Meta
Scifor Technologies

Script. Sculpt. Socialize

A

# INTERNSHIP TRAINING REPORT

ON

# Skin Disease Detection

Submitted By

**Shivam Garg (STB02001)**

Submitted In Partial Fulfillment of the Requirement for The Award of

**Intern in Artificial Intelligence / Machine Learning**

Under the Guidance

**Ms. Divija Ameta**
**(Project Guide)**

**Mr. Joseph Antony Kattukaran**
**( Director)**

# CONTENTS

# CERTIFICATE

## Scifor Technologies

This is to certify that Report entitled "**Skin Disease Detection**" which is submitted by **Shivam Garg (STB02001)** in partial fulfillment of the requirement for the award of **Intern in Artificial Intelligence / Machine Learning to Scifor Technologies, Bangalore** is a record of the candidates own work carried out by them under my supervision.

The documentation embodies results of original work, and studies are carried out by the student themselves and the contents of the report do not from the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

# ACKNOWLEDGEMENT

I would like to express my best sense of gratitude & endeavor with respect to **Divija Ameta (Project Guide)** for suggesting the problems of scholarly guidance and expert supervision during the course of this project. Special thanks to **Mr. Joseph Antony Kattukaran( Director)** .

I am very thankful to **Divija Ameta (Project Guide)** the project guide for constant simulated discussion, encouraging new ideas about this project.

**Ms.Divija Ameta**
**(Project Guide)**

## ABSTRACT

Skin diseases are more common than other diseases. Skin diseases may be caused by fungal infection, bacteria, allergy, or viruses, etc. The advancement of lasers and Photonics based medical technology has made it possible to diagnose skin diseases much more quickly and accurately. But the cost of such a diagnosis is still limited and very expensive. So, image processing techniques help to build automated screening systems for dermatology at an initial stage. The extraction of features plays a key role in helping to classify skin diseases. Computer vision has a role in the detection of skin diseases in a variety of techniques. Due to deserts and hot weather, skin diseases are common in Saudi Arabia. This work contributes to the research of skin disease detection. We proposed an image processing-based method to detect skin diseases. This method takes the digital image of disease effect skin area, then uses image analysis to identify the type of disease. Our proposed approach is simple, fast and does not require expensive equipment other than a camera and a computer. The approach works on the inputs of a color image. Then resize the  image to extract features using a pre-trained convolutional neural network. After that classified feature using Multiclass SVM. Finally, the results are shown to the user, including the type of disease, spread, and severity. The system successfully detects 7 different types of skin diseases with an accuracy rate of 88%.

## INTRODUCTION TO THE PROBLEM STATEMENT

The problem statement for skin disease classification using deep learning involves developing a system or model that can accurately classify and identify different types of skin diseases based on input images. The goal is to leverage deep learning techniques to automate the process of diagnosing skin conditions, providing a faster and potentially more accurate means of identification compared to traditional methods.

## DESCRIPTION OF DATASET

HAM10000 ("Human Against Machine with 10000 training images") dataset - a large collection of multi-source dermatoscopic images of pigmented lesions

The dermatoscopic images are collected from different populations, acquired and stored by different modalities. The final dataset consists of 10015 dermatoscopic images.

It has 7 different classes of skin cancer which are listed below :

Melanocytic nevi
Melanoma
Benign keratosis-like lesions
Basal cell carcinoma
Actinic Keratoses and Intraepithelial Carcinoma
Pyogenic Granulomas and Hemorrhage
Dermatofibroma

# CODING

```
import seaborn as sns
import matplotlib.pyplot as plt
from imblearn.over_sampling import RandomOverSampler
import numpy as np
from sklearn.model_selection import train_test_split
import os, cv2
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, Flatten, Dense,
MaxPool2D

import pandas as pd
data = pd.read_csv('/content/drive/MyDrive/hmnist_28_28_RGB.csv')
data.head()

y = data['label']
x = data.drop(columns = ['label'])

tabular_data =
pd.read_csv('/content/drive/MyDrive/HAM10000_metadata.csv')
tabular_data.head()

classes = {4: ('nv', ' melanocytic nevi'), 6: ('mel', 'melanoma'),
2 :('bkl', 'benign keratosis-like lesions'),
1:('bcc' , ' basal cell carcinoma'),
5: ('vasc', ' pyogenic granulomas and hemorrhage'),
0: ('akiec', 'Actinic keratoses and intraepithelial carcinoma'),
3: ('df', 'dermatofibroma')}
```

```
sns.countplot(x = 'dx', data = tabular_data)
plt.xlabel('Disease', size=12)
plt.ylabel('Frequency', size=12)
plt.title('Frequency Distribution of Classes', size=16)

bar, ax = plt.subplots(figsize = (10,10))
plt.pie(tabular_data['sex'].value_counts(), labels =
tabular_data['sex'].value_counts().index, autopct="%.1f%%")
plt.title('Gender of Patient', size=16)

bar, ax = plt.subplots(figsize=(10,10))
sns.histplot(tabular_data['age'])
plt.title('Histogram of Age of Patients', size=16)

value = tabular_data[['localization', 'sex']].value_counts().to_frame()
value.reset_index(level=[1,0 ], inplace=True)
temp = value.rename(columns = {'localization':'location', 0: 'count'})

bar, ax = plt.subplots(figsize = (12, 12))
sns.barplot(x = 'location',  y='count', hue = 'sex', data = temp)
plt.title('Location of disease over Gender', size = 16)
plt.xlabel('Disease', size=12)
plt.ylabel('Frequency/Count', size=12)
plt.xticks(rotation = 90)

oversample = RandomOverSampler()
x,y  = oversample.fit_resample(x,y)

x = np.array(x).reshape(-1,28,28,3)
print('Shape of X :',x.shape)
```

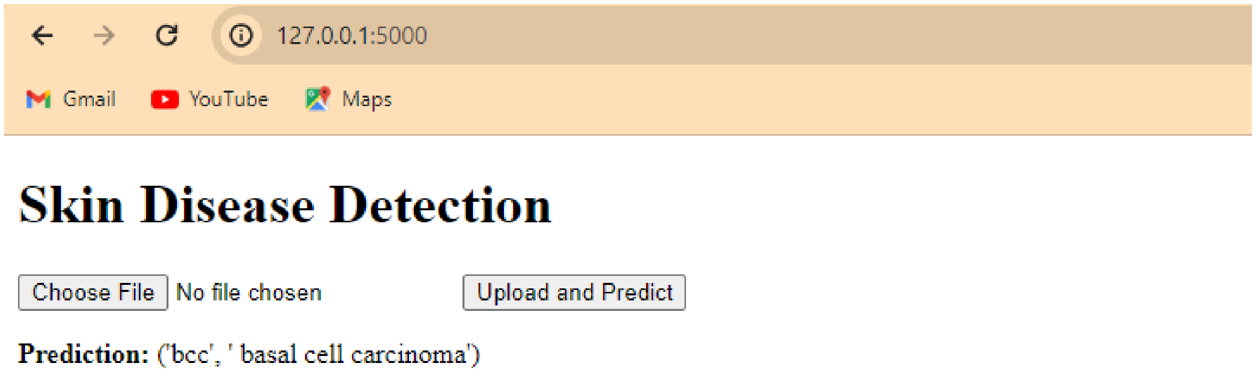
```
x = (x-np.mean(x))/np.std(x)
X_train, X_test, Y_train, Y_test = train_test_split(x,y, test_size=0.2,
random_state=1)

model = Sequential()
model.add(Conv2D(16, kernel_size = (3,3), input_shape = (28, 28, 3),
activation = 'relu', padding = 'same'))
model.add(Conv2D(32, kernel_size = (3,3), activation = 'relu'))
model.add(MaxPool2D(pool_size = (2,2)))
model.add(Conv2D(32, kernel_size = (3,3), activation = 'relu', padding =
'same'))
model.add(Conv2D(64, kernel_size = (3,3), activation = 'relu'))
model.add(MaxPool2D(pool_size = (2,2), padding = 'same'))
model.add(Flatten())
model.add(Dense(64, activation='relu'))
model.add(Dense(32, activation='relu'))
model.add(Dense(7, activation='softmax'))
model.summary()

callback = tf.keras.callbacks.ModelCheckpoint(filepath='best_model.h5',
                                              monitor='val_acc', mode='max',
                                             verbose=1)

model.compile(loss = 'sparse_categorical_crossentropy',
              optimizer = 'adam',
               metrics = ['accuracy'])
history = model.fit(X_train,
                    Y_train,
                    validation_split=0.2,
                    batch_size = 128,
                        epochs = 20,
                     callbacks=[callback])
```

```
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'val'], loc='upper left')
plt.show()

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

model.load_weights('best_model.h5')
loss, acc = model.evaluate(X_test, Y_test, verbose=2)

from google.colab.patches import cv2_imshow
srcdir = '/content/drive/MyDrive/HAM10000_images_part_1'
count=0
for temp in os.listdir(srcdir):
    img = cv2.imread(os.path.join(srcdir, temp))
    cv2.imwrite(temp, img)
    cv2_imshow(img)
    img = cv2.resize(img, (28, 28))
    result = model.predict(img.reshape(1, 28, 28, 3))
    max_prob = max(result[0])
    class_ind = list(result[0]).index(max_prob)
    class_name = classes[class_ind]
    print(class_name)
    count+=1
    if count>10:
        Break

import base64
import numpy as np
import cv2

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, Flatten, Dense,
MaxPool2D

def stringToRGB(base64_string):
    imgdata = base64.b64decode(str(base64_string))
    im_arr = np.frombuffer(imgdata, dtype=np.uint8)
    img = cv2.imdecode(im_arr, flags=cv2.IMREAD_COLOR)
    return img


def get_model():
  model = Sequential()
  model.add(Conv2D(16, kernel_size = (3,3), input_shape = (28, 28, 3),
activation = 'relu', padding = 'same'))
  model.add(Conv2D(32, kernel_size = (3,3), activation = 'relu'))
  model.add(MaxPool2D(pool_size = (2,2)))
  model.add(Conv2D(32, kernel_size = (3,3), activation = 'relu', padding
= 'same'))
  model.add(Conv2D(64, kernel_size = (3,3), activation = 'relu'))
  model.add(MaxPool2D(pool_size = (2,2), padding = 'same'))
  model.add(Flatten())
  model.add(Dense(64, activation='relu'))
  model.add(Dense(32, activation='relu'))
  model.add(Dense(7, activation='softmax'))
  return model

import os
from twilio.rest import Client
from custom.credentials import token, account


def whatsapp_message(token, account, to_number, message):
        client = Client(account, token)
        from_number = 'whatsapp:+14155238886'
        to_number = 'whatsapp:'+ to_number
        client.messages.create(body=message, from_ = from_number, to=
to_number)
```

```
from flask import Flask, request
import socket
import numpy as np
import io
import cv2
import json
import base64

#custom
from custom.credentials import token, account
from custom.essentials import stringToRGB, get_model
from custom.whatsapp import whatsapp_message

'''Get host IP address'''
hostname = socket.gethostname()
IPAddr = socket.gethostbyname(hostname)

app = Flask(__name__)

# Simple http endpoint
@app.route('/get_name', methods = ['GET', 'POST'])
def get_name():
  return 'hello'
  # if request.method == 'POST':
  #   name = request.form.get('name')
  #   return 'your name is '+name

# Simple http endpoint
@app.route('/<string>')
def hello(string):
  return string

@app.route('/encode')
def encode():
  img = 'test_images/test.png'
  image = cv2.imread(img)
  with open(img, 'rb') as f:
    im_b64 = base64.b64encode(f.read())
    encoded_string = base64.b64encode(image)
  return im_b64

@app.route('/disease_detect', methods=["GET", "POST"])
def disease_detect():
  input_string = request.data
  img = json.loads(input_string)

  #taking input from API request
  patient_name = img['patient name']
  doctor_name = img['doctor name']
  patient_number = img['patient number']
  doctor_number = img['doctor number']
  result_img = stringToRGB(img['img'])

  model_name = 'Model/best_model.h5'
  model = get_model()
  model.load_weights(model_name)
  classes = {4: ('nv', ' melanocytic nevi'), 6: ('mel', 'melanoma'), 2 :('bkl',
'benign keratosis-like lesions'), 1:('bcc' , ' basal cell carcinoma'), 5:
('vasc', ' pyogenic granulomas and hemorrhage'), 0: ('akiec', 'Actinic
keratoses and intraepithelial carcinoma'),  3: ('df', 'dermatofibroma')}
```

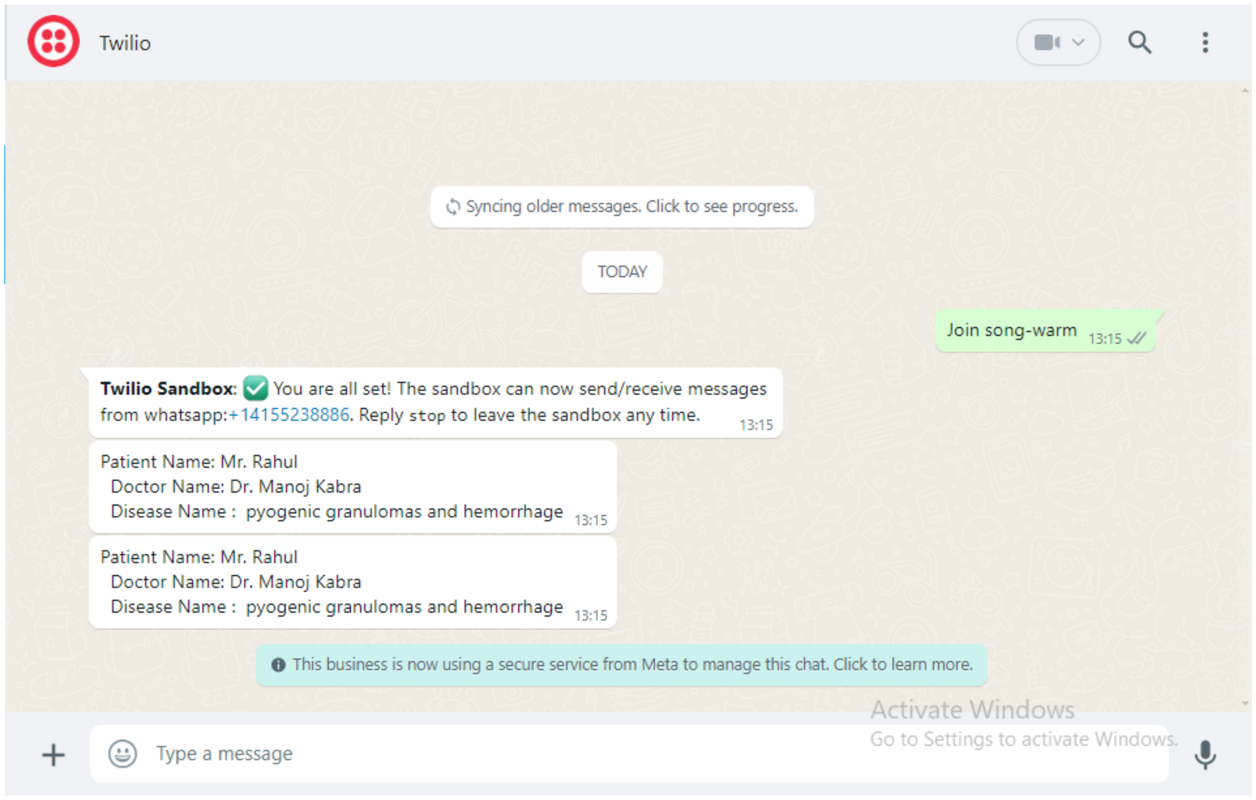
```
img = cv2.resize(result_img, (28, 28))
  result = model.predict(img.reshape(1, 28, 28, 3))
  result = result[0]
  max_prob = max(result)

if max_prob>0.80:
    class_ind = list(result).index(max_prob)
    class_name = classes[class_ind]
    # short_name = class_name[0]
    full_name = class_name[1]
  else:
    full_name = 'No Disease' #if confidence is less than 80 percent then
"No disease"

#whatsapp message
  message = '''
  Patient Name: {}
  Doctor Name: {}
  Disease Name : {}

  '''.format(patient_name, doctor_name, full_name)
  #send whatsapp message to patient
  whatsapp_message(token, account, patient_number, message)
  # sleep(5)
  whatsapp_message(token, account, doctor_number, message)
  return 'Success'

  if __name__ == '__main__':
  # app.debug = True
  app.run(host='0.0.0.0', port=5000, debug=True)
```

# RESULTS



Skin Disease Detection

**Prediction:** ('bcc', ' basal cell carcinoma')


Twilio
Syncing older messages. Click to see progress.
TODAY
Type a message
Activate Windows
Go to Settings to activate Windows.

Join song-warm 13:15

**Twilio Sandbox:** ✅ You are all set! The sandbox can now send/receive messages from whatsapp:+14155238886. Reply `stop` to leave the sandbox any time. 13:15

Patient Name: Mr. Rahul
Doctor Name: Dr. Manoj Kabra
Disease Name : pyogenic granulomas and hemorrhage 13:15

Patient Name: Mr. Rahul
Doctor Name: Dr. Manoj Kabra
Disease Name : pyogenic granulomas and hemorrhage 13:15

This business is now using a secure service from Meta to manage this chat. Click to learn more.

## CONCLUSION

Detection of skin diseases is a very important step to reduce death rates, disease transmission and the development of the skin disease. Clinical procedures to detect skin diseases are very expensive and time-consuming. Image processing techniques help to build automated screening systems for dermatology at an initial stage. The extraction of features plays a key role in helping to classify skin diseases.

## FUTURE SCOPE

Jason Fried says, “When is your product or service finished? When should you put it out on the market? When is it safe to let people have it? Probably a lot sooner than you are comfortable with. Once your product does what it needs to do, get it out there [7]. Just because you have still got a list of things to do does not mean it is not done. Do not hold everything else up because of a few leftovers. You can do them later. And doing them later may mean doing them better, too. [7]. There are many enhancements and extensions which will be added in the future, first, the method of detect skin disease must be on the mobile application developed, then detection the skin lesion in Dermis layer of the skin, finally must detect all the skin disease in the world and degree of disease.

# References

[1] Arifin, S., Kibria, G., Firoze, A., Amini, A., & Yan, H. (2012) “Dermatological Disease Diagnosis Using Color-Skin Images.” Xian: International Conference on Machine Learning and Cybernetics.

[2] Yasir, R., Rahman, A., & Ahmed, N. (2014) “Dermatological Disease Detection using Image Processing and Artificial Neural Network. “Dhaka: International Conference on Electrical and Computer Engineering.

[3] Santy, A., & Joseph, R. (2015) “Segmentation Methods for Computer Aided Melanoma Detection.” Global Conference on Communication Technologies.

[4] Zeljkovic, V., Druzgalski, C., Bojic-Minic, S., Tameze, C., & Mayorga, P. (2015) “ Supplemental Melanoma Diagnosis for Darker Skin Complexion Gradients.” Pan American Health Care Exchanges.

[5] Suganya R. (2016) “An Automated Computer Aided Diagnosis of Skin Lesions Detection and Classification for Dermoscopy Images.” International Conference on Recent Trends in Information Technology.

[6] Alam, N., Munia, T., Tavakolian, K., Vasefi, V., MacKinnon, N., & Fazel-Rezai, R. (2016) “Automatic Detection and Severity Measurement of Eczema Using Image Processing.” IEEE.

[7] SOMMERVILLE, I., “Software Engineering”. 9th 2011.