cancer detection algorithm development and dermatological research.

The International Skin Imaging Collaboration developed the ISIC archive for engaging clinicians and computer vision researchers. The ISIC 2019 dataset includes dermoscopic image databases for common skin lesions.

The dataset was published on December 17, 2018, by (Tschandl, Codella, Gutman, Celebi, Helba, Marchetti, Dusza, Kalloo, Liopyris, Mishra, Kittler, Halpern, Combalia, Codella, Rotemberg, Helba, Vilaplana, Reiter, Carrera, Barreiro, Halpern, Puig, Malvehy).

The dataset comprises 25,352 images divided into Test, Train, and Validate subsets. The Test subset includes 1,930 images (7.6%), Train consists of 21,512 images (84.8%), and validate contains 1,910 images (7.5%).

The images are colorful, contain images of size 600 × 450 and 1024 × 1024, and format is JPG. Gathered from multiple global sources, such as healthcare providers, clinics, and research centers specializing in dermatology.

It includes eight different skin diseases, including five non-cancerous diseases and three cancerous diseases:
Non-Cancerous:
Melanocytic nevus: Test - 965, Train - 11000, Valid - 931
Actinic keratosis: Test - 75, Train - 716, Valid - 76
Benign keratosis: Test - 203, Train - 2215, Valid - 206
Dermatofibroma: Test - 11, Train - 206, Valid - 22
Vascular lesion: Test - 24, Train - 202, Valid – 27
Cancerous:
Melanoma: Test - 360, Train - 3812, Valid - 350
Basal cell carcinoma: Test - 250, Train - 2820, Valid - 253
Squamous cell carcinoma: Test - 42, Train - 542, Valid – 45

There are two problems in ISIC 2019:
a) The number of samples in each class varies widely, especially in the training and validation folder.
b) There is hair on the skin that covers parts of the disease.

The consequences of the problems:

a) bias during the training process towards classes with a greater number of samples and biased predictions, where the model tends to predict the majority class, ignoring the minority classes.
b) Overfitting or Poor Generalization where the model may be overfit on the classes with more samples and fail to generalize and learn distinguishing features samples.
c) Hair can introduce noise and interference in the images and cover key features of skin diseases and making it harder for the model to distinguish patterns or textures for classification.

So, we can solve these problems using some strategies and techniques like

a) Apply data augmentation techniques to artificially increase the number of samples in the classes with fewer images. Techniques like rotation, flipping, scaling, and adding noise can help diversify the dataset.
b) Marge with other dataset to increase samples of classes with fewer samples and balance the number of images for all disease.
c) Reducing the number of images for disease with many images to balance them with diseases with a small number of images.
d) Apply various denoising techniques like median filtering, non-local means denoising, wavelet transforms, thresholding to reduce the impact of noise caused by hair.

2.4. PAD-UFES-20

The PAD-UFES-20 dataset [5] is a skin lesion benchmark composed of clinical images collected from smartphone devices and a set of patient clinical data containing up to 21 features. It was developed to tackle skin lesion analysis.

The goal of the PAD-UFES-20 dataset is to support future research and the development of new tools to assist clinicians in detecting skin cancer.

Dataset was authored by a team of researchers including Andre G. C. Pacheco, Gustavo R. Lima, Amanda S. Salomão, Breno A. Krohling, Igor P,

and many more. The dataset was collected from 2018 to 2019 and was released in October 2020

The dataset consists of 1373 patients, 1641 skin lesions, and 2298 images for six different diagnostics: three skin diseases and three skin cancers. In addition, each image has up to 21 patient clinical features including the patient's age, skin lesion location, Fitzpatrick skin type, and skin lesion diameter also four identifying features (patient ID, lesion ID, Image ID, and if the sample is biopsy-proven) in CSV file.
The images are stored in two separate folders:
imgs_part_1.zip 1.16 GB
imgs_part_2.zip 1.05 GB
imgs_part_3.zip 1.14 GB

All images are raw, no image processing to enhance visualization, and format is PNG. Since the images in this dataset are collected using different smartphone devices, they present different resolutions, sizes, and lighting conditions. Thus, it aims to simulate the real world.

Among six skin diseases in the dataset, there are three cancerous conditions and three non-cancerous conditions represented.

Non-Cancerous:

- Actinic Keratosis - 730 images.
- Melanocytic Nevus of Skin - 244 images.
- Seborrheic Keratosis - 235 images.

Cancerous:

- Squamous Cell Carcinoma - 192 images.
- Basal Cell Carcinoma of skin - 845 images.
- Malignant Melanoma - 52 images.

There are problems in PAD-UFES-20 dataset:

a) Variability in image Properties because images are collected from different smartphone devices may vary in resolution, size, and lighting conditions.

b) Images in folders are not previously categorized but it's needed to be put in its categories using meta data which require additional code for this process.

The consequence of the problems is that model might struggle to generalize to unseen or differently captured data, affecting their real-world applicability.

These problems can be resolved using some strategies like using standardization techniques, image preprocessing (resizing, normalization) can help mitigate the variability across images and improve model robustness.

2.5. BCN 20000

The BCN 20000 dataset [6] is a collection of dermoscopic images of skin lesions that can be used for skin cancer diagnosis using artificial intelligence. It was created by researchers from the Hospital Clinic in Barcelona and contains images of lesions from different body locations, sizes, and colors. The dataset was released in 2019 and is one of the datasets used in the ISIC Challenge.

It aims to study the problem of unconstrained classification of dermoscopic images of skin cancer, including lesions found in hard-to-diagnose locations (nails and mucosa), large lesions which do not fit in the aperture of the dermoscopy device, and hypo-pigmented lesions.

The BCN 20000 dataset, published in 2019 by Combalia et al., was curated by a multidisciplinary team including researchers from Hospital Clínic Barcelona, IBM Research AI, Memorial Kettering Cancer Center, Kitware, and Universitat.

the BCN20000 dataset, composed of 19424 dermoscopic images of skin lesions captured from 2010 to 2016.

The BCN 20000 dataset comprises JPEG images with resolutions spanning from 600x450 to 4288x2848 pixels, offering a diverse set of colorful images.

Dermoscopic images were captured using dermoscopic attachments on three high-resolution cameras and stored on a hospital server.

The BCN20000 dataset eight different skin diseases, including five non-cancerous diseases and three cancerous diseases:

Non-Cancerous:

Nevus - 9,467
Seborrheic Keratosis - 1,099 images
Actinic Keratosis - 1,082 images
Dermatofibroma - 1,030 images
Vascular Lesions – 587 images

Cancerous:

Melanoma - 1,113 images
Basal Cell Carcinoma - 5,016 images
Squamous Cell Carcinoma - 1,030 images

There are two problems in BCN20000:

a) The number of samples in each class is imbalanced.
b) There is hair on the skin that covers parts of the disease.

The consequences of the problems:

a) bias during the training process towards classes with a greater number of samples and biased predictions, where the model tends to predict the majority class, ignoring the minority classes.
b) Overfitting or Poor Generalization where the model may be overfit on the classes with more samples and fail to generalize and learn distinguishing features samples.
c) Hair can introduce noise and interference in the images and cover key features of skin diseases and making it harder for the model to distinguish patterns or textures for classification.

So, we can solve these problems using some strategies and techniques like

a) Apply data augmentation techniques to artificially increase the number of samples in the classes with fewer images. Techniques like rotation, flipping, scaling, and adding noise can help diversify the dataset.
b) Marge with other dataset to increase samples of classes with fewer samples and balance the number of images for all disease.
c) Reducing the number of images for disease with many images to balance them with diseases with a small number of images.
d) Apply various denoising techniques like median filtering, non-local means denoising, wavelet transforms, thresholding to reduce the impact of noise caused by hair.

2.6. Derm7pt

The Derm7pt dataset [7] by Kawahara includes 1011 resized colorful skin lesion images. It assesses seven-point checklist-based prediction, offering 15 disease categories with varying image counts. Train-validation-test splits allocate 413, 203, and 395 cases respectively, enabling comparative analysis of computerized and expert diagnoses in dermatology.

Created to evaluate computerized image-based prediction of the seven-point checklist and compare it with human experts. The authors also proposed a multitask multimodal neural network that can simultaneously predict the diagnosis and the seven criteria from clinical and dermoscopy images.

The dataset was published in March 2019 by Jeremy Kawahara, Sara Daneshvar, Giuseppe Argenziano, and Ghassan Hamarneh.

The dataset consists of 1011 clinical and dermoscopic color images of skin lesions. use 413 cases to train the model, 203 cases to validate design decisions and 395 cases to test and report results.

Images are colorful, Images are in JPEG format. All images were resized to 512×512×3.

The Derm7pt dataset has 15 different skin diseases, including 13 non-cancerous diseases and 2 cancerous diseases:

Non-Cancerous:

Blue nevus - 28 images
Clark nevus - 399 images
Combined nevus - 13 images
Congenital nevus - 17 images
Dermal nevus - 33 images
Recurrent nevus - 6 images
Reed or spitz nevus - 79 images
Dermatofibroma - 20 images
Lentigo - 24 images
Melanosis - 16 images
Miscellaneous - 8 images
Vascular lesion -29 images
Seborrheic keratosis - 45 images

Cancerous:

Basal cell carcinoma - 42 images
Melanoma- 252 images

There are two problems in derm7pt:

a) Imbalanced Dataset for example there are 399 images for Clark nevus but only 6 for Recurrent nevus.
b) Limited Data With only 1011 images, the dataset might be too small for training.

The consequences of the problems:

a) Imbalances in the dataset can lead to biased predictions, model perform well on the majority class.
b) Poor performance of a weak model.

We can solve these problems with:

a) Apply data augmentation techniques to artificially increase the number of samples in the classes with fewer images. Techniques like rotation, flipping, scaling, and adding noise can help diversify the dataset.

b) Marge with other dataset to increase samples of classes with fewer samples and balance the number of images for all disease.

2.7. ISIC 2017

The ISIC 2017 dataset [8] is used in challenge consisted of 3 tasks: lesion segmentation, dermoscopic feature detection, and disease classification. For each, data consisted of images and corresponding ground truth annotations, split into training 2000 image, validation 150image, and holdout test 600 image datasets. Predictions could be submitted on validation and test datasets. The data was collected from different sources.

The goal of the challenge is to help participants develop image analysis tools to enable the automated diagnosis of melanoma from dermoscopic images. Image analysis of skin lesions is composed of 3 parts, Part 1: Lesion segmentation Part 2: Detection and localization of Visual dermoscopic Features, Part 3: Disease classification.

The dataset was published in 8 Jan 2018 by (Noel C. F. Codella, David Gutman, M. Emre Celebi, Brian Helba, Michael A. Marchetti, Stephen W. Dusza, Aadi Kalloo, Konstantinos Liopyris, Nabin Mishra, Harald Kittler, Allan Halpern)

Use training data 2000 images for participants to engage in all 3 components of lesion image analysis. validation dataset 150 images and blind held-out test dataset 600 images.

Data was collected from various sources and different imaging devices. Images are colorful, images in JPEG, PNG and Json format with different resolutions.

The ISCI 2017 dataset has three skin diseases, including two non-cancerous diseases and one cancerous disease:

Non-Cancerous

seborrheic keratosis :254 train, 42 valid, 90 test.

benign nevi:1372 train, 78 valid, 393 test.

Cancerous:
melanoma: 374 train, 30 valid, 117 test.

The Problem of ISIC 2017 is the dataset is imbalanced, as there are more benign nevi than melanoma or seborrheic keratosis images.

Consequence of problem is Poor performance of the model (poor generalization)

Solution of problems is that Employ data augmentation techniques to artificially augment the volume of samples within classes featuring fewer images. Utilizing methods such as rotation, flipping, scaling, and introducing noise can effectively enhance the diversity of the dataset.

2.8. DermNet

The DermNet dataset [9] is the world's leading free dermatology resource, dedicated to providing comprehensive information on skin conditions. Designed for healthcare professionals, it offers evidence-based content, including a library of over 20,000 dermatology images.

DermNet provides authoritative skin information to healthcare professionals and patients, promotes awareness, supports dermatology research and education, and collaborates with organizations for similar objectives.

DermNet was founded in 1996 by three dermatologists: Dr Amanda Oakley, Dr Mark Duffill, and Dr Marius Rademaker. The current Editor-in-Chief is Dr Ian Coulson, who took over from Dr Delwyn Dyall-Smith in 2021.

The dataset contains 19,500 images in 23 folders but have approximately 40 skin disease categories, organized into train 15,500 images, and test 4,000 images.

The images are stored in JPEG format and have a resolution of 600 x 450 pixels and colored.

There are approximately 40 skin diseases with 2 cancerous conditions and 38 non-cancerous and we will make lines under cancer conditions.

Acne and Rosacea: 1152,
Actinic Keratosis and Basal Cell: 1437
Atopic Dermatitis: 612,
Bullous Disease: 561,
Cellulitis Impetigo and bact: 361
Eczema: 2544,
Exanthems and Drug Eruptions: 505,
Herpes: 507
Hair Loss Photos Alopecia: 299,
Light and Disorders of Pigmentation: 711
Lupus and Connective Tissue: 525,
Melanoma and Nevi and Mole: 579
Nail Fungus: 1301,
Poison Ivy/Contact Dermatitis: 325
Psoriasis Lichen Planus: 1757,
Scabies Lyme: 539,
Systemic: 758
Seborrheic Keratoses: 1714,
Tinea Ringworm: 1625
Urticaria Hives: 265,
Vascular Tumors: 603,
Vasculitis: 521
Warts Molluscum: 2358

The problems of dermnet dataset are:

a) the dataset is that Within each of the 23 folders, various types and forms of skin diseases may coexist, and there is randomness in the combination of images and diseases in all folders.
b) images for each disease does not exceed 200 images.

The consequences of this problem are:

a) The diverse range of skin diseases in a single folder increases complexity of training deep learning models. Manual handling of dataset for labeling and annotation becomes more challenging due to multiple diseases in each folder and time-consuming and error-prone to assign accurate labels to each image.

b) Poor and weak the model predictions

Solutions of these problems are:

a) Clearly organize the data within each folder to separate different types of skin diseases. This could involve creating subfolders for each disease or adopting a structured naming convention.
b) This dataset can be used as a secondary source to increase the main dataset.

2.9. PH^2

The PH2 dataset [10] acquired at Hospital Pedro Hispano in Portugal, contains 200 melanocytic lesions, categorized into 80 common nevi, 80 atypical nevi, and 40 melanomas. This valuable dataset aids in dermatology research and diagnosis using dermoscopic images.

The PH2 database is intended to be freely accessible for research and benchmarking, addressing the current challenge of inconsistent evaluation sets in the field of dermoscopic image analysis.

The PH2 dermoscopic image database was published on July 8, 2013, by Dra. Joana Rocha and Dra. Marta Pereira.

PH Dataset images folder: Inside this folder there is a dedicated folder for every image of the database, which contains the original dermoscopic image, the binary mask of the segmented lesion as well as the binary masks of the color classes presented in the skin lesion [11].

The PH2 dataset features 768 x 560 pixels images of skin lesions with various characteristics and bmp format, including colors, pigment network, dots/globules, streaks, regression areas, and blue-whitish veil.

It includes three different skin diseases, including two non-cancerous diseases and one cancerous disease:

Non-Cancerous:

80 images of common nevi.
80 images of atypical nevi.

Cancerous:
40 images of melanomas.

The problem with this dataset is that small datasets not large enough for deep learning.

Consequence of this problem is that Deep learning require large amounts of data to learn effectively.

we can solve these problems using some strategies and techniques like
a) Split data into 80% train / 10% test / 10% validate.
b) Data Augmentation to Enhance dataset diversity.

2.10. Dermofit

The Dermofit dataset [12] develops an interactive skin cancer image database indexing tool, where users compare 'live' skin lesions/spots to images selected from a database. This could potentially improve the diagnostic accuracy of dermatologists and save lives.

The project goal is to develop an interactive skin cancer image database indexing tool, where users compare live skin lesions/spots to images selected from a database. Based on user feedback on the goodness of the selections, the database will be searched for better matches.

The project starts on August 1, 2008, and continues until July 31, 2011 by Rees, Aldridge, Fisher, and Ballerini.

This data will require purchase. It is not divided into train, test, and valid folders.

There are 1300 images are normal RGB captured with a quality SLR camera under controlled (ring flash) indoor lighting.

It includes ten different skin diseases, including seven non-cancerous diseases and three cancerous diseases:
Non-Cancerous:
Actinic Keratosis – 45 images

Melanocytic Nevus (Mole) - 331 images
Seborrhoeic Keratosis: 257 images
Intraepithelial carcinoma - 78 images
Pyogenic Granuloma - 24 images
Haemangioma - 96 images
Dermatofibroma - 65 images

Cancerous:

Melanoma - 76 images
Basal Cell Carcinoma - 239 images
Squamous Cell Carcinoma - 88 images

The problems of dermofit are:

a) The Dermofit Image Library comprises only 1,300 images across ten different lesion types. Some classes, such as Pyogenic Granuloma, have a small number of images (24)
b) Access to the Dermofit Image Library involves a one-off £75 licence fee. Submission must occur via an institutional email address, and approval by an authorized representative of the institution is required.

consequences of the Dermofit problems are:

a) overfitting and reduced generalization.
b) creating potential access barriers for individual researchers and limiting dataset accessibility.

Solutions of problems are:

a) Increase the effective size of the dataset through augmentation techniques, such as rotation, flipping, and scaling, to provide additional variations for model training.
b) Institutions could consider waiving fees for researchers or providing grants to cover the costs, ensuring broader access to the Dermofit Image Library.

Table 2.1 Datasets Comparison

Name	Year	# of Diseases	# of Non-Cancerous	# of Cancerous	# of Images	Splitting	# of Images for each disease in Test	# of images for each disease in Train	# of Images for each disease in Validate	Goal
HAM10000	21-Dec-18	7	5	2	10015	N/A	Actinic Keratoses 327 Basal Cell Carcinoma 514 Benign Keratosis-Like Lesions 1099 Dermatofibroma 115 Melanoma 1113 Melanocytic Nevi 6705 Vascular Lesions 142	N/A	N/A	Aims to facilitate the training of neural networks for automated diagnosis.
ISIC 2019	11-Dec-19	8	5	3	25352	Test 1930 Train 21512 Val 1910	Melanoma 360 , Melanocytic 965, Basal cell 250, Actinic keratosis 75, Benign keratosis 203, Dermatofibroma 11, Vascular lesion 24, Squamous cell 42.	Melanoma 3812, Melanocytic 11000, Basal cell 2820, Actinic Keratosis 716, Benign Keratosis 2215, Dermatofibroma 206, Vascular lesion 202, Squamous cell 542.	Melanoma 350, Melanocytic 931, Basal cell 253, Actinic Keratosis 76, Benign Keratosis 206, Dermatofibroma 22, Vascular lesion 27, Squamous cell 45.	ISIC archive for engaging clinicians and computer vision researchers
PAD-UFES-20	18-Jul-18	6	3	3	2298	N/A	Actinic Keratosis 730 Melanocytic Nevus of Skin 244 Seborrheic Keratosis 235 Squamous Cell Carcinoma 192 Basal Cell Carcinoma of skin 845 Malignant Melanoma 52	N/A	N/A	developingof new tools to assist clinicians to detect skin cancer and kesions.

Table 2.2 Datasets Comparison (continued)

BCN 20000	6-Aug-19	8	5	3	19424	N/A	Nevus 9467 Seborrheic Keratosis 1099 Actinic Keratosis 1082 Dermatofibroma 1030 Vascular Lesions 587 Melanoma 1113 Basal Cell Carcinoma 5016 Squamous Cell Carcinoma 1030	N/A	N/A	classification of dermoscopic images of skin lesions and cancer
Derm7pt	1-Mar-19	15	13	2	1011	Train 413 Test 395 Val 203	Basal cell carcinoma 42 Blue nevus 28 Clark nevus 399 Combined nevus 13 Congenital nevus 17 Dermal nevus 33 Recurrent nevus 6 Reed or spitz nevus 79 Melanoma 252 Dermatofibroma 20 Lentigo 24 Melanosis 16 Miscellaneous 8 Vascular lesion 29 Seborrheic keratosis 45	N/A	N/A	evaluate computerized image-based prediction of the seven-point checklist and compare it with human experts
ISIC 2017	8-Jan-18	3	2	1	2750	Train 2000 Test 600 Val 150	Melanoma 117 Seborrheic keratosis 90 Benign nevi 393	Melanoma 374 Seborrheic keratosis 254 Benign nevi 1372	Melanoma 30 Seborrheic keratosis 30 Benign nevi 90	Support research and development of algorithms for automated diagnosis

Table 2.3 Datasets Comparison (continued)

Dermnet	14-Apr-20	23	21	2	19500	Train 15500 test 4000	Acne and Rosacea 312 Actinic Keratosis Basal Cell 288 Atopic Dermatitis 123 Bullous Disease 113 Cellulitis Impetigo and Bact 73 Eczema 309 Exanthems and Drug Eruptions 101 Hair Loss Photos Alopecia 60 Herpes HPV 102 Light and Disorders of Pigmentat 143 Lupus and Connective Tissue 105 Melanoma Nevi and Mole 116 Nail Fungus 261 Poison Ivy /Contact Dermatitis 65 Psoriasis Lichen Planus 352 Scabies Lyme 108 Seborrheic Keratoses 343 Systemic 152 Tinea Ringworm 325 Urticaria Hives 53 Vascular Tumors 121 Vasculitis 105 Warts Molluscum 272	Acne and Rosacea 840 Actinic Keratosis Basal Cell 1149 Atopic Dermatitis 489 Bullous Disease 448 Cellulitis Impetigo and Bact 288 Eczema 1235 Exanthems and Drug Eruptions 404 Hair Loss Photos Alopecia 239 Herpes HPV 405 Light and Disorders of Pigmentat 568 Lupus and Connective Tissue 420 Melanoma Nevi and Mole 463 Nail Fungus 1040 Poison Ivy /Contact Dermatitis 260 Psoriasis Lichen Planus 1405 Scabies Lyme 431 Seborrheic Keratoses 1371 Systemic 606 Tinea Ringworm 1300 Urticaria Hives 212 Vascular Tumors 482 Vasculitis 416 Warts Molluscum 1086	N/A	Provides authoritative skin information to healthcare professionals and patients
PH2	3-Jul-13	3	2	1	200	N/A	Common nevi 80 Atypical nevi 80 Melanomas 40	N/A	N/A	Facilitate comparative studies on both classification algorithms of dermoscopic images.
Dermofit	1-Aug-08	10	7	3	1300	N/A	Actinic Keratosis 45 Melanocytic Nevus 331 Seborrhoeic Keratosis: 257 Intraepithelial carcinoma 78 Pyogenic Granuloma 24 Haemangioma 96 Dermatofibroma 65 Melanoma 76 Basal Cell Carcinoma 239 Squamous Cell Carcinoma 88	N/A	N/A	aims to develop an interactive tool for indexing a skin cancer image database, allowing users to compare live skin lesions.

2.12. Conclusion

In conclusion, the meticulous comparison of skin disease datasets has led to the selection of ISIC 2019 dataset for our project. Recognizing their status as the most challenged source of dermatological data.

This selection of ISIC2019 is based on four important criteria. Firstly, the availability of the dataset where it can be accessed. Secondly, the primary objective of the dataset focuses on training neurons for artificial intelligence, rather than being exclusively tailored to the medical domain. Thirdly, the dataset contains an appropriate number of skin diseases. Fourth, each type of skin disease contains a relatively large number of images. which lays a solid foundation for the subsequent phases of our project.

Related work and algorithms

Skin disease recognition and classification using deep learning techniques rely on the effectiveness of algorithms employed for image analysis and pattern recognition. However, the choice of algorithms is crucial as they directly impact the accuracy and robustness of the classification system. In this documentation, we provide a comprehensive overview of the current state-of-the-art algorithms utilized in skin disease recognition and classification tasks. At the end of the algorithms chapter, based on our comparison, we will determine the most suitable set of algorithms to be used in our project.

3.1. Introduction

Research papers were discussed in some detail in terms of the models and algorithms used in them, in addition to explaining the structure of each algorithm in terms of layers, hyperparameters, and the build of the model, in addition to the data set that was used, preprocessing techniques performed on the data set, and the final accuracy of the model.

We rely on ten points to detail each algorithm:
Paper Name: Name of the paper used for skin disease classification.

Publication/Author: The publication date and authors of the research paper or study where the algorithm was introduced or applied.

Dataset used: The dataset used for training, testing, and validating the algorithm, including its source, size, composition, attributes like image resolution, format, and diversity, and whether it includes dermoscopic examinations or skin surface microscopy.

Preprocessing Techniques: Description of any preprocessing methods or techniques applied to the input data before feeding it into the algorithm.

Structure: This part provides an overview of an algorithm used for skin disease classification, including its architecture, functions, and

hyperparameters. It also details the key functions performed by the algorithm's components, such as activation, loss, and optimization algorithms.

Layers: It covers the number of layers it uses and their components.

Model building: When describing the process of model building in the context of skin disease classification, it involves the steps taken to construct, train, and optimize the machine learning or deep learning model used for classification tasks.

Result: This section, specifically focusing on evaluation metrics such as Accuracy, Recall, and Precision, serves to quantitatively assess the performance of the skin disease classification model.

Limitation: It's included factors that may impact the performance, reliability, and generalizability of the model.

Observation about pons and pros: regarding the use of skin disease classification algorithms can help provide a balanced view of their potential benefits and limitations from our point of view.

3.2. ResNet-152

Skin Lesions Classification Using Convolutional Neural Networks in Clinical Images [13].

This paper was published on 6 Dec 2018 by Danilo Barros Mendes, Nilton Correia da Silva.

This paper merge three datasets:

- The MED-NODE dataset has 170 images divided between 70 melanoma and 100 nevus cases.
- The Edinburgh Dermofit Image Library dataset has 1,300 images divided into 10 lesions.

- Atlas dataset has 3816, images Basal Cell Carcinoma 1,561, Lentigo 69, Malignant Melanoma 228, Melanocytic nevus (mole) 626, Seborrheic keratosis 897, Wart 435.

After merging the datasets, 10% to 20% of them were taken as a test, and the remaining part was increased using data augmentation. After that, it was divided into 80% for training and 20% for validation.

Increase images count by Rotation 0.5, Random zoom 0.4, Flip horizontally 0.7, Flip vertically 0.5, Random distortion 0.8, Lightning variance 0.5. The images of diseases in the data were increased by 20 to 30 times.

Structure: ResNet-152 model is discussed in this paper.
ReLU activation function
Neither optimizer nor loss function is mentioned in this paper.
Hyperparameter:
Base in 0.01, weight decay 0.00001, momentum 0.9, gamma 0.1, batch size 5, max tier 176180, test tier 22023, test interval 2000, step size 17618, tier size 12, epochs to 10.

ResNet-152 consists of 152 layers with 1 MaxPool layer, 151 conv layer, and 1 avgPool layer.

Transfer learning applied: ResNet-152 pre-trained for the ImageNet dataset first.

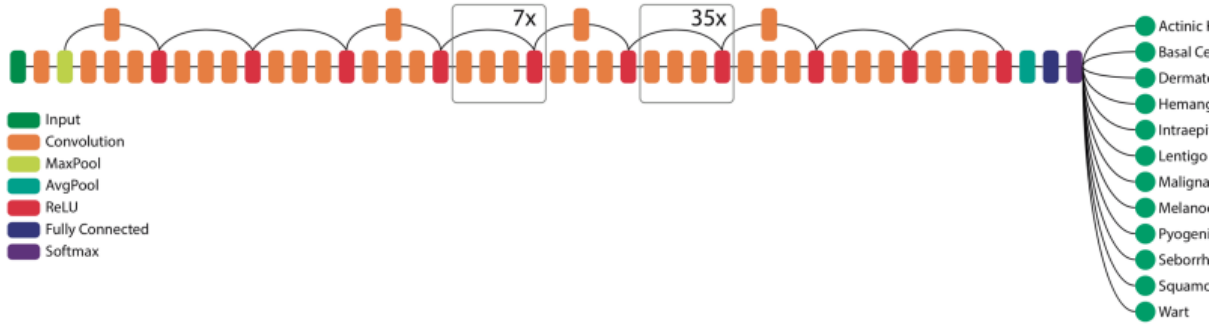


Fig.1 Architecture of Resnet152

Input Layer: Initially receives input data, often images, formatted to the appropriate dimensions for processing.

Convolutional Layers: multiple residual blocks facilitate learning deeper, each containing several convolutional layers responsible for feature extraction.
Max Pooling: reducing spatial dimensions and enhancing computational efficiency.
Rectified Linear Unit (ReLU): An activation function applied after convolutional layers.
Average Pooling: Towards the end of the network, average pooling is applied, aiding in further abstraction and feature summarization.
Fully Connected Layer: Positioned at the end of the network, these dense layers, enabling the network to output predictions.
SoftMax Activation: The final layer utilizes the SoftMax function, converting the network's raw outputs into probabilities for different classes in classification tasks.

Finally, the model used in the testing phase was the product of the iteration number 38,000. This training phase took an uninterrupted total time of 35 hours (approximately 167 seconds for every 50 iterations).
All 11 lesions except for Actinic Keratosis had accuracy over 80% in the confusion matrix, accounting for 78% total accuracy. However, this metric has a bias due to uneven class distribution.
Predicted as Hemangioma, Basal Cell Carcinoma with 100% confidence,
Predicted as Melanocytic Nevus with 97% confidence.

Metric has a bias attached to it, since the distribution of the classes is not even.
It is necessary to further test the model with more data, with more diversity (different ethnicities and ages), and then investigate the results for improvements.

Pros: All 11 lesions apart from Actinic Keratosis had accuracy over 80%.

Cons: Actinic Keratosis accuracy is less than other diseases because its number of images is very small compared to other diseases which lead to bias, A very small number of images were determined for the testing process.

3.3. ResNet-101

Early Detection of Skin Cancer Using Deep Learning Architectures: Resnet-101 and Inception-v3 [14].

published in October 2019 by Ahmet Demir, Feyza Yilmaz, Onur Kose

Dataset is taken from ISIC-Archive, which is balanced dataset of images for benign skin moles and malignant skin moles.
The training data set contains 2437 images in total, 1330 for benign class and 1107 for malignant class.
660 images are used in the test process, 360 benign and 300 malignant.
200 which are taken from the training part of original dataset randomly are used in validation process, 100 in benign and 90 in malignant.

ResNet-101 Uses residual connections to address the vanishing gradient problem and allows for the training of very deep networks.
ResNet-101 model contains 104 convolutional layers in total, which are used to extract features from the input images. Which are divided into 33 blocks of layers, each consisting of one or more convolutional layers, followed by a summation operator that combines the output of the block with the input of the block or a previous block possible by using residual connections which can be written as y = F(x) + x.

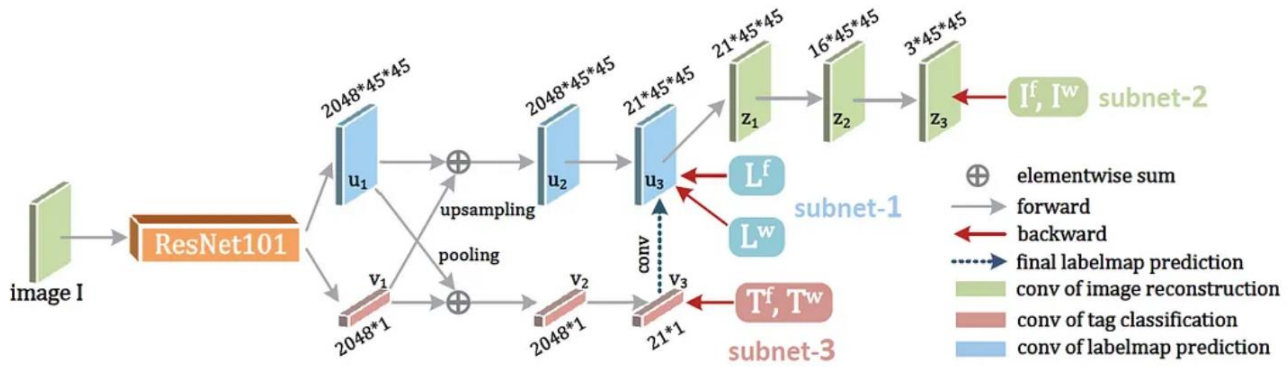


Fig 2. Architecture of Resnet101

The convolutional layers use different filter sizes, strides, and padding to control the size and shape of the output feature maps. ResNet-101 model used residual connections to enable the flow of information and gradients across the blocks, and to avoid the vanishing gradient problem that affects deep neural networks.
Out of the 33 blocks, 29 blocks use the previous block's output directly by using a residual connection and use it as the first operand of the summation operator at the end of the block. The second operand is the output of the convolutional layers within the block.
The remaining 4 blocks use the previous block's output indirectly by applying a convolution layer with a filter size of 1x1 and a stride of 1, followed by a batch normalization layer, which performs normalization and scaling operations. The output of this layer is then used as the first operand of the summation operator at the end of the block. The second operand is the same as before.

The use of 1x1 convolution and batch normalization layers helps to reduce the number of parameters and to increase the efficiency and stability of the model.
The training process of these two architectures is completed in 60 epochs. The learning rate is determined as 0.001 at the beginning of the training process and this ratio diminishes gradually at every stage of training.
ResNet-101 train and loss lines are almost constant after 50 epochs. No farther increase of accuracy was reached after this point. Validation loss is nearly constant after 30 epochs.

ResNet-101 has a validation accuracy of 0.8900, a benign accuracy of 0.8277, a malign accuracy of 0.8566, and an F-1 score of 0.8409.

Pros of ResNet-101 that it has many layers, allowing it to capture complex features and patterns in images also It utilizes residual connections, which help in addressing the vanishing gradient problem.

And ResNet-101 may also suffer from overfitting if the data is not sufficient or noisy.

3.4. Inception-v3

Early Detection of Skin Cancer Using Deep Learning Architectures: Resnet-101 and Inception-v3

October 2019 and published by Ahmet Demir, Feyza Yilmaz, Onur Kose

Dataset is taken from ISIC-Archive, which is balanced dataset of images for benign skin moles and malignant skin moles.
The training data set contains 2437 images in total, 1330 for benign class and 1107 for malignant class.
660 images are used in the test process, 360 benign and 300 malignant.
200 which are taken from the training part of original dataset randomly are used in validation process, 100 in benign and 90 in malignant.

Inception-v3 is a commonly used image recognition model that has been shown to attain an accuracy rate of greater than 78.1% on the ImageNet dataset. Composed of a 42-layer deep neural network

Inception-v3 model contains 42-layer deep neural network and also consists of symmetric and asymmetric building blocks, including convolutions, max pooling layers, average pooling, dropouts, and fully connected layers.
Inception-v3 Uses a combination of convolutional layers with different filter sizes and pooling operations to capture features at different scales.

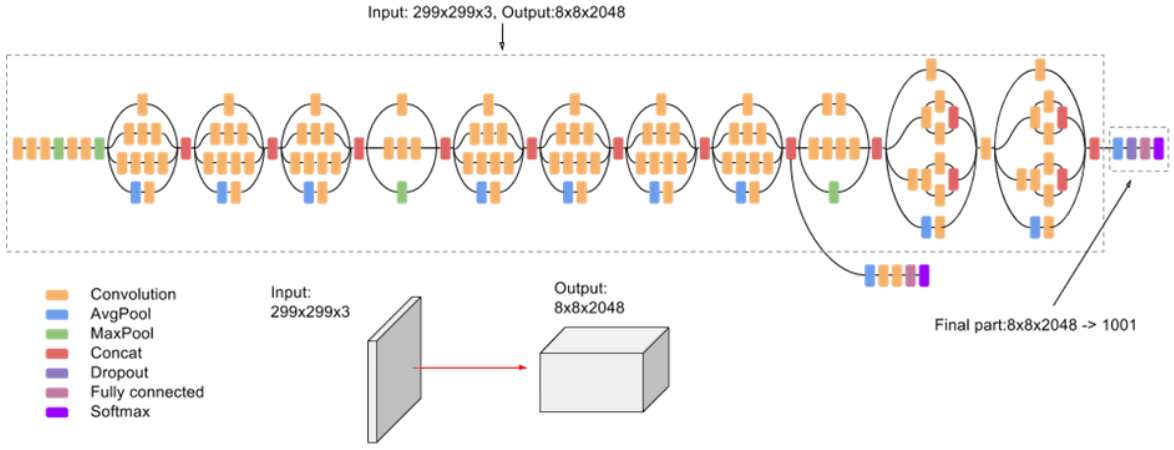


Fig3. Architecture of Inceptionv3

Inception-v3 train, loss and validation loss is scarcely constant after 40 epochs. The train and validation accuracy saturation points are reached after 40 epochs.
Inception-V3 has slightly higher values than ResNet-101 in all four metrics Validation Accuracy, Benign Accuracy, Malign Accuracy and F-1 Score. Where Inception-V3 has a validation accuracy of 0.9000, a benign accuracy of 0.8861, a malign accuracy of 0.8600, and an F-1 score of 0.8742. ResNet-101 has a validation accuracy of 0.8900, a benign accuracy of 0.8277, a malign accuracy of 0.8566, and an F-1 score of 0.8409.

Cons that both can be computationally expensive due to its large number of layers and complex architecture.
Inception-V3 may also lose some information or details by using multiple filter sizes and concatenation.

3.5. DenseNet-121

Classification of Skin Cancer with Deep Transfer Learning Method [15].

It published on Oct.10,2022 and published by Doaa Khalid Abdulridha Al-Saedi, Serkan Savaş

The dataset used is an open-access dataset called the International Skin Imaging Collaboration (ISIC) skin cancer which included 1800 benign and 1497 malignant tumor images.

In preprocessing dataset were converted to 224 x 224 x 3 dimensions to be used in the pre-trained model also implementing the preprocess input function for every transfer learning algorithm and This function applies image scaling.
In the next step in this phase contains feature extraction pre-trained models were used to extract features through transfer learning.

Use every kind of pre-trained model was implemented separately without the output layer from the original model and added three layers in the end of each model, the first and the second layers are same in their Components.

Every layer of these consists of 128 neurons with ReLU activation function the feature selection and classification are optimized using a newly created Red Fox Optimization (DRFO) algorithm,
The training Loss value decreased below a very small value of 0.025, especially after the 10th epoch.

DenseNet including those in the same dense block and transition layers, spread their weights over several inputs, deeper layers can utilize characteristics that were collected earlier. 120 Convolutions and 4 Avg Pool make up DenseNet-121

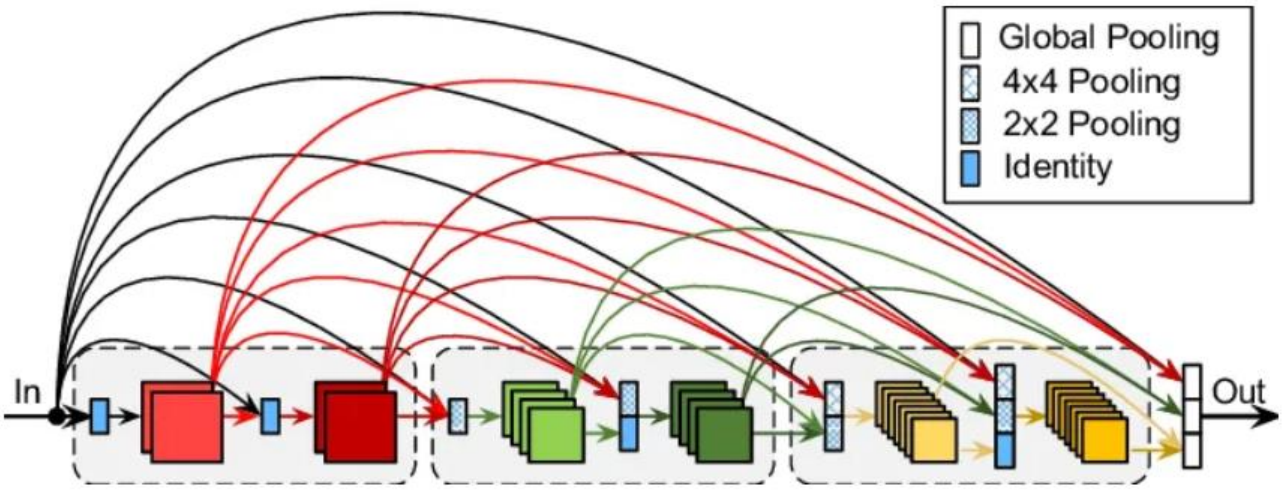


Fig.4 Architecture of DenseNet121

They use data set (ISIC) which included 1800 benign and 1497 malignant tumor images, use preprocessing dataset were converted to 224 x 224 x 3 dimensions to be used in the pre-trained model also implementing the preprocess input function for every transfer learning algorithms and This function apply image scaling and use algorithm (densent121) which including the same dense block and transition layers, spread their weights over several inputs, deeper layers can utilize characteristics that were collected earlier. 120 Convolutions and 4 Avg Pool make up DenseNet-121.They use the feature selection and classification are optimized using a newly created Red Fox Optimization (DRFO) algorithm.in the end they got best accuracy rate that can be achieved with the densent121 is 99.6% in the

study. In addition, the validation accuracy rate of this model has reached a high rate of 98.35%

The study uses various models, including DenseNet, Xception, InceptionResNetV2, ResNet50, and MobileNet, to achieve high accuracy rates. The model is implemented separately without the output layer and adds three hidden layers for feature extraction. The DenseNet121 model achieved a 99.6% accuracy rate. However, it faces challenges in dealing with similar images and the similarity of colors in melanoma lesions. To address these issues, data augmentation techniques are employed.

3.6. DenseNet-201

Classification of Skin Cancer Lesions Using Explainable Deep Learning [16].

It published in 10 May 2023and published by Muhammad Zia Ur Rehman, Fawad Ahmed, Suliman A. Alsuhibany, Sajjad Shaukat Jamal, Muhammad Zulfiqar Ali and Jawad Ahmad.

The dataset available with name (Skin Cancer: Malignant vs. Benign),
is part of ISIC archive it consists of two categories: malignant and benign. A total of 3297 images are present in this dataset.
The category (benign: contains 1800 images), (malignant:contains 1497 images)
They used) contrast expansion technology (which is used in image processing to improve the clarity of features in the image. Which helps in seeing details better, especially in cases where some points aren't defined or faint.

In this model 201 layers of the original model, DenseNet201 model consists of 4 dense blocks and 3 transition layers that act as connections between the two dense blocks, Inside the dense block, each convolution layer is connected to other convolution layers. After every dense block, the size of the feature map is increased. The transition layers act as the down sampling layers. In DenseNet201, down sampling is performed using average pooling.

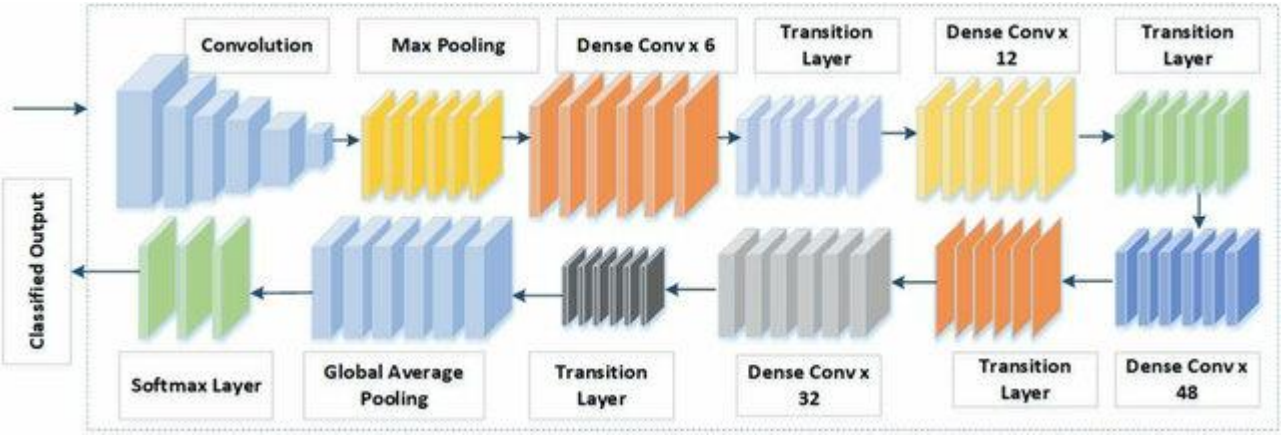


Fig.5 Architecture of DenseNet201

They use part of ISIC archive it consists of two categories: malignant and benign. A total of 3297 images are present in this dataset. The category benign: contains 1800 images, malignant: contains 1497images.
Used contrast expansion technology, which is used in image processing to improve the clarity of features in the image. model consists of 4 dense blocks and 3 transition layers that act as connections between the two dense blocks, Inside the dense block, each convolution layer is connected to other convolution layers. After every dense block, the size of the feature map is increased. The transition layers act as the down sampling layers. In DenseNet201, down sampling is performed using average pooling.

It Reach to Accuracy: 95.50%, Sensitivity: 93.96%, Specificity: 97.06%, Precision: 97.02%, F1 Score: 95.46%.

Gradient-weighted Class Activation Mapping (Grad-CAM) offers feature learning insights and Kaggle data availability. However, it lacks recall and provides limited information on training parameters like epochs, learning rate, and optimization strategy.

3.7. MobileNet v1

Skin Lesion Analyzer: An Efficient Seven-Way Multi-class Skin Cancer Classification Using MobileNet [17].

It will be published on 26 May 2020 and published by Saket S. Chaturvedi, Kajol Gupta & Prakash S. Prasad.

Dataset used is an open-access dataset called HAM10000, it is a benchmark dataset with over 50% of lesions confirmed by pathology. The dataset consists of a total of 10.015 dermoscopy images.
The dermoscopy images in the dataset were downscaled to 224 * 224-pixel resolution from 600 * 450-pixel resolution to make images compatible with MobileNet model.
There are 10.015 images in the dataset that were split into the training ser 9077 images & validation 938 images.

The MobileNet convolutional neural network was utilized. It was pretrained on a dataset of 1,280,000 images, which included 1,000 object classes from the 2014 ImageNet Challenge.
The model was trained on a training set containing 38,569 images.

The architecture used in the study consists of 25 layers, including four Conv2D layers, seven Batch Normalization layers, seven ReLU layers, three ZeroPadding2D layers, and single DepthwiseConv2D, Global Average Pooling, Dropout, and Dense layers (refer to Fig. 2 for a visual representation).

Fig.6 Architecture of MobileNetV1

The weighted average of precision, recall, and f1-score were found to be 89%, 83%, and 83%, respectively.

The model's performance was evaluated using various metrics, including accuracy, precision, recall, and F1-score.
The model achieved high precision and recall values for all seven skin lesion classes, indicating its ability to correctly classify different types of lesions.

Limited Dataset Size for Some Classes:
The ISIC 2019 dataset used for training and testing the model contains a limited number of images for some skin lesion classes, such as Dermatofibroma and Vascular Lesion. This imbalance in the dataset could potentially affect the model's performance on these classes.
Limited Clinical Expertise: The paper does not involve dermatologists or other medical experts in the evaluation of the model's performance. Clinical expertise is important to assess the model's ability to correctly classify skin lesions in a way that is meaningful for medical diagnosis.

Pros about its High accuracy: The proposed deep learning model achieved an overall accuracy of 91.3% on the ISIC 2019 test set, which is comparable to the performance of dermatologists.
Multi-class classification: The model can classify seven different types of skin lesions.
Cons about it No balance between images of diseases. This imbalance in the dataset could potentially affect the model's performance on these classes.
The model was trained on a dataset that may contain biases, such as overrepresentation of certain skin types or lesion characteristics. This could potentially lead to biased predictions, particularly for underrepresented groups.

3.8. MobileNet v2 & LSTM

Classification of Skin Disease Using Deep Learning Neural Networks with MobileNetV2 and LSTM [18].

It will be published on 18 April 2021 by Parvathaneni NagaSrinivasu, Jalluri Gnana SivaSai, Mohammed Fazal Ijaz, Akash Kumar, Wonjoon Kim &James Jin Kang.

The HAM10000 dataset is used in this paper. This dataset contains more than 10,000 dermoscopy images.

The input size of the image is 224 × 224 × 3. The first two values (224× 224) indicate the height and width of the image. These values should always be greater than 32. The third value suggests that it has 3 input channels. The proposed architecture has 32 filters, and the filter size is 3 × 3 × 3 × 32.
A random (rand) function is applied to split the data into the training data (7224) and validation data (1255).

The main function of the paper is to bring in the state of art technique, namely the MobileNet V2 with LSTM component for the purpose of the precise classification of skin disease from the image that is captured from the mobile device.
MobileNet V2 is used in classifying the type of skin disease, and LSTM is used to enhance the performance of the model by maintaining the state information of the features that it comes across in the previous generation of the image classification.
The proposed MobileNet V2 with LSTM performance is evaluated through the hyperparameters like training and validation loss measures that determine the proposed model's capabilities.

The MobileNet V2 architecture consists of a residual layer and a downsizing layer with a ReLu component. It includes three sub- layers: 1 × 1 convolution with ReLu6, Depth-Wise Convolution, and a 1 × 1 convolution layer without non-linearity. ReLu6 is used in the output domain for robustness and randomness. All layers have the same output channels, and a 3 × 3 filter is used for training.

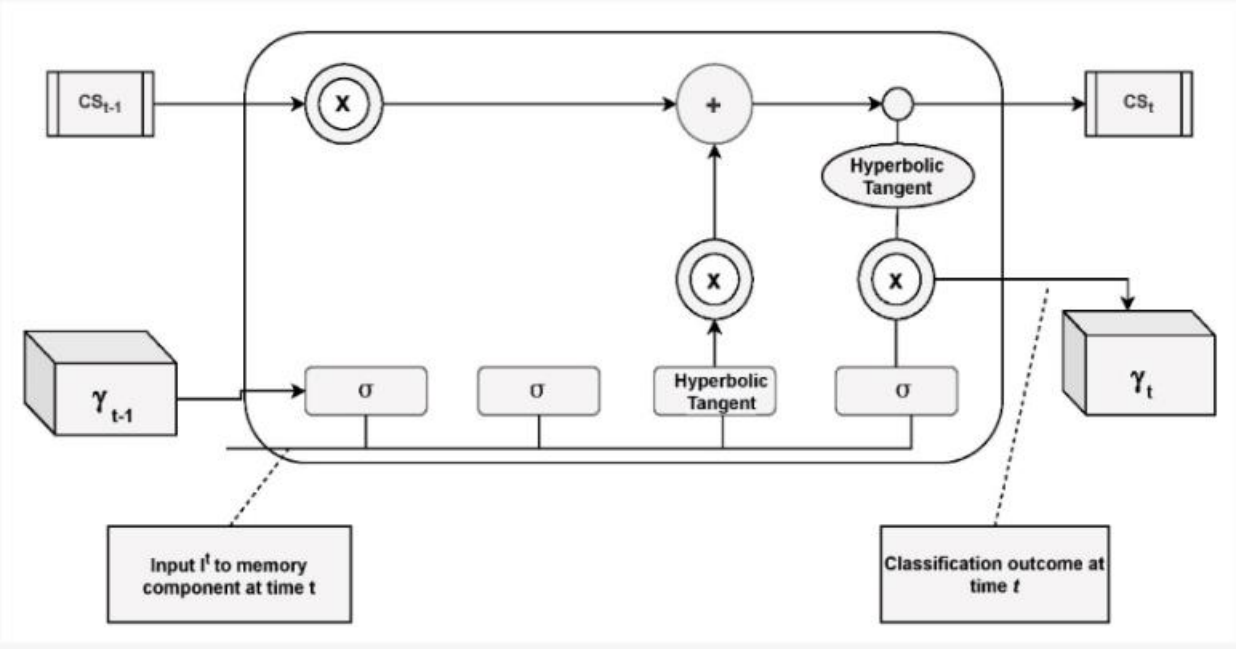


Fig.7 component of LSTM

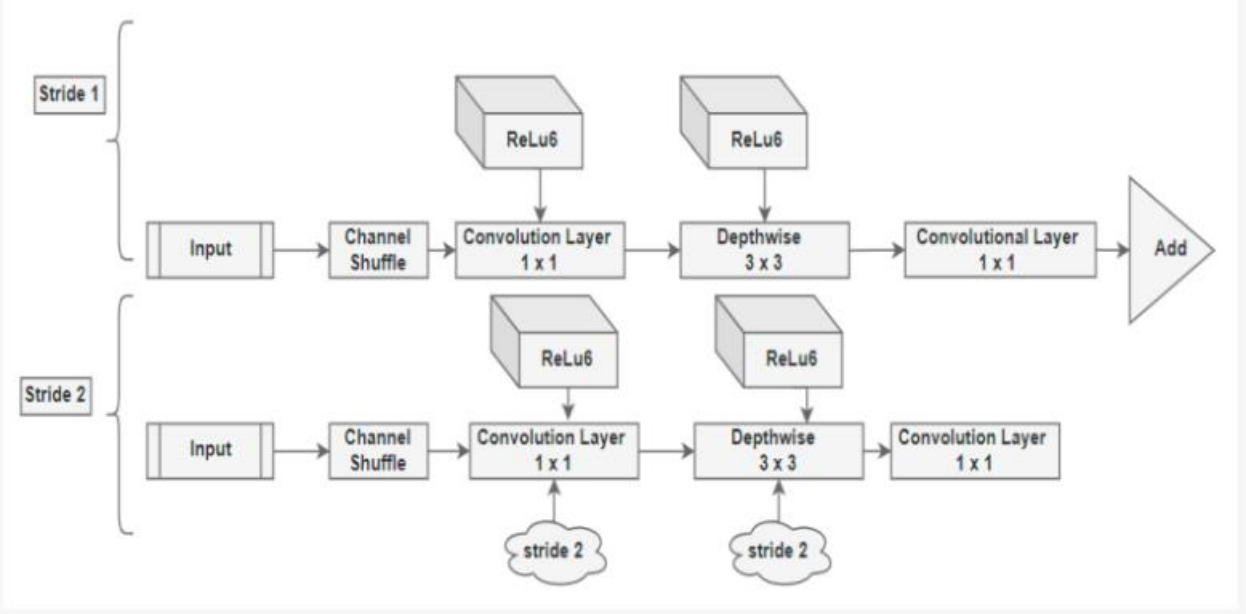


Fig.8 Architecture of MobileNetV2

Here is a general overview of the process:
Data Collection: Gather a dataset of skin images, the paper used the specific dataset used (HAM10000).

Preprocessing: preprocess the collected images to ensure they are standardized & suitable for training the model. This may involve resizing,

cropping, normalizing pixel values & applying data augmentation techniques.

Model architecture: the paper mentions the use of a deep learning neural network based on MobileNet & LSTM. MobileNet V2 -> is a convolutional neural network architecture optimized for mobile and embedded devices. LSTM (Long Short-Term Memory) is a type of recurrent neural network that can capture temporal dependencies. including the number of layers, the configuration of the LSTM, and any additional components.
Training: Split the preprocessed dataset into training and validation sets.
Evaluation: Evaluate the trained model on the validation set to assess its performance.
Testing: Once the model is trained and evaluated, you can use it to classify new, unseen skin disease images.
The Fine-Tuned Neural Network-based skin disease classification model has achieved a reasonable accuracy of 89.90% for the validation set.
The model achieved high precision and recall values for all seven skin lesion classes, indicating its ability to correctly classify different types of lesions.
From limitations A random (rand) function is applied to split the data into the training data (7224) and validation data (1255).
The considered dataset is slightly imbalanced because some skin diseases are more, and some are less in number.

Pros that the main advantage of using the MobileNet architecture is that the model needs comparatively less computational effort than the conventional CNN model.
Cons that No balance between images of diseases. This imbalance in the dataset could potentially affect the model's performance on these classes.

3.9. VGGNet

Classification of skin cancer using VGGNet model structures [19].

It Published in 8/12/2022 by Volkan KAYA, İsmail AKGÜL

In this study, a dataset containing images of benign skin cancers and malignant skin cancers was use (Kaggle,2021).
In this dataset, 1800 benign skin cancer images with 224x224 pixel size and 1497 malignant skin cancer images are in RGB color format. A total of 3297 skin images are divided into training (80%), and test (20%).

There was no mention of image processing explicitly, but they said the VGGNet architecture takes an input image in RGB color format with a size of 224×224 pixels during the training and a 3×3 filter is used in each convolution layer.

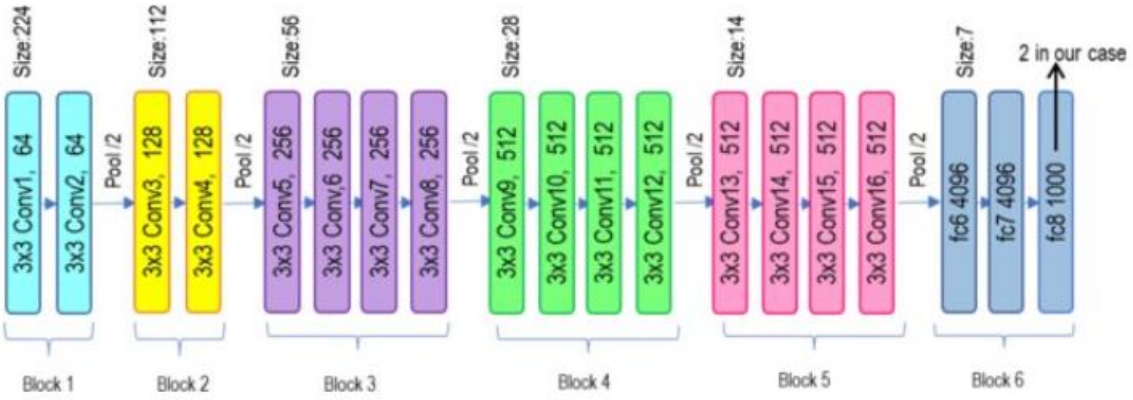


Fig.9 Architecture of VGG19

VGGNet is a convolutional neural network architecture with multi-layer GPU support, used for object recognition models. It uses RGB input images, 3x3 filters, and a 2x2 pooling layer to reduce data size.

In this study, VGG-11, VGG-13, VGG-16, and VGG-19 model structures for the training of these models, a dataset containing 1000 benign and 1000 malignant skin cancer types was used, and it was seen that the VGG-11 model had superior performance compared to other model structure.

VGGNet consists of several layers, including input, convolutional, pooling, fully connected, and output layers.
The specific number of layers in VGGNet depends on the model structure being used.
VGG-11: 18 layers
VGG-13: 20 layers

VGG-16: 23 layers
VGG-19: 26 layers

The models were implemented using the Python programming language, these model constructs were trained and tested using a dataset of 3297 skin cancer images.
The best classification success accuracy was obtained with the VGG-11 model structure, with 83 and 0.45 loss rates.
The model structure with the lowest classification success accuracy was 81 with VGG-19 and the loss rate was 0.37 As a result.
These model constructs were trained and tested using a dataset of 3297 skin cancer images. According to the training and test results, the best classification success accuracy was obtained with the VGG-11 model structure, with 83% and 0.45 loss rates. The model structure with the lowest classification success accuracy was 81% with VGG-19 and the loss rate was 0.37.

From Pros The deep learning methods used in VGGNet models, which analyze raw data and classify complex structures, have shown successful results in disease recognition and detection, including skin cancer.
-VGGNet model structures, such as VGG-11, VGG-13, VGG-16, and VGG-19, have shown good success in classifying skin cancer types with high accuracy.

From Cons The study did not provide detailed information about the specific architecture and training process of the VGGNet models, such as the number of layers and optimization techniques used.
-The dataset used in the study consisted of two classes of skin cancer, which may limit the generalizability of the models to datasets with multiple classifications.

3.10. VGG16

Identification and Classification of Skin Diseases using Deep Learning Techniques [20].

Posted by Venkatesh R Pai1, Soujanya G Pai, P M Suhasi and P M Rekha1
Date: March 1st, 2023.

The collected dataset consists of 5 classes namely, Impetigo (Bacterial), Cellulitis (Bacterial), Sporotrichosis (Fungal), Ringworm (Fungal), and Healthy Skin. This contains nearly 500 images where the frequency of each class is unequal.

Label Encoding and One-hot Coding The 5 class labels of the dataset are Cellulitis Bacteria, Healthy Skin, impetigo Bacteria, Ringworm Fungus, and Sporotrichosis Fungus. These class labels are converted to the integers 0,1,2,3, and 4 respectively.
Normalization Each pixel of an image is divided by 255 which places all the pixel values in the common range of [0,1]
Balancing Dataset For the image dataset in consideration, the Class Weight method is performed. If class weights are set to 'balanced', the model automatically adjusts the class weights inversely proportional to the frequency of each class.
Image Augmentation It reduces overfitting and increases the ability of the CNN model to generalize while extracting the features from the images.
Augmentation techniques such as width shift, height shift, zooming, and vertical, and horizontal flips are performed.

The paper proposes a methodology that utilizes deep learning techniques, specifically a Convolutional Neural Network (CNN) with the fine-tuned version of the VGG16 architecture, for identifying and classifying skin diseases caused by bacteria and fungi using non-dermoscopic images.
-The dataset used in the study consists of five classes: Cellulitis, Impetigo (bacterial skin diseases), Ringworm, Sporotrichosis (fungal skin diseases), and Healthy Skin.

VGG16 consists of 21 layers but only 16 layers of those are trainable. The architecture includes 13 convolutional layers, 5 max-pooling layers, and 3 fully connected layers. Several small filters are used throughout the network with max-pooling operations being performed periodically. Instead of using

large filters, VGG16 architecture considers smaller filters with more depth in the network.

The system adopts a Convolutional Neural Network (CNN) as the deep learning algorithm for image processing applications, The system makes use of the fine-tuned version of the VGG16 architecture, which is a type of CNN, to leverage the power of transfer learning.
The system trains the CNN model on the preprocessed images and evaluates its performance.

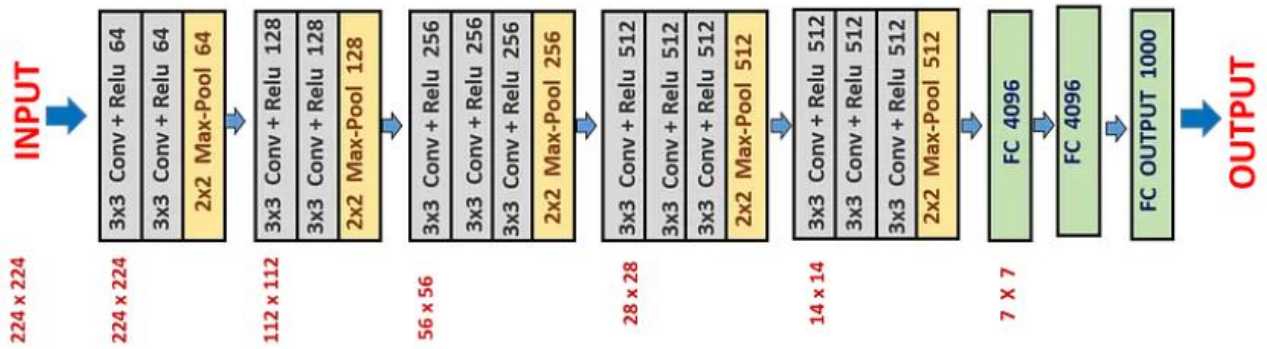


Fig.10 Architecture of VGG16

The developed model is tested with the testing set images that consisted of 97 images. The model achieved a test accuracy of 86% and an F1-score of 85%.

From Pros Deep learning techniques, specifically Convolutional Neural Networks (CNN), are known to be highly effective for image processing tasks, making them suitable for identifying and classifying skin diseases.

The use of transfer learning, specifically the fine-tuned version of the VGG16 architecture, leverages pre-trained models and can improve the accuracy and performance of the system.

From Cons The system relies on non-dermoscopic images, which may limit its effectiveness In accurately diagnosing certain skin diseases that require dermoscopic images for accurate analysis.

The dataset used for training the model consists of only five classes of skin diseases, which may not cover the full spectrum of skin conditions, potentially limiting the model's ability to accurately classify less common or rare skin diseases.

Table 3.1 *Algorithms Comparison*

Paper Name	Dataset	Preprocessing	Algoritm Structure	Evaluation Matrix	Limitation
Skin Lesions Classification Using Convolutional Neural Networks in Clinical Images	MED-NODE dataset has 170 images. Atlas dataset has 3816 images. Dermofit Image Library dataset has 1,300 images	Data Augmentation: increase images count by Rotation 0.5, Random zoom 0.4, Flip horizontally 0.7, Flip vertically 0.5, Random distortion 0.8, Lightning variance 0.5.	This paper discusses the ResNet-152 model, which consists of 152 layers with a MaxPool layer, 151 convolution layers, and 1 average pool layer.	All 11 lesions, except Actinic Keratosis, had over 80% accuracy in the confusion matrix, accounting for 78% total accuracy. Examples include Haemangioma, Basal Cell Carcinoma, and Melanocytic Nevus.	Model Bias and Need for Improvement • Metric has bias due to uneven class distribution. • Model needs more data and diversity.
Skin Disease Classification Using Deep Neural Networks with Resnet	The dataset is collected from websites Kaggle	Image resizing and Data Augmentation: increase images count by cropping, rotating, padding, transposing like top-bottom transposition, left-right rotation.	This paper discusses the ResNet model, a CNN with 27 pooling layers and 22 total layers, based on the Inception architecture.	Model's accuracy was 99.87% for training and 98.84% for testing. The loss was 0.0457 for training and 0.1099 for testing.	N/A
Classification of Images of Skin Lesion Using Deep Learning	The ("Human Against Machine with 10000 images") dataset	The preprocessing involves merging image data (part 1&2 HAM) from different parts, reading a CSV file, creating informative columns, handling missing values, and checking data types.	The CNN uses 8 layers, including Conv2D, Pooling, Flatten, and Fully Connected layers, to convert feature maps into 1D vectors, with a final 1D vector converted into a 1D vector.	The Model provides achieve accuracy levels above 80% on average for all 7 types of skin diseases.	Our model has some misclassification for Basal cell carcinoma, Vascular sores, and Melanocytic nevi, but Actinic keratosis has the least misclassified type, ensuring good accuracy and precision for other classes.
Automated Diagnosis of Skin Disease MultiClass Image Classification Using Deep Convolution Neural Network	The DermaNet dataset , focusing on nine different skin diseases	Data pre-processing normalizes image size to 120x120, considering accuracy, space, and time complexity, and data sampling ensures good performance for deep learning model.	Convolution Neural Network uses convolutional layers for feature extraction, filter size, output size, pooling layers, and fully connected layers for classification tasks.	The deep convolution neural network enhances skin lesion image classification performance, achieving a weighted average of 96% after 25 iterations, enhancing precision, recall, and f1-score.	N/A

Table 3.4 *Algorithms Comparison (continued)*

Early Detection of Skin Cancer Using Deep Learning Architectures: Resnet-101 and Inception-v3	Dataset is taken from ISIC-Archive	No preprocessing was mentioned in this paper.	The paper employs ResNet-101 and Inception-v3 convolutional neural network models for image feature extraction, each with 104 and 42 layers, respectively, and symmetric and asymmetric building blocks.	Inception-V3 and ResNet-101 have varying validation and benign accuracy rates, with Inception-V3 having a higher accuracy of 0.9000 and ResNet-101 having a higher accuracy of 0.8900.	N/A
Classification of Skin Cancer with Deep Transfer Learning Method	the International Skin Imaging Collaboration (ISIC)	The dataset was preprocessed to 224x224x3 dimensions, implementing preprocess input functions for transfer learning algorithms and image scaling, and then used for feature extraction through transfer learning.	This paper uses the DenseNet-121 model, which includes dense block and transition layers, 120 convolutions, and 4 average pool for deeper layers to utilize collected characteristics.	Accuracy : 99.6%, Recall: 99.4, Precision: 99.7%, F1-Score: 99.5%,	N/A
Classification of Skin Cancer Lesions Using Explainable Deep Learning	ISIC archive and includes 3297 images	Contrast expansion technology enhances image processing clarity, improving detail perception, particularly in cases where points are unclear or faint.	This paper uses the DenseNet201 model, ReLu6 as an activation function, Gradient-weighted Class Activation Mapping for feature learning, and average pooling for downsampling.	Accuracy: 95.50%, Sensitivity: 93.96%, Specificity: 97.06%, Precision: 97.02%, F1 Score: 95.46%.	N/A
Skin Lesion Analyzer: An Efficient Seven-Way Multi-class Skin Cancer Classification Using MobileNet.	The HAM10000 dataset is used in this paper	The dermoscopy images in the dataset were downscaled to 224 * 224-pixel resolution from 600 * 450-pixel resolution to make images compatible with MobileNet model.	This paper uses a MobileNet model consisting of 25 layers, including Conv2D, BatchNormalization, ReLU, ZeroPadding2D, and DepthwiseConv2D layers, for a visual representation.	The weighted average of precision, recall, and f1-score were found to be 89%, 83%, and 83%, respectively.	ISIC 2019 Model Training and Testing • Limited images for Dermatofibroma and Vascular Lesion classes. • Potentially affects model performance.

Table 3.2 Algorithms Comparison (continued)

Classification of Skin Disease Using Deep Learning Neural Networks with MobileNet V2 and LSTM	The HAM10000 dataset is used in this paper	The image's input size is 224 x 224 x 3, with 32 input channels and 32 filters, with height and width values greater than 32.	This paper uses a MobileNet V2 architecture with a residual layer, downsizing layer, ReLu component, three sub-layers, ReLu6 for robustness and randomness, and a 3 x 3 filter for training.	The Fine-Tuned Neural Network-based skin disease classification model has achieved a reasonable accuracy of 89.90% for the validation set.	Data Splitting with Random Function • Splits data into training and validation. • Data imbalance due to varying skin disease numbers.
Classification of skin cancer using VGGNet model structures	A total of 3297 skin images are divided into training (80%), and test (20%)	The VGGNet architecture uses an RGB input image with 224x224 pixels size for training, with a 3x3 filter used in each convolution layer.	VGG-11: 18 layers VGG-13: 20 layers VGG-16: 23 layers VGG-19: 26 layers	The VGG-11 model structure achieved the highest classification success accuracy with 83% accuracy and 0.45 loss rates, while VGG-19 had the lowest accuracy at 81% with 0.37 loss rates.	N/A
Identification and Classification of Skin Diseases using Deep Learning Techniques.	5 classes namely, Impetigo (Bacterial), Cellulitis (Bacterial), Sporotrichosis (Fungal), Ringworm (Fungal), and Healthy Skin.	5 class labels, images resized, labeled, normalized, balanced, and augmented using techniques like width shift, height shift, zooming, and vertical/horizontal flips.	The VGG16 architecture fine-tuned the Convolutional Neural Network (CNN) to include 21 layers, with 16 trainable layers, 13 convolutional layers, 5 max-pooling layers, and 3 fully connected layers, using smaller filters.	The developed model is tested with the testing set images that consisted of 97 images. The model achieved a test accuracy of 86% and an F1-score of 85%.	N/A

3.12. Conclusion

After extensive research, we based our algorithm selection for the project on several key factors outlined in previous papers. These factors included the attributes of the dataset and the performance metrics of various models.

Our dataset selection process prioritized experiments conducted with DenseNet-121. We chose this model based on its demonstrated capacity to capture intricate features and effectively manage complex image data. Notably, they consistently achieved high levels of accuracy, which was deemed satisfactory for our project's objectives.

We chose DenseNet-121 because it showed high efficiency with an accuracy of up to 99% in the process of classifying between malignant and benign skin diseases. Therefore, we decided to use it to detect more types of skin diseases and not just classify between two types (malignant and benign).

Proposed Model

4.1. Introduction

The proposed model for skin disease detection is based on a CNN architecture designed to classify various types of skin diseases from dermatoscopic images. The motivation behind choosing CNN is its proven effectiveness in image recognition tasks. We try to achieve more inspiring results and contribute to the development of skin disease classification applications using deep learning.

4.2. Dataset

Description of the Dataset:

The dataset used in this project is the ISIC 2019 dataset, a comprehensive collection of 25,352 dermatoscopic images categorized into eight different skin lesion classes. The dataset is a resource for developing and evaluating skin diseases detection algorithms and conducting dermatological research. It was published on December 17, 2018, by a group of researchers including Tschandl, Codella and others.

Dataset Composition:

The dataset is divided into three subsets:

Training Set: 21,512 images (84.8%)

Validation Set: 1,910 images (7.5%)

Test Set: 1,930 images (7.6%)

The images in the dataset vary in size, predominantly 600 × 450 pixels and 1024 × 1024 pixels and are stored in JPG format. These images were sourced from a variety of global healthcare providers, clinics, and research centers specializing in dermatology.

Focused Skin Diseases:

For the scope of this project, we focus on four specific skin diseases:

Skin Disease	Training Images	Validation Images	Test Images
Actinic Keratosis	944	76	75
Dermatofibroma	420	22	43
Melanocytic Nevus	4428	931	423
Vascular Lesion	347	27	24

Table 3 Skin Diseases That Have Been Worked On

Data Preprocessing and Augmentation:

Given the unbalanced distribution across labels, data augmentation techniques are crucial to enhance the variability and robustness of the model. The following augmentation techniques were applied to the training dataset and validation dataset

Augmentation Technique	Parameter
Rescaling	1.0/255
Rotation	rotation_range=40
Width Shift	width_shift_range=0.2
Height Shift	height_shift_range=0.2
Shear	shear_range=0.2
Zoom	zoom_range=0.2
Horizontal Flip	horizontal_flip=True
Fill Mode	fill_mode=nearest
Feature-wise Standard Normalization	featurewise_std_normalization=True

Table 4 Data Augmentations

Additionally, the dull razor technique was employed for hair removal to enhance image clarity and focus on the lesion area. All images were resized to 224 × 224 pixels to maintain uniformity and compatibility with the model input requirements.

4.3. Model Architecture

The model architecture consists of a base DenseNet121 model with custom convolutional, pooling, dropout, and fully connected layers tailored for skin disease classification. Batch Normalization and dropout layers were used for regularization and improved training performance.

The base model is DenseNet121, loaded with pre-trained ImageNet weights. It includes convolutional blocks with skip connections, facilitating feature reuse and gradient flow.

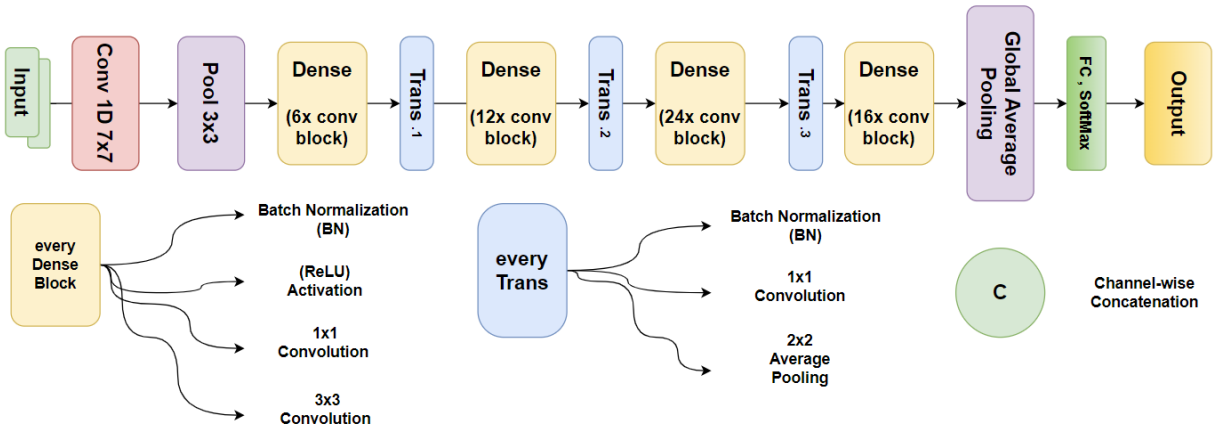


Fig 11 DenseNet-121 architecture

Custom layers were added on top of the base model to adapt it to the specific task of skin disease classification.

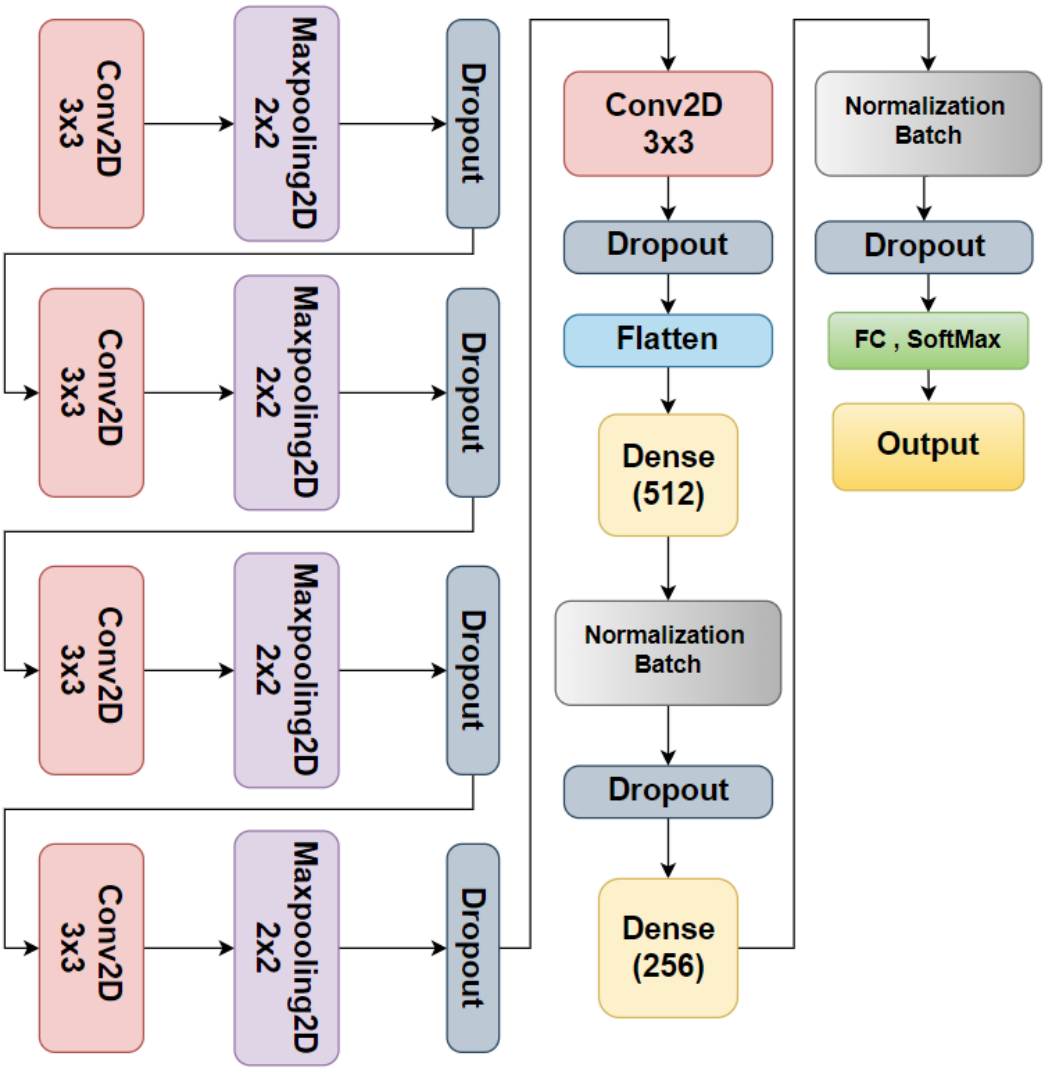


Fig12 Custom layers architecture

Convolutional Layers:
Layer 1: 256 filters, (3, 3) kernel size, ReLU activation, padding='same'
Layer 2: 128 filters, (3, 3) kernel size, ReLU activation, padding='same'
Layer 3: 64 filters, (3, 3) kernel size, ReLU activation, padding='same'
Layer 4: 32 filters, (3, 3) kernel size, ReLU activation, padding='same'

Pooling Layers:
Layer 1: Max pooling with (2, 2) pool size, padding='same'
Layer 2: Max pooling with (2, 2) pool size, padding='same'

Layer 3: Max pooling with (2, 2) pool size, padding='same'

Dropout Layers:
Dropout rate of 0.3 after Layer 1
Dropout rate of 0.2 after Layer 2 and Layer 3 and Layer 4

Flatten Layer:
Flatten the output of the convolutional layers to prepare for fully connected layers.

Fully Connected Layers:
Layer 1: 512 units, ReLU activation, with BatchNormalization and dropout rate of 0.5
Layer 2: 256 units, ReLU activation, with BatchNormalization and dropout rate of 0.3

Output Layer:
Dense layer with SoftMax activation for multi-class classification.

4.4. Training Procedure

The model was trained on a system with the following specifications:
RAM: 16 GB, GPU: NVIDIA GTX 1650 Ti, CPU: Intel Core i7-10750H, Framework: TensorFlow 2.10, Python Version: 3.10, CUDA Toolkit: 11.2, cudnn: 8.1.0, Package Manager: Anaconda 2.5.2

The hyperparameters used for training the model are as follows:
Epochs: 100, Batch Size: 100, Learning Rate: 0.001, Optimizer: Adam, Loss Function: Categorical Cross-Entropy

To evaluate the performance of the model, the following metrics were used:
Accuracy: Measures the overall correctness of the model's predictions.
Precision: Indicates the proportion of true positive results among all positive predictions.
Recall: Represents the proportion of true positive results among all actual positive cases.

F1-Score: The harmonic means of precision and recall, providing a balance between the two.

These metrics were chosen to provide a comprehensive evaluation of the model's performance, ensuring that both the accuracy and the quality of the predictions are considered.

Experimental Results

5.1. Main experiment

This experiment was carried out using densenet121, and Cubic SVM was combined with it, as mentioned in the proposed model chapter, and the evaluation matrix contains accuracy, precision, Recall, F1-Score

Accuracy measures the overall correctness of the model by comparing the number of correct predictions to the total number of predictions.

$$\text{Accuracy} = \frac{Tp + Tn}{Tp + Tn + Fp + Fn}$$

Precision measures the proportion of positive identifications that were actually correct.

$$Precision = \frac{Tp}{Tp + Fp}$$

Recall measures the proportion of actual positives that were correctly identified.

$$Recall = \frac{Tp}{Tp + Fn}$$

Since Precision and Recall can be defined as above, substituting them into the F1-Score equation gives:

$$F1Score = \frac{Precision * Recall}{Precision + Recall}$$

The table below presents a comprehensive evaluation of our classification model's performance across four types of skin lesions: Actinic Keratosis, Dermatofibroma, Melanocytic Nevus, and Vascular Lesion. This table includes key metrics such as Precision, Recall, F1-Score, and Support, providing a detailed assessment of how well the model predicts each class.

	Precision	**Recall**	**F1-Score**	**Support**
Actinic keratosis	0.97	0.89	0.93	75
Dermatofibroma	0.85	0.95	0.9	43
Melanocytic nevus	0.99	0.98	0.98	423
Vascular lesion	0.8	1	0.89	24
Accuracy			**0.96**	565
Macro avg	0.9	0.96	0.93	565
Weighted avg	0.97	0.96	0.97	565

Table 5 Model Evaluation Metrics of densenet121

Overall, the model achieves an impressive accuracy of 0.96. The macro average and weighted average metrics also indicate strong and consistent performance across all classes, validating the robustness of our classification model. These metrics are crucial for understanding the strengths and potential areas for improvement in our model's predictions.

The integration of a **Cubic SVM** classifier with DenseNet121 further enhanced the accuracy, achieving a test accuracy of **0.99** in the best experimental setup.

The below image represents the confusion matrix for our classification model, which was trained to identify four different types of skin lesions: Actinic Keratosis, Dermatofibroma, Melanocytic Nevus, and Vascular Lesion. A confusion matrix is a useful tool for evaluating the performance of a classification algorithm. It provides a detailed breakdown of correct and incorrect predictions, allowing us to see how well the model distinguishes between different classes.

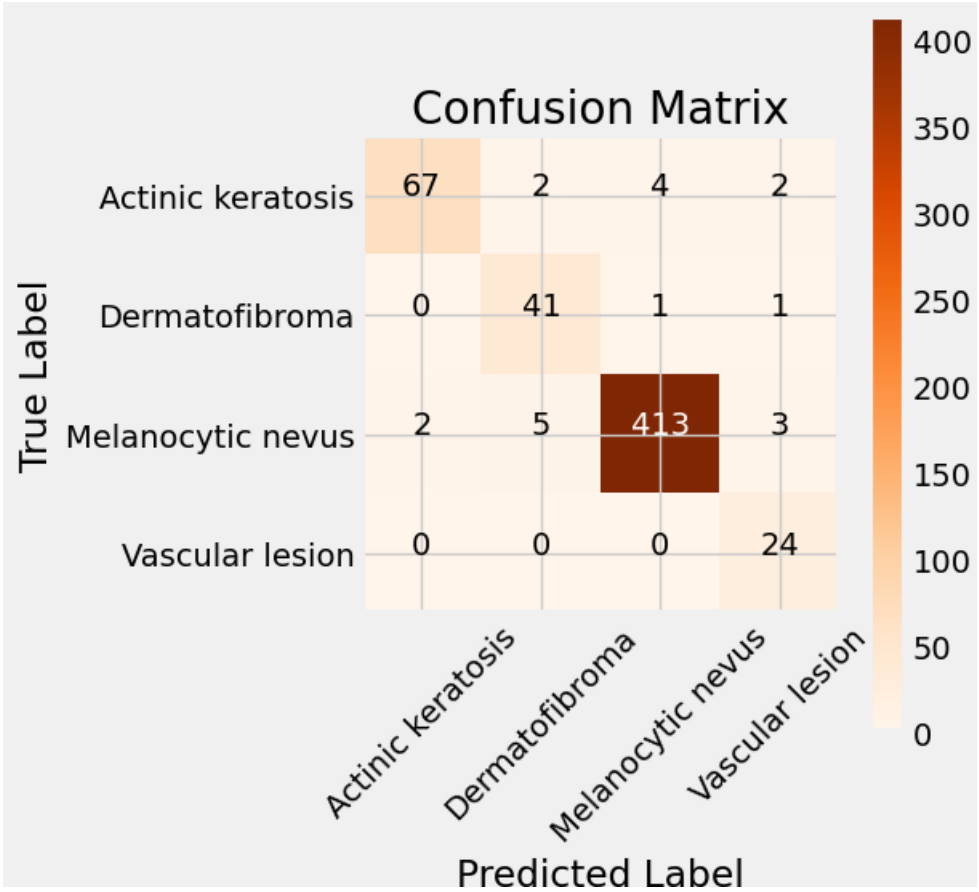


Fig 13 Confusion matrix of densenet121

5.2. Experiments comparisons

We have done many experiments where we adjust each hyperparameter independently just until we get the best result for the model containing that hyperparameter, and then we fixed the best result for that hyperparameter and change another hyperparameter until we get the best result and so on with all hyperparameters such as target size, unfreeze layers ,epochs, batch size, learning rate, dropout, early stopping, reduced learning rate, loss function, and optimizer. Therefore, we obtained very many results and experiments, so we presented a comparison between the best of these results in a way that shows the sequence in development.

Date	2-Apr	3-May	15-Apr using Cubic SVM method	30-Apr Cubic SVM method Under sampling
Epochs	10	15	100	100
Batch Size	32	64	100	100
Target Size	224,224	250,250	224,224	224,224
Reduce Learning Rate	No	Yes (factor=0.1, patience=15)	No	No
Unfreeze Layers	30 top layers	30 top layers	30 top layers	30 top layers
Early Stop	No	No	(monitor='loss', mode='min', patience=10)	(monitor='loss', mode='min', patience=10)
Dropout	0.3 | 0.5	0.5 | 0.5	0.5 | 0.3	0.5 | 0.3
Test Accuracy	0.86	0.914	0.959	**0.964**
Test Loss	0.717	0.422	0.137	0.127
Cubic SVM Test Accuracy	No	No	0.9	0.99
Cubic SVM Test Loss	No	No	0.09999	0.01

Table 6 Experiments Comparisons

Web development

6.1. Front-End Development

Our skin disease classification web application features a thoughtfully designed front-end that prioritizes user experience, information delivery, and responsiveness. This section provides an overview of the structure, technologies, libraries, and methodologies used in developing our front-end.

The website opens with a Header section that includes the website logo and a navigation bar with links to Home, Discover, App, and About Us sections. This layout ensures easy navigation and a cohesive user experience right from the start.

Following the header, the Intro Section engages users by introducing the application's purpose and value. This section aims to immediately capture the user's interest and provide a clear understanding of what the app offers and how it can benefit them.

Next, the Discover Dermatology section educates users about the various skin diseases our app can classify. It provides brief and informative descriptions of each condition, helping users gain a better understanding of their potential diagnoses and the importance of dermatological health.

The Consultation Section is designed to facilitate user interaction with dermatological experts. It provides a platform for users to seek more detailed information and professional advice about their skin conditions, enhancing the overall value of the application.

In the App Section, users can upload images of their skin conditions via an upload button or drag-and-drop functionality. Once an image is uploaded, the app processes it to classify the disease, offering a diagnosis along with treatment advice. This section is the core functionality of our application, providing users with immediate and actionable insights into their skin health.

The About Us section provides information about the development team, highlighting the expertise and efforts behind the project. It gives users a glimpse into the people responsible for the app, fostering trust and transparency.

Following this is the Acknowledgment Section, where we express our gratitude to the engineering and medical supervisors who contributed to the project. This section is a testament to the collaborative efforts that made the application possible.

Finally, the Footer mirrors the header with the navigation bar and logo, ensuring consistency and easy access to other sections of the website. It helps users navigate the site seamlessly and ensures that important links are always within reach.

The website is built using React v18, a powerful framework that allows us to create a dynamic and responsive user interface. This ensures that the website performs optimally across various devices, from desktops to mobile phones.

To enhance functionality, we employed several libraries. Axios is used for making API requests to the backend, efficiently handling responses and errors. React-Dropzone provides a seamless drag-and-drop interface for image uploads, enhancing the user experience in the App section. React-Slideshow is utilized to create engaging and interactive slideshows, adding a visually appealing element to the site.

In terms of tools, we used ESLint for code linting, which helps maintain high code quality and consistency by identifying and correcting issues in JavaScript code. Vite was employed for module management, significantly improving the development experience by providing a fast and efficient build tool.

Our data flow is designed to ensure smooth interaction between the front-end and backend. Using Axios, the front-end fetches data from the backend API, handling responses and errors to provide a seamless user experience. State

management is another critical aspect, where the application state is efficiently updated to reflect changes, ensuring that the user interface remains dynamic and responsive.

Overall, our front-end development strategy focuses on creating a robust, user-friendly, and responsive web application. By leveraging modern technologies, libraries, and tools, we have developed an interface that not only meets the functional requirements but also provides an engaging and informative user experience.

6.2. Back-End Development

Our back-end development is focused on providing a robust and efficient server that handles image classification requests for skin diseases. The back-end is developed using Flask, a lightweight web framework for Python, specifically version 3.0.3. This framework allows us to create web servers and APIs with minimal setup and code, making it an ideal choice for our application.

We start by importing the necessary libraries. Flask is used for creating the web server, flask_cors for handling Cross-Origin Resource Sharing (CORS) to allow our front-end to communicate with the back-end, and TensorFlow for loading and using our pre-trained machine learning model. Additionally, we use the Keras API, which is part of TensorFlow, for image preprocessing and model predictions.

The Flask application is initialized, and CORS is enabled to allow requests from different origins. This setup is crucial for enabling seamless interaction between our front-end (which might be hosted on a different domain) and the back-end.

We then load our pre-trained DenseNet model, which is stored in the 'densenet_svc_model.h5' file. This model is designed to classify skin diseases based on the images provided by the users. The model was trained on a dataset of skin disease images and can predict one of several classes: Actinic keratosis, Dermatofibroma, Melanocytic nevus, and Vascular lesion.

To provide a simple interface for users and ensure that our server is working, we define a home route (/) that returns a greeting message when accessed. This route serves as a basic check to confirm that the server is running correctly.

The core functionality of our back-end lies in the (/upload) route, which is set up to handle POST requests. When a user uploads an image through this endpoint, the server first checks if the file is present in the request. If not, it returns an error message. Once a file is detected, the image data is read and processed. The image is loaded and resized to 224x224 pixels to match the input size expected by our model. It is then converted to an array, expanded in dimensions, and normalized by dividing pixel values by 255.

The processed image is fed into the DenseNet121 model to make a prediction. The model outputs a set of probabilities corresponding to each class, and the class with the highest probability is selected as the predicted class. This class is then mapped to its corresponding label from the class_labels list.

Finally, the server responds with a JSON object containing the predicted label. This response is sent back to the front-end, which can then display the result to the user.

Overall, our back-end leverages Flask to create a responsive and efficient server that integrates a machine learning model for classifying skin diseases, handling user uploads, and returning predictions in real-time.

6.3. Deployment

In this section, we outline the deployment process for our skin disease classification web application. The deployment strategy ensures that our application is accessible, scalable, and performs efficiently.

For deploying our application, we utilized DigitalOcean, a cloud infrastructure provider renowned for its simplicity and scalability. DigitalOcean offers virtual servers known as Droplets, which are highly

configurable and can be tailored to meet specific needs. For our project, we chose DigitalOcean's Basic Droplets, which provided us with 1 vCPU, 1GB RAM, and 35GB of storage. These resources are sufficient for our initial deployment and allow for easy scaling as the application's user base grows. The Droplets provided a stable and robust environment for running our application, ensuring that it performs efficiently and remains responsive under load.

To host our React application and manage HTTP requests efficiently, we employed Nginx as our web server. Nginx is known for its high performance, stability, and low resource consumption, making it an ideal choice for serving our application. It handles incoming traffic, serves static files, and proxies API requests to the Flask back-end seamlessly.

The deployment process began with setting up a new Droplet on DigitalOcean, selecting the Basic Droplet configuration to meet our requirements. After accessing the Droplet via SSH, we installed the necessary software packages, including Nginx, Python, and required Python libraries. This initial setup provided the foundational environment required to run both our front-end and back-end applications.

The React application was built using the build process, generating static files that were then moved to the appropriate directory for Nginx to serve. Concurrently, the Flask back-end application, along with the machine learning model, was deployed and configured to run as a service. This ensures that the back-end remains active and responsive to incoming API requests, providing the necessary functionality for our application.

Configuring Nginx to serve the React application involved setting it up to handle static files and proxy API requests to the Flask back-end.

By leveraging DigitalOcean's scalable infrastructure and Nginx's efficient web serving capabilities, we have deployed a robust and scalable web application. This setup not only ensures high availability and performance but also provides a foundation for future growth and enhancements.

Conclusion and Future work

7.1. Conclusion

In conclusion, we explored key skin disease datasets, including HAM10000, ISIC 2019, PAD-UFES-20, BCN20000, Derm7pt, ISIC 2017, DermNet, PH2, and Dermofit, and provided valuable insights into their structures and characteristics, and examine their limitations and proposed solutions aim to enhance the quality and reliability of these datasets, laying a foundation for their optimal utilization in decision-making processes and applications dependent on accurate data. Building on our exploration, our project will rely on the ISIC 2019 dataset.

In our analysis of deep learning models such as ResNet-152, ResNet101, Inception-v3, DenseNet-121, DenseNet-201, MobileNet v1, MobileNet v2 & LSTM, VGGNet, and VGG16, we have gained a comprehensive understanding of their effectiveness in skin disease classification. By delving into their origins, dataset usage, preprocessing techniques, structures, layers, model-building processes, and result metrics, we are equipped with valuable knowledge for implementing robust deep learning model. Building on our exploration we chose DenseNet-121 algorithm based on their demonstrated capacity to capture intricate features and effectively manage complex image data.

The proposed CNN-based model for skin disease detection has demonstrated significant effectiveness in classifying various skin lesion types from dermatoscopic images. Leveraging the ISIC 2019 dataset, the model was trained and evaluated using a robust set of metrics, including accuracy, precision, recall, and F1-score, across four specific skin diseases: Actinic Keratosis, Dermatofibroma, Melanocytic Nevus, and Vascular Lesion.

Key highlights from our experimental results include:

- The model achieved a high overall accuracy of 0.96, indicating its strong capability in correctly classifying the skin lesion images.

- High precision and recall metrics across all classes, with particularly notable performance for Melanocytic Nevus (Precision: 0.99, Recall: 0.98, F1-Score: 0.98), underscoring the model's reliability in identifying the most common lesion in the dataset.
- Consistent macro and weighted average metrics validate the model's robustness and balanced performance across different classes, highlighting its potential for real-world dermatological applications.

To enhance the model's performance, extensive data preprocessing and augmentation techniques were employed to address the imbalance in the dataset and improve the model's generalization capabilities. The application of techniques such as rescaling, rotation, width and height shifts, shear, zoom, horizontal flip, and feature-wise standard normalization contributed to the model's high accuracy and robustness.

Furthermore, the model's architecture, based on DenseNet121 with custom convolutional, pooling, dropout, and fully connected layers, was carefully designed to optimize feature extraction and classification performance. The inclusion of pre-trained ImageNet weights and the strategic addition of batch normalization and dropout layers facilitated efficient training and regularization.

Experimental comparisons revealed that fine-tuning hyperparameters such as epochs, target size, learning rate, dropout rates, and early stopping criteria significantly impacted the model's performance. The integration of a Cubic SVM classifier further enhanced the accuracy, achieving a test accuracy of 0.99 in the best experimental setup.

In summary, the proposed model exhibits excellent potential for accurate skin disease classification, paving the way for advanced dermatological diagnostic tools. The thorough evaluation and iterative optimization underscore the model's capability and readiness for practical deployment in clinical settings. Future work will focus on expanding the dataset to include

more diverse skin lesions and exploring additional augmentation and optimization techniques to further improve model performance.

7.2. Future work

In the next phases of development for our skin disease classification web application, we aim to enhance the platform's capabilities and extend its reach to a broader audience. Here are some key areas we plan to focus on:

Expanding Service Reach: We aim to increase the number of dermatologists available on our platform and expand the service to new regions. By doing so, we can provide more comprehensive coverage and ensure that patients from diverse locations have access to expert dermatological care. This expansion will involve targeted outreach and partnerships with medical institutions to onboard qualified dermatologists.

User Profile Management: To improve user experience and ensure continuity of care, we plan to introduce user profiles. This feature will allow users to keep track of their personal data, medical history, and previous diagnoses. By maintaining detailed medical records, users and doctors can access a patient's history seamlessly, facilitating more accurate diagnoses and personalized treatment plans.

Categorize Users Groups: We will enhance our platform by creating distinct categories for users, specifically dividing membership into patients and doctors. This differentiation will enable us to tailor the user interface and functionalities to meet the specific needs of each group. For patients, this includes simplified appointment scheduling and access to their medical history, while for doctors, it includes tools for patient management and collaboration with peers.

Research and Publication: Given our achievement of an accuracy rate of up to 0.99 in skin disease classification, we will prepare a comprehensive research paper detailing our methodologies and findings. This paper will contribute to the academic and medical communities by presenting new results in the field of dermatological diagnostics. Publishing our research will

not only validate our work but also provide a foundation for future advancements and collaborations in medical AI.

By focusing on these areas, we aim to not only improve our current service but also contribute meaningfully to the field of dermatology through innovation and research. Our commitment to enhancing user experience, expanding access to care, and advancing medical knowledge underpins our future development goals.

References

[1] Asiedu, K. B. Retrieved, WHO's inaugural global meeting on Neglected Tropical Diseases affecting the skin urges intensified efforts to address their impact, World Health Organization. (2023, March 31), https://www.who.int/news/item/31-03-2023-who-first-global-meeting-on-skin-ntds-calls-for-greater-efforts-to-address-their-burden

[2] Tschandl, P., Rosendahl, C., & Kittler, H. (2018). The HAM10000 dataset, a large collection of multi-sources dermatoscopic images of common pigmented skin lesions, Scientific Data, (2018) https://doi.org/10.1038/sdata.2018.161

[3] Skin Cancer MNIST: HAM10000, Kaggle, (2018, September 20). https://www.kaggle.com/datasets/kmader/skin-cancer-mnist-ham10000

[4] International Skin Imaging Collaboration. (n.d.), ISIC, https://www.isic-archive.com/

[5] Pacheco, A. G. C., Lima, G. R., Salomão, A. S., Krohling, B. A., Biral, I. P., De Angelo, G. G., Alves, F. C. R., Esgario, J. G. M., Simora, A. C., Castro, P. B. C., Rodrigues, F. B., Frasson, P., Krohling, R. A., Knidel, H., Santos, M. C. S., Santo, R. B. D. E., Macedo, T. L. S. G., Canuto, T. R. P., & De Barros, L. F. S. PAD-UFES-20: A skin lesion dataset composed of patient data and clinical images collected from smartphones. Data in Brief, 32, 106221,(2020), https://doi.org/10.1016/j.dib.2020.106221

[6] Combalia, M. BCN20000: Dermoscopic lesions in the wild, arXiv.org, (2019, August 6), https://arxiv.org/abs/1908.02288

[7] J. Kawahara, S. Daneshvar, G. Argenziano and G. Hamarneh, "Seven-Point Checklist and Skin Lesion Classification Using Multitask Multimodal Neural Nets," in *IEEE Journal of Biomedical and Health Informatics*, vol. 23, no. 2, pp. 538-546, March 2019, https://doi.org/10.1109/JBHI.2018.2824327.

[8] Codella, N. C. F., Gutman, D., Celebi, M. E., Helba, B., Marchetti, M. A., Dusza, S. W., Kalloo, A., Liopyris, K., Mishra, N., Kittler, H., &

Halpern, A. Skin Lesion Analysis Toward Melanoma Detection: A Challenge at the 2017 International Symposium on Biomedical Imaging (ISBI), Hosted by the International Skin Imaging Collaboration (ISIC), ArXiv:1710.05006 [Cs], (2018), https://arxiv.org/abs/1710.05006

[9] Dr Ian Coulson. DermNet. (2021). https://dermnetnz.org/

[10] Mendonca, T., Ferreira, P. M., Marques, J. S., Marcal, A. R. S., & Rozeira, J. PH2 - A dermoscopic image database for research and benchmarking. 2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), (2013), https://doi.org/10.1109/embc.2013.6610779

[11] ADDI - Automatic computer-based Diagnosis system for Dermoscopy Images, (2013), https://www.fc.up.pt/addi/ph2%20database.html

[12] Rees, Aldridge, Fisher, Ballerini, A Color and Texture Based Hierarchical K-NN Approach to the Classification of Non-melanoma Skin Lesions, Color Medical Image Analysis, Lecture Notes in Computational Vision and Biomechanics 6 (M. E. Celebi, G. Schaefer (eds.)), (2013) https://licensing.edinburgh-innovations.ed.ac.uk/product/dermofitimage-library

[13] Mendes, D. B. Skin lesions classification using convolutional neural networks in clinical images. arXiv.org, (2018, December 6), https://arxiv.org/abs/1812.02316

[14] Demir, A., Yilmaz, F., & Kose, O. Early detection of skin cancer using deep learning architectures: resnet-101 and inception-v3, TIPTEKNO 2019, https://doi.org/10.1109/tiptekno47231.2019.8972045

[15] Al-Saedi, D. K. A., & Savaş, S. Classification of Skin Cancer with Deep Transfer Learning Method, Computer Science, (2022), https://doi.org/10.53070/bbd.1172782

[16] Rehman, M. Z. U., Ahmed, F., Alsuhibany, S. A., Jamal, S. S., Ali, M. Z., & Ahmad, J. Classification of skin cancer lesions using explainable deep learning, Sensors, 22(18), 6915, (2022), https://doi.org/10.3390/s22186915

[17] Chaturvedi, S. S., Gupta, K., & Prasad, P. S. Skin lesion analyzer: An efficient Seven-Way multi-class skin cancer classification using MobileNet, In Advances in intelligent systems and computing (pp. 165–176),(2020), https://doi.org/10.1007/978-981-15-3383-9_15

[18] Srinivasu, P. N., SivaSai, J. G., Ijaz, M. F., Bhoi, A. K., Kim, W., & Kang, J. J. Classification of Skin Disease Using Deep Learning Neural Networks with MobileNet V2 and LSTM, Sensors, 21(8), 2852, (2021), https://doi.org/10.3390/s21082852.

[19] Kaya, V., & Akgül, İ. Classification of skin cancer using VGGNet model structures. Gümüşhane Üniversitesi Fen Bilimleri Enstitüsü Dergisi, (2022), https://doi.org/10.17714/gumusfenbil.1069894

[20] Pai, V. R., Pai, S. G., Suhasi, P. M., & Rekha, P. M. Identification and Classification of Skin Diseases using Deep Learning Techniques, Research Square (Research Square), (2023), https://doi.org/10.21203/rs.3.rs-2628782/v1.

Appendices

9.1. Code

9.1.1. Front-End code

Index.html

```html
<!doctype html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <link rel="icon" type="image/svg+xml" href="/vite.svg" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Vite + React</title>
  </head>
  <body>
    <div id="root"></div>
    <script type="module" src="/src/main.jsx"></script>
  </body>
</html>
```

Main.jsx

```jsx
import React from 'react'
import ReactDOM from 'react-dom/client'
import App from './App.jsx'
import './index.css'

ReactDOM.createRoot(document.getElementById('root')).render(
  <React.StrictMode>
    <App />
  </React.StrictMode>,
)
```

App.jsx

src > App.jsx > App

```jsx
import Header from "./components/Header";
import styles from "./App.module.css";
import HomeSection from "./components/HomeSection";
import DiseasesSection from "./components/DiseasesSection";
import ConsultSection from "./components/ConsultSection";
import AppSection from "./components/AppSection";
import Main from "./components/Main";
import AboutUsSection from "./components/AboutUsSection";
import Acknowledge from "./components/Acknowledge";
import { useRef } from "react";
import Footer from "./components/Footer";
function App() {
  const home = useRef(null);
  const discover = useRef(null);
  const app = useRef(null);
  const aboutUs = useRef(null);

  return (
    <div className={styles.container}>
      <Header appRef={app} aboutUsRef={aboutUs} discoverRef={discover} />
      <Main>
        <>
          <HomeSection homeRef={home} />
          <DiseasesSection discoverRef={discover} />
          <ConsultSection />
          <AppSection appRef={app} />
          <AboutUsSection aboutUsRef={aboutUs} />
          <Acknowledge />
          <Footer
            homeRef={home}
            appRef={app}
            aboutUsRef={aboutUs}
            discoverRef={discover}
          />
        </>
      </Main>
    </div>
  );
}

export default App;
```

Header.jsx

src > components > Header.jsx > Header

```jsx
import { useEffect, useState } from "react";
import BurgerBar from "./BurgerBar";
import styles from "./Header.module.css";
import Logo from "./Logo";
import PageNav from "./PageNav";
function Header({ appRef, aboutUsRef, discoverRef }) {
  const [isShowNav, setisShowNav] = useState(false);
  const [windowWidth, setWindowWidth] = useState(window.innerWidth);

  function handleShowNav() {
    setisShowNav((showNav) => !showNav);
  }
  function handleWindowResize() {
    setWindowWidth(window.innerWidth);
  }
  window.addEventListener("resize", handleWindowResize);
  useEffect(() => {
    window.addEventListener("resize", handleWindowResize);

    return () => {
      window.removeEventListener("resize", handleWindowResize);
    };
  }, []);
```

```jsx
  return (
    <header className={styles.header}>
      <Logo />

      {windowWidth > 900 ? (
        <div className={styles.navBar}>
          <PageNav
            appRef={appRef}
            aboutUsRef={aboutUsRef}
            discoverRef={discoverRef}
          />
        </div>
      ) : isShowNav ? (
        <PageNav
          setisShowNav={setisShowNav}
          windowWidth={windowWidth}
          isShowNav={isShowNav}
          appRef={appRef}
          aboutUsRef={aboutUsRef}
          discoverRef={discoverRef}
        />
      ) : (
        <PageNav
          setisShowNav={setisShowNav}
          windowWidth={windowWidth}
          isShowNav={isShowNav}
          appRef={appRef}
          aboutUsRef={aboutUsRef}
          discoverRef={discoverRef}
        />
      )}
      {windowWidth <= 900 && <BurgerBar onClick={handleShowNav} />}
    </header>
  );
}

export default Header;
```

Logo.jsx

src > components > Logo.jsx > ...

```jsx
import styles from "./Logo.module.css";
function Logo() {
  return <img className={styles.logo} src="logo.png" alt="logo" />;
}

export default Logo;
```

PageNav.jsx

src > components > PageNav.jsx > PageNav > handleScrollTo

```jsx
import styles from "./PageNav.module.css";

function PageNav({
  homeRef,
  appRef,
  aboutUsRef,
  discoverRef,
  isShowNav,
  setisShowNav,
  windowWidth,
}) {
  function handleScrollTo(elemRef) {
    window.scrollTo({
      top: elemRef.current.offsetTop,
      behavior: "smooth",
    });
    if (windowWidth <= 900) {
      setisShowNav(false);
    }
  }
  return (
    <ul
      className={
        isShowNav && windowWidth <= 900
          ? styles.pageNavSmall
          : !isShowNav && windowWidth <= 900
          ? styles.pageNavHiddenSmall
          : styles.pageNav
      }
    >
      <li onClick={() => handleScrollTo(homeRef)}>Home</li>
      <li onClick={() => handleScrollTo(discoverRef)}>Discover</li>
      <li onClick={() => handleScrollTo(appRef)}>App</li>
      <li onClick={() => handleScrollTo(aboutUsRef)}>About us</li>
    </ul>
  );
}

export default PageNav;
```

BurgerBar.jsx

src > components > BurgerBar.jsx > BurgerBar

```jsx
import styles from "./BurgerBar.module.css";
function BurgerBar({ onClick }) {
  return (
    <div onClick={onClick} className={styles.burgerBar}>
      <div></div>
      <div></div>
      <div></div>
    </div>
  );
}

export default BurgerBar;
```

HomeSection.jsx

src > components > HomeSection.jsx > HomeSection

```jsx
import styles from "./HomeSection.module.css";
function HomeSection({ homeRef }) {
  return (
    <section className={styles.homeSection} ref={homeRef}>
      <div>
        <h1>
          Caring for you &<br />
          your family
        </h1>
        <p>
          Our target help you determine
          <br />
          the stage of your skin disease
        </p>
        <button>TRY NOW</button>
      </div>
      <div>
        <img src="homeImg.png" alt="homeImg" />
      </div>
    </section>
  );
}

export default HomeSection;

```

DiseasesSection.jsx

src > components > DiseasesSection.jsx > DISEASES

```jsx
import styles from "./DiseasesSection.module.css";
import { Slide } from "react-slideshow-image";
import "react-slideshow-image/dist/styles.css";
const DISEASES = [
  {
    name: "Melanocytic",
    info: "Melanocytic lesions are common and can be either benign or malignant. They arise from melanocytes, which are pigment-producing cells.",
    imgSrc: "diseases/melanocity.png",
  },
  {
    name: "Actinic keratosis",
    info: "Actinic keratosis, also known as solar keratosis, is a precancerous skin condition caused by prolonged exposure to ultraviolet (UV) radiation from the sun.",
    imgSrc: "diseases/actinic.png",
  },
  {
    name: "Dermatofibroma",
    info: "Dermatofibroma is a benign skin lesion that usually appears as a firm nodule or bump on the skin. It is composed of fibrous tissue and often occurs on the legs.",
    imgSrc: "diseases/dermatofibroma.png",
  },
  {
    name: "Vascular lesion",
    info: "Vascular lesions are abnormalities of blood vessels in the skin. They can appear as birthmarks, hemangiomas, or other vascular growths.",
    imgSrc: "diseases/vascular.png",
  },
];

function DiseasesSection({ discoverRef }) {
  return (
    <section ref={discoverRef} className={styles.diseasesSection}>
      <h1>Discover Dermatology</h1>
      <div>
        <Slide
          slidesToScroll={1}
          slidesToShow={1}
          indicators={true}
          autoplay={true}
        >
          {DISEASES.map((disease, i) => (
            <div className={styles.card} key={i}>
              <div>
                <img src={disease?.imgSrc} alt="homeImg.png" />
              </div>
              <div>
                <h2>{disease.name}</h2>
                <p>{disease.info}</p>
              </div>
            </div>
          ))}
        </Slide>
      </div>
    </section>
  );
}

export default DiseasesSection;
```

ConsultSection.jsx

src > components > ConsultSection.jsx > ConsultSection

```jsx
import styles from "./ConsultSection.module.css";
function ConsultSection() {
  return (
    <section className={styles.consultSection}>
      <p>{"Don't Wait For Tomorrow "}</p>
      <h1>We Welcome Your Questions & Comments</h1>
      <button>Get A Free Consultation</button>
    </section>
  );
}

export default ConsultSection;

```

AppSection.jsx

src > components > AppSection.jsx > AppSection > setDiseasesName

```jsx
import styles from "./AppSection.module.css";
import successStyles from "./SuccessSection.module.css";
import { useState } from "react";
//import { useRef} from "react";
import { useDropzone } from "react-dropzone";
import axios from "axios";
import Loading from "./Loading";
import ResultSection from "./ResultSection";

const DISEASES = [
  {
    name: "Melanocytic nevus",
    info: [
      "Sun Protection: Use sunscreen, wear protective clothing, and avoid tanning beds.",
      "Know Risk Factors: Understand your melanoma risk factors.",
      "Monitor Changes: Watch for mole changes like itching or rapid growth.",
      "Healthy Lifestyle: Maintain a healthy diet and lifestyle.",
      "Educate Yourself: Stay informed about skin health.",
      "Consult Dermatologist: Schedule regular skin checks.",
    ],
  },
  {
    name: "Actinic keratosis",
    info: [
      "Sun Protection: Use sunscreen, wear protective clothing, and avoid tanning.",
      "Regular Skin Checks: Monitor skin for new growths or changes in existing spots.",
      "Dermatologist Visits: Schedule regular skin checks with a dermatologist.",
      "Moisturize: Keep skin hydrated and moisturized.",
      "Quit Smoking: If applicable, quit smoking to reduce skin damage.",
      "Protective Clothing: Wear long sleeves, pants, and hats outdoors.",
      "Educate Yourself: Learn about AK and its risk factors.",
      "Healthy Lifestyle: Maintain a balanced diet, exercise, and manage stress.",
      "Stay Hydrated: Drink plenty of water for overall skin health.",
    ],
  },
```

```
  {
    name: "Dermatofibroma",
    info: [
      "Monitor Skin Changes: Regularly check for changes in dermatofibromas.",
      "Avoid Irritation: Refrain from scratching or picking at dermatofibromas.",
      "Protective Clothing: Wear protective clothing and sunscreen.",
      "Consult a Dermatologist: Seek medical advice for painful or growing dermatofibromas.",
      "Educate Yourself: Learn about dermatofibromas and treatment options.",
      "Moisturize: Keep skin moisturized to reduce dryness.",
    ],
  },
  {
    name: "Vascular lesion",
    info: [
      "Monitor Changes: Regularly check for changes in vascular lesions.",
      "Protect from Sun: Shield lesions from direct sunlight with clothing and sunscreen.",
      "Consult Dermatologist: Seek advice for new or changing lesions, especially if they bleed or itch.",
      "Avoid Trauma: Take precautions to prevent injury to areas with lesions.",
      "Educate Yourself: Learn about vascular lesions and treatment options.",
      "Moisturize: Keep skin moisturized to reduce dryness.",
      "Monitor for Symptoms: Be aware of symptoms near lesions and report them.",
      "Maintain Healthy Lifestyle: Follow a balanced diet, exercise, and manage stress.",
    ],
  },
];
```

```
function AppSection({ appRef }) {
  // const fileInputRef = useRef();
  const [loading, setLoading] = useState(false);
  const [uploadSuccess, setUploadSuccess] = useState(false);
  const [image, setImage] = useState(null);
  const [diseasesName, setDiseasesName] = useState(null);
  console.log(image);

  const onDrop = (acceptedFiles) => {
    const file = acceptedFiles[0];
    const formData = new FormData();
    formData.append("file", file);
    setLoading(true);
    setImage(URL.createObjectURL(file));

    axios
      .post("http://127.0.0.1:5000/upload", formData, {
        headers: {
          "Content-Type": "multipart/form-data",
        },
      })
      .then((response) => {
        console.log(response.data["prediction"]);
        setLoading(false);
        setUploadSuccess(true);
        setDiseasesName(response.data["prediction"]);
      })
      .catch((error) => {
        setLoading(false);
        console.error(error);
      });
  };

  const { getRootProps, getInputProps } = useDropzone({
    onDrop,
    multiple: false,
    accept: "image/*",
  });
```

```
105   const onFileChange = (event) => {
106     if (event.target.files.length > 0) {
107       setLoading(true);
108       onDrop([event.target.files[0]]);
109     }
110   };
111
112   return (
113     <section ref={appRef} className={styles.appSection}>
114       <h1>Check Your Self & Your Patients Easly</h1>
115       {loading ? (
116         <div>
117           <Loading />
118         </div>
119       ) : uploadSuccess ? (
120         <ResultSection
121           image={image}
122           diseasesName={diseasesName}
123           setUploadSuccess={setUploadSuccess}
124           setImage={setImage}
125           setDiseasesName={setDiseasesName}
126         />
127       ) : (
128         <div className={styles.uploadContainer} {...getRootProps()}>
129           <h1>
130             Upload an image
131             <br />
132             for disease detection
133           </h1>
134           <button type="button" /*onClick={onButtonClick}*/>
135             Upload Image
136           </button>
137           <input
138             {...getInputProps()}
139             /*ref={fileInputRef}*/
140             style={{ display: "none" }}
141             onChange={onFileChange}
142           />
143             <p>or drop a file</p>
144           </div>
145         )}
146       </section>
147     );
148   }
149
150   export default AppSection;
151
```

ResultSection.jsx

src > components > ResultSection.jsx > ResultSection

```jsx
import styles from "./ResultSection.module.css";
const DISEASES = [
  {
    name: "Melanocytic nevus",
    info: [
      "Sun Protection: Use sunscreen, wear protective clothing, and avoid tanning beds.",
      "Know Risk Factors: Understand your melanoma risk factors.",
      "Monitor Changes: Watch for mole changes like itching or rapid growth.",
      "Healthy Lifestyle: Maintain a healthy diet and lifestyle.",
      "Educate Yourself: Stay informed about skin health.",
      "Consult Dermatologist: Schedule regular skin checks.",
    ],
  },
  {
    name: "Actinic keratosis",
    info: [
      "Sun Protection: Use sunscreen, wear protective clothing, and avoid tanning.",
      "Regular Skin Checks: Monitor skin for new growths or changes in existing spots.",
      "Dermatologist Visits: Schedule regular skin checks with a dermatologist.",
      "Moisturize: Keep skin hydrated and moisturized.",
      "Quit Smoking: If applicable, quit smoking to reduce skin damage.",
      "Protective Clothing: Wear long sleeves, pants, and hats outdoors.",
      "Educate Yourself: Learn about AK and its risk factors.",
      "Healthy Lifestyle: Maintain a balanced diet, exercise, and manage stress.",
      "Stay Hydrated: Drink plenty of water for overall skin health.",
    ],
  },
  {
    name: "Dermatofibroma",
    info: [
      "Monitor Skin Changes: Regularly check for changes in dermatofibromas.",
      "Avoid Irritation: Refrain from scratching or picking at dermatofibromas.",
      "Protective Clothing: Wear protective clothing and sunscreen.",
      "Consult a Dermatologist: Seek medical advice for painful or growing dermatofibromas.",
      "Educate Yourself: Learn about dermatofibromas and treatment options.",
      "Moisturize: Keep skin moisturized to reduce dryness.",
    ],
  },
```

```
  {
    name: "Vascular lesion",
    info: [
      "Monitor Changes: Regularly check for changes in vascular lesions.",
      "Protect from Sun: Shield lesions from direct sunlight with clothing and sunscreen.",
      "Consult Dermatologist: Seek advice for new or changing lesions, especially if they bleed or itch.",
      "Avoid Trauma: Take precautions to prevent injury to areas with lesions.",
      "Educate Yourself: Learn about vascular lesions and treatment options.",
      "Moisturize: Keep skin moisturized to reduce dryness.",
      "Monitor for Symptoms: Be aware of symptoms near lesions and report them.",
      "Maintain Healthy Lifestyle: Follow a balanced diet, exercise, and manage stress.",
    ],
  },
];

function ResultSection({
  image,
  diseasesName,
  setUploadSuccess,
  setDiseasesName,
  setImage,
}) {
  function resetAnswer() {
    setUploadSuccess(false);
    setImage(null);
    setDiseasesName(null);
  }

  return (
    <section className={styles.resultSection}>
      <div className={styles.first}>
        {image && (
          <img
            src={image}
            alt="Uploaded"
            onLoad={() => console.log("Image loaded")}
          />
        )}
      </div>
```

```
        {diseasesName && (
          <div className={styles.second}>
            <h1>{diseasesName}</h1>
            <div>
              <h2>Some Advices</h2>
              <ol>
                {DISEASES.filter(
                  (item) => item.name === diseasesName
                )[0]?.info.map((item, i) => (
                  <li key={i}>{item}</li>
                ))}
              </ol>
            </div>
            <button onClick={resetAnswer}>Try Again</button>
          </div>
        )}
      </section>
    );
  }

  export default ResultSection;
```

AboutUsSection.jsx

src > components > AboutUsSection.jsx > AboutUsSection

```jsx
import styles from "./AboutUsSection.module.css";
import { Slide } from "react-slideshow-image";
import "react-slideshow-image/dist/styles.css";
const TEAM = [
  {
    name: "Mohamed Kamal",
    photo: "team/mk.png",
    avatar: "avatars/mk.png",
    jobTitle: "Front-End Developer",
    phones: ["01227300872"],
    socialMedia: {
      whatsApp: "https://wa.me/201227300872",
      linkedIn: "https://www.linkedin.com/in/mohamedkamal-in/",
    },
  },
  {
    name: "Banoub Nagy",
    photo: "team/bn.png",
    avatar: "avatars/bn.png",
    jobTitle: "AI Developer",
    phones: ["01024071289"],
    socialMedia: {
      whatsApp: "https://wa.me/201024071289",
      linkedIn: "https://www.linkedin.com/in/banoub-nagy-edwar-b33b15277/",
    },
  },
  {
    name: "Maryam Mohamed",
    photo: "team/mm.png",
    avatar: "avatars/mm.png",
    jobTitle: "AI Developer",
    phones: ["01006517272", "01210400751"],
    socialMedia: {
      whatsApp: "https://wa.me/201006517272",
      linkedIn: "https://www.linkedin.com/in/maryam-mohamedd/",
      facebook: "https://www.facebook.com/profile.php?id=100012562810123",
    },
  },
```

```
  {
    name: "Rowan Abdelkader",
    photo: "team/ra.png",
    avatar: "avatars/ra.png",
    jobTitle: "AI Developer",
    phones: ["01229671297", "01507166626"],
    socialMedia: {
      whatsApp: "https://wa.me/01229671297",
      linkedIn: "https://www.linkedin.com/in/rowan-abdelkader-877000244",
      facebook: "https://www.facebook.com/rowan.roro.3?mibextid=LQQJ4d",
    },
  },
  {
    name: "Sherif Mohamed",
    photo: "team/sm.png",
    avatar: "avatars/sm.png",
    jobTitle: "Back-End Developer",
    phones: ["01012569196"],
    socialMedia: {
      whatsApp: "https://wa.me/01012569196",
      linkedIn: "https://www.linkedin.com/in/sherif-mohamed-6a97141b8/",
    },
  },
  {
    name: "Raed Mohamed",
    photo: "team/rm.png",
    avatar: "avatars/rm.png",
    jobTitle: "Front-End Developer",
    phones: ["01277053851"],
    socialMedia: {
      whatsApp: "https://wa.me/01277053851",
      linkedIn: "https://www.linkedin.com/in/raed--mohamed/",
    },
  },
];
```

```jsx
function AboutUsSection({ aboutUsRef }) {
  return (
    <section ref={aboutUsRef} className={styles.aboutUsSection}>
      <h1>About us</h1>

      <Slide
        slidesToScroll={1}
        slidesToShow={1}
        indicators={true}
        autoplay={false}
      >
        {TEAM.map((teamMember, i) => (
          <div className={styles.card} key={i}>
            <div className={styles.first}>
              <img src={teamMember.photo} width="200px" alt="homeImg.png" />
            </div>
            <div className={styles.last}>
              <span className={styles.avatarNameContainer}>
                <img
                  src={teamMember?.avatar}
                  className={styles.avatar}
                  alt="avatar"
                />
                <h2>{teamMember.name}</h2>
              </span>
              <h3>{teamMember.jobTitle}</h3>
              <div className={styles.phones}>
                {teamMember.phones.map((phone, i) => (
                  <div key={i}>
                    <img width={40} src="phone.png" alt="phone-icon" />
                    <p>{phone}</p>
                  </div>
                ))}
              </div>
              <div className={styles.socialMedia}>
                {teamMember.socialMedia.linkedIn && (
                  <a target="_blank" href={teamMember.socialMedia?.linkedIn}>
                    <img width={35} src="linkedin.png" alt="linkedIn-icon" />
```

```
74  function AboutUsSection({ aboutUsRef }) {
85          {TEAM.map((teamMember, i) => (
110                       <a target="_blank" href={teamMember.socialMedia?.linkedIn}>
111                         <img width={35} src="linkedin.png" alt="linkedIn-icon" />
112                       </a>
113                     )}
114                     {teamMember.socialMedia.whatsApp && (
115                       <a target="_blank" href={teamMember.socialMedia?.whatsApp}>
116                         <img width={35} src="whatsapp.png" alt="whatsApp-icon" />
117                       </a>
118                     )}
119                     {teamMember.socialMedia.facebook && (
120                       <a target="_blank" href={teamMember.socialMedia?.facebook}>
121                         <img width={35} src="facebook.png" alt="facebook-icon" />
122                       </a>
123                     )}
124                     {teamMember.socialMedia.twitter && (
125                       <a target="_blank" href={teamMember.socialMedia?.twitter}>
126                         <img width={35} src="twitter.png" alt="twitter-icon" />
127                       </a>
128                     )}
129                   </div>
130                 </div>
131               </div>
132             ))}
133         </Slide>
134       </section>
135     );
136   }
137
138   export default AboutUsSection;
139
```

Acknowledge.jsx

src > components > Acknowledge.jsx > Acknowledge

```
1   import styles from "./Acknowledge.module.css";
2   function Acknowledge() {
3     return (
4       <section className={styles.acknowledgeSection}>
5         <h1>
6           Skin Diseases Detection
7           <br />
8           Graduation project supervised by Dr. Yasser & TA. Yassmen
9         </h1>
10      </section>
11    );
12  }
13
14  export default Acknowledge;
15
```

Footer.jsx

src > components > Footer.jsx > Footer

```jsx
import styles from "./Footer.module.css";
import Logo from "./Logo";
import PageNav from "./PageNav";
function Footer({ homeRef, appRef, aboutUsRef, discoverRef }) {
  return (
    <footer className={styles.footer}>
      <Logo />
      <div>
        <p>Navigation</p>
        <PageNav
          homeRef={homeRef}
          appRef={appRef}
          aboutUsRef={aboutUsRef}
          discoverRef={discoverRef}
        />
      </div>
    </footer>
  );
}

export default Footer;

```

9.1.2. Back-End code

```python
from flask import Flask, request, jsonify
from flask_cors import CORS
import tensorflow as tf
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import numpy as np
import os
import io

app = Flask(__name__)
CORS(app)
print(tf.__version__)
model = load_model('densenet_svc_model.h5')
class_labels = ['Actinic keratosis', 'Dermatofibroma', 'Melanocytic nevus', 'Vascular lesion']
```

```python
@app.route('/')
def home():
    return "Hello from the server!!"

@app.route('/upload', methods=['POST'])
def predict():
    if 'file' not in request.files:
        return jsonify({'error': 'No file'}), 400

    file = request.files['file']
    img_data = file.read()

    img = image.load_img(io.BytesIO(img_data), target_size=(224, 224))
    img = image.img_to_array(img)
    img = np.expand_dims(img, axis=0)
    img /= 255.0
    # print("####################################",img,"####################################")
    prediction = model.predict(img)
    # print("####################################",prediction,"####################################")
    predicted_class = np.argmax(prediction, axis=-1)
    label = class_labels[predicted_class[0]]
    print(label)
    return jsonify({'prediction': label})

if __name__ == '__main__':
    app.run(debug=True)
```

9.1.3. AI code

Skin disease detection - CNN pre-trained Model: Densenet121

`Project Type: Deep Learning Project (Skin Disease Classification)`

Loading Important Library

```
import itertools
import os
import numpy as np
import tensorflow as tf
import cv2
from tensorflow.keras import Model
from tensorflow.keras.preprocessing.image import ImageDataGenerator, array_to_img, img_to_array, load_img
from tensorflow.keras.applications import DenseNet121
from tensorflow.keras.layers import Conv2D, Dense, Dropout, BatchNormalization ,MaxPooling2D,Flatten
from tensorflow.keras.optimizers import Adam
import matplotlib.pyplot as plt
from tensorflow.keras.callbacks import EarlyStopping
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.svm import SVC
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
```

[1]

Directory Paths

- `train_data_dir`: Path to the directory containing training data.
- `test_data_dir`: Path to the directory containing test data.
- `val_data_dir`: Path to the directory containing validation data.

Parameters

- `epochs`: This parameter determines the number of times the entire dataset will be passed forward and backward through the neural network during the training process. Each pass constitutes one epoch.
- `batch_Train`: Specifies the number of training examples utilized in one iteration of training. It affects the speed and stability of training, as well as memory consumption.
- `batch_Test`: Similar to `batch_Train`, but specifies the number of test examples utilized in one iteration during evaluation. It affects the efficiency of model evaluation.
- `image_size`: Defines the dimensions (height and width) to which input images will be resized before being fed into the model. Ensuring uniform image size is often necessary for compatibility with neural network architectures.
- `num_classes`: the number of distinct classes or categories present in the classification task.
- `learning_rate`: Determines the step size at which the model parameters are updated during training. It influences the convergence and stability of the training process.

```
train_data_dir = "./train"
test_data_dir = "./test"
val_data_dir = "./val"

# Parameters
epochs = 50
batch_Train = 100
batch_Test = 100
image_size = (224, 224)
num_classes = 4
learning_rate = 0.001
```

Data Augmentation Parameters

- `datagen_args`: Parameters for data augmentation including rescaling, rotation, shifting, flipping, etc.

Data Augmentation Generators

- `train_datagen`, `val_datagen`: Generators for augmenting training and validation data respectively.
- `test_datagen`: Generator for rescaling test data.

Image Preprocessing Functions

- `apply_dull_razor`: Applies a dull razor effect to an image.
- `preprocess_image`: Preprocesses an image by applying dull razor effect.

```
# Define data augmentation parameters in one dictionary
datagen_args = dict(
    rescale=1.0/255,
    rotation_range=40,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest',
    featurewise_std_normalization=True
)

# Data augmentation for training and validation data
train_datagen = ImageDataGenerator(**datagen_args)
val_datagen = ImageDataGenerator(**datagen_args)

# Only rescaling for the test data
test_datagen = ImageDataGenerator(rescale=1.0/255.0)
```

```python
def apply_dull_razor(image):
    grayScale = cv2.cvtColor(image, cv2.COLOR_RGB2GRAY)
    kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (17, 17))
    blackhat = cv2.morphologyEx(grayScale, cv2.MORPH_BLACKHAT, kernel)
    bhg = cv2.GaussianBlur(blackhat, (5, 5), cv2.BORDER_DEFAULT)
    _, mask = cv2.threshold(bhg, 10, 255, cv2.THRESH_BINARY)
    inpainted_image = cv2.inpaint(image, mask, 1, cv2.INPAINT_TELEA)
    return inpainted_image

def preprocess_image(img):
    img = cv2.cvtColor(img, cv2.COLOR_RGB2BGR)  # Convert RGB to BGR for OpenCV processing
    img = apply_dull_razor(img)
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)  # Convert back to RGB
    return img

# Assuming these variables are defined: train_data_dir, image_size, batch_Train, test_data_dir, batch_Test, val_data_dir
sample_dir = train_data_dir
sample_image_files = [os.path.join(sample_dir, f) for f in os.listdir(sample_dir) if os.path.isfile(os.path.join(sample_dir, f))][:3]
```

```python
for img_path in sample_image_files:
    img = load_img(img_path)
    x = img_to_array(img)
    x = x.reshape((1,) + x.shape)

    print("Original Image:")
    plt.imshow(img)
    plt.axis('off')
    plt.show()

    # Apply augmentation to the image and display
    i = 0
    for batch in train_datagen.flow(x, batch_size=1):
        plt.figure(i)
        imgplot = plt.imshow(array_to_img(batch[0]))
        plt.axis('off')
        plt.show()
        i += 1
        if i % 3 == 0:  # Display three augmented versions
            break
```

```python
# Flow from directory for training, testing, and validation
train_generator = train_datagen.flow_from_directory(
    train_data_dir,
    target_size=image_size,
    batch_size=batch_Train,
    class_mode='categorical'
)

test_generator = test_datagen.flow_from_directory(
    test_data_dir,
    target_size=image_size,
    batch_size=batch_Test,
    class_mode='categorical'
)

val_generator = val_datagen.flow_from_directory(
    val_data_dir,
    target_size=image_size,
    batch_size=batch_Train,
    class_mode='categorical'
)
```

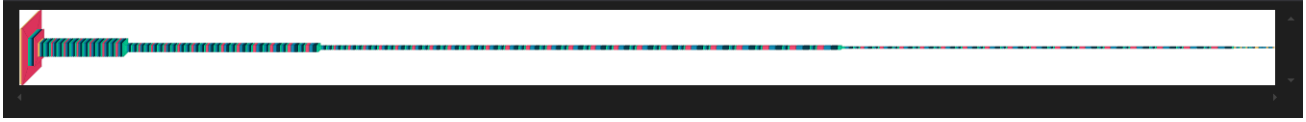
```python
import os
import matplotlib.pyplot as plt
import seaborn as sns

# Directory containing the training data
train_data_dir = "./train"

# Get the class names from the directory structure
class_names = [d.name for d in os.scandir(train_data_dir) if d.is_dir()]

# Count the number of images in each class
class_counts = {class_name: len(os.listdir(os.path.join(train_data_dir, class_name))) for class_name in class_names}

# Sort the classes for better readability
class_counts = dict(sorted(class_counts.items(), key=lambda item: item[1], reverse=True))

# Plot the distribution
plt.figure(figsize=(10, 8))
sns.barplot(x=list(class_counts.keys()), y=list(class_counts.values()), palette="viridis")

# Adding labels on top of the bars
for index, value in enumerate(class_counts.values()):
    plt.text(index, value + 50, str(value), ha='center', va='bottom', fontsize=12, fontweight='bold')

plt.xlabel('Class', fontsize=14)
plt.ylabel('Number of images', fontsize=14)
plt.title('Number of images per class in training data', fontsize=16)
plt.xticks(rotation=45, fontsize=12)
plt.yticks(fontsize=12)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()

plt.show()
```

Model Architecture of DenseNet121

```
# Load the DenseNet121 model with pre-trained ImageNet weights
base_model = DenseNet121(weights='imagenet', include_top=False, input_shape=(224, 224, 3))
inputs = base_model.output

x = Conv2D(256, (3, 3), activation='relu', padding='same')(inputs)
x = MaxPooling2D(pool_size=(2, 2), padding='same')(x)
x = Dropout(0.3)(x)

x = Conv2D(128, (3, 3), activation='relu', padding='same')(x)
x = MaxPooling2D(pool_size=(2, 2), padding='same')(x)
x = Dropout(0.2)(x)

x = Conv2D(64, (3, 3), activation='relu', padding='same')(x)
x = MaxPooling2D(pool_size=(2, 2), padding='same')(x)
x = Dropout(0.2)(x)
x = Conv2D(32, (3, 3), activation='relu', padding='same')(x)
# Removed the last max pooling layer to avoid dimensionality reduction beyond 1x1
x = Dropout(0.2)(x)

# Flatten the output of the convolutional layers
x = Flatten()(x)
# Fully-connected layers with BatchNormalization
x = Dense(512, activation='relu')(x)
x = BatchNormalization()(x)
x = Dropout(0.5)(x)
x = Dense(256, activation='relu')(x)
x = BatchNormalization()(x)
x = Dropout(0.3)(x)

# Output layer
predictions = Dense(num_classes, activation='softmax')(x)
# Define the model
model = Model(inputs=base_model.input, outputs=predictions)
# Compile the model
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model.summary()
```

```
import visualkeras
visualkeras.layered_view(model, to_file='RES.png')
```

Python



Freeze all layers in the base model initially, then unfreeze the top 10 layers for fine-tuning.

Compile the model with Adam optimizer and categorical crossentropy loss, using accuracy as the evaluation metric.

```
# Initially, freeze all layers
for layer in base_model.layers:
    layer.trainable = False

# Unfreeze the top 30 layers
for layer in base_model.layers[-30:]:
    layer.trainable = True


model.compile(optimizer=Adam(learning_rate=learning_rate), loss='categorical_crossentropy', metrics=['accuracy'])
```

[7]

```
callback_stop = tf.keras.callbacks.EarlyStopping(monitor='loss', mode='min', verbose=2,patience=10,min_delta = 0.001),
history = model.fit(train_generator, validation_data=val_generator, epochs=epochs,callbacks=[callback_stop])
```

[8]

Define a function to plot training and validation loss and accuracy over epochs.

```
def plot_training(hist):

    tr_acc = hist.history['accuracy']
    tr_loss = hist.history['loss']
    val_acc = hist.history['val_accuracy']
    val_loss = hist.history['val_loss']

    # Determine the best epochs
    index_loss = np.argmin(val_loss)
    val_lowest = val_loss[index_loss]
    index_acc = np.argmax(val_acc)
    acc_highest = val_acc[index_acc]

    # Set the plot's size and style
    plt.figure(figsize=(15, 6))
    plt.style.use('fivethirtyeight')

    # Create epoch array
    epochs = list(range(1, len(tr_acc) + 1))

    # Plot Training and Validation Loss
    plt.subplot(1, 2, 1)
    plt.plot(epochs, tr_loss, 'r-', label='Training Loss')
    plt.plot(epochs, val_loss, 'g--', label='Validation Loss')
    plt.scatter(index_loss + 1, val_lowest, s=150, c='blue', label=f'Best Epoch: {index_loss + 1}')
    plt.title('Training and Validation Loss')
    plt.xlabel('Epoch')
    plt.ylabel('Loss')
    plt.legend()
    plt.grid(True)
```

```python
    # Plot Training and Validation Accuracy
    plt.subplot(1, 2, 2)
    plt.plot(epochs, tr_acc, 'b-', label='Training Accuracy')
    plt.plot(epochs, val_acc, 'g--', label='Validation Accuracy')
    plt.scatter(index_acc + 1, acc_highest, s=150, c='red', label=f'Best Epoch: {index_acc + 1}')
    plt.title('Training and Validation Accuracy')
    plt.xlabel('Epoch')
    plt.ylabel('Accuracy')
    plt.legend()
    plt.grid(True)

    plt.tight_layout()
    plt.show()
```

```python
plot_training(history)
```

Define a test data generator with image rescaling and flow from a directory for image classification tasks.

```python
test_datagen = ImageDataGenerator(rescale=1./255)

test_generator = test_datagen.flow_from_directory(
    r'D:\Senior 4\Graduat_Project\semester 2\Project\isic original\test',
    target_size=(224, 224),
    batch_size=batch_Test,
    class_mode='categorical',
    shuffle=False
)
```

Python

```
Found 565 images belonging to 4 classes.
```

```python
class_labels = ["Actinic keratosis", "Dermatofibroma", "Melanocytic nevus", "Vascular lesion"]
```

Python

evaluation and analysis Results

```
test_loss, test_accuracy = model.evaluate(test_generator)
print("Test Accuracy:", test_accuracy)
print("Test Loss:", test_loss)

# Predict on the test data
predictions = model.predict(test_generator)
y_pred = np.argmax(predictions, axis=1)

# True labels
y_true = test_generator.classes

# Calculate the confusion matrix
conf_matrix = confusion_matrix(y_true, y_pred, labels=np.arange(len(class_labels)))

# Plotting the confusion matrix
plt.figure(figsize=(6, 6))
plt.imshow(conf_matrix, interpolation='nearest', cmap=plt.cm.Oranges)
plt.title('Confusion Matrix')
plt.colorbar()

tick_marks = np.arange(len(class_labels))
plt.xticks(tick_marks, class_labels, rotation=45)
plt.yticks(tick_marks, class_labels)

thresh = conf_matrix.max() / 2.
for i, j in itertools.product(range(conf_matrix.shape[0]), range(conf_matrix.shape[1])):
    plt.text(j, i, conf_matrix[i, j],
             horizontalalignment='center',
             color='white' if conf_matrix[i, j] > thresh else 'black')

plt.tight_layout()
plt.ylabel('True Label')
plt.xlabel('Predicted Label')
plt.show()
```

```
predictions = model.predict(test_generator)
y_pred = np.argmax(predictions, axis=1)

# True labels
y_true = test_generator.classes

# Generate and print classification report
report = classification_report(y_true, y_pred, target_names=test_generator.class_indices.keys())
print("Classification Report:\n", report)
```

```
6/6 [==============================] - 32s 6s/step - loss: 0.1391 - accuracy: 0.9646
Test Accuracy: 0.9646017551422119
Test Loss: 0.1391231119632721
6/6 [==============================] - 7s 686ms/step
```

Extract features from a model using a data generator.

```python
def extract_features(model, data_generator):
    features = []
    labels = []
    for inputs_batch, labels_batch in data_generator:
        features_batch = model.predict(inputs_batch)
        features.extend(features_batch)
        labels.extend(np.argmax(labels_batch, axis=1))  # Convert from one-hot to class labels
        if len(features) >= len(data_generator):
            break
    return np.array(features), np.array(labels)
```

Use extracted features to train a Support Vector Machine (SVM) classifier.

```python
test_features, test_labels = extract_features(model, test_generator)

svc_model = Pipeline([
    ('scaler', StandardScaler()),
    ('svc', SVC(kernel='poly', degree=3))
])
svc_model.fit(test_features, test_labels)
```

```python
# Optionally, you can save the trained SVC model
from joblib import dump
dump(svc_model, '3.joblib')
```

Load a pre-trained Support Vector Classifier (SVC) model, predict using extracted features, and evaluate performance.

```python
from joblib import load
import numpy as np

# Load the pre-trained SVC model from a joblib file
svc_model = load('3.joblib')

# Predict using the trained SVC
predictions_svc = svc_model.predict(test_features)

# Evaluate the predictions
accuracy_svc = np.mean(predictions_svc == test_labels)
misclassification_rate = 1 - accuracy_svc  # This is the proportion of incorrect predictions

print("Test Accuracy (SVC):", accuracy_svc)
print("Misclassification Rate (Loss):", misclassification_rate)
```

Python

```
Test Accuracy (SVC): 0.99
Misclassification Rate (Loss): 0.010000000000000009
```

```python
# Save the entire model (DenseNet121 + SVC) for later use
model.save("3.h5")
```

Python

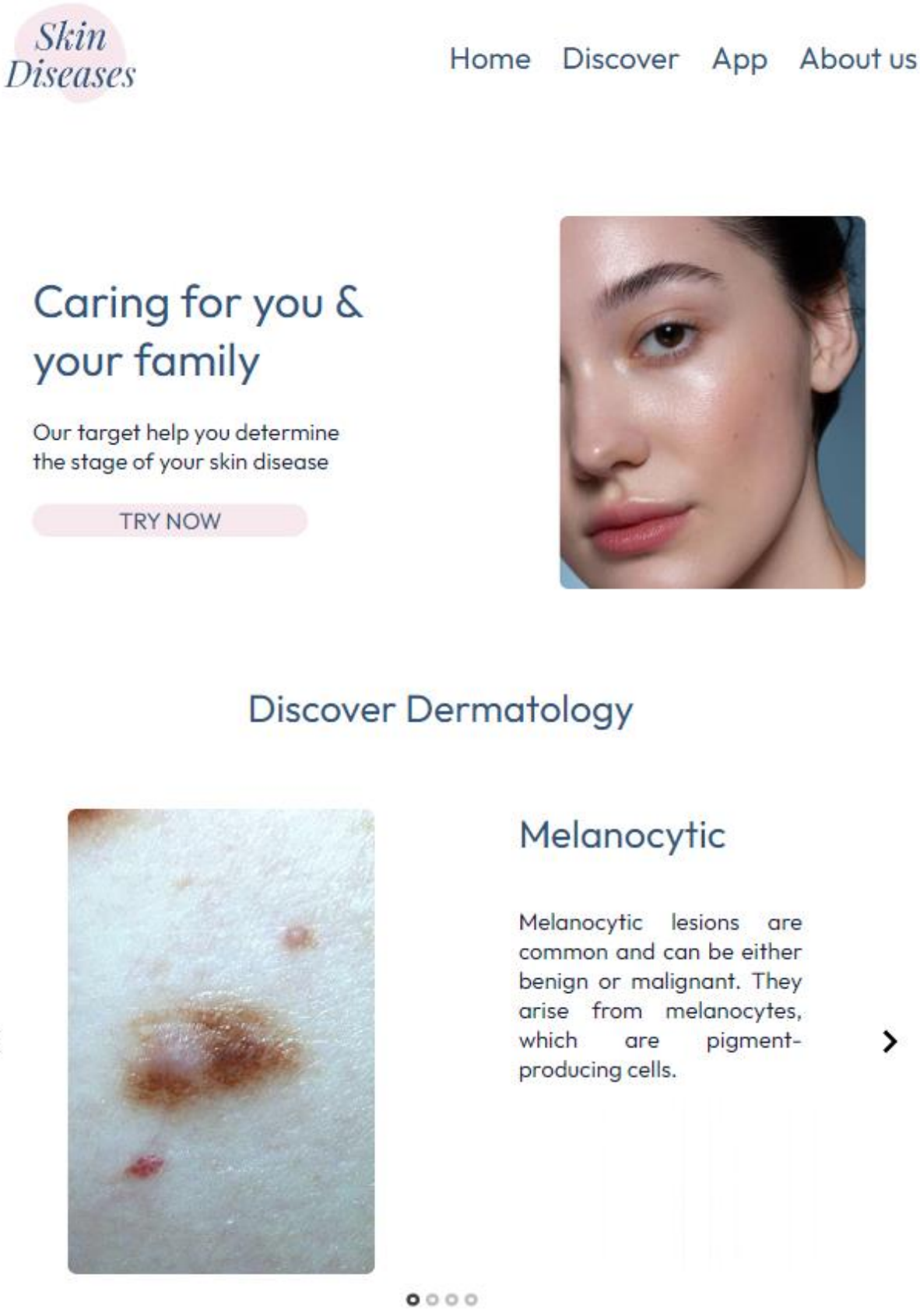
loading File.h5 and Make Prediction

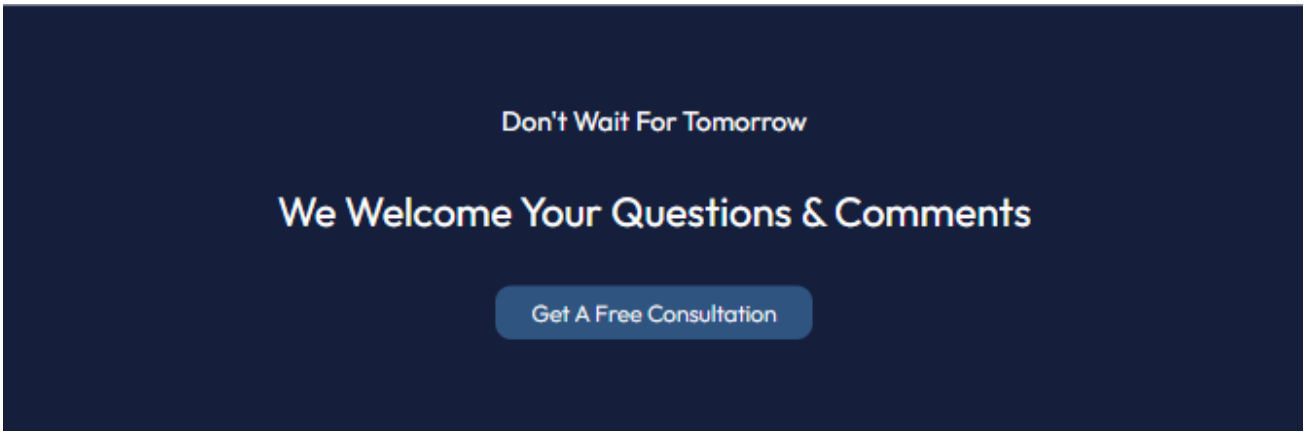
```
from tensorflow.keras.preprocessing import image


model_loaded = tf.keras.models.load_model('3.h5')

# Define a function to predict the class and probabilities of an image
def predict_image(model, img_path, class_names):
    # Load and preprocess the image
    img = image.load_img(img_path, target_size=(224, 224))
    img_array = image.img_to_array(img)
    img_array = np.expand_dims(img_array, axis=0)
    img_array /= 255.0  # Rescale pixel values to [0, 1]

    # Make prediction
    predictions = model.predict(img_array)

    # Get the predicted class and its probability
    predicted_class_index = np.argmax(predictions)
    predicted_class = class_names[predicted_class_index]
    probability = predictions[0, predicted_class_index]

    return img, predicted_class, probability

# Define the class names (you may need to adjust this based on your dataset)
class_names = ['Actinic keratosis', 'Dermatofibroma', 'Melanocytic nevus', 'Vascular lesion']

# Path to the image you want to predict
img_path = r'D:\Senior 4\Graduat_Project\semester 2\Project\isic original\test\Vascular lesion\ISIC_0070899.jpg'
# Make prediction
img, predicted_class, probability = predict_image(model_loaded, img_path, class_names)

# Display the predicted class and probabilities
print("Predicted Class:", predicted_class)
print("Probability:", probability)
```

```
  # Show the image with the prediction
  plt.figure(figsize=(4, 4))
  plt.imshow(img)
  plt.title(f'Predicted: {predicted_class} (Probability: {probability:.2f})')
  plt.axis('off')  # Turn off axis numbers and ticks
  plt.show()
```

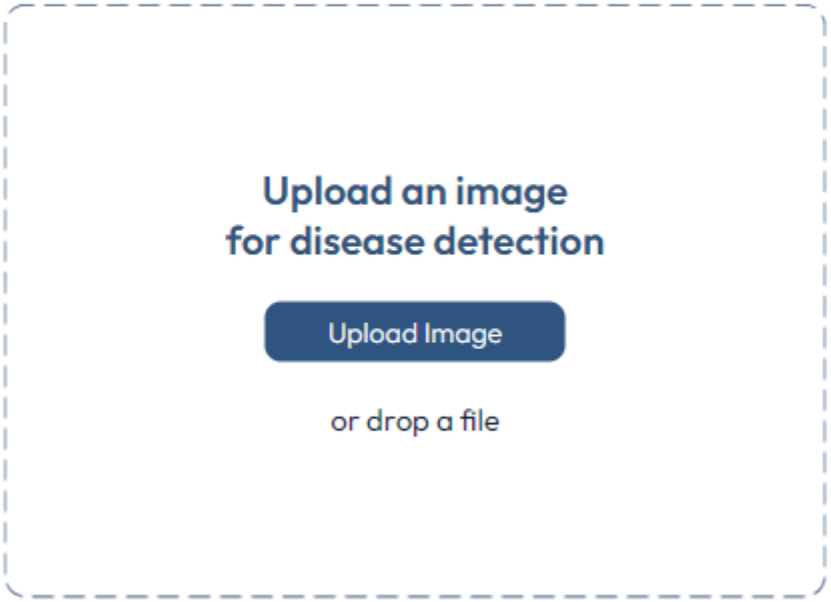
```
1/1 [==============================] - 3s 3s/step
Predicted Class: Vascular lesion
Probability: 0.99999976
```

9.2. Images


Skin Diseases
Home
Discover
App
About us
Caring for you & your family
Our target help you determine the stage of your skin disease
TRY NOW
Discover Dermatology
Melanocytic
Melanocytic lesions are common and can be either benign or malignant. They arise from melanocytes, which are pigment-producing cells.



Check Your Self & Your Patients Easly



Check Your Self & Your Patients Easly

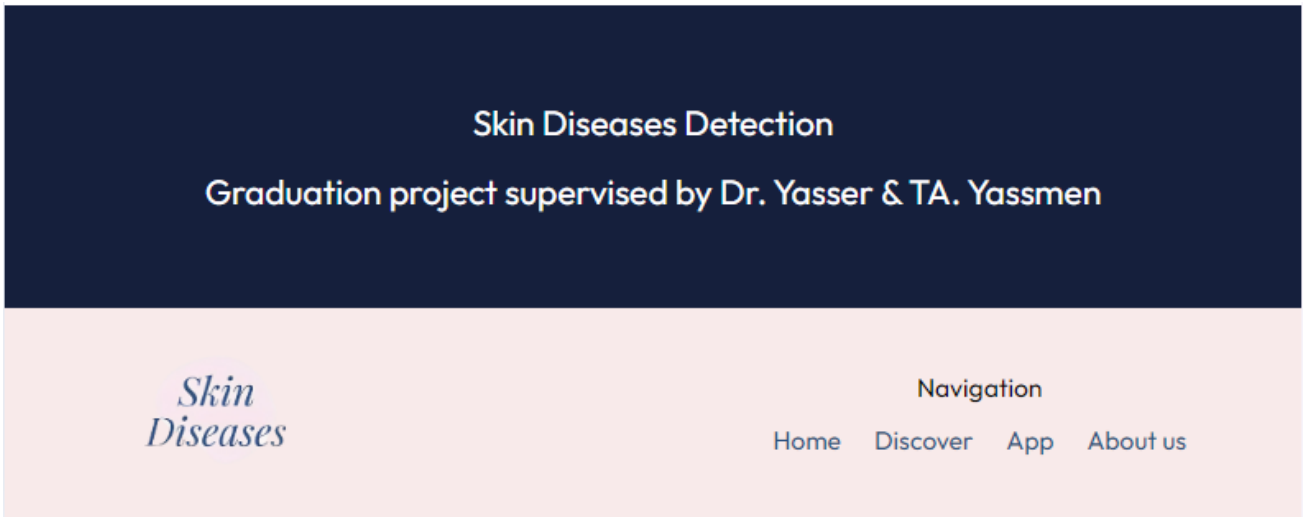
Vascular lesion

Some Advices

1. Monitor Changes: Regularly check for changes in vascular lesions.
2. Protect from Sun: Shield lesions from direct sunlight with clothing and sunscreen.
3. Consult Dermatologist: Seek advice for new or changing lesions, especially if they bleed or itch.
4. Avoid Trauma: Take precautions to prevent injury to areas with lesions.
5. Educate Yourself: Learn about vascular lesions and treatment options.
6. Moisturize: Keep skin moisturized to reduce dryness.
7. Monitor for Symptoms: Be aware of symptoms near lesions and report them.
8. Maintain Healthy Lifestyle: Follow a balanced diet, exercise, and manage stress.

Try Again

About us


Skin Diseases Detection
Graduation project supervised by Dr. Yasser & TA. Yassmen
Skin Diseases
Navigation
Home
Discover
App
About us