

Skin Cancer Detection Using Convolutional Neural Network

Dharaneesh S, Gowtham R, Sridhar S, Vindhiya R

Bachelor Of Technology

Department Of Information Technology

SNS College Of Technology,Coimbatore-35

ABSTRACT

The project is entitled “Skin Cancer Detection using CNN”. The main objective of this project is to detect the cancer cells at an early stage to prevent the cancer deaths. Skin cancer is a type of cancer that grows in the skin tissue, which can cause damage to the surrounding tissue, disability, and even death. The accuracy of diagnosis and the early proper treatment can minimize and control the harmful effects of skin cancer. In the proposed system the cancer cells are identified and classified using Convolutional Neural Network (CNN). In this method Adam optimizer is used with a learning rate of 0.01. Adam optimizer provides the best performance with an accuracy value of 89%. Three different types of skin cancers are classified in this system namely, Melanoma, Squamous Cell Carcinoma (SCC) and Basal Cell Carcinoma (BCC). Finally it is converted into a website for easy access to the user. The proposed approach not only learns a global map for skin lesions, but also acquires the local contextual information, such as lesion boundary. It can, therefore, accurately segment lesions within a given skin image, even in the presence of fuzzy boundaries and complex textures.

CHAPTER - 1

INTRODUCTION

The abnormal growth of cells which cannot be controlled by healthy cells by a transformation in the skin is known as skin cancer. The early diagnosis of skin cancer is required because it spreads fast. Automated skin lesion classification in dermoscopy images is an essential way to improve the diagnostic performance and reduce cancer deaths. Skin cancer is a malignancy disease that is often found in Indonesia. The most common skin cancers in Indonesia are basal cell carcinoma 65.5%, followed by squamous cell carcinoma 23% and melanoma 7.9%. Even though the number of Melanoma cancer is smaller than Basal Cell Carcinoma and Squamous Cell Carcinoma, the death rate tends to be greater, which causes 75% of deaths from skin cancer. The most dangerous skin cancer is melanoma, which has a high death rate, especially if it is not early detected. Non-melanoma skin cancers, such as basal cell carcinoma (bcc) and squamous cell carcinoma (sc) are more common and only partially lead to disability or death. Accurate diagnosis and early detection of skin cancer can avoid the worst effects of skin cancer. The automatic skin disorders classification can help people in identifying skin disorders that occur and immediately consult with medical personnel to get appropriate medical treatment. The proposed method in this study uses CNN with Adam optimizer. The optimizer is used to increase the performance of the CNN. It is used to tune the parameters like epoch, learning rate etc.. This method is used to detect and classify the cancer cells at an early stage and prevent cancer deaths.

CHAPTER - 2

LITERATURE REVIEW

“Skin Cancer Diagnostic using Machine Learning Techniques - Shearlet Transform and Naïve Bayes Classifier”

AUTHOR: S. Mohan Kumar, J. Ram Kumar, K. Gopalakrishnan

In this study, the classification of melanoma using shearlet transform coefficients and naïve

Bayes classifiers is discussed. The melanoma images are decomposed by the shearlet transform. Then, from the shearlet coefficients, predefined numbers of (50, 75 and 100) coefficients are selected from the decomposed subbands. The selected subband coefficients are directly applied to the naïve Bayes classifier. Performance of the skin cancer classification system is measured in terms of accuracy. The accuracy obtained here is 90%.

“Melanoma Skin Cancer Detection using Image Processing and Machine Learning” AUTHOR:

Vijayalakshmi M M

In this paper a completely automated system of dermatological disease recognition through lesion images, a machine intervention in contrast to conventional medical personnel-based detection is discussed. Our model is designed into three phases: compromising of data collection and augmentation, designing model and finally prediction. They have used multiple AI algorithms like Convolutional Neural Network and Support Vector Machine and amalgamated it with image processing tools to form a better structure, leading to higher accuracy of 85% **“Computer Aided Melanoma Skin Cancer Detection Using Image Processing”**

AUTHORS: Shivangi Jain, Vandana Jagtap, Nitin Pise

In this paper, they have presented a computer aided method for the detection of Melanoma Skin Cancer using Image Processing tools. The input to the system is the skin lesion image and then by applying novel image processing techniques, it analyses it to conclude about the presence of skin cancer. The Lesion Image analysis tools checks for the various Melanoma parameters Like Asymmetry, Border, Colour, Diameter, (ABCD) etc. by texture, size and shape analysis for image

segmentation and feature stages. The extracted feature parameters are used to classify the image as Normal skin and Melanoma cancer lesion.

“Deep Learning Solutions for Skin Cancer Detection and Diagnosis” AUTHORS: Hardik Nahata, Sathya P. Singh

This project aims to develop a skin cancer detection CNN model which can classify the skincancer types and help in early detection [5]. The CNN classification model will be developed in Python using Keras and Tensorflow in the backend. The model is developed and tested with different network architectures by varying the type of layers used to train the network including but not limited to Convolutional layers, Dropout layers, Pooling layers and Dense layers. The model will also make use of Transfer Learning techniques for early convergence. The model will be tested and trained on the dataset collected from the International Skin Imaging Collaboration (ISIC) challenge archives.

“Skin Cancer Detection Using Convolutional Neural Network” AUTHORS: Mahamudul Hasan, Surajit Das Barman, Samia Islam

This paper proposed an artificial skin cancer detection system using image processing and machine learning method. The features of the affected skin cells are extracted after the segmentation of the dermoscopic images using feature extraction technique. A deep learning based method convolutional neural network classifier is used for the stratification of the extracted features. An accuracy of 89.5% have been achieved after applying the publicly available data set.

CHAPTER - 3 SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

Skin cancer is a type of cancer that grows in the skin tissue, which can cause damage to the surrounding tissue, disability, and even death. The accuracy of diagnosis and the early proper treatment can minimize and control the harmful effects of skin cancer. In the existing system, the cancer cells are detected using SVM, Naive Bayes and various other classification methods. The accuracy level is low in these methods. It also has some limitations like High Computational load and poor discriminatory power, LBP doesn't differentiate the local texture region, FNN is slow training for large feature set. Less accuracy in classification

3.2 PROPOSED SYSTEM:

In our proposed system, a Convolution neural network (CNN) is used and there exist three layers. First layer is the input layer where the data sets are trained on. Input layer collects data that are delivered and adding some weight with it that goes to hidden layers. Neurons of hidden layers separate the features from the data to find out a pattern. The pattern is then used as the basis to output layers that select appropriate classes. The accuracy level and loss is calculated. Finally, binary classification is used for cancer cells. Using Convolution Neural Network (CNN) the three types of cancer cells are classified namely Melanoma, Basal Cell Carcinoma (BCC) and Squamous Cell Carcinoma (SCC).

CHAPTER - 4

SYSTEM SPECIFICATION

4.1 HARDWARE SPECIFICATIONS:

System : INTEL

Hard Disk : 500 GB

Monitor Resolution : 1024 x 768 (or) Higher RAM : 4 GB

4.2 SOFTWARE SPECIFICATIONS: Operating System : Windows (or) Linux Software Tools :

Python 3.7, Thonny IDE

CHAPTER - 5 METHODOLOGY

5.1 MODULES

IMPORTING THE DATASET:

The image datasets are collected and placed in different folders with different names and uploaded for pre-processing. All subjects were independently labeled as “normal” or “Scc,bcc,Melanoma”. Labeling was first evaluated with the original images on a picture archiving communication system (PACS) and secondly with the resized images that were used for the actual learning data. Datasets were defined as the internal dataset and temporal dataset, with the temporal dataset used to evaluate the test. The internal dataset was randomly split into training (70%), validation (15%), and test (15%) subsets. The distribution of the internal test dataset consisted of 32% Normal, 32% SCC, and 34% BCC and 32% Melanoma.

DATASET PRE-PROCESSING

Images come in different shapes and sizes. They also come through different sources. Taking all these variations into consideration, we need to perform some pre-processing on any image data. The first step of data pre-processing is to make the images of the same size. Here we have used auto resizing for training to make all the images in the dataset to convert into same resolution. Feature extraction is done to reduce the amount of redundant data in the given dataset. Also, the reduction of the data facilitates speed learning and generalization steps in the machine learning process.

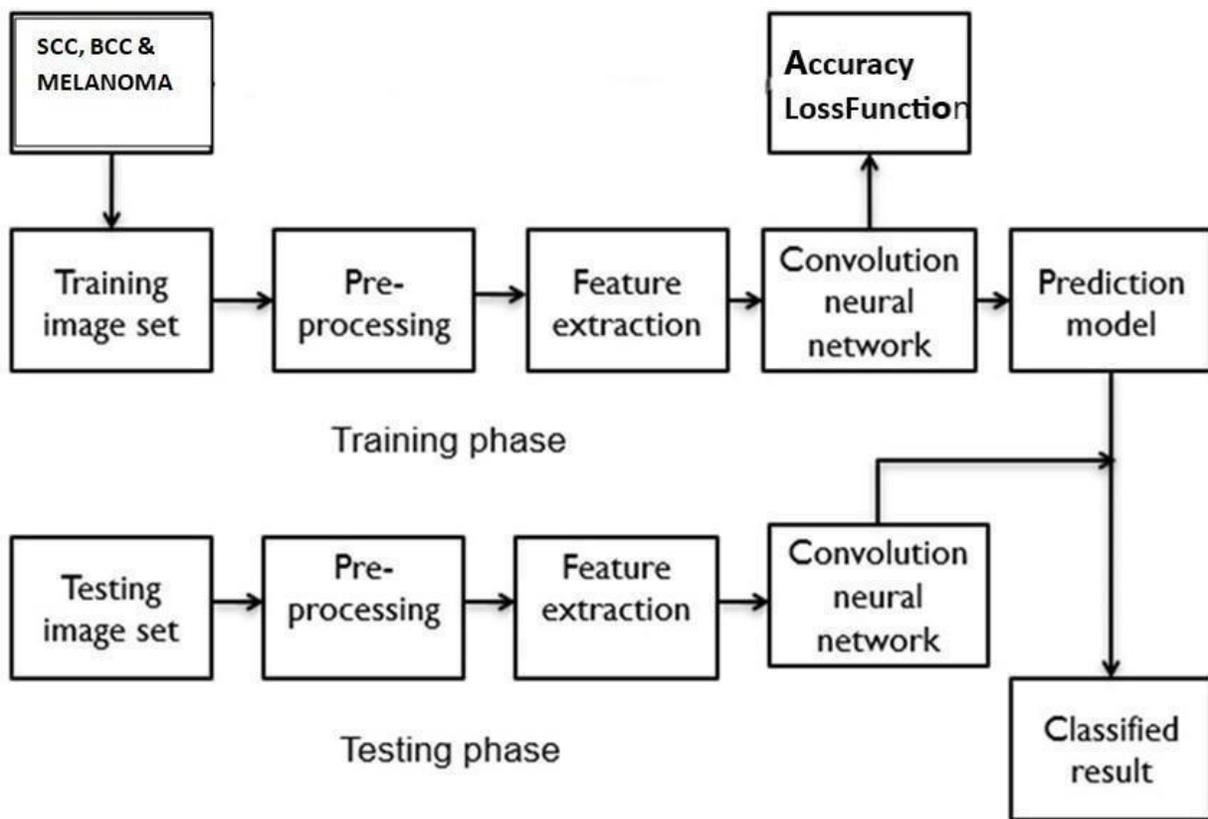
CREATION OF CNN MODEL:

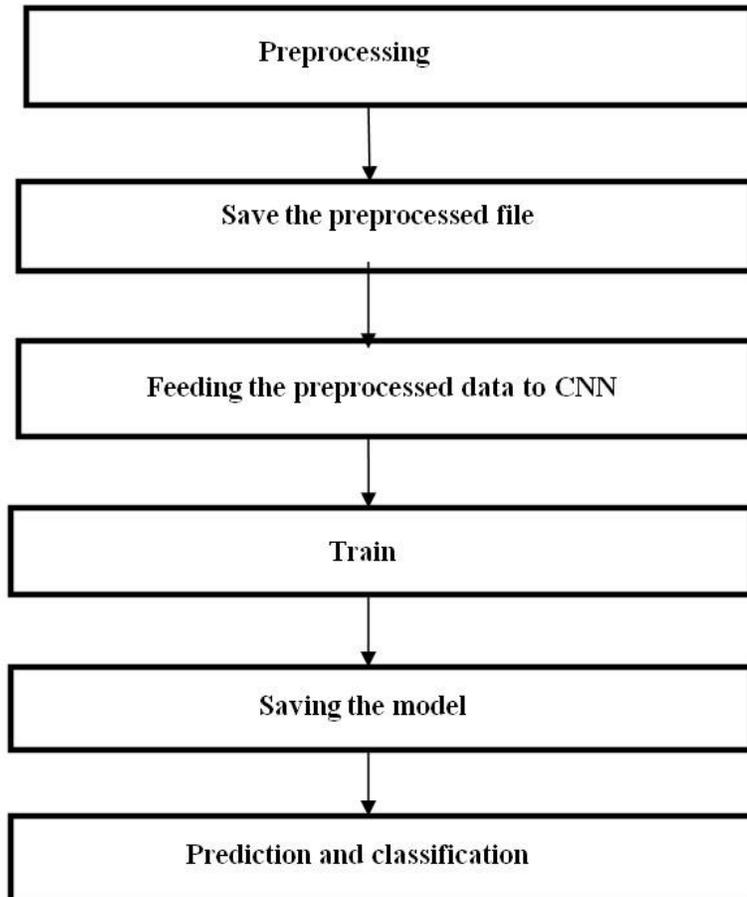
After data preprocessing the CNN training model is created using resnet architecture. The accuracy of the model is optimized using hyper parameter adjustment. Adam optimizer is used with a learning rate of 0.01. Adam optimizer provides the best performance with an accuracy value of 89%. Optimizer is used to increase the performance of the CNN model. It is used to tune parameters like epoch, learning rate etc..

TESTING AND CLASSIFICATION

After training the dataset the model file is created and during classification test image is imported and pre-processed and CNN prediction is done using the model file and the result is classified. During this phase using the trained model the cancer cells are classified namely, Melanoma, Squamous cell carcinoma (scc) and Basal cell carcinoma (bcc). Finally the results will be displayed in a webpage for easy access to the user.

5.2 DESIGN AND BLOCK DIAGRAM





The image datasets are collected and placed in different folders with different names and uploaded for pre-processing. All subjects were independently labeled as “normal” or “Scc,bcc,Melanoma”. Labeling was first evaluated with the original images on a picture archiving communication system (PACS) and secondly with the resized images that were used for the actual learning data. Datasets were defined as the internal dataset and temporal dataset, with the temporal dataset used to evaluate the test. The first step of data pre-processing is to make the images of the same size. Here we have used auto resizing for training to make all the images in the dataset to convert into same resolution. Feature extraction is done to reduce the amount of redundant data in the given dataset. Also, the reduction of the data facilitates speed

learning and generalization steps in the machine learning process. After data preprocessing the CNN training model is created using resnet architecture. The accuracy of the model is optimized using hyper parameter adjustment. Adam optimizer is used with a learning rate of 0.01. Adam optimizer provides the best performance with an accuracy value of 89%. Optimizer is used to increase the performance of the CNN model. It is used to tune parameters like epoch, learning rate etc.. After training the dataset the model file is created and during classification test image is imported and pre-processed and CNN prediction is done using the model file and the result is classified. During this phase using the trained model the cancer cells are classified namely, Melanoma, Squamous cell carcinoma(scc) and Basal cell carcinoma(bcc).

5.3 IMPLEMENTATION:

CNN:

In deep learning, a convolutional neural network (CNN) is a type of deep neural networks, which deals with the set of data to extract information about that data. Like images, sounds or videos etc. can be used in the CNN for the data extraction. There are mainly three things in CNN. First one is local receptive field and then shared weight and biases and the last one is activation and pooling. In CNN, first the neural networks are trained using a heavy set of data so that the CNN can extract the feature of given input. When the input is given, first image preprocessing is done then the feature extraction occurs on the basis of set of data stored and then the classification of data is done and output is shown as the result. The CNN can deal with those input only for what the neural network is trained and the data is saved. They are used in image and video recognition, recommender systems, image classification, medical image analysis, and natural language processing.

DATASET:

One major advantage of using CNNs over NNs is that you do not need to flatten the input images to 1D as they are capable of working with image data in 2D. This helps in retaining the “spatial” properties of images. So here we are using x-ray data base which consist of three categories Frontal, Maxillary and Normal.

Activation function

Activation function serves as a decision function and helps in learning of intricate patterns. The selection of an appropriate activation function can accelerate the learning process. In literature, different activation functions such as sigmoid, tanh, maxout, SWISH, ReLU, and variants of ReLU, such as leaky ReLU, ELU, and PReLU are used to inculcate non-linear combination of features.

Operations using NumPy:

Using NumPy, a developer can perform the following operations:

- Mathematical and logical operations on arrays.
- Fourier transforms and routines for shape manipulation.
- Operations related to linear algebra. NumPy has in-built functions for linear algebra and random number generation.

TENSORFLOW IMPLEMENTATION:

The main components in a TensorFlow system are the client, which uses the Session interface to communicate with the master, and one or more worker

processes, with each worker process responsible for arbitrating access to one or more computational devices (such as CPU cores or GPU cards) and for executing graph nodes on those devices as instructed by the master. We have both local and distributed implementations of the TensorFlow interface. The local implementation is used when the client, the master, and the worker all run on a single machine in the context of a single operating system process (possibly with multiple devices, if for example, the machine has many GPU cards installed). The distributed implementation shares most of the code with the local implementation, but extends it with support for an environment where the client, the master, and the workers can all be in different processes on different machines. In our distributed environment, these different tasks are containers in jobs managed by a cluster scheduling system. These two different modes are illustrated. Most of the rest of this section discusses issues that are common to both implementations, while This discusses some issues that are particular to the distributed implementation.

CHAPTER -6 RESULTS

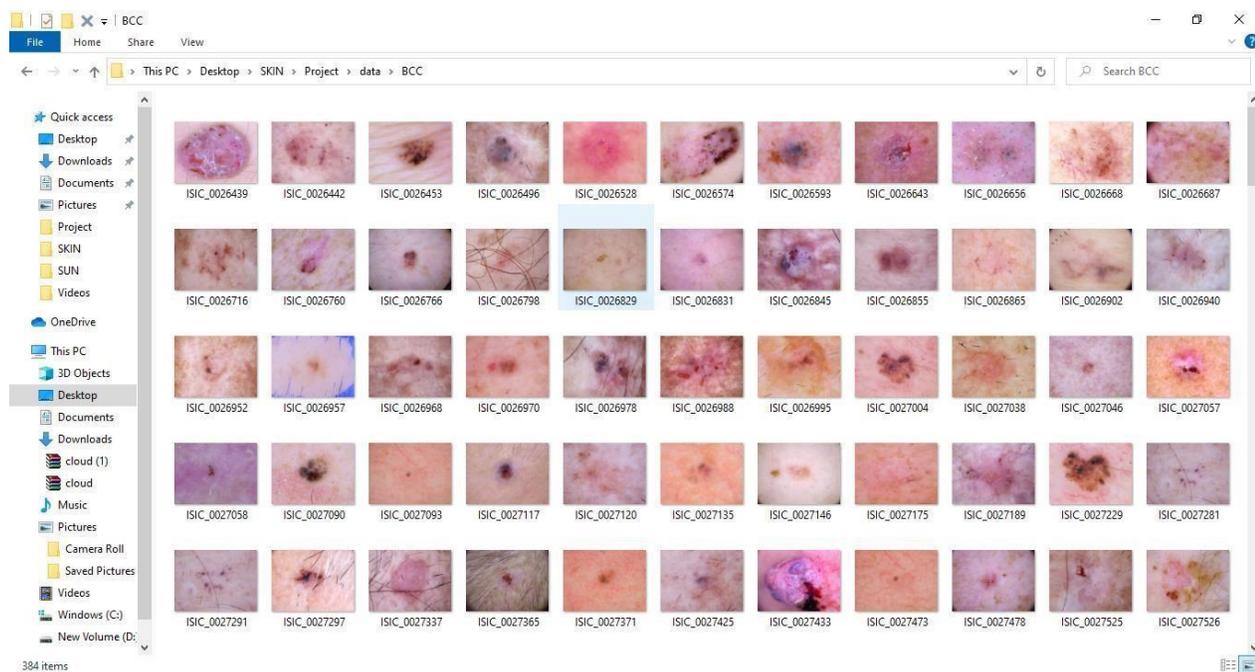


Fig.6.1 DATASET FOR BCC

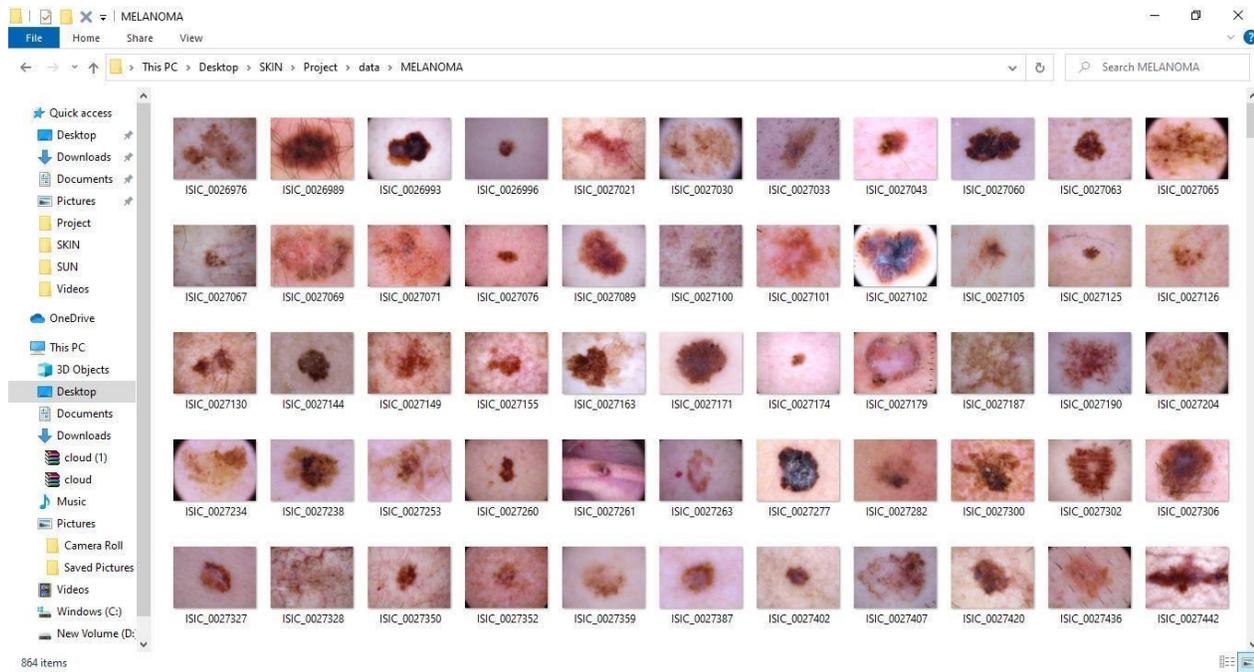


Fig:6.3 DATASET FOR SCC

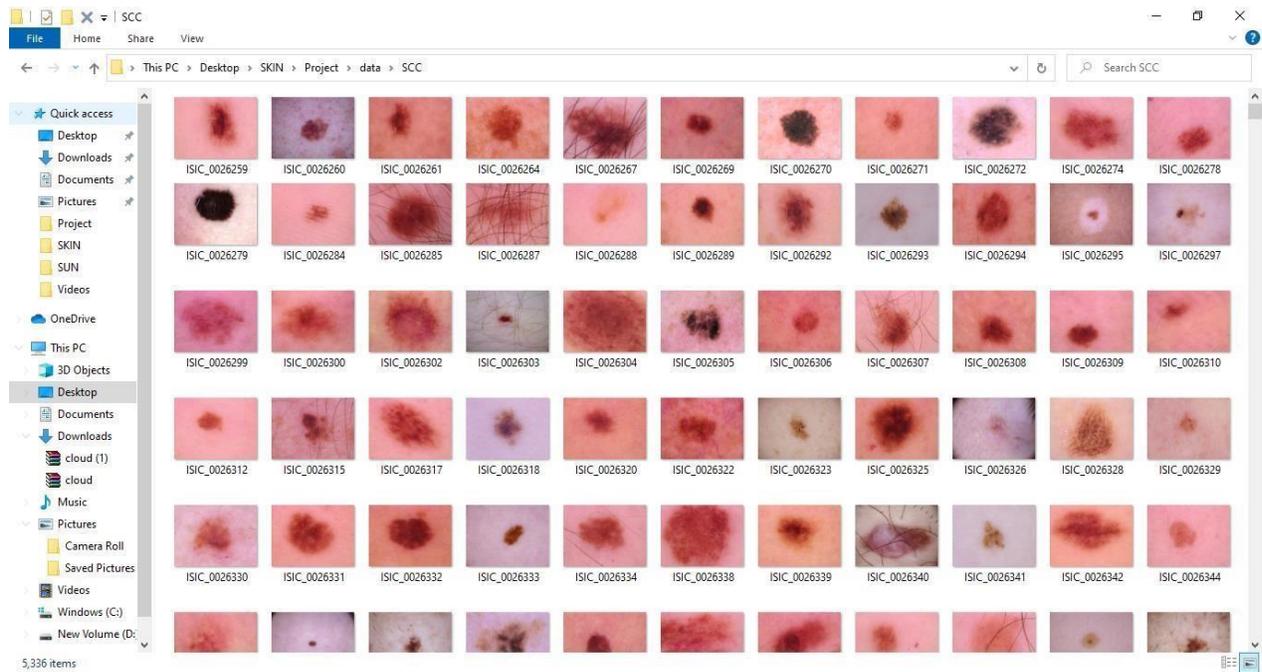


Fig:6.2 DATASET FOR MELANOMA

```
Thonny - C:\Users\elcot\Desktop\SKIN\Project\main.py @ 70:1
File Edit View Run Tools Help

main.py x main1.py x
16
17 # The size of the images that your neural network will use
18 IMG_SIZE = 50
19
20 # Checking or all images in the data folder
21 for category in CATEGORIES :
22     path = os.path.join(DATADIR, category)
23     for img in os.listdir(path):
24         img_array = cv2.imread(os.path.join(path, img), cv2.IMREAD_GRAYSCALE)
25         img_array = cv2.equalizeHist(img_array)
26         #img_array = cv2.Canny(img_array, threshold1=3, threshold2=10)
27         #img_array = cv2.medianBlur(img_array,1)
28
29 training_data = []
30
31 def create_training_data():
32     for category in CATEGORIES :
33         path = os.path.join(DATADIR, category)
34         class_num = CATEGORIES.index(category)
35         for img in os.listdir(path):

Shell x
322/322 [=====] - 74s 228ms/step - loss: 0.1904 - accuracy: 0.9278 - val_loss: 0.3076 - val_accuracy: 0.8959
Saved model to disk
2022-04-21 15:16:00.329317: W tensorflow/python/util/util.cc:368] Sets are not currently considered sequences, but this may change in the futu
re, so consider avoiding using them.

Python 3.7.3 (C:\Users\elcot\AppData\Local\Programs\Python\Python37\python.exe)
>>> %Run main.py
>>>
```

Fig:6.4: ACCURACY

```
Thonny - C:\Users\elcot\Desktop\SKIN\Project\main1.py @ 69:1
File Edit View Run Tools Help

main.py x |main1.py x
1 import tensorflow as tf
2 from tensorflow.python.keras.models import Sequential
3 from tensorflow.python.keras.layers import Dense, Dropout, Activation, Flatten, Conv2D, MaxPooling2D
4 import pickle
5 from keras.models import model_from_json
6 from keras.models import load_model
7 import matplotlib.pyplot as plt
8 import numpy as np
9 # Opening the files about data
10 X = pickle.load(open("X.pickle", "rb"))
11 y = pickle.load(open("y.pickle", "rb"))
12
13 # normalizing data (a pixel goes from 0 to 255)
14 X = X/255.0
15
16 # Building the model
17 model = Sequential()
18 model.add(Conv2D(32, (3, 3), input_shape = X.shape[1:]))
19 model.add(Activation("relu"))
20 model.add(MaxPooling2D(pool_size=(2, 2)))

Shell x
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
Epoch 1/5
322/322 [=====] - 81s 228ms/step - loss: 0.3335 - accuracy: 0.8872 - val_loss: 0.3027 - val_accuracy: 0.8906
Epoch 2/5
322/322 [=====] - 81s 253ms/step - loss: 0.2801 - accuracy: 0.8986 - val_loss: 0.2847 - val_accuracy: 0.8976
Epoch 3/5
322/322 [=====] - 80s 248ms/step - loss: 0.2487 - accuracy: 0.9075 - val_loss: 0.2986 - val_accuracy: 0.8924
Epoch 4/5
322/322 [=====] - 78s 243ms/step - loss: 0.2245 - accuracy: 0.9153 - val_loss: 0.2867 - val_accuracy: 0.8924
Epoch 5/5
```

Fig:6.5: ACCURACY

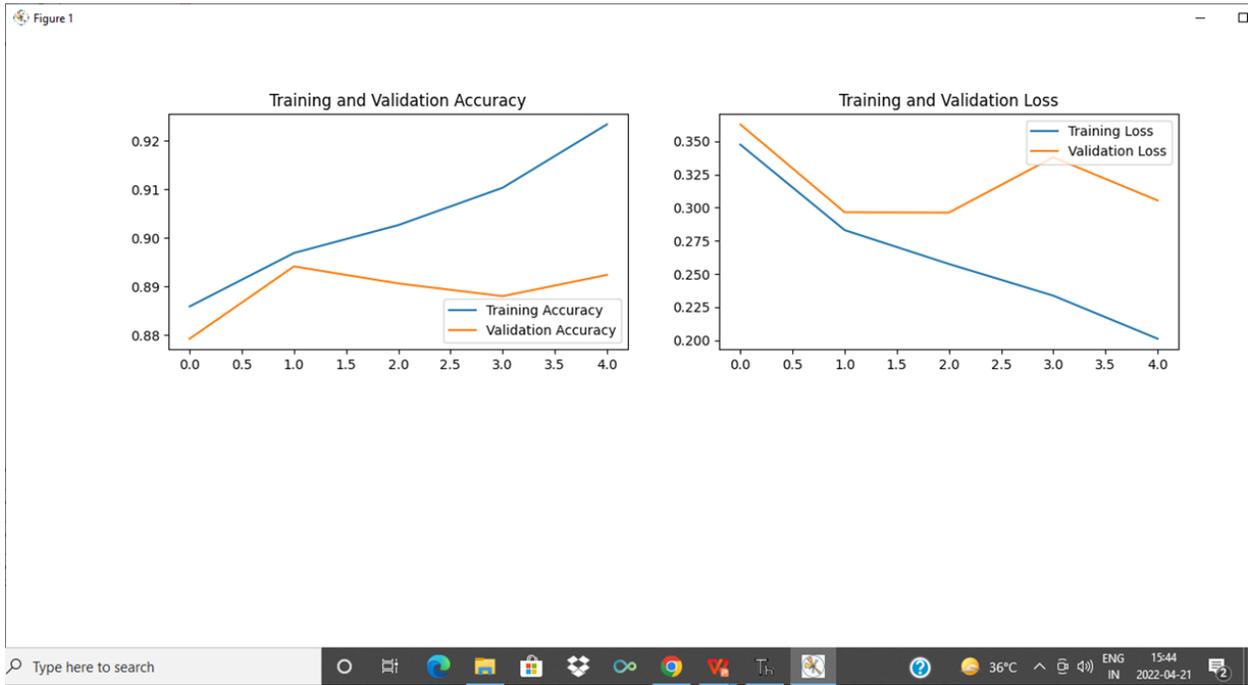


Fig:6.6: GRAPH FOR ACCURACY

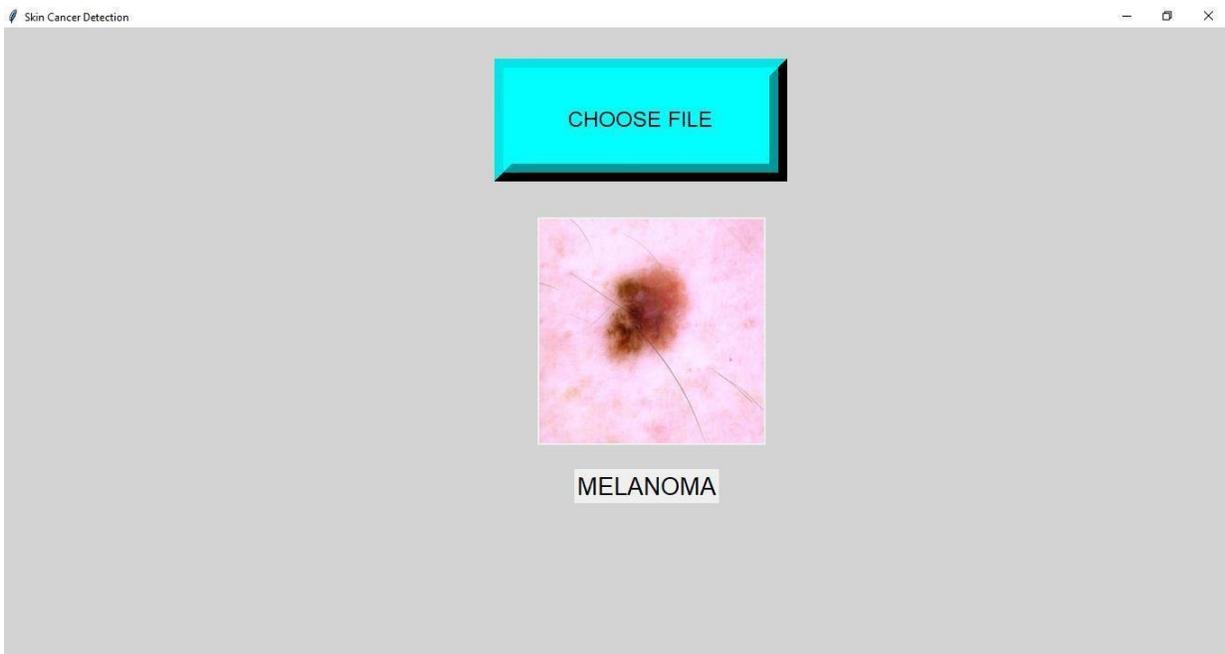


Fig 6.7: MELANOMA CLASSIFICATION

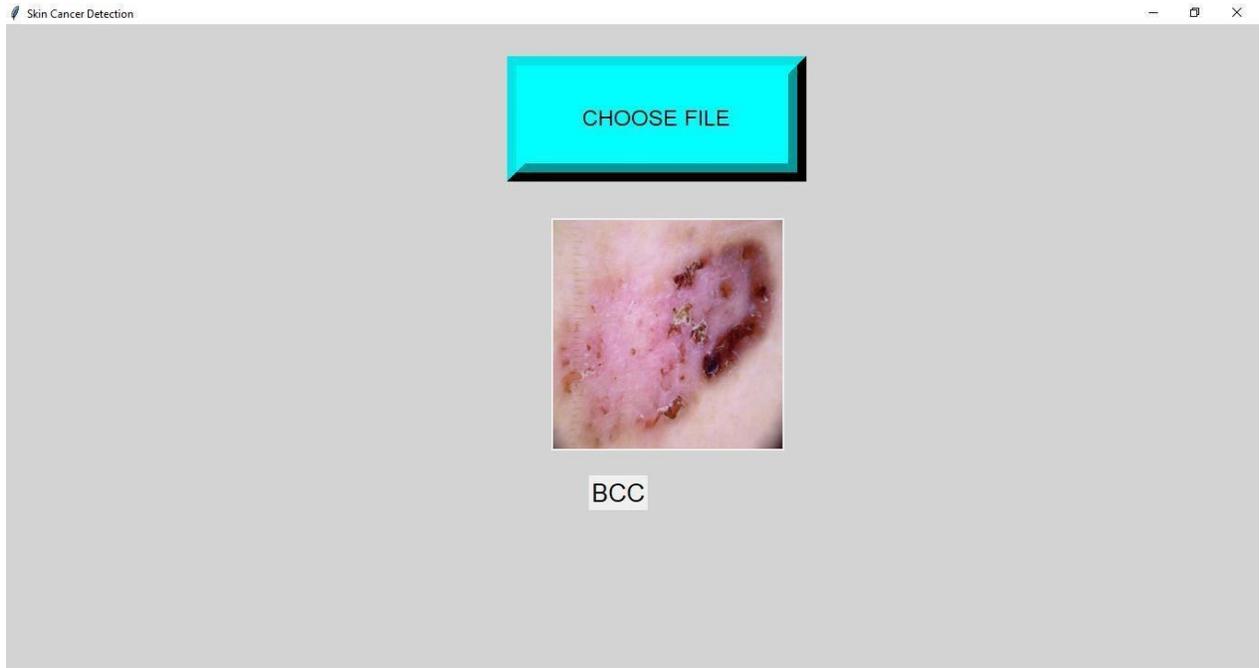


Fig 6.8: BCC CLASSIFICATION

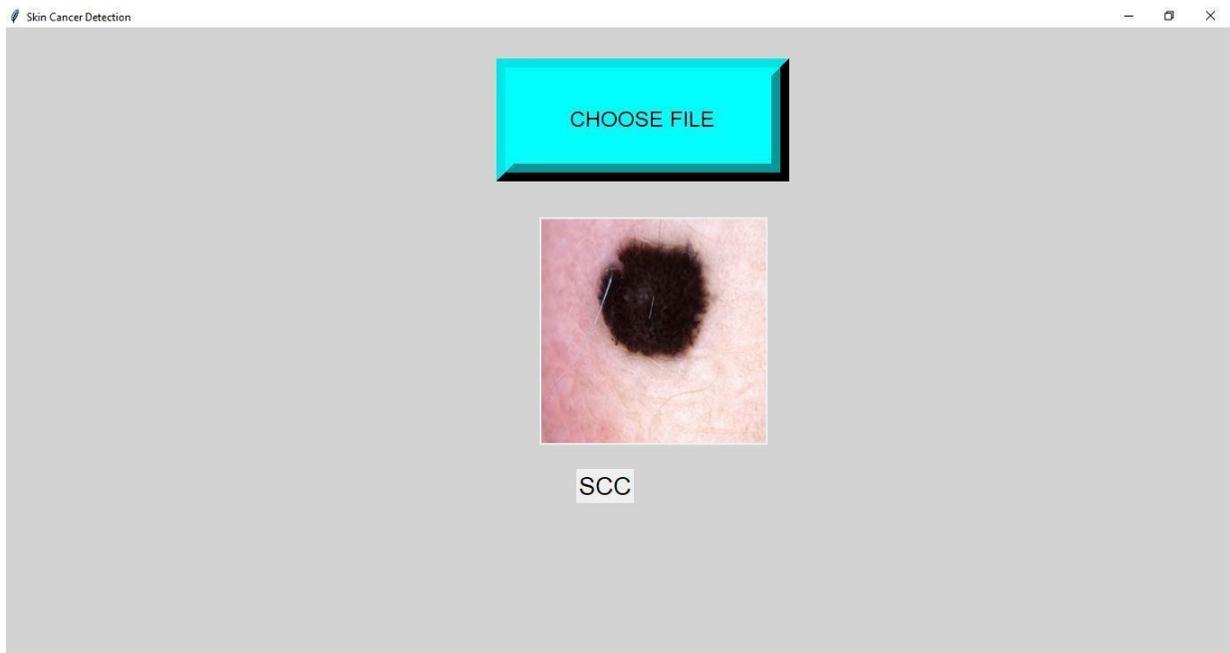


Fig 6.9: SCC CLASSIFICATION

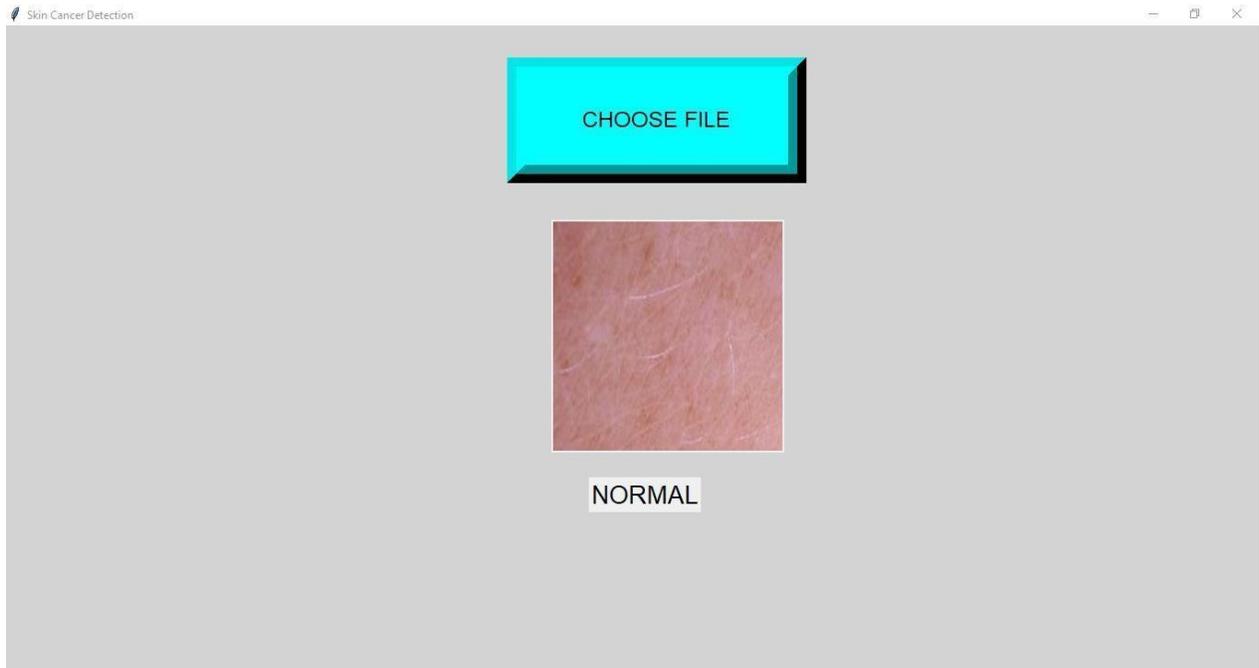


Fig 6.10: NORMAL SKIN CLASSIFICATION

CHAPTER-7 CONCLUSION

In this project, we have proposed a reliable and robust method for skin cancer detection in highly cluttered images using CNN. The cluttered images are obtained using dermoscopy images. The image sequences also provide the candidate cancer region proposals done by multilevel graph cut. We have introduced a verification step in which the proposed region is classified into SCC, BCC, Melanoma, Normal classes, Thus, determining whether the proposed region is truly affected by skin cancer or not. We applied CNN features to machine learning algorithms to achieve better performance.

FUTURE SCOPE:

For future improvements we are planning to train CNN on a large dataset with more labeled skin lesions and collect real-time skin clinical images from the hospital for more skin cancer diagnosis, use transfer learning models such as VGG-16 and mobile net to obtain higher performance.

APPENDIX

```
import numpy as np
import os
from matplotlib import pyplot as plt
import cv2
import random
import pickle

file_list = []
class_list = []
DATADIR = "data"

# All the categories you want your neural network to detect
CATEGORIES=["BCC","MELANOMA","NORMAL","SCC"]

# The size of the images that your neural network will use
IMG_SIZE = 50

# Checking or all images in the data folder for category in CATEGORIES :

path = os.path.join(DATADIR, category)
for img in os.listdir(path):

img_array = cv2.imread(os.path.join(path, img), cv2.IMREAD_GRAYSCALE)
img_array = cv2.equalizeHist(img_array)

#img_array = cv2.Canny(img_array, threshold1=3, threshold2=10)
#img_array = cv2.medianBlur(img_array, 1)

training_data = []
def
```

```
create_training_data():          for category in CATEGORIES :

path = os.path.join(DATADIR, category) class_num

= CATEGORIES.index(category) for img in os.listdir(path): try

:

img_array = cv2.imread(os.path.join(path, img), cv2.IMREAD_GRAYSCALE) #img_array =

cv2.Canny(img_array, threshold1=3, threshold2=10)

#img_array = cv2.medianBlur(img_array,1) img_array = cv2.equalizeHist(img_array) new_array =

cv2.resize(img_array, (IMG_SIZE, IMG_SIZE)) training_data.append([new_array, class_num except

Exception as e: pass create_training_data()

random.shuffle(training_data) X = [] #features y = [] #labels for features, label in training_data:

X.append(features) y.append(label)

X = np.array(X).reshape(-1, IMG_SIZE, IMG_SIZE, 1)

# Creating the files containing all the information about your model pickle_out = open("X.pickle", "wb")

pickle.dump(X, pickle_out) pickle_out.close() pickle_out = open("y.pickle", "wb") pickle.dump(y,

pickle_out) pickle_out.close() pickle_in = open("X.pickle", "rb")

X = pickle.load(pickle_in)

import tensorflow as tf from tensorflow.python.keras.models import Sequential

from tensorflow.python.keras.layers import Dense, Dropout, Activation, Flatten,

Conv2D, MaxPooling2D

import pickle from keras.models import model_from_json from keras.models import load_model import

matplotlib.pyplot as plt import numpy as np

# Opening the files about data X = pickle.load(open("X.pickle", "rb")) y =
```

```
pickle.load(open("y.pickle", "rb"))
# normalizing data (a pixel goes from 0 to 255)

X = X/255.0 # Building the model model = Sequential() model.add(Conv2D(32, (3, 3), input_shape =
X.shape[1:])) model.add(Activation("relu")) model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.4)) model.add(Flatten()) model.add(Dense(128, activation='relu'))
model.add(Dense(4, activation='softmax')) # Compiling the model using some basic parameters
model.compile(loss="sparse_categorical_crossentropy", optimizer="adam", metrics=["accuracy"])
y=np.array(y)# Training the model, with 40 iterations

# validation_split corresponds to the percentage of images used for the validation phase compared to all the
images history = model.fit(X, y, batch_size=32, epochs=5, validation_split=0.1)

# Saving the model model_json =model.to_json() with open("model.json", "w") as json_file :

json_file.write(model_json) model.save_weights("model.h5") print("Saved model to disk")
model.save('CNN.model')

# Printing a graph showing the accuracy changes during the training phase acc = history.history['accuracy']
val_acc = history.history['val_accuracy'] loss = history.history['loss'] val_loss = history.history['val_loss']
acc=np.array(acc) val_acc=np.array(val_acc) loss=np.array(loss) val_loss=np.array(val_loss) epochs_range
= range(5) plt.figure(figsize=(15, 15)) plt.subplot(2, 2, 1) plt.plot(epochs_range, acc, label='Training
Accuracy') plt.plot(epochs_range, val_acc, label='Validation Accuracy') plt.legend(loc='lower right')
plt.title('Training and Validation Accuracy') plt.subplot(2, 2, 2) plt.plot(epochs_range, loss, label='Training
Loss')

plt.plot(epochs_range, val_loss, label='Validation Loss') plt.legend(loc='upper right') plt.title('Training and
Validation Loss') plt.show() import numpy as np import matplotlib.pyplot as plt import cv2 import os import
```

```
tensorflow as tf from keras.models import load_model

from tkinter import * import tkinter.messagebox import PIL.Image import PIL.ImageTk from tkinter import
filedialog

CATEGORIES=["BCC","MELANOMA","NORMAL","SCC"]

root = Tk() root.title("Skin Cancer Detection") root.state('zoomed') root.configure(bg='#D3D3D3')
root.resizable(width = True, height = True) value = StringVar() panel = Label(root) model =
tf.keras.models.load_model("CNN
.model") def prepare(file):

IMG_SIZE = 50 img_array = cv2.imread(file, cv2.IMREAD_GRAYSCALE) img_array =
cv2.equalizeHist(img_array)

#img_array = cv2.Canny(img_array, threshold1=3, threshold2=10)#img_array =
cv2.medianBlur(img_array,1) new_array = cv2.resize(img_array,
(IMG_SIZE,IMG_SIZE)) return
new_array.reshape(-1, IMG_SIZE, IMG_SIZE, 1) defdetect(filename):

prediction = model.predict(prepare(filename)) prediction
= list(prediction[0]) print(prediction) l=CATEGORIES[prediction.index(max(prediction))]
print(CATEGORIES[prediction.index(max(prediction))])
)

value.set(CATEGORIES[prediction.index(max(prediction))]) i=int(prediction.index(max(prediction))) def
ClickAction(event=None): filename = filedialog.askopenfilename() img = PIL.Image.open(filename) img =
img.resize((250,250)) img = PIL.ImageTk.PhotoImage(img) global panel panel
= Label(root, image = img) panel.image = img panel =panel.place(relx=0.435,rely=0.3) detect(filename)
```

```
button = Button(root, text='CHOOSE FILE', font=(None, 18),
activeforeground='red', bd=20, bg='cyan', relief=RAISED, height=3, width=20, command=ClickAction)
button = button.place(relx=0.40, rely=0.05)

result = Label(root, textvariable=value, font=(None, 20)) result = result.place(relx=0.465, rely=0.7)
root.mainloop()
```

REFERENCES

- [1] S. Mohan Kumar, J. Ram Kumar, K. Gopalakrishnan **“Skin Cancer Diagnostic using Machine Learning Techniques - Shearlet Transform and Naïve Bayes Classifier”** International Journal of Engineering and Advanced technology (IJEAT)ISSN: 2249-8958.
- [2] Vijayalakshmi M M Assistant Professor, Department of Information Science & Engineering GSSSIETW, Mysuru, Karnataka, India **“Melanoma Skin Cancer Detection using Image Processing and Machine Learning”** International Journal of Trend in Scientific Research and Development (IJTSRD) e-ISSN: 2456 - 6470.
- [3] Shi Wang¹ and Melika Hamian Payame Noor University (PNU), Tehran, Iran **“Skin Cancer Detection Based on Extreme Learning Machine and a Developed Version of Thermal Exchange Optimization”** Hindawi Computational Intelligence and Neuroscience.
- [4] Mohammad Ali Kadampur, Sulaiman AI Riyae, **“Skin cancer detection: Applying a deep learning based model driven architecture in the cloud for classifying dermal cell images”** Informatics in Medicine Unlocked.
- [5] Kinnor Das, Clay J. Cockerell, Anant Patil, Paweł Pietkiewicz, Mario Giulini, Stephan Grabbe and Mohamad Goldust **“Machine Learning and**

Its Application in Skin Cancer”, Internatioal Journal Environmental andPublic Health.

- [6] M.Vidhya, Maya V Karki, Ramaiah Institute of Technology, Bangalore, India **“Skin Cancer Detection using Machine Learning Techniques”** IEEE International Conference on Electronics, Computing and Communication Technologies.
- [7] Mario Fernando Jojoa Acosta, Liesle Yail Caballero Tovar, Maria BegonyaGarciaZapirian **“Melanoma diagnosis using deep learning techniques on dermatoscopicimages”** Technical Advance.
- [8] Shivangi Jain, Vandana Jagtap, Nitin Pise **“Computer Aided Melanoma Skin Cancer Detection Using Image Processing”** Procedia Computer Science.
- [9] Hardik Nahata, Sathya P. Singh **“Deep Learning Solutions for Skin Cancer Detection and Diagnosis”**In book: Machine Learning with HealthCare Perspective
- [10] Mahamudul Hasan, Surajit Das Barman, Samia Islam **“Skin Cancer Detection Using Convolutional Neural Network”** 5th Internatuional Conference



NoviTech
the innovation partner

www.novitech.in

info@novitech.in

TO WHOM IT MAY CONCERN

This is to certify that “**DHARANEESH S (19IT009), GOWTHAM R (19IT013), SRIDHAR S (19IT046), VINDHIYA R (19IT057)**” Final year, B.Tech Information Technology Students of SNS College of Technology has successfully completed their project titled “**Skin Cancer Detection Using Convolutional Neural Network**” under the guidance of **NoviTech R&D Pvt Ltd** during the academic year 2022-2023.

For NoviTech R&D Pvt Ltd



Vinothkumar A*

(Manager)

Head Office :

2nd Floor, Sai Sruthi Complex, 3 Ramarkoil Street, Ramnagar,
Coimbatore - 641 009. Tamilnadu.
Ph : 9597115101

Branch Office :

No: 65 /707, Basement Floor, Kaloor Towers (Opp. AJ HALL),
K.K Road, Kaloor, Kochi - 682 017.
Ph : 0484 2955101