VIT
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

CSE 4099 - CAPSTONE PROJECT
AN INTELLIGENT SYSTEM FOR HAIR IMAGE SCALP DISEASE DETECTION AND DIAGNOSIS

FINAL VIVA-VOCE
NAME: VARSHA R
REGISTRATION NUMBER: 18BCE1344
FACULTY: DR. ASHA S

S.NO	OUTLINE
1	INTRODUCTION
2	PROBLEM STATEMENT
3	RESEARCH OBJECTIVES
4	PROPOSED SYSTEM
5	MODULES
6	RESULTS AND DISCUSSION
7	INFERENCE
8	FUTURE WORK
9	FACULTY APPROVAL SCREENSHOT
10	REFERENCES

INTRODUCTION

- In today's world, when human interaction is limited and early disease detection is critical, it is of utmost importance to invest in and create efficient and effective techniques of diagnosis.
- With people being reluctant or even incapable to visit clinics and specialists, they can't identify any ailments they have until it is past the point of no return.
- This project hence means to deliver an answer where individuals can without much hesitation be analyzed for normal infections by just gaining an image of their scalp, a little body part equipped for giving us profound insights, using their smartphone.
- This projects aims to help with analyzing a picture of the scalp and reach inferences, to help the expert in system without human intercession.
- The human scalp is equipped for demonstrating indications of different ailments which are reflected by its surface covering, surface, shading and mathematical highlights.
- In this project, we propose an electronic scalp examination technique dependent on quantitative and subjective highlights like region, baldness, width, shading and surface of the scalp.

PROBLEM STATEMENT

Hair care can be realised in a professional hair care shop or medical and cosmetic clinic, but it is very expensive. Recently, owing to the enhancement in computation ability of intelligent devices and decrease in prices, it is feasible to have an inexpensive hair and scalp analysis system.The image processing technique applied to the human scalp, can also be used suitably by veterinarians to determine a solution for the hair loss in pets.

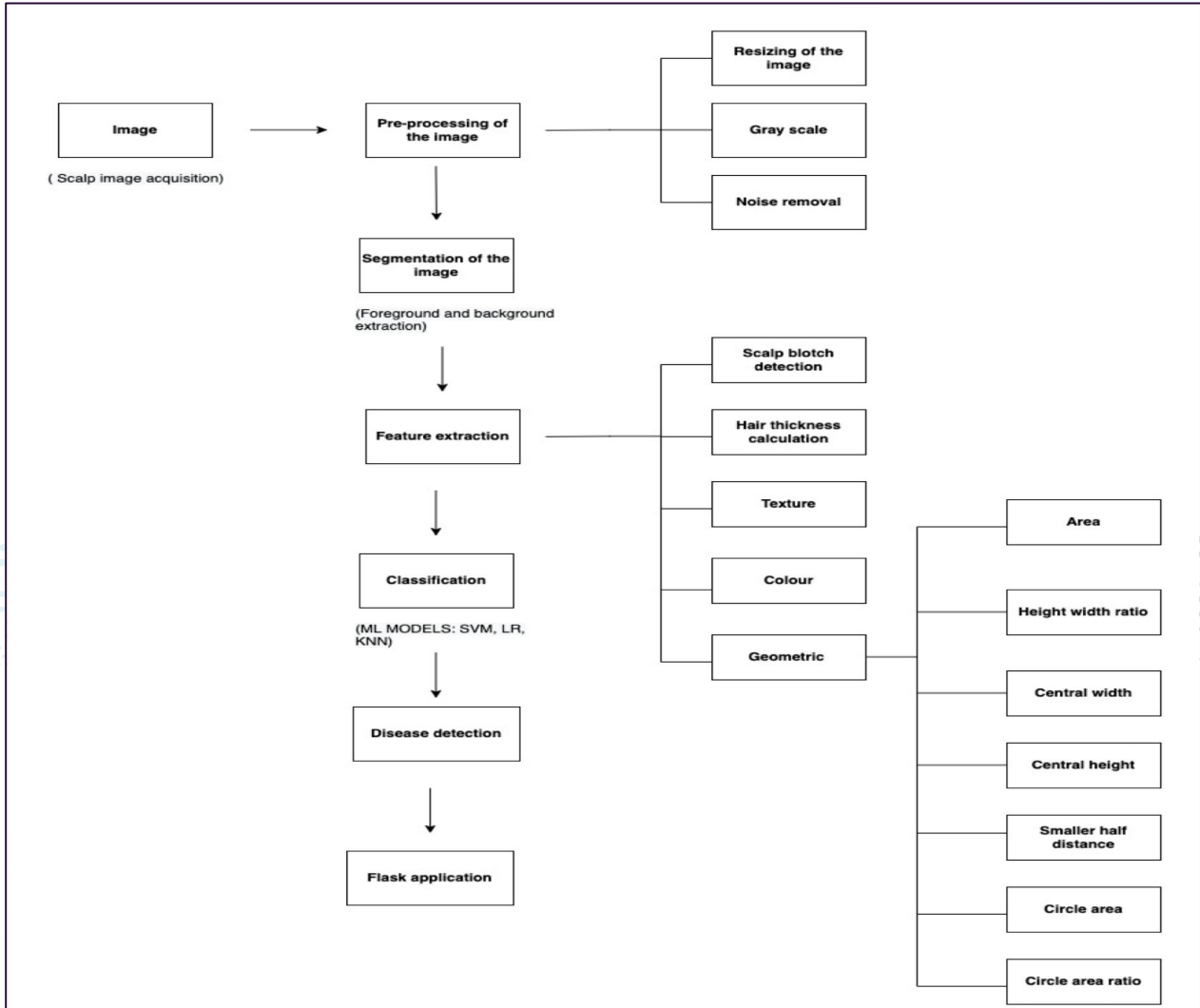
RESEARCH OBJECTIVES

- This research offers an intelligent scalp inspection and diagnostic method for hair scalp health that is based on Machine learning techniques.
- The suggested system can detect the condition of the user's scalp automatically.
- Furthermore, by using machine learning algorithms, we may continually increase the amount of samples to improve accuracy.
- The features include hair thickness, hair density and scalp blotch. We show the effectiveness of our project by extensive experiments on the prototype system.
- As a result, we can get the quantitative data on the scalp, including bacteria, allergies, dandruff, grease, and hair loss.
- With the help of proposed model, the clinical experts will be able to get a second opinion that will help them take proper decisions for diagnosing the presence of this disease in patients.

PROPOSED SYSTEM

This project attempts to detect diseases such as Tinea Captisis, Alopecia Areata and Melanoma using well known Image Processing techniques and Machine Learning algorithms. The images are passed through a pipeline in which they are first pre-processed and segmented after which various features are extracted. The extracted features are used to build classification models and detect the diseases in scope of this support vector classifier, KNN and logistic regression. A comparative study of the models is performed. Once, that is performed, and the best Machine Learning model with maximum accuracy is identified, the intention is to build a website or an application so that once a new image is uploaded by the user, the disease is identified and we can also provide them additional resources about the next steps based on the diseases. This indeed helps users identify and resolve the issue with just a scalp image, making it user friendly.

PROPOSED SYSTEM ARCHITECTURE



MODULES

LITERATURE SURVEY

1

To find the existing work and their drawbacks and the results they have produced to get an insight into the work that has been performed

IMAGE ACQUISITION

2

To build a database from extracting images from various databases to improve the accuracy of the models.A total of 170 images were collected of which 47 were Tinea Capitis, 46 were Alopecia Areata, 48 were

and the rest

healthy scalp

IMAGE PREPROCESSING

3

A grayscale conversion is done prior to appropriate feature being extracted.

Images are resized to 560x560 for computational uniformity and model

convergence.

NOISE REDUCTION

4

To mitigate the technical deficiencies of a recording as well as to separate the desired physiological process from interfering processes, a five by five kernel is used as a 2D filter for noise removal.

MODULES

IMAGE SEGMENTATION

5

To segment the images foreground-background extraction using the GrabCut algorithm..The GrabCut algorithm is chosen over other methods due to the non-uniformity of the background. This algorithm performs well in such a condition and the reason lies in the uncertainty principle where positions and frequencies of the signal cannot be determined at th same time.

FEATURE EXTRACTION

6

They include scalp botch (balding) using skeletonisation, hair count using hough transform algorithm, texture to identify coating, smoothness, lines and small pores, chromatic and geometric features including area, height width ratio, central width, central height, smaller half distance, circle area and circle area ratio

MACHINE LEARNING MODELS

7

The features extracted from the images are used to build three different classification models -Support Vector Machine, KNN and Logistic Regression.

FEATURE EXTRACTION

01

COLOUR FEATURE

Extraction of the color feature was performed using the Manhattan Distance, i.e by calculating the RGB values of the pixels and the base color values. The color of an image is therefore identified as the reference color to which a majority of the foreground pixels have a minimum distance.

Formula : |x2-x1+ |y2-y1|

02

HAIR COUNT

For hair count, we made use of line detection algorithm. For this, I chose hough transform over convolution based technique,the advantage is that we can detect the shape even if it is broken or distorted a little. It is done so by creating a 2D array and returns the an array of (x,y) where x is measured in pixels and y is measured in radians.

03

SKELETONISATION

To detect the balding of the hair, we make use of skeletonisation.Skeletonization is a process for reducing foreground regions in a binary image to a skeletal remnant that largely preserves the extent and connectivity of the original region while throwing away most of the original foreground pixels.

GEOMETRIC FEATURE EXTRACTION

04

AREA

Area is calculated by measuring the number of coloured pixels.

05

WIDTH RATIO

It is the ratio of the height of the scalp (h) and the width of the scalp (w)

06

CENTRAL WIDTH

The width is measured as the horizontal distance from the central pixel to the farthest point on the x axis.

GEOMETRIC FEATURE EXTRACTION

07

CENTRAL HEIGHT

The height is measured as the vertical distance from the central pixel to the farthest point on the y axis.

08

SMALLER HALF DISTANCE

Smaller half distance is the half distance of either the height or the width, the condition for it to be chosen is depending on which of them is shorter.

9

CIRCLE AREA

It is defined as the area of the circle present on the scalp using the formula of the smaller half distance.

10

CIRCLE AREA RATIO

It is the ratio between the circle area and the area of the scalp

TEXTURAL FEATURE EXTRACTION

11

NUMBER OF CONTOURS

The number of contours along with the area of contours and the per contour area was achieved by converting the gray scale image by using a Gaussian blur filter and by applying an adaptive threshold on the blurred images. Next, the contours were calculated using the thresholded images and a condition was given to filter the images that had the textural images.

12

AREA OF THE CONTOUR

Once we get the number of contours, we check if the number of contours is less than 1000, and append it to the area

13

LENGTH OF THE CONTOUR

Once we get the area of the contour, we calculate the length of the contour by using cv2.arclength() and append it by adding the total to the length of the contour.

RESULTS & DISCUSSION

Melanoma :

Alopecia Areata:

Tinea Captisis

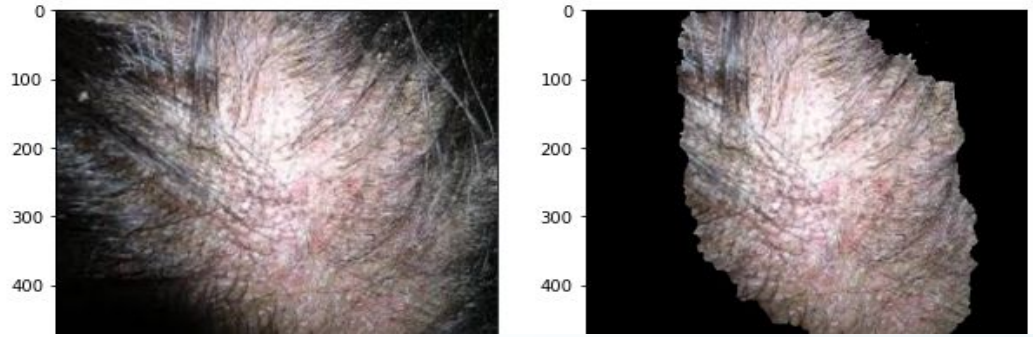
Healthy scalp:

SEGMENTATION OF IMAGES

BEFORE SEGMENTATION

AFTER SEGMENTATION

In [74]:
```
#segmentation
img=data[46]
mask = np.zeros(img.shape[:2],np.uint8)

bgdModel = np.zeros((1,65),np.float64)
fgdModel = np.zeros((1,65),np.float64)

rect = (1,1,545,545) #rect = (start_x, start_y, width, height)

cv2.grabCut(img,mask,rect,bgdModel,fgdModel,5,cv2.GC_INIT_WITH_RECT)
mask2 = np.where((mask==2)|(mask==0),0,1).astype('uint8')
fimg = img*mask2[:,:,np.newaxis]


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

ax1 = fig.add_subplot(121)
ax1.imshow(img)

ax2 = fig.add_subplot(122)
ax2.imshow(fimg)

#plt.colorbar()
plt.show()

fimg=cv2.cvtColor(fimg,cv2.COLOR_BGR2RGB)
cv2.imwrite('/Users/varsha/Desktop/Capstone project/segmented_1/s46.jpg', fimg)
```



COLOUR FEATURE CODE

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import cv2
from skimage.color import rgb2gray
from skimage.io import imread, imshow
from scipy import ndimage
from PIL import Image,ImageChops
#from colours import color_analysis1
import math
import os
import glob
import random


from collections import Counter
import webcolors
import cv2
from PIL import Image

df=pd.read_csv('/Users/varsha/Desktop/Capstone project/resp2.csv')
print(df.head())
list(df.columns)
df.shape

# COLOURS = {   "red": (255, 0, 0),
#               "green" : (0,255,0),
#               "blue":(0,0,255),
#               "pink":(255,192,203),
#               "white":(255,250,250),
#               "yellow":(255,255,0),
#               "purple":(221,160,221)
#               }

COLOURS = { "yellow": (51, 42, 31),
            "white":(65,50,90),
            "peach":(50,65,146),
            "black": (80,90,100)
          }
```

```
def classify(rgb_tuple):

    manhattan = lambda x,y : abs(x[0] - y[0]) + abs(x[1] - y[1]) + abs(x[2] - y[2])
    distances = {k: manhattan(v, rgb_tuple) for k, v in COLOURS.items()}
    color = min(distances, key=distances.get)
    return color


def color_analysis(image):
    color_counter = Counter({color: 0 for color in Counter(COLOURS)})
    for pixel_count, RGB in image.getcolors(image.width * image.height):
        if(RGB!=(0,0,0)):
            color_name = classify(RGB)
            color_counter[color_name] += pixel_count


    for color in color_counter:
        pixel_count = image.width * image.height
        color_counter[color] = color_counter[color] / pixel_count


    colour = max(color_counter, key=color_counter.get)

    return colour

img_dir = "/Users/varsha/Desktop/Capstone project/segmented_2"
data_path = os.path.join(img_dir,'*g')
files = glob.glob(data_path)
```

COLOUR FEATURE CODE AND RESULT

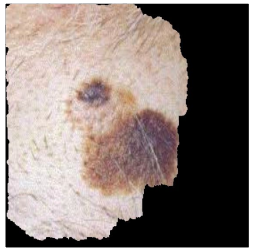
```
img_dir = "/Users/varsha/Desktop/Capstone project/segmented_2"
data_path = os.path.join(img_dir,'*g')
files = glob.glob(data_path)

colour=[]
for i in range(47,94):
    for f1 in files:
        tmp=f1.split("/")
        a=tmp[-1].split("s")
        b=a[1].split(".")
        n=b[0]
        if(i==int(n)):
            img = Image.open(f1).convert('RGB')
            col=color_analysis(img)
            colour.append(col)

df['colour']=colour
print(df.head())

with pd.option_context('display.max_rows', None, 'display.max_columns', None):  # more options can be specified also
    print(df)

df.colour.unique()

df.to_csv (r'/Users/varsha/Desktop/Capstone project/2.csv', index = False, header=True)
```

	colour
0	peach
1	yellow
2	peach
3	peach
4	red
5	pink
6	peach
7	peach
8	yellow
9	peach
10	peach
11	yellow
12	peach
13	pink
14	peach
15	pink
16	peach
17	yellow

COLOUR FEATURE RESULT

BLACK (150,80,121)

RGB for this image:
(141.7101849489796,
84.47229591836735,
101.7430006377551)

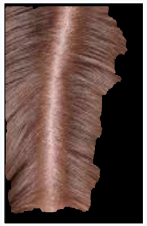


WHITE (65,50,90)

RGB for this image:
(58.226316964285715,
49.748376913265304,
44.12170599489796)

PEACH (50,65,146)

RGB for this image :
(61.276983418367344,
59.127589285714286,
55.75493303571429)



YELLOW (80, 90, 100)

RGB for this image:
(73.43804528061224,
68.75244579081632,
67.17905931122449)

GEOMETRIC FEATURES CODE

```python
#Area
area=[]
for img in df['segmented_img']:
    cnt=0
    img=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    img=cv2.filter2D(img, -1, kernel)
    for i in range(0,img.shape[0]):
        for j in range(0,img.shape[0]):
            if(img[i][j]!=0):
                cnt+=1

    area.append(cnt)

df['area']=area
print(df.head())
```

```python
central_width=[]
central_height=[]
height_width_ratio=[]
mid_x=280
mid_y=280
for img in df['segmented_img']:
    img=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    #img=cv2.filter2D(img, -1, kernel)
    left=0
    right=0
    top=0
    bot=0
    for i in range(mid_x,0,-1):
        if(img[mid_x][i]!=0):
            left+=1

    for j in range(mid_x,560):
        if(img[mid_x][j]!=0):
            right+=1

    for k in range(mid_y,0,-1):
        if(img[k][mid_y]!=0):
            top+=1

    for l in range(mid_y,560):
        if(img[l][mid_y]!=0):
            bot+=1

    h=top+bot
    w=left+right
    h_w=h/w
    height_width_ratio.append(h_w)
    central_width.append(w)
    central_height.append(h)

df['height_width_ratio']=height_width_ratio
df['central_width']=central_width
df['central_height']=central_height
print(df.head())
```

GEOMETRIC FEATURES CODE

```
#Smaller-Half-distance, Circle Area, Square Area
smaller_half_dist=[]
circle_area=[]
circle_area_ratio=[]
square_area=[]
square_area_ratio=[]
small=0
circ_a=0

for a in df.index:
    small=min(df['central_width'][a],df['central_height'][a])
    shd=small/2
    circ_a=math.pi*(shd**2)
    circ_a_rat=circ_a/df['area'][a]
    square_a=4*(shd**2)
    square_a_rat=square_a/df['area'][a]
    square_area.append(square_a)
    square_area_ratio.append(square_a_rat)
    smaller_half_dist.append(shd)
    circle_area.append(circ_a)
    circle_area_ratio.append(circ_a_rat)


df['smaller_half_dist']=smaller_half_dist
df['circle_area']=circle_area
df['circle_area_ratio']=circle_area_ratio
print(df.head())
```

GEOMETRIC FEATURES RESULT

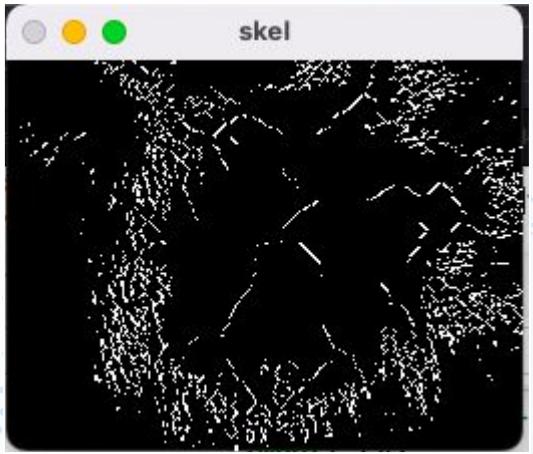
```
                                          segmented_img    area  \
0  [[[0, 0, 0], [0, 0, 0], [0, 0, 0], [0, 0, 0], ...   98741
1  [[[0, 0, 2], [0, 0, 2], [0, 0, 2], [0, 0, 2], ...  167617
2  [[[1, 0, 0], [1, 0, 0], [1, 0, 0], [1, 0, 0], ...  223022
3  [[[0, 0, 0], [0, 0, 0], [0, 0, 0], [0, 0, 0], ...  198454
4  [[[0, 0, 0], [0, 0, 0], [0, 0, 0], [0, 0, 0], ...  148689

   height_width_ratio  central_width  central_height
0            1.515284            229             347
1            1.235437            412             509
2            0.902128            470             424
3            1.153318            437             504
4            0.844920            374             316
   ID  Tinea captisis  Alopecia areata  Melanoma  Healthy scalp     Img  \
0   1               1                0         0              0  1.jpeg
1   2               1                0         0              0  2.jpeg
2   3               1                0         0              0  3.jpeg
3   4               1                0         0              0  4.jpeg
4   5               1                0         0              0  5.jpeg

                                          segmented_img    area  \
0  [[[0, 0, 0], [0, 0, 0], [0, 0, 0], [0, 0, 0], ...   98741
1  [[[0, 0, 2], [0, 0, 2], [0, 0, 2], [0, 0, 2], ...  167617
2  [[[1, 0, 0], [1, 0, 0], [1, 0, 0], [1, 0, 0], ...  223022
3  [[[0, 0, 0], [0, 0, 0], [0, 0, 0], [0, 0, 0], ...  198454
4  [[[0, 0, 0], [0, 0, 0], [0, 0, 0], [0, 0, 0], ...  148689

   height_width_ratio  central_width  central_height  smaller_half_dist  \
0            1.515284            229             347              114.5
1            1.235437            412             509              206.0
2            0.902128            470             424              212.0
3            1.153318            437             504              218.5
4            0.844920            374             316              158.0

     circle_area  circle_area_ratio
0   41187.065087           0.417122
1  133316.625848           0.795365
2  141195.740223           0.633102
3  149986.701866           0.755776
4   78426.719004           0.527455
```

SKELETONISATION

In [*]:
```
#skeletonization- just image- see if you should count lines with this
import cv2
import numpy as np


img = cv2.imread('/Users/varsha/Desktop/Capstone project/images_1/4.jpeg',0)
size = np.size(img)
skel = np.zeros(img.shape,np.uint8)

ret,img = cv2.threshold(img,127,255,0)
element = cv2.getStructuringElement(cv2.MORPH_CROSS,(3,3))
done = False

while( not done):
    eroded = cv2.erode(img,element)
    temp = cv2.dilate(eroded,element)
    temp = cv2.subtract(img,temp)
    skel = cv2.bitwise_or(skel,temp)
    img = eroded.copy()
    #thinned = cv2.ximgproc.thinning(cv2.cvtColor(img, cv2.COLOR_RGB2GRAY))

    zeros = size - cv2.countNonZero(img)
    if zeros==size:
        done = True


print(zeros)
print(skel)
cv2.imshow("skel",skel)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

```
50310
```

SKELETONISATION RESULT



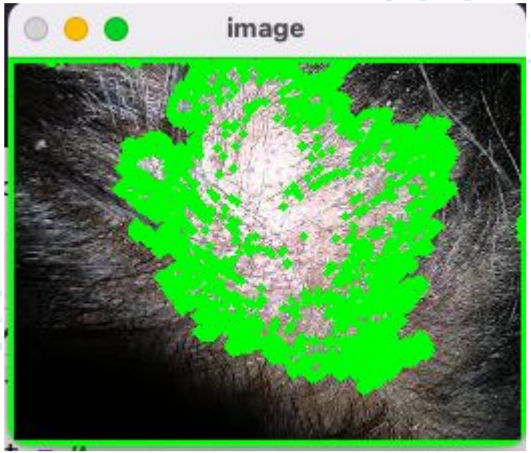
50310

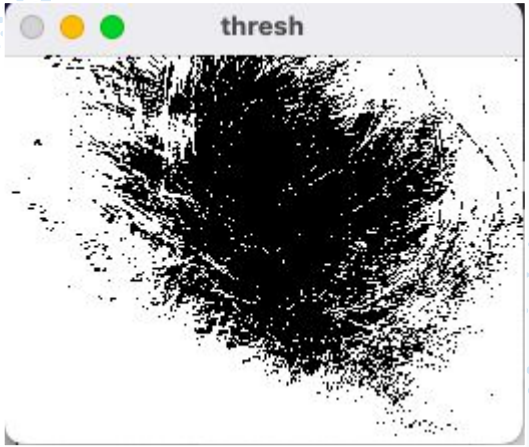
HAIR COUNT CODE

```
#hough tranform to count the number of lines-check if you should with segmented/ normal okay
import cv2

lines = []
image = cv2.imread('/Users/varsha/Desktop/Capstone project/images_1/47.jpeg')
gray = cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
thresh = cv2.threshold(gray, 120, 255, cv2.THRESH_BINARY_INV)[1]

cnts = cv2.findContours(thresh, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
cnts = cnts[0] if len(cnts) == 2 else cnts[1]

lines = 0
for c in cnts:
    cv2.drawContours(image, [c], -1, (36,255,12), 3)
    lines += 1


df['lines']=lines
print(lines)
cv2.imshow('thresh', thresh)
cv2.imshow('image', image)
cv2.waitKey()
```

```
409
```

HAIR COUNT RESULT





409

TEXTURAL FEATURE CODE

```
#Contours (Texture/Patches)
num_cont=[]
area_cont=[]
per_cont=[]
for i in range(2,91):
    for f1 in files:
        tmp=f1.split("/")
        a=tmp[-1].split("s")
        b=a[1].split(".")
        n=b[0]
        if(i==int(n)):
            img = cv2.imread(f1,1)
            gray = cv2.cvtColor(img,cv2.COLOR_RGB2GRAY)
            blur = cv2.GaussianBlur(gray, (3,3),0)
            thresh = cv2.adaptiveThreshold(blur, 255, cv2.ADAPTIVE_THRESH_GAUSSIAN_C, cv2.THRESH_BINARY_INV, 205, 1)
            contours, _ = cv2.findContours(thresh, cv2.RETR_LIST, cv2.CHAIN_APPROX_SIMPLE)
            filtered = []
            tot_are=0
            tot_per=0
            for c in contours:
                if cv2.contourArea(c) < 1000:
                    continue
                filtered.append(c)

            for c in filtered:
                area = cv2.contourArea(c)
                p = cv2.arcLength(c,True)
                tot_are+=area
                tot_per+=p


            num_cont.append(len(filtered))
            area_cont.append(tot_are)
            per_cont.append(tot_per)


df['num_contours']=num_cont
df['area_contours']=area_cont
df['len_contours']=per_cont
print(df.head())
```

TEXTURAL FEATURE RESULT

	ID	Tinea captisis	Alopecia areata	Melanoma	Healthy scalp	Img	pic	num_contours	area_contours	len_contours
0	1	1	0	0	0	1.jpeg	[[[60, 67, 75], [60, 66, 75], [58, 65, 73], [5...	3	104954.5	5440.028013
1	2	1	0	0	0	2.jpeg	[[[92, 81, 77], [84, 73, 69], [66, 55, 51], [5...	6	376831.5	9532.359303
2	3	1	0	0	0	3.jpeg	[[[6, 58, 98], [6, 58, 98], [17, 69, 108], [28...	10	342755.5	17617.359080
3	4	1	0	0	0	4.jpeg	[[[4, 0, 0], [5, 1, 0], [7, 3, 0], [9, 5, 0], ...	10	485682.5	12543.058113
4	5	1	0	0	0	5.jpeg	[[[12, 4, 1], [12, 4, 1], [13, 5, 2], [13, 5, ...	6	326303.0	11330.159495

MACHINE LEARNING MODELS

```
In [59]: import pandas as pd
         import numpy as np
         from sklearn.model_selection import train_test_split
         from sklearn.metrics import f1_score, mean_squared_error, accuracy_score
```

```
In [60]: df=pd.read_csv('/Users/varsha/Desktop/orig2.csv')
         df.head()
```

Out[60]:

	ID	Tinea captisis	Alopecia areata	Melanoma	Healthy scalp	Img	pic	segmented_img	colour	num_contours	...	height_width_ratio	central_width	centra
0	1	1	0	0	0	1.jpeg	[[[60 67 75]\n [60 66 75]\n [58 65 ...	[[[0 0 0]\n [0 0 0]\n [0 0 0]\n ...\n [0 0...	(48.514639668367344, 45.396026785714284, 44.45...	3	...	1.515284	229	
1	2	1	0	0	0	2.jpeg	[[[92 81 77]\n [84 73 69]\n [66 55 ...	[[[0 0 2]\n [0 0 2]\n [0 0 2]\n ...\n [0 0...	(51.35983099489796, 42.72837691326531, 35.8404...	6	...	1.235437	412	
2	3	1	0	0	0	3.jpeg	[[[6 58 98]\n [6 58 98]\n [17 69 1...	[[[1 0 0]\n [1 0 0]\n [1 0 0]\n ...\n [0 0...	(36.319072066326534, 28.56017538265306, 30.999...	10	...	0.902128	470	

MACHINE LEARNING MODELS

In [123]:
```
#SVM
from sklearn.svm import SVC
svmclf = SVC()
svmclf.fit(X_train, y_train)

# Storing the predictions of the non-linear model
y_pred = svmclf.predict(X_test)


from sklearn.metrics import r2_score
# Evaluating the performance of the non-linear model
print('Accuracy for healthy scalp : ' +str(accuracy_score(y_test, y_pred)))
from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test,y_pred))
print(classification_report(y_test,y_pred))
```

In [174]:
```
#KNN
from sklearn.neighbors import KNeighborsClassifier
neigh = KNeighborsClassifier(n_neighbors=3)
neigh.fit(X_train, y_train)
y_pred=neigh.predict(X_test)
print('Accuracy for melanoma: ' +str(accuracy_score(y_test, y_pred)))
from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test,y_pred))
print(classification_report(y_test,y_pred))
```

MACHINE LEARNING MODELS

In [62]:
```
df['colour1'] = df['colour1'].map( {'yellow':1, 'pink':2, 'white':3,'peach':4,'red':5,'black':6} )
```

In [63]:
```
X=df[['area','central_height','central_width','height_width_ratio','smaller_half_dist',
 'circle_area','circle_area_ratio','num_contours','area_contours','len_contours','zeroes ','colour1']]
```

In [64]:
```
y = df['Tinea captisis'].to_numpy()
```

In [65]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4, shuffle=True)
```

In [66]:
```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
'''df['area']=sc.transform(df['area'])
df['height']=sc.transform(df['height'])
df['width']=sc.transform(df['width'])
df['height_width_ratio']=sc.transform(df['height_width_ratio'])'''

X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

In [67]:
```
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(solver='lbfgs')
classifier.fit(X_train, y_train)
```

Out[67]:
```
LogisticRegression()
```

INFERENCE FOR COLOUR FEATURE

DISEASE	MEAN RED	MEAN GREEN	MEAN BLUE
TINEA CAPTISIS	66.54	56.37	51.75
ALOPECIA AREATA	66.10	54.96	50.61
MELANOMA	90.29	69.12	61.18
HEALTHY SCALP	63.67	52.96	49.22

INFERENCE FOR TEXTURAL FEATURES

DISEASE	NUMBER OF CONTOURS	AREA OF CONTOURS	LENGTH OF CONTOURS
TINEA CAPTISIS	6.32	290119.7	10845.17
ALOPECIA AREATA	4.46	253261.4	8595.91
MELANOMA	6.97	232475.3	9240.73
HEALTHY SCALP	6.67	257113.3	13052.42

INFERENCE FOR GEOMETRIC FEATURES

DISEASE	AREA	HEIGHT WIDTH RATIO	CENTRAL WIDTH	CENTRAL HEIGHT	SMALLER HALF DISTANCE	CIRCLE AREA	CIRCLE AREA RATIO
TINEA CAPTISIS	167961.4	1.0899	410.934	432.869	190.2	117723.9	0.686
ALOPECIA AREATA	158345.4	1.1987	370.217	417.847	171.6	98762.14	0.602
MELANOMA	184629	1.2065	392.270	449.791	185.6	114174.23	0.594
HEALTHY SCALP	186891.1	1.1986	408.793	468.275	191.7	118030.96	0.634

INFERENCE FOR SKELETONISATION

DISEASE	SKELETONISATION
TINEA CAPTISIS	43699.3
ALOPECIA AREATA	37759.5
MELANOMA	29527.1
HEALTHY SCALP	29542.2

INFERENCE FOR HAIR COUNT

DISEASE	HAIR COUNT
TINEA CAPTISIS	88.5
ALOPECIA AREATA	49.95
MELANOMA	119.43

LOGISTIC REGRESSION RESULTS

Accuracy for melanoma: 0.7794117647058824
[[44 7]
[8 9]]

	precision	recall	f1-score	support
0	0.85	0.86	0.85	51
1	0.56	0.53	0.55	17
accuracy			0.78	68
macro avg	0.70	0.70	0.70	68
weighted avg	0.78	0.78	0.78	68

Accuracy for Alopecia areata: 0.6764705882352942
[[41 5]
[17 5]]

	precision	recall	f1-score	support
0	0.71	0.89	0.79	46
1	0.50	0.23	0.31	22
accuracy			0.68	68
macro avg	0.60	0.56	0.55	68
weighted avg	0.64	0.68	0.63	68

Accuracy for tinea capitis: 0.7058823529411765
[[41 9]
[11 7]]

	precision	recall	f1-score	support
0	0.79	0.82	0.80	50
1	0.44	0.39	0.41	18
accuracy			0.71	68
macro avg	0.61	0.60	0.61	68
weighted avg	0.70	0.71	0.70	68

Accuracy for healthy scalp: 0.8529411764705882
[[54 3]
[7 4]]

	precision	recall	f1-score	support
0	0.89	0.95	0.92	57
1	0.57	0.36	0.44	11
accuracy			0.85	68
macro avg	0.73	0.66	0.68	68
weighted avg	0.83	0.85	0.84	68

SVM RESULTS

Accuracy for healthy scalp : 0.8823529411764706
[[57 0]
[8 3]]

	precision	recall	f1-score	support
0	0.88	1.00	0.93	57
1	1.00	0.27	0.43	11
accuracy			0.88	68
macro avg	0.94	0.64	0.68	68
weighted avg	0.90	0.88	0.85	68

Accuracy for Alopecia areata : 0.7794117647058824
[[50 3]
[12 3]]

	precision	recall	f1-score	support
0	0.81	0.94	0.87	53
1	0.50	0.20	0.29	15
accuracy			0.78	68
macro avg	0.65	0.57	0.58	68
weighted avg	0.74	0.78	0.74	68

Accuracy for tinea capitis : 0.7647058823529411
[[48 4]
[12 4]]

	precision	recall	f1-score	support
0	0.80	0.92	0.86	52
1	0.50	0.25	0.33	16
accuracy			0.76	68
macro avg	0.65	0.59	0.60	68
weighted avg	0.73	0.76	0.73	68

Accuracy for Melanoma : 0.7647058823529411
[[46 0]
[16 6]]

	precision	recall	f1-score	support
0	0.74	1.00	0.85	46
1	1.00	0.27	0.43	22
accuracy			0.76	68
macro avg	0.87	0.64	0.64	68
weighted avg	0.83	0.76	0.71	68

KNN RESULTS

Accuracy for melanoma: 0.7794117647058824
[[45 6]
[9 8]]

	precision	recall	f1-score	support
0	0.83	0.88	0.86	51
1	0.57	0.47	0.52	17
accuracy			0.78	68
macro avg	0.70	0.68	0.69	68
weighted avg	0.77	0.78	0.77	68

Accuracy for Alopecia areata: 0.6617647058823529
[[38 10]
[13 7]]

	precision	recall	f1-score	support
0	0.75	0.79	0.77	48
1	0.41	0.35	0.38	20
accuracy			0.66	68
macro avg	0.58	0.57	0.57	68
weighted avg	0.65	0.66	0.65	68

Accuracy for Tinea captisis: 0.6764705882352942
[[37 11]
[11 9]]

	precision	recall	f1-score	support
0	0.77	0.77	0.77	48
1	0.45	0.45	0.45	20
accuracy			0.68	68
macro avg	0.61	0.61	0.61	68
weighted avg	0.68	0.68	0.68	68

Accuracy for Healthy scalp: 0.8529411764705882
[[55 3]
[7 3]]

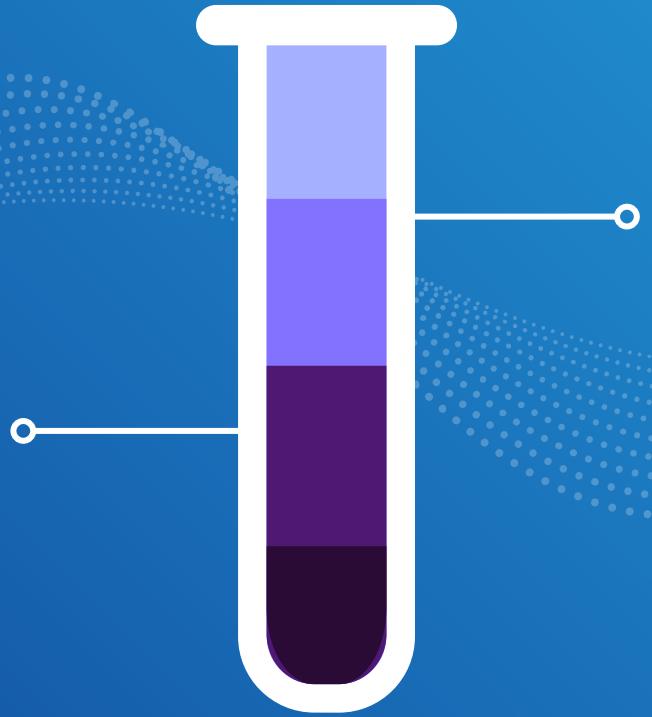
	precision	recall	f1-score	support
0	0.89	0.95	0.92	58
1	0.50	0.30	0.37	10
accuracy			0.85	68
macro avg	0.69	0.62	0.65	68
weighted avg	0.83	0.85	0.84	68

FUTURE WORK

FEATURE EXTRACTION HAS BEEN DONE

MACHINE LEARNING MODEL

KNN,SVC AND LR WILL BE PERFORMED ON THE DATASET AND A COMPARISON WILL BE PERFORMED



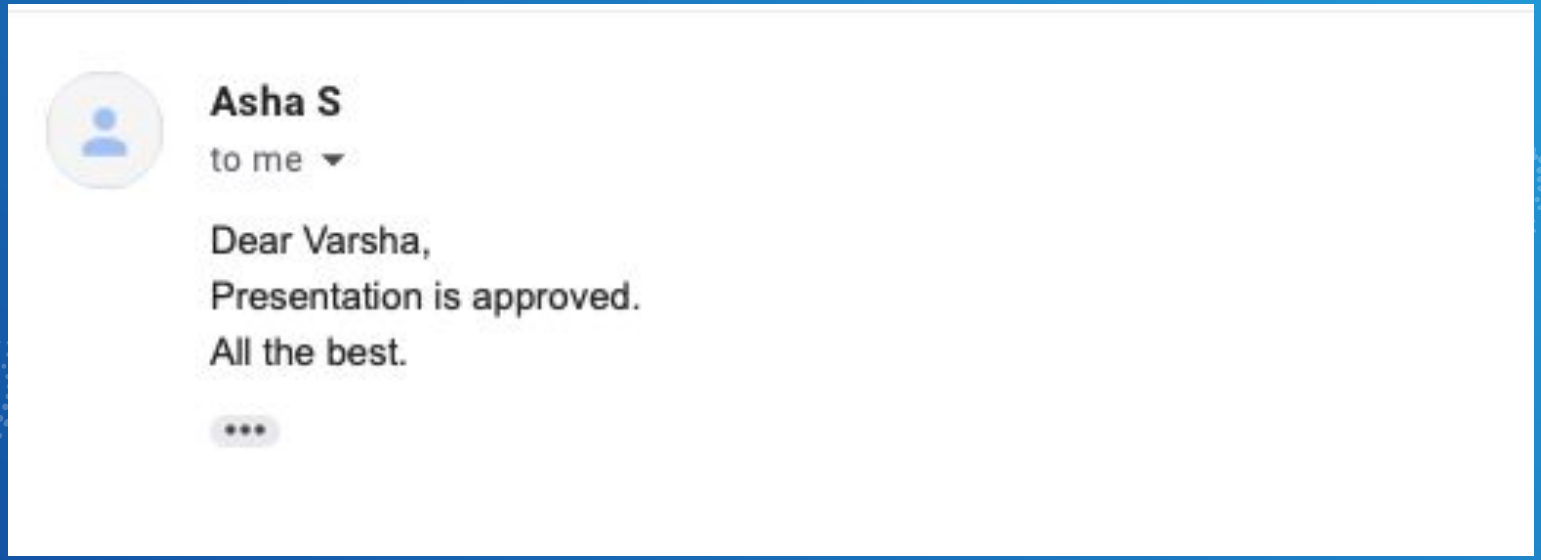
WEB APP/MOBILE APP

INPUT AS IMAGE AND OUTPUT AS DISEASE WITH FURTHER TREATMENT

CONCLUSION

Thus, we can draw conclusions that the scalp of the hair and the health of the person are in correlation with each other. The images of the scalp are captured with the help of a smartphone. Various features such as colour, texture and geometric features were put into use and they were classified using different classifiers. Our system can be improved by including other edge detection algorithms and by implementing localized intensity methods.We were able to extract 13 features and inferred multiples values in the tables above, Alopecia Areata has the maximum accuracy using SVM, Melanoma has maximum accuracy using Logistic Regression. Tinea capitis has maximum accuracy using SVM and healthy scalp has high accuracy in all the models as compared to the other diseases.

FACULTY APPROVAL SCREENSHOT



JOURNAL SUBMISSION

Submitting the paper to the following link:

https://www.journals.elsevier.com/advanced-engineering-informatics

ELSEVIER

ADVANCED ENGINEERING INFORMATICS

The science of supporting knowledge-intensive activities

Advanced Engineering Informatics

Supports *open access*

8.6
CiteScore

5.603
Impact Factor

REFERENCES

- Lee S, Lee JW, Choe SJ, et al. Clinically Applicable Deep Learning Framework for Measurement of the Extent of Hair Loss in Patients With Alopecia Areata. JAMA Dermatol. 2020;156(9):1018–1020. doi:10.1001/jamadermatol.2020.2188
- Ahn CS, Suchonwanit P, Foy CG, Smith P, McMichael AJ. Hair and Scalp Care in African American Women Who Exercise. *JAMA Dermatol.* 2016;152(5):579–580. doi:10.1001/jamadermatol.2016.0093
- W. Chang et al., "A Mobile Device-Based Hairy Scalp Diagnosis System Using Deep Learning Techniques," 2020 IEEE 2nd Global Conference on Life Sciences and Technologies (LifeTech), 2020, pp. 145-146, doi: 10.1109/LifeTech48969.2020.1570617332.
- H. Benhabiles et al., "Deep Learning based Detection of Hair Loss Levels from Facial Images," 2019 Ninth International Conference on Image Processing Theory, Tools and Applications (IPTA), 2019, pp. 1-6, doi: 10.1109/IPTA.2019.8936122.
- Lacarrubba F, Verzì AE, Micali G. Newly Described Features Resulting From High-Magnification Dermoscopy of Tinea Capitis. *JAMA Dermatol.* 2015;151(3):308–310. doi:10.1001/jamadermatol.2014.3313
- Sunyong Seo, Jinho Park, "Trichoscopy of Alopecia Areata: Hair Loss Feature Extraction and Computation Using Grid Line Selection and Eigenvalue", *Computational and Mathematical Methods in Medicine*, vol. 2020, Article ID 6908018, 9 pages, 2020. https://doi.org/10.1155/2020/6908018
- Wang, Wei-Chien, Liang-Bi Chen, and Wan-Jung Chang. 2018. "Development and Experimental Evaluation of Machine-Learning Techniques for an Intelligent Hairy Scalp Detection System" Applied Sciences 8, no. 6: 853. https://doi.org/10.3390/app8060853
- Gupta, Aditya, Ivanova, Iordanka, Renaud, Helen, How good is artificial intelligence (AI) at solving hairy problems? A review of AI applications in hair restoration and hair disorders, 34, 10.1111/dth.14811, Dermatologic Therapy
- Development and qualification of a machine learning algorithm for automated hair counting,Jarek P Sacha and Tamara L Caterino and Brian K. Fisher and Gregory J. Carr and Robert Scott Youngquist and Brian D'Alessandro and Anthony Melione and Douglas Canfield and Wilma Fowler Bergfeld and Melissa Peck Piliang and Raghu Kainkaryam and Mike G Davis,International Journal of Cosmetic Science, 2021, 43, S34 - S41
- Sewoong Kim, Jihun Kim, Minjoo Hwang, Manjae Kim, Seong Jin Jo, Minkyu Je, Jae Eun Jang, Dong Hun Lee, and Jae Youn Hwang, "Smartphone-based multispectral imaging and machine-learning based analysis for discrimination between seborrheic dermatitis and psoriasis on the scalp," Biomed. Opt. Express 10, 879-891 (2019)