INTEGRATING DETECTION AND CLASSIFYING SKIN LESION: A COMPREHENSIVE APPROACH BY INTEGRATING IMAGE PROCESSING AND PREDICTIVE MODELLING FOR EARLY DIAGNOSIS AND PROGNOSIS

Report submitted to SASTRA Deemed to be University
As the requirement for the course

CSE300 – MINI PROJECT

Submitted By

Akash R

(Reg.No: 225003009, B.Tech-CSE)

Mohamed Anwardheen M

(Reg.No: 225003089, B.Tech-CSE)

Vignesh S

(Reg.No: 225003170, B.Tech.CSE)

May 2024

SCHOOL OF COMPUTING

THANJAVUR, TAMILNADU, INDIA-613 401

SASTRA

ENGINEERING · MANAGEMENT · LAW · SCIENCES · HUMANITIES · EDUCATION

DEEMED TO BE UNIVERSITY

(U/S 3 of the UGC Act,1956)

THINK MERIT | THINK TRANSPARENCY | THINK SASTRA

THANJAVUR | KUMBAKONAM | CHENNAI

SCHOOL OF COMPUTING

THANJAVUR, TAMILNADU, INDIA-613 401

Bonafide Certificate

This is to certify that the report titled **"Integrating Detection and Classifying Skin Lesion: A Comprehensive Approach by Integrating Image Processing and Predictive Modelling for Early Diagnosis and Prognosis"** submitted as a requirement for the course, CSE300: **MINI PROJECT** for B.Tech. is bonafide record of the work done by **Mr. Akash R (Reg.No: 225003009, B.Tech-CSE), Mr. Mohamed Anwardheen M (Reg.No: 225003089, B.Tech-CSE), Mr. Vignesh S (Reg.No: 225003170, B.Tech-CSE)** during the academic year 2023-24, in the School of Computing, under my supervision.

Signature of Project Supervisor :

Name with Affiliation : **Dr. Ramakrishnan**

Date :

Mini Project *Viva voce* held on _____

Examiner 1 **Examiner 2**

ACKNOWLEDGEMENTS

We pay our sincere obeisance to the God Almighty for his grace and infinite mercy and for showing on us his choicest blessings.

We would like to express our thanks to our Chancellor **Prof. R. Sethuraman,** Vice Chancellor **Dr. S. Vaidhyasubramaniam** and Registrar **Dr. R. Chandramouli** for having given us an opportunity to be a student of this esteemed institution.

We express our deepest thanks to **Dr. V. Ramaswamy**, Dean and **Dr. A. Alli Rani**, Associate Dean, Srinivasa Ramanujan Centre for their constant support and suggestions when required without any reservations.

We express our gratitude to HOD in charge **Dr. V.Kalaichelvi**/ACP/CSE, Srinivasa Ramanujan Centre for his constant support and valuable suggestion for the completion of the project.

We exhibit our pleasure in expressing our **Dr**, **R.Ramakrishnan/Sr.AP/CSE**, our guide for her ever-encouraging spirit and meticulous guidance for the completion of the project.
,

We would like to place on record the benevolent approach and pain taking efforts of guidance and correction of **Dr. G Revathy APIII/CSE,** the project coordinator and all department staffs to whom we owe our hearty thanks for ever.

Without the support of our parents and friends this project would never have become reality. We dedicate this work to our well-wishers, with love and affection.

LIST OF FIGURES

Abbreviations

AI	Artificial Intelligence
DCNN	Deep Convolution Neural Network
LLM	Large Language Model
NV	Melanocytic Nevi
MEL	Melanoma
BKL	Benign Keratosis-like Lesion
BCC	Basal Cell Carcinoma
AKIEC	Actinic Keratoses
VASC	Vascular Lesions
DF	Dermatofibroma
PSO	Particle Swarm Optimization
TF-IDF	Term Frequency – Inverse Document Frequency

Abstract

Skin cancer, particularly malignant skin lesions, poses a significant threat, emphasizing the crucial need for early detection and treatment. Using HAM10000 dermatoscopic image dataset to identify and classify the various types of skin lesions. Initial preprocessing involves the application of an adaptive median filter for effective denoising, ensuring the quality of subsequent analysis. Particle Swarm Optimization (PSO) coupled with K-Means for initial segmentation. And the processed images are then subjected to a U-net architecture, a proven encoder-decoder structure widely employed in medical image segmentation. Subsequently, a deep convolutional neural network (DCNN) is utilized to classify distinct classes of skin lesions based on the segmented image. In addition to the prior model, incorporating GPT-2 Language Model, Fine-Tuned on Dermatology Question-Answer Dataset. This model enabled the development of Chatbot capable of extracting symptoms and providing potential causes and treatments. Using extracted User-symptoms, TF-IDF vectorizer and a naive Bayes classifier into the system enhances the prediction of skin lesions. Eventually, integrating both image classification and Chatbot using Django Web Framework. This sophisticated approach of combining advanced image processing techniques with LLM, aims to enhance the accuracy and efficiency of skin lesion diagnosis and prognosis. The accuracy obtained for image classification is 96.75 % for HAM10000 Dataset and for LLM is 47.33%.

KEY WORDS: Skin lesions, UNet, DCNN, PSO, GPT-2, Django Framework

Table of Contents

CHAPTER 1

SUMMARY OF THE BASE PAPER

Title: Skin cancer detection using dual optimization based deep learning network.

Journal name: Elsevier

Publisher: E. Gomathi, M. Jayasheela, M. Thamarai, M. Geetha

Year: 2023

Introduction:

Skin cancer poses a significant health burden globally, the abnormal growth of skin cells which is caused by mutations of DNA and can form tumors at various locations/parts of the body disrupting normal functions of organs and tissues and our day-to-day activities. It varies in terms of size, color and in severe cases it can bleed. It is estimated that one in five people develop skin cancer during their lifetime. According to WHO, males are more affected by this than females. While skin diseases like dermatitis and rashes tend to go away after a period, skin cancer does not respond to traditional practices and treatments. Although many techniques have been available for identifying skin cancer, one such technique is biopsy which is an appropriate method for identifying skin cancer. But it is a tedious and expensive process. This is where the specialist technique called dermascopy comes into play, several efficient solutions have been obtained using optimization techniques. By taking advantage of advanced technologies and methodologies, this paper seeks to simplify the diagnostic process, enabling early identification of skin cancer and ultimately improving patient outcomes.

Novelty:

This paper introduces a new approach to improve accuracy of classifying skin lesions called Dual Optimization Based Deep Learning Network (DODL net). In contrast to the existing methods for the same problem, DODL net uses two optimization algorithms called Bacterial Foraging Optimization (BFO) and Particle Swarm Optimization (PSO) which improves the proposed method to optimal image thresholding that is used to extract features from the segmented image from U-net architecture. Ultimately, Deep CNN classifies different classes of skin lesions based on extracted features.

Architecture and Proposed Method:

This paper introduces a DODL net for identifying and classifying 7 different classes of skin lesions with improved accuracy. This paper discusses several methods before classifying the skin lesion images. Data pre-processing is a crucial step for any classifying models. Data augmentation methods are employed to artificially enhance the size training dataset.

Adaptive Median filter is used to reduce the noise in image which reduces distortions and enhances the features of dermatoscopic images. The noise reduced image is then subjected to

U-net for segmentation, A encoder-decoder structure, it addresses issues with traditional CNN that are utilized for medical image segmentation.

The segmented lesion images are taken as an input to dual optimization algorithm to extract relevant features. The skin images are first transformed from RGB to IHLS color space for accurate mapping of skin color then the image is subjected to BFO and PSO which are used to perform optimal image thresholding. Then the feature extracted image is given as input for DCNN for classifying the skin lesion.

System Architecture Diagram:

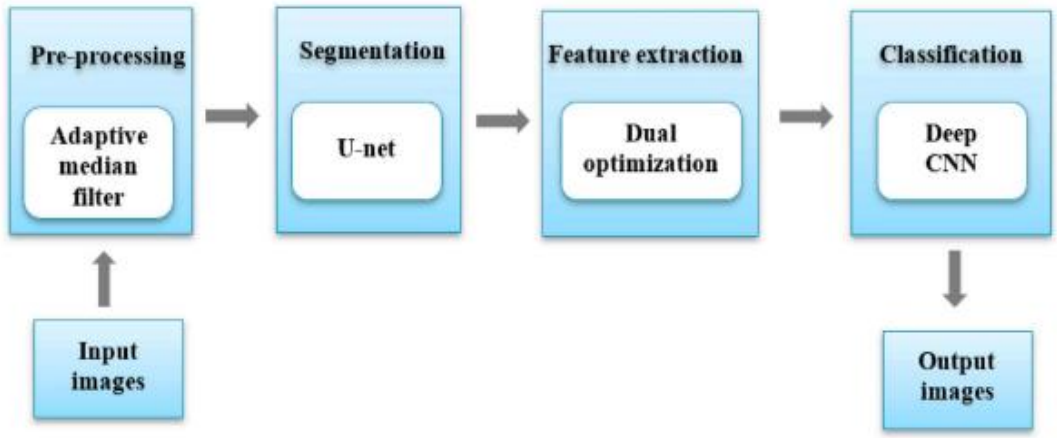

Fig 1.1 Workflow of the proposed method in base paper.

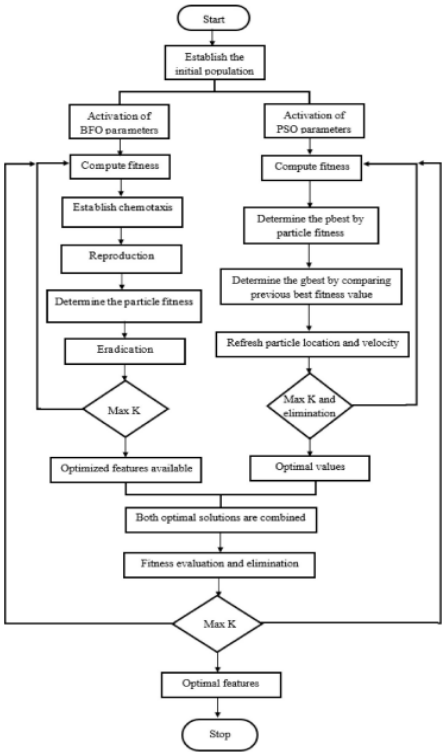

Fig 1.2 Architecture of BFO and PSO used in base paper.

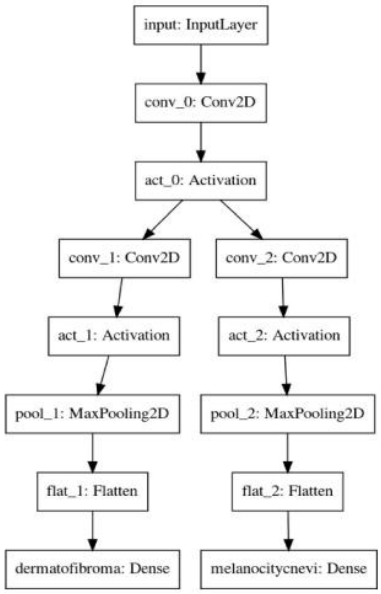

Fig 1.3 Architecture of U-Net used in base paper.

CHAPTER 2

MERITS AND DEMERITS OF THE BASE PAPER

Existing solutions:

- In 2022 Joseph and Olagbars investigated the impact of image pre-processing on the effectiveness of a saliency segmentation approach for skin lesions. The target was achieved by utilizing the CHC Otsu technique, which combines color histogram clustering and Otsu thresholding. The outcomes validate the performance of the CHC-Otsu technique attains the accuracy of 92%
- In 2022 Khalal and Dhannoon developed AlexUnet and AlexUnet+ based on U- shaped structures for skin lesion segmentation. The AlexUnet model is a pre-trained light U-Net based on AlexNet that uses ImageNet as an encoder. In AlexUnet+, another encoder was added, which used VGG11 already trained on ImageNet to encode AlexUnet. The accuracy attained by the AlexUnet+ was 98% and 97% for AlexUnet.
- Is 2022 Garg et al. proposed method for automatically segmenting a skin lesion region. In dermoscopy images, it was employed to identify the locations of skin lesions. With respective accuracy value of 91.9%, these datasets were highly accurate. When compared to previous strategies, the suggested algorithm offers higher accuracy and a lower mistake rate.
- In 2021 Didar et al. had proposed a machine learning method for skin cancer detection was thoroughly studied on a regular basis. To classify images of lesions, it focuses on ANNs, CNNs, KNNs and RBFNs. Additionally, when classifying image data. Due to its stronger connection to computer vision than other neural network architectures, CNN performs better than other varieties of neural networks.
- In 2021 Subramanian et al. developed a Convolution neural network to identify and categorize the type of cancer using clinical image data from the past. Using the CNN model, skin cancer may be detected with an accuracy of 80.45%, a false negatives rate in the prediction of less than 10%. The best way to diagnose skin cancer is the CNN method.

From this, we conclude that the existing methods only achieve good results on classification and segmentation. Thus, accurate detection of different forms of skin cancer remains a challenging task based on its features. All the above existing techniques frequently produce undesirable blur and alter the texture of the lesion. Furthermore, these prior techniques had high computational cost. To overcome these challenges, the novel DODI net is proposed for accurate identification of different types of skin cancer.

Merits:

The merits of this research paper can be summarized as follows:

- **Novel approach:** The proposed DODL net laid down a novel approach for skin lesion detection by integrating deep learning techniques along with optimization algorithms that aims on enhancing the accuracy and efficiency of diagnosis.
- **Segmentation and Feature extraction:** By using U-net for image segmentation and dual optimization algorithms (BFO and PSO) for extracting features, the DODL net can effectively identify and extract relevant features from skin lesion images.
- **High Accuracy:** The DODL net has high accuracy levels, achieving 99.11% accuracy on the MNIST HAM10000 dataset. It outperforms existing methods such as AlexUnet+, CNN, YOLO, and Transfer learning based ResNet by considerable margins.
- The paper outlines potential areas for future research, including exploring evolutionary techniques and faster search spaces for optimization algorithms, as well as incorporating advanced deep learning techniques to enhance the robustness and accuracy of skin lesion detection systems.

Demerits:

The demerits of this research paper can be summarized as follows:

- **Limited Generalization:** While achieving high accuracy on the MNIST HAM10000 dataset, the DODL net may not generalize well with other similar datasets, indicating its limitation to adapt to diverse data distributions.
- **Dependency on Data Quality:** The effectiveness of the DODL net relies on the quality and representation of the training data, focusing on the importance of robust data collection and preprocessing techniques.
- **Complexity and Resource Requirements:** Implementing and training the DODL net may require considerable computational resources and expertise, potentially limiting its accessibility and scalability for widespread use.

CHAPTER 3

INTRODUCTION

The need to identify skin cancers at the earliest is of paramount importance so that treating the patients at the early stages becomes easy and more lives can be saved from these malignancies. Image Classification using DCNN helps to identify the cancer using the dermatoscopic images, additionally using chatbot the details like treatment, causes for that skin lesion can be known to the user.

Proposed Method:

Image Enhancement:

- Adaptive median filter is used for spatial processing to reduce and enhance image quality.

Image Segmentation:

- PSO and K-Means are used to enhance the initial segmentation of images.
- U-Net, an encoder-decoder structure used for accurate and efficient segmentation of medical images.

Image Classification:

- DCNN, which is effective for finding patterns in images to classify objects and classes.
- Especially used for medical image classification.

GPT-2 Fine-Tuning and Chatbot Integration:

- Fine-tuning the transformer model based on QAdataset which contains 1460 questions and answers related to skin diseases.
- To enhance user interaction and provide thoughtful information on skin condition through natural language conversation.

Extraction and Prediction:

- Using NLP, TF-IDF vectorizer, Naive bayes algorithm to extract symptoms and predict disease.
- Provides information about the predicted disease such as symptoms, causes and prevention methods.

CHAPTER 4

EXPLANATION

System Architecture Design:

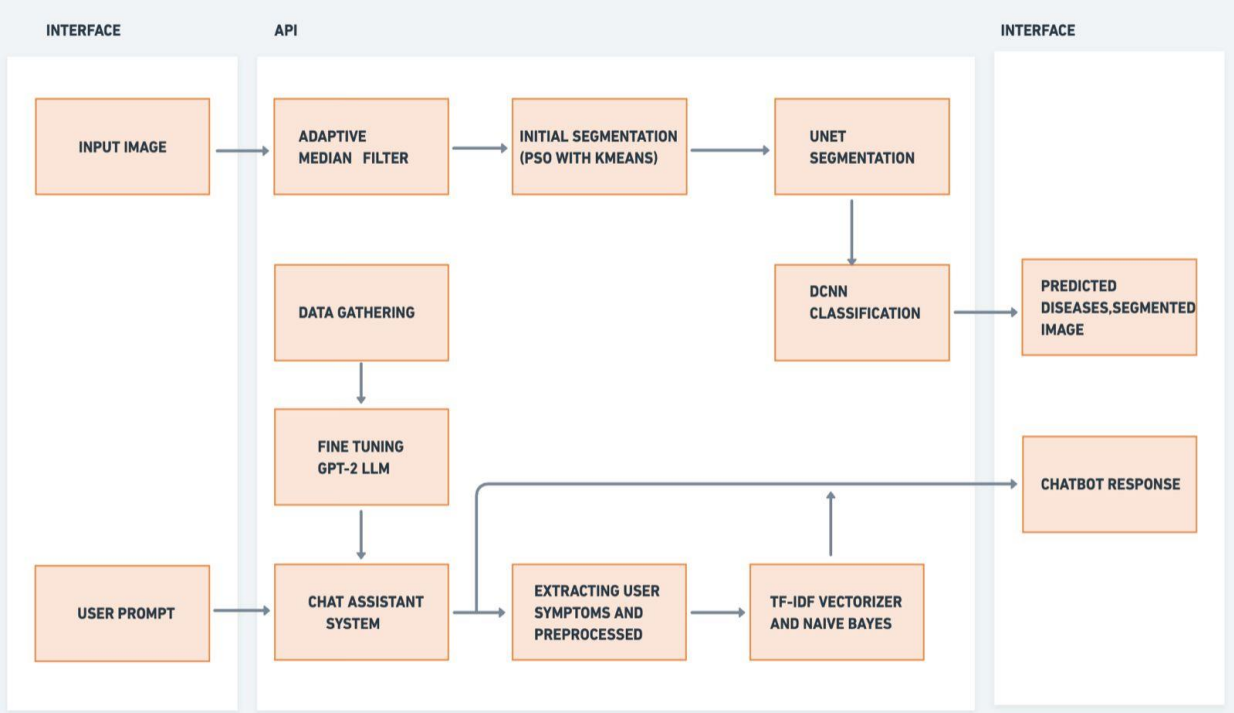

Fig 4.1 Architecture design of proposed method.

Adaptive Median Filter:

An image processing technique which is employed to denoise the image without losing crucial features of the image. It has the capability to dynamically adjust to image characteristics, which makes it well suited for handling diverse noise levels, especially on the dermascopic images.

Initial Segmentation Using PSO and K-Means:

K-Means clustering is the popular unsupervised machine learning algorithm used for clustering datapoints to the nearest centroid based on the mean of the assigned points. On other hand, PSO is a population-based optimization technique inspired by social behavior of bird flocks. The swarm of particles iteratively adjusts its position in search of optimal solution by updating its velocities based on their individual and g-best position.

Thus, swarm of particles iteratively adjusts its position in search of optimal solution by updating its velocities based on their individual and g-best position.

Segmentation Using U-Net:

It's a CNN architecture designed for biomedical image segmentation purposes, well known for its Encoder-decoder structure. The encoder captures varying abstraction of level of

features and the decoder reconstructs the segmented image. Here the initial segmented image is given as input for the U-Net for enhanced image segmentation accuracy .

Classification Using DCNN:

Its utilized for its ability to learn the hierarchical features directly from Images. DCNN has remarkable performance especially on the image classification tasks, including segmented medical image. By training the DCNN on segmented lesion images from U-Net, it can accurately classify the image into the categories in which class label it belongs to.

LLM: Fine-Tuned GPT-2:

A transformer architecture family, known for its remarkable performance in natural language processing tasks. Fine tuning GPT2 involves further training the model on specific datasets to adapt its parameter to works well for a domain or tasks. In this work, we have fined tuned GPT-2 with dermatology QA dataset aims to improve its ability to generate accurate respond to the dermatology related queries, thereby offering a valuable insight to users.

Chatbot Integration:

Build a chatbot system employing the fine-tuned GPT-2 model to comprehend user queries and provide relevant responses. Additionally, the chatbot prompts users with expert-like diagnosis queries, guiding them into structured assessment process, hereby user can engage with the chatbot to describe their symptoms that they have experienced and receive preliminary evaluation.

Symptoms Extraction and Disease Prediction:

NLP technique, TF_IDF vectorizer, which is a numerical representation technique for the information retrieval and classification tasks.

Naive bayes, a probabilistic machine learning algorithm commonly used for classification tasks. In the context of symptoms extraction and disease prediction, TF-IDF vectorizer and naïve bayes algorithm are employed, the TF-IDF vectorizer which converts the user symptoms into numerical vectors, capturing the important words. Then the naïve bayes algorithm then predicts the most likely skin diseases based on the numerical vectors, improving prediction accuracy.

Django Framework:

a high-level python web framework, provides necessary tools, security for building the web application. Integrating both image classification and a chatbot into an interactive experience using the Django framework offers a robust and user-friendly platform for users to engage with the system.

CHAPTER 5

SOURCE CODE

1.UNET MODEL:

```
import warnings

warnings.filterwarnings('ignore')

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import cv2

import os

import glob as gb

from tqdm.auto import tqdm

from sklearn.model_selection import train_test_split

import tensorflow as tf

from tensorflow import keras

from tensorflow.keras.models import Model

from tensorflow.keras import layers

from tensorflow.keras.layers import Conv2D,MaxPooling2D,BatchNormalization , Conv2DTranspose , UpSampling2D

from tensorflow.keras.losses import BinaryCrossentropy

from tensorflow.keras.optimizers import Adam

from tensorflow.keras.metrics import Precision, Recall

IMG_SIZE = 256

BATCH_SIZE = 32

BUFFER_SIZE = 1000

IMG_DIR = '/kaggle/input/ham1000-segmentation-and-classification/images'

MASK_DIR = '/kaggle/input/ham1000-segmentation-and-classification/masks'
```

```
def img_mask_path(img_dir , mask_dir) :
    image_path = sorted(gb.glob(os.path.join(img_dir , '*.jpg')))
    masks_path = sorted(gb.glob(os.path.join(mask_dir , '*.png')))
    image_path = np.array(image_path)
    masks_path = np.array(masks_path)

    return image_path , masks_path
imgs_path , masks_path = img_mask_path(IMG_DIR , MASK_DIR)
x_train , x_test , y_train , y_test = train_test_split(imgs_path , masks_path , train_size = 0.85,)
len(x_train) , len(x_test)
def path_to_img(image_path, mask_path):
    image = tf.io.read_file(image_path)
    image = tf.image.decode_jpeg(image, channels=3)
    image = tf.image.resize(image, (IMG_SIZE, IMG_SIZE))
    image = tf.cast(image, tf.float32) / 255.
    mask = tf.io.read_file(mask_path)
    mask = tf.image.decode_jpeg(mask, channels=1)
    mask = tf.image.resize(mask, (IMG_SIZE, IMG_SIZE))
    mask = tf.cast(mask, tf.float32) / 255.
    return image, mask
AUTOTUNE = tf.data.experimental.AUTOTUNE
train_set = tf.data.Dataset.from_tensor_slices((x_train , y_train))
train_set = train_set.map(path_to_img , num_parallel_calls=AUTOTUNE)
train_set = train_set.shuffle(BUFFER_SIZE).batch(BATCH_SIZE).prefetch(buffer_size = AUTOTUNE)
test_set = tf.data.Dataset.from_tensor_slices((x_test , y_test))
test_set = test_set.map(path_to_img , num_parallel_calls=AUTOTUNE)
test_set = test_set.batch(BATCH_SIZE).prefetch(buffer_size = AUTOTUNE)
print(f"the size of the train dataloader {len(train_set)} batches of {BATCH_SIZE}”)
```

```
print(f"the size of the test dataloader {len(test_set)} batches of {BATCH_SIZE}”)
img_sample , mask_sample = next(iter(train_set))
img_sample.shape , mask_sample.shape
fig , axis = plt.subplots(3 , 2 , figsize = (15,10))
for i in range(3):
    img1 = img_sample[i].numpy()
    axis[i, 0].imshow(img1)
    axis[i, 0].set(title = f"Original Image")
    axis[i, 0].axis('off')
    img2 = mask_sample[i].numpy()
    axis[i, 1].imshow(img2)
    axis[i, 1].set(title = f"Mask Image")
    axis[i, 1].axis('off')
plt.subplots_adjust(wspace=0.0)
def UNET() :
    inputs = keras.Input(shape = (IMG_SIZE , IMG_SIZE , 3))

    x = Conv2D(32, 3, strides=1, padding="same")(inputs)
    x = BatchNormalization()(x)
    x = layers.Activation("relu")(x)

    skip_connections = []

    for filters in [64, 128, 256 , 512]:
        x = Conv2D(filters , 3 , strides = 1 , padding = 'same')(x)
        x = BatchNormalization()(x)
        x = layers.Activation("relu")(x)

        x = Conv2D(filters , 3 , strides = 1 , padding = 'same')(x)
```

```
    x = BatchNormalization()(x)
    x = layers.Activation("relu")(x)
    skip_connections.append(x)
    x = MaxPooling2D(2, strides=2, padding="same")(x)

x = Conv2D(1024 , 3 , strides = 1 , padding = 'same')(x)
x = BatchNormalization()(x)
x = layers.Activation("relu")(x)

x = Conv2D(1024 , 3 , strides = 1 , padding = 'same')(x)
x = BatchNormalization()(x)
x = layers.Activation("relu")(x)

for filters in [512, 256 , 128 , 64] :
    x = Conv2DTranspose(filters , 2 , strides = 2 , padding = "same")(x)
    skip_connection = skip_connections.pop()
    x = layers.add([x , skip_connection])

    x = Conv2D(filters , 3 , strides = 1 , padding = 'same')(x)
    x = BatchNormalization()(x)
    x = layers.Activation("relu")(x)

    x = Conv2D(filters , 3 , strides = 1 , padding = 'same')(x)
    x = BatchNormalization()(x)
    x = layers.Activation("relu")(x)

outputs = Conv2D(1 , 1 , strides = 1 , activation = "sigmoid")(x)
```

```
    model = Model(inputs , outputs)


    return model
model = UNET()
model.summary()
model.compile(Adam(0.0002) , BinaryCrossentropy(), metrics=['accuracy', Precision(name = 'precision'), Recall(name = 'recall')])
with tf.device("/GPU:0") :
    history = model.fit(
        train_set ,
        epochs = 2 ,
        batch_size = BATCH_SIZE ,
        validation_data = test_set ,
        workers = 8 ,
        use_multiprocessing = True
    )
import matplotlib.pyplot as plt


plt.figure(figsize=(10, 10))


# Plot Accuracy
plt.subplot(2, 1, 1)
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('UNET Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.grid(True)
```

```
# Plot Loss
plt.subplot(2, 1, 2)
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('UNET Loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.grid(True)
plt.show()
train_f1_score = (2 * history.history['precision'][-1] * history.history['recall'][-1]) / (history.history['precision'][-1] + history.history['recall'][-1])
test_f1score = (2 * history.history['val_precision'][-1] * history.history['val_recall'][-1]) / (history.history['val_precision'][-1] + history.history['val_recall'][-1])
print(f"The training loss is : {history.history['loss'][-1]:0.2f}\n")
print(f"The training accuracy is : {(history.history['accuracy'][-1]*100):0.2f}%\n")
print(f"The training precision is : {history.history['precision'][-1]:0.2f}\n")
print(f"The training recall is : {history.history['recall'][-1]:0.2f}\n")
print(f"The F1 score of the training set is : {train_f1_score:0.4f}\n”)
print(f"The testing loss is : {history.history['val_loss'][-1]:0.2f}\n")
print(f"The testing accuracy is : {(history.history['val_accuracy'][-1]*100):0.2f}%\n")
print(f"The testing precision is : {history.history['val_precision'][-1]:0.2f}\n")
print(f"The testing recall is : {history.history['val_recall'][-1]:0.2f}\n")
print(f"The F1 score of the testing set is : {test_f1score:0.4f}\n”)
import numpy as np
import matplotlib.pyplot as plt
metrics= ['Loss', 'Accuracy', 'Precision', 'Recall', 'F1 Score']
training_metrics = [history.history['loss'][-1],
                    history.history['accuracy'][-1],
```

```
                history.history['precision'][-1],
                history.history['recall'][-1],
                train_f1_score]
testing_metrics = [history.history['val_loss'][-1],
                history.history['val_accuracy'][-1],
                history.history['val_precision'][-1],
                history.history['val_recall'][-1],
                test_f1score]
plt.figure(figsize=(10, 6))
index = np.arange(len(metrics))
plt.bar(index, training_metrics, 0.35, label='Training')
plt.bar(index + 0.32, testing_metrics,0.35, label='Testing')

plt.xlabel('Metrics')
plt.ylabel('Values')
plt.title('Metrics Comparison')
plt.xticks(index + 0.35 / 2, metrics)
plt.legend()
plt.show()
x_testsample , y_testsample = next(iter(test_set))
y_pred = loaded_model.predict(x_testsample)
def draw(test_images , test_masks , y_preds):

    plt.figure(figsize = (20, 10))
    index = 0
    n = np.random.randint(y_preds.shape[0])
    for i in range(18):
        plt.subplot(3, 6, (i + 1))
        if index == 0:
```

```
            plt.imshow(test_images[n])
            plt.imshow(test_images[n])
            plt.title('Original Image')
            index = 1
            plt.axis('off')
        elif index == 1:
            plt.imshow(test_masks[n])
            plt.imshow(test_masks[n], alpha = 0.2, cmap = 'viridis')
            plt.title('Original Mask')
            index = 2
            plt.axis('off')
        elif index == 2 :
            plt.imshow(test_masks[n])
            plt.imshow(y_preds[n], alpha = 0.2, cmap = 'viridis')
            plt.title('Predict Mask')
            index = 0
            n = np.random.randint(y_preds.shape[0])
            plt.axis('off')



    plt.legend()
draw(x_testsample, y_testsample , y_pred)
model.save("/content/drive/MyDrive/skin_cancer_segmentation.h5”)
import tensorflow as tf
import cv2
import numpy as np
# Load the saved model
loaded_model                                                    =
tf.keras.models.load_model('/content/drive/MyDrive/skin_cancer_segmentation.h5’)
```

```
import matplotlib.pyplot as plt

plt.subplot(1, 2, 1)
plt.imshow(x_testsample[0])
plt.title('Original Image')

# Display the predicted mask
plt.subplot(1, 2, 2)
plt.imshow(y_testsample[0]>0.5, cmap='gray')
plt.title('Predicted Mask')

plt.show()
```

2.ADAPTIVE MEDIAN FILTER :

```
import cv2
import os
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt

def resize_image(image_path, target_size=(256, 256)):
    img = Image.open(image_path)
    img_resized = img.resize(target_size, Image.Resampling.BOX)
    img_array = np.array(img_resized)
    return img_array

def adaptive_median_filter(channel, max_size):
    height, width = channel.shape
    filtered_channel = np.zeros_like(channel)
```

```
    for i in range(height):
        for j in range(width):
            filtered_channel[i, j] = adaptive_median_value(channel, i, j, max_size)

    return filtered_channel

def adaptive_median_value(channel, i, j, max_size):
    height, width = channel.shape
    window_size = 3
    while window_size <= max_size:
        half_size = window_size // 2
        window = channel[max(0, i - half_size):min(height, i + half_size + 1),
                         max(0, j - half_size):min(width, j + half_size + 1)]

        median = np.median(window)

        if window[0, 0] < median < window[-1, -1]:
            return median
        else:
            window_size += 2

    return channel[i, j]
def apply_and_display_adaptive_median_filter(image_path, max_filter_size):
    resized_img = resize_image(image_path, target_size=(256, 256))
    filtered_img = np.zeros_like(resized_img)
    for c in range(3):
        filtered_img[:, :, c] = adaptive_median_filter(resized_img[:, :, c], max_filter_size)
    fig1, axis = plt.subplots(1, 2, figsize=(8, 4))
    axis[0].imshow(resized_img)
```

```
    axis[0].set_title('Resized Original')
    axis[0].axis('off')
    axis[1].imshow(filtered_img)
    axis[1].set_title('Adaptive Median Filter')
    axis[1].axis('off')
    plt.show()
image_path = '/content/ISIC_0024377.jpg'
max_filter_size = 1
apply_and_display_adaptive_median_filter(image_path, max_filter_size)
def apply_and_display_adaptive_median_filter(image_path, max_filter_size):
    resized_img = resize_image(image_path, target_size=(256, 256))
    filtered_img = np.zeros_like(resized_img)
    for c in range(3):
        filtered_img[:, :, c] = adaptive_median_filter(resized_img[:, :, c], max_filter_size)
    # fig, axes = plt.subplots(1, 2, figsize=(8, 4))
    return filtered_img
```

3.PSO WITH KMEANS :

```
import cv2
import numpy as np
from sklearn.cluster import KMeans
from pyswarm import pso
from google.colab.patches import cv2_imshow

image = cv2.imread("/content/ISIC_0024364.jpg")
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
image = cv2.resize(image, (256, 256))
pix = image.reshape(-1, 3)
num_centroids = 5
```

```
num_particles = num_centroids * 3
lower_bound = np.zeros(3 * num_centroids)
upper_bound = np.ones(3 * num_centroids) * 255
def obj_function(pos):
    centroids = pos.reshape(-1, 3)
    labels = KMeans(n_clusters=len(centroids), init=centroids, n_init=1).fit_predict(pix)
    mse = np.mean((pix- centroids[labels]) ** 2)
    return mse
result, _ = pso(objective_function, lower_bound, upper_bound, swarmsize=num_particles)
centroids = result.reshape(-1, 3)
labels = KMeans(n_clusters=len(centroids), init=centroids, n_init=1).fit_predict(pixels)
segmented_image = centroids[labels].reshape(image.shape)
cv2_imshow(image)
cv2_imshow(segmented_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

4.DEEP CNN :

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MultiLabelBinarizer
from sklearn.metrics import classification_report
from tensorflow.keras.preprocessing.image import load_img, img_to_array
from tensorflow.keras import layers, models
from tensorflow.keras.utils import to_categorical
csv_path = '/content/GroundTruth.csv'
df = pd.read_csv(csv_path)
MEL=df['MEL']
NV=df['NV']
```

```
BCC=df['BCC']
AKIEC=df['AKIEC']
BKL=df['BKL']
DF=df['DF']
VASC=df['VASC’]
count_mel=df['MEL'].value_counts()[1]
count_nv=df['NV'].value_counts()[1]
count_bcc=df['BCC'].value_counts()[1]
count_akiec=df['AKIEC'].value_counts()[1]
count_bkl=df['BKL'].value_counts()[1]
count_df=df['DF'].value_counts()[1]
count_vasc=df['VASC'].value_counts()[1]
import matplotlib.pyplot as plt
counts = [count_mel, count_nv, count_bcc, count_akiec, count_bkl, count_df, count_vasc]
classes = ['MEL', 'NV', 'BCC', 'AKIEC', 'BKL', 'DF', 'VASC']
plt.figure(figsize=(10, 6))
plt.bar(classes, counts, color='skyblue')
plt.xlabel('Classes')
plt.ylabel('Counts')
plt.title('Counts of Skin Lesion Classes')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
image_path=df["image"]
labels= df.drop(['image'], axis=1)
ls = []
for (mask_path, image_path1) in zip(masks_path, image_path):
    if image_path1 in mask_path:
      ls.append(image_path1)
```

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(imgs_path , y_one_hot, test_size=0.2,
random_state=42)

print(f"The training set has {len(X_train)} images and {len(y_train)} labels")

print(f"The testing set has {len(X_test)} images and {len(y_test)} labels")

X_train.shape,y_train.shape

def map_fn1(image_path):
  img = tf.io.read_file(image_path)
  filtered_image= apply_and_display_adaptive_median_filter(image_path, 1)
  #img = tf.image.decode_jpeg(filtered_image, channels=3)
  img = tf.image.resize(filtered_image, (IMG_SIZE, IMG_SIZE))
  img = filtered_image / 255.
  mask = loaded_model.predict(np.expand_dims(img, axis=0))
  mask = np.where(mask > 0.5, 1, 0)
  prediction1 = np.squeeze(mask)
  prediction2 = np.expand_dims(prediction1, axis=-1)
  img_array = np.repeat(prediction2, 3, axis=2)
  return img_array
image_height = 256
image_width = 256
num_channels = 3
model = models.Sequential([
    layers.Conv2D(32, (3, 3), activation='relu', input_shape=(256, 256, 3)),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Flatten(),
    layers.Dense(128, activation='relu'),
    layers.Dense(7, activation='sigmoid’)
```

```
])
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy’])
num_classes = 7
epochs = 4
iteration = 0
loss_history = []
accuracy_history = []
for epoch in range(epochs):
   for batch_index, (image_filename, label) in enumerate(zip(X_train, y_train)):
      image = map_fn1(image_filename)
      label = label.reshape((1, num_classes))
      loss, accuracy = model.train_on_batch(np.expand_dims(image, axis=0), label)
      iteration += 1
      if iteration % 100 == 0:
         print(f'Epoch {epoch + 1}/{epochs}, Iteration {iteration}, Loss: {loss}, Accuracy:
{accuracy}')
         loss_history.append(loss)
         accuracy_history.append(accuracy)
    print(f'Epoch {epoch + 1}/{epochs}, Loss: {loss}, Accuracy: {accuracy}’)
model.save("/content/drive/MyDrive/deepcnnT7.h5”)
```

5.FINE TUNING LLM:

```
import pandas as pd
df=pd.read_csv("Downloads/combined_data.csv”)
df.head(5)
df.describe()

import torch
from transformers import GPT2Tokenizer, GPT2LMHeadModel
from torch.utils.data import Dataset, DataLoader

class QADataset(Dataset):
```

```
    def __init__(self, csv_file, tokenizer, max_length):
        self.data = pd.read_csv(csv_file)
        self.tokenizer = tokenizer
        self.max_length = max_length

    def __len__(self):
        return len(self.data)

    def __getitem__(self, idx):
        question = self.data.iloc[idx]['prompt']
        response = self.data.iloc[idx]['response']

        inputs = self.tokenizer.encode_plus(question, response, add_special_tokens=True,
return_tensors='pt', truncation=True)
        input_ids = inputs['input_ids']
        attention_mask = inputs['attention_mask']

        # Dynamic padding
        max_length = max(input_ids.size(1), self.max_length)
        padded_input_ids = torch.zeros((1, max_length), dtype=torch.long)
        padded_attention_mask = torch.zeros((1, max_length), dtype=torch.long)

        padded_input_ids[:, :input_ids.size(1)] = input_ids
        padded_attention_mask[:, :attention_mask.size(1)] = attention_mask

        return {'input_ids': padded_input_ids.squeeze(), 'attention_mask':
padded_attention_mask.squeeze()}
tokenizer = GPT2Tokenizer.from_pretrained('gpt2')
model = GPT2LMHeadModel.from_pretrained('gpt2')
dataset = QADataset('Downloads/combined_data.csv', tokenizer, max_length=350)
dataloader = DataLoader(dataset, batch_size=8, shuffle=True)
num_train_epochs = 3
learning_rate = 1e-5
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
model.to(device)
model.train()
optimizer = torch.optim.AdamW(model.parameters(), lr=learning_rate)

for epoch in range(num_train_epochs):
    total_loss = 0
    for batch_idx, batch in enumerate(dataloader):
        input_ids = batch['input_ids'].to(device)
        attention_mask = batch['attention_mask'].to(device)
```

```
        optimizer.zero_grad()

        outputs = model(input_ids=input_ids, attention_mask=attention_mask, labels=input_ids)
        loss = outputs.loss
        loss.backward()
        optimizer.step()

        total_loss += loss.item()

        # Print progress after each batch
        if batch_idx % 10 == 0:  # Adjust the value to control the frequency of printing
            print(f'Epoch {epoch+1}/{num_train_epochs}, Batch
{batch_idx+1}/{len(dataloader)}, Loss: {loss.item()}')
    print(f'Epoch {epoch+1}/{num_train_epochs}, Total Loss: {total_loss/len(dataloader)}')
model.save_pretrained('fine_tuned_skin_diseases_model')
tokenizer.save_pretrained('fine_tuned_skin_diseases_model')
```

6.DJANGO FRAMEWORK(Views.py):

```
from django.template.response import TemplateResponse

from django.conf import settings

from django.utils.datastructures import MultiValueDictKeyError

from django.core.files.storage import FileSystemStorage

from django.http import HttpResponse

from django.shortcuts import render

from django.http import JsonResponse

from django.views.decorators.csrf import csrf_exempt

from transformers import GPT2LMHeadModel, GPT2Tokenizer

import nltk

import joblib

from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.metrics.pairwise import cosine_similarity

import torch
```

```
import torch
from transformers import GPT2Tokenizer, GPT2LMHeadModel
import random
import nltk
import numpy as np
import numpy as np
import base64
from PIL import Image
import tensorflow as tf
import cv2
import numpy as np
from sklearn.cluster import KMeans
from pyswarm import pso
import json
from django.http import JsonResponse
from tensorflow.keras import backend as K
import matplotlib.pyplot as plt
import cv2
import numpy as np
import spacy
from sklearn.cluster import KMeans
from pyswarm import pso
IMG_SIZE=256
# Adaptive median filter

def resize_image(image_path, target_size=(256, 256)):
    img = tf.io.read_file(image_path)
    img=tf.image.decode_jpeg(img, channels=3)
    img_resized =tf.image.resize(img , (IMG_SIZE , IMG_SIZE))
```

```
    img_array = np.array(img_resized)
    return img_array

def adaptive_median_filter(channel, max_size):
    height, width = channel.shape
    filtered_channel = np.zeros_like(channel)

    for i in range(height):
        for j in range(width):
            filtered_channel[i, j] = adaptive_median_value(channel, i, j, max_size)

    return filtered_channel

def adaptive_median_value(channel, i, j, max_size):
    height, width = channel.shape
    window_size = 3

    while window_size <= max_size:
        half_size = window_size // 2
        window = channel[max(0, i - half_size):min(height, i + half_size + 1),
                         max(0, j - half_size):min(width, j + half_size + 1)]

        median = np.median(window)

        if window[0, 0] < median < window[-1, -1]:
            return median
        else:
            window_size += 2
```

```
    return channel[i, j]


def apply_and_display_adaptive_median_filter(image_path, max_filter_size):
    resized_img = resize_image(image_path, target_size=(256, 256))
    filtered_img = np.zeros_like(resized_img)
    for c in range(3):
        filtered_img[:, :, c] = adaptive_median_filter(resized_img[:, :, c], max_filter_size)
    return filtered_img




# Map function for converting the image path to mask image by applying adaptive median filter
def map_fn1(image_path):
  img = tf.io.read_file(image_path)
  filtered_image= apply_and_display_adaptive_median_filter(image_path, 1)
  #filtered_image=segment_image(filtered_image,num_centroids=5)
  #img = tf.image.decode_jpeg(filtered_image, channels=3)
  #img = tf.image.resize(filtered_image, (IMG_SIZE, IMG_SIZE))
  img = filtered_image / 255.
  mask = model.predict(np.expand_dims(img, axis=0))
  mask = np.where(mask > 0.5, 1, 0)
  prediction1 = np.squeeze(mask)
  prediction2 = np.expand_dims(prediction1, axis=-1)
  img_array = np.repeat(prediction2, 3, axis=2)
  return img_array


# PSO OPTIMIZATION
```

```
def segment_image(img, num_centroids=5):
    # Load the image
    image = img
    image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    image = cv2.resize(image, (256, 256))  # Resize to 256x256

    # Flatten the image to a 2D array of pixels
    pixels = image.reshape(-1, 3)

    # Define the objective function for PSO
    def objective_function(positions):
        # Reshape the positions to cluster centroids
        centroids = positions.reshape(-1, 3)
        # Assign each pixel to the closest centroid
        labels = KMeans(n_clusters=len(centroids), init=centroids, n_init=1).fit_predict(pixels)
        # Calculate the mean squared error between the original pixels and the centroids
        mse = np.mean((pixels - centroids[labels]) ** 2)
        return mse

    # Define the bounds for the PSO optimization
    num_particles = num_centroids * 3  # Number of particles in PSO
    lower_bound = np.zeros(3 * num_centroids)
    upper_bound = np.ones(3 * num_centroids) * 255

    # Perform PSO optimization
    result, _ = pso(objective_function, lower_bound, upper_bound, swarmsize=num_particles)
    # Reshape the optimized result to get the final centroids
```

```
    centroids = result.reshape(-1, 3)
    # Assign each pixel to the closest centroid
    labels = KMeans(n_clusters=len(centroids), init=centroids, n_init=1).fit_predict(pixels)

    # Replace the pixels with the centroid values
    segmented_image = centroids[labels].reshape(image.shape)

    return segmented_image

# Load the model configuration
with open('/Users/akash/skincancer/tensored-django/core12/core12/model_config1.json', 'r') as json_file:
    config = json.load(json_file)

# Convert the loaded JSON string to a dictionary
config_dict = json.loads(config)

# Modify the configuration to remove the unsupported parameter
config_dict.pop('groups', None)  # Remove the 'groups' parameter

# Convert the modified configuration back to JSON format
modified_config = json.dumps(config_dict)

# Load the model with modified configuration and weights
model = tf.keras.models.model_from_json(modified_config)

# Load the model weights
model.load_weights('/Users/akash/Downloads/Skin cancer Segmentation.h5')
deepmodel=tf.keras.models.load_model('/Users/akash/Downloads/deepcnnT7.h5')
```

```
def home(request):
    # image = cv2.imread("/Users/akash/skincancer/tensored-
django/core12/core12/ISIC_0024307.jpg")
    # image = cv2.resize(image, (256, 256))
    # image = image.astype('float32') / 255.0
    # prediction = model.predict(np.expand_dims(image, axis=0))[0]
    # prediction = (prediction > 0.5).astype(np.uint8)
    # img_rgb = np.repeat(prediction, 3, axis=2)
    # img_batch = np.expand_dims(img_rgb, axis=0)
    if request.method == 'POST' and request.FILES.get('imageFile'):
        # Process the uploaded image and make predictions using your model
        image_file = request.FILES['imageFile']
        print(image_file)
        img_array = np.frombuffer(image_file.read(), np.uint8)
        img = cv2.imdecode(img_array, cv2.IMREAD_COLOR)
        predicter = map_fn1("/Users/akash/Downloads/"+str(image_file))
        img_resized = segment_image(img, num_centroids=5)
        img = cv2.resize(img_resized, (256, 256))
        img = img / 255.0  # Normalize pixel values to [0, 1]
        img = np.expand_dims(img, axis=0)
        prediction = model.predict(img)
        prediction = (prediction > 0.5).astype(np.uint8)
        prediction1 = np.squeeze(prediction)
        prediction2 = np.expand_dims(prediction1, axis=-1)
        img_array = np.repeat(prediction2, 3, axis=2)
        cv2.imwrite('maske.png',img_array* 255)
        image_array_rgb = np.repeat(prediction, 3, axis=3)
        prediction2 = deepmodel.predict(np.expand_dims(predicter, axis=0))
        x=np.argmax(prediction2)
```

```
        diseases = ['MELANOMA','MELANOCYTIC NEVI','BASAL CELL
CARCINOMA','ACTINIC KERATOSIS','BENIGN KERATOSIS-LIKE LESIONS
','DERMATOFIBROMA','VASCULAR LESION']
        predicted_disease = diseases[x]
        with open('/Users/akash/skincancer/tensored-django/core12/maske.png', 'rb') as f:
            mask_image_data = base64.b64encode(f.read()).decode('utf-8')
        return JsonResponse({'prediction': predicted_disease, 'mask_image': mask_image_data})
    return render(request,"index.html",{})



def predict(request):
    if request.method == 'POST' and request.FILES.get('image'):
        # Process the uploaded image and make predictions using your model
        image_file = request.FILES['image']
        # Your model prediction code goes here

        # For demonstration purposes, let's assume the prediction is a string
        prediction = "Some prediction result"
        return JsonResponse({'prediction': prediction})
    else:
        return JsonResponse({'error': 'Image file not found or request method not allowed'})



# VIEWS FOR CHATBOT
#loading the TF_IDF Vectorizer and naive bayes model
machine_model= joblib.load('/Users/akash/skincancer/tensored-django/core12/machine.pkl')


symptoms_list = []


# Function to generate response
```

sentences =["Are you experiencing intense itching in specific areas of your body?",

"Have you noticed any red patches or raised bumps on your skin?",

"Do you observe any thickened skin or areas that feel rough to the touch?",

"Have you noticed any raw or swollen areas on your skin?",

"Have you observed any changes in skin color or patches of discoloration?",

"Do you experience dry skin that seems persistent or worsening?",

"Have you noticed any temporary changes in skin color after certain activities or exposures?",

"Has your infant shown any signs of a scalp rash?",

"Have you noticed any oozing or crusting on your skin?",

"Do you feel an increased sensitivity in certain areas of your skin?",

"Have you experienced any skin infections recently?",

"Have you noticed any abnormalities in your hair, such as thinning or changes in texture?",

"Have you observed any abnormalities in your nails, such as discoloration or thickening?",

"Are you experiencing disturbances in sleep that may be related to your skin condition?",

"Do you find yourself experiencing emotional distress due to your skin condition?",

"Have you noticed any changes in your stress levels that coincide with your skin symptoms?",

"Are you experiencing any eye problems that you think may be related to your skin condition?",

"Have you noticed any issues with the skin inside your mouth?",

"Do you observe any swelling in the affected areas?",

"Do you have dry, cracked skin in certain areas?",

"Have you observed any blisters forming on your skin?",

"Do you feel that your skin is becoming thicker or more leathery?",

"Are you experiencing any pain or tenderness in these areas?",

"Have you observed any localized rashes?",

"Do you see any linear patterns in the affected skin?",

"Have you experienced hives or welts?",

"Are you noticing any skin warmth in these areas?",

"Do you observe scaling or flaking of the skin?",

"Have you noticed any changes in skin coloration?",

"Are you experiencing swelling in addition to pain or tenderness?",

"Are you bothered by an itchy rash?",

"Do you feel a burning sensation along with the other symptoms?",

"Are you finding that these areas are sensitive to touch?",

"Have you noticed rough or dry patches of skin?",

"Do you have cracked or peeling skin?",

"Have you observed any signs of inflammation?",

"Are you noticing any skin lesions?",

"Have you observed any thickening of the skin?",

"Are you experiencing any discharge from the blisters?",

"Are you experiencing inflammation inside your ear canal?",

"Have you noticed any scales around your eyebrows?",

"Do you have a rash around your nose?",

"Are you experiencing irritation in skin folds?",

"Do you feel soreness in specific areas of your skin?",

"Have you observed greasy scales on your scalp?",

"Have you noticed any unusual hair loss?",

"Do you have cracked skin in certain areas?",

"Are you sensitive to changes in weather?",

"Do you experience chronic recurrence of symptoms?",

"Is there involvement of your eyelids in the symptoms?",

"Do you have flaky skin on your chest or back?",

"Have you noticed any redness in your skin?",

"Are you experiencing increased sensitivity in certain areas?",

"Have you observed any yellowish patches on your skin?",

"Do you feel a burning sensation on your skin?",

"Have you noticed any thickening of the skin?",

"Are you experiencing pimples or acne-like eruptions?",

"Do you detect a foul odor associated with the symptoms?",

"Are you experiencing any psychological distress due to these symptoms?",

"Have you noticed coin-shaped patches on your skin?",

"Are you experiencing intense itching?",

"Do you observe redness and inflammation in specific areas?",

"Are you dealing with dry, scaly skin?",

"Have you noticed any clear blisters on your skin?",

"Do you feel pain or tenderness in these areas?",

"Have you observed any skin discoloration?",

"Does the rash worsen under certain conditions or triggers?",

"Have you experienced any secondary infections in the affected areas?",

"What factors seem to aggravate or trigger flare-ups of the rash?",

"Have you noticed any crusty patches on your skin?",

"Are you experiencing weeping or oozing from the affected skin?",

"Do you feel that your skin is becoming thicker?",

"Do you have sensitive skin?",

"Are you experiencing a burning sensation in these areas?",

"Do you feel your skin is hot to the touch?",

"Are you sensitive to changes in weather?",

"Do you experience itching induced by sweat?",

"Have you noticed a pattern of recurrence with these symptoms?",

"Do you find that this condition has a chronic nature?",

"Are you experiencing emotional distress related to your skin condition?",

"Are you having difficulties with sleep due to your skin symptoms?",

"Do you feel more irritable than usual?",

"Have you been experiencing increased anxiety?",

"Are you feeling symptoms of depression?",

"Have you withdrawn socially due to your skin condition?",

"Do you feel a loss of confidence because of your skin?",

"Are you avoiding certain activities or situations because of your skin?",

"Have you noticed any difficulty concentrating?",

"Do you feel that your quality of life is impaired due to your skin condition?",

"Have you noticed any peeling or flaking skin?",

"Do you feel that your skin is thickened or calloused?",

"Are you experiencing scaling or fissures?",

"Have you noticed blisters that are weeping or oozing?",

"Do you observe any pustules or vesicles on your skin?",

"Are there erosions or scabbing present?",

"Have you noticed any changes in skin color or texture?",

"Do you observe any changes in your nails, especially if eczema affects fingers or toes?",

"Is there involvement of the palms or soles of your feet?",

"Do you experience itching specifically from the blisters?",

"Do you find the blisters to be painful?",

"Have you experienced recurrence of these symptoms?",

"Are there erosions or scabbing present?",

"Do you experience hypersensitivity in these areas?",

"Have you noticed any changes in sensation?",

"Have you observed any nail changes?",

"Is there involvement of hair follicles?",

"Do the blisters itch?",

"Do the blisters cause pain?",

"Do you find that these symptoms recur?",

"Is this condition chronic in nature?",

"Do you feel a burning sensation on your skin?",

"Do you feel a stinging sensation in certain areas?",

"Do you have red patches on your skin with silvery scales?",

"Have you noticed any abnormalities in your nails?",

"Do you experience nail lifting or separation?",

"Do you notice any yellow or brown discoloration in your nails?",

"Are your nails crumbling?",

"Do you have lesions in skin folds?",

"Do you experience scalp plaques or scales?",

"Are you experiencing scalp itching or burning?",

"Do you have lesions in your ears?",

"Have you noticed any lesions on your eyelids?",

"Are there any lesions present on your lips?",

"Have you noticed any small, red spots on your skin?",

"Do you observe fine scales accompanying these spots?",

"Are you experiencing itching in these areas?",

"Do you have lesions on your trunk?",

"Are there lesions present on your limbs?",

"Do you have lesions on your scalp?",

"Have you noticed lesions on your face?",

"Are there lesions present on your neck?",

"Do you have lesions in your ears?",

"Are there lesions present on your palms?",

"Do you observe lesions on the soles of your feet?",

"Have you noticed any genital lesions?",

"Are there any lesions on your nails?",

"Do you experience lesions in your joints?",

"Are there lesions present on your buttocks?",

"Do you have lesions on your back?",

"Are there any lesions on your thighs?",

"Do you observe lesions around your knees?",

"Are there any lesions on your elbows?",

"Do you have lesions on your wrists?",

"Are there lesions present around your ankles?",

"Do you notice lesions on your fingers?",

"Are there any lesions on your toes?",

"Do you have lesions on your chest?",

"Are there lesions present on your abdomen?",

"Do you find that the symptoms worsen with sweating or heat?",

"Have you noticed any improvement in cooler or drier environments?",

"Are you experiencing emotional distress due to these symptoms?",

"Do you have a rash in your armpits?",

"Do you notice a rash in your groin area?",

"Are you experiencing a rash under your breasts?",

"Do you have a rash on your buttocks?",

"Have you noticed a rash in your genital area?",

"Do you have a rash on your neck?",

"Are you experiencing a rash in skin folds?",

"Do you have a rash in your underarms?",

"Have you noticed a rash on your inner thighs?",

"Do you have a rash in your abdominal folds?",

"Are you experiencing a rash in your intergluteal cleft?",

"Do you have a rash on your underbelly?",

"Have you noticed a rash in your underarm creases?",

"Do you have a rash in your breast creases?",

"Are you experiencing a rash in your buttock creases?",

"Do these patches have a shiny or glossy appearance?",

"Are you experiencing joint pain?",

"Do you notice any joint stiffness?",

"Are your joints swollen?",

"Do you feel inflammation in your joints?",

"Have you noticed a reduction in your range of motion?",

"Do you experience morning stiffness in your joints?",

"Are you feeling fatigued?",

"Do you notice tenderness around your joints?",

"Have you observed swelling in your fingers?",

"Have you observed swelling in your toes?",

"Have you noticed any changes such as pitting, crumbling, or separation in your nails?",

"Are you experiencing lower back pain?",

"Do you have sausage-like swelling of your fingers or toes?",

"Do you experience pain in your heels?",

"Do you have any joint deformities?",

"Do you have fluid-filled blisters?",

"Have any of your blisters ruptured?",

"Are you experiencing the formation of scabs or crusts?",

"Do you feel pain or discomfort during urination?",

"Have you had a fever recently?",

"Are you experiencing headaches?",

"Do you have muscle aches?",

"Have you noticed swollen lymph nodes?",

"Are you feeling fatigued?",

"Have you experienced flu-like symptoms?",

"Do you feel a burning sensation in any area?",

"Are you experiencing a sore throat?",

"Do you have difficulty swallowing?",

"Have you noticed any enlarged lymph nodes?",

"Are you experiencing nausea?",

"Have you vomited recently?",

"Are you having diarrhea?",

"Do you experience genital or anal pain?",

"Are bowel movements painful for you?",

"Are you experiencing abdominal pain?",

"Have you lost your appetite?",

"Do you have sensitivity to light?",

"Do you feel sensitive to touch?",

"Do you feel sensitive to temperature changes?",

"Have you experienced tingling in the affected area?",

"Do you feel malaise?",

"Are you feeling irritable?",

"Are you having diarrhea?",

"Have you vomited recently?",

"Are you experiencing dehydration?",

"Do you have difficulty breathing?",

"Do you have an earache?",

"Are you sensitive to touch?",

"Have you experienced any complications?",

"Have you experienced a high fever recently?",

"Have you noticed any new moles or growths on your skin?",

"Have you observed any changes in the size, shape, or color of existing moles?",

"Do you notice irregular shapes in your moles?",

"Have you observed blurred borders around any moles?",

"Do you see any variation in color within your moles?",

"Are any of your moles itching?",

"Do you experience tenderness or pain in any mole?",

"Have you noticed any bleeding or oozing from your moles?",

"Do you feel any scaly or crusty texture on your moles?",

"Are any portions of your moles elevated or raised?",

"Have you observed darkening of any mole?",

"Do you notice any redness or inflammation around your moles?",

"Are your moles sensitive to touch?",

"Have you noticed rapid enlargement of any mole?",

"Do you observe any changes in texture of your moles?",

"Have you observed development of nodules within any mole?",

"Do you notice ulceration or erosion on any mole?",

"Do you experience persistent itching in any mole?",

"Do you feel persistent pain in any mole?",

"Have you experienced persistent bleeding or oozing from any mole?",

"Do any of your moles exceed 6mm in size?",

"Have you observed irregular surfaces on any mole?",

"Do you notice irregular pigmentation within any mole?",

"Have you noticed evolution or changes in any moles over time?",

"Do any moles stand out from others on your skin?",

"Have you noticed any moles growing rapidly?",

"Do you perceive any changes in sensation within any mole?",

"Do you observe any changes in symmetry of any mole?",

"Have you noticed changes in color distribution within any mole?",

"Do you observe changes in elevation of any mole?",

"Do you have difficulty sleeping?",

"Are you sensitive to light?",

"Have you experienced nausea?",

"Have you vomited recently?",

"Are you having diarrhea?",

"Do you have an earache?",

"Are you feeling dizzy?",

"Are you experiencing dehydration?",

"Do you have difficulty breathing?",

"Have you noticed a rapid heartbeat?",

"Are you feeling confused?",

"Have you experienced convulsions or seizures?",

"Are you concerned about secondary bacterial infections?",

```
"Have you had symptoms of encephalitis?",
]
random.shuffle(sentences)
# Initialize a set to keep track of asked sentences
asked_sentences = set()
# Load the fine-tuned model and tokenizer
model_path = "/Users/akash/skincancer/tensored-
django/core12/core12/skin_diseases_expert_chatbot"
tokenizer = GPT2Tokenizer.from_pretrained(model_path)
model1 = GPT2LMHeadModel.from_pretrained(model_path)
# Set the model to evaluation mode

model1.eval()
def generate_response(prompt, model, tokenizer):
    # Encode the prompt
    input_ids = tokenizer.encode(prompt, return_tensors='pt')
    # Generate output with attention mask
    attention_mask = torch.ones_like(input_ids)
    output = model1.generate(input_ids, max_length=200, num_return_sequences=1,
temperature=1, pad_token_id=tokenizer.eos_token_id, attention_mask=attention_mask)
    # Decode the generated output
    generated_text = tokenizer.decode(output[0], skip_special_tokens=True)
    return generated_text

def tokenize_and_filter_nouns(user_input):
    tokens = nltk.word_tokenize(user_input)
    nouns = [word for word, pos in nltk.pos_tag(tokens) if pos.startswith('NN')]
    return nouns
count1=0
diagnose=False
```

```
sentences1=""
@csrf_exempt
def chatbot(request):
    global count1
    global diagnose
    global sentences1
    global symptoms_list
    if request.method == 'POST':
        user_input = request.POST.get('user_input')
        if user_input=="diagnose":
            count1=0
            diagnose=True
        response_messages = []
        if count1<3 and diagnose:
            count1=count1+1
            # Ask the selected sentence

            # Add the asked sentence to the set
            if "yes" in user_input:
                noun_list = tokenize_and_filter_nouns(sentences1)
                print(noun_list)
                extracted_symptoms=noun_list
                symptoms_list.extend(extracted_symptoms)

            if "no" in user_input.lower():
                noun_list=[]
            else:
                noun_list = tokenize_and_filter_nouns(user_input)
                extracted_symptoms=noun_list
```

```
            symptoms_list.extend(extracted_symptoms)
        for sentence in sentences:
            if sentence not in asked_sentences:
                break
        sentences1=sentence
        print("Chatbot: " + sentences1)
        response_messages= sentence
            # Prepare the response for this iteration
        response_data = {
                "messages": response_messages,
                "symptoms_list": symptoms_list
            }
        asked_sentences.add(sentence)
           # Return the response
        print(symptoms_list)
        return JsonResponse(response_data)
    if count1==3 and diagnose:
        count1=count1+1
        sentence="Other than 'Do you have any other symptoms?' please mention it."

        # Ask the selected sentence
        print("Chatbot: " + sentence)

        # Add the asked sentence to the set
        noun_list = tokenize_and_filter_nouns(user_input)
        extracted_symptoms=noun_list
        symptoms_list.extend(extracted_symptoms)
        response_messages= sentence
            # Prepare the response for this iteration
```

```
        response_data = {
            "messages": response_messages,
            "symptoms_list": symptoms_list
          }

          # Return the response
        print(symptoms_list)
        return JsonResponse(response_data)
    if count1==4 and diagnose:
        noun_list = tokenize_and_filter_nouns(user_input)
        extracted_symptoms=noun_list
        symptoms_list.extend(extracted_symptoms)
        count1=count1+1
        symptoms_list=filter_words(symptoms_list)
        filtered_words = []
        for word in symptoms_list:
            if word.lower() != "yes":
                filtered_words.append(word)
        print(filtered_words)

        symptoms_ls=[i.lower() for i in filtered_words]
        # Extracting nouns and adjectives from user input for prediction
        user_symptoms = " ".join(symptoms_ls)

        # Make a prediction
        predicted_disease = machine_model.predict([user_symptoms])
        sentence="from your response the predicted disease may be
"+str(predicted_disease)+" ,it's advisable to consult your doctor for further confirmation as
diagnostic predictions can sometimes be inaccurate."
        response_messages= sentence
```

```
            response_data = {
                "messages": response_messages,
                "symptoms_list": symptoms_list
            }

            # Return the response
            return JsonResponse(response_data)
        else:
            diagnose=False
            user_input = request.POST.get("user_input")
            # Generate response based on user input
            response = generate_response(user_input, model1, tokenizer)
            # Print the response
            response_messages=response
            # Prepare the response for this iteration
            response_data = {
                "messages": response_messages,
                "symptoms_list": symptoms_list
            }

            # Return the response
            return JsonResponse(response_data)
    else:
        return render(request, 'chatbot.html')

def filter_words(words):
    # Load the English language model
    nlp = spacy.load("en_core_web_sm")
```

```
    # Initialize an empty list to store filtered words
    filtered_words = []

    # Iterate over each word in the input list
    for word in words:
        # Perform POS tagging
        doc = nlp(word)
        # Check if the word is not a pronoun (PRON) or determiner (DET)
        if not any(token.pos_ in ['PRON', 'DET'] for token in doc):
            filtered_words.append(word)
            return filtered_words
```

7.IMAGE_CLASSIFICATION(Template):

```
{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Image Classification</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f0f0f0;
            margin: 0;
            padding: 0;
        }

        .container {
            max-width: 600px;
```

```
    margin: 20px auto;
    padding: 20px;
    background-color: #fff;
    border-radius: 8px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    position: relative;
}

h2 {
    margin-top: 0;
    color: #333;
}

input[type="file"] {
    display: none;
}

.btn {
    padding: 10px 20px;
    margin: 0 10px;
    background-color: #007bff;
    color: #fff;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    transition: background-color 0.3s ease;
}

.btn:hover {
```

```
    background-color: #0056b3;
}

#imageDisplay {
    margin-top: 20px;
    text-align: center;
}

#imageDisplay img {
    max-width: 100%;
    border-radius: 8px;
    box-shadow: 0 0 5px rgba(0, 0, 0, 0.1);
}

#predictionResult {
    margin-top: 20px;
    text-align: center;
    font-size: 18px;
    color: #333;
}

#predictedImage {
    margin-top: 20px;
    max-width: 100%;
    display: block;
    border-radius: 8px;
    box-shadow: 0 0 5px rgba(0, 0, 0, 0.1);
}
```

```
        #chatbotBtn {
            position: absolute;
            top: 10px;
            right: 10px;
        }
    </style>
</head>
<body>
    <div class="container">
        <h2>Image Classification</h2>
        <button id="chatbotBtn" class="btn" onclick="goToChatbot()">Chatbot</button>
        <form id="imageForm" enctype="multipart/form-data" method="post" action="">
            {% csrf_token %}
            <label for="imageFile" class="btn">Upload Image</label>
            <input type="file" name="imageFile" id="imageFile" accept="image/*" onchange="uploadImage()">
            <button type="button" class="btn" onclick="predictImage()">Predict</button>
        </form>
        <div id="imageDisplay"></div>
        <div id="predictionResult"></div>
        <img id="predictedImage" src="{% static 'images/default.png' %}" alt="Predicted Image"> <!-- Display predicted image here -->
    </div>

    <script>
        function uploadImage() {
            var formData = new FormData();
            var fileInput = document.getElementById('imageFile');
            var imageDisplay = document.getElementById('imageDisplay');
```

```
    formData.append('imageFile', fileInput.files[0]);

    // Display the selected image
    var reader = new FileReader();
    reader.onload = function(e) {
        var img = document.createElement('img');
        img.src = e.target.result;
        imageDisplay.innerHTML = '';
        imageDisplay.appendChild(img);
    }
    reader.readAsDataURL(fileInput.files[0]);
}

function predictImage() {
    var formData = new FormData();
    var fileInput = document.getElementById('imageFile');
    var predictionResult = document.getElementById('predictionResult');
    var predictedImage = document.getElementById('predictedImage');

    formData.append('imageFile', fileInput.files[0]);

    // Submit the form with image data to the server for prediction
    var xhr = new XMLHttpRequest();
    xhr.open('POST', '', true);
    xhr.setRequestHeader("X-CSRFToken", "{{ csrf_token }}");
    xhr.onload = function () {
        if (xhr.status == 200) {
            // Display the prediction result
            var response = JSON.parse(xhr.responseText);
```

```
                predictionResult.textContent = 'Prediction: ' + response.prediction;
                // Display the predicted image
                predictedImage.src = '' +
JSON.parse(xhr.responseText).mask_image;
            } else {
                console.error('Error:', xhr.statusText);
                predictionResult.textContent = 'Error occurred during prediction.';
            }
        };
        xhr.send(formData);
    }

    function goToChatbot() {
        window.location.href = "{% url 'chatbot' %}";
    }
  </script>
</body>
</html>
```

8.CHATBOT(Template):

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Interactive Chatbot</title>
  <style>
    /* Chat container */
    .chat-container {
      border: 1px solid #ccc;
```

```
    border-radius: 5px;
    overflow: hidden;
    height: 400px;
    margin-bottom: 20px;
    font-family: Arial, sans-serif;
}

/* Chat messages */
.chat-messages {
    padding: 10px;
    overflow-y: scroll;
    height: 100%;
}

/* User message */
.user-message {
    background-color: #f0f0f0;
    color: #333;
    border-radius: 10px;
    padding: 10px;
    margin-bottom: 10px;
    max-width: 70%;
    word-wrap: break-word;
    align-self: flex-end; /* Align user messages to the right */
}

/* Chatbot message */
.chatbot-message {
    background-color: #e5f5fc;
```

```
    color: #31708f;
    border-radius: 10px;
    padding: 10px;
    margin-bottom: 10px;
    max-width: 70%;
    word-wrap: break-word;
    align-self: flex-start; /* Align chatbot messages to the left */
}

/* Input container */
.input-container {
    display: flex;
    justify-content: space-between;
    align-items: center;
}

/* Input field */
.input-field {
    flex: 1; /* Take remaining space */
    padding: 10px;
    border-radius: 5px;
    border: 1px solid #ccc;
    margin-right: 10px;
    font-size: 16px;
    outline: none; /* Remove default outline */
}

/* Send button */
.send-btn {
```

```
            padding: 10px 20px;
            border-radius: 5px;
            background-color: #4CAF50;
            color: white;
            border: none;
            cursor: pointer;
            font-size: 16px;
            outline: none; /* Remove default outline */
        }

        /* Send button on hover */
        .send-btn:hover {
            background-color: #45a049;
        }
    </style>
</head>
<body>
    <h1 style="text-align: center;">Interactive Chatbot</h1>

    <div class="chat-container" id="chat-container">
        <div class="chat-messages" id="chat-messages">
            <div class="chatbot-message">Welcome to the Interactive Chatbot. Do you want to diagnose a skin condition or talk to the chatbot?</div>
        </div>
    </div>

    <div class="input-container">
        <input type="text" id="user-input" class="input-field" placeholder="Type 'Diagnose' to know more about your symptoms">
        <button onclick="sendMessage()" class="send-btn">Send</button>
```

```
</div>

<script>
    function sendMessage() {
        var userInput = document.getElementById('user-input').value;
        addUserMessage(userInput);
        fetch('/chatbot/', {
            method: 'POST',
            headers: {
                'Content-Type': 'application/x-www-form-urlencoded',
                'X-CSRFToken': getCookie('csrftoken')
            },
            body: 'user_input=' + encodeURIComponent(userInput)
        })
        .then(response => response.json())
        .then(data => {
            addChatbotMessage(data.messages);
        });
        document.getElementById('user-input').value = ''; // Clear input field
    }

    function addUserMessage(message) {
        var chatMessages = document.getElementById('chat-messages');
        var userMessageElement = document.createElement('div');
        userMessageElement.className = 'user-message';
        userMessageElement.innerHTML = message;
        chatMessages.appendChild(userMessageElement);
        chatMessages.scrollTop = chatMessages.scrollHeight; // Scroll to bottom
    }
```

```
        function addChatbotMessage(message) {
            var chatMessages = document.getElementById('chat-messages');
            var chatbotMessageElement = document.createElement('div');
            chatbotMessageElement.className = 'chatbot-message';
            chatbotMessageElement.innerHTML = message;
            chatMessages.appendChild(chatbotMessageElement);
            chatMessages.scrollTop = chatMessages.scrollHeight; // Scroll to bottom
        }

        function getCookie(name) {
            var cookieValue = null;
            if (document.cookie && document.cookie !== '') {
                var cookies = document.cookie.split(';');
                for (var i = 0; i < cookies.length; i++) {
                    var cookie = cookies[i].trim();
                    if (cookie.substring(0, name.length + 1) === (name + '=')) {
                        cookieValue = decodeURIComponent(cookie.substring(name.length + 1));
                        break;
                    }
                }
            }
            return cookieValue;
        }
    </script>
</body>
</html>.
```

CHAPTER 6

OUTPUT SNAPSHOT

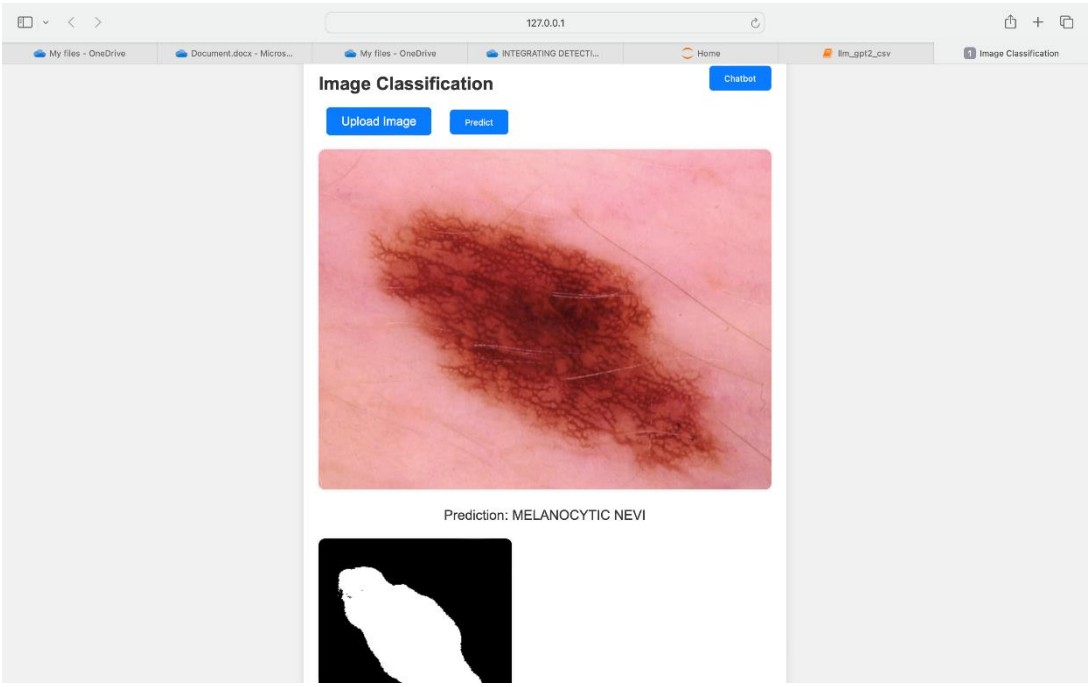

Fig 6.1 Skin Cancer classfication for uploaded image

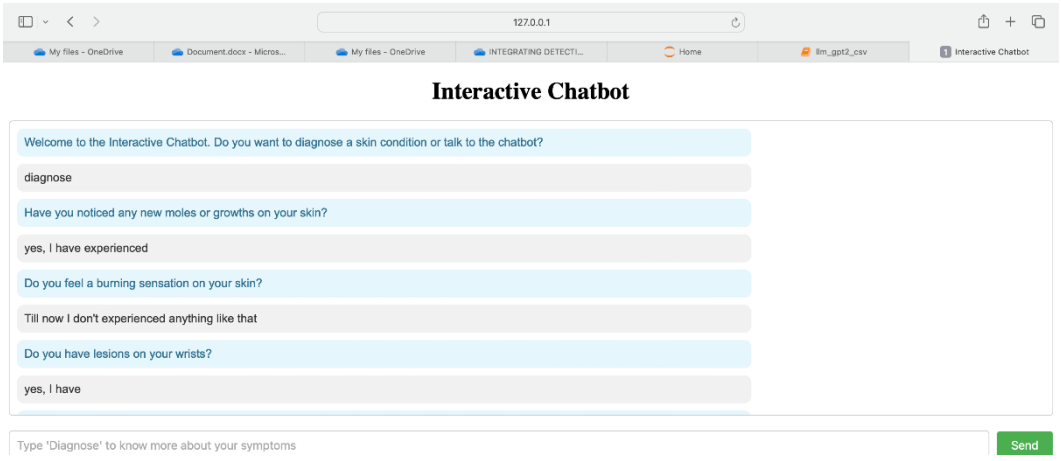

Fig 6.2 Interface for interactive chatbot

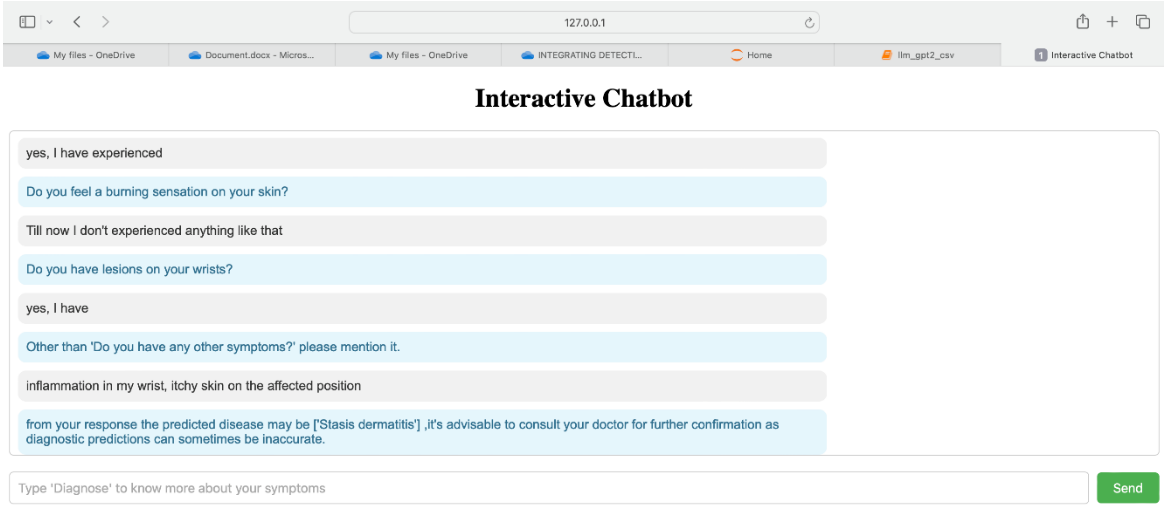

Fig 6.3 Chatbot predicted diseases for user response

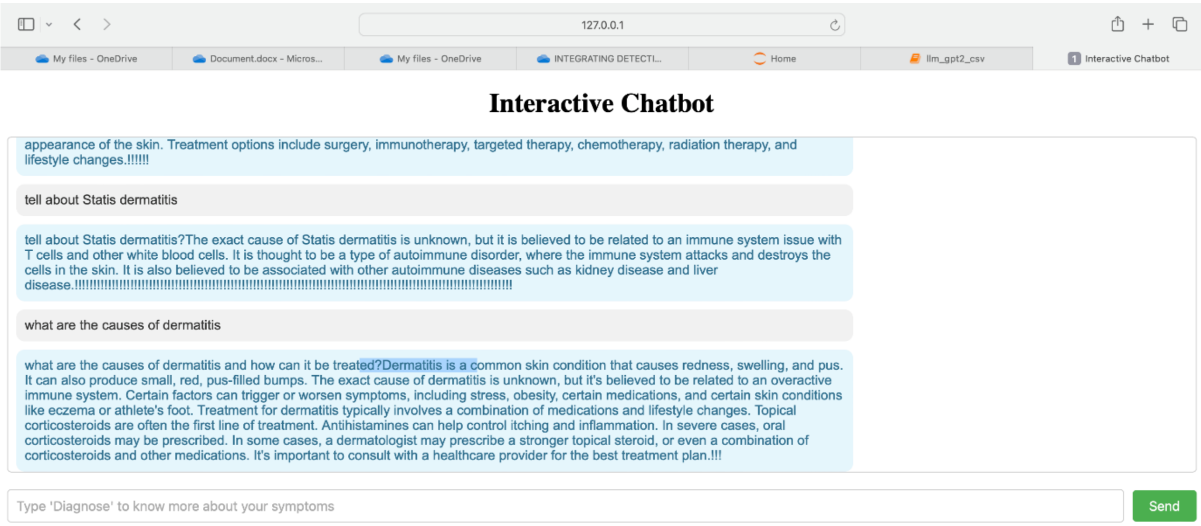

Fig 6.4 Chatbot Response for user skin queries

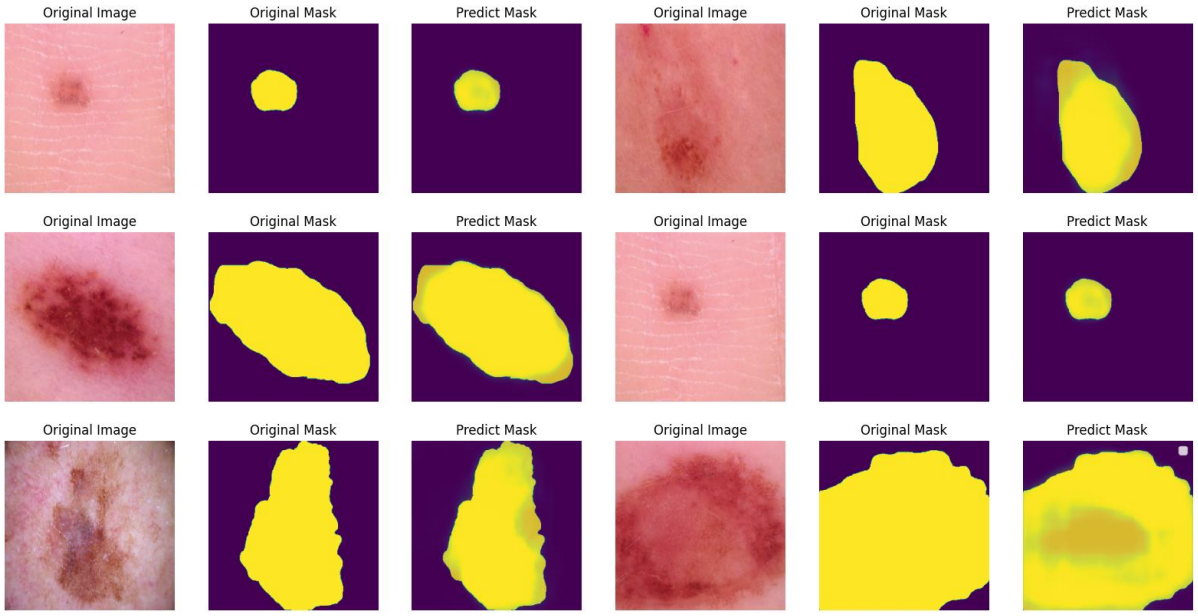

Fig 6.5 Unet output for testing data

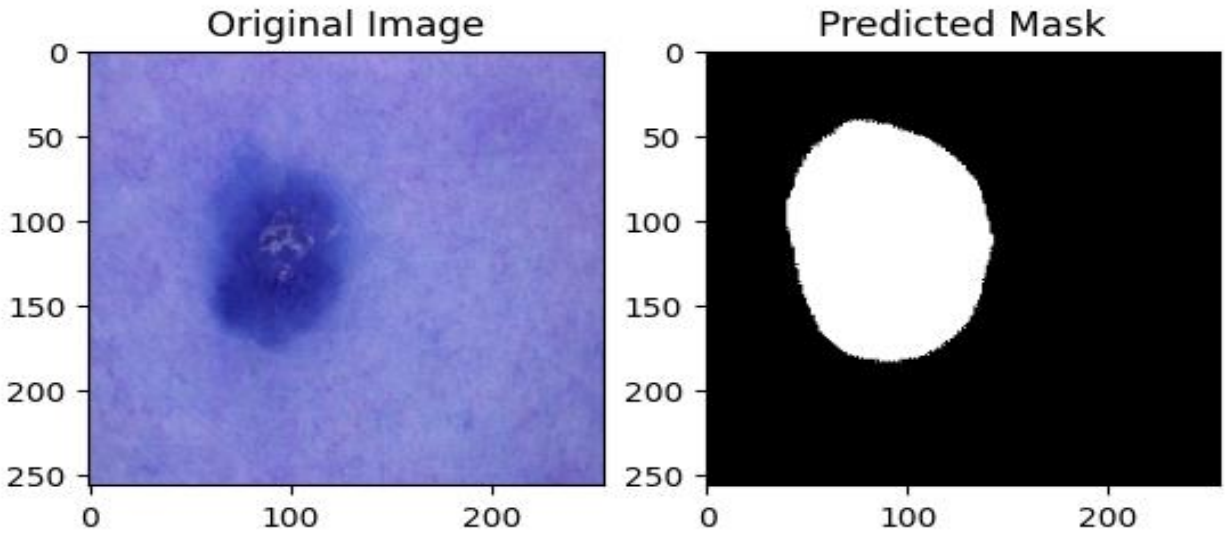

Fig 6.6 Prediction output of Unet without intial segmentation

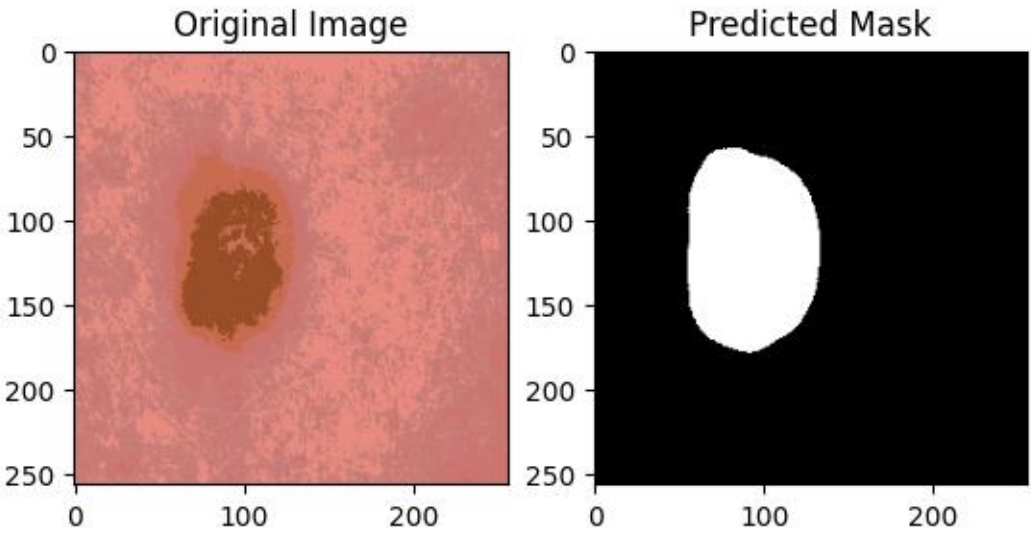

Fig 6.7 Prediction output of Unet with intial segmentation

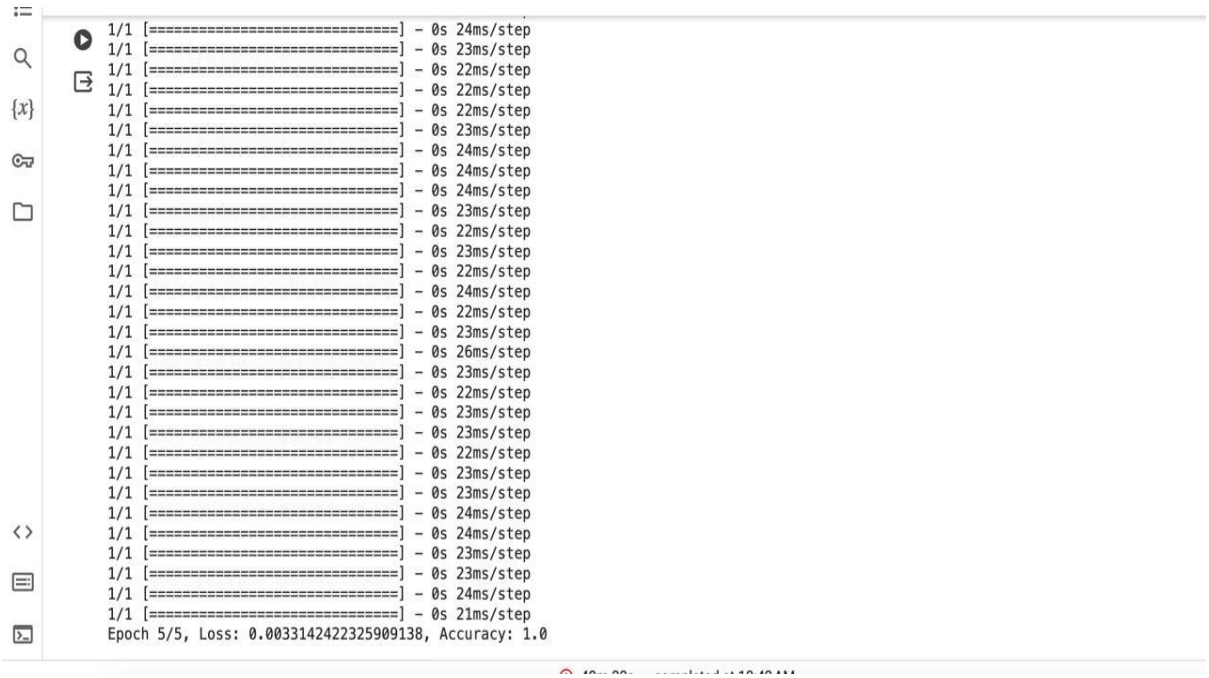

Fig 6.8 Accuracy and Loss

CHAPTER 7

CONCLUSION AND FUTURE PLANS

1. Advanced Image processing techniques combined with deep learning architecture achieved a high accuracy rate of 99.6 for skin cancer image classification on ham10000 dataset.
2. The initial segmentation with PSO and K-means provides a rough segmentation based on color similarity followed by U-Net segmentation provides a better segmentation result even for unknown dataset.
3. Integration of GPT-2 language model fined tuned on dermatology dataset enabled with chatbot interface symptoms extraction and responding to medical related queries.
4. TF-IDF vectorizer and naïve bayes classifier enhanced prediction accuracy based on user provided symptoms.
5. Since the image classification for real-time images is not that accurate. In the future we will prioritize improving our model for real-time images by training it with diverse datasets. Expanding the LLM model for increased dermatological dataset diversity and size.

CHAPTER 8

REFERENCES

1. Akash Kumar, V., Mishra, V. and Arora, M. (2021) *'Convolutional neural networks for malignant and benign cell classification using dermatoscopic images', 2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)* [Preprint]. doi:10.1109/icicv50876.2021.9388605.
2. Daghrir, J. et al. (2020) *'Melanoma skin cancer detection using deep learning and classical machine learning techniques: A hybrid approach', 2020 5th International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)* [Preprint]. doi:10.1109/atsip49331.2020.9231544.
3. Vidya, M. and Karki, M.V. (2020) *'Skin cancer detection using machine learning techniques', 2020 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)* doi:10.1109/conecct50063.2020.9198489.
4. Gong, X. and Xiao, Y. (2021) *'A skin cancer detection interactive application based on CNN and NLP', Journal of Physics: Conference Series,* 2078(1), p. 012036. doi:10.1088/1742- 6596/2078/1/012036.
5. Islam, Md.K. et al. (2021) *'Melanoma skin lesions classification using deep convolutional neural network with transfer learning', 2021 1st International Conference on Artificial Intelligence and Data Analytics (CAIDA)* [Preprint]. doi:10.1109/caida51941.2021.9425117.
6. Ahammed, M., Mamun, Md.A. and Uddin, M.S. (2022) *'A machine learning approach for skin disease detection and classification using image segmentation', Healthcare Analytics*, 2, p. 100122. doi:10.1016/j.health.2022.100122.
7. Jasna, Aiswarya, Krishnapriya, Sisira, & Vettath, J. N. (2021). *D-PREDICT: DISEASE PREDICTION USING GAUSSIAN NAIVE BAYES ALGORITHM.*

CHAPTER 9

APPENDIX-BASE PAPER

Biomedical Signal Processing and Control 84 (2023) 104968

Contents lists available at ScienceDirect

Biomedical Signal Processing and Control

journal homepage: www.elsevier.com/locate/bspc

Skin cancer detection using dual optimization based deep learning network

E. Gomathi [a,*], M. Jayasheela [b], M. Thamarai [c], M. Geetha [d]

[a] Associate Professor, Department of Electronics and Communication Engineering, KIT-Kalaignarkarunanidhi Institute of Technology, Coimbatore, India
[b] Professor, Department of Electronics and Communication Engineering, KIT-Kalaignarkarunanidhi Institute of Technology, Coimbatore
[c] Professor, Department of Electronics and Communication Engineering, Sri Vasavi Engineering College, Andhra Pradesh
[d] Assistant Professor, Electronics and Communication Engineering, KIT-Kalaignar Karunanidhi Institute of Technology, Coimbatore

ARTICLE INFO

Keywords:
Skin cancer
Deep learning
Bacterial foraging optimization
Particle swarm optimization
Dermoscopic images

ABSTRACT

Skin cancer is the most perilous kind of cancer, which is a most important public health problem. Skin cancer can be prevented and treated more effectively if malignant lesions are identified in its early stages. In artificial intelligence, machine learning and deep learning algorithms are used to classify data with high accuracy based on features in the input images. In this work, a novel dual optimization based deep learning network (DODL net) has been proposed for detecting the skin cancer. Initially, the dermoscopic images are gathered from MNIST HAM10000 dataset. The collected images are pre-processed using adaptive median filter and these noise-free images are processed in U-net for segmenting the particular region of the skin lesion. Further, the dual optimization algorithms which is hybridization of Bacterial Foraging Optimization (BFO) and Particle Swarm Optimization (PSO) are utilized for extracting the features from the segmented images. Finally, the Deep Convolutional neural network (CNN) classifies the seven different classes of skin cancer based on the extracted features. The performance of the DODL net has been evaluated using specific parameters such as precision, recall, F1 score and accuracy. The accuracy attained by the proposed DODL net is 98.76% for MNIST HAM10000 dataset.

1. Introduction

Skin cancer is an abnormal growth of skin cells. Mutations in the DNA of skin cells proliferate uncontrollably and aggregate into a mass of cancer cells. A huge brown area with darker flecks is one of the symbols of skin cancer [1]. It can also change color, size or feel and it can bleed. In 2020, there will be more than 1.5 million new cases of skin cancer worldwide. It is estimated that one in five people will develop skin cancer during their lifetimes [2]. There are approximately 9,500 skin cancer diagnoses every day. According to WHO, males made up 7650 more new cases than females (5320), with 54% and 46% of all cases being males, respectively [3]. While rashes, on the other hand tend to go away over time, skin cancer cells do not respond to conventional treatments and spread to other areas of the skin quickly [4]. Skin cancer diagnosis relies heavily on Deep learning and Convolutional neural network (CNN) technology. Some of the systems base their diagnosis of melanoma on dermoscopic images, while others employ digital images [5]. The diagnosis of melanoma included a variety of features and machine learning techniques, although the majority of the systems used the ABCD (Asymmetry, Border, Colour, and Diameter) criterion for feature extraction and detection [6]. Based on the calculated features, the neural network and ABCD rules are used to classify the framed skin lesion.

The classification of skin lesions images with a high classification rate, CNN is employed in conjunction with training certain existing models to compare into respective cancer [7]. Skin lesions are defined as areas of the skin that are aberrant in relation to other areas of the skin. It can be classified into two categories: Main and Secondary, which might result in skin cancer [8]. The biopsy technique is an appropriate method for identifying skin cancer [9,10]. The test is sent to other research facilities for testing in this biopsy procedure, which is a time-consuming and expensive process [11]. A specialist technique called dermoscopy can be used to take high- resolution, enlarged pictures of skin. Surface skin reflectance is eliminated as well as light control assistance [12]. Several efficient solutions have been obtained with the help of optimization techniques, mostly due to recent advances in evolutionary techniques. Optimization methods are closely adapted to image thresholding problems based on their various target challenges [13]. The aim of optimization is to find the best possible solution from the available search space, and the algorithms are in charge of using the objective function to evaluate the best solution by assessing the entire search

* Corresponding author.
E-mail address: gomathi.ece.00@gmail.com (E. Gomathi).

https://doi.org/10.1016/j.bspc.2023.104968
Received 8 February 2023; Received in revised form 1 April 2023; Accepted 15 April 2023
Available online 28 April 2023
1746-8094/

space. The contributions of this paper are presented:

- The main purpose of this work is to present a novel dual optimization based deep learning network (DODL net) for detecting the types of skin cancer.
- Initially, the images are collected and processed in U-net for segmenting the particular region of the skin lesion.
- Further, the dual optimization algorithms (BFO and PSO) are used to extract the features from the segmented images.
- The use of dual optimization algorithms aggrandized the proposed method to optimal image thresholding.
- Finally, the Deep CNN classifies the seven different classes of skin cancer based on the extracted features.

The rest of this work has been prearranged as follows. Section-2 summaries with the literature works, Section-3 comprises the detailed explanation of the proposed methodology for shine cancer detection, Section-4 includes with results and discussion and finally Section-5 enfolds with conclusion and future work.

2. Related works

In recent years, many studies for detecting skin cancer in its early stages was proposed by researchers. Somme of the most recent studies is presented in this section.

In 2022 Joseph and Olugbara [14] had investigated the impact of image pre-processing on the effectiveness of a saliency segmentation approach for skin lesions. The target was achieved by utilizing the CHC-Otsu technique, which combines colour histogram clustering and Otsu thresholding. The outcomes validate the performance of the CHC-Otsu technique attains the accuracy of 92%.

In 2022 Khalaf, and Dhannoon [15] had developed AlexUnet and AlexUnet+ based on U-shaped structures for skin lesion segmentation. The AlxUnet model is a pre-trained light U-Net based on AlexNet that uses ImageNet as an encoder. In Alexunet+, another encoder was added, which used VGG11 already trained on ImageNet to encode AlxUnet. The accuracy attained by the AlexUnet+ was 98% and 97% for AlexUnet.

In 2022 Garg et al. [16] proposed method for automatically segmenting a skin lesion region. In dermoscopy images, it was employed to identify the locations of skin lesions. With respective accuracy value of 91.9%, these datasets were highly accurate. When compared to previous strategies, the suggested algorithm offers higher accuracy and a lower mistake rate.

In 2021 Didar et al. [17] had proposed a machine learning method for skin cancer detection were thoroughly studied on a regular basis. In order to classify images of lesions, it focuses on ANNs, CNNs, KNNs and RBFNs. Additionally, when classifying image data. Due of its stronger connection to computer vision than other neural network architectures, CNN performs better than other varieties of neural networks.

In 2021 Subramanian et al. [18] developed a Convolution neural network to identify and categorize the type of cancer using clinical image data from the past. Using the CNN model, skin cancer may be detected with an accuracy of 80.45%, a false negatives rate in the prediction of less than 10%. The best way for diagnosing skin cancer is then the CNN method.

In 2021 Javaid et al. [19] developed a machine learning and image processing are used to build a novel based technique of classifying skin cancer. In order to segment the images, the OTSU thresholding technique is then used. Following segmentation, the segmented images were used to extract features, such as GLCM features.

In 2019 Unver et al. [20] introduced deep learning-based YOLO and Grabcut algorithm for segmenting the skin lesions in dermoscopic images. This method performs lesion segmentation in four phases: hair removal, location detection, segment the background region, and processed using morphological operators. The proposed pipeline model achieved 93.3% of accuracy on ISBI 2017 dataset.

In 2018 Pathan et al. [21] had developed an automated method for lesion segmentation using lesion-specific characteristics. The dermoscopic hair characteristics were considered when designing a revolutionary hair recognition algorithm. Additionally, chroma-based geometric deformable models are employed to distinguish the lesion from adjacent skin. The speed function includes the chrominance characteristics of the lesion to halt progress at lesion boundaries. This model achieves an average accuracy of 93.4% by using ISIC 2016 dataset.

In 2022 Ghazal et al. [22] had developed a customized pretrained Deep Convolutional Neural Network (DCNN) for identifying both malignant and benign tumors. The final layers of the pre-trained AlexNet model were changed in accordance with the suggested system issue. The binary classification detection modifies the softmax layer. The accuracy obtained using the suggested system model is 87.1%, which is better than accuracy obtained using conventional methods of categorization.

In 2021 Abayomi-Alli et al. [23] had presented a data augmentation method based on covariant Synthetic Minority Oversampling Technique (SMOTE) to address the data scarcity and class imbalance issues. The new data augmentation technique for melanoma skin cancer effectively detection. The suggested approach based on data oversampling in a nonlinear lower-dimensional inserting manifold for fabricating tumor images.

In 2022 Nawaz et al. [24] created an improved deep learning (DL) approach-based combination of DenseNet77 and U-net. In order to compute the more representative set of image features, the DenseNet77 network have implemented at the encoder unit of the U-net system. The decoder of the U-net subsequently segments the derived keypoints. The suggested approach achieved segmentation accuracy for the ISIC-2017 and ISIC-2018 datasets of 99.21% and 99.51%, respectively.

From this literature review, the existing methods only achieve good results on classification and segmentation. Thus, accurate detection of different forms of skin cancer remains a challenging task based on its features. All the above existing techniques frequently produce undesirable blur and alter the texture of the lesion. Furthermore, these prior techniques had high computational cost. To overcome these challenges, the novel DODL net is proposed for accurate identification of different types of skin cancer.

3. Background

The authors [25] presented a loss function for training CNN-based segmentation methods with the intention of specifically decreasing the Hausdorff Distance, it is a common metric for assessing medical image segmentation techniques. The authors have applied spherical convolution kernels of varying radii on segmentation probability maps. Their findings presented that the proposed method decreased Hausdorff Distance and reduced segmentation errors using segmentation metrics. Medical image modalities employ segmentation methods to classify regions of interest (ROI). However, because of the blurred appearance contrast induced by the imaging technique, these modern segmentation methods often struggle to predict the ROI boundary areas. In order to overcome this challenge, the authors presented machine learning inspired method and it achieved higher segmentation precision [26]. Even though, the recent segmentation techniques achieved remarkable results, it still requires massive, representative as well as-annotated datasets. The authors have compared the advantages and conditions of the surveyed methodologies and offered and offered novelties, strategies as well as analytical findings for dealing with unreliable datasets [27].

The authors presented an unsupervised algorithm that focus on synergistic image and feature alignment and proposed method enhances segmentation network by combining. The presentation of images across domains can be transformed with adversarial learning and deep supervision, while the derived features are enhanced in terms of domain-invariance [28]. In order to reduce the cost of computing operations, the authors presented AdaResU-Net, a multi-objective adaptive

convolutional neural network specializing in medical image segmentation. The proposed method is a static architecture that employs U-Nets to enhance effective learning with various hyperparameters. A multi-objective evolutionary algorithm was used to optimize the network while minimizing network size and achieves higher segmentation efficiency with less than 30% trainable parameters [29].

The suggested method [30] was evaluated on medical image datasets and proposed technique. A new image segmentation strategy for diabetic retinopathy is presented. An improved bacterial foraging algorithm combined Levi distribution was employed to segment the diabetic retinopathy images. The suggested method was evaluated on DRIVE and STARE datasets. In feature-based classification techniques, arbitrary functions are selected manually and linked to the image features and output static data which is not dependent on time parameter. These manually selected functions are responsible for obtaining important feature [31]. Upon applying mean, variance and other possible parameters, the classification process commences based on features. The following points are regarded as major drawbacks of this technique [32]:

- Defining or extracting features containing important information in datasets are dependent on the application [33].
- It becomes a tedious process when selecting appropriate weights for image classification
- It increases the chance of complexity as well as calculation cost if correlated features are included.

Most of the raw data-based solutions with the data point require similarity among the pairs that involve the process standard evaluation and classification algorithms. Based on the image characteristics, the choice of algorithm is made with the consideration of accuracy and complexity. However, small number of features require high dimensional inputs (number of samples in different time intervals), which is found to be a major drawback of this method. Naive Bayes is regarded as a simple and potential classifier that could be summarized as follows [34,35]:

Considering a probabilistic model features F1, …, Fn with their associated class (C) in the form P(CjF1;:::; Fn), the label for this particular feature set could be determined by selecting the class with high conditional probability. Finally, the maximum conditional probability among the class P (CjF1;:::; Fn) could be determined with the P(C)P (F1;:::; FnCj) as P(F1;:::; Fn) is common.

$$ArgmaxC = P(CjF1;:::;Fn) \tag{1}$$

This method based on Bayes' theorem:

$$P(CjF1;:::;Fn) = P(C)P(F1;:::;FnjC)/P(F1;:::;Fn \tag{2}$$

Though Naive Bayes provides a robust solution and it is a very simple algorithm, the assumptions made with the features as conditionally independent within the given class label are not usually accurate [36].

Bacterial Foraging Optimization: By employing this optimization unnecessary and irrelevant feature are removed since results in the function subset being the most typical ones. The locations of bacteria in the entire size of the search rooms are 0 or 1, indicating whether or not it is selected as essential for future generation. The bacteria switch to a new random place in every iteration of chemotaxis. As presented in the equation (3), the location of i^{th} bacteria in j^{th} chemotaxis along with k^{th} reproduction stage is determined and used in the proposed method.

$$\Theta^i(j,k) = F_1, F_2, ..., F_m \tag{3}$$

Here, m represents length of features vector. For the next round = 1 or 0 (z = 1, 2, … m) for z^{th} feature is determined.

Particle Swarm Optimization: This algorithm consists of randomly distributed particles with random allocated velocities. Particles move in a d-dimensional problem space, and clustering transform to convergence at global optima. The flow of particles in search space is determined by both entities and their neighbors in the swarm population's travelling experience. If i^{th} particle in the swarm is located at $X_{id}(t)$ with velocity $V_{id}(t)$. Therefore, velocity and location of the particle at next iteration is $V_{id}(t)$ and $X_{id}(t)$ as presented in the equation (4):

$$\begin{aligned} V_{id}(t+1) &= w.V_{id}(t) + c_1.r_1[p_{id}(t) - x_{id}(t)] + c_2.r_2[g_d(t) - x_{id}(t)], x_{id}(t+1) \\ &= x_{id}(t).V_{id}(t+1) \end{aligned} \tag{4}$$

Here, w represents inertia constant achieving an equilibrium with local as well as global search. c_1 and c_2 represents velocity constants, r_1 and r_2 represents independent random numbers of uniform distribution in interval [−1, 1]. $p_{id}(t)$ represents coordinates of local optimum location, $g_d(t)$ represents coordinates of global optimum location by i^{th} particle.

4. Proposed methodology

In this section, a dual optimization based deep learning network (DODL net) has been proposed for detecting the skin cancer. The workflow of the proposed model is shown in Fig. 1. Initially, the images are collected from MNIST: HAM10000 dataset and these images are pre-processed using adaptive median filter. The pre-processed images are given to U-net for segmenting the particular region of the skin lesion. Further, the dual optimization algorithms (BFO and PSO) are used to extract the features from the segmented images. Based on these extracted features, the Deep CNN classifies the seven different classes of skin cancer.

4.1. Dataset description

In this work, the dermoscopic images are gathered from MNIST HAM10000 dataset which is published by International Skin Image Collaboration (ISIC). The dataset was offered as a dataset for the Kaggle detection challenge competition and comprises of dermoscopic images of pigmented lesions derived from Skin Cancer MNIST: HAM10000. The dermoscopic images are collected from various populations, and the dataset included 10,015 dermoscopic images are used for the experimental analysis. Moreover, the gathered dataset is divided into two parts before being fed to the model: 80% as training data and 20% as test data. This dataset is splitted into seven different classes of skin cancer such as Actinic keratoses (akiec), Benign keratosis-like lesions (bkl), Basal cell carcinoma (bcc), dermatofibroma (df), Melanoma (mel), Melanocytic Nevi (nv), and Vascular lesions (vasc). Fig. 2. depicts the sample images of each type of skin cancer.

4.2. Data pre-processing

Pre-processing is essential for emphasizing the subtle variations in dermoscopic images and minimizing the noise. It is used to reduce the unwanted distortions and to enhance the features of the input dermoscopic images. The adaptive median filter is employed to reduce the noise from the dermoscopic images. Data augmentation methods are employed to artificially enhance size of the training dataset, since it helps to avoid over-fitting while training the machine learning model. By training machine learning models on more data variants, able to build skilled models with increased capacity to generalize and identify images. The dermoscopic images are stable to the orientations and transformation methods to create image variants.

4.3. Segmentation phase

U-Net is a type of CNN architecture that is extended with few deviations in the traditional structure. The U-Net is the most prominent encoder-decoder structure that was widely employed for medical image segmentation. Because of its perfect symmetric form and skip

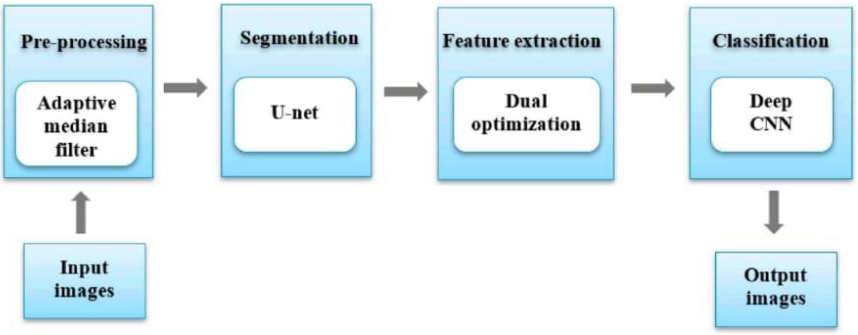

Fig. 1. The workflow of the proposed method.

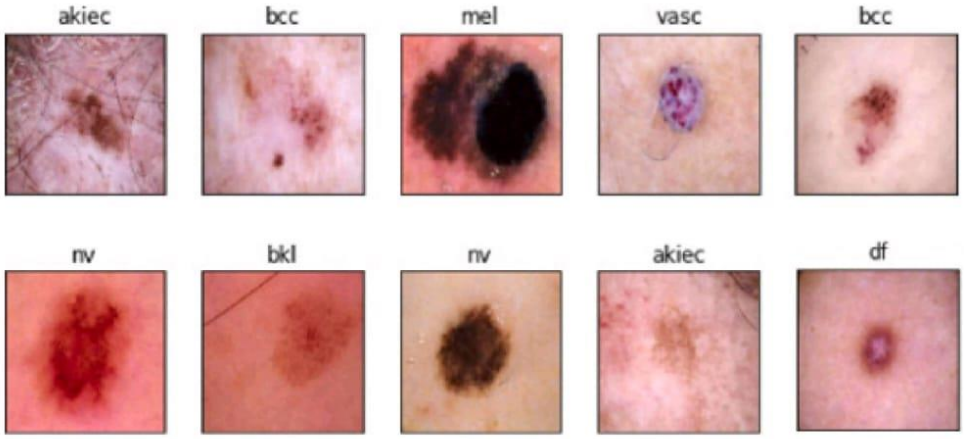

Fig. 2. Sample images of different skin cancer in the dataset.

connection, the U-Net addresses issues with traditional CNN networks that are utilized for medical image segmentation. The U-Net Structure was established as an advancement on fully connected networks generated for segmenting medical images. It is also enlarged in such a way that it can accomplish well on smaller training datasets by augmenting the data provided with medical datasets. The structure of U-net as displayed in Fig. 3.

The U-net architecture composed of numerous layers and that flows in contracting and extending directions. Initially the input layer takes the enhanced datasets. The contraction section compensates with several contraction blocks. Each block accepts the inputs and feed it into the convolutional layers with max-pooling layer. After each block, the amount of feature mappings doubles, allowing architecture to efficiently learn complicated structures. The bottom layer serves as a connection between the contraction layer and the expansion layer. The procedure starts with importing the dataset into network, then rescaling each image. In the processing stage image rescaling is intended to allow for fast and better images. After the rescaling operation, the images are matched. Skin lesion segmentation is a semantic segmentation challenge with much skewed positive and negative samples with lesions constitute such a small part of dermoscopic images. Each convolution procedure begins with the image being filtered, following by non-linearity, and finally max-pooling, which minimises the image size. Following that, the appropriate layers in contracting and extending routes are concatenated, followed by a deconvolutional i.e., up-sampling procedure that increases the image size. The segmented lesion images are given to the feature extraction phase.

4.4. Feature extraction phase

In the feature extraction phase, the segmented lesion images are taken as an input to the dual optimization algorithms for extracting the most relevant features. A proper colour space should be selected for each image processing procedure based on the various colour spaces and their range. A proper colour space improves segmentation precision in subsequent image processing stages. As a result of observing the use of various colour spaces, IHLS method is selected with adequate performance. Another viable reason for selecting this method is that it works effectively with deep neural networks. The skin image is first transformed from RGB to IHLS colour space for accurate mapping of skin colour, The BFO and PSO approach is used to perform optimal thresholding. As discussed in Section 3, BFO and PSO was explained and its function are included, succinctly, the proposed method employs BFO and PSO approach to obtain optimal threshold to perform skin

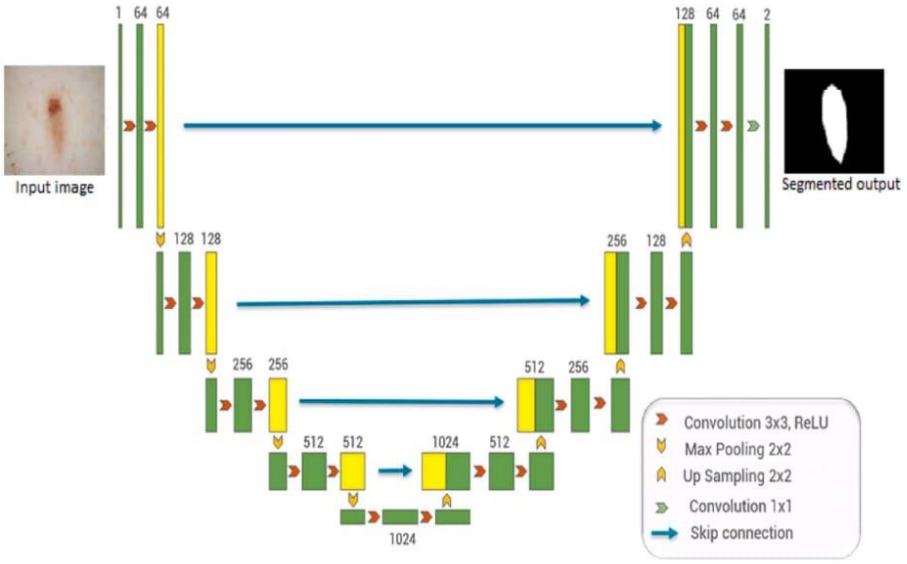

Fig. 3. Architecture of U-network.

segmentation as presented in Fig. 4 and its parameters are presented in Table 1.

By combining BFO and PSO as presented in Fig. 3, global searches and ideal solutions were obtained, even though algorithm is forced into local optima, but it has superior convergence rates. The optimal solutions are stored in descending order after a certain number of full swims. After the chemotactic steps are finished, all bacteria are further mutated by PSO operator in the current process. During this process, all bacteria are stochastically drawn to gbest locations, and BFO handles local searches in different regions.

4.5. Classification phase

CNN is a popular deep learning framework that has been used to solve categorization difficulties. It can acquire both variables and algorithms at the same time, resulting in higher accuracy rate on huge datasets. CNNs can be learned to simulate smooth, strong nonlinear processes, making them suitable for predicting nonlinear changes. The architecture of the implemented CNN is displayed in Fig. 5. Each layer of data in a CNN is represented as a 3D array with the size of $h \times w \times d$, where h and w are spatial dimensions and d represents the channel dimension. The first layer in the image is represented by the pixel size $h \times w$, while the colour channels are represented by the pixel size d. Path-connected receptive fields represent areas in the images as positions in higher layers. Convnets are traditionally created using translation invariance.

By employing permutohedral lattice coordinate system derived from Gauss convolution for its faster processing. The proposed method maps the coordinates of the point for each vertex of a frame in the new coordinate by barycentric interpolation in a permutohedral lattice rather than to a point in the new coordinate and barycentric interpolation produces weight of each vertex and it increases the value of this point if two separate points are predictable to the same vertex co-ordinates.

The resultant of each Conv2D and MaxPooling2D layer is a three-dimensional tensor of form, as presented in Fig. 4 in-terms of height, width and channels. As the development of the neural network, the height and width proportions start to diminish. The first layer handles the number of output channels for each Conv2D layer. As the width and height of each Conv2D layer decreases, additional output channels are included. As a complete network, final resultant tensor from the convolutional layer is given to one or more dense layers for further categorization.

By integrating Dice Loss and Boundary Loss, boundary loss can be controlled, hence the proposed method applied the following boundary loss for skin lesion image segmentation. The composite loss is represented in equation (5) and its objective to overcome the class imbalance problem.

$$\alpha L_{DR}(\theta) + (1-\alpha) L_B(\theta) \tag{5}$$

First, the regularized Dice Loss could be determined with the following equation (6),

$$L_{DR}(\theta) = 1 - 2(\omega_R \sum_{p \int W} r(p) s_\theta(p) \omega_B \sum_{p \int W} (1 - r(p))(1 - s_\theta(p)))/((\omega_R \sum_{p \int W} (r(p) + s_\theta(p)) + ((\omega_B \sum_{p \int W} (2 - r(p) - s_\theta(p)))) \tag{6}$$

Next, the Boundary loss could be determined as

$$L_B(\theta) = \varnothing R_{(p)} s_\theta(p) \tag{7}$$

where if $p \in R$ then $\varnothing R_{(p)} = ||p - z_\theta R(p)||$, otherwise $\varnothing R_{(p)} = -||p - z_\theta R(p)||$. On the other hand, for the foreground $\sum_W (r(p) + f(s_\theta(p)))$ could be used and for the background $\sum_W (1 - r(p))(1 - f(s_\theta(p)))$ could be used. The weight of $L_{DR}(\theta)$ is given by $\omega_R = 1/(\sum_{p \in W} r(p))^2$ and the$\omega_R = 1/(\sum_{p \in W} (1 - r(p)))^2$. The Ω denotes the pixel set in the entire spatial domain.

When compared to image detection and classification, image segmentation did not experience a significant boost in its enhancements with deep learning applications. However, there are many areas in which image registration may be enhanced and explored using deep

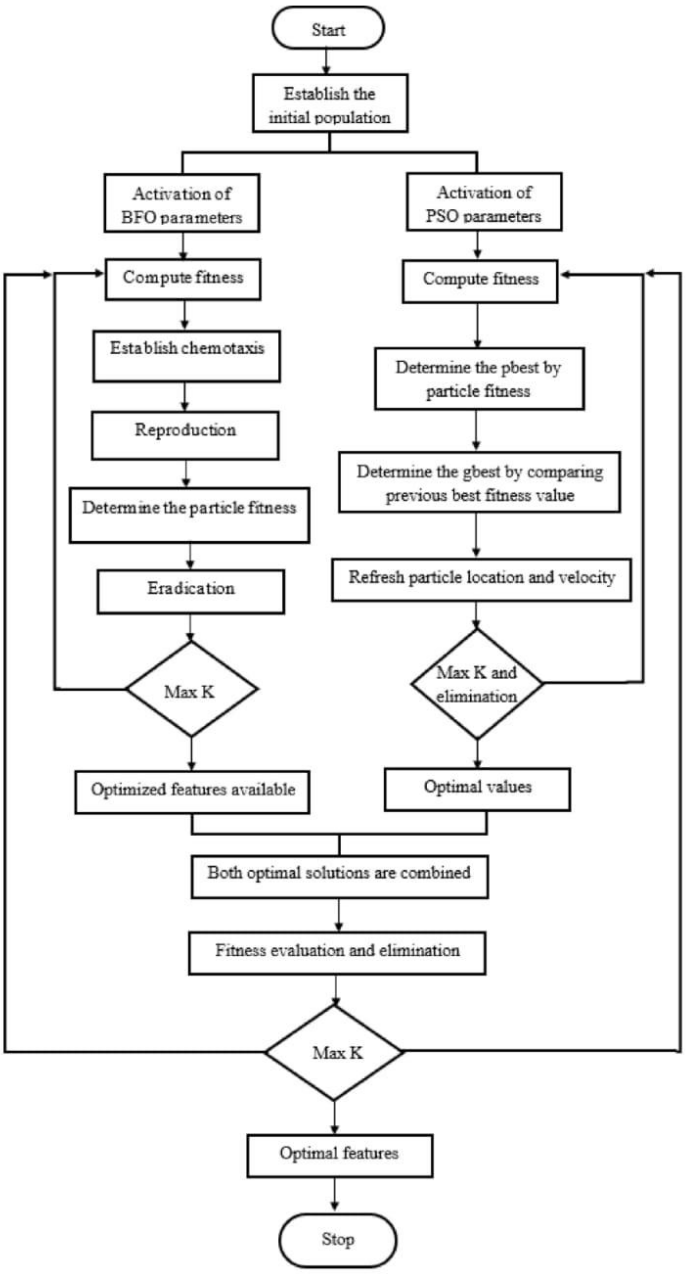

Fig. 4. Flowchart of BFO and PSO.

Table 1
Parameters of dual optimization algorithm.

Parameters	Values
Number of Bacterium	30
Number of Chemotatic Stages	10
Length of Swimming	10
Number of Reproduction Stages	5
Number of Eradication Events	3
Repellent Width	5
Attraction Width	0.3
Repellent Height	0.1
Attraction Depth	0.3
Population size of Particle Swarm	30
Generations	300
Inertia threshold	300
Acceleration Coefficient	3
Number of Load Flows (Population * Generations)	3000

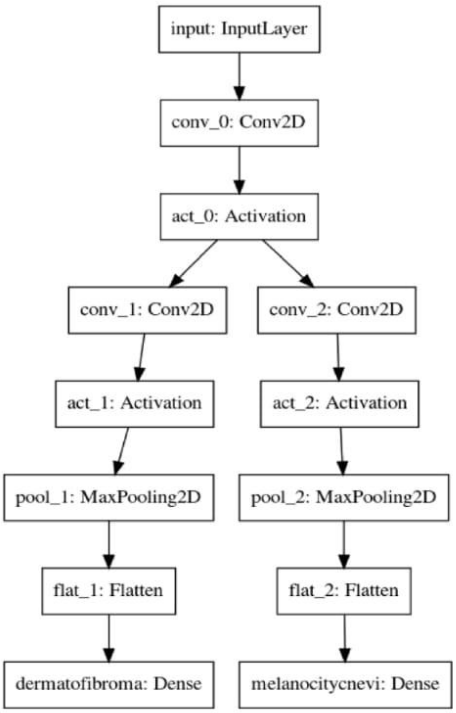

Fig. 5. Architecture of the implemented CNN.

learning. For example, in point-based segmentation, deep learning can determine optimal feasibilities. Prior operators were introduced, and a neural network based on the classical filtered back-projection was built in CNN, which improved roughly constrained angle geometries that could not be addressed by standard analytic inversion methods. However, there were flaws in the discretization and initialization of the filtering processes, which could be corrected by using an effective learning method. This method could be used with systems that learn additional de-streaking sparsifying transforms, such as variational networks. These approaches can be enhanced by creating a matrix inverse that formed a circulant matrix, which included convolution operations, leading to the development of filtering, back-projection, algorithm for rebinning in the re-projection style, which suffers from less resolution loss than conventional interpolation-based techniques. The kernel is altered in a data-dependent manner using the acquired probabilities and it is used to obtain the classification results of skin cancer.

5. Results and discussion

The experimental procedures are carried out using the MATLAB 2019b deep learning toolbox. From the result study, the dermoscopic images from the MNIST HAM10000 dataset are employed for the classification of seven different classes of skin cancer. The proposed DODL framework performs based on deep learning, since the performance analysis and comparative analysis was discussed in this section.

5.1. Performance analysis

The proposed approach was examined and compared to well-known algorithms, as well as analyzed in terms of some metrics. The efficiency of the proposed DODL net was determined using evaluation metrics specificity, precision, accuracy, recall (sensitivity) and F1 score.

$$Precision = \frac{TP}{(TP + FP)} \tag{8}$$

$$Recall = \frac{TP}{(TP + FN)} \tag{9}$$

$$Specificity = \frac{TP}{(TN + FP)} \tag{10}$$

$$F1 - measure = \frac{2 \times (Precision \times Recall)}{Precision + Recall} \tag{11}$$

$$Accuracy = \frac{TP + TN}{Totalnumberofsamples} \tag{12}$$

where True Positive (TP) represents skin cancer identified correctly and False Negative (FN) represents partial skin cancer identified incorrectly. The efficiency of the proposed network for the classification of different types of skin lesion is illustrated in Table 2 and it is graphically represented in Fig. 6.

The results obtained by the proposed DODL net for classifying the seven different types of skin cancer that are represented as class-0 to class-6 as exposed in Table 2. The effectiveness of the model in terms of precision, recall, f1 score and accuracy. The performance evaluation of the proposed DODL net is graphically represented in Fig. 6. The proposed DODL net obtains the overall accuracy of 99.11% for MNIST: HAM10000 dataset. Moreover, the proposed DODL net attains the overall precision, recall, specificity and F1 score of 96.02%, 95.37%, 96.33% and 95.32% respectively.

The confusion matrix is used to validate the proposed DODL net as illustrated in Fig. 7. The performance result acquired by the proposed DODL net for classifying different classes of skin cancer i.e., nv-0, mel-1, bkl-2, bcc-3, akiec-4, vasc-5, and df-6 as exposed in the confusion matrix.

Table 2
Performance assessment of the proposed DODL net.

Classes	Accuracy	Specificity	Precision	Recall	F1 score
0	98.24	96.21	95.21	97.14	97.24
1	98.51	97.05	96.05	94.85	95.02
2	99.06	95. 28	95.18	95.21	94.35
3	98.24	96.27	96.23	94.07	95.08
4	98.95	96.08	95.09	94.56	96.17
5	99.07	96.22	96.20	94.54	94.28
6	99.25	97.24	98.24	97.25	95.15

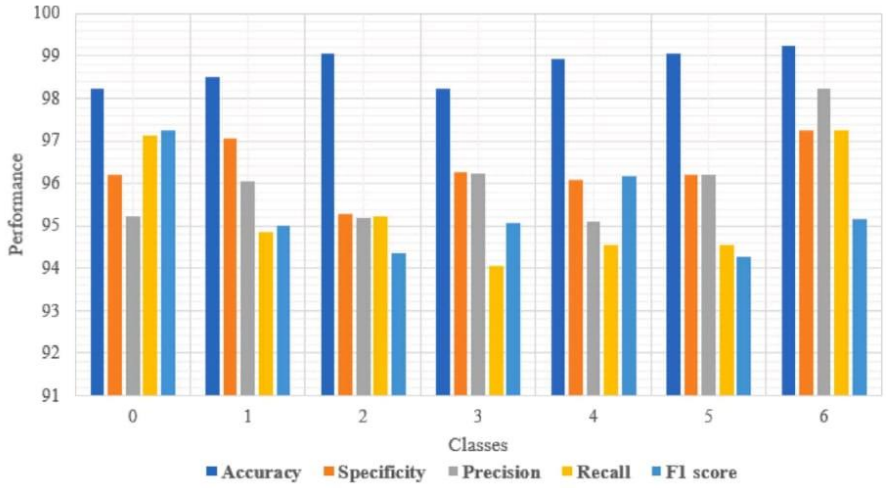

Fig. 6. Graphical portrayal of performance analysis for different skin lesion classes.

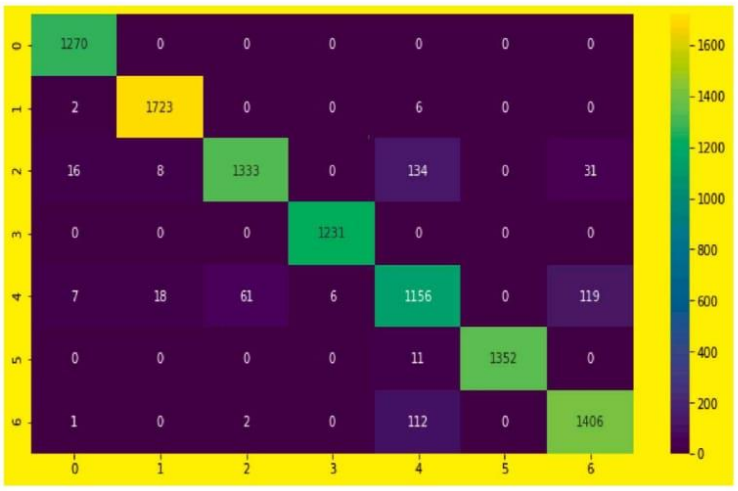

Fig. 7. Confusion matrix of DODL net.

The proposed network achieves reliable accuracy of 98.76% as shown in Figs. 8 and 9. In Fig. 8, the accuracy curve can be seen as the number of epochs on the horizontal axis and the range of accuracy on the vertical axis. As the number of epochs rises, the accuracy of the DODL net is also increases. Fig. 9 displays the epoch and loss range, which shows that when the epochs are increased, the loss of the DODL net decreases. The network is trained for different epochs since the proposed DODL net achieves best results in the 10 epochs. The proposed DODL net achieves high accuracy range for identifying the different classes of skin lesion images.

An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. The ROC generated for seven classes that was determined via TPR and FPR parameters is depicted in Fig. 10 of skin cancer i.e., nv-0, mel-1, bkl-2, bcc-3, akiec-4, vasc-5, and df-6 as exposed in the ROC curve. The proposed approach achieved higher AUC of 0.992, 0.985, 0.99, 0.992, 0.989, 0.995 and 0.995 for the above mentioned seven different classes respectively.

5.2. *Comparative analysis*

The competence of each traditional models is estimated to evaluate that the fallouts of the proposed DODL net attains high accuracy. The assessment was performed between the proposed DODL net with different classifiers like SVM, KNN, RF and DT. The performance assessment was performed through different metrics like precision,

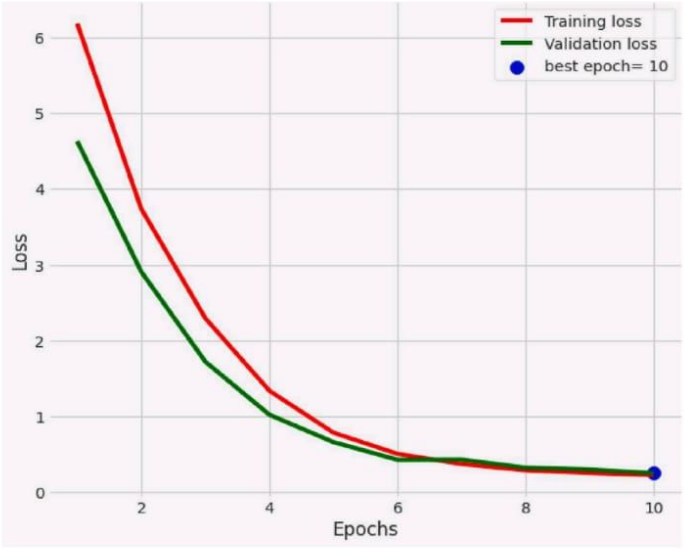

Fig. 8. Training and Validation Loss.

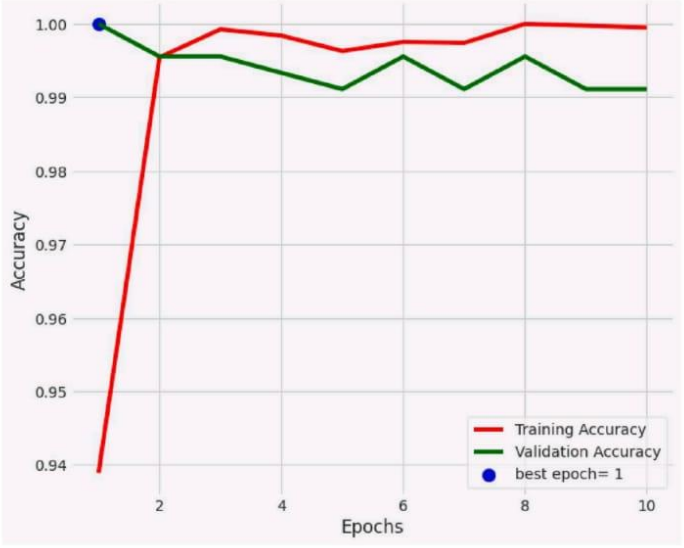

Fig. 9. Training and Validation Accuracy.

specificity, recall, f1 score and accuracy of each technique and the accuracy obtained by proposed DODL net is 99.11%, which was higher than the classic classifiers.

Table 3 shows the comparison was conducted on different ML classifiers using specific metrics for achieving the best classification accuracy. However, the classic classifiers not achieved better results compared to the proposed CNN. The proposed CNN increases the overall accuracy range by 1.06%, 9.37%, 6.33% and 7.65%better than SVM, KNN, RF and DT respectively.

The sample images from the gathered MNIST HAM10000 dataset are used in the testing phase to determine the accuracy of different techniques, as given in Table 4. The comparison was made between state-of-

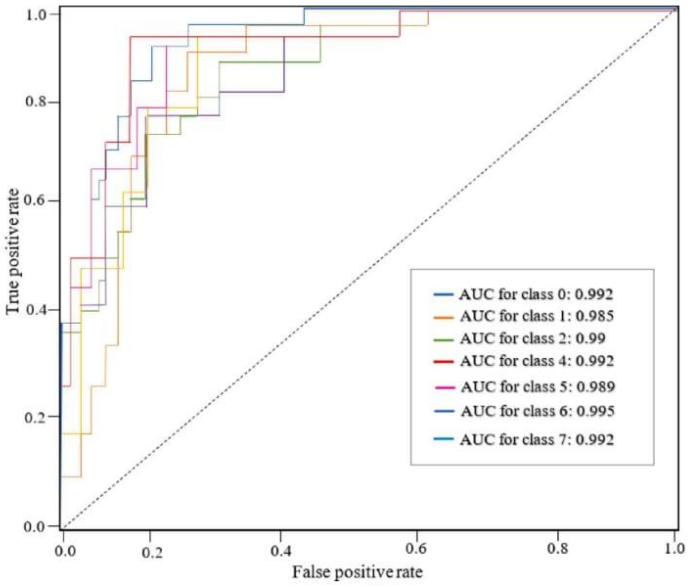

Fig. 10. ROC curve of the proposed DODL model.

Table 3
Comparison between traditional ML classifiers and CNN network.

Classifiers	Accuracy	Specificity	Precision	Recall	F1 score
SVM	93.6	92.5	94.4	92.5	93.4
KNN	89.5	88.4	86.2	85.3	85.7
RF	92.5	91.8	89.0	88.7	88.8
DT	91.2	90.2	87.5	84.6	86.0
CNN	98.76	95.4	96.02	95.37	95.32

Table 4
Accuracy comparison of existing and Proposed model based on MNIST HAM10000 dataset.

Authors	Methods	Accuracy
Khalaf and Dhannoon [15]	AlexUnet+	97.5%
Subramanian [18]	CNN	80.45%
Unver [20]	YOLO	91.2%
Garg [37]	ResNet	90.51%
Proposed	DODL net	98.76%

art models using MNIST HAM10000 dataset by attaining the high classification accuracy. The DODL net improves the overall accuracy range by 1.27%,18.53%, 7.65% and 8.35% better than AlexUnet+ [15], CNN [18], YOLO [20] and Transfer learning based ResNet [37] respectively. However, the existing approaches not achieved well compared to the proposed network based on the MNIST HAM10000 dataset. From Table 4, the DODL approach is clearly superior than other methods.

Table 5 demonstrates the accuracy assessment of proposed DODL approach with dissimilar datasets such as ISIC 2020, Dermnet and PH2. The proposed model achieves 98.76% of accuracy based on the gathered MNIST HAM 10,000 dataset. As a result of this comparison, the proposed network performs poorly on the other datasets, but is highly accurate on the MNIST HAM 10,000 dataset. Moreover, the proposed model yields low accuracy for dermoscopic image analysis using ISIC 2020, Dermnet and PH2 datasets. The proposed model only trained on 10,000 images but Dermnet dataset has 19,500 images and it also has 25 types of skin cancer. So, the proposed model not perform well in identifying the various types of cancer specifically on the Dermnet dataset. The proposed DODL framework improves the overall accuracy of 0.74%, 2.54% and 1.31% better than ISIC 2020, Dermnet and PH2 datasets respectively.

Table 5
Accuracy comparison of Proposed DODL model with various datasets.

Datasets	Accuracy (%)
MNIST HAM10000	98.76
ISIC 2020	98.02
Dermnet dataset	96.25
PH2 dataset	97.46

Table 6 show the efficiency analysis of the proposed model based on the reduction of training data. The training data was reduced from 80% to 50%, since the training data was reduced the quality and validity of the results was changed. The efficiency of the proposed model was very low when trained with 50% of dermoscopic images. The performance of the proposed model increases as the training data increases respectively.

Table 6
Performance analysis with the variations of training data.

Training data	Accuracy	Specificity	Precision	Recall	F1 score
80%	98.76	96.33	96.02	95.37	94.32
70%	96.46	95.21	94.25	95.44	95.51
60%	95.24	94.15	95.04	93.68	92.45
50%	93.05	91.02	90.12	91.24	91.08

6. Conclusion

This work proposed a novel DODL net for detecting the skin cancer. Initially, the dermoscopic images are gathered from publicly available dataset. The collected images are processed in U-net for segmenting the particular region of the skin lesion. Further, the dual optimization algorithms which is combination of BFO and PSO are used to extract the features from the segmented images. Finally, the Deep CNN classifies the seven different classes of skin cancer based on the extracted features. The performance of the DODL net has been evaluated using specific parameters such as precision, recall, F1 score and accuracy. The accuracy attained by the proposed DODL net is 99.11% for MNIST HAM10000 dataset. The DODL net improves the overall accuracy range by 1.27%, 18.53%, 7.65% and 8.35% better than AlexUnet+, CNN, YOLO and Transfer learning based ResNet respectively. The proposed DODL model shows high accuracy level in MNIST HAM 10,000 dataset, but does not perform well on different datasets, so it will be trained on different datasets for further analysis. Evolutionary techniques and deep neural networks have certain problems that need to be addressed and are left for further investigation. In order to compute various aspects of medical image-based systems with arbitrary structures, effective inference for CNN and faster search space for bacterial foraging optimization and particle swarm optimization will be deliberated in future work. Moreover, the proposed system can be extended with advanced deep learning techniques and optimization algorithms to increase the robustness and accuracy in the skin cancer detection.

Author Contributions statement

The authors confirm contribution to the paper as follows: Study conception and design: E.Gomathi, M. Jayasheela; Data collection: M. Thamarai; Analysis and interpretation of results: M.Geetha; Draft manuscript preparation: E.Gomathi, M.Thamarai. All authors reviewed the results and approved the final version of the manuscript.

Research funding.

No Financial support.

Availability of data and material

Data sharing is not applicable to this article as no new data were created or analyzed in this Research.

Human and Animal Rights

This article does not contain any studies with human or animal subjects performed by any of the authors.

Informed consent

I certify that I have explained the nature and purpose of this study to the above-named individual, and I have discussed the potential benefits of this study participation. The questions the individual had about this study have been answered, and we will always be available to address future questions.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgements

The authors would like to thank the reviewers for all of their careful, constructive and insightful comments in relation to this work.

References

[1] M. Naghdi, M. Ghovvati, N. Rabiee, S. Ahmadi, N. Abbariki, S. Sojdeh, A. Ojaghi, M. Bagherzadeh, O. Akhavan, E. Sharifi, M. Rabiee, Magnetic nanostructures in nanomedicine revolution: a review of growing magnetic nanocomposites in biomedical applications. Adv. Colloid Interface Sci. (2022) 102771.10.1016/j.cis.2022.102771.

[2] V.K. Kohli, C. Kohli, A. Singh, Comprehensive multiple-choice questions in pathology: a study guide, Springer Nat. (2022).

[3] A.A. Jeny, A.N.M. Sakib, M.S. Junayed, K.A. Lima, I. Ahmed, M.B. Islam, SkNet: A convolutional neural networks-based classification approach for skin cancer classes, in: 2020 23rd International Conference on Computer and Information Technology (ICCIT) IEEE, December 2020, pp. 1-6. 10.1109/ICCIT51783.2020.9392716.

[4] S. Subha, D.J.W. Wise, S. Srinivasan, M. Preetham, B. Soundarlingam, Detection and differentiation of skin cancer from rashes, in: 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC) IEEE, 2020, pp. 389–393, https://doi.org/10.1109/ICESC48915.2020.9155587.

[5] J. Daghrir, L. Tlig, M. Bouchouicha, M. Sayadi, Melanoma skin cancer detection using deep learning and classical machine learning techniques: a hybrid approach, in: 2020 5th International Conference on Advanced Technologies for Signal and Image Processing (ATSIP) IEEE, 2020, pp. 1–5, https://doi.org/10.1109/ATSIP49331.2020.9231544.

[6] A. Kamboj, A color-based approach for melanoma skin cancer detection, in: 2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC) IEEE, 2018, pp. 508–513, https://doi.org/10.1109/ICSCCC.2018.8703309.

[7] M.K. Islam, M.S. Ali, M.M. Ali, M.F. Haque, A.A. Das, M.M. Hossain, D.S. Duranta, M.A. Rahman, Melanoma skin lesions classification using deep convolutional neural network with transfer learning, in: 2021 1st International Conference on Artificial Intelligence and Data Analytics (CAIDA) IEEE, 2021, pp. 48–53, https://doi.org/10.1109/CAIDA51941.2021.9425117.

[8] M. Vidya, M.V. Karki, Skin cancer detection using machine learning techniques, in: In 2020 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT) IEEE, 2020, pp. 1–5, https://doi.org/10.1109/CONECCT50063.2020.9198489.

[9] J. Tournier, R.E. Smith, D. Raffelt, R. Tabbara, T. Dhollander, M. Pietsch, D. Christiaens, B. Jeurissen, C. Yeh, A. Connelly, MRtrix3: a fast, flexible and open software framework for medical image processing and visualisation, bioRxiv (2019).

[10] S. Kazeminia, C. Baur, A. Kuijper, B. Ginneken, N. Navab, S. Albarqouni, A. Mukhopadhyay, GANs Med. Image Anal. ArXiv, abs/1809.06222, 2020. <https://doi.org/10.1016/j.artmed.2020.101938>.

[11] K.V. Prasad, B. Susmitha, T. Chennu, K.M.S.B. Murthy, M.V. Datta, Skin Cancer lesions classification using probabilistic neural network, J. Phys. Conf. Ser. 2335 (1) (2022) 012028), https://doi.org/10.1088/1742-6596/2335/1/012028.

[12] V.A. Kumar, V. Mishra, M. Arora, Convolutional neural networks for malignant and benign cell classification using dermatoscopic images, in: In 2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV) IEEE, 2021, pp. 1040–1044, https://doi.org/10.1109/ICICV50876.2021.9388605.

[13] F.T. Chan, Z.X. Wang, A. Goswami, A. Singhania, M.K. Tiwari, Multi-objective particle swarm optimisation based integrated production inventory routing planning for efficient perishable food logistics operations, Int. J. Prod. Res. 58 (17) (2020) 5155–5174.

[14] S. Joseph, O.O. Olugbara, Preprocessing effects on performance of skin lesion saliency segmentation, Diagnostics 12 (2) (2022) 344, https://doi.org/10.3390/diagnostics12020344.

[15] M. Khalaf, B.N. Dhannoon, Skin Lesion Segmentation based on U-Shaped Network, Karbala Int J Mod Sci 8 (3) (2022) 493–502.

[16] S. Garg, J. Balkrishan, Skin lesion segmentation in dermoscopy imagery, Int. Arab J. Inf. Technol. 19 (1) (2022) 29–37.

[17] M. Dildar, S. Akram, M. Irfan, H.U. Khan, M. Ramzan, A.R. Mahmood, S.A. Alsaiari, Saeed, A.H.M.M.O. Alraddadi, M.H. Mahnashi, Skin cancer detection: a review using deep learning techniques, Int. J. Environ. Res. Public Health 18 (10) (2021) 5479.

[18] R.R. Subramanian, D. Achuth, P.S. Kumar, K.N. kumar Reddy, S. Amara, and A.S. Chowdary, Skin cancer classification using Convolutional neural networks, in: 2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence) IEEE, 2021, pp. 13-19. 10.1109/Confluence51648.2021.9377155.

[19] A. Javaid, M. Sadiq, F. Akram, Skin cancer classification using image processing and machine learning, in: In 2021 international Bhurban conference on applied sciences and technologies (IBCAST) IEEE, 2021, pp. 439–444, https://doi.org/10.1109/IBCAST51254.2021.9393198.

[20] H.M. Ünver, E. Ayan, Skin lesion segmentation in dermoscopic images with combination of YOLO and grabcut algorithm, Diagnostics 9 (3) (2019) 72, https://doi.org/10.3390/diagnostics9030072.

[21] S. Pathan, K.G. Prabhu, P.C. Siddalingaswamy, Hair detection and lesion segmentation in dermoscopic images using domain knowledge, Med. Biol. Eng. Comput. 56 (11) (2018) 2051–2065, https://doi.org/10.1007/s11517-018-1837-9.

[22] T.M. Ghazal, S. Hussain, M.F. Khan, M.A. Khan, R.A. Said, M. Ahmad, Detection of benign and malignant tumors in skin empowered with transfer learning. Comput. Intell. Neurosci. (2022).

[23] O.O. Abayomi-Alli, R. Damasevicius, S. Misra, R. Maskeliunas, A. Abayomi-Alli, Malignant skin melanoma detection using image augmentation by oversamplingin nonlinear lower-dimensional embedding manifold, Turk. J. Electr. Eng. Comput. Sci. 29 (8) (2021) 2600–2614.

[24] M. Nawaz, T. Nazir, M. Masood, F. Ali, M.A. Khan, U. Tariq, N. Sahar, R. Damaševičius, Melanoma segmentation: a framework of improved DenseNet77

and UNET convolutional neural network, Int. J. Imag. Syst. Technol. 32 (6) (2022) 2137–2153.
[25] Convolutional Neural Networks. IEEE Transactions on Medical Imaging, 39, 499-513.
[26] D. Yang, H. Roth, X. Wang, Z. Xu, A. Myronenko, D. Xu, Enhancing Foreground Boundaries for Medical Image Segmentation. ArXiv, abs/2005.14355. (2020).10.48550/arXiv.2005.14355.
[27] N. Tajbakhsh, L. Jeyaseelan, Q. Li, J.N. Chiang, Z. Wu, X. Ding, Embracing imperfect datasets: a review of deep learning solutions for medical image segmentation, Med. Image Anal 63 (2020), 101693, https://doi.org/10.1016/j.media.2020.101693.
[28] C. Chen, Q. Dou, H. Chen, J. Qin, P. Heng, Unsupervised bidirectional cross-modality adaptation via deeply synergistic image and feature alignment for medical image segmentation, IEEE Trans. Med. Imaging 39 (2020) 2494–2505, https://doi.org/10.1109/TMI.2020.2972701.
[29] M.G. Calisto, S. Lai-Yuen, AdaResU-Net: Multiobjective adaptive convolutional neural network for medical image segmentation, Neurocomput 392 (2020) 325–340, https://doi.org/10.1016/j.neucom.2019.01.110.
[30] N. Rajini, Image segmentation for diabetic retinopathy using modified bacterial foraging optimization algorithm, Ind. J. Public Health Res. Dev. 10 (2019) 1313–1319, https://doi.org/10.4018/978-1-7998-3222-5.ch007.
[31] D.C. Benz, G. Benetos, G. Rampidis, E.V. Felten, A. Bakula, A. Šustar, K. Kudura, M. Messerli, T. Fuchs, C. Gebhard, A. Pazhenkottil, P. Kaufmann, R. Buechel, Validation of deep-learning image reconstruction for coronary computed tomography angiography: Impact on noise, image quality and diagnostic accuracy, J. Cardiovasc. Comput. Tomogr. (2020), https://doi.org/10.1016/j.jcct.2020.01.002.
[32] B.D. Vos, B.H. Velden, J. Sander, K. Gilhuijs, M. Staring, I. Išgum, Mutual information for unsupervised deep learning image registration, Medical Imaging: Image Processing. (2020), https://doi.org/10.1117/12.2549729.
[33] G. Haskins, U. Kruger, P. Yan, Deep learning in medical image registration: a survey, Mach. Vis. Appl. 31 (2020) 1–18, https://doi.org/10.1007/s00138-020-01060x.
[34] D. Karimi, H. Dou, S. Warfield, A. Gholipour, Deep learning with noisy labels: exploring techniques and remedies in medical image analysis, Med. Image Anal 65 (2020), 101759, https://doi.org/10.1016/j.media.2020.101759.
[35] H. Chan, R.K. Samala, L. Hadjiiski, C. Zhou, Deep learning in medical image analysis, Adv. Exp. Med. Biol. 1213 (2020) 3–21.
[36] Y. Fu, Y. Lei, T. Wang, W. Curran, T. Liu, X. Yang, Deep learning in medical image registration: a review, Phys. Med. Biol. (2020).
[37] R. Garg, S. Maheshwari, A. Shukla, Decision support system for detection and classification of skin cancer using CNN, in: Innovations in Computational Intelligence and Computer Vision: Proceedings of ICICV 2020, Springer, Singapore, 2021, pp. 578-586.