PROJECT REPORT

AI-based localization and classification of skin disease with erythema

Team ID:PNT2022TMID26211

Team Members:

kumaran.S(team leader)

balu.C(team member)

bomminani.S.jeswanth(team member)

jeevanadhan M.L(team member)

INDEX:

1. Introduction

1.1 Project Overview

More than 125 million individuals worldwide suffer from psoriasis, and skin cancer rates have been rising quickly over the past few decades, with melanoma being the most diverse form of the disease. Skin conditions may cause issues in the body, including the transmission of the illness from one person to another, if they are not treated at an early stage.

We are developing a model that is used for the early detection and prevention of psoriasis and skin cancer in order to solve the aforementioned issue. In general, the diagnosis of skin diseases depends on many traits like colour, form, texture, etc. Here, a person can take skin-related pictures, which will subsequently be sent to a trained model. The model examines the image to determine whether or not the subject has a skin condition.

1.2 Purpose

The illnesses are not regarded as skin diseases, and ultraviolet sun rays are the main cause of skin tone problems. Even though skin disease is a common disease for which early identification and classification are crucial for patient success and recovery, dermatologists perform the bulk of non-invasive screening tests simply with the naked eye. It is difficult to create a reliable and effective algorithm for automatically detecting skin disease and its severity because the characteristics of skin images vary widely. Such images must be processed automatically for skin analysis, which calls for a quantitative discriminator to distinguish between the diseases.

2. Literature Survey

2.1 Existing problem

Unresolved public health issue One of the most prevalent health issues in people is skin illness. The significance of these diseases for public health is undervalued given their profound effects on the individual, the family, the social life of patients, and their heavy economic burden.

2.2 References

[1] J. Kawahara and G. Hamarneh, “Multi-resolution-tract CNN with hybrid pretrained and skin-lesion trained layers,” in International Workshop on Machine Learning in Medical Imaging, pp. 164–171, Springer, New York, NY, USA, 2016.

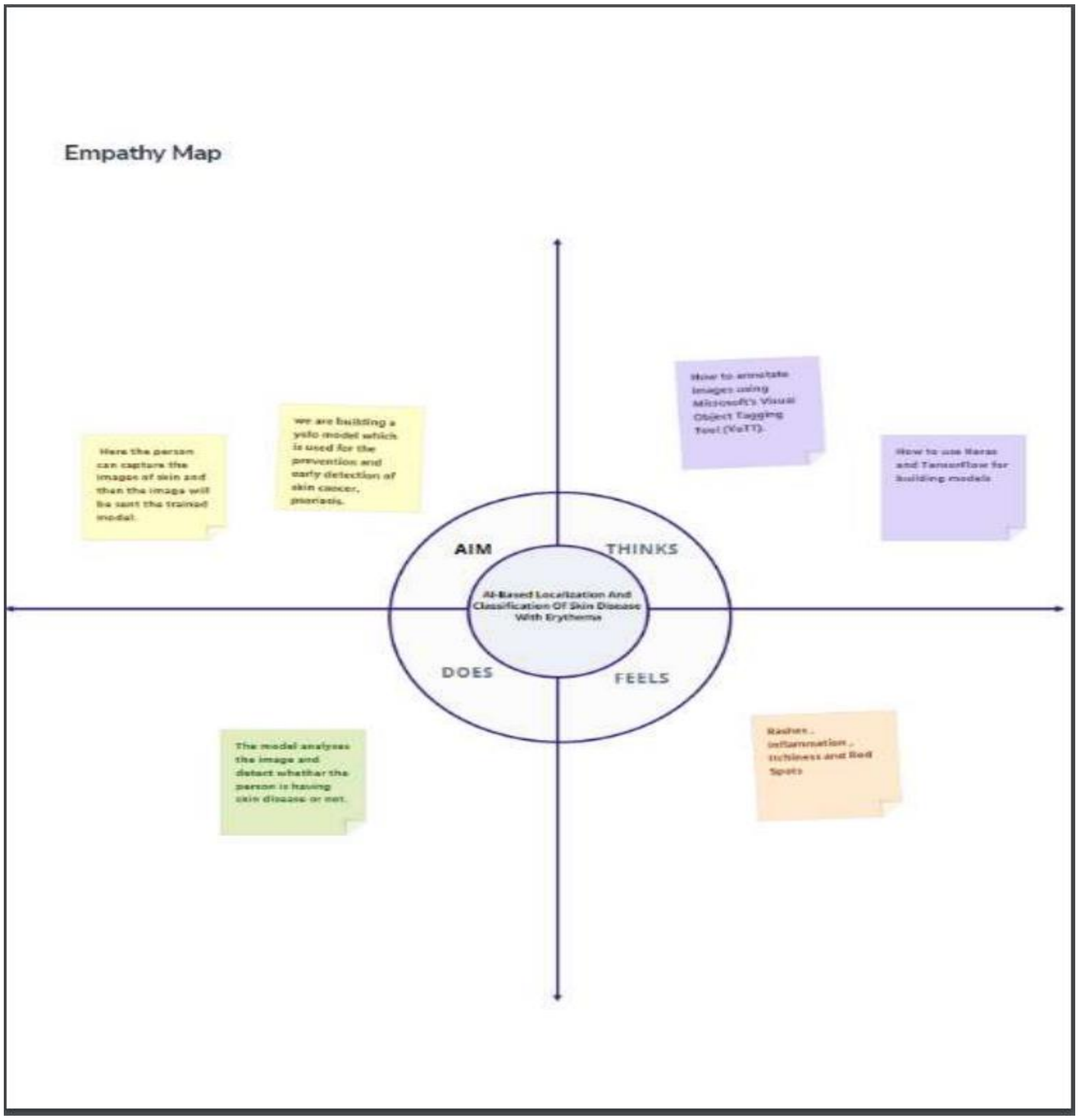
[2] S. Verma, M. A. Razzaque, U. Sangtongdee, C. Arpnikanondt, B. Tassaneetrithep, and A. Hossain, “Digital diagnosis of Hand, Foot, and mouth disease using hybrid deep neural networks,” IEEE Access, vol. 9, pp. 143481–143494, 2021.

[3] P. P. Rebouças Filho, S. A. Peixoto, R. V. Medeiros da Nobrega´ et al., “Automatic histologically-closer classification of skin lesions,” Computerized Medical Imaging and Graphics, vol. 68, pp. 40–54, 2018.

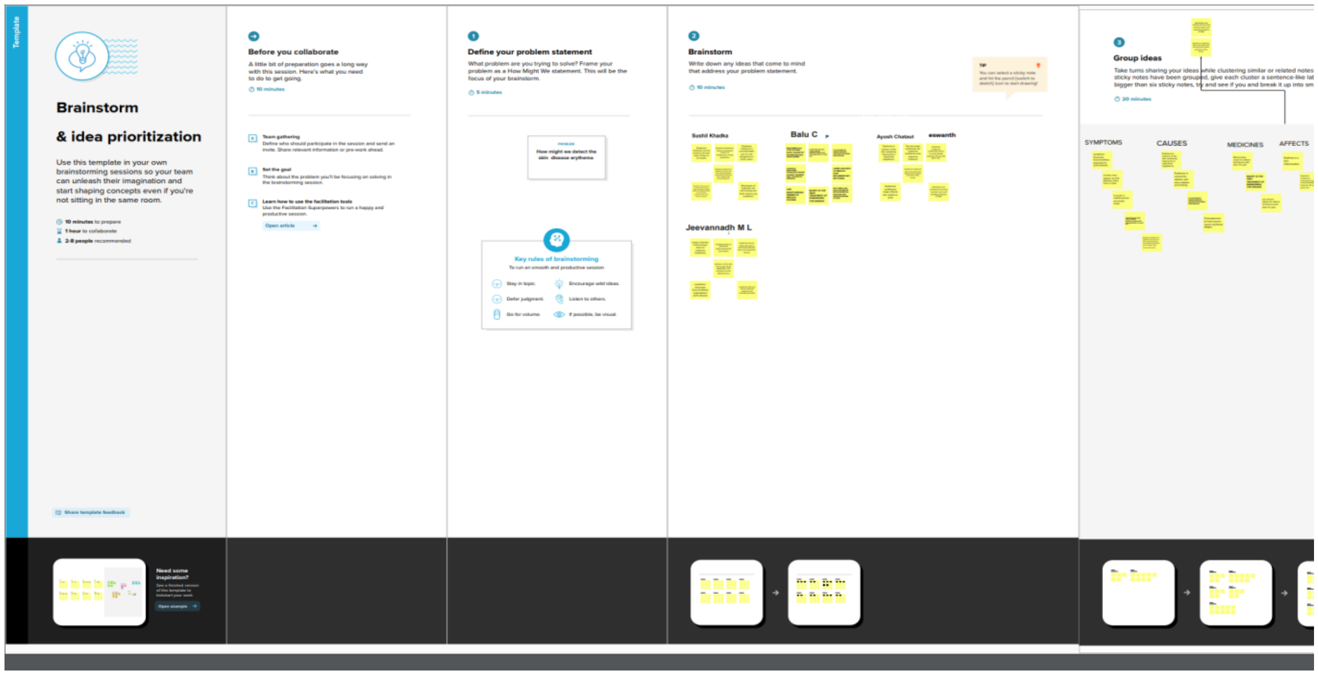
2.3 Problem Statement Definition

Erythema multiforme is an inflammatory skin disorder characterized by patches of red, raised skin that often look like targets and usually are distributed symmetrically over the body. (See also Overview of Hypersensitivity and Reactive Skin Disorders.) • Erythema multiforme is usually caused by a reaction to an infection, usually herpes simplex virus. • Typical symptoms include red patches with purple-gray centers (target lesions) that suddenly appear on the palms and soles, arms and legs, and face and may later spread to the body. • Many people have mouth sores. • The diagnosis is by a doctor's recognition of the target lesions. • This disorder resolves without treatment, but symptoms can be treated with corticosteroids, antihistamines, and anesthetics applied to the skin. • If people have frequent attacks and the doctor thinks herpes simplex virus is the cause, an antiviral drug may help prevent recurrences. Most cases are caused by a reaction to infection with the herpes simplex virus. Some people with erythema multiforme develop herpesvirus cold sores on their lips several days before an attack of erythema multiforme. Less often, cases are caused by bacterial infections (such as mycoplasma), drugs, vaccines, infections with other viruses (especially hepatitis C), and certain noninfectious diseases that affect the immune system, such as systemic lupus erythematosus. Doctors are unsure exactly how erythema multiforme happens, but a type of immune reaction is suspected. Some cases of erythema multiforme do not have a clear cause. Attacks of erythema multiforme may last 2 to 4 weeks. Some people have only one attack, but some have multiple recurrences. Recurrences are common, especially when the cause is herpes simplex virus. The frequency of recurrence usually decreases with time3. Ideation and Proposed Solution

3.1 Empathy Map Canvas


Empathy Map
Here the person can capture the images of skin and then the image will be sent the trained modal.
we are building a yolo model which is used for the prevention and early detection of skin cancer, psoriasis.
How to annotate images using Microsoft's Visual Object Tagging Tool (VoTT).
How to use Keras and Tensorflow for building models
AIM
THINKS
AI-Based Localization And Classification Of Skin Disease With Erythema
DOES
FEELS
The model analyses the image and detect whether the person is having skin disease or not.
Rashes , inflammation , Itchiness and Red Spots

3.2 Ideation and Brainstorming



Template

Brainstorm

& idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

10 minutes to prepare
1 hour to collaborate
2-8 people recommended

Share template feedback

Need some inspiration?
See a finished version of this template to kickstart your work.
Open example

Before you collaborate
A little bit of preparation goes a long way with this session. Here's what you need to do to get going.
10 minutes

A Team gathering
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B Set the goal
Think about the problem you'll be focusing on solving in the brainstorming session.

C Learn how to use the facilitation tools
Use the Facilitation Superpowers to run a happy and productive session.
Open article

1 Define your problem statement
What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.
5 minutes

PROBLEM
How might we detect the skin disease erythema

Key rules of brainstorming
To run an smooth and productive session

Stay in topic.
Encourage wild ideas.
Defer judgment.
Listen to others.
Go for volume.
If possible, be visual.

2 Brainstorm
Write down any ideas that come to mind that address your problem statement.
10 minutes

TIP
You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

Sushil Khadka
Balu C
Ayush Chataut
eswanth
Jeevannadh M L

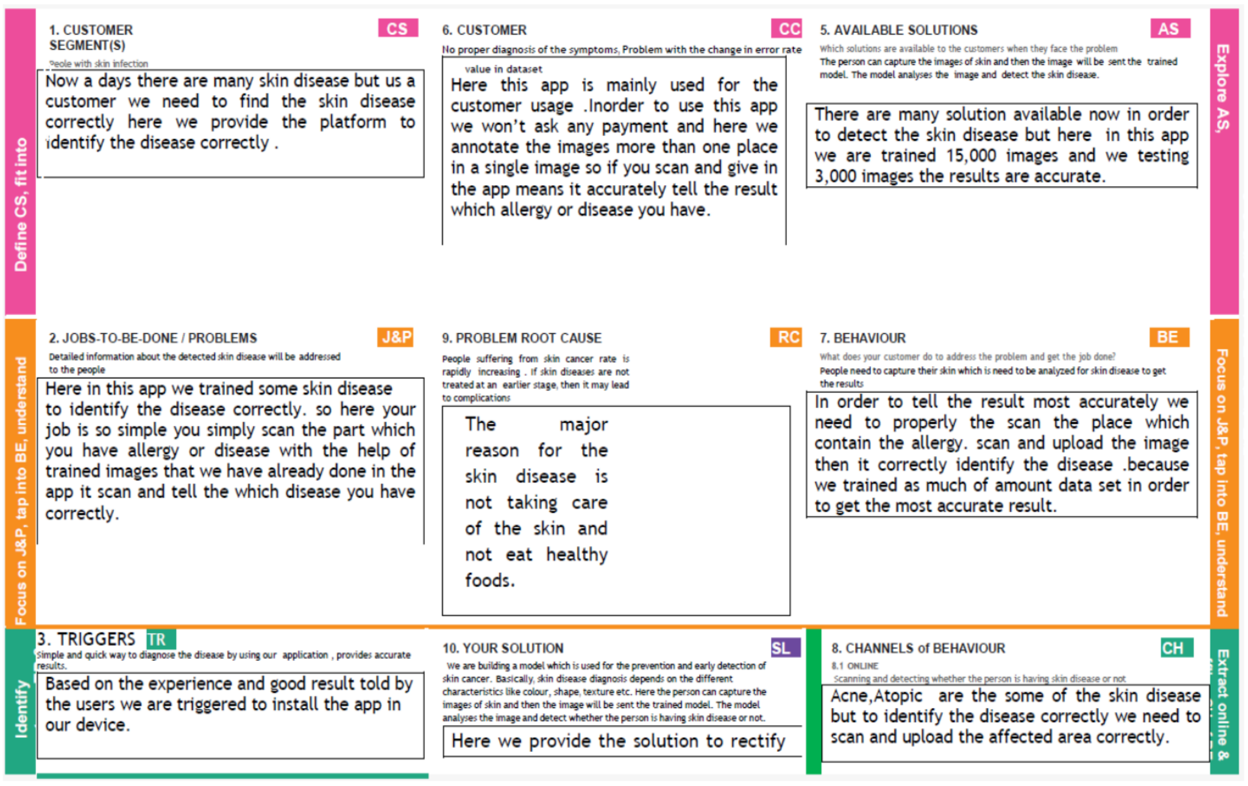
3 Group ideas
Take turns sharing your ideas while clustering similar or related notes sticky notes have been grouped, give each cluster a sentence-like lab bigger than six sticky notes, try and see if you and break it up into sm
20 minutes

SYMPTOMS
CAUSES
MEDICINES
AFFECTS

3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Skin conditions are a growing problem in modern society.
2.	Idea / Solution description	All conditions that irritate, clog, or harm your skin are considered skin diseases, and they can be treated with medicine and good skin care.
3.	Novelty / Uniqueness	There are certain considerations to take into account when it comes to ethnic skin. For individuals with ethnic skin, which usually includes those who are African, American, or Caribbean. 1. Discolorations brought on by dermatitis and acne 2.Melasma 3.Keloid scarring 4.Skin cancer
4.	Social Impact / Customer Satisfaction	Feeling of stress-anxiety ,anger depression,shamesocial isolation,low self –esteem and embarrasment
5.	Business Model (Revenue Model)	Technology is always changing, and this has enormous effects on every industry.
6.	Scalability of the Solution	Using the best design practises, the appropriate framework and medicine, we can boost throughput. Realistic picture processing provides the best user experience.

3.4 Problem Solution


Define CS, fit into

1. CUSTOMER SEGMENT(S) CS

Peole with skin infection

Now a days there are many skin disease but us a customer we need to find the skin disease correctly here we provide the platform to identify the disease correctly .

6. CUSTOMER CC

No proper diagnosis of the symptoms, Problem with the change in error rate value in dataset

Here this app is mainly used for the customer usage .Inorder to use this app we won't ask any payment and here we annotate the images more than one place in a single image so if you scan and give in the app means it accurately tell the result which allergy or disease you have.

5. AVAILABLE SOLUTIONS AS

Which solutions are available to the customers when they face the problem

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 the skin disease.

There are many solution available now in order to detect the skin disease but here in this app we are trained 15,000 images and we testing 3,000 images the results are accurate.

Explore AS,

Focus on J&P, tap into BE, understand

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

Detailed information about the detected skin disease will be addressed to the people

Here in this app we trained some skin disease to identify the disease correctly. so here your job is so simple you simply scan the part which you have allergy or disease with the help of trained images that we have already done in the app it scan and tell the which disease you have correctly.

9. PROBLEM ROOT CAUSE RC

People suffering from skin cancer rate is rapidly increasing . If skin diseases are not treated at an earlier stage, then it may lead to complications

The major reason for the skin disease is not taking care of the skin and not eat healthy foods.

7. BEHAVIOUR BE

What does your customer do to address the problem and get the job done?

People need to capture their skin which is need to be analyzed for skin disease to get the results

In order to tell the result most accurately we need to properly the scan the place which contain the allergy. scan and upload the image then it correctly identify the disease .because we trained as much of amount data set in order to get the most accurate result.

Focus on J&P, tap into BE, understand

Identify

3. TRIGGERS TR

Simple and quick way to diagnose the disease by using our application , provides accurate results.

Based on the experience and good result told by the users we are triggered to install the app in our device.

10. YOUR SOLUTION SL

We are building a model which is used for the prevention and early detection of skin cancer. 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.

Here we provide the solution to rectify

8. CHANNELS of BEHAVIOUR CH

8.1 ONLINE

Scanning and detecting whether the person is having skin disease or not

Acne,Atopic are the some of the skin disease but to identify the disease correctly we need to scan and upload the affected area correctly.

Extract online &

4. Requirement Analysis

4.1 Functional requirements

Acquisition of the image, pre-processing steps like creating a colour gradient in the image, picture cropping, region of interest isolation, thresholding, and clustering, Feature extraction from images, Deep learning and CNN are used in the system training for the YOLO Model for Skin Disease Classification. Application administration, skin disease diagnosis, data retrieval, and data manipulation have separate access.

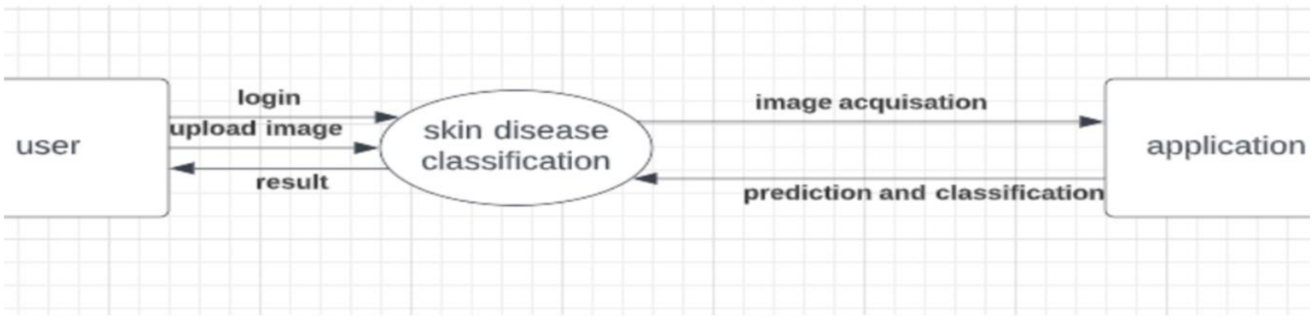
4.2 Non-Functional requirements

Software Quality Attributes, Prediction, Accuracy.

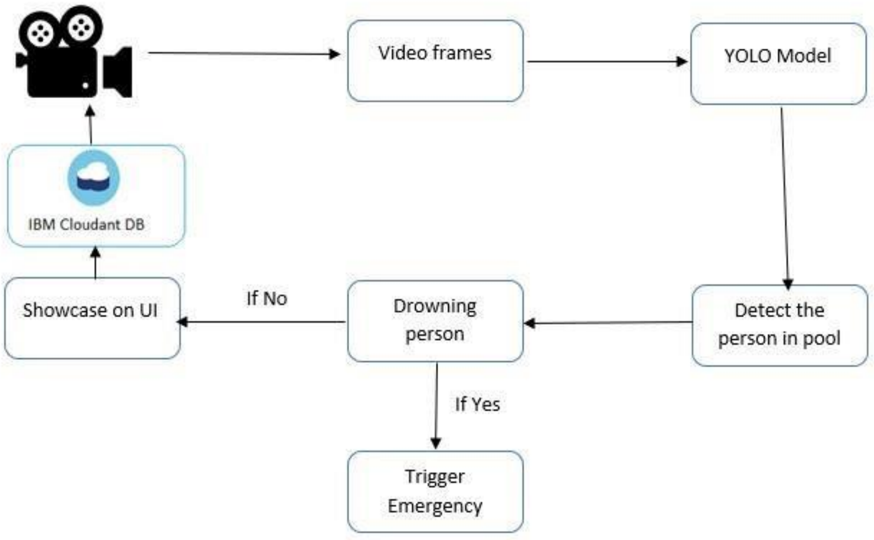
5. Project Design

5.1 Data Flow Diagram

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



5.2 Solution and Technical Architecture



5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
	Login	USN-4	As a user, I can register for the application through Gmail	I can register through Gmail.	Medium	Sprint-1
		USN-5	As a user, I can log into the application by entering email & password	I can also receive logout credential.	High	Sprint-1
	Interface	USN-6	As a user, the interface should be easy to access.	I can receive login credentia	Medium	Sprint-2
PATIENT (Web user)	dashboard	USN-7	As a user I can specify the information(Skin color,skin tone, skin texture,screening etc)	I can able to know about how depth the disease is.	High	Sprint-1
PATIENT(input)	View manner	USN-8	As a user, I can view disease details in visual representation (images).	I can easily understand by using images visually.	High	Sprint-1
	Color visiblity	USN-9	As a user, I can able to see the skin color due to infected area	I can easily know about the condition of skin color.	High	Sprint-2
	knowledge	USN-10	As a user, I can able to know about the disease details in early stage	I can easily know whether I have disease or not	High	Sprint-1

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Administrator	knowlege	USN-11	An administrator who Is handling the website should update and take care of the application	Admin should monitor the records properly	medium	Sprint-2

6. Project Planning and Scheduling

6.1 Sprint Planning and Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	5	High	Kumaran S Balu C
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application.	1	High	M.L.Jeevanandhan B.S.Jeswanth
Sprint-1		USN-3	As a user, I can take a photo and upload in a Application.	5	High	Kumaran S M.L.Jeevanandhan
Sprint-1		USN-4	As a user, I can receive a result of image I uploaded	2	Medium	Balu.C Kumaran S
Sprint-1		USN-5	As a user, I can consult a doctor according to the result I receive.	7	High	M.L.Jeevanandhan KumaranS
Sprint-2	Dashboard	USN-6	As a user, I can enter the age and professional	2	Medium	Balu C B.S.Jeswanth
Sprint-2		USN-7	As a user, I can upload a photo also from a gallery and drive	3	Low	Balu.C
Sprint-2		USN-8	As a user, I can re upload a photo whether it show a some error	15	Medium	Kumaran S Balu C

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-3		USN-9	As a user, I can re upload a photo whether it show a some error	15	Low	Kumaran S Balu C
Sprint-3		USN-10	As a user, I can download the result of application for the uploaded image by using a download button	5	High	Kumaran S M.L.Jeevanandhan
Sprint-4		USN-11	As a admin,I can maintain data of the user safely	2	HIgh	Balu.C Kumaran S
Sprint-4		USN-12	As a admin,i can manage a image and updation of disease for train a app regularly.	18	Medium	M.L.Jeevanandhan KumaranS

6.2 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

7. Coding and Solutioning

pip3 install tensorflow tensorflow_hub matplotlib seaborn numpy pandas sklearn imblearn

import tensorflow as tf import

tensorflow_hub as hub import

matplotlib.pyplot as plt import

```
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 runs

tf.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/skincancer/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 dataset
download_and_extract_dataset()  # preparing data
# generate CSV metadata file to read img paths and labels from
it def 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"]))
```

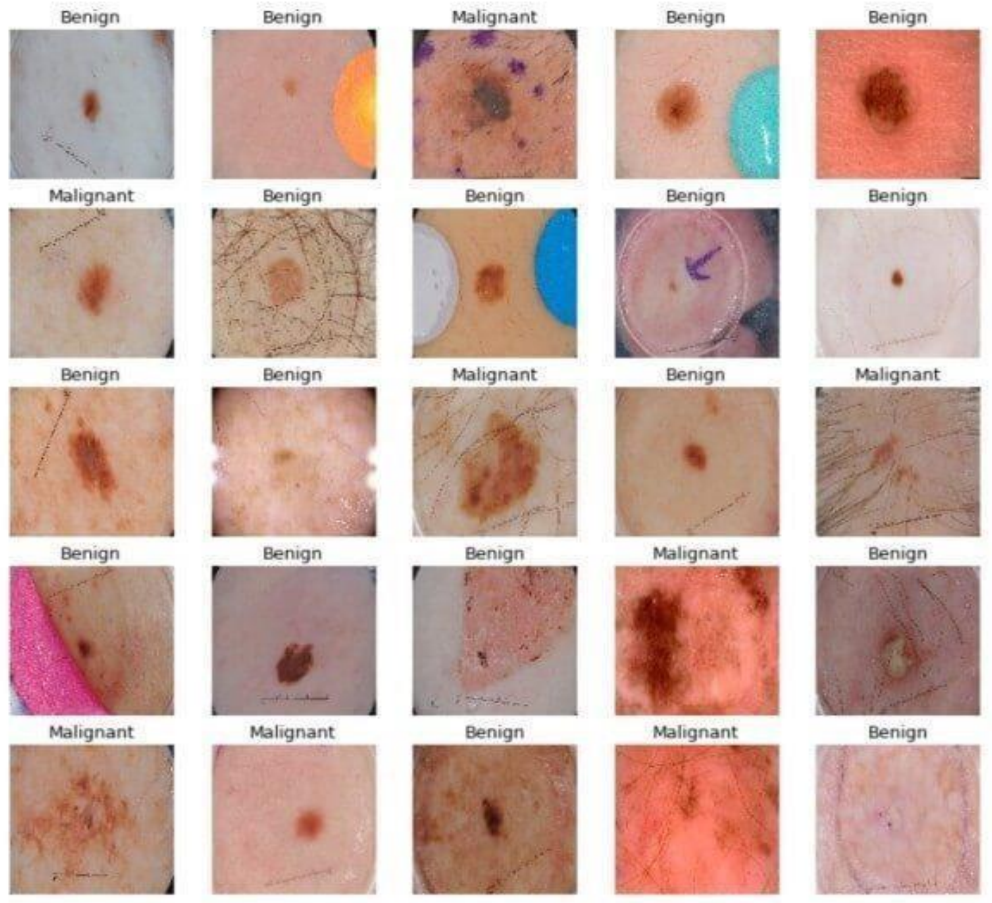
Output:

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

```
# preprocess data def

decode_img(img):


# convert the compressed string to a 3D uint8 tensor   img =

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 string   img =

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
benignvsmalignant_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
benignvsmalignant_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: 0.8281
Epoch 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)
```

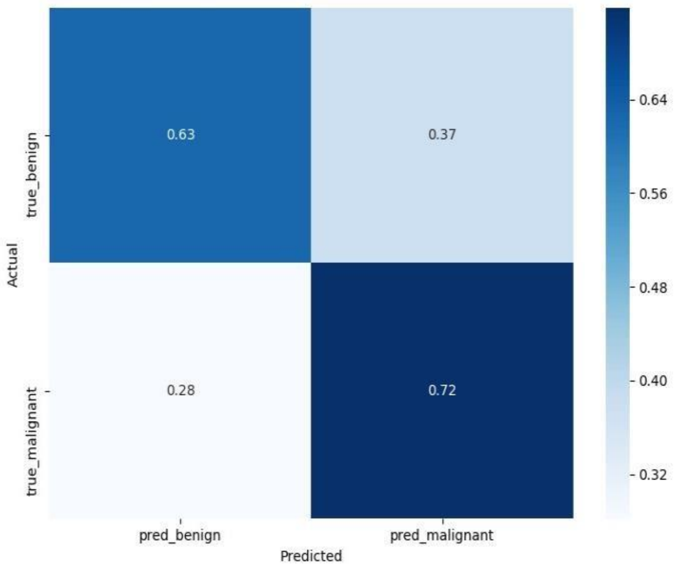
Output:

```
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)
```

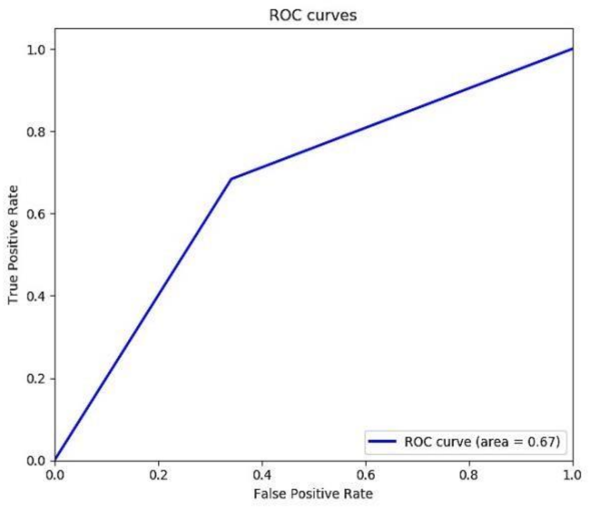
Output:

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 figure     plt.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)
```

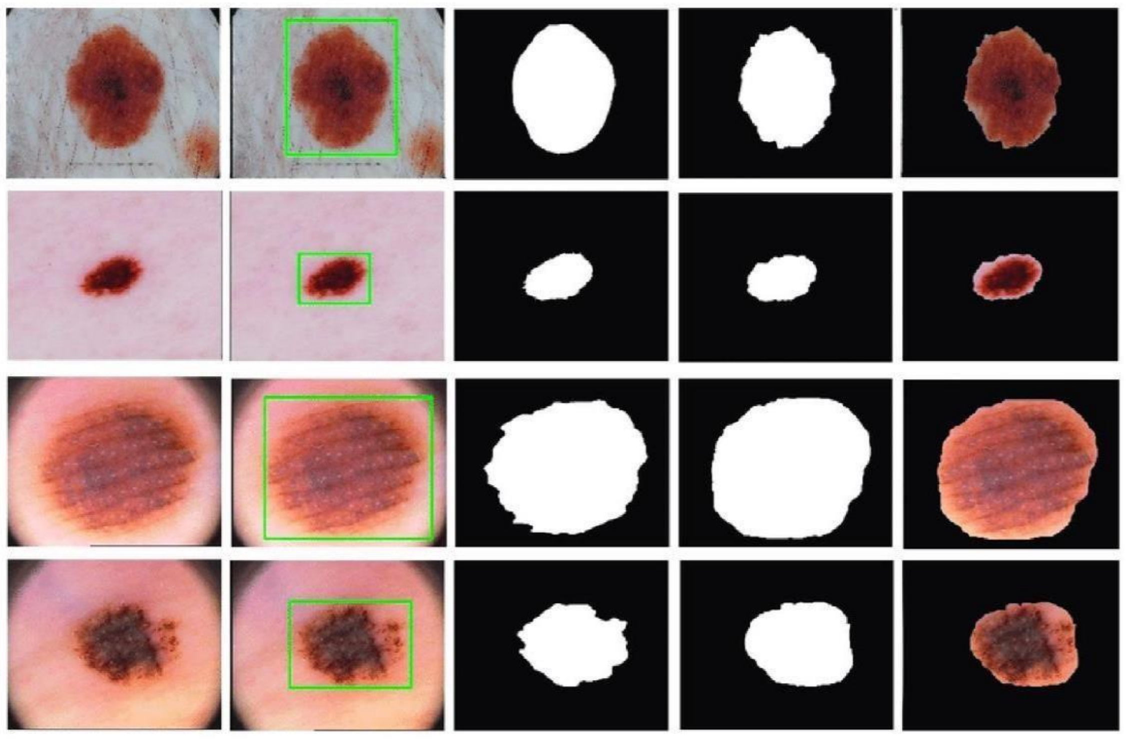
Output:



ROC AUC: 0.671

8. Results

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



9. Advantages andDisadvantages

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 in the absence of a substantial dataset and high-quality images, sufficient accuracy rates can be reached. Furthermore, we have shown that state-of-the-art CNN models can outperform models created by earlier research when proper data pre-processing, self-supervised learning, transfer learning, and special CNN design approaches are used. The location of the disease can also be determined with correct segmentation, which is useful for pre-processing the data required for classification and enables the CNN model to concentrate on the pertinent area. Last but not least, in contrast to past studies, our method enables the classification of many diseases within a single image. Modern modelling techniques will make it possible to use more data that is of higher quality. of CAD in the field of dermatology.

11. Future Scope

The Median filter is used in this implementation of the Structural Co-Occurrence matrices for feature extraction in the classification of skin diseases and the pre-processing techniques. This filter aids in the removal of salt and pepper noise in image processing, improving image quality. Normally, skin diseases are regarded as a risk factor around the world. While other current models like FFT + SCM, SVM + SCM, KNN + SCM, and SCM + CNN only achieve 80%, 83%, 85%, and 82% of the classification accuracy outcomes, respectively, our proposed technique achieves 97%. SCM is used to control the feature extraction technique, and future work will depend on how accurately it can diagnose skin disorders.

12. Appendix

GitHub Link: https://github.com/IBM-EPBL/IBM-Project-4138-1658721256

Team Id: PNT2022TMID26211