Image Analysis and Computer Vision Final Project Report

Name: Sai Mounica Chenuri Venkata V Lakshmi Phalguna

WSU ID:Z963A577

CNN:

CNN is used for image classification for high accuracy, So the classification can be done by connecting the entire neural network or by partially dropping certain network connections by adding certain probability for dropout layer.

Imported necessary Libraries

```
from keras.preprocessing.image import ImageDataGenerator, load_img
import pandas as pd
import matplotlib.pyplot as plt
import os
import random
import cv2
import glob
from keras.preprocessing.image import img_to_array
import numpy as np
from sklearn.model_selection import train_test_split
from tensorflow.keras.utils import to_categorical, plot_model
from keras.models import Sequential
from keras.layers import BatchNormalization, Conv2D, MaxPooling2D, Activation, Flatten, Dropout, Dense
from keras import backend as B
from skimage.feature import hog
from sklearn import svm
```

- All the mentioned libraries are used to implement the model,
- Created a function and assigned labels to the given dataset
- Labels were dived into labels and gender set

Males =0 and Females =1

```
labels = []
for lab1 in copy.Male:
 if lab1 == 0:
   lab = 0
 else:
 lab = 1
 labels.append([lab])
```

- And normalized the data by dividing with the largest pixel value and append the values to fit the data properly.
- Append is used to add the data to end of the given list

```
gender = []
for img in images:
 imgfile = cv2.imread(img)
```

- We read the image from the image file
- Resizing of image is done so that it can fit the model without throwing any errors

```
 imgfile = cv2.resize(imgfile,(64, 64))
 imgfile = img_to_array(imgfile)
 gender.append(imgfile)
```

- Used train_test_split for dividing the dataset into test and train samples using random value of 70,
- For any given model we divide the data into test and train samples, so that we can train the model on the train set and test the model on test set of the data,
- When taking the random state=70 the data is divided at a randomstate of 70 by shuffling the data,
- We also used categorical to assign the values based on the cateogory, since we have 2 classes that is features and Male and female, the gender classification is set in labels and the features are stored in genders[]

```
(a_train, a_test, b_train, b_test) = train_test_split(gender, labels, test_size=0.35, random_state=70)
b_train = to_categorical(b_train, num_classes=2)
b_test = to_categorical(b_test, num_classes=2)
#using default values for image data generator to generate random state of images
```

- We create a augummented set by using ImageDatagenerator from python library this function already has it’s own predefined values. But I Have tried changing the values to fit the model that I have created.

```
aug = ImageDataGenerator(rotation_range=10, width_shift_range=0.2, height_shift_range=0.2, shear_range=0.1, zoom_range=0.2,
                    horizontal_flip=True, fill_mode='nearest')
```

- Now Creating the CNN model
- With neural networks, used relu and sigmoid and output layer as softmax
- As this is a 2 class classifier male and female so class is =2
- We maintain input shape to the image dimension
- For the model to be more precises, added few dropout functions=0.25
- We used 2DConv 32,64,128

```
model.add(Conv2D(32, (3,3), padding="same", input_shape=img_size))
model.add(Activation("relu"))
model.add(MaxPooling2D(pool_size=(3,3)))
model.add(Dropout(0.25))
model.add(Conv2D(64, (3,3), padding="same"))
model.add(Activation("sigmoid"))

model.add(Conv2D(64, (3,3), padding="same"))
model.add(Activation("relu"))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

model.add(Conv2D(128, (3,3), padding="same"))
model.add(Activation("sigmoid"))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())
model.add(Dense(cls))
model.add(Activation("softmax"))
model.summary()
```

- printing out the model summary and declaring the values
- used Optimizer Adam
- loss function binary_crossentropy to know the actual difference between male and female since we are just using the model to classify if it’s male or Female we use binary as loss function.

- size = 32
- epochs = 25 this is the value given to the network for it to pass through the entire layer for 25 times, for more precision we can even increase the value.
- lr = 1e-3
- and fitting the model using the

```
cnn = model.fit(aug.flow(a_train, b_train, batch_size=size),
                validation_data=(a_test,b_test),epochs=epochs, verbose=1)
```

- and printing out the accuracy score

```
accuracy = model.evaluate(a_test, b_test)
accuracy*100
```

and the accuracy report =0.5909

HOG features:

HOG is used to extract features from a given image data set, this feature is used for object detection

Using the same kind of features from above and defining the model for HOG:

```
def hogmodel(ht, w, d):
  model = Sequential()
  img_size = (ht, w, d)
  fd=model.add(Flatten())
  fd, hog_image = hog(imgfile, orientations=9, pixels_per_cell=(8, 8),
                cells_per_block=(2, 2), visualize=True, multichannel=True)
  return model
```

trying to print out the model

fitting the dataset using HOG features,

- printing out the model summary and declaring the values
- used Optimizer Adam
- loss function binary_crossentropy to know the actual difference between male and female since we are just using the model to classify if it’s male or Female we use binary as loss function.
- size = 32
- epochs = 25 this is the value given to the network for it to pass through the entire layer for 25 times, for more precision we can even increase the value.
- lr = 1e-3
- and fitting the model using the

-

```
model = hogmodel(ht=64, w=64,d=3)

epochs = 25

lr = 1e-3

opt = Adam(learning_rate=lr, decay=lr/epochs)

model.compile(loss=”binary_crossentropy”, optimizer=opt, metrics=[“accuracy”])

HOG = model.fit(aug.flow(hog_x_train, hog_y_train),

                validation_data=(hog_x_test,hog_y_test),epochs=epochs, verbose=1)
```

- We try to fit the HOG features using SVM classifier, it is a linear regression which takes in values and trains the data set using those values
- And we then predict the values using the test data from the test data that we have created
- We the plot the accuracy report
- We see that the accuracy is 0.4253 which is less that that of CNN
- So we use CNN classifier because it is better compared to HOG.

```
hog_clf = svm.SVC()
hog_clf.fit(hog_x_train,hog_y_train)
# Predict on the test features, print the results
hog_y_pred = hog_clf.predict(hog_x_test)
hog_y_pred
accuracy = model.evaluate(hog_x_test, hog_y_test)
accuracy*100
```