A Project Report on

Automated Skin Lesion Analyzer

Submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Engineering

in

Information Technology

by

Hrithik Prasad(18104038)
Aslam Pathan(18104057)
Zubair Mangral(17104072)

Under the Guidance of

Prof. Ganesh Gourshete

Department of Information Technology
NBA Accredited
A.P. Shah Institute of Technology
G.B.Road,Kasarvadavli, Thane(W), Mumbai-400615
UNIVERSITY OF MUMBAI

Academic Year 2021-2022

Approval Sheet

This Project Report entitled ***"Automated Skin Lesion Analyzer"*** Submitted by ***"Hrithik Prasad"(18104038),"Aslam Pathan"(18104057),"Zubair Mangral"(17104072),"***is approved for the partial fulfillment of the requirenment for the award of the degree of ***Bachelor of Engineering*** in ***Information Technology*** from ***University of Mumbai***.

(Prof. Ganesh Gourshete)
Guide

Prof. Kiran Deshpande
Head Department of Information Technology

Place:A.P.Shah Institute of Technology, Thane
Date:

CERTIFICATE

This is to certify that the project entitled ***"Automated Skin Lesion Analyzer"*** submitted by ***"Hrithik Prasad" (18104038),"Aslam Pathan" (18104057),"Zubair Mangral" (17104072)*** for the partial fulfillment of the requirement for award of a degree ***Bachelor of Engineering*** in ***Information Technology***,to the University of Mumbai,is a bonafide work carried out during academic year 2021-2022.

(Prof. Ganesh Gourshete)
Guide

Prof. Kiran Deshpande
Head Department of Information Technology

Dr. Uttam D.Kolekar
Principal

External Examiner(s)

1.

2.

Place: A.P.Shah Institute of Technology, Thane
Date:

Acknowledgement

We have great pleasure in presenting the report on **Automated Skin Lesion Analyzer** We take this opportunity to express our sincere thanks towards our guide **Prof. Ganesh Gourshete** Department of IT, APSIT thane for providing the technical guidelines and suggestions regarding line of work. We would like to express our gratitude towards his constant encouragement, support and guidance through the development of project.

We thank **Prof. Kiran B. Deshpande** Head of Department,IT, APSIT for his encouragement during progress meeting and providing guidelines to write this report.

We thank **Prof. Vishal S. Badgujar** BE project co-ordinator, Department of IT, APSIT for being encouraging throughout the course and for guidance.

We also thank the entire staff of APSIT for their invaluable help rendered during the course of this work. We wish to express our deep gratitude towards all our colleagues of APSIT for their encouragement.

Student Name1:Hrithik Prasad
Student ID1:18104038

Student Name2:Aslam Pathan
Student ID2:18104057

Student Name3:Zubair Mangral
Student ID3:17104072

Declaration

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, We have adequately cited and referenced the original sources. We also declare that We have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Hrithik Prasad 18104038)

(Aslam Pathan 18104057)

(Zubair Mangral 17104072)

Date:

Abstract

Dermatology is one of the most unpredictable and difficult terrains to diagnose due its complexity. In the field of dermatology, many a times extensive tests are to be carried out so as to decide upon the skin condition the patient may be facing. The time may vary from practitioner to practitioner.
This is also based on the experience of that person too. So, there is a need of a system which can diagnose the skin diseases without any of these constraints.We propose an automated image based system for recognition of skin diseases using machine learning classification.This system will utilize computational technique to analyze, process, and relegate the image data predicated on various features of the images.
Skin images are filtered to remove unwanted noise and also process it for enhancement of the image. Feature extraction using complex techniques such as Convolutional Neural Network (CNN), classify the image based on the algorithm of softmax classifier and obtain the diagnosis report as an output. This system will give more accuracy and will generate results faster than the traditional method, making this application an efficient and dependable system for dermatological disease detection.

Contents

List of Figures

List of Abbreviations

ML:	Machine Learning
akiec:	Comma Seeperated Value File
bcc:	basal cell carcinoma
bkl:	benign keratosis-like lesions
df:	dermatofibroma
mel:	melanoma
nv:	melanocytic nevi
vasc:	vascular lesions
PDF:	Portable Document Format
CNN:	Convolutional Neural Networks Algorithm
ECS:	Escape key
CSV:	Comma Seeperated Value File

Chapter 1

Introduction

Our proposed system for this project is to use a deep learning architecture which is more suitable for mobile and embedded based vision applications where there is lack of computation power and then convert the whole model into tensorflow.js format to deploy in the production environment so that users can access the end product any where around the world. Diagnosis of skin diseases is a problem on which research is going on since last 5-10 years. People have tried so many approaches to solve this problem. This problem using image processing techniques by feature extraction and segmentation.since this is method is related to fine tuning it is not so reliable process to detect skin diseases as the relation between skin and its diseases can not be catched with fine tuned models. Using 3 layer neural network model which takes inputs as colour, area, shape and other hard typed properties. which also need human assistance to get reliable inputs so this lacks for complete automated system for diagnosis..
Skin disease typically results from environmental factors along with other causes. The necessary tools required for early detection of these diseases are still not readily available in most populations globally. Here the proposed paper provides an approach to detect various kinds of these diseases. The user gives input of the skin disease image, which then the system processes, does feature extraction using CNN algorithm and use softmax image classifier to diagnose diseases. If no disease is found, the system provides a negative result. Thus in this paper, a novel dermoscopy detection and classification method based on Convolutional Neural network (CNN) is proposed.

1.1 Design Approach and details

Our proposed system comprises of data preprocessing stage, training stage, evaluation stage. During data preprocessing stage we remove duplicates from obtained image data set and augment the existing image set for better accuracy rates. In model training stage we have built input pipelines and customize mobile net by adding dense layers for classification of skin disease and adding adaptive learning techniques for faster approach to global minimum. In evaluation stage we generate report with confusion matrix, training curves with different top accuracies.

Chapter 2

Literature Review

In [1] this paper, the authors proposed a model to use medical imaging to detect skin lesion in input skin images. They used methodology to create a prototype system to detect skin disease. The objective of this project is to identify skin lesion based on the input skin images texture analysis based on thresholding and neural network to detect and diagnose skin disease.

In[2] this research paper, the input images obtained from the user is processed to predict skin disease presence or absence from a new input image. The input image of a user would be obtained using android application. In this system, the application would ask the user with many questions and disease type is predicted using the end user answers. At last the proposed system, suggests medicinal descriptions, surgery and medicinal drug based on the skin disease trained model. Skin diseases like Eczema, Fungal infection and Urticaria are been analyzed in this project. This question and answer based application doesn't provide promising results every time.

In [3]This paper briefs importance of skin disease as skin diseases are most common now a days as skin allergies are keep on increasing because of the environmental changes. In this paper, both image processing and data mining techniques has been proposed. Experimental results are performed using matlab tool. The input images are obtained from the dataset.

Sr. No.	Authors	Paper Title	Methodologies	Findings
1	Santhi H, Gopichand G, K.Pavan Koushik, A.Nithin Krishna, D. Sai Tharun	An Automated Skin Lesion Analyser Using CNN with Adaptive Learning (2019)	Deep learning algorithm that includes convolutional neural network and residual network and to minimize the loss function of algorithm with respect to time we have used adaptive learning strategy so that w can obtain global minima of the loss function in comparatively less time	1.Predict the disease up to 95% accuracy but due to small datasets may have bad prediction after few years 2.Disease that can be predicted are Melanocytic nevi, Melanoma, Benign keratosis, Basal cell carcinoma, Actinic Keratoses, Vascular skin lesions, Dermato fibroma
2	J. Rathod, V. Waghmode, A.Sodha and P.Bhavatha nkar	Diagnosis of skin diseases using Convolutional Neural Networks (2018)	1.Computational technique to analyse, process, and relegate the image data 2.Skin images are filtered to remove unwanted noise. 3.Feature extraction using CNN 4.Classify the image based on the algorithm of softmax classifier and obtain the diagnosis report as an output	1.Initial training gives the accuracy of 70% approximately. This can be increased by increasing the training data set. 2.Five diseases were initial tested, which can be further increased in the future. A large data set can increase the accuracy to more than 90 percent

Figure 2.1: Literature Review

Chapter 3

Objectives

3.1 We have identified following as our objectives:

1. To classify seven types of cancer like Actinic keratoses and intraepithelial carcinoma ,basal cell carcinoma (bcc),benign keratosis-like lesions (bkl),dermatofibroma (df),melanoma (mel),melanocytic nevi (nv),vascular lesions (angiomas, angiokeratomas, pyogenic granulomas and hemorrhage, vasc).

2. To implement a deep learning model that predicts with high accuracy.

3. To reduce Human efforts efficient: The model reduces the human efforts, cost as well as errors without time consuming and with accuracy.

4. To deploy our trained model in our web application to serve it online.

5. To make User-friendly website: It is made possible to provide normal citizen an easy way for detecting skin disease.

Chapter 4

Project Design

4.1 Existing System Architecture

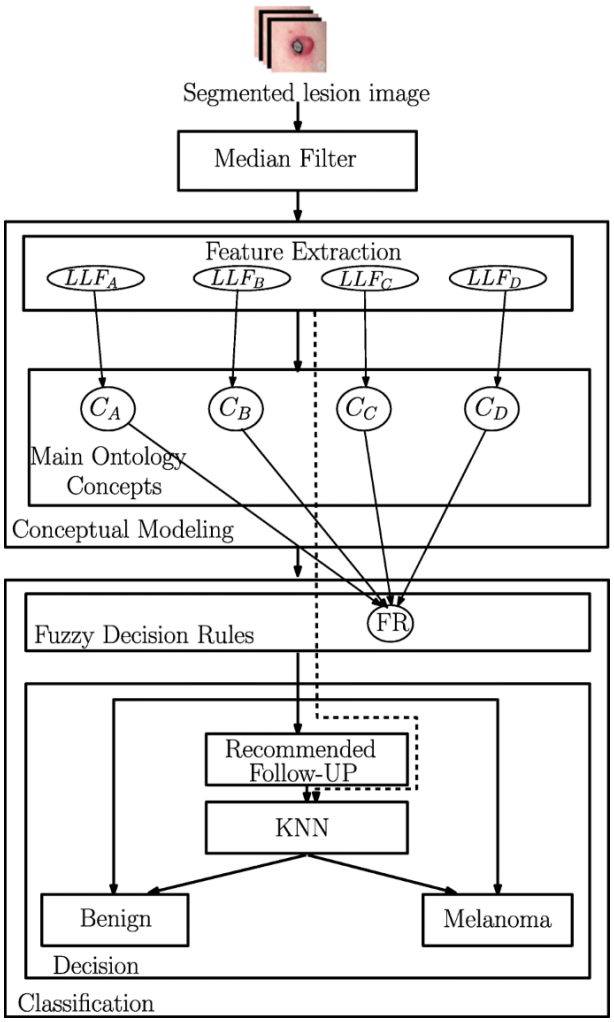

Figure 4.1: Existing System Architecture

Skin cancer classification can be done with the identification of his algorithm such that the test samples and the training samples are loaded in the databases. Samples are cate-

gorized by evaluating the nearest diameter to the preparation case. kNN classifier expands this suggestion by captivating the k adjacent position with proclaiming the indication of the mainstream. It is unique to choose k values. Values which are greater in the value of k help in reducing the effects of noisy levels in the pixels rate inside the training data set and opt the value of k is frequently executed during cross-validation. Here the several methods obtainable for this problem is to select a subset of the training data such that classification by the 1-NN rule using the values of many subsets.

4.1.1 Segmented lesion Image:

Image segmentation involves partitioning an input image into different segments with strong correlation with the region of interest (RoI) in the given image [1, 2]. The aim of medical image segmentation is to represent a given input image in a meaningful form to study the anatomy, identify the region of interest (RoI), measure the volume of tissue to measure the size of tumor, and help in the deciding the dose of medicine, planning of treatment prior to applying radiation therapy, or calculating the radiation dose. Image segmentation helps in analysis of medical images by highlighting the region of interest.

4.1.2 Feature extraction can be accomplished manually or automatically:

1)Manual feature extraction requires identifying and describing the features that are relevant for a given problem and implementing a way to extract those features. In many situations, having a good understanding of the background or domain can help make informed decisions as to which features could be useful. Over decades of research, engineers and scientists have developed feature extraction methods for images, signals, and text. An example of a simple feature is the mean of a window in a signal.

2)Automated feature extraction uses specialized algorithms or deep networks to extract features automatically from signals or images without the need for human intervention. This technique can be very useful when you want to move quickly from raw data to developing machine learning algorithms. Wavelet scattering is an example of automated feature extraction.

With the ascent of deep learning, feature extraction has been largely replaced by the first layers of deep networks – but mostly for image data. For signal and time-series applications, feature extraction remains the first challenge that requires significant expertise before one can build effective predictive models.

4.1.3 Conceptual Modeling:

1) Conceptual models formally describe "aspects of the physical and social world around us for the purposes of understanding and communication". Modern ML proceeds without support from external knowledge sources and relies heavily on the training data and learning algorithms. Conceptual models can represent specific objectives and goals for an ML project.

In addition, the complexity of some of the models (e.g., neural networks, deep learning) has made models opaque and difficult to explain, opening a stream of research in explainable AI. Explainability and transparency are fundamental, particularly in highly regulated fields such as healthcare. For example, it is critical to understanding how a model concluded the likelihood of diagnosis for a patient. Research has explored ways to improve training data by referencing domain ontologies and incorporating rules such as "all birds are animals" into a model. Thus, instead of learning the concept of "animal" from the data, a model can use the domain ontology to infer that an instance labeled as a bird is also an animal. Finally, conceptual models can assist organizations to comprehend the scope of ML in their business operations (i.e., identify the processes affected by ML).

4.2 Proposed System Architecture

In the proposed architecture user gives input of the skin disease image, which then the system processes, does feature extraction using CNN algorithm and use softmax image classifier to diagnose diseases. If no disease is found, the system provides a negative result. Thus in this paper, a novel dermoscopy detection and classification method based on Convolutional Neural network (CNN) is proposed.

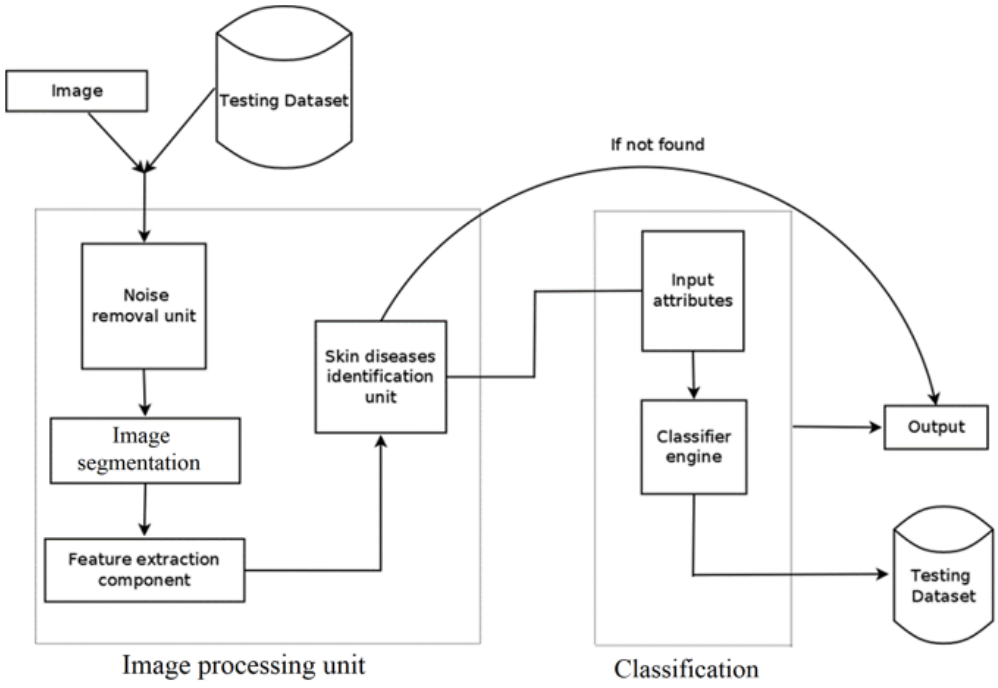

Figure 4.2: Proposed System Architecture

4.2.1 Image Aquisition

mage Acquisition is the first step in any image processing system. The general aim of any image acquisition is to transform an optical image (real-world data) into an array of numerical data which could be later manipulated on a computer. Image acquisition is achieved by suitable cameras. We use different cameras for different applications. If we need an X-ray image, we use a camera (film) that is sensitive to X-rays. If we want an infrared image, we use cameras that are sensitive to infrared radiation. For normal images (family pictures, etc.), we use cameras that are sensitive to the visual spectrum. The images are acquired either through camera or through locally stored device.Whatever might the source, it is veryessential that the input image is clear and precise. For this, a high quality image is required

4.2.2 Image Pre-Processing

Image processing is a way to convert an image to a digital aspect and perform certain functions on it, in order to get an enhanced image or extract other useful information from it. It is a type of signal time when the input is an image, such as a video frame or image and output can be an image or features associated with that image. Usually, the Image Processing system includes treating images as two equal symbols while using the set methods used.It is one of the fastest growing technologies today, with its use in various business sectors. Graphic Design forms the core of the research space within the engineering and computer science industry as well. Image processing basically involves the following three steps.Importing an image with an optical scanner or digital photography.Analysis and image management including data compression and image enhancement and visual detection patterns such as satellite imagery.It produces the final stage where the result can be changed to an image or report based on image analysis.Image processing is a way by which an individual can enhance the quality of an image or gather alerting insights from an image and feed it to an algorithm to predict the later things.

Libraries involved in Image Processing:

1. Scikit-image.
2. OpenCV.
3. Pillow.
4. Matplotlib.
5. SciPy.
6. SimplelTK.
7. Mahotas.

4.2.3 Data storage component to maintain testing and training data images:

The training data should be as close as possible to the data on which predictions are to be made. For example, if your use case involves blurry and low-resolution images (such as from a security camera), your training data should be composed of blurry, low-resolution images. In general, you should also consider providing multiple angles, resolutions, and backgrounds for your training images. AutoML Vision Object Detection models can't generally predict labels that humans can't assign. So, if a human can't be trained to assign labels by looking at the image for 1-2 seconds, the model likely can't be trained to do it either.

4.2.4 Classifier to identify the type of skin disease:

Softmax classifier used here is the last layer of the network that yields actual probability of each label.

The architecture contains two major parts Image processing unit and classification unit. Image processing unit will enhance the image by removing noise and unwanted parts of

the skin and then the image will be segmented into different segments to differentiate from normal skin after that features of the image will be extracted to find out whether skin is infected or not.

1. Noise removal unit: Removes the unwanted pigments and hair from the image.
2. Image enhancement unit and segmentation: Brings the affected part into focus by enhancing the area and also segmenting the area into various segments so as to differentiate it from the normal skin.
3. Feature Extraction Component: Feature extraction is one of the major step in any classification oriented problems. Features are the key for both training and testing purposes. This features contains the important information about the image which will be used to identify the disease.
4. Skin Disease Identification unit: Finds out whether the skin is benign or malignant.
5. Input Attributes: All the important attributes such as asymmetry, border, color, diameter, evolution etc which were extracted from image are now given as a input to Part II that is the classifier part

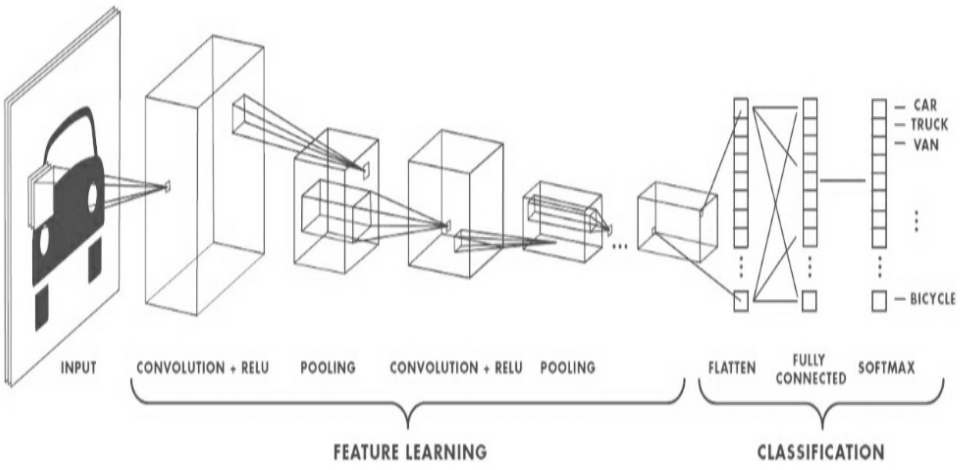

Figure 4.3: Convolutional Neural Network

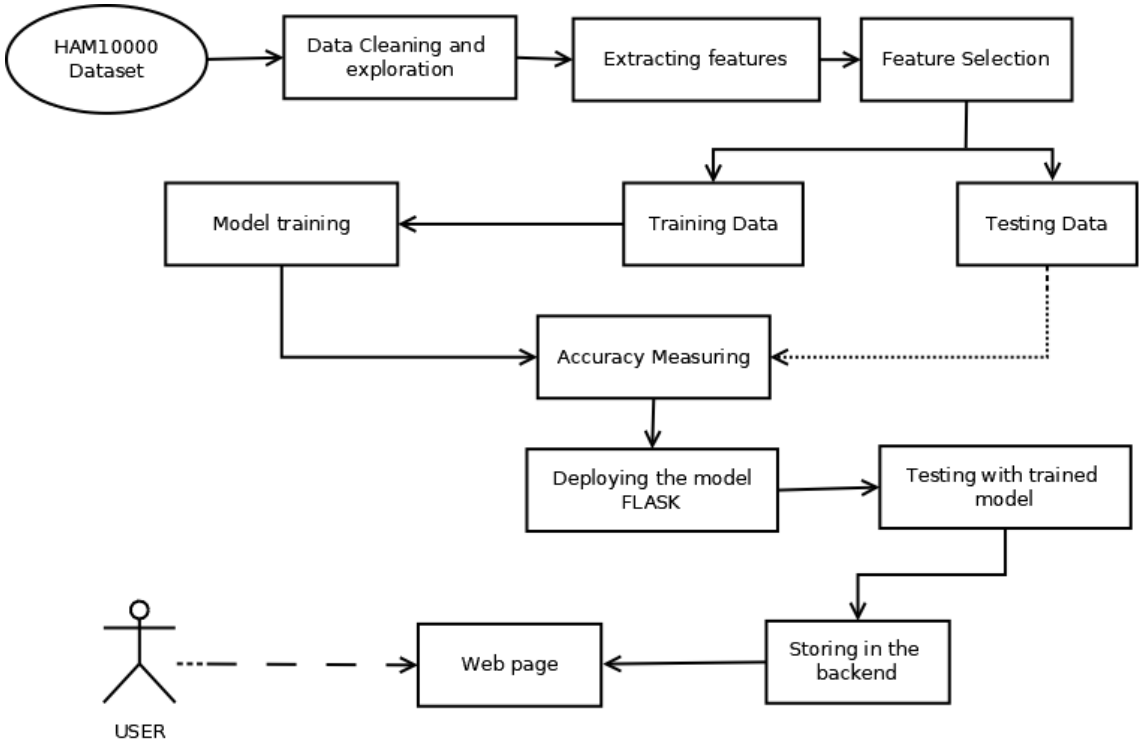

Figure 4.4: Flow Diagram

Chapter 5

Project Implementation

5.1 Imported Libraries

This chapter contains the snapshots of the code written to implement different modules of the system. It starts with creating the main flask app through which the execution starts and then creating a main function which would handle the flow by calling the next functions as needed. Below is the screen shot of our first interface code that is the front-end code from which the execution begins and the user sees the page.

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import os
import pickle
from glob import glob
import seaborn as sns
from PIL import Image

np.random.seed(42)
from sklearn.metrics import confusion_matrix

import keras
from keras.utils.np_utils import to_categorical
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten, Conv2D, MaxPool2D
from sklearn.model_selection import train_test_split
from scipy import stats
from sklearn.preprocessing import LabelEncoder
from keras.preprocessing.image import load_img
from keras.preprocessing.image import img_to_array
```

Figure 5.1: Imported Libraries

Figure 5.1 shows the list of libraries used for the project. These all libraries are necessary in order to implement different web and machine learning (ML) related functions.Out of all the libraries important onces are:

1. numpy: It is very useful for performing mathematical and logical operations on Arrays. So it is used to compute values from the two image arrays of the user for comparing purpose.

2. pillow: It is a library that can process images and videos to identify objects and faces in it.

3. matplotlib: It is used for creating static, animated, and interactive visualizations in Python.

4. Pickle: Pickle is the standard way of serializing objects in Python

5. seaborn: Is is a Python data visualization library based on matplotlib

6. keras: Keras an open-source software library that provides a Python interface for artificial neural networks.

5.2 Data visualization

```
skin_df = pd.read_csv('data/HAM10000/HAM10000_metadata.csv')

SIZE=32

# label encoding to numeric values from text
le = LabelEncoder()
le.fit(skin_df['dx'])
LabelEncoder()
print(list(le.classes_))

skin_df['label'] = le.transform(skin_df["dx"])
print(skin_df.sample(10))


# Data distribution visualization
fig = plt.figure(figsize=(12,8))

ax1 = fig.add_subplot(221)
skin_df['dx'].value_counts().plot(kind='bar', ax=ax1)
ax1.set_ylabel('Count')
ax1.set_title('Cell Type');

ax2 = fig.add_subplot(222)
skin_df['sex'].value_counts().plot(kind='bar', ax=ax2)
ax2.set_ylabel('Count', size=15)
ax2.set_title('Sex');

ax3 = fig.add_subplot(223)
skin_df['localization'].value_counts().plot(kind='bar')
ax3.set_ylabel('Count',size=12)
ax3.set_title('Localization')

ax4 = fig.add_subplot(224)
sample_age = skin_df[pd.notnull(skin_df['age'])]
sns.distplot(sample_age['age'], fit=stats.norm, color='red');
ax4.set_title('Age')
```

Figure 5.2: Data visualization

Figure 5.2 code helps to understand the data visualization through various plotting methods and label encoding. Lbael encoading to convert to numeric values from text

5.3 Balancing data

```
#Balance data.
df_0 = skin_df[skin_df['label'] == 0]
df_1 = skin_df[skin_df['label'] == 1]
df_2 = skin_df[skin_df['label'] == 2]
df_3 = skin_df[skin_df['label'] == 3]
df_4 = skin_df[skin_df['label'] == 4]
df_5 = skin_df[skin_df['label'] == 5]
df_6 = skin_df[skin_df['label'] == 6]


n_samples=500
df_0_balanced = resample(df_0, replace=True, n_samples=n_samples, random_state=42)
df_1_balanced = resample(df_1, replace=True, n_samples=n_samples, random_state=42)
df_2_balanced = resample(df_2, replace=True, n_samples=n_samples, random_state=42)
df_3_balanced = resample(df_3, replace=True, n_samples=n_samples, random_state=42)
df_4_balanced = resample(df_4, replace=True, n_samples=n_samples, random_state=42)
df_5_balanced = resample(df_5, replace=True, n_samples=n_samples, random_state=42)
df_6_balanced = resample(df_6, replace=True, n_samples=n_samples, random_state=42)

#Combined single dataframe
skin_df_balanced = pd.concat([df_0_balanced, df_1_balanced,
                              df_2_balanced, df_3_balanced,
                              df_4_balanced, df_5_balanced, df_6_balanced])

#Balanced
print(skin_df_balanced['label'].value_counts())
```

Figure 5.3: Balancing data

Figure 5.3 Since the data comes naturally heavily imbalanced and with lots of null values. Data balancing is important, here first all the classes with less images are are increased and classes with heavy number are decreased so every class hay a 500 images. Once the data is balanced it is combined into a single dataframe.

5.4 Defining path and train test

```
#reading images based on image ID
image_path = {os.path.splitext(os.path.basename(x))[0]: x
                    for x in glob(os.path.join('data/HAM10000/', '*', '*.jpg'))}

#Defining path and adding in new coloumn
skin_df_balanced['path'] = skin_df['image_id'].map(image_path.get)
#path to read images
skin_df_balanced['image'] = skin_df_balanced['path'].map(lambda x: np.asarray(Image.open(x).resize((SIZE,SIZE))))


n_samples = 5
# Plotting
fig, m_axs = plt.subplots(7, n_samples, figsize = (4*n_samples, 3*7))
for n_axs, (type_name, type_rows) in zip(m_axs,
                                         skin_df_balanced.sort_values(['dx']).groupby('dx')):
    n_axs[0].set_title(type_name)
    for c_ax, (_, c_row) in zip(n_axs, type_rows.sample(n_samples, random_state=1234).iterrows()):
        c_ax.imshow(c_row['image'])
        c_ax.axis('off')

#Convert dataframe column of images into numpy array
X = np.asarray(skin_df_balanced['image'].tolist())
X = X/255.
Y=skin_df_balanced['label']
Y_cat = to_categorical(Y, num_classes=7)
#Split to training and testing
x_train, x_test, y_train, y_test = train_test_split(X, Y_cat, test_size=0.25, random_state=42)
```

Figure 5.4: Defining path train test

Here the images path are defined and added them into new column based on their image ID for that we have used lambda function. converted dataframe cloumn of images into numpy array finall we split them into training and testing

5.5 Machine learning model

```
num_classes = 7

model = Sequential()
model.add(Conv2D(256, (3, 3), activation="relu", input_shape=(SIZE, SIZE, 3)))

model.add(MaxPool2D(pool_size=(2, 2)))
model.add(Dropout(0.3))

model.add(Conv2D(128, (3, 3),activation='relu'))

model.add(MaxPool2D(pool_size=(2, 2)))
model.add(Dropout(0.3))

model.add(Conv2D(64, (3, 3),activation='relu'))

model.add(MaxPool2D(pool_size=(2, 2)))
model.add(Dropout(0.3))
model.add(Flatten())

model.add(Dense(32))
model.add(Dense(7, activation='softmax'))
model.summary()

model.compile(loss='categorical_crossentropy', optimizer='Adam', metrics=['acc']
```

Figure 5.5: Machine learning model

In our sequential model we use Conv2D layer of 256 with relu as an activation layer in second layer maxpool with dropout of 0.3 this occurs two times we las use a flatten with softmax as activation atlast the model is complied to categrical with adam optimizer.

5.6 Index html

```
<!DOCTYPE html>
<html lang="eng">

    <head>

        <link rel="stylesheet" href="style.css">

        <title>Automated Skin Lesion Analyzer</title>
        <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrC

    </head>

    <body>
    <div class="main">
        <div class="navbar">
            <div class="icon">
                <h2 class="logo">Automated Skin Lesion Analyer</h2>
            </div>

            <div class="menu">
                <ul>
                    <li><a href="#">HOME</a></li>
                    <li><a href="#">ABOUT</a></li>
                    <li><a href="#">SERVICE</a></li>
                    <li><a href="#">DESIGN</a></li>
                    <li><a href="#">CONTACT</a></li>
                </ul>
            </div>

            <div class="search">
                <input class="srch" type="search" name="" placeholder="Type To text">
                <a href="#"> <button class="btn">Search</button></a>
            </div>

        </div>
        <div class="content">
            <h1>Skin Cancer <br><span>Detection</span> <br>Course</h1>
            <p class="par">Automated Skin Lesion Analyzer can classify 7 types of dangerous skin cancer disease<br>

                Melanocytic nev(nv), Melanoma(mel),Benign keratosis-like lesions(bkl),<br>Basal cell carcinoma(bcc),Actinic keratose(akiec),Vascular lesions(vaasc),Dermatof
                <br> </p>



    <div class="upload-file">
                <form class = "file-form" action="/" method="post", enctype="multipart/form-data">
                    <input class="file-form-input" type="file" , name = "file"/>
                    <button class="btn btn-success btn-lg">Predict image <button>
                </form>
            </div>

            {% if prediction %}
                <p class="text-center"> Image is a {{prediction}}</p>
```

Figure 5.6: Index html

The above code snippet shows the HTML code for the web page user looks at start which contains some CSS and it uses BootStrap5 which makes it responsive. The next step in the users process of execution is to interact through the web by uploading the affected skin lesion. Here the picture is mandatory. After getting the necessary inputs the user can click on the Predict button which will take him to the next process where it is redirected to results page.

5.7 App.py

```
import keras
from keras_applications import vgg16
from keras.utils.np_utils import to_categorical # used for converting labels to one-hot-encoding
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten, Conv2D, MaxPool2D
from sklearn.model_selection import train_test_split
from scipy import stats
from sklearn.preprocessing import LabelEncoder

from keras.preprocessing.image import load_img
from keras.preprocessing.image import img_to_array
from keras_vggface.vggface import VGGFace
from keras_vggface.utils import preprocess_input
from keras_vggface.utils import decode_predictions



app = Flask(__name__)
model = pickle.load(open('model.pkl','rb'))
UPLOAD_FOLDER = "\SkinLesionAnalyzer\static"

ALLOWED_EXT = set(['jpg' , 'jpeg' , 'png' , 'jfif'])
def allowed_file(filename):
    return '.' in filename and \
           filename.rsplit('.', 1)[1] in ALLOWED_EXT

classes = {
  0: 'akiec, Actinic Keratoses (Solar Keratoses) or intraepithelial Carcinoma (Bowen's disease)',
  1: 'bcc, Basal Cell Carcinoma',
  2: 'bkl, Benign Keratosis',
  3: 'df, Dermatofibroma',
  4: 'mel, Melanoma',
  5: 'nv, Melanocytic Nevi',
  6: 'vasc, Vascular skin lesion'

}

def model_predict(img_path, model):
    img = load_img(img_path, target_size=(224, 224))

    # Preprocessing the image
    x = Image.img_to_array(img)
    x = np.true_divide(x, 255)
    x = np.expand_dims(x, axis=0)

    preds = model.predict(x)
    return preds

@app.route('/', methods=['GET'])
def home():
    return render_template("index.html")

@app.route('/',methods=['POST'])
def predict():
    file = request.files['file']
    image_path = "static/images" + file.filename
    file.save(image_path)

    yhat = model.predict(image_path)
```

Figure 5.7: App.py

This is the stage where the image is extracted from the form and then passed to the preprocess function which in turn returns a single string containing the preprocess data and then we use our predict function to the results of the text depicting as classification of the lesion. This is displayed on the web on the results page which shows the result along with the input which was provided.

Chapter 6

Testing

6.1 Functional Testing

6.1.1 Unit Testing

Out of different standard testing methods available, unit testing has been used as our main testing method. It is a testing process that tests individual components of the application in terms of its functionality and working. In our automated skin lesion analyzer web application, we have few such units in place that perform their respective action based on the user's input. These small units include getting data from user through a form, cleaning the data and passing it to the model, updating results on the next page, give feedback form to take input from user's end. From the user's end it includes providing feedback to the prediction which was done by the machine. And then the next component is retraining based on the feedback received form the user. For all the above-mentioned components we found testing them individually has best suited here. Unit testing has also been used because it helps in finding any issues at earlier stage and in the individual component rather than finding it later in the entire application.

6.1.2 Integration Testing

After each unit is tested individually, it is integrated with other units to create a functioning module that can perform specific tasks. These are then tested as a combined group through integration testing to ensure whole segment of the application behave as a expected. So in this system once the individual components like image processing and image classification is combined and developed together on the single web page which is then tested as a whole.

6.2 Non Functional Testing

6.2.1 Compatibility Testing

Compatibility testing is used to see how an application will work in different environments. It is used to check that the application is compatible with different operating systems, browsers and platforms. The framework we used to develop this application is Flask - Python. It is framework that is used to create web applications. This system being an web application is compatible on any operating system and also supported on different web browsers and devices that have a working webcam and microphone.

6.2.2 Load Testing

Load testing checks how the software behaves under normal and peak conditions. This is done to determine how much work the software can handle before performance is affected. The load can be checked by allowing multiple users to use the system simultaneously to see how the system is performing

Chapter 7

Result

7.1 User Interface

This chapter includes the results which has been produced by the system in terms of different functionalities. These results have been demonstrated in the form of snapshots taken from the working system and describing what each one of them does. The flow in this chapter is then divided into three parts:

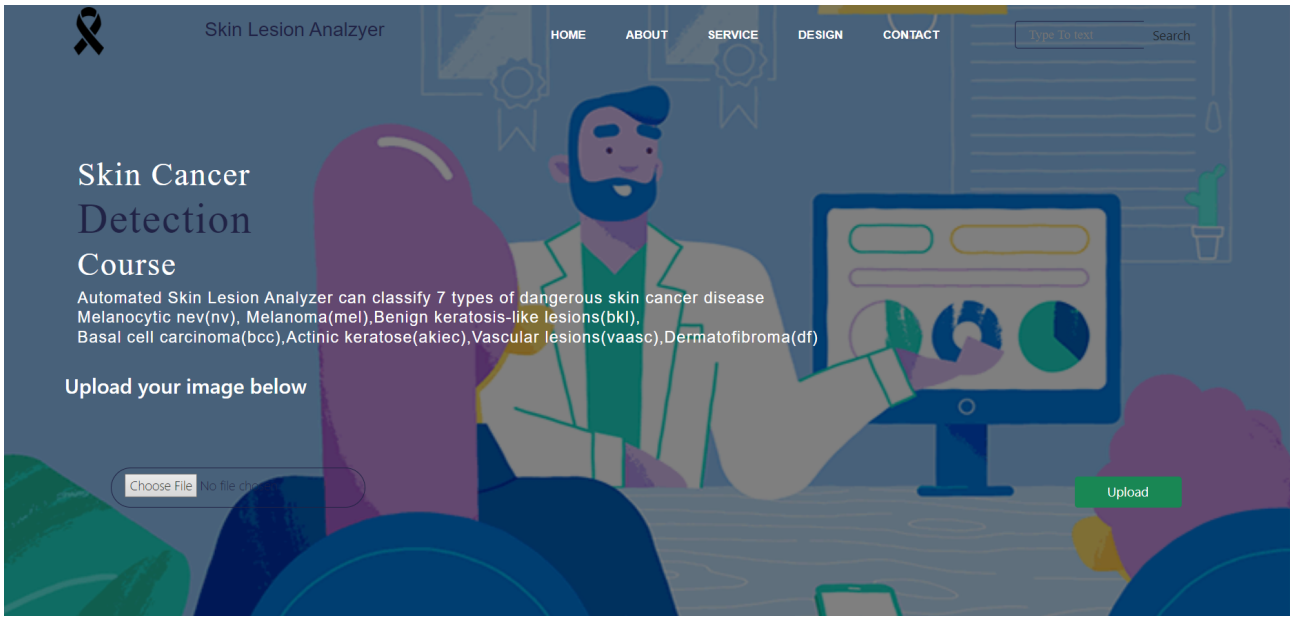

Figure 7.1: User interface

7.2 Taking user image

Figure 7.2: Taking Image from User

7.3 Showing Results

Figure 7.3: Showing Result

Chapter 8

Conclusions and Future Scope

8.1 Conclusion

The proposed system is able to detect the skin disease with promising results combining computer vision and machine learning techniques. It can be used to help people from all over the world and can be used in doing some productive work. The tools used are free to use and are available for the user, hence, the system can be deployed free of cost. The application developed is light-weight and can be used in machines with low system specifications. It has also a simple user interface for the convenience of the user. The image processing and machine learning algorithms were successfully implemented.

8.2 Future Scope

In the future

1. Can be used as a reliable real time teaching tool for medical students in the dermatology stream.
2. Can reduce the human effort.Can be used in rural areas where dermatologist are hard to find.
3. If a patient identifies something suspicious on their skin, it is recommended that they have this spot- checked by a doctor
4. The patient's primary care provider will examine the suspicious spot and note its size, shape, color and texture, and whether there is any active bleeding or scaling of the lesion

Bibliography

[1] Damilola A. Okuboyejo, Oludayo O. Olugbara, and Solomon A. Odunaike, "Automating Skin Disease Diagnosis Using Image Classification," Proceedings of the World Congress on Engineering and Computer Science 2013 Vol II, WCECS 2013, 23-25 October, 2013, San Francisco, USA.

[2] R. Yasir, M. A. Rahman and N. Ahmed, "Dermatological disease detection using image processing and artificial neural network," 8th International Conference on Electrical and Computer Engineering, Dhaka, 2014, pp. 687-690, doi: 10.1109/ICECE.2014.7026918.

[3] Ambad, Pravin S., and A. S. Shirat, "A Image Analysis System to Detect Skin Diseases," IOSR Journal of VLSI and Signal Processing (IOSR-JVSP), Volume 6, Issue 5, Ver. I (Sep. - Oct. 2016), PP 17-25.

[4] R S Gound, Priyanka S Gadre, Jyoti B Gaikwad and Priyanka K Wagh, "Skin Disease Diagnosis System using Image Processing and Data Mining," International Journal of Computer Applications 179(16):38-40, January 2018.

[5] V. B. Kumar, S. S. Kumar and V. Saboo, "Dermatological disease detection using image processing and machine learning," 2016 Third International Conference on Artificial Intelligence and Pattern Recognition (AIPR), Lodz, 2016, pp. 1-6, doi: 10.1109/ICAIPR.2016.7585217.

[6] M. Shamsul Arifin, M. Golam Kibria, A. Firoze, M. Ashraful Amini and Hong Yan, "Dermatological disease diagnosis using color-skin images," 2012 International Conference on Machine Learning and Cybernetics, Xian, 2012, pp. 1675-1680, doi: 10.1109/ICMLC.2012.6359626.

[7] Raja, K.S., Kiruthika, U. An Energy Efficient Method for Secure and Reliable Data Transmission in Wireless Body Area Networks Using RelAODV. Wireless Pers Commun 83, 2975–2997 (2015). https://doi.org/10.1007/s11277-015-2577-x

Chapter 9

Appendices

9.1 Appendix A

Project 1
1)Installing Python and Anaconda
Anaconda, a Python distribution that comes with most of the required machine learning packages that you'll need to use: NumPy, SciPy, Scikit-Learn, Pandas, and many more https://www.anaconda.com/distribution/

2)Installing Spyder IDE
Spyder IDE comes with all the libraries pre-installed so we need not to install tensorflow/keras seperately.

3) Download virtual env
py -m pip install –user virtualenv

4) Creating Virtual Environment
py -m venv env

5) Activating environment
./env/Scripts/activate

Project 2
1)Create a model that predicts the classification of skin lesion out of seven classes:

Steps to perform
Perform exploratory data analysis and feature engineering and then apply feature engineering. Follow up with a deep learning model to predict the class of skin lesion image.

Tasks
1. Feature Transformation Transform categorical values into numerical values (discrete).

2. Exploratory data analysis of different factors of the dataset.

3. Additional Feature Engineering - You will check the correlation between features and will drop those features that have a strong correlation. This will help reduce the number of features and will leave you with the most relevant features.

4. Modeling - After applying Exploratory Data Analysis (EDA) and feature engineering, you are now ready to build the predictive models. In this part, you will create a deep learning

model using Keras with tensorfow backend.

Project 3
Build a Convolutional Neural Network (CNN) model that classifes the given lesion images correctly outof seven classes.
You are provided with the following resources that can be used as inputs for your model:
1. A collection of images of seven class skin lesion. These images are of different sizes with varied lighting conditions.

2. Code template containing the following code blocks:

(a). Import modules (Part 1).

(b). Set hyper parameters (Part 2).

(c). Read image data set (Part 3).

(d). Run tensorfow model (Part 4).

You are expected to write the code for CNN image classifcation model (between Parts 3 and 4) using tensorfow that trains on the data and calculates the accuracy score on the test data.

python app.py

The python program will execute and we will get a localhost url of http://127.0.0.1:5000 Now the application is running which can be accessed in browser.

Publication

Paper entitled **"Automated skin lesion analyzer "** is presented at **"Third International Conference on Intelligent Computing, Instrumentation and Control Technologies/Journal Name: ICICICT -2022"** by **"Hrithik Prasad,Aslam Pathan,Zubair Mangral".**