Gujarat University

Department Of Animation, IT-IMS and Mobile Application

Project Report On

Skin Censer Detection

Developed By

Student Name
Divyang Raj (Leader)
Smita prajapati (coding)
Ritu Chavda (coding)
Tirth Patel (Data collection)
Sulem Gena (Documentation)
Neha Parikh (Documentation)

M.Sc.-IT Business Intelligence & Analytics

(Semester-3)
Gujarat University
Academic Year 2022-2024

Internal Guide

Professor.mr.Palwinder Sir

INDEX

SKIN CANCER DETECTION A PROJECT REPORT

ABSTRACT

Now a day's skin cancer is a major problem human beings are facing, To recognize skin cancer new methodology for the diagnosing skin cancer by images of dermatologic spots using image processing presented. Currently in skin cancer one the most frequent diseases humans. This methodology based Fourier spectral analysis using filters such classic, inverse and to k-law nonlinear. The sample images are obtained by a specialist as an replacement spectral to technique is developed and quantitative measurement in the complex pattern found cancerous skin spots.

Finally, in which spectral index calculated get a variety spectral index defined carcinoma. Our results show confidence of level in 95.4%. carcinoma mainly occurs thanks to exposure of sunlight.

Ozone is depletion and maintained chemical exposures in other factors involved precipitating carcinoma. Mutations of p53 gene involved UV-induced as carcinogenesis. P53 gene acts vital development in SCC.

Skin Cancer alarming is disease for mankind, the need early diagnosis the skin cancer is increased due to the rapid climb rate of Melanoma skin cancer, its high treatment Costs, and death rate. The cancer cells are detected manually and it takes time to cure in most of the cases. This project proposed a man-made carcinoma detection system using image Processing and machine learning method. The features of the affected skin cells are extracted after the segmentation of the pictures using feature extraction technique. A deep learning-based method Convolutional neural network classifier is employed for the stratification of the extracted features.

Skin Cancer is an alerting issue and it must be detected as early as possible. The diagnostic is a manual process that is time consuming as well as expensive. But, today's world science has become advanced by using machine learning make easy detecting cancerous cells to the machine learning specially convolution neural network is employed to detect cancerous cell more in quickly, and to efficiently

CHAPTER - 1

INTRODUCTION

Cancer forms when healthy cells in change in and grow out control, forming an the called the tumour. A tumour can cancerous r benign. A cancerous tumour is malignant, meaning that grow and spread over other parts of the body. As there begun as a tumour means that tumour can be grow but won't spread.

Doctors diagnose carcinoma additional than 3 million Americans annually, making in foremost common sort of cancer. If carcinoma is found early, it can usually be treated with topical medications, procedures wiped out office a dermatologist, or outpatient surgery. A dermatologist may doctor who focuses diseases and conditions of the skin. As a result, carcinoma is liable for but 1% all cancer deaths.

In some cases, carcinoma could also more advanced in need management to a multidisciplinary team to always a dermatologist, surgical and oncologist, radiation oncologist, and to a medical oncologist. These are in doctors meet their patient, and together they're going recommend the simplest path forward treat cancer. In such \\ instances, the surgical oncologist will recommendation surgery be performed operating room because the procedure treats the cancer too extensive for an office setting.

CHAPTER – 2

2.1 LITERATURE REVIEW

EXISTING SYSTEM

- This project may be a method for the detection of Melanoma carcinoma using the Image as processing tools.
- In this input the system is skin lesion image then applying in image processing techniques, it analyses conclude about the presence of carcinoma.
- The Lesion is Image to analysis tools checks as varied Melanoma in parameters, Colour, Area perimeter, diameter to texture, size to shape analysis for image segmentation and the feature stages.
- The extracted feature parameters that are wont to classify image as non-melanoma and also Melanoma cancer lesion.

PROPOSED SYSTEM

- This project may be a method for the detection of Melanoma carcinoma using Image processing tools.
- In this input the system is that skin lesion image then applying image processing techniques, it analyses conclude about the presence carcinoma.
- In Lesson to Image analysis tools checks in the varied Melanoma parameters, Colour, Area perimeter, diameter etc texture, size and shape analysis for image segmentation and the feature stages.
- The extracted to feature parameters wont of classify the image as Non-Melanoma and Melanoma cancer lesion. Through poll we are getting to collect patient after treatment.

CHAPTER – 3

METHODOLOGY

DATA COLLECTION

- Dataset used for this are extracted from Kaggle towards skin cancer Detection.
- It consists of 10000 images of skin cancer.
- The training data consists of 8000 images and testing data consists of 2000 images

IMAGE PREPROCESSING

Image preprocessing is done by using OPEN CV and NUMPY.

OpenCV

- OpenCV-Python library of Python bindings in designed unravel computer vision problems.
- OpenCV-Python makes use Num py, by which may highly optimized library numerical operations a MATLAB-style syntax.
- All tin Open CV array are structures converted a and from Num py arrays.
- This also makes it easier to integrate other a library is that use Num py SciPy and Matplotlib.
- OpenCV to be capable image analysis and processing.

NumPy

Import- NumPy: as np

- NumPy, that stands Numerical Python, be a library consisting of multidimensional as array objects and set a routine for processing those arrays.
- Using as Num Py, mathematical and logical on operations are arrays in often performed.
- The array object in NumPy is named Nd array, it provides tons of supporting functions that make working with Nd array very easy.
- NumPy is an open-source numerical Python library. Num Py an extension of Numeric and Num array.
- Num py contains random number generators. NumPy may wrapper around library implemented in C.
- Pandas is objects reply heavily NumPy objects. Essentially, Pandas extends NumPy.

IMAGE SEGMENTATION & FEATURE EXTRACTION:

Image segmentation is a process of dividing image into regions or categories. In the thermoscopic images two types of fabric things first normal skin and second is lesion area so here we have done segmentation with Otsu thresholding technique. Using Texture-Based segmentation extracting the features from the image. GLCM (Gray Level Co-occurrence Matrix) is the statistical method examining the spatial relationship between the pixel. This technique works by creating the co-occurrence matrix were to calculate the frequency of occurrence of a pixel with the grey-level value is adjacent to a pixel with grey-level value j in any given direction and selected separating distance The GLCM matrix gives four statistics Correlation, Contrast, Energy, Homogeneity.

There some problem in segmentation of thermoscopic images due to the contrast of images like under segmentation and over-segmentation so we are concentrating on segmentation based on texture features.

IMAGE CLASSIFICATION

Deep learning is one of the best techniques for image classification. Based on the texture features we are training the dataset for classification. Here first we are giving Extracted feature to the Neural network for checking performance of image classification then we are using CNN (Convolutional Neural Network) it is one of the deep learning techniques for classification, Dermoscopic images classification is done in 7 classes .Melanocytic nevi', 'Melanoma', 'Benignkeratosis', 'Basal cell carcinoma', 'Actinic keratoses', 'Vascular lesions', ' Dermatofibroma ' it is done by using automated extracted features by CNN images. In this step, we are passing Preprocess Images to the CNN classification.

CHAPTER - 4

SYSTEM REQUIREMENTS & LIBRARIES

SOFTWARE REQUIREMENTS

Operating System: Windows, Mac OS, Linux

LIBRARIES USED

In Skin Cancer Detection we use some libraries in python. The list of libraries is: -

- TensorFlow
- Pandas
- NumPy
- OpenCV
- Keras

TensorFlow

- TensorFlow is a free and open-source software library for machine learning.
- It are often across the range of tasks but features a particular specialise to training on inference of deep neural in networks.
- Tenso r flow may a symbolic math library supported data flow differentiable with programming.
- GPU/CPU computing where an equivalent code are often executed to do both architectures .
- High on scalability computation across in machines and large data sets import tensorflow

```
import tensorflow as tf.
```

Pandas

- Pandas is a popular Python a library on data analysis.
- It directly in related Machine Learning. As all we have the dataset to be prepared before the training.
- In this case, Pandas are handy it as an developed \\to data extraction and preparation.
- It provides the implementation is high-level find data structures a wide variety tools for data analysis.
- It provides many methods for as groping, combining and filtering data.

```
import pandas as pd
```

NumPy

- NumPy means for Numerical Python, in a library of multidimensional array an objects to collection of routines for processing those arrays.
- Using Num Py, mathematical and logical operations we in arrays can be performed.
- Num py to open-source numerical Python library.

```
import numpy as np
```

OpenCV

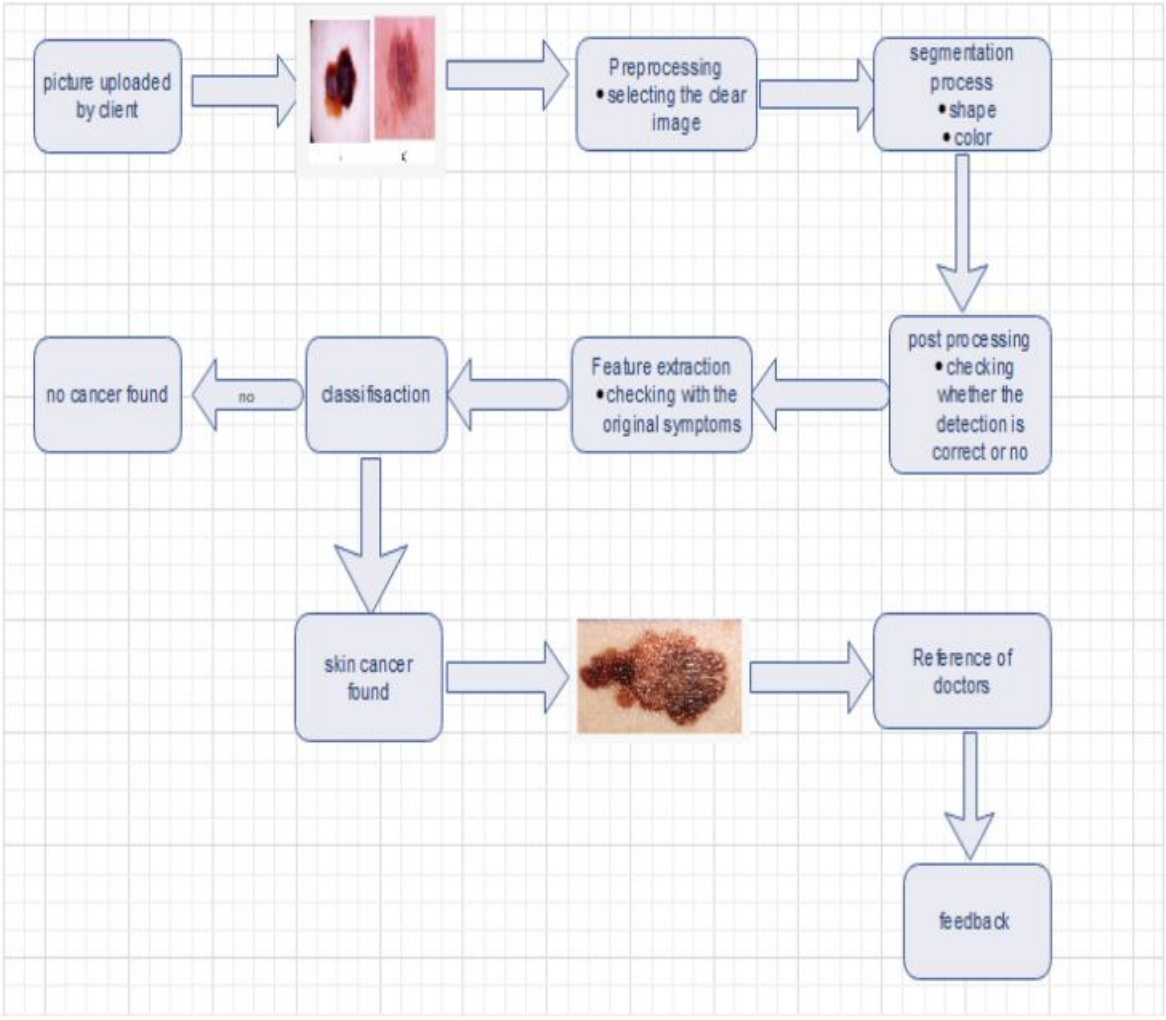
- OpenCV-Python to have library Python bindings designed and s computer vision problems.
- Open CV is capable in image analysis and processing.

Keras

- Keras is a popular Machine Learning library for Python to do.
- It high-level neural networks (API) capable in the running top on Tensor Flow, CNTK, aTheano.
- Keras makes really to ML beginners an build a design a Neural Network.
- One in best thing about Keras that allows easy to fast prototyping.

```
from tensorflow import keras
```

CHAPTER – 5



SYSTEM ARCHITECTURE

In this figure of system architecture diagram we have clearly explained the steps for detecting 7 types of skin cancer.

First step comes here is taking picture from the client or customer for detecting.

After this next step is preprocessing which is used to convert the picture to gray scale and reshaping is also done and the next step is segmentation process in which the shape and colour of the symptom or the patch will be identified.

Next step is post processing in which the detections done in the before steps are correct or not, after his feature extraction is done in which the

symptoms given in the picture by client is compared with the original cancer symptoms.

Next step here comes is classification in which the website gives whether it is cancer or not.

CHAPTER - 6

MODULES

We have 3 modules in Skin Cancer Detection . They are :-

- Detection
- Testing
- Reference & Feedback

DETECTION

Detection module used them detect the image of skin cancer. In this we detect images from skin cancer by using " FEED FORWARD NEURAL NETWORK ALGORITHM " .

- A feed forward neural network have biologically inspired by classification which algorithm.
- It consist of number of simple to neuron-like as processing in units, organized layers. Every unit in a layer connected with in the units in the previous layer. This is they are called feedforward neural networks.
- The feed forward neural network is the in first and simplest type of artificial neural network devised. In the network, the information in one direction—forward—from a input nodes,through the hidden to nodes and to the output nodes. There non cycles in loops in the network.
- Two basic feed-forward neural networks(FFNNs) created using TensorFlow in deep learning library in Python.
- Steps required build an simple feed-forward neural network to Tenso r Flow by explaining each step details. For before actual building an neural network, some preliminary steps recommended to discussed.

The summarized steps are as follows:

1. Reading the training data (inputs and outputs)

2. Building to connect an neural networks layers

3. Building a loss function to assess the prediction error

4. Create the training loop for training network and updating parameters

5. Applying some testing data to assess the network prediction accuracy

This module briefly introduces the core concepts employed in modern convolutional neural networks, with an emphasis on methods that have been proven to be effective for tasks such as object detection and semantic segmentation. Basic network architectures, common components and helpful tools for constructing and training networks are described.

TESTING

Testing module is used to test and predict the image of skin cancer. For testing we used “Evaluation function from keras “ .

- Evaluation a is process during development to the model check whether this model fit for given problem and corresponding data.
- Keras provides a function, evaluate which does evaluation of the model.
- There are three main arguments,

1.Test data

2.Test data label

3.verbose - true r false

Keras separate an portion of your training data to validation of dataset and evaluate that performance of your model on validation dataset to each epoch. You can do this by setting the validation split argument on the fit() function to a percentage of the size of your training dataset .

CHAPTER-7

WORKING

PHOTOS OF 7 CLASSES

PREPROCESSING : OpenCV gray scale reading

RESHAPING to 28*28

CREATING TRAINING & TESTDATASETS(algorithm:stratified)

CREATING CONVOLUTIONAL NEURAL NETWORKS & FEED FORWARD NEURAL

ACTIVATION FUNCTIONS RELU FOR TOP LAYERS SOFTMAX FOR LAST LAYER

TRAINING WITH EARLYN STOPPING AND PATIENCE WITH 5

SAVING THE MODEL

TESTING WITH EVALUATION FUNCTION FROM KERAS

Here comes the flowchart of this project skin cancer detection:

In any project first comes collecting of dataset so here we have taken dataset from the famous website known as “Kaggle” which consists of 10000 images and from that we have taken 8000 images training and the a remaining images for testing .so this project mainly have 7 class of classification for skin cancer.

PHOTOS OF 7 CLASSES

As our project classifies 7 types of skin cancers first we have to collect the sample images.so this step helps in collecting the images.

PREPROCESSING

After taking the image from the customer , the image may or mayn't have clarity so to avoid this problem and to make the program to classify the image we are using OpenCV gray scale to make it clear.

RESHAPING

Now comes the other task , that every image given by the customer will not be in the same size so to avoid this we have done reshapping all the images to 28*28 size. So this makes the program to run successfully.

CREATING TRAINING & TESTDATASETS

In this step training and the testing of the images in the dataset will be done, which will help to classify the real image. This is done using stratified shuffle split algorithm.

Stratified Shuffle Split cross-validator:

Provides train or test indices to separate data in train/test sets. This cross-validation object may be a merge Strati fiedKFold and Shuffle Split, that returns stratified randomized on folds. The folds made by preserving share of samples for each class.

CREATING CONVOLUTIONAL NEURAL NETWORKS & FEED FORWARD NEURAL

In this step creation of Convolution neural network(CNN) and Feed forward neural is done for detection of cancer in the pictures provided by the customer.

ACTIVATION FUNCTIONS

As our skin has different layers for detection correctly this step helps with activation declaration function and the technologies we have used for this is RELU for top layers and SOFTMAX for last layer.

TRAINING WITH EARLYN STOPPING AND PATIENCE WITH 5

The training is done by EARLYN stopping and patience has been done by 5.

SAVING THE MODEL

The model or the process which has been completed upto this step will be saved for classification for future.

TESTING WITH EVALUATION FUNCTION

When the client or the customer uploads his picture the evaluation function from keras will be used for testing whether the particular person have cancer or not and the gray scaling and reshaping of picture also is done.

CREATING PREPROCESSING FUNCTION FOR INPUT TO THE MODEL

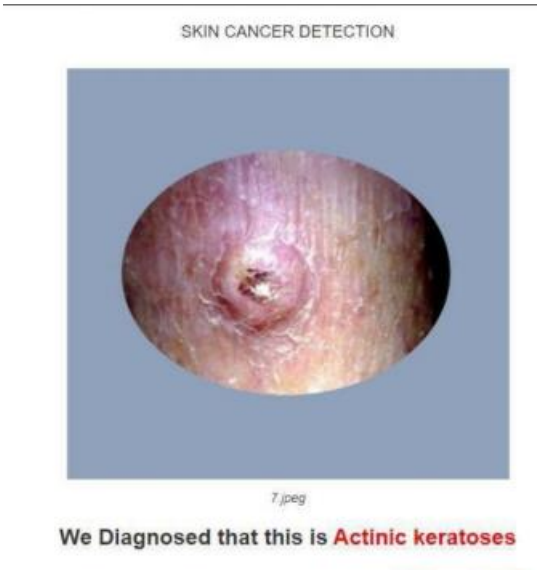
This is the last step in the whole process and in this step preprocessing function for input to the model is done. And the final output will be displayed on the webpage. So this will be the working process of this project.

CHAPTER-8

RESULT

Here is the output screenshot where we can know whether a person has cancer or not This picture is for detecting Melanoma cancer which is one of the type of skin cancer.

This picture is for detecting Melanoma cancer which is one of the type of skin cancer



This is the picture where we can find the reference of doctor

CHAPTER-9

SAMPLE CODE

[4]:
```
import tensorflow as tf
from keras.callbacks import ReduceLROnPlateau
from imblearn.over_sampling import RandomOverSampler
from keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.utils import to_categorical
from tensorflow import keras
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix , classification_report
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

[5]:
```
# from google.colab import drive
# drive.mount('/content/drive')
```

[6]:
```
FilePath = "hmnist_28_28_RGB.csv"
dataSet = pd.read_csv(FilePath)
dataSet.head()
```

[6]:

	pixel0000	pixel0001	pixel0002	pixel0003	pixel0004	pixel0005	pixel0006	pixel0007	pixel0008	pixel0009	...	pixel2343	pixel2344	pixel2345	pixel2346	pixel2347	pixel2348	pixel2349	pi
0	192	153	193	195	155	192	197	154	185	202	...	173	124	138	183	147	166	185	
1	25	14	30	68	48	75	123	93	126	158	...	60	39	55	25	14	28	25	
2	192	138	153	200	145	163	201	142	160	206	...	167	129	143	159	124	142	136	
3	38	19	30	95	59	72	143	103	119	171	...	44	26	36	25	12	17	25	
4	158	113	139	194	144	174	215	162	191	225	...	209	166	185	172	135	149	109	

5 rows × 2353 columns

label

[11]:
```
Label = np.array(Label)
Label
```

[11]:
```
array([2, 2, 2, ..., 6, 6, 6], dtype=int64)
```

[12]:
```
classes = {4: ('nv', ' melanocytic nevi'),
           6: ('mel', 'melanoma'),
           2 :('bkl', 'benign keratosis-like lesions'),
           1:('bcc' , ' basal cell carcinoma'),
           5: ('vasc', ' pyogenic granulomas and hemorrhage'),
           0: ('akiec', 'Actinic keratoses and intraepithelial carcinomae'),
           3: ('df', 'dermatofibroma')}
```

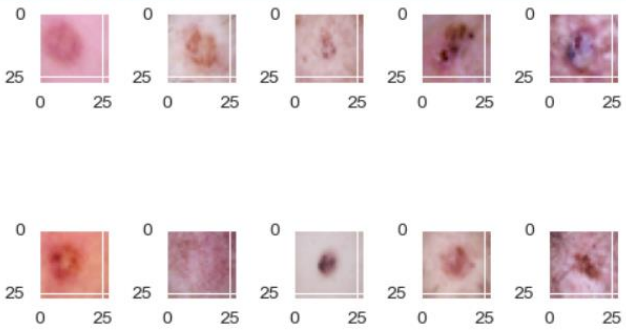
[13]:
```
#SPLIT DATA INTO TRAIN AND TEST DATA
X_train , X_test , y_train , y_test = train_test_split(Data , Label , test_size = 0.25 , random_state = 49)
```

[14]:
```
 print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(35201, 28, 28, 3)
(11734, 28, 28, 3)
(35201,)
(11734,)
```

[42]:
```
#plotting images
f , ax = plt.subplots(2,5)
f.set_size_inches(5, 5)
k = 0
for i in range(2):
    for j in range(5):
        ax[i,j].imshow(X_train[k].reshape(28,28,3))
        k = k + 1
    plt.tight_layout()
```

```
C:\Users\SMITA\AppData\Local\Temp\ipykernel_14960\670107465.py:9: UserWarning: The figure layout has changed to tight
  plt.tight_layout()
```



[16]:
```
#one hot encoding
y_train = to_categorical(y_train)

y_test = to_categorical(y_test)
```

[17]:
```
print(y_train)
```

```
[[0. 0. 0. ... 0. 0. 0.]
 [1. 0. 0. ... 0. 0. 0.]
 [0. 1. 0. ... 0. 0. 0.]
 ...
 [1. 0. 0. ... 0. 0. 0.]
 [0. 0. 0. ... 0. 0. 0.]
 [0. 0. 1. ... 0. 0. 0.]]
```

[18]:
```
#Data Augmentation
datagen = ImageDataGenerator(rescale=(1./255)
                             ,rotation_range=10
                             ,zoom_range = 0.1
                             ,width_shift_range=0.1
                             ,height_shift_range=0.1)
testgen = ImageDataGenerator(rescale=(1./255))
```

[19]:
```
learning_rate_reduction = ReduceLROnPlateau(monitor='val_accuracy'
                                            , patience = 2
                                            , verbose=1
                                            ,factor=0.5
                                            , min_lr=0.00001)
```

[20]:

```
#CNN Model
def My_Model():
  input_ = keras.layers.Input(shape = [28,28,3])
  x = keras.layers.Conv2D(32 , (3,3) , activation='relu',padding='same' , kernel_initializer='he_normal')(input_)
  x = keras.layers.MaxPooling2D()(x)
  x = keras.layers.BatchNormalization()(x)
  x = keras.layers.Conv2D(64 , (3,3) , activation='relu',padding='same' , kernel_initializer='he_normal')(x)
  x = keras.layers.Conv2D(64 , (3,3) , activation='relu',padding='same' , kernel_initializer='he_normal')(x)
  x = keras.layers.MaxPooling2D()(x)
  x = keras.layers.BatchNormalization()(x)
  x = keras.layers.Conv2D(128 , (3,3) , activation='relu',padding='same' , kernel_initializer='he_normal')(x)
  x = keras.layers.Conv2D(128 , (3,3) , activation='relu',padding='same' , kernel_initializer='he_normal')(x)
  x = keras.layers.MaxPooling2D()(x)
  x = keras.layers.BatchNormalization()(x)
  x = keras.layers.Conv2D(256 , (3,3) , activation='relu' ,padding='same', kernel_initializer='he_normal')(x)
  x = keras.layers.Conv2D(256 , (3,3) , activation='relu' ,padding='same', kernel_initializer='he_normal')(x)
  x = keras.layers.MaxPooling2D()(x)
  flatten  = keras.layers.Flatten()(x)
  classifier = keras.layers.Dropout(rate = 0.2)(flatten)
  classifier = keras.layers.Dense(units = 256 , activation = 'relu' , kernel_initializer = 'he_normal')(classifier)
  classifier = keras.layers.BatchNormalization()(classifier)
  classifier = keras.layers.Dense(units = 128 , activation = 'relu' , kernel_initializer = 'he_normal')(classifier)
  classifier = keras.layers.BatchNormalization()(classifier)
  classifier = keras.layers.Dense(units = 64 , activation = 'relu' , kernel_initializer = 'he_normal')(classifier)
  classifier = keras.layers.BatchNormalization()(classifier)
  classifier = keras.layers.Dense(units = 32 , activation = 'relu' , kernel_initializer = 'he_normal' , kernel_regularizer=keras.regularizers.L1L
  classifier = keras.layers.BatchNormalization()(classifier)
  classifier = keras.layers.Dense(units = 7 , activation='softmax' ,kernel_initializer="glorot_uniform" , name = 'classifier')(classifier)

  return keras.models.Model(inputs = input_ ,outputs = classifier)
```

model = My_Model()

model.summary()

Model: "model"

_____

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 28, 28, 3)]	0
conv2d (Conv2D)	(None, 28, 28, 32)	896
max_pooling2d (MaxPooling2D)	(None, 14, 14, 32)	0
batch_normalization (BatchNormalization)	(None, 14, 14, 32)	128
conv2d_1 (Conv2D)	(None, 14, 14, 64)	18496
conv2d_2 (Conv2D)	(None, 14, 14, 64)	36928
max_pooling2d_1 (MaxPooling2D)	(None, 7, 7, 64)	0
batch_normalization_1 (BatchNormalization)	(None, 7, 7, 64)	256
conv2d_3 (Conv2D)	(None, 7, 7, 128)	73856
conv2d_4 (Conv2D)	(None, 7, 7, 128)	147584
max_pooling2d_2 (MaxPooling2D)	(None, 3, 3, 128)	0

batch_normalization_2 (Bat (None, 3, 3, 128) 512
chNormalization)

conv2d_5 (Conv2D) (None, 3, 3, 256) 295168

conv2d_6 (Conv2D) (None, 3, 3, 256) 590080

max_pooling2d_3 (MaxPoolin (None, 1, 1, 256) 0
g2D)

flatten (Flatten) (None, 256) 0

dropout (Dropout) (None, 256) 0

dense (Dense) (None, 256) 65792

batch_normalization_3 (Bat (None, 256) 1024
chNormalization)

dense_1 (Dense) (None, 128) 32896

batch_normalization_4 (Bat (None, 128) 512
chNormalization)

dense_2 (Dense) (None, 64) 8256

batch_normalization_5 (Bat (None, 64) 256
chNormalization)

dense_3 (Dense) (None, 32) 2080

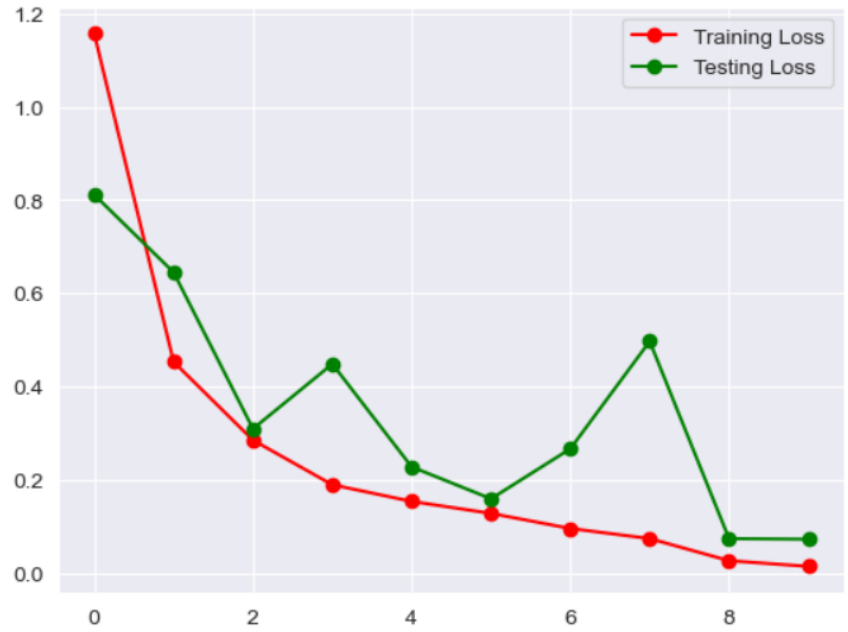
batch_normalization_6 (Bat (None, 32) 128
chNormalization)

classifier (Dense) (None, 7) 231

===
Total params: 1275079 (4.86 MB)
Trainable params: 1273671 (4.86 MB)
Non-trainable params: 1408 (5.50 KB)

[28]:
```
#training acc vs testing acc graph
plt.plot(history.history["accuracy"] , 'ro-' , label = "Training Accuracy")
plt.plot(history.history["val_accuracy"] , 'go-' , label = "Testing Accuracy")
plt.legend()
plt.show()
```

[29]:
```
#training loss vs testing loss graph
plt.plot(history.history["loss"] , 'ro-' , label = "Training Loss")
plt.plot(history.history["val_loss"] , 'go-' , label = "Testing Loss")
plt.legend()
plt.show()
```



[30]:
```
#predicting
y_pred  = model.predict(X_test).round()
```

```
367/367 [==============================] - 5s 13ms/step
```

[31]:
```
target_names = [f"{classes[i]}" for i in range(7)]
print(classification_report(y_test , y_pred , target_names =target_names ))
```

```
                                                    precision    recall  f1-score   support

('akiec', 'Actinic keratoses and intraepithelial carcinomae')       1.00      1.00      1.00      1667
                          ('bcc', ' basal cell carcinoma')          0.98      1.00      0.99      1689
                  ('bkl', 'benign keratosis-like lesions')          0.97      0.97      0.97      1651
                              ('df', 'dermatofibroma')              1.00      1.00      1.00      1629
                          ('nv', ' melanocytic nevi')               0.94      0.91      0.93      1663
            ('vasc', ' pyogenic granulomas and hemorrhage')         1.00      1.00      1.00      1680
                                  ('mel', 'melanoma')               0.97      0.97      0.97      1755

                                          micro avg                 0.98      0.98      0.98     11734
                                          macro avg                 0.98      0.98      0.98     11734
                                       weighted avg                 0.98      0.98      0.98     11734
                                        samples avg                 0.98      0.98      0.98     11734
```

Open CV

[1]:
```
import cv2
```

[5]:
```
img1 = cv2.imread("skin cencer\HAM10000_images_part_1\ISIC_0024308.jpg")
```

[6]:
```
print(img1.shape)
```

```
(450, 600, 3)
```

[7]:
```
cv2.imshow("skin",img1)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

CONCLUSION

In the proposed system, Image Pre-Processing, Image Segmentation and Image Classification steps are performed for categorizing skin lesion images into melanoma or benign. Data augmentation technique is used in Convolutional Neural Network for increasing the number of images which leads to better performance of proposed method. Experimental results show an accuracy of CNN algorithm developed with data augmentation is higher than the CNN algorithm created without data augmentation. The proposed method detects melanoma faster than the biopsy method. The proposed method can be extended to identify different types of skin related diseases. In this project

we also designed for the reference of doctors and a feedback form which is used to know the experience of the patients.

REFERENCES

1.Skin cancer dataset,

Skin Cancer MNIST: HAM10000 (kaggle.com)

2. https://uwaterloo.ca/vision-image-processing-lab/research-demos/skin-cancer-detection

3.Skin cancer statistics,

https://www.wcrf.org/cancer-trends/skin-cancer-statistics/

4. Geoffrey E. Hinton Alex Krizhevsky, Ilya Sutskever. 2012. ImageNet Classification with

Deep Convolutional Neural Networks. Neural Information Processing Systems (2012).

5. Xino Yao. 1999. Evolving artificial neural networks. Proc. IEEE 87, 9 (1999), 1423—1447

6. mirreza Mahbod, Gerald Schaefer, Chunliang Wang, Rupert Ecker, Isabella Ellinger, "Skin

Lesion Classification Using Hybrid Deep Neural Networks", IEEE, International

Conference on Acoustics, Speech and Signal Processing (ICASSP), pp.1229-1233,2019.

7. Balazs Harangi, Agnes Baran , Andras Hajdu, "Classification of Skin Lesions Using An

Ensemble of Deep Neural Networks", IEEE, 40th Annual International Conference of the

IEEE Engineering in Medicine and Biology Society (EMBC), pp.2575-2578, 2018.

8. Signs and Symptoms of Skin Cancer,

https://www.cancer.org/cancer/melanoma-skin-cancer/detection- diagnosis-staging/signs-

and-symptoms.html, accessed date: Mar 30,2020.

9. Tests for Melanoma Skin Cancer, https://www.cancer.org/cancer/melanoma- skin-

cancer/detection- diagnosis-staging/how-diagnosed.html, accessed date: Mar 30, 2020.