PROJECT REPORT

AI-BASED LOCALIZATION AND CLASSIFICATION OF SKIN DISEASE WITH SKIN ERYTHEMA

PROJECT ID	**:** PNT2022TMID03793
TEAM MEMBERS	**:** Dhaarani A Rajpriya R Arunpriya M Swetha S Sivaranjani D

INDEX

INTRODUCTION

1.1.Project Overview

Now a day's people are suffering from skin diseases, More than 125 million people suffering from Psoriasis also skin cancer rate is rapidly increasing over the last few decades especially Melanoma is most diversifying skin cancer. If skin diseases are not treated at an earlier stage, then it may lead to complications in the body including spreading of the infection from one individual to the other. To overcome the above problem we are building a model which is used for the prevention and early detection of skin cancer, psoriasis. Basically, skin disease diagnosis depends on the different characteristics like colour, shape, texture etc. Here the person can capture the images of skin and then the image will be sent the trained model. The model analyses the image and detect whether the person is having skin disease or not.

1.2.Purpose

The diseases are not considered skin diseases, and skin tone is majorly suffered from the ultraviolet rays from the sun. However, dermatologists perform the majority of non-invasive screening tests simply with the naked eye, even though skin illness is a frequent disease for which early detection and classification are essential for patient success and recovery. The characteristic of the skin images is diversified so that it is a challenging job to devise an efficient and robust algorithm for automatic detection of skin disease and its severity. Automatic processing of such images for skin analysis requires quantitative discriminator to differentiate the diseases.

LITERATURE SURVEY

2.1.Existing problem

Erythema is a broad category of skin condition that can impact any area of the skin and mucous membranes. It usually occurs in response to disease or infection in reaction to a drug. Severity of the rash ranges from mild to life threatening. It is an abnormal redness of skin or mucous membranes. Capillary congestion causes the condition, and red splotches on the hands or feet are classic examples of it.

2.2.References

PAPER 1: Deep Learning in Skin Disease Image Recognition: A Review

PUBLICATION YEAR: 11 November 2020

AUTHOR NAME: LING-FANG LI1 , XU WANG1 , WEI-JIAN HU 1 , NEAL N. XIONG 2 , (Senior Member, IEEE), YONG-XING DU 1 , AND BAO-SHAN LI1.

JOUNAL NAME: IEEE ACCESS (https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=6287639)

SUMMARY:

The application of deep learning methods to diagnose diseases has become a new research topic in the medical field. In the field of medicine, skin disease is one of the most common diseases, and its visual representation is more prominent compared with the other types of diseases. Accordingly, the use of deep learning methods for skin disease image recognition is of great significance and has attracted the attention of researchers. In this study, we review 45 research efforts on the identification of skin disease by using deep learning technology since 2016. We analyze these studies from the aspects of disease type, data set, data processing technology, data augmentation technology, model for skin disease image recognition, deep learning framework, evaluation indicators, and model performance. Moreover, we summarize the traditional and machine learning-based skin disease diagnosis and treatment methods. We also analyze the current progress in this field and predict four directions that may become the research topic in the future. Our results show that the skin disease image recognition method based on deep learning is better than those of dermatologists and other computer-aided treatment methods in skin disease diagnosis, especially the multi deep learning model fusion method has the best recognition effect.

CONCLUSION:

Forty-five relevant papers have been identified to obtain their concerns about the areas and skin disease types. These papers are utilized as basis to study the used data source, the preprocessing and data expansion techniques, the technical details of the models, and the performance indicators' overall performance. Deep learning models AlexNet, VGG, GoogleNet, and ResNet are widely used in skin disease recognition. Researchers often use the multimodel fusion technology to improve the performance of models. In future work, we plan to apply the concepts and best practices of deep learning described in this survey to other medical fields that have not fully utilized this technology. This survey aims to encourage many researchers to conduct deep learning experiments and apply the model of deep learning in the field of computer vision involving medicine, thereby achieving smart and convenient development for the medical industry.

PAPER 2: Progressive Transfer Learning and Adversarial Domain Adaptation for Cross-Domain Skin Disease Classification

PUBLICATION YEAR: MAY 2020

AUTHOR NAME: Yanyang Gu, Zongyuan Ge, Member, IEEE, C. Paul Bonnington, Senior Member, IEEE, and Jun Zhou, Senior Member, IEEE

JOURNAL NAME: IEEE JOURNAL OF BIOMEDICAL AND HEALTH INFOMATICS

SUMMARY:

Deep learning has been used to analyze and diagnose various skin diseases through medical imaging. However, recent researches show that a well trained deep learning model may not generalize well to data from different cohorts due to domain shift. Simple data fusion techniques such as combining disease samples from different data sources are not effective to solve this problem. In this paper, we present two methods for a novel task of cross-domain skin disease recognition. Starting from a fully supervised deep convolutional neural network classifier pre-trained on ImageNet, we explore a two-step progressive transfer learning technique by fine-tuning the network on two skin disease datasets. We then propose to adopt adversarial learning as a domain adaptation technique to perform invariant attribute translation from source to target domain in order to improve the recognition performance. In order to evaluate these two methods, we analyze generalization capability of the trained model on melanoma detection, cancer detection and cross-modality learning tasks on two skin image datasets collected from different clinical settings and cohorts with different disease distributions. The experiments prove the effectiveness of our method in solving the domain shift problem.

CONCLUSION:

In this work, they quantitatively validate the model generalization for different datasets from two perspectives. One is applying parameter-based progressive transfer learning to share transferable knowledge from task-different source domain and task-same but dataset-different intermediate domain with target domain. In the second method, we generalize the model for different datasets by integrating images from other datasets after translating with cycle consistent generative networks (Cycle-GAN). In this way, the model can be generalized for dataset-different domain as well as modality-different domain. Our experiments show the improvements for both overall multi-class classification accuracy and binary classification accuracy on both source domain datasets and target domain datasets. The improvement in binary classification is especially outstanding, which, in real cases, is more expected as the missing rate of melanoma shall be lowered. In the future, to further improve the classification performance, an algorithm can be developed that may contain discriminant features from both the training sets. The fusion could be developed by constructing a hybrid training parameter set from the two training set parameters which were extracted on individual data sets. Although this work applies domain adaptation to skin disease imaging dataset augmentation, we believe this scheme may inspire more studies on applications that lack training data, especially in general medical imaging applications.

PAPER 3: Self-Paced Balance Learning for Clinical Skin Disease Recognition

PUBLICATION YEAR: August 2020

AUTHOR NAME: Jufeng Yang, Xiaoping Wu, Jie Liang, Xiaoxiao Sun, Ming-Ming Cheng, Paul L. Rosin, and Liang Wang

JOURNAL NAME: IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS

SUMMARY:

Class imbalance is a challenging problem in many classification tasks. It induces biased classification results for minority classes that contain less training samples than others. Most existing approaches aim to remedy the imbalanced number of instances among categories by resampling the majority and minority classes accordingly. However, the imbalanced level of difficulty of recognizing different categories is also crucial, especially for distinguishing samples with many classes. For example, in the task of clinical skin disease recognition, several rare diseases have a small number of training samples, but they are easy to diagnose because of their distinct visual properties. On the other hand, some common skin diseases, e.g., eczema, are hard to recognize due to the lack of special symptoms. To address this problem, we propose a self-paced balance learning (SPBL) algorithm in this paper. Specifically, we introduce a comprehensive metric termed the complexity of image category that is a combination of both sample number and recognition difficulty. First, the complexity is initialized using the model of the first pace, where the pace indicates one iteration in the self-paced learning paradigm. We then assign each class a penalty weight that is larger for more complex categories and smaller for easier ones, after which the curriculum is reconstructed by rearranging the training samples. Consequently, the model can iteratively learn discriminative representations via balancing the complexity in each pace. Experimental results on the SD-198 and SD-260 benchmark data sets demonstrate that the proposed SPBL algorithm performs favorably against the state-of-the-art methods. We also demonstrate the effectiveness of the SPBL algorithm's generalization capacity on various tasks, such as indoor scene image recognition and object classification.

CONCLUSION:

In this paper, they address the class imbalance issue and propose a novel SPBL algorithm that is trained using samples from easy to hard. They also propose a novel insight that in real-world applications, the class imbalance problem is not only due to the imbalanced distribution of class sizes but also the imbalanced recognition difficulty. Inspired by that, we propose both the PWU and CR strategies that ensure that the model learns a comprehensively balanced representation in each SPL procedure. They conduct experiments on two imbalanced data sets about clinical skin disease recognition tasks and several other imbalanced problems. The results indicate that both components of the proposed algorithm are effective and demonstrate the advantage of the SPBL against the state-of-the-art methods.

PAPER 4: A Visually Interpretable Deep Learning Framework for Histopathological Image-Based Skin Cancer Diagnosis

PUBLICATION YEAR: MAY 2021

AUTHOR NAME: Shancheng Jiang, Huichuan Li, and Zhi Jin , Member, IEEE

JOURNAL NAME: IEEE JOURNAL OF BIOMEDICAL AND HEALTH INFORMATICS

SUMMARY:

Owing to the high incidence rate and the severe impact of skin cancer, the precise diagnosis of malignant skin tumors is a significant goal, especially considering treatment is normally effective if the tumor is detected early. Limited published histopathological image sets and the lack of an intuitive correspondence between the features of lesion areas and a certain type of skin cancer pose a challenge to the establishment of high?quality and interpretable computer-aided diagnostic (CAD) systems. To solve this problem, a light-weight attention mechanismbased deep learning framework, namely, DRANet, is proposed to differentiate 11 types of skin diseases based on a real histopathological image set collected by us during the last 10 years. The CAD system can output not only the name of a certain disease but also a visualized diagnostic report showing possible areas related to the disease. The experimental results demonstrate that the DRANet obtains significantly better performance than baseline models (i.e., InceptionV3, ResNet50, VGG16, and VGG19) with comparable parameter size and competitive accuracy with fewer model parameters. Visualized results produced by the hidden layers of the DRANet actually highlight part of the class?specific regions of diagnostic points and are valuable for decision making in the diagnosis of skin diseases.

CONCLUSION:

In this study, they propose a novel deep learning-based CAD system for skin cancer diagnosis. The output of this system is not only the label of a certain disease but also a visualized diagnostic report showing possible areas related to the disease, which can be considered as diagnostic points. The visualized diagnostic report is generated by the attention mechanism embedded at the upper part of our deep structure, and the whole framework is trained with only classification labels in an end-to-end way. The Spatial Attention-Oriented mask branch helps the Trunk branch to capture regions of interest related to the target label, and the Channel Attention-Oriented mask branch is able to model interdependencies between the channels of feature maps from the trunk branch in a computationally efficient way while enhancing the representational power of the trunk branch throughout every SEA module. To further enhance the model adaptability for different settings of the training batch size and the configuration of hardware, we introduce FRN layers for eliminating the dependence between samples or channels of the same sample. In the case study, all comparative experiments are conducted with a real histopathological image set that was collected and maintained by us during the last 10 years. Results validate the positive effects of the combination of two mask branches in the SEA module and the robustness of the FRN layers on a smallscale training set. Our DRANet achieves significantly better performance than baseline well-known image classification models with comparable parameter size and competitive accuracy with fewer parameters and smaller computational complexity. With light model size and relatively cheap computational costs, our DRANet can be deployed on common PCs or even a tiny mobile chip that might be embedded in medical image equipment. Visualized results generated by the CAM module actually highlight part of the class-specific regions of the diagnostic points, and they are valuable for decision making in the diagnosis of skin diseases.

PAPER 5: Necrolytic migratory erythema is an important visual cutaneous clue of glucagonoma **PUBLICATION YEAR:** August 2022

AUTHOR NAME: Wei Li1,6, XueYang1,6, Yuan Deng2 , Yina Jiang2 , Guiping Xu3 , Enxiao Li4 , YinyingWu4 , Juan Ren5 , Zhenhua Ma1 , Shunbin Dong1 , Liang Han1 , Qingyong Ma1 , ZhengWu1 & ZhengWang1

JOURNAL NAME: www.nature.com/scientificreports

SUMMARY:

Erythema is an extremely rare and slow-growing functional pancreatic neuroendocrine tumor arising from islet alpha cells in the tail of the pancreas. It usually presents with glucagonoma syndrome associated with characteristic clinical symptoms, including necrolytic migratory erythema (NME), diabetes mellitus (DM), stomatitis, anemia, deep vein thrombosis (DVT), weight loss, diarrhea and other symptoms. With the exception of NME, other clinical manifestations are nonspecific, which accounts for the delay in diagnosis in most cases and also for the fact that at least 50% of cases already have metastatic disease at the time of diagnosis. NME is observed in approximately 70–90% of patients diagnosed with glucagonoma. This rash is usually widespread, and the major sites of involvement are the perioral region, trunk, extremities and perineum. The distinguishing feature of NME is annular erythematous plaques with central bullous, ulcerative lesions surrounded by brown pigment, which are usually pruritic and painful. The histological features of this skin lesion include parakeratosis, hyperkeratosis, spongiosis of the epidermis with necrolysis, loss of the granular layer, vacuolization of keratinocytes, and perivascular and interstitial inflammation. This paper summarizes the clinical characteristics of seven typical patients with glucagonoma followed at our hospital during the past 10 years. Our cumulative experiences (including diagnosis and treatment) may help clinicians to better recognize, diagnose and treat glucagonoma. This study was approved by the Ethics Committee of the First Affiliated Hospital of Xi’an Jiao tong University and the study was conducted in accordance with the approved guidelines. Informed consent was obtained from all subjects and/or their legal guardian(s). We reviewed the database and collected seven cases of glucagonoma in the past 10 years. Patients with clinical presentations of skin manifestation (the skin rash is characterized by an intense erythematous lesion, which shows superficial epidermal necrosis and spreads in a centrifugal pattern), glucagonoma syndrome, elevated plasma glucagon, and a pathological diagnosis of pancreatic islet cell tumor were included in this cohort. The medical records of the included patients were reviewed. Tumor diameters were obtained from CT scan measurements. Follow?up data, including patients' follow-up status, symptoms (skin rash), recovery and administration of other therapies, were acquired from hospital medical records or by phone interviews with the patients, relatives, or general practitioners.

CONCLUSION:

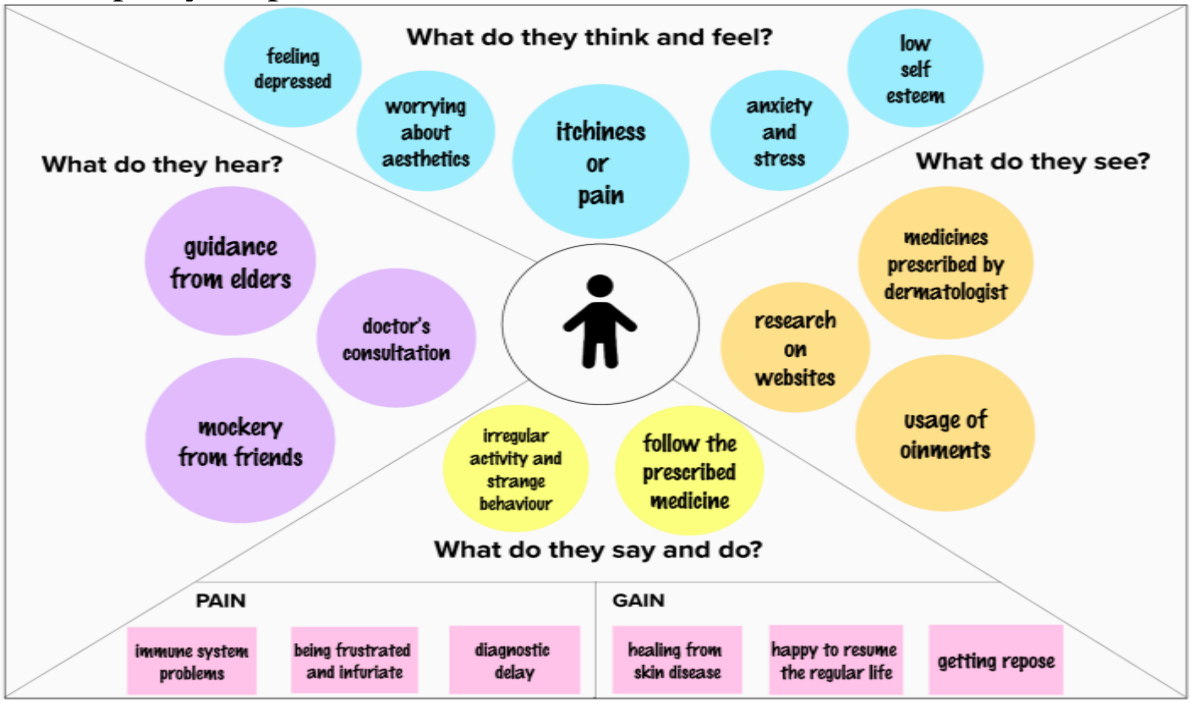
Surgical removal is considered to be the only definitive and curative treatment for pancreatic glucagonoma and NME7 . Optional operations included simple enucleation (< 2 cm) with peripancreatic lymph dissection, pancreaticoduodenectomy with peripancreatic lymph dissection, distal pancreatectomy with peripancreatic lymph dissection and splenectomy. However, more than half of all glucagonomas present with metastatic disease, most commonly liver metastasis. It has been reported that synchronous resection of pancreatic neuroendocrine tumors and liver metastasis (more than 30% of the liver tissue retained) provides a more favorable outcome. Liver transplantation may be considered as a potential therapeutic approach for unresectable hepatic metastases arising from pancreatic glucagonoma20. TACE might also be a safe therapeutic approach for liver metastasis arising from NETs because of the highly vascular and blood supply that primarily derives from the hepatic artery21. In addition, RFA is usually performed in combination with surgery, which has certain advantages in removing isolated metastases22. Medical therapy for glucagonoma, including chemotherapeutics, somatostatin analogous, PRRT and molecular targeted drugs, are also effective in controlling clinical symptoms and tumour growth7,16. In conclusion, erythema is a rare type of functional NET. Since NME might be the only clue for the early detection of this tumour, it is very important to correctly diagnose NME in a timely manner Currently, surgical intervention is the only definitive treatment for this disease. Medical therapy is effective for symptom control and metastatic disease management.

2.3.Problem Statement Definition

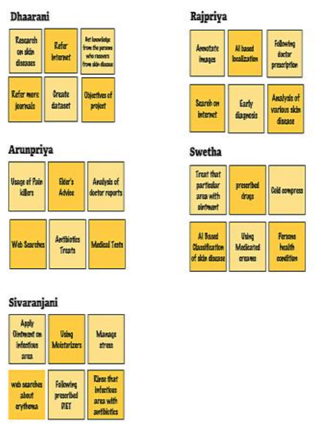
- Now a day's people are suffering from skin diseases, More than 125 million people suffering from Psoriasis also skin cancer rate is rapidly increasing over the last few decades especially Melanoma is most diversifying skin cancer.
- If skin diseases are not treated at an earlier stage, then it may lead to complications in the body including spreading of the infection from one individual to the other.
- The skin diseases can be prevented by investigating the infected region at an early stage.
- The characteristic of the skin images is diversified so that it is a challenging job to devise an efficient and robust algorithm for automatic detection of skin disease and its severity.
- Skin tone and skin colour play an important role in skin disease detection. Colour and coarseness of skin are visually different.
- Automatic processing of such images for skin analysis requires quantitative discriminator to differentiate the diseases.

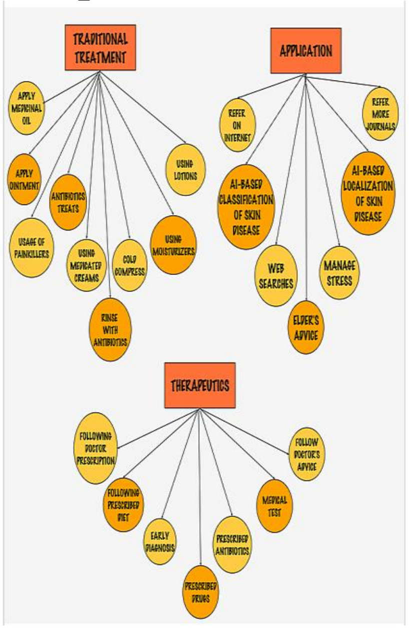
IDEATION & PROPOSED SOLUTION

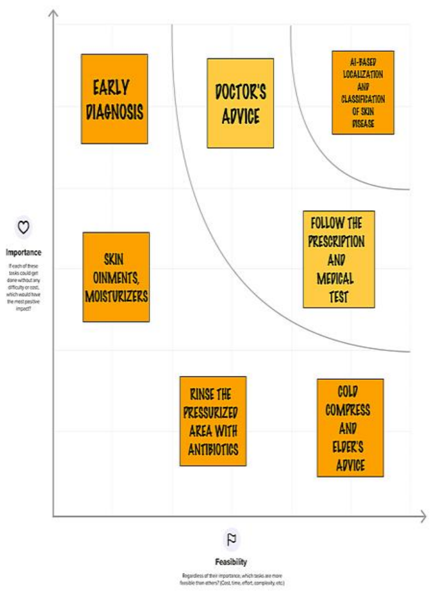
3.1.Empathy Map Canvas



3.2.Ideation & Brainstorming


Dhaarani
Research on skin diseases
Refer Internet
Get knowledge from the persons who recovers from skin disease
Refer more journals
Create dataset
Objectives of project
Rajpriya
Annotate images
AI based localization
Following doctor prescription
Search on internet
Early diagnosis
Analysis of various skin disease
Arunpriya
Usage of Pain killers
Elder's Advice
Analysis of doctor reports
Web Searches
Antibiotics Treats
Medical Tests
Swetha
Treat that particular area with ointment
prescribed drugs
Cold compress
AI Based Classification of skin disease
Using Medicated creams
Persons health condition
Sivaranjani
Apply Ointment on infectious area
Using Moisturizers
Manage stress
web searches about erythema
Following prescribed DIET
Rinse that infectious area with antibiotics




EARLY DIAGNOSIS
DOCTOR'S ADVICE
AI-BASED LOCALIZATION AND CLASSIFICATION OF SKIN DISEASE
SKIN OINMENTS, MOISTURIZERS
FOLLOW THE PRESCRIPTION AND MEDICAL TEST
RINSE THE PRESSURIZED AREA WITH ANTIBIOTICS
COLD COMPRESS AND ELDER'S ADVICE
Importance
If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?
Feasibility
Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)

3.2.Proposed Solution

- ➤ To overcome the above problem we are building a model which is used for the prevention and early detection of skin cancer, psoriasis.
- ➤ Basically, skin disease diagnosis depends on the different characteristics like colour, shape, texture etc. Here the person can capture the images of skin and then the image will be sent the trained model.
- ➤ The model analyses the image and detect whether the person is having skin disease or not.

Novelty:

- ➤ The novelty proposed in this approach is we have collected the dataset on our own.
- ➤ We have also annotated the images by ourselves.

Social Impact:

- ➤ The model which will be built by us is very useful for the users to find the type of disease quickly and get the correct the medicine as soon as possible.
- ➤ We ensure to the users that our model diagnoses the diseases well.

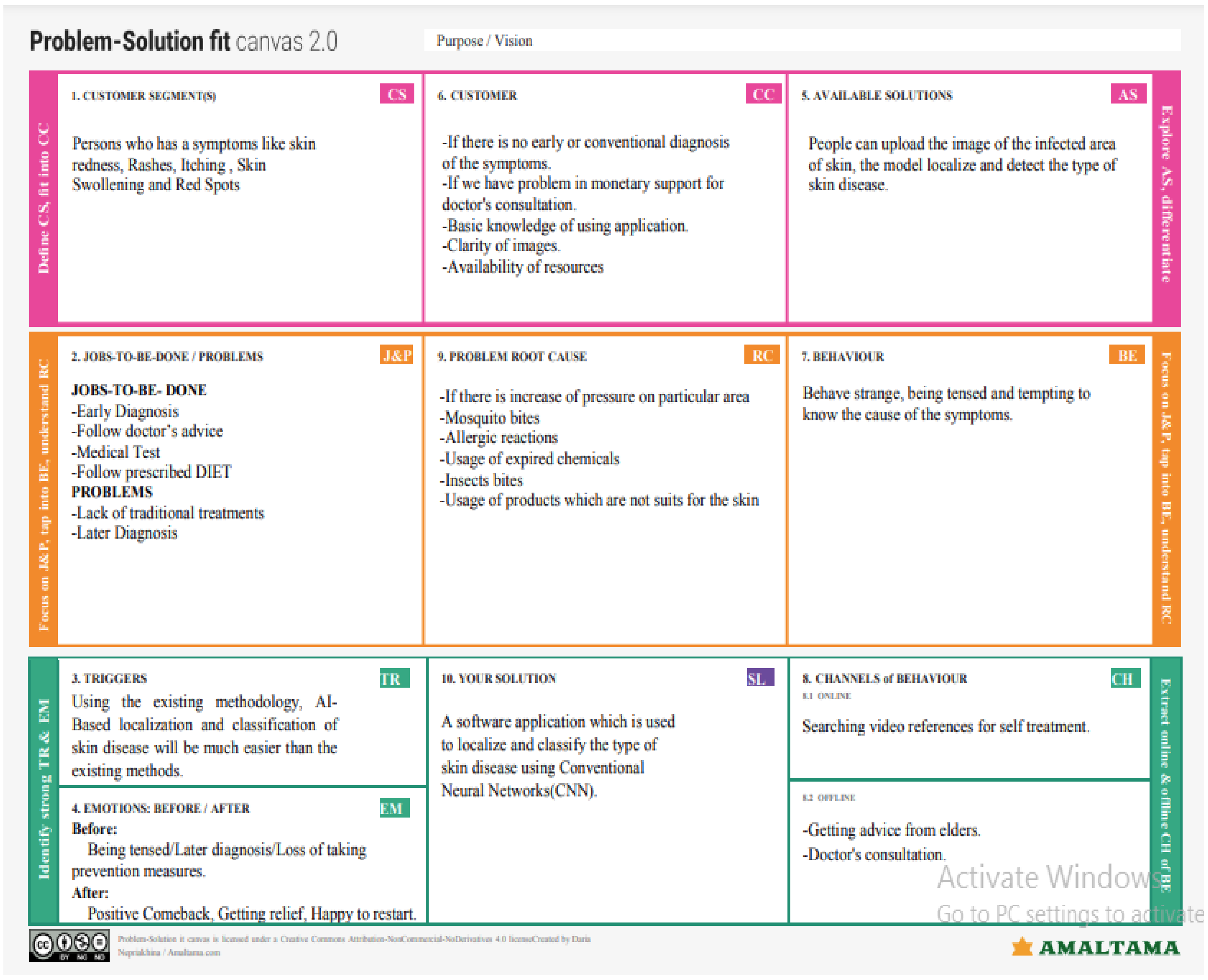
Business Model:

- ➤ Health Care Sector(Hospitals).
- ➤ Can generate revenue through direct customers.
- ➤ Can collaborate with health care sector and generate revenue from their customers.
- ➤ Contributing the corporate social responsibility by providing better solutions to the healthcare and to patients.

Scalability of the solution:

- ➤ To detect how much area is affected by the skin disease with accuracy.
- ➤ Classification of Skin disease will help doctors to diagnose the disease effectively.

3.4.Problem Solution Fit



Problem-Solution fit canvas 2.0

Purpose / Vision

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

Persons who has a symptoms like skin redness, Rashes, Itching , Skin Swollening and Red Spots

6. CUSTOMER CC

-If there is no early or conventional diagnosis of the symptoms.
-If we have problem in monetary support for doctor's consultation.
-Basic knowledge of using application.
-Clarity of images.
-Availability of resources

5. AVAILABLE SOLUTIONS AS

People can upload the image of the infected area of skin, the model localize and detect the type of skin disease.

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

JOBS-TO-BE- DONE
-Early Diagnosis
-Follow doctor's advice
-Medical Test
-Follow prescribed DIET
PROBLEMS
-Lack of traditional treatments
-Later Diagnosis

9. PROBLEM ROOT CAUSE RC

-If there is increase of pressure on particular area
-Mosquito bites
-Allergic reactions
-Usage of expired chemicals
-Insects bites
-Usage of products which are not suits for the skin

7. BEHAVIOUR BE

Behave strange, being tensed and tempting to know the cause of the symptoms.

Focus on J&P, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TR

Using the existing methodology, AI-Based localization and classification of skin disease will be much easier than the existing methods.

4. EMOTIONS: BEFORE / AFTER EM

Before:
Being tensed/Later diagnosis/Loss of taking prevention measures.
After:
Positive Comeback, Getting relief, Happy to restart.

10. YOUR SOLUTION SL

A software application which is used to localize and classify the type of skin disease using Conventional Neural Networks(CNN).

8. CHANNELS of BEHAVIOUR CH

8.1 ONLINE

Searching video references for self treatment.

8.2 OFFLINE

-Getting advice from elders.
-Doctor's consultation.

Extract online & offline CH of BE

AMALTAMA

REQUIREMENT ANALYSIS

4.1.Functional requirement

FR No.	**Functional Requirement (Epic)**	**Sub Requirement (Story/ Sub-Task)**
FR-1	User Registration	Registration through Form Registration through Gmail Registration throughLinkedIN
FR-2	User Confirmation	Confirmation via EmailConfirmati on via OTP
FR-3	User Interface	Login form
FR-4	Image processing	Yolo model and convolutional neural network use deeplearning. Identifies an object in a picture.
FR-5	Accessing cloudservices	Submit the information to the Cloudant database. Obtainthe information that the user needs

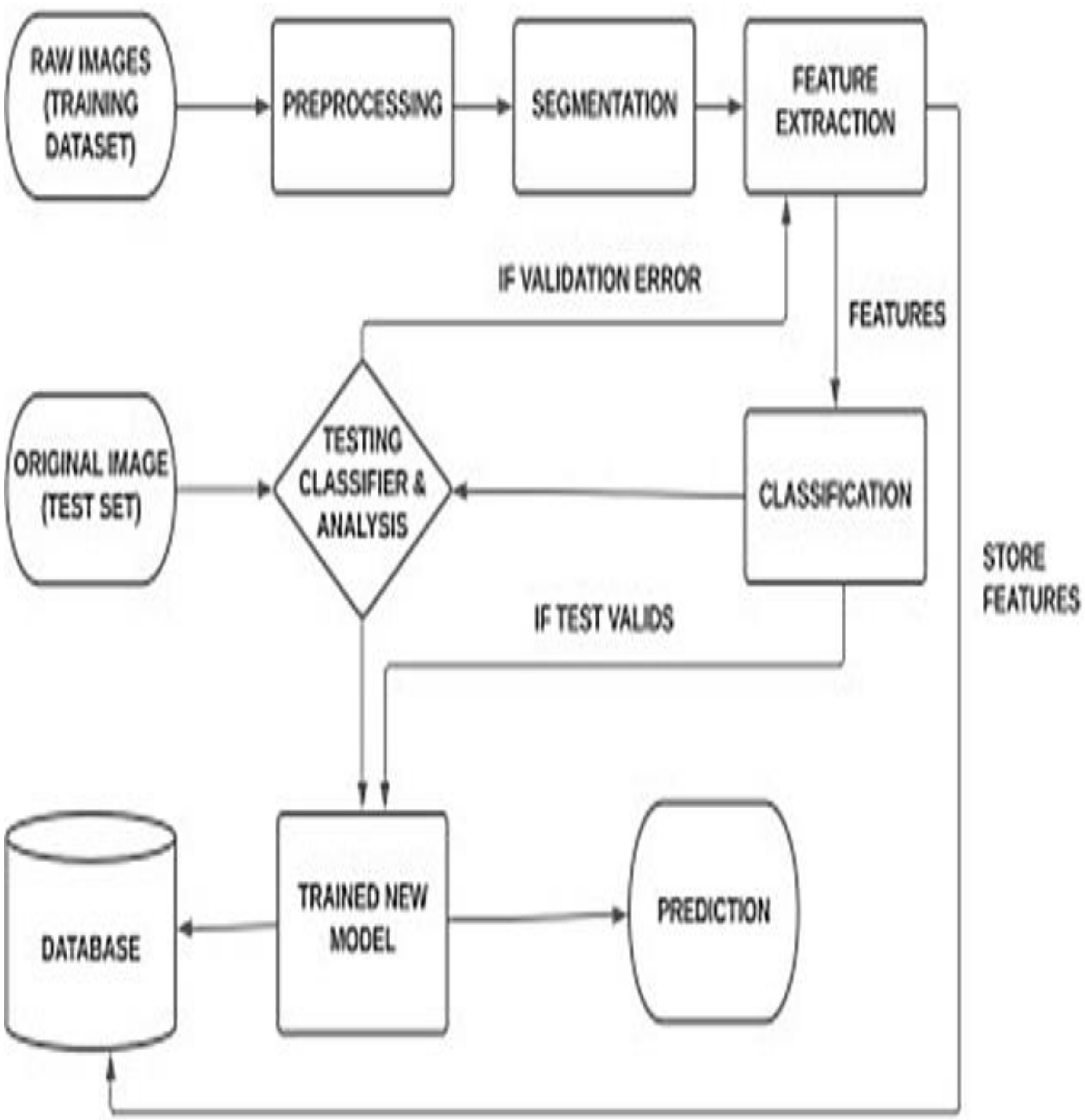
4.2.Non-Functional requirement

FR No.	**Non-Functional Requirement**	**Description**
NFR-1	**Usability**	Anyone with access to the application can utilise it.Understanding of the internet and computers
NFR-2	**Security**	User safety is enabled for security through two-factorauthentication.
NFR-3	**Reliability**	Since it makesuse of reputable cloud providers like IBM, it is quite trustworthy.
NFR-4	**Performance**	Performance is superior than those of otherapplications and marketgoods.
NFR-5	**Availability**	Accessible via the play storeand usable on all devices.

NFR-6	**Scalability**	Utilizing cloudservices increases scalability.

PROJECT DESIGN

5.1.Data flow diagrams

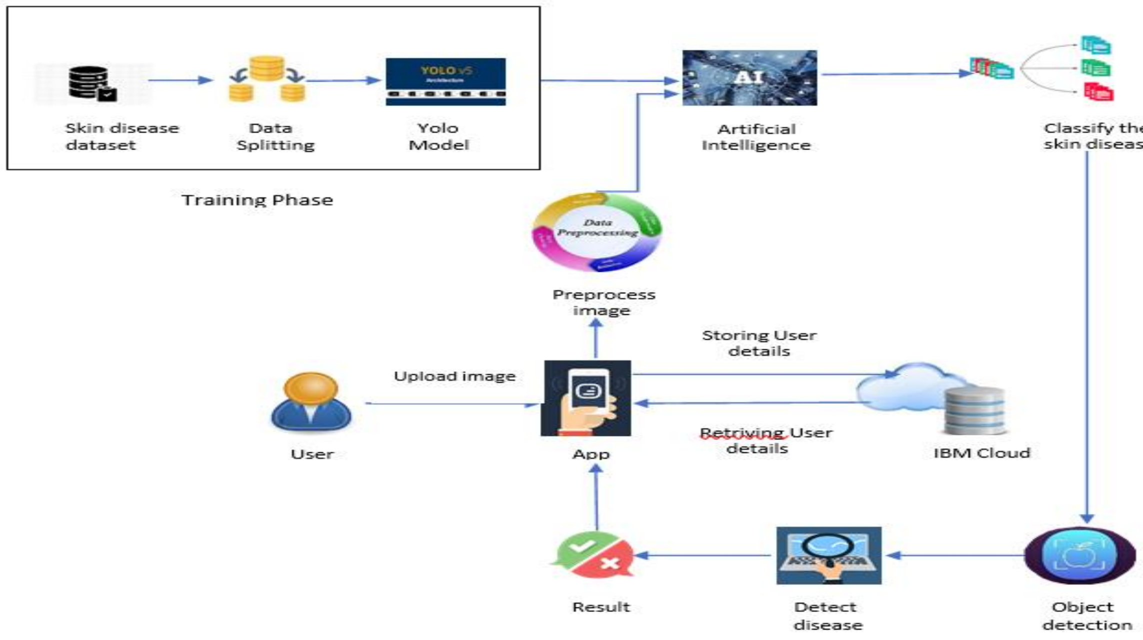


5.2.Solution & Technical Architecture

Solution Architecture:

- The Skin Disease Image Dataset is taken from Google and is processedand annotated using Robot flow tool.
- This processes data set is splitted into Training (70%),Validation (20%)and Testing(10%) datasets.
- This processed dataset is trained using Yolov and OpenCV python libraries.
- An Artificial Intelligence model is built after training for considerablenumber of epochs which is shown from the training phase diagram.
- A Full Stack Web App is build using React and Flask with cloud Database provided by IBM.
- The skin diseased image uploaded by the user from the frontend is fed intothe AI model built which classifies what type of disease and localizes the areas affected. This process takes place at the backend.
- This output result is finally displayed to the user.

Solution Architecture Diagram:



Technology Architecture:

In the past, the skills required to make an accurate dermatological diagnosis have required exposure to thousands of patients over many years. However, in recent years, artificial intelligence (AI) has made enormous advances, particularly in the area of image classification.

This has led computer scientists to apply these techniques to develop algorithms that are able to recognize skin lesions, particularly melanoma. Since 2017, there have been numerous studies assessing the accuracy of algorithms, with some reporting that the accuracy matches or surpasses that of a dermatologist. While the principles underlying these methods are relatively straightforward, it can be challenging for the practising dermatologist to make sense of a plethora of unfamiliar terms in this domain.

Here we explain the concepts of AI, machine learning,neural networks and deep learning, and explore the principles of how these tasks are accomplished. We critically evaluate the studies that have assessed the efficacy of these methods and discuss limitations and potential ethical issues. The burden of skin cancer is growing within the Western world, with major implications for both population skin health and the provision of dermatology services.AI has the potential to assist in the diagnosis of skin lesions and may have particular value at the interface between primary and secondary care. The emerging technology represents an exciting opportunity fordermatologists, who are the individuals best informed to explore the utility of this powerful novel diagnostic tool, and facilitate itssafe and ethical implementation within several healthcare system.

FIGURE1:

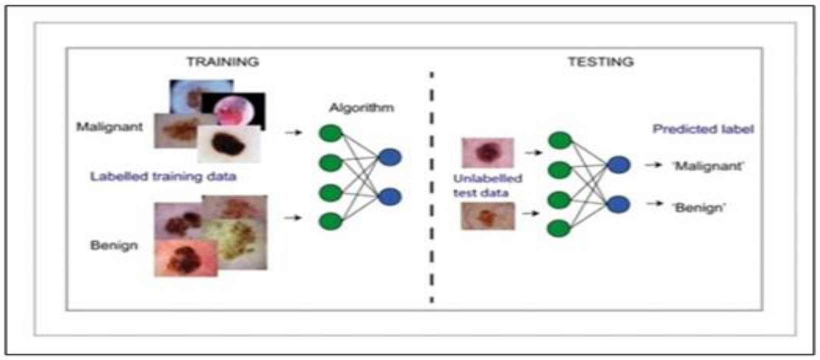


FIGURE2:

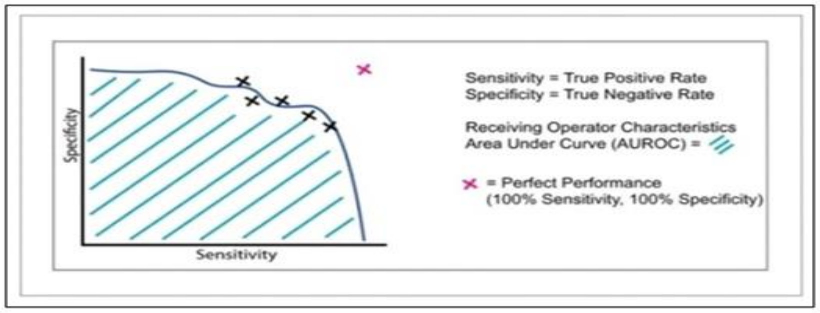


FIGURE3:

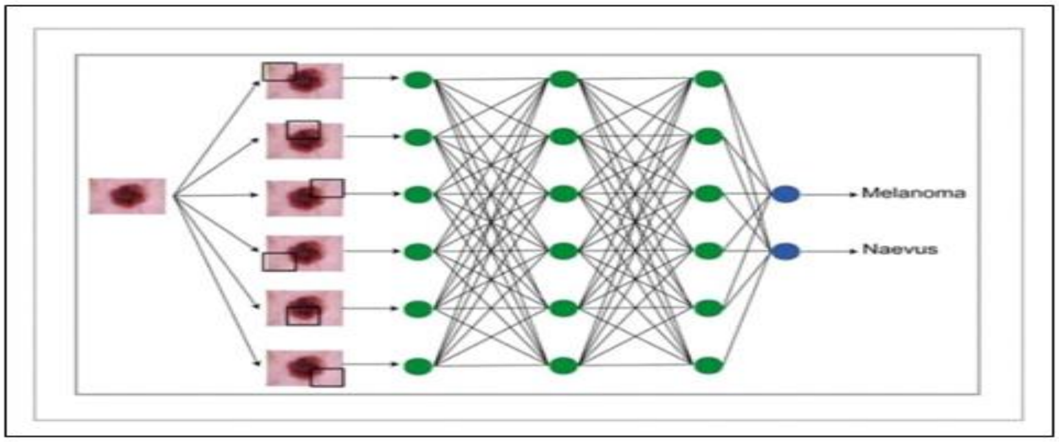


FIGURE4:

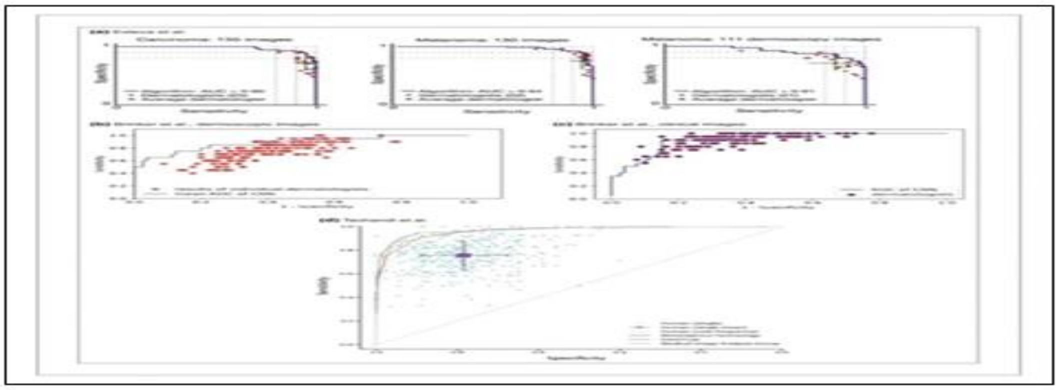
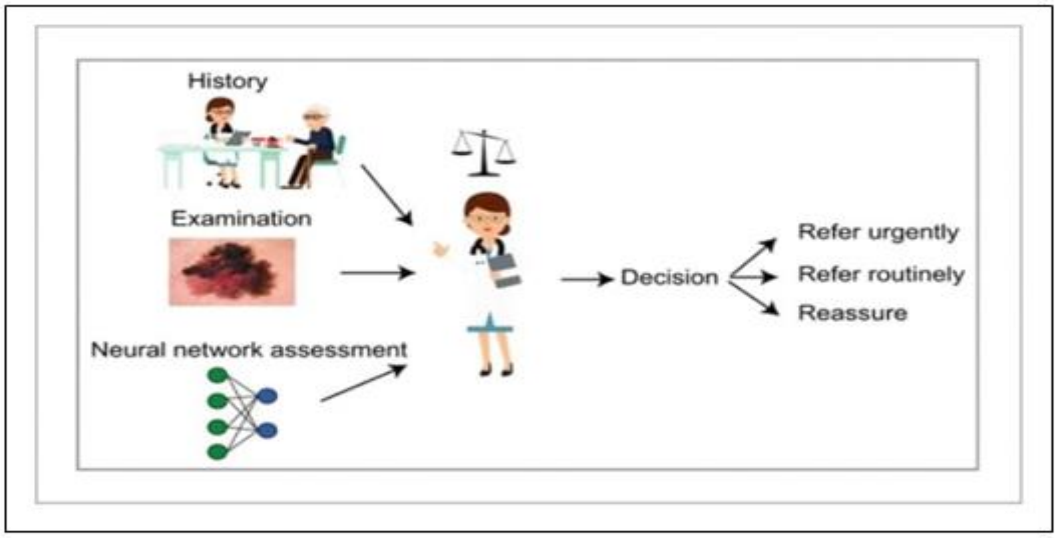


FIGURE5:



PROJECT PLANNING & SCHEDULING

6.1. Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story /Task	Story Points	Priority	Team Members
Sprint-1	Create Dataset	USN-1	Create the dataset with 50 images perskin disease	3	High	Rajpriya R, Dhaarani A, Swetha S, Arunpriya M
Sprint-2	Annotate images	USN-2	Annotate images using Microsoft VOTT into four phases	3	High	Sivaranjini D, Swetha S, Rajpriya R, Dhaarani A
Sprint-3	Training Yolo and Build python code	USN-3	Download and Convert pre- trained weights. Train Yolov3 detector and build the source code	3	High	Arunpriya M, Sivaranjini D
Sprint-4	Cloudant DB	USN-4	Create cloud account, create serviceinstance, launch cloudant DB and create the database	3	High	Swetha S, Rajpriya R

Sprint-4	Registration	USN-5	As a user, I can register for the application by entering my email, password, and confirming my password.	3	High	Dhaarani A
Sprint-4		USN-6	As a user, I will receive confirmation email once I have registered for the application	2	Medium	Sivaranjani D
Sprint-4		USN-7	As a user, I can register for the application through mobile number	3	High	Arunpriya M
Sprint-4		USN-8	As a user. I will Receive confirmation SMS	3	High	Rajpriya R
Sprint-4	Login	USN-9	As a user, I can log into the application by entering login credentials	3	High	DhaaraniA, Swetha S

Sprint-4	Dashboard	USN-10	As a user, I can upload my images and get my details of skin diseases	3	High	Sivaranjini D, Arunpriya M
Sprint -4	Logout	USN-11	As a user, I can logout successfully	2	Medium	Dhaarani A, Rajpriya R

6.2. Sprint Delivery Schedule

TITLE	DESCRIPTION	DATE
Literature Survey & Information Gathering	Literature survey on the selected project & gathering information by referring the, technical papers,research publications etc.	3 SEPTEMBER 2022
Prepare Empathy Map	Prepare Empathy Map Canvas to capture the user Pains & Gains, Prepare list of problem statements	10 SEPTEMBER 2022
Ideation	List the by organizing the brainstorming session and prioritize the top 3 ideas based on the feasibility & importance.	10 SEPTEMBER 2022
Proposed Solution	Prepare the proposed solution document, which includes the novelty, feasibility of idea, business model, social impact, scalability of solution, etc.	10 SEPTEMBER 2022
Problem Solution Fit	Prepare problem - solution fit document.	1 OCTOBER 2022

Solution Architecture	Prepare solution architecture document.	19 OCTOBER 2022
Customer Journey	Prepare the customer journey maps to understand the user interactions & experiences with the application.	15 OCTOBER 2022
Data Flow Diagrams	Draw the data flow diagrams and submit for review.	18 OCTOBER 2022
Technology Architecture	architecture diagram.	20 OCTOBER 2022
Prepare Milestone & Activity List	Prepare the milestones & activity list of the project.	23 OCTOBER 2022
Project Development - Delivery of Sprint-1, 2, 3 & 4	Develop & submit the developed code by testing it.	IN PROGRESS..

CODING & SOLUTIONING

7.1.Feature 1

In this project we are implementing a feature which is creating a website for AI-based localization and classification of skin disease with erythema. By using this feature we can login into the website and upload photos.

Code for login page:

```
<!DOCTYPE html>
<html >
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>SKIN CLASSIFICATION</title>
<link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300' rel='stylesheet'
type='text/css'>
<link rel="stylesheet" href="{{ url_for('static', filename='css/style.css') }}">
<link href='https://fonts.googleapis.com/css?family=Merriweather' rel='stylesheet'>
<link href='https://fonts.googleapis.com/css?family=Josefin Sans' rel='stylesheet'>
<link href='https://fonts.googleapis.com/css?family=Montserrat' rel='stylesheet'>
<style>
.header {
                        top:0;
                        margin:0px;
                        left: 0px;
                        right: 0px;
                        position: fixed;
                        background-color: #566573 ;
                        color: #7FE228;
                        box-shadow: 0px 8px 4px grey;
                        overflow: hidden;
                        padding-left:20px;
                        font-family: 'Josefin Sans';
                        font-size: 2vw;
                        width: 100%;
                        height:8%;
                        text-align: center;
                }
                .topnav {
 overflow: hidden;
 background-color: #D817B8;
```

```
}
.topnav-right a {
  float: left;
  color: #40E0D0;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
  font-size: 18px;
}
.topnav-right a:hover {
  background-color: #40E0D0;
  color: black;
}
.topnav-right a.active {
  background-color: #40E0D0;
  color: white;
}
.topnav-right {
  float: right;
  padding-right:100px;
}
.SigIn{
margin-top:-70px;
}
body {
  background-color:#0EE579 ;
  background-repeat: no-repeat;
  background-size:cover;
  background-position: 0px 0px;
   }
.SignIn{
            margin-top:100px;
}
form {border: 3px solid #f1f1f1; margin-left:400px;margin-right:400px;}

input[type=text], input[type=email],input[type=number],input[type=password] {
  width: 100%;
  padding: 12px 20px;
  display: inline-block;
  margin-bottom:18px;
  border: 1px solid #ccc;
  box-sizing: border-box;
}
button {
  background-color: #239B56;
  color: white;
  padding: 14px 20px;
  margin-bottom:8px;
  border: none;
  cursor: pointer;
  width: 100%;
}
button:hover {
  opacity: 0.8;
}
.cancelbtn {
  width: auto;
  padding: 10px 18px;
  background-color: #f44336;
}
.imgcontainer {
  text-align: center;
  margin: 24px 0 12px 0;
}
img.avatar {
  width: 30%;
  border-radius: 50%;
}
.container {
  padding: 16px;
}
span.psw {
  float: right;
  padding-top: 16px;
}
/* Change styles for span and cancel button on extra small screens */
@media screen and (max-width: 300px) {
  span.psw {
     display: block;
     float: none;
  }
  .cancelbtn {
     width: 100%;
  }
}
</style>
</head>
<body style="font-family:Montserrat;">
<div class="header">
 <div style="width:50%;float:left;font-size:2vw;text-align:left;color:white; padding-top:1%">SKIN
CLASSIFICATION</div>
  <div class="topnav-right" >
    <a  href="{{url_for('home')}}">Home</a>
    <a href="{{url_for('login')}}">Log In</a>
    <a class="active"  href="{{url_for('signup')}}">Sign Up</a>

  </div>
</div>
```

```
<div id="SignIn" class="SignIn">
            <form action="{{url_for('afterreg')}}" method="post">
                        <div class="imgcontainer">
      <h2>SIGN UP</h2>
      <h6 class="information-text">Provide valid details</h6>
                        </div>
                        <div class="container">
                                    <input type="text" placeholder=" Name" name="name" required><br>
                                    <input type="email" placeholder=" Email ID" name="_id" required><br>
                                    <input type="password" placeholder=" Password" name="psw" required>

                                    <button type="submit">Register</button><br>
                        </div>
                        <div class="container" style="background-color:#5DADE2 ">
    <div class="psw">Already have an account?&nbsp; &nbsp;<a href="l{{url_for('login')}}">Sign
In</a></div>
  </div>
            </form>
</div>
</body>
</html>
```

7.2.Feature 2

In this project we are using CNN. CNN is used for effective classification. CNN is computionally efficient and we can classify the images without much human intervention.

Code for building CNN:

```
# Part 1 - Building the CNN
#importing the Keras libraries and packages
from keras.models import Sequential
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense, Dropout
from keras import optimizers

# Initialing the CNN
classifier = Sequential()

# Step 1 - Convolution Layer
classifier.add(Convolution2D(32, 3,  3, input_shape = (64, 64, 3), activation = 'relu'))

#step 2 - Pooling
classifier.add(MaxPooling2D(pool_size =(2,2)))
```

```
# Adding second convolution layer
classifier.add(Convolution2D(32, 3,  3, activation = 'relu'))
classifier.add(MaxPooling2D(pool_size =(2,2)))

#Adding 3rd Concolution Layer
classifier.add(Convolution2D(64, 3,  3, activation = 'relu'))
classifier.add(MaxPooling2D(pool_size =(2,2)))


#Step 3 - Flattening
classifier.add(Flatten())

#Step 4 - Full Connection
classifier.add(Dense(256, activation = 'relu'))
classifier.add(Dropout(0.5))
classifier.add(Dense(10, activation = 'softmax'))

#Compiling The CNN
classifier.compile(
            optimizer = 'adam',
            loss = 'categorical_crossentropy',
            metrics = ['accuracy'])

#Part 2 Fittting the CNN to the image
from keras.preprocessing.image import ImageDataGenerator
train_datagen = ImageDataGenerator(
        rescale=1./255,
        shear_range=0.2,
        zoom_range=0.2,
        horizontal_flip=True)

test_datagen = ImageDataGenerator(rescale=1./255)

training_set = train_datagen.flow_from_directory(
        'Data/train',
        target_size=(64, 64),
        batch_size=32,
        class_mode='categorical')

test_set = test_datagen.flow_from_directory(
        'Data/test',
        target_size=(64, 64),
        batch_size=32,
        class_mode='categorical')

model = classifier.fit_generator(
        training_set,
        steps_per_epoch=100,
        epochs=100,
        validation_data = test_set,
        validation_steps = 6500
      )

#Saving the model
import h5py
classifier.save('Trained_Model.h5')

print(model.history.keys())
import matplotlib.pyplot as plt

# summarize history for accuracy
plt.plot(model.history['acc'])
plt.plot(model.history['val_acc'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()

# summarize history for loss
plt.plot(model.history['loss'])
plt.plot(model.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```

7.3.Source Code

```
pip3 install tensorflow tensorflow_hub matplotlib seaborn numpy pandas sklearn imblearn

  import tensorflow as tf
  import tensorflow_hub as
  hub
  import matplotlib.pyplot as
  pltimport numpy as np
  import pandas as pd
  import seaborn as
  sns
  from tensorflow.keras.utils import get_file
from sklearn.metrics import roc_curve, auc, confusion_matrix
from imblearn.metrics import sensitivity_score, specificity_score

  import os
  import glob
  import zipfile
  import
```

```
random

# to get consistent results after multiple
runstf.random.set_seed(7)
np.random.seed(7)
random.seed(7)

# 0 for benign, 1 for malignant
class_names = ["benign",
"malignant"]
```

Preparing the Dataset

```
def download_and_extract_dataset():
  # dataset  from  https://github.com/udacity/dermatologist-
  ai# 5.3GB
  train_url =  "https://s3-us-west-1.amazonaws.com/udacity-dlnfd/datasets/skin-
  cancer/train.zip"# 824.5MB
  valid_url =  "https://s3-us-west-1.amazonaws.com/udacity-dlnfd/datasets/skin-
  cancer/valid.zip"# 5.1GB
  test_url = "https://s3-us-west-1.amazonaws.com/udacity-dlnfd/datasets/skin-
  cancer/test.zip"for i, download_link in enumerate([valid_url, train_url, test_url]):
    temp_file = f"temp{i}.zip"
    data_dir = get_file(origin=download_link, fname=os.path.join(os.getcwd(), temp_file))
    print("Extracting", download_link)
    with zipfile.ZipFile(data_dir, "r") as z:
      z.extractall("data")
    # remove the temp
    file
    os.remove(temp_file
    )

# comment the below line if you already downloaded the
datasetdownload_and_extract_dataset()
# preparing data
# generate CSV metadata file to read img paths and labels from
itdef generate_csv(folder, label2int):
    folder_name =
    os.path.basename(folder)labels =
    list(label2int)
    # generate CSV file
    df = pd.DataFrame(columns=["filepath",
    "label"])i = 0
    for label in labels:
        print("Reading", os.path.join(folder, label, "*"))
        for filepath in glob.glob(os.path.join(folder, label,
            "*")):df.loc[i] = [filepath, label2int[label]]
            i += 1
    output_file = f"{folder_name}.csv"
    print("Saving", output_file)
    df.to_csv(output_file)
```

# generate CSV files for all data portions, labeling nevus and seborrheic keratosis

```
# as 0 (benign), and melanoma as 1 (malignant)
# you should replace "data" path to your extracted dataset path
# don't replace if you used download_and_extract_dataset() function
generate_csv("data/train", {"nevus": 0, "seborrheic_keratosis": 0, "melanoma": 1})
generate_csv("data/valid", {"nevus": 0, "seborrheic_keratosis": 0, "melanoma": 1})
generate_csv("data/test", {"nevus": 0, "seborrheic_keratosis": 0,
"melanoma": 1})# loading data
train_metadata_filename =
"train.csv" valid_metadata_filename
= "valid.csv" # load CSV files as
DataFrames
df_train =
pd.read_csv(train_metadata_filename)
df_valid =
pd.read_csv(valid_metadata_filename)
n_training_samples = len(df_train)
n_validation_samples = len(df_valid)
print("Number of training samples:", n_training_samples)
print("Number of validation samples:",
n_validation_samples)
train_ds = tf.data.Dataset.from_tensor_slices((df_train["filepath"],
df_train["label"])) valid_ds =
tf.data.Dataset.from_tensor_slices((df_valid["filepath"], df_valid["label"]))
```

Output:

```
Number of training samples: 2000
Number of validation samples: 150
```

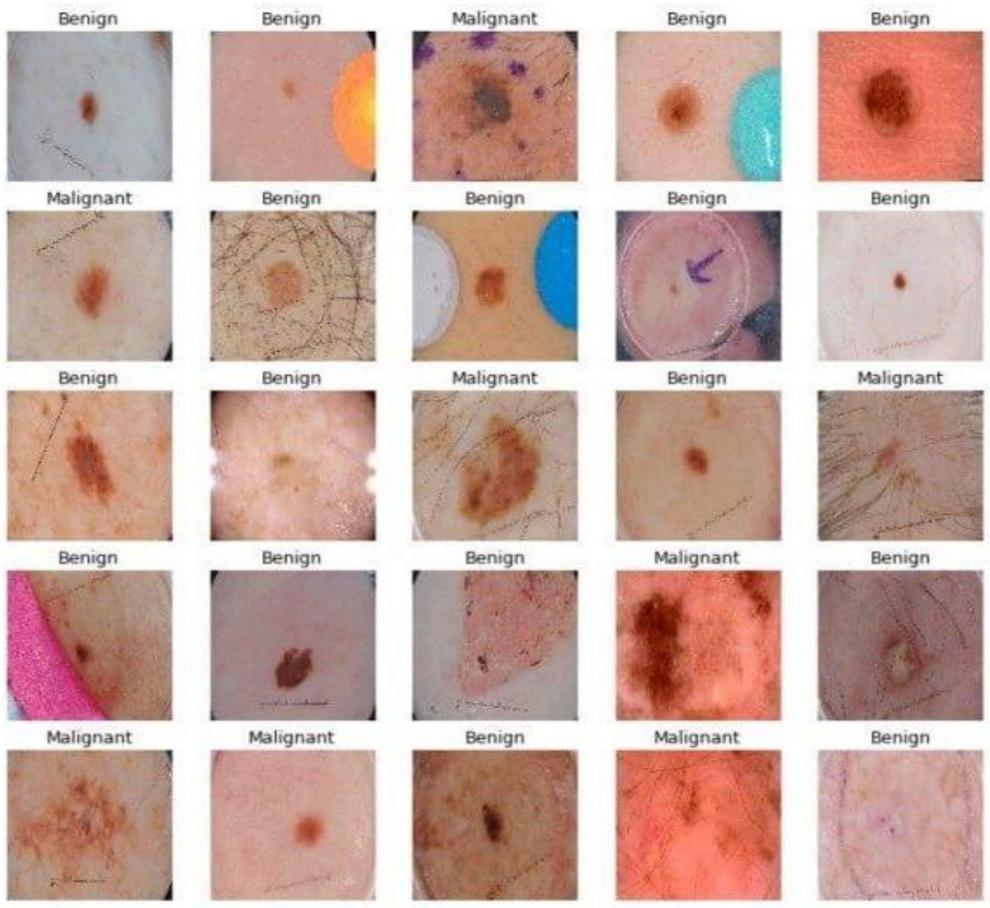
# preprocess data

```
def
decode_img(img):

# convert the compressed string to a 3D uint8
  tensorimg = tf.image.decode_jpeg(img,
  channels=3)
  # Use `convert_image_dtype` to convert to floats in the [0,1]
  range.img = tf.image.convert_image_dtype(img, tf.float32)
  # resize the image to the desired
  size.return tf.image.resize(img,
  [299, 299])
```

```
def process_path(filepath, label):
  # load the raw data from the file as a
  stringimg = tf.io.read_file(filepath)
  img =
  decode_img(img)
  return img, label


valid_ds                                    =
valid_ds.map(process_path)
train_ds                                    =
train_ds.map(process_path)          #
test_ds = test_ds
for image, label in train_ds.take(1):
    print("Image shape:", image.shape)
    print("Label:", label.numpy())
Image shape: (299, 299, 3)
Label: 0
# training
parameters
batch_size = 64
optimizer =
"rmsprop"


def prepare_for_training(ds, cache=True, batch_size=64,
shuffle_buffer_size=1000):if cache:
      if isinstance(cache, str):
        ds =
      ds.cache(cache)else:
        ds = ds.cache()
  # shuffle the dataset
  ds =
  ds.shuffle(buffer_size=shuffle_buffer_size)#
  Repeat forever
  ds = ds.repeat()
  # split to
  batches
  ds = ds.batch(batch_size)
  # `prefetch` lets the dataset fetch batches in the background while the
  model# is training.
  ds =
  ds.prefetch(buffer_size=tf.data.experimental.AUTOTUNE)
  return ds

valid_ds = prepare_for_training(valid_ds, batch_size=batch_size, cache="valid-cached-
data") train_ds = prepare_for_training(train_ds, batch_size=batch_size, cache="train-
cached-data") batch = next(iter(valid_ds))
```

```
def show_batch(batch):
  plt.figure(figsize=(12,12
  ))for n in range(25):
     ax =
     plt.subplot(5,5,n+1)
     plt.imshow(batch[0][
     n])
     plt.title(class_names[batch[1][n].numpy()].title())
     plt.axis('off')

show_batch(batch)
```

Output:



```
# building the model
# InceptionV3 model & pre-trained weights
module_url = "https://tfhub.dev/google/tf2-preview/inception_v3/feature_vector/4"

m = tf.keras.Sequential([
   hub.KerasLayer(module_url, output_shape=[2048],
   trainable=False), tf.keras.layers.Dense(1,
   activation="sigmoid")
])
```

```
m.build([None, 299, 299, 3])
m.compile(loss="binary_crossentropy", optimizer=optimizer, metrics=["accuracy"])
m.summary()
```

Output:

```
Model: "sequential"
_________________________________________________________________
Layer (type)              Output Shape           Param #
=================================================================
keras_layer (KerasLayer)    multiple                21802784
_________________________________________________________________
dense (Dense)               multiple               2049
=================================================================

Total params: 21,804,833
Trainable params: 2,049
Non-trainable params: 21,802,784
```

Training the Model

```
model_name = f"benign-vs-malignant_{batch_size}_{optimizer}"
tensorboard = tf.keras.callbacks.TensorBoard(log_dir=os.path.join("logs",
model_name)) # saves model checkpoint whenever we reach better weights
modelcheckpoint = tf.keras.callbacks.ModelCheckpoint(model_name + "_{val_loss:.3f}.h5",
save_best_only=True, verbose=1)

history = m.fit(train_ds, validation_data=valid_ds,
            steps_per_epoch=n_training_samples //
            batch_size,
            validation_steps=n_validation_samples // batch_size, verbose=1, epochs=100,
            callbacks=[tensorboard, modelcheckpoint])
```

Output:

```
Train for 31 steps, validate for 2 steps
Epoch 1/100
30/31 [============================>.] - ETA: 9s - loss: 0.4609 - accuracy: 0.7760
Epoch 00001: val_loss improved from inf to 0.49703, saving model to benign-vs-
malignant_64_rmsprop_0.497.h5
31/31 [==============================] - 282s 9s/step - loss: 0.4646 - accuracy: 0.7722 - val_loss: 0.4970 -
val_accuracy: 0.8125
<..SNIPED..>
Epoch 27/100
30/31 [============================>.] - ETA: 0s - loss: 0.2982 - accuracy: 0.8708
Epoch 00027: val_loss improved from 0.40253 to 0.38991, saving model to benign-vs-
malignant_64_rmsprop_0.390.h5
31/31 [==============================] - 21s 691ms/step - loss: 0.3025 - accuracy: 0.8684 -
val_loss: 0.3899 - val_accuracy: 0.8359
<..SNIPED..>
Epoch 41/100
30/31 [============================>.] - ETA: 0s - loss: 0.2800 - accuracy: 0.8802
```

```
Epoch 00041: val_loss did not improve from 0.38991
31/31 [==============================] - 21s 690ms/step - loss: 0.2829 - accuracy: 0.8790 -
val_loss: 0.3948 - val_accuracy: 0.8281Epoch
42/100
30/31 [============================>.] - ETA: 0s - loss: 0.2680 - accuracy: 0.8859
Epoch 00042: val_loss did not improve from 0.38991
31/31 [==============================] - 21s 693ms/step - loss: 0.2722 - accuracy: 0.8831 -
val_loss: 0.4572 - val_accuracy: 0.8047
```

Model Evaluation:

```
# evaluation

# load testing set

test_metadata_filename = "test.csv"



df_test = pd.read_csv(test_metadata_filename)

n_testing_samples = len(df_test)

print("Number of testing samples:", n_testing_samples)

test_ds = tf.data.Dataset.from_tensor_slices((df_test["filepath"], df_test["label"]))

def prepare_for_testing(ds, cache=True, shuffle_buffer_size=1000):

  if cache:

    if isinstance(cache, str):

      ds = ds.cache(cache)

    else:

      ds = ds.cache()

  ds = ds.shuffle(buffer_size=shuffle_buffer_size)

  return ds

test_ds = test_ds.map(process_path)

test_ds = prepare_for_testing(test_ds, cache="test-cached-data")
```

```
Number of testing samples:

600# evaluation

# load testing set

test_metadata_filename =

"test.csv"
```

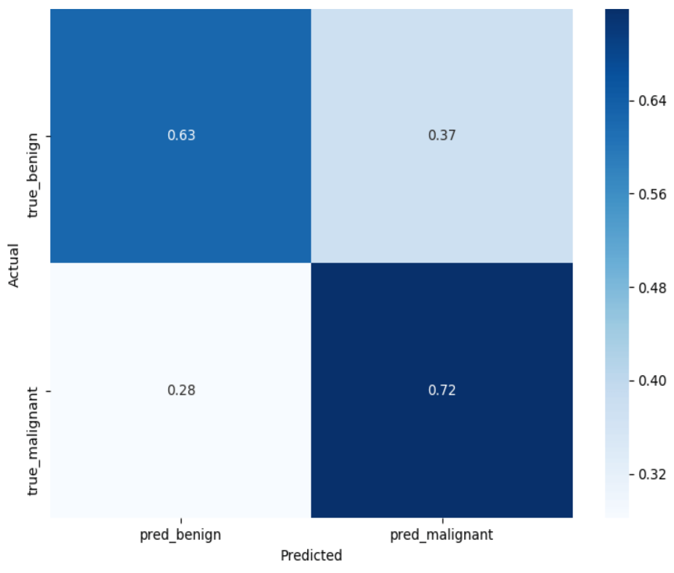
```
df_test =

pd.read_csv(test_metadata_filename)

n_testing_samples = len(df_test)

print("Number of testing samples:", n_testing_samples)

test_ds = tf.data.Dataset.from_tensor_slices((df_test["filepath"], df_test["label"]))



def prepare_for_testing(ds,  cache=True,

 shuffle_buffer_size=1000):if cache:

   if isinstance(cache,

   str):ds =

   ds.cache(cache) else:

     ds = ds.cache()

      ds =

 ds.shuffle(buffer_size=shuffle_buffer_size)

 return ds

test_ds = test_ds.map(process_path)

test_ds = prepare_for_testing(test_ds, cache="test-cached-data")

# load the weights with the least loss
m.load_weights("benign-vs-malignant_64_rmsprop_0.390.h5")
print("Evaluating the model...")
loss, accuracy = m.evaluate(X_test, y_test, verbose=0)
print("Loss:", loss, "  Accuracy:", accuracy)
```

Output:

```
Evaluating the model...
Loss: 0.4476394319534302  Accuracy: 0.8
```

```
def  get_predictions(threshold=None):
 """
 Returns predictions for binary classification given `threshold`
 For instance, if threshold is 0.3, then it'll output 1 (malignant) for that
 sample if the probability of 1 is 30% or more (instead of 50%)
 """
 y_pred =
 m.predict(X_test) if not
 threshold:
   threshold = 0.5
 result =
 np.zeros((n_testing_samples,)) for i in
 range(n_testing_samples):
   # test melanoma
   probability if y_pred[i][0]
   >= threshold:
     result[i] = 1
   # else, it's 0 (benign)
 return result

threshold = 0.23
# get predictions with 23% threshold
# which means if the model is 23% sure or more that is
malignant,# it's assigned as malignant, otherwise it's benign
y_pred = get_predictions(threshold)
def plot_confusion_matrix(y_test, y_pred):
 cmn = confusion_matrix(y_test,
 y_pred)# Normalise
 cmn = cmn.astype('float') / cmn.sum(axis=1)[:,
 np.newaxis]# print it
 print(cmn)
 fig, ax = plt.subplots(figsize=(10,10))
 sns.heatmap(cmn, annot=True, fmt='.2f',
           xticklabels=[f"pred_{c}" for c in class_names],
           yticklabels=[f"true_{c}" for c in class_names],
              cmap="Blues"
           )
 plt.ylabel('Actual')
 plt.xlabel('Predicted')
 # plot the resulting confusion
 matrix plt.show()

plot_confusion_matrix(y_test, y_pred)
```

Output:

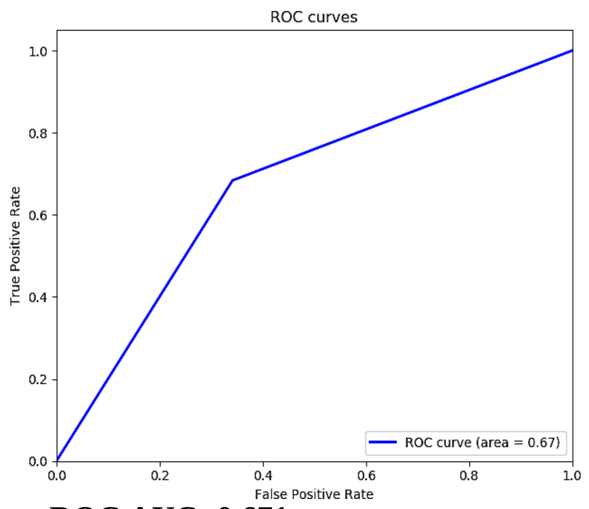


```
sensitivity = sensitivity_score(y_test,
y_pred) specificity =
specificity_score(y_test, y_pred)

print("Melanoma Sensitivity:", sensitivity)
print("Melanoma Specificity:", specificity)
```

Output:
Melanoma Sensitivity: 0.717948717948718
Melanoma Specificity: 0.6252587991718427

```
def plot_roc_auc(y_true, y_pred):
    """
    This function plots the ROC curves and provides the
    scores."""
    # prepare for
    figureplt.figure()
    fpr, tpr, _ = roc_curve(y_true, y_pred)


    # obtain ROC AUC
    roc_auc = auc(fpr,
    tpr)# print score
    print(f"ROC AUC:
    {roc_auc:.3f}")# plot ROC
    curve
    plt.plot(fpr, tpr, color="blue", lw=2,
             label='ROC curve (area = {f:.2f})'.format(d=1,
    f=roc_auc))plt.xlim([0.0, 1.0])
    plt.ylim([0.0, 1.05])
    plt.xlabel('False Positive
    Rate')plt.ylabel('True
    Positive Rate') plt.title('ROC
    curves')
    plt.legend(loc="lower right")
    plt.show()

plot_roc_auc(y_test, y_pred)
```

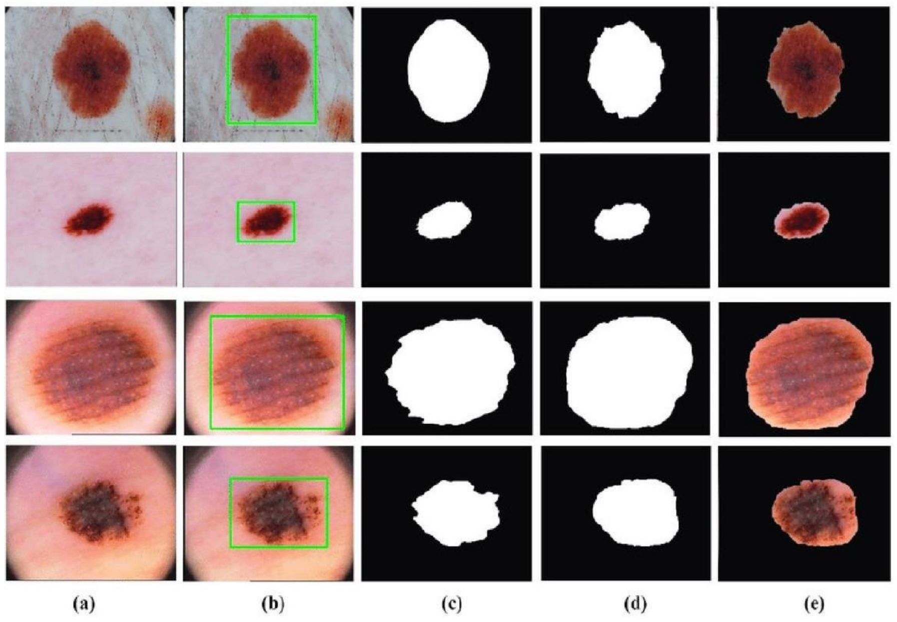
Output:



ROC AUC: 0.671

8.RESULT

The final results are based on the accuracy results in the form of the melanoma and the non-melanomaskin diseases classifications.



9.ADVANTAGES AND DISADVANTAGES

9.1.Advantages

Instant Response, improves prediction of Skin Disease, no referral needed, Saves Money and Time, and Confidential Advice.

9.2.Disadvantages

Network Connectivity and Accuracy.

10.CONCLUSION

We have shown that even without a large dataset and high-quality images, it is possible to achieve sufficient accuracy rates. In addition, we have shown that current state-of-the-art CNN models can outperform models created by previous research, through proper data pre-processing, self-supervised learning, transfer learning, and special CNN architecture techniques. Furthermore, with accurate segmentation, we gain knowledge of the location of the disease, which is useful in the pre-processing of data used in classification, as it allows the CNN model to focus on the area of interest. Lastly, unlike previous studies, our method provides a solution to classify multiple diseases within a single image. With higher quality and a larger quantity of data, it will be viable to use state-of-the-art models to enable the use of CAD in the field of dermatology.

11.FUTURE SCOPE

This implementation of the Structural Co-Occurrence matrices for feature extraction in the skin diseases classification and the pre-processing techniques are handled by using the Median filter, this filter helps toremove the salt and pepper noise in the image processing; thus, it enhances the quality of the images, and normally, the skin diseases are considered as the risk factor in all over the world. Our proposed approach provides 97% of the classification of the accuracy results while another existing model such asFFT + SCM gives 80%, SVM + SCM gives 83%, KNN + SCM gives 85%, and SCM + CNN gives 82%. Future work is dependent on the increased support vector machine’s accuracy in classifying skin illnesses, and SCM is used to manage the feature extraction technique.