INTELLIGENT ALBUM CREATOR USING FACE RECOGNITION

1 INTRODUCTION

1.1 Overview

Over the past decades, photo collections have grown in size due to the rise of digital photography and reduced cost of storage. The introduction of digital photography especially in camera phones further increased the size of photo collections. People tend to take more and more pictures to cherish their memories but that collections end up with tens of thousands of pictures stored on their Pc's / Phones without a proper organization. so there should We are building A face recognition application in our mobile so that whenever you upload your pictures they should get segregated in to respective folders.

1.2 Purpose

To help the people in this problem so that user can efficiently use the application o that he/she may classify the pictures within the short period of time and automatically diverge the images to the respected folders created by the user. The primary goal is to predict the images and divert them into the respected folders with the respected problem.
The main objective of the project is to predict the person present in the image. Here, input for the project is images. If the image is having any person matching with users trained model then that person is recognized and the corresponding image will be directed into respected folders.

2 LITERATURE SURVEY

2.1 Existing Problem

Many people are now well equipped with the technology whether it can be smart phones, cameras, laptops etc. The introduction of digital photography especially in camera phones further increased the size of photo collections. But there is no proper organization technique for this problem. Piling of images may confuse the user .This is the one of the major and most required need for this kind of generation where several companies too working for the betterment for this solution.

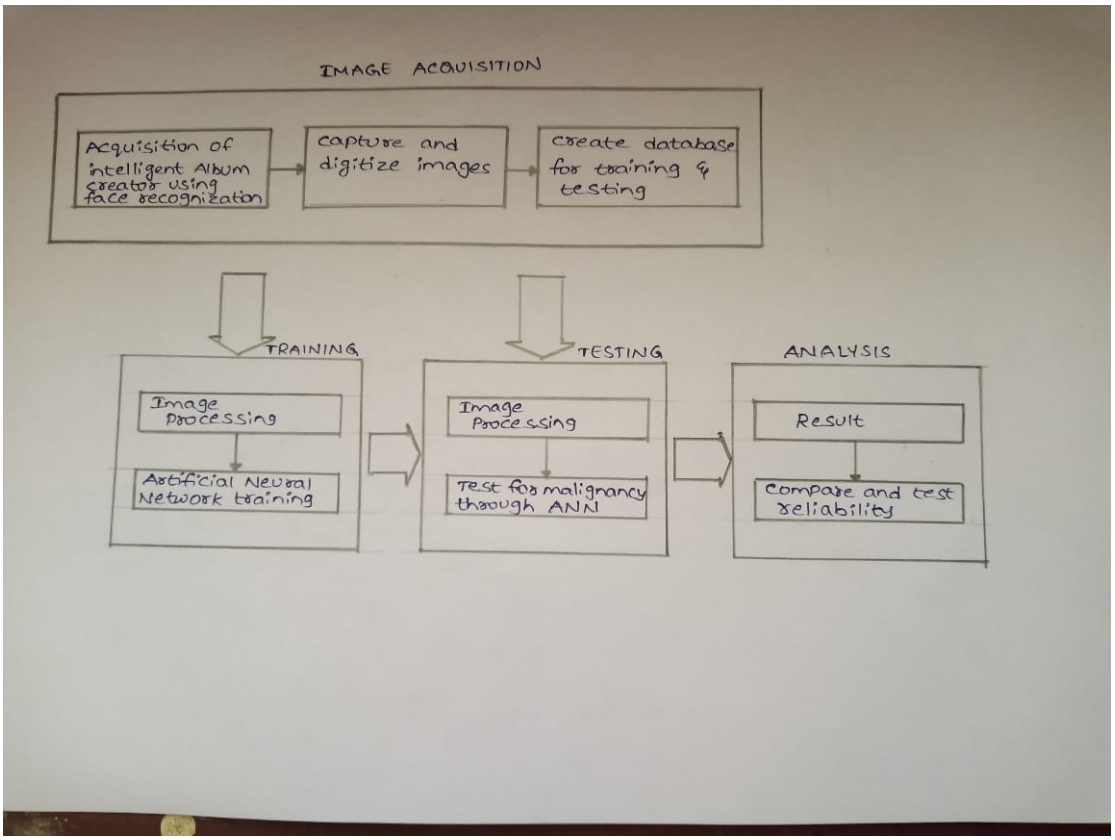
2.2 Proposed Solution

The solution which we put forth is named as the “INTELLIGENT ALBUM CREATOR USING FACE RECOGNITION”. It is developed by using the face recognition techniques such as deep convolution networks. The proposed system consists of a web application by which user can select an image for the prediction of the respected image.

Here effort has been made to classify the images of people for further face recognition applications using convolution neural networks (CNNs). Since the CNNs are capable of handling a large amount of data and can estimate the features automatically, they have been utilized for the task of image classification. The standard Family, friends, colleagues dataset has been selected as the working database for this approach.

3 THEORITICAL ANALYSIS

3.1 Block Diagram



3.2 Hardware / Software Designing

Firstly ,we created the dataset which is of 400 images comprising of three folders such as family , freinds and collegues .Then we created a new python file and we navigated the directory for the needed path and then we uploaded the required modules such as keras,cnn,dense,pooling
etc for the requirement purpose.

secondly ,we made a model and then add the convolution layer and the pooling layer and the dense and output layer as required.we utilized the activation functions such as relu and softmax functions in the model.Next by import the datagenerator module we navigated to the datset folder and downloaded the train and test data required and check the number of images in the folder and printed the classes indices too.Now,we compiled the model and then we run the model by giving epochs as 10 and the steps for epoch as 100 it takes a few time as the model has to learn the features by the technique of extracting.After the learning cycle is over we created a requires .h5 file namely p1.h5 and we optimized the model too and then we have writtened the code for the prediction and given images for the model for prediction purpose of which it predicted correctly as reqired and trained.we made few adjustments if the errors generated and the errors are rectified and then we move on to the web application building task.

Finally,we created the html frontend page.we created the python file of which it is important for the communication between the html and the model file which is p1.h5.we wrritend the script code in the main.js file .after doing the neccessary steps in making them in sink with each other we were onto the nextstep of executing the project.With the help of Anaconda prompt first we navigated to the folder where the project resides and then we executed our flask file and then opened the localhost at port address 8000 where our project will be projected we performed the predictions by giving the required images and predictions were correctly acheived.At last we uploaded the application onto the IBM cloud .

Our project do require the software’s such as jupyter to code the model and to train the model .Anaconda prompt for navigation and application running purpose.IBM cloud so as to deploy the model in the cloud so everyone can access it without no problem of time and area. Github is used for the deployment of the project.Html , Bootstrap , css and JavaScript were used for the building of the frontend of the application. Web frame work -Flask is also used and also we use the p1.h5 file which will be the file of the model trained .

4 EXPERIMENTAL INVESTIGATIONS

Our database contains 400 images belongs to 3 classes.

1.Family – we deployed the pictures of our friend as the family dataset.

2.Freinds – we deployed the pictures of one person as a friends dataset .

3.Collegues – we deployed the pictures of another person as a colleagues dataset.

and these classes are labelled as 0,1,2 respectively.

The methodology we used in our model is image classification. For this we used Convolution Neural Networks (CNNs).

Image classification - assigning pixels in the image to categories or classes of interest **.**Image classification is a process of mapping numbers to symbols. In order to classify a set of data into different classes or categories, the relationship between the data and the classes into which they are classified must be well understood. To achieve this by computer, the computer must be trained. Training is key to the success of classification .Classification techniques were originally developed out of research in Pattern Recognition field .

CNNs:

➢ A convolutional neural network (CNN or ConvNet) is one of the most popular algorithms for deep learning, a type of machine learning in which a model learns to perform classification tasks directly from images, video, text, or sound.

➢ CNNs are particularly useful for finding patterns in images to recognize objects, faces, and scenes. They learn directly from image data, using patterns to classify images and eliminating the need for manual feature extraction.

➢ Applications that call for object recognization and computer vision — such as self-driving cars and face recognization— rely heavily on CNNs. Depending on your application, you can build a CNN from scratch, or use a pretrained model with your data set .

➢ Using CNNs for deep learning has become increasingly popular due to three important factors:

➢ CNNs eliminate the need for manual feature extraction—the features are learned directly by the CNN.

➢ CNNs produce state-of-the-art recognition results.

➢ CNNs can be retrained for new recognition tasks, enabling you to build on preexisting networks.

➢ A convolutional neural network can have tens or hundreds of layers that each learn to detect different features of an image. Filters are applied to each training image at different resolutions, and the output of each convolved image is used as the input to the next layer. The filters can start as very simple features, such as brightness and edges, and increase in complexity to features that uniquely define the object.

➢ Like other neural networks, a CNN is composed of an input layer, an output layer, and many hidden layers in between.

Four of the most common layers are: convolution, activation or ReLU, softmax and pooling.

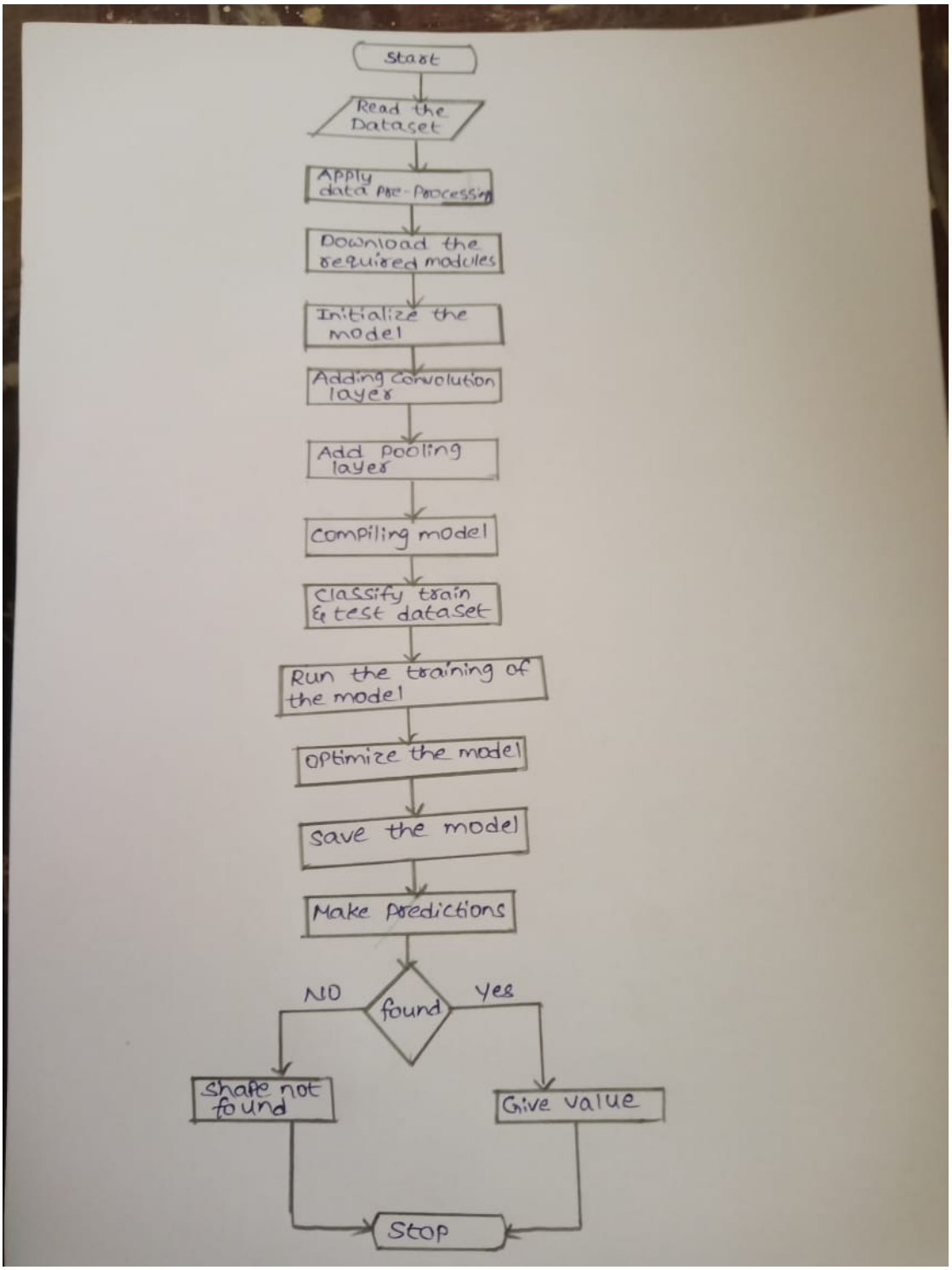
Convolution puts the input images through a set of convolutional filters, each of which activates certain features from the images.

Rectified linear unit (ReLU) allows for faster and more effective training by mapping negative values to zero and maintaining positive values. This is sometimes referred to as activation, because only the activated features are carried forward into the next layer.

Softmax function a wonderful activation function that turns numbers aka logits into probabilities that sum to one. Softmax function outputs a vector that represents the probability distributions of a list of potential outcomes.

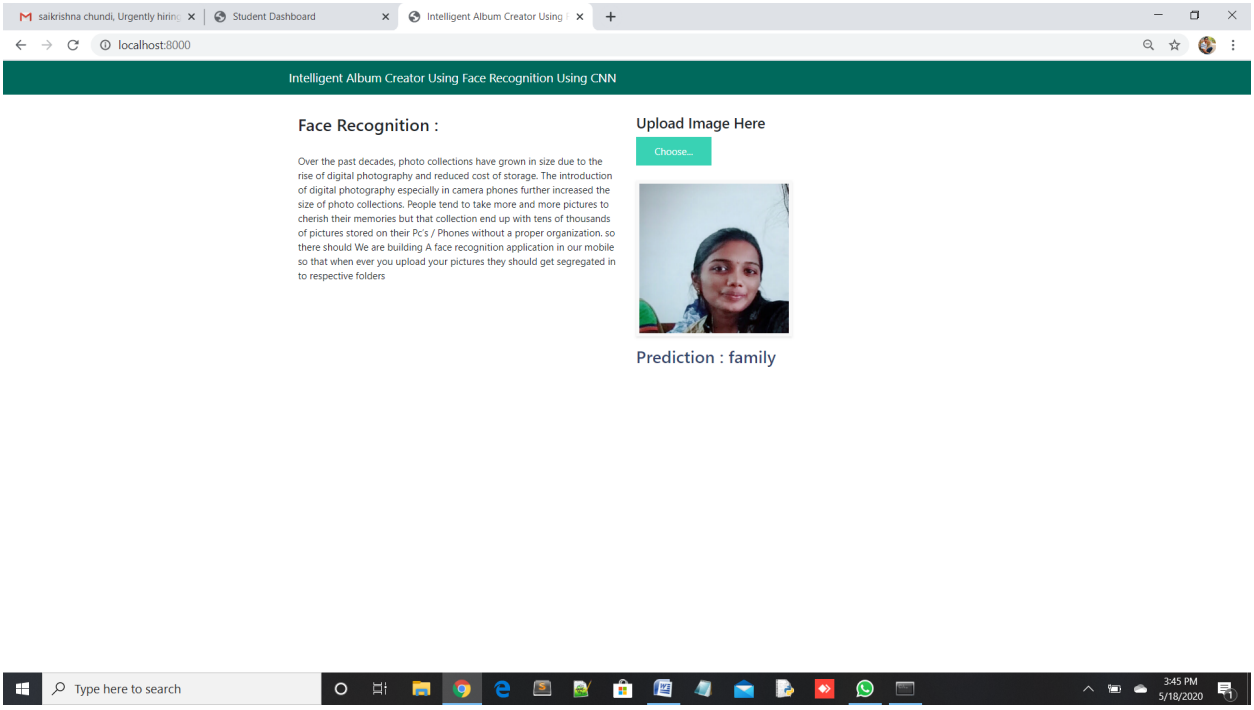
Pooling simplifies the output by performing nonlinear down sampling, reducing the number of parameters that the network needs to learn. These operations are repeated over tens or hundreds of layers, with each layer learning to identify different features.

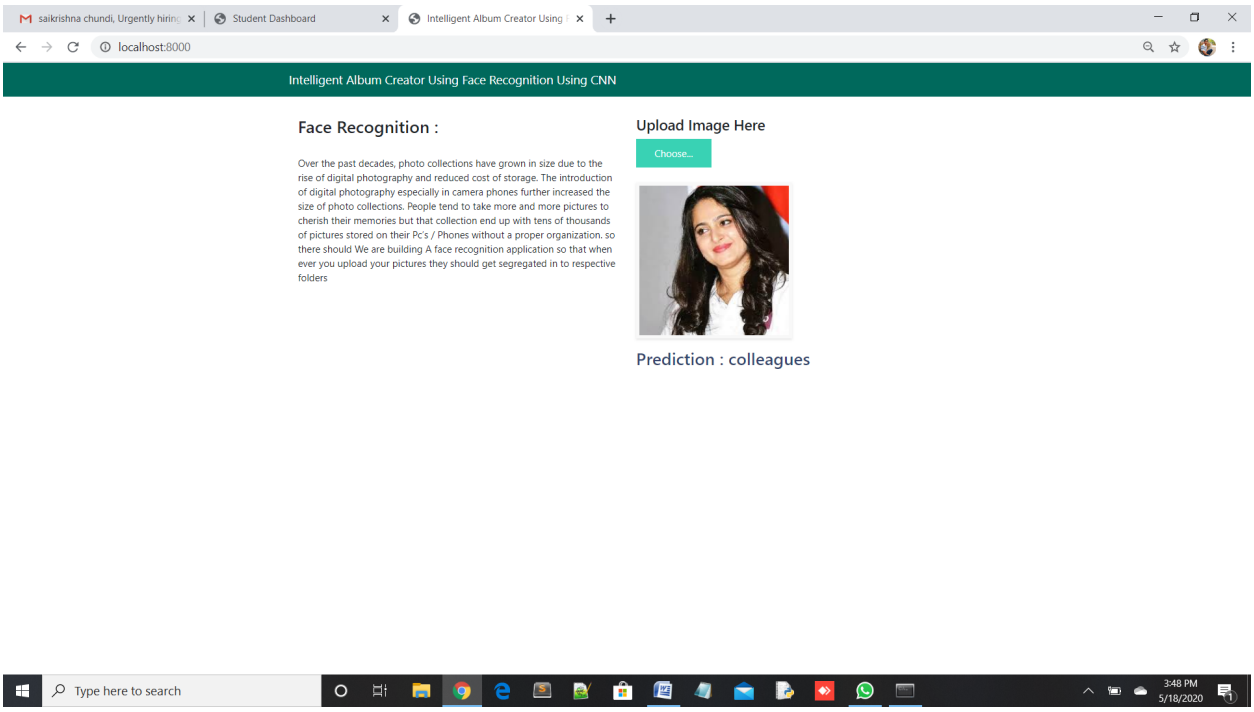
5 FLOWCHART


Start
Read the Dataset
Apply data Pre-Processing
Download the required modules
Initialize the model
Adding convolution layer
Add pooling layer
compiling model
classify train & test dataset
Run the training of the model
optimize the model
save the model
Make predictions
found
NO
Yes
shape not found
Give value
Stop

6 RESULT

Finally model is built with a accuracy rate of 77.07% and a web application is built and results are predicted.





7 ADVANTAGES & DISADVANTAGES

Full Automation is an advantage instead of manual recognition, which is done by security guards or the official representatives outside of company's premises, the facial recognition tech automates the identification process and ensures its flawlessness every time without any haltings. You won't even need an employee to monitor the cameras 24/7.Automation means convenience and reduces the expenses too. Therefore, any entrepreneur would be fond of the fact that image identification systems are fully automated.

Easy Integration Process is also an advantage most of the time, integratable facial recognition tools work pretty flawlessly with the existing security software that companies have installed. And they're also easy to program for interaction with a company's computer system.

Why is it great for business? Well, you won't need to spend additional money and time on redeveloping your own software to make it suitable for FRT integration. Everything will be already adaptable.

Image Size & Quality is a disadvantage

It's obvious that a facial recognition is a super advanced software that requires HQ digital cameras for algorithms to operate accurately. A face-detection system captures a face in the photo or screen-shot from a video, then the relative size of that face image will be compared with the size of enrolled one. So, the photo's quality here affects the whole face recognition process, how well it would be done. Imagine, the already small size picture is coupled with a distance that was between a target and a CCTV… What proportions will the detected face have? No more than 100×200 pixels.

Pretty hard to get a clear identity in such case. What's more, scanning a photo for varying face sizes is a processor-intensive task. Most systems allow identification of a face-size range to eliminate false recognition and speed up image processing. But the initial investment in such face tracking software is not a cheap one, however, it will pay off in no time.

8 APPLICATIONS

1.Person recognization
2.Home security by recognizing the people entering house etc

9 CONCLUSION

Face recognition technology has one goal: to identify human faces. Out of all biometric systems available, it is the least intrusive and fastest technology.Instead of asking people to scan their fingerprints, hands or irises, face recognition systems can take pictures of people's faces unobtrusively. In most cases, the subjects are not even aware of the process. This removes the feeling of being under surveillance or invasion of privacy.

Facial recognition systems are also highly effective in producing results, all thanks to the use of neural network algorithms especially convolutional neural networks.

10 FUTURE SCOPE

1.DIAGNOSE DISEASES - Face recognition can be used to diagnose diseases that cause detectable changes in appearance. As an example, the National Human Genome Institute Research Institute, uses face recognition to detect a rare disease called DiGeorge syndrome, in which there is a portion of the 22nd chromosome missing. Face recognition has helped diagnose the disease in 96% of cases. As algorithms get even more sophisticated, face recognition will become an invaluable diagnostic tool for all sorts of conditions.

2. UNLOCK PHONES - A variety of phones including the latest iPhone are now using face recognition to unlock phones. This technology is a powerful way to protect personal data and ensure that, if a phone is stolen, sensitive data remains inaccessible by the perpetrator.

3.VIDEO UPLOADTION

11 BIBILOGRAPHY

1.Sudharashan Ravichandra

Hands – on deep learning algorithms with python

2.online links –

https://machinelearningmastery.com/multi-class-classification-tutorial-keras-deep-learning-library/

https://keras.io/getting_started/

APPENDIX

1.Source Code

```
import os

os.getcwd()

os.chdir("C:/Users/sai/Desktop/Remote internship 2020/Data sets")

os.getcwd()


from keras.models import Sequential

from keras.layers import Dense

from keras.layers import Convolution2D

    from keras.layers import MaxPooling2D

    from keras.layers import Flatten


    #Intialize the model

    model=Sequential()


    # Add Convolution Layer

    model.add(Convolution2D(32,(3,3),input_shape=(64,64,3),activation="relu"))
```

```
#Add Pooling Layer

model.add(MaxPooling2D(pool_size = (2, 2)))


#Add Flattening Layer

model.add(Flatten())


#Add Hidden Layer

model.add(Dense(init="random_uniform",activation="relu",output_dim=150))


#Add Output layer

model.add(Dense(init="random_uniform",activation="softmax",output_dim=3))


#Compile the model

model.compile(loss="categorical_crossentropy",optimizer="sgd",metrics=["accu
racy"])


from keras.preprocessing.image import ImageDataGenerator


train_datagen = ImageDataGenerator(rescale = 1./255,

                                   shear_range = 0.2,

                                   zoom_range = 0.2,

                                   horizontal_flip = True)


test_datagen = ImageDataGenerator(rescale = 1./255)


x_train = train_datagen.flow_from_directory('dataset/traindata',

                                            target_size = (64, 64),
```

```
                                    batch_size = 32,
                                       class_mode = 'categorical')
x_test = test_datagen.flow_from_directory('dataset/testdata',
                                  target_size = (64, 64),
                                  batch_size = 32,
                                  class_mode = 'categorical')


print(x_train.class_indices)




# model.fit_generator(x_train,samples_per_epoch =
80,epochs=25,validation_data=x_test,nb_val_samples=90)
model.fit_generator(x_train,
                     steps_per_epoch = 100,
                     epochs = 10,
                    validation_data = x_test,
                     validation_steps = 63)


model.save("p1.h5")


!pip install opencv-python


from keras.models import load_model
import numpy as np
import cv2
```

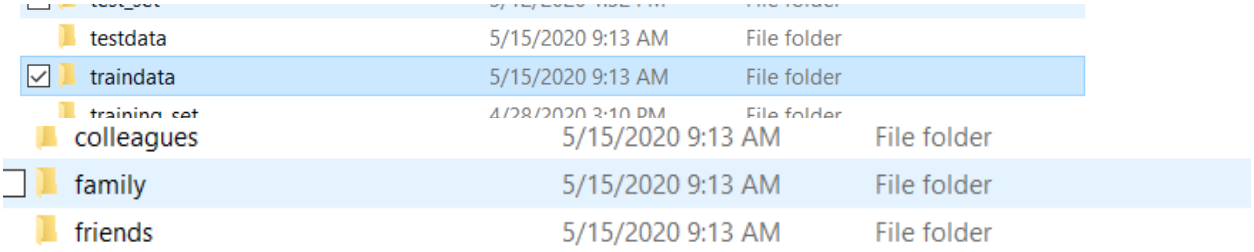
```
model = load_model('p1.h5')
# model.compile(loss='categorical_crossentropy',
#                     optimizer='adam',
#                     metrics=['accuracy'])

from skimage.transform import resize

def detect(frame):
    try:
        img = resize(frame,(64,64))
        img = np.expand_dims(img,axis=0)
        if(np.max(img)>1):
            img = img/255.0
        prediction = model.predict(img)
        print(prediction)
        prediction = model.predict_classes(img)
        print(prediction)
    except AttributeError:
        print("shape not found")

frame=cv2.imread("prabhas.jpg")
data = detect(frame)

frame=cv2.imread("varshini.jpg")
data = detect(frame)
```

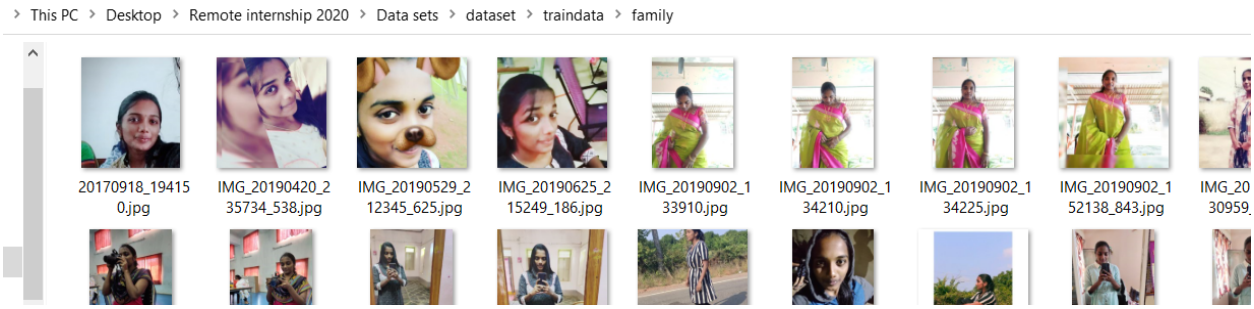
frame=cv2.imread("anushka.jpg")

data = detect(frame)

2.screenshots of tasks

1.Create Train And Test Folders





2.Import The ImageDataGenerator Library

```
from keras.preprocessing.image import ImageDataGenerator
```

3.Configure ImageDataGenerator Class

```
train_datagen = ImageDataGenerator(rescale = 1./255,
                                   shear_range = 0.2,
                                   zoom_range = 0.2,
                                   horizontal_flip = True)

test_datagen = ImageDataGenerator(rescale = 1./255)
```

4.Apply ImageDataGenerator Functionality To Trainset And Testset

```
x_train = train_datagen.flow_from_directory('dataset/traindata',
                                                 target_size = (64, 64),
                                                 batch_size = 32,
                                                     class_mode = 'categorical')
x_test = test_datagen.flow_from_directory('dataset/testdata',
                                               target_size = (64, 64),
                                               batch_size = 32,
                                               class_mode = 'categorical')
```

5.Importing The Model Building Libraries

```
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
```

6.Initializing The Model

```
#Intialize the model
model=Sequential()
```

7.Adding CNN Layers

```
# Add Convolution Layer
model.add(Convolution2D(32,(3,3),input_shape=(64,64,3),activation="relu"))
```

```
#Add Pooling Layer
```

8.Adding Dense Layers

```
#Add Pooling Layer
model.add(MaxPooling2D(pool_size = (2, 2)))
```

```
WARNING:tensorflow:From C:\Users\komali\Anaconda3\lib\site-
n.max_pool is deprecated. Please use tf.nn.max_pool2d inste
```

```
#Add Flattening Layer
model.add(Flatten())
```

```
#Add Hidden Layer
model.add(Dense(init="random_uniform",activation="relu",output_dim=150))
```

```
C:\Users\komali\Anaconda3\lib\site-packages\ipykernel_launcher.py:2: UserWarning: Update yo
I: `Dense(activation="relu", units=150, kernel_initializer="random_uniform")`
```

9.Configure The Learning Process

```
#Add Output layer
model.add(Dense(init="random_uniform",activation="softmax",output_dim=3))
```

```
C:\Users\komali\Anaconda3\lib\site-packages\ipykernel_launcher.py:2: UserWarning: Update your `Dense` ca
```

```
#Compile the model
model.compile(loss="categorical_crossentropy",optimizer="sgd",metrics=["accuracy"])
```

10.Train And Test The Model

```
print(x_train.class_indices)
```

```
{'colleagues': 0, 'family': 1, 'friends': 2}
```

```
# model.fit_generator(x_train,samples_per_epoch = 80,epochs=25,validation_data=x_test,nb_val_samples=90)
model.fit_generator(x_train,
                        steps_per_epoch = 100,
                        epochs = 10,
                    validation_data = x_test,
                        validation_steps = 63)
```

```
WARNING:tensorflow:From C:\Users\komali\Anaconda3\lib\site-packages\keras\backend\tensorflow_backend.py:422: The name tf.glo
bal_variables is deprecated. Please use tf.compat.v1.global_variables instead.

Epoch 1/10
100/100 [==============================] - 242s 2s/step - loss: 1.0565 - accuracy: 0.4465 - val_loss: 1.0035 - val_accuracy:
0.4526
Epoch 2/10
100/100 [==============================] - 222s 2s/step - loss: 0.9585 - accuracy: 0.5519 - val_loss: 0.9741 - val_accuracy:
0.5474
Epoch 3/10
100/100 [==============================] - 221s 2s/step - loss: 0.8734 - accuracy: 0.6068 - val_loss: 1.0725 - val_accuracy:
0.5895
Epoch 4/10
100/100 [==============================] - 221s 2s/step - loss: 0.8218 - accuracy: 0.6503 - val_loss: 0.8312 - val_accuracy:
0.5895
Epoch 5/10
100/100 [==============================] - 218s 2s/step - loss: 0.7603 - accuracy: 0.6771 - val_loss: 1.2282 - val_accuracy:
0.4737
Epoch 6/10
100/100 [==============================] - 182s 2s/step - loss: 0.7260 - accuracy: 0.6826 - val_loss: 1.0803 - val_accuracy:
0.5684
Epoch 7/10
100/100 [==============================] - 187s 2s/step - loss: 0.7081 - accuracy: 0.6990 - val_loss: 0.7433 - val_accuracy:
0.4947
Epoch 8/10
100/100 [==============================] - 184s 2s/step - loss: 0.6419 - accuracy: 0.7277 - val_loss: 1.1795 - val_accuracy:
0.4842
```

11.optimize The Model and save the model

```
model.save("p1.h5")
```

```
!pip install opencv-python
```

```
Requirement already satisfied: opencv-python in c:\users\komali\anaconda3\lib\site-packages
Requirement already satisfied: numpy>=1.14.5 in c:\users\komali\anaconda3\lib\site-packages
```

```
from keras.models import load_model
import numpy as np
import cv2
```

```
model = load_model('p1.h5')
# model.compile(loss='categorical_crossentropy',
#                         optimizer='adam',
#                         metrics=['accuracy'])

from skimage.transform import resize

def detect(frame):
    try:
        img = resize(frame,(64,64))
        img = np.expand_dims(img,axis=0)
        if(np.max(img)>1):
            img = img/255.0
        prediction = model.predict(img)
        print(prediction)
        prediction = model.predict_classes(img)
        print(prediction)
    except AttributeError:
        print("shape not found")
```

```
frame=cv2.imread("prabhas.jpg")
data = detect(frame)
```

```
[[0.07615554 0.09769117 0.82615334]]
[2]
```

```
frame=cv2.imread("varshini.jpg")
data = detect(frame)
```

```
[[0.05283424 0.837048   0.11011768]]
[1]
```

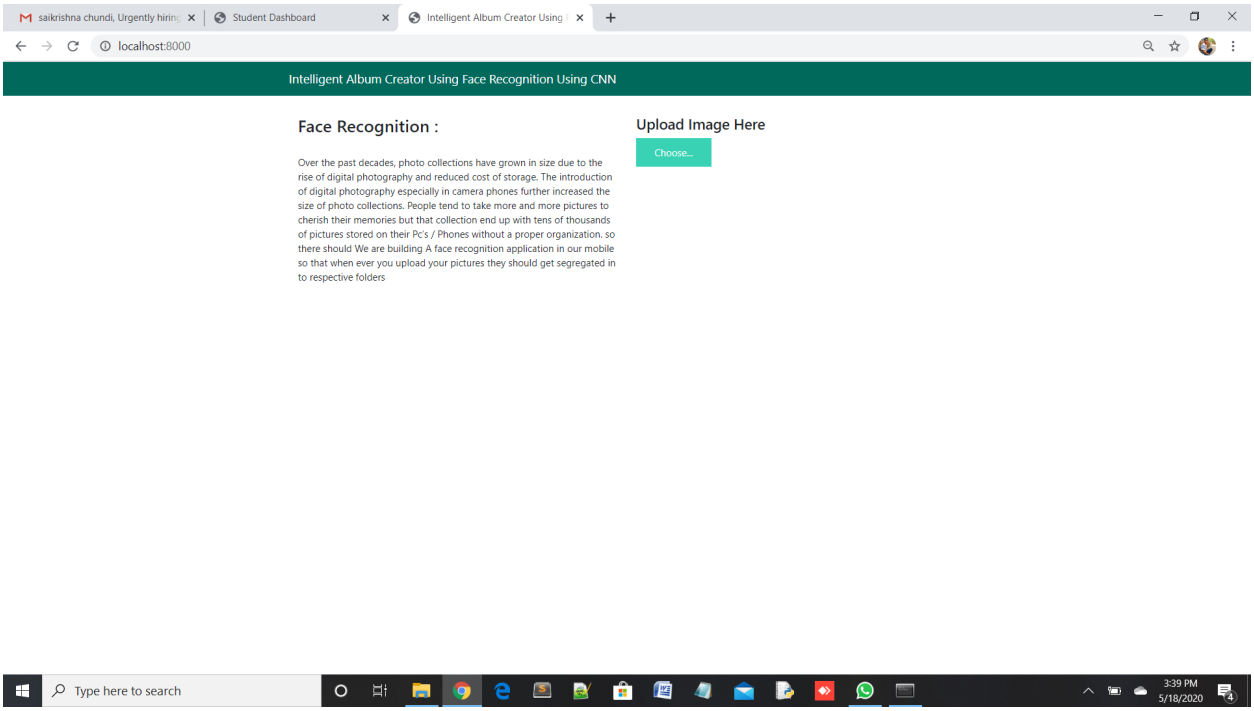
```
frame=cv2.imread("anushka.jpg")
data = detect(frame)
```

```
[[0.5498079  0.13528776 0.31490433]]
[0]
```

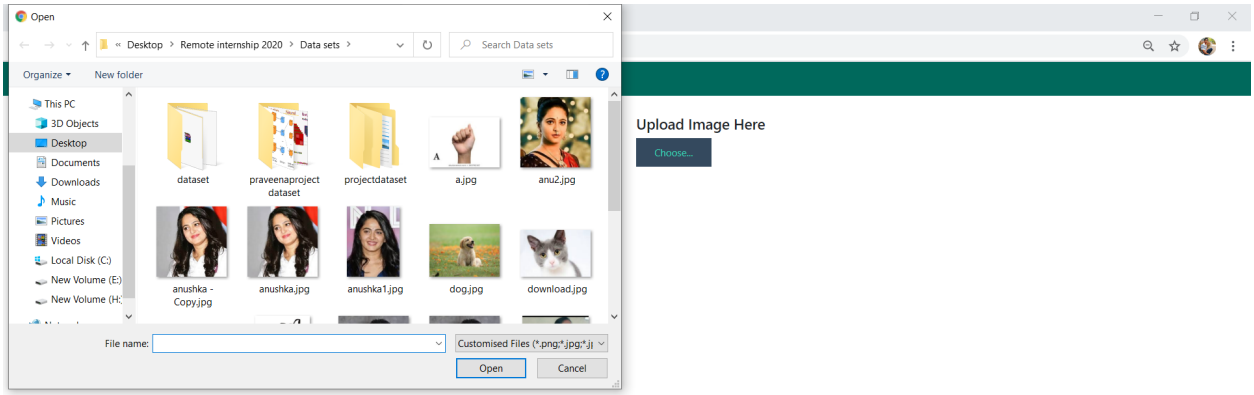
12.Model Building

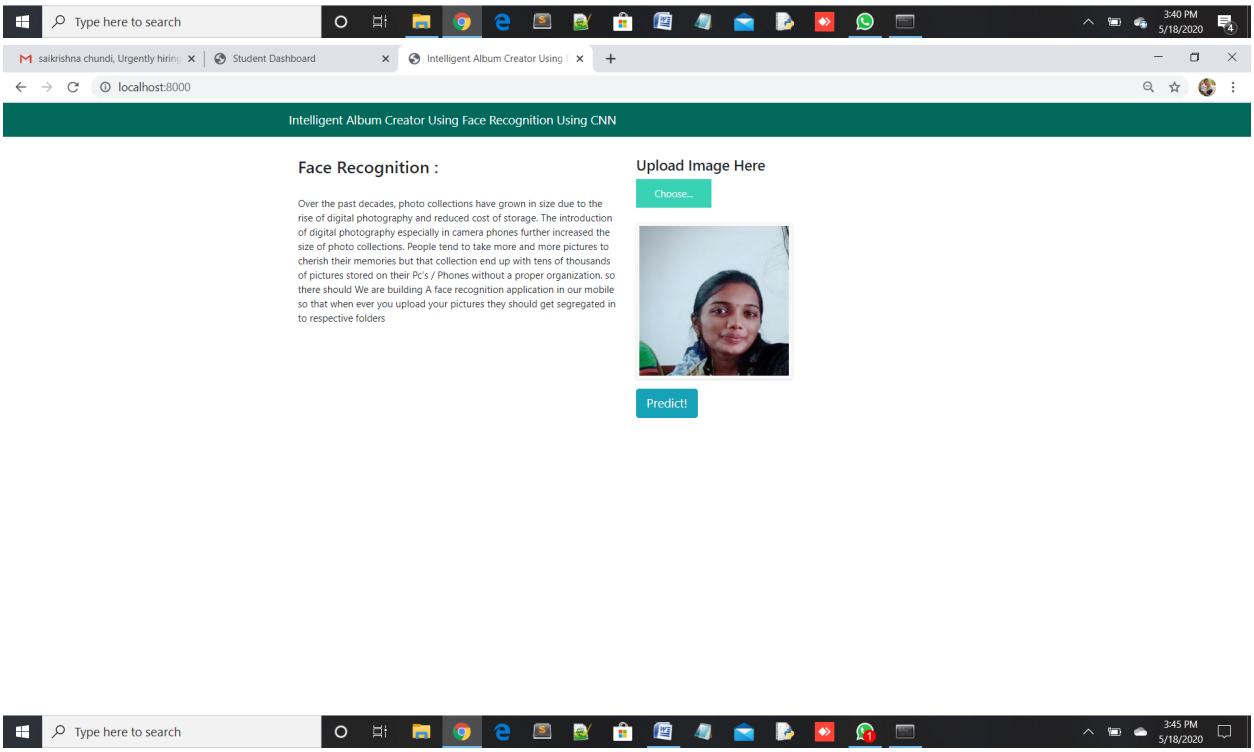
realproject

Name	Date modified	Type	Size
css	5/16/2020 3:59 PM	File folder	
js	5/16/2020 3:59 PM	File folder	
kubernetes	5/16/2020 4:00 PM	File folder	
static	5/16/2020 3:59 PM	File folder	
templates	5/16/2020 3:59 PM	File folder	
uploads	5/18/2020 3:48 PM	File folder	
.cfignore	12/26/2019 2:18 AM	CFIGNORE File	1 KB
.dockerignore	12/26/2019 2:18 AM	DOCKERIGNORE F...	1 KB
.gitignore.txt	12/26/2019 2:18 AM	Text Document	1 KB
hello.py	5/18/2020 3:22 PM	Python File	2 KB
LICENSE	12/26/2019 2:18 AM	File	12 KB
manifest.yml	4/17/2020 5:31 AM	YML File	1 KB
p1.h5	5/16/2020 12:17 PM	H5 File	18,043 KB
Procfile	12/26/2019 2:18 AM	File	1 KB
README.md	12/26/2019 2:18 AM	MD File	6 KB
README-kubernetes.md	12/26/2019 2:18 AM	MD File	4 KB
requirements.txt	5/10/2020 11:47 PM	Text Document	1 KB
runtime.txt	5/10/2020 11:38 PM	Text Document	1 KB
setup.py	12/26/2019 2:18 AM	Python File	1 KB
vcap-local.template.json	12/26/2019 2:18 AM	JSON File	1 KB

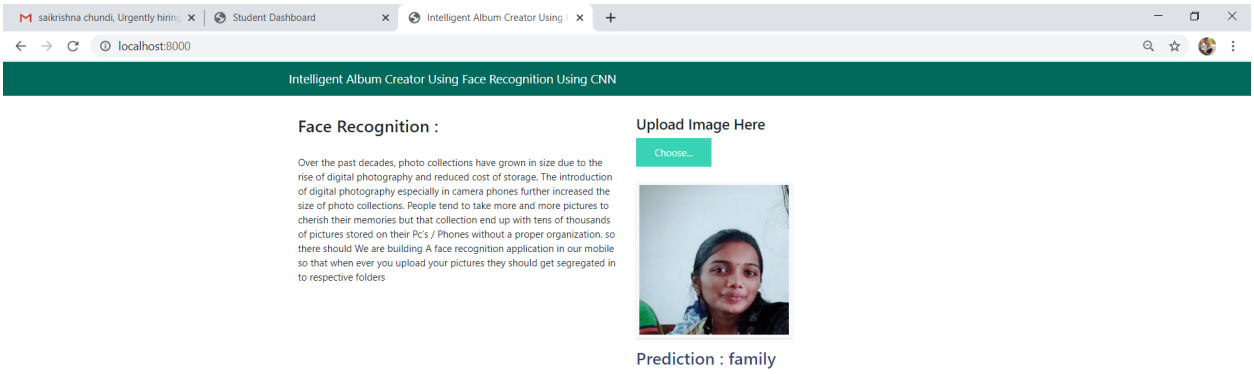


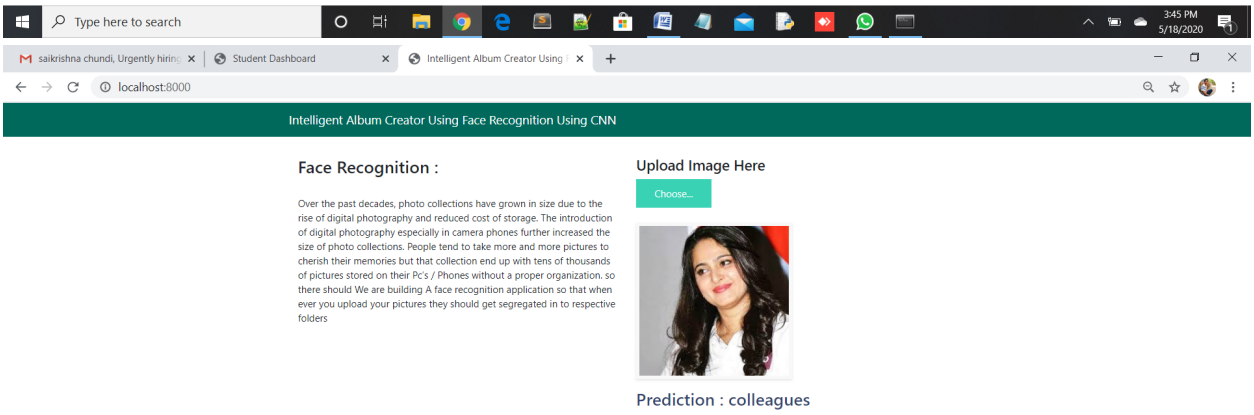
13. Send Clicked Image To Model





14. Display Predicted Text





15. Segregate The Pictures