IMAGE RECOGNITION

TO

DETECT MULTIPLE ATTRIBUTES

Statement of Work

Course: AIDI 1002-01 : Final Project

Course Facilitator: Marcos Bittencourt

Submitted by:

Tarun Kumar Punna

PROJECT OVERVIEW:

Image recognition is one of the many applications of Machine Learning, it can solve problems for security purposes, object detection, face detection, healthcare, entertainment, among others. This application has an enormous potential to help our society1 , so it is important to find new uses for this tool, improve the current methods and get more accurate and useful insights from it.

In this project, the researcher creates three models to predict three attributes out of the same picture, it uses the Inception-V3 as based model for image recognition , and the data set uses is the CelebA dataset , which contains over 200,000 celebrity images and 40 binary attribute annotations per image.

PROBLEM STATEMENT:

Many machine learning algorithms used in face recognition or object detection are built to detect a specific attribute, for example, the mood of a person in a picture, the algorithm will predict if the person is happy, sad, neutral, surprised, etc. What if we want to explore many non-exclusive attributes that are of a picture at the same time? As to classify if the person is smiling or not and at the same time know if is wearing eyeglasses or if is wearing a hat. Most probably the same algorithm will not perform well predicting two or more attributes. This is a classification problem for each of the selected attributes, one model per attribute, but using the same image. The output will be the predicted attribute of each model.

The use of different algorithms, one of them predicting a specific attribute will give us better insights of the picture to be analyzed, results will be measure against the real target and the algorithms will be replicable to pictures outside of the dataset used in this project.

The goal is to build three classification models based on image recognition, the attributes to be predicted are:

- If the subject is smiling or not.

- If the subject is female or male.

- If the subject is young or not.

DATA SOURCES

A popular component of computer vision and deep learning revolves around identifying faces for various applications from logging into your phone with your face or searching through surveillance images for a particular suspect. This dataset is great for training and testing models for face detection, particularly for recognising facial attributes such as finding people with brown hair, are smiling, or wearing glasses. Images cover large pose variations, background clutter, diverse people, supported by a large quantity of images and rich annotations.

This data was originally collected by researchers at MMLAB, The Chinese University of Hong Kong (specific reference in Acknowledgment section).

Contents:

Overall

- 202,599 number of face images of various celebrities
- 10,177 unique identities, but names of identities are not given
- 40 binary attribute annotations per image
- 5 landmark locations

Data Files:

- imgalignceleba.zip: All the face images, cropped and aligned
- listevalpartition.csv: Recommended partitioning of images into training, validation, testing sets. Images 1-162770 are training, 162771-182637 are validation, 182638-202599 are testing
- listbboxceleba.csv: Bounding box information for each image. "x1" and "y1" represent the upper left point coordinate of bounding box. "width" and "height" represent the width and height of bounding box
- listlandmarksalign_celeba.csv: Image landmarks and their respective coordinates. There are 5 landmarks: left eye, right eye, nose, left mouth, right mouth
- listattrceleba.csv: Attribute labels for each image. There are 40 attributes. "1" represents positive while "-1" represents negative

METRICS

The metrics to measure the performance of the models are:

Accuracy:

Accuracy is a common metric to measure binary classifiers, this adapts well to the problem to be solved, the corresponding formula to this metrics is:

$$accuracy = \frac{True\ Positives + True\ Negatives}{Total\ number\ of\ pictures}$$

Where:

- True Positives: Number of times the model predicted to the positive class and it was the positive class.
- True Negatives: Number of times the model predicted to the negative class and it was the negative class.
- Total number of pictures: Total number of pictures included in the test data set.

F1 Score:

F1 score is a metric that consider precision and recall for test accuracy. This measure gives relevant information about how the positive class is being predicted. The perfect model will have a value of 1 and the worst a value of 0.

$$F1\ score = 2 * \frac{precision * recall}{precision + recall}$$

Where:

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

RESEARCH TECHNIQUES

Image Classification:

Contextual image classification, a topic of pattern recognition in computer vision, is an approach of classification based on contextual information in images. "Contextual" means this approach is focusing on the relationship of the nearby pixels, which is also called neighborhood.

Feature Engineering:

Discovery analysis collects data and consolidates it into a single source that can be easily and instantly evaluated. Once your raw data is converted, follow the train of thought by drilling down into the data. When a trend is identified, it unearths the contributing factors.

Deep Learning:

Deep learning is part of a broader family of machine learning methods based on artificial neural networks with representation learning. Learning can be supervised, semi-supervised or unsupervised.

Predictive Analysis:

Predictive analysis helps to answer the question "What might happen?" by extracting information from existing data to determine patterns and then form assumptions of future outcomes and trends. With this, we will describe what is the best course of action.

DEVELOPMENT ENVIORNMENT

Python

Python is an interpreted, high-level, general-purpose programming language used for performing the statistical analysis. When applying the technique of Web Scraping, Python coding will scrap the internet for selected data.

Google Colab

Colab you can import an image dataset, train an image classifier on it, and evaluate the model, all in just a few lines of code. Colab notebooks execute code on Google's cloud servers, meaning you can leverage the power of Google hardware, including GPUs and TPUs. Colab is used extensively in the machine learning community.

TensorFlow

TensorFlow is a free and open-source software library for machine learning. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks. TensorFlow is a symbolic math library based on dataflow and differentiable programming.

Keras

Keras is an open-source library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library. Up until version 2.3 Keras supported multiple backends, including TensorFlow, Microsoft Cognitive Toolkit, R, Theano, and PlaidML.

ANALYSIS PROCESS

- Collect the data
    - Images collected from various sources in all sections.
- Clean the data
    - Clean Secure storage of cleaned datasets
- Perform Feature Engineering
- Develop the Data Preprocessing Pipeline
- Develop the Model Architecture
- Split the data into train, test, and validation data.
- Train Model using Training Data Set.
- Perform Validation using validation dataset
- Test Model
- Evaluate the model
- Form and document conclusions

IMPLEMENTATION PLAN

Item	Phase	Major Tasks / Milestones	Dates
1.	Project Organization	Setup	Oct 28, 2020
2.	Business Understanding & Problem Discovery	Statement of Work	Nov 06, 2020
3.	Data Acquisition & Understanding	Data Collection & EDA	Nov 23, 2020
4.	ML Modeling & Evaluation	Evaluate learning algorithms and Prototype model architecture	Nov 23, 2020
5.	Delivery & Acceptance	Development & deployment of software pipeline & Solution endpoint	Dec 18, 2020

Data Acquisition

Data Exploration

The input data for this project is the CelebA dataset6 , which contains 202,599 number of face images of various celebrities and 40 binary attribute annotations per image, some of the attributes are: male, bald, wearing lipstick, wavy or straight hair, etc. To this project, the selected attributes are: Smiling, Male and Young. For each of these attributes a binary classification model will be created. Images are in format *.jpg with dimension 178x218x3 RGB.

File name: 197126.jpg
- Not Smiling
- Female
- Young

File name: 193309.jpg
- Smiling
- Female
- Young

File name: 202506.jpg
- Smiling
- Male
- Young

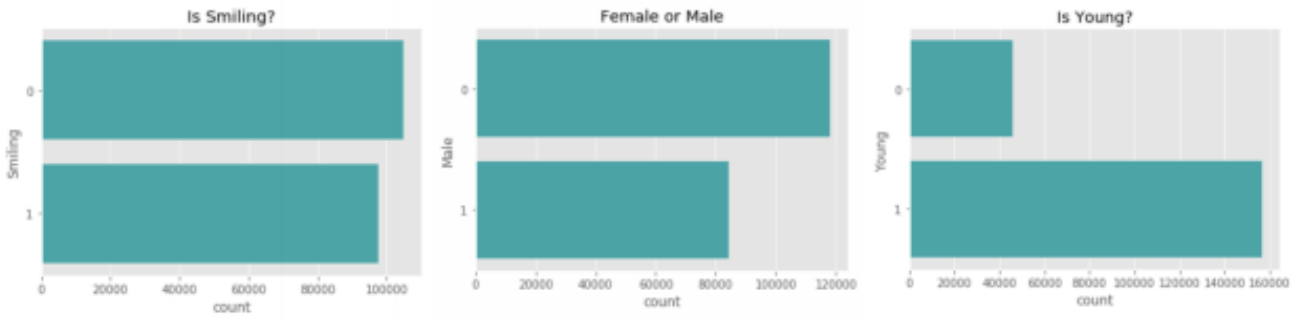
File name: 195516.jpg
- Not Smiling
- Male
- Not Young

In order to train the model, the data set include a recommended partition: Images 1-162770 are training, 162771-182637 are validation, 182638-202599 are testing. This recommendation will be used on this project to select images and create subsets.

Exploratory Visualization

The plots below show how is the distribution of the selected labels across the data set. In figure 1 the Smiling attribute is balanced in the data set, for the Male attribute, the chart shows there are more female celebrities than males and for the Young attribute, most of the images are of young celebrities. This analysis indicates that it will be necessary balance the data set for each model in order to train them. Otherwise, the performance of the model will be impacted7 . For the information needed for this project, there are no abnormalities, which means that every image has its label.

Distribution of labels across the data set



Algorithms and Techniques

The classifier is a Convolutional Neural Network for each model (one for each of the selected attributes: Smiling, Male and Young). The models will be based in the Inception-V3 model trained with the ImageNet dataset8 and using transfer learning to build the new classifiers. Also, Data Augmentation for image preprocessing will be used.

Data Augmentation

Data Augmentation allows to generate images with modifications to the original ones, this allows the model to learn from these variations (changing angle, size, and position), being able to predict better never seen images that could have the same variations. The parameters to be modified in this project for data augmentation are:

- rotation_range: Degree range for random rotations. o
- width_shift_range: Width shift range to move the image.
- height_shift_range: Height shift range to move the image.
- shear_range: Shear angle in counter-clockwise direction in degrees.
- zoom_range: Range for random zoom.
- horizontal_flip: flip images horizontally.

For more details about the available parameters: https://keras.io/preprocessing/image/. For Preprocessing parameters and example of the results in this project: See “Data Preprocessing” section.

Transfer Learning

The justification of this technique (transfer learning) is because of the fact that DNNs are computationally expensive to train, in order to get good result is necessary to train using millions of labeled images and machine equipped with GPUs, and this is what I Inception-V3 offers: a pre-trained 8 http://www.image-net.org 5 model that has learn from millions of images, being able to identify in lower layers features such as texture, colors, contrast, etc. and is able to transfer this knowledge to new model, where some layers can be removed in other to continue training the network to solve a new problem (classification problem in this case).

The parameters than can be tuned using CNNs transfer learning are:

- Number of layers: Number of layers to be used by the base model.
- Trainable layers: Layers that will be trained, the remaining ones are locked and weights
- unmodified after new training.
- Type of layers: Fully connected, convolutional or pooling.
- Activation functions: Function uses to calculate the output of a layer.
- Optimizer: Optimizer technique used to train9.
- Learning rate: Intensity in the model to learn during training.
- Momentum: in SGD (Stochastic gradient descent optimizer), parameter that takes into
- account previous steps to keep learning.
- Loss: Optimization score function10.
- Metrics: metrics to be evaluated by the model during training and testing.
- Epochs: Number of iterations over the entire data provided.

Reduce the number of samples

Due to computing resource limitations of the researcher to process all the images, the number of the images has been reduced. Basically, using a Mac Book Pro 2017 (2.9 GHz Intel Core i7, 16 GB RAM and 4 GB GPU), it will take days to train each model, stressing the machine and with risk to burn some pieces of hardware, as the CPUs reached over 100º Celsius, even using an external fan; GPU processing is not supported for MacOS by TensorFlow since release 1.2 (https://www.tensorflow.org/install/install_mac). Instead, the researcher is using a Windows 10 Machine with 2.9 GHz Intel Core i7, 16GB RAM, 2GB NVIDIA GeForce GTX 950M), which accelerate the process thanks to the use of GPU. Also, additional software has been installed to run TensorFlow on GPU, such as CUDA11. The guide to install it is in TensorFlow website12.

The number of images to be used by each model is:

- Training: 20000 images
- Validation: 5000 images
- Test: 5000 Images

Step 1: Data Exploration

We will be using the CelebA Dataset, which includes images of 178 x 218 px. Below is an example of how the pictures looks like.

In [10]:
```
# set variables
main_folder = 'C:/Users/tarun/Desktop/Artificial Intelligence/Algorithms/Capstone/Data/'
images_folder = main_folder + 'img_align_celeba'

EXAMPLE_PIC = images_folder + '/000506.jpg'

TRAINING_SAMPLES = 10000
VALIDATION_SAMPLES = 2000
TEST_SAMPLES = 2000
IMG_WIDTH = 178
IMG_HEIGHT = 218
BATCH_SIZE = 16
NUM_EPOCHS = 20

print(EXAMPLE_PIC)
```

```
C:/Users/tarun/Desktop/Artificial Intelligence/Algorithms/Capstone/Data/img_align_celeba/000506.jpg
```

Load the attributes of every picture

File: list_attr_celeba.csv

In [11]:
```
# import the data set that include the attribute for each picture
df_attr = pd.read_csv(main_folder + 'list_attr_celeba.csv')
df_attr.set_index('image_id', inplace=True)
df_attr.replace(to_replace=-1, value=0, inplace=True) #replace -1 by 0
df_attr.shape
```

Out[11]: (202599, 40)

List of the available attribute in the CelebA dataset

40 Attributes

In [12]:
```
# List of available attributes
for i, j in enumerate(df_attr.columns):
    print(i, j)
```

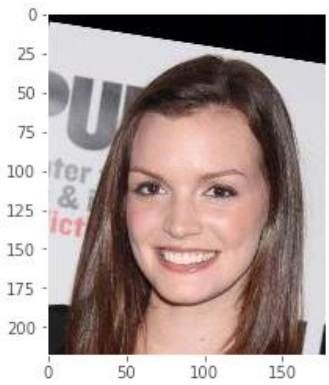
```
0 5_o_Clock_Shadow
1 Arched_Eyebrows
2 Attractive
3 Bags_Under_Eyes
4 Bald
5 Bangs
6 Big_Lips
7 Big_Nose
8 Black_Hair
9 Blond_Hair
10 Blurry
11 Brown_Hair
12 Bushy_Eyebrows
13 Chubby
14 Double_Chin
15 Eyeglasses
16 Goatee
17 Gray_Hair
18 Heavy_Makeup
19 High_Cheekbones
20 Male
21 Mouth_Slightly_Open
22 Mustache
23 Narrow_Eyes
24 No_Beard
25 Oval_Face
26 Pale_Skin
27 Pointy_Nose
28 Receding_Hairline
29 Rosy_Cheeks
30 Sideburns
31 Smiling
32 Straight_Hair
33 Wavy_Hair
34 Wearing_Earrings
35 Wearing_Hat
36 Wearing_Lipstick
37 Wearing_Necklace
38 Wearing_Necktie
39 Young
```

Example of a picture in CelebA dataset

178 x 218 px

[13]:
```
# plot picture and attributes
img = load_img(EXAMPLE_PIC)
plt.grid(False)
plt.imshow(img)
df_attr.loc[EXAMPLE_PIC.split('/')[-1]][['Smiling','Male','Young']] #some attributes
```

[13]:
```
Smiling   1
Male      0
Young     1
Name: 000506.jpg, dtype: int64
```



Data Preprocessing

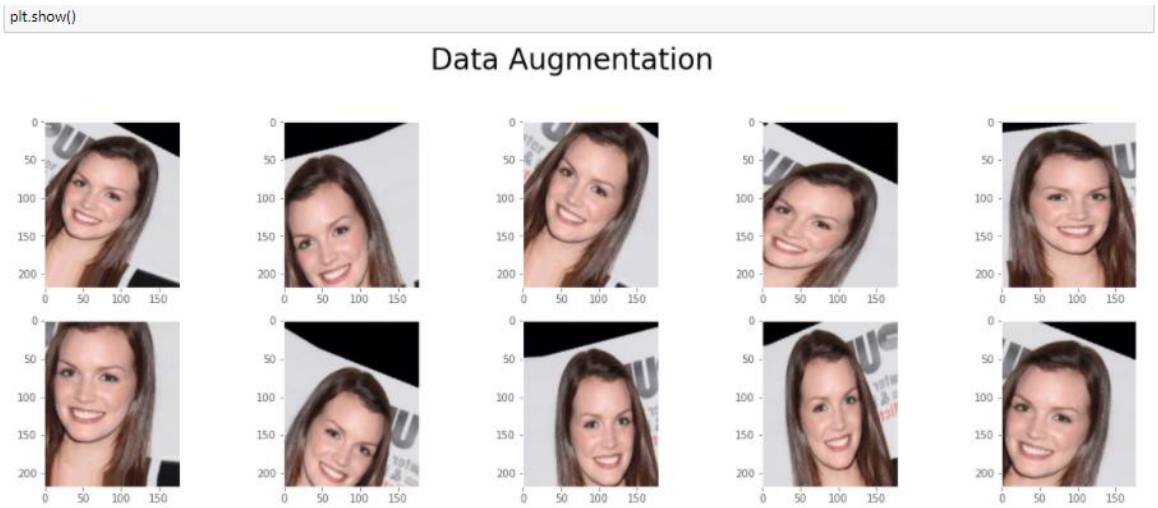
The steps for data preprocessing used on this project are: • Images to Arrays Inception-V3 trained model is based on TensorFlow, this requires as input a 4D NumPy array. Keras and TensorFlow offer this solution out of the box using the preprocess_input method15.

• Data Augmentation As explained in the Algorithms and Techniques section, Data Augmentation has been used to generate images with modifications to the original ones, this allows the model to learn from these variations (changing angle, size and position), being able to predict better never seen images that could have the same variations. Parameters has been tuned as below:

- rotation_range = 30
- width_shift_range = 0.2
- height_shift_range = 0.2
- shear_range = 0.2
- zoom_range = 0.2
- horizontal_flip = True

The result is a data generator of modified images that will be used as input to train the model.

Example of the result using Data Augmentation:



Split the Dataset into Train, Validation and Test

Step 2: Split Dataset into Training, Validation and Test

The recommended partitioning of images into training, validation, testing of the data set is:

- 1-162770 are training
- 162771-182637 are validation
- 182638-202599 are testing

The partition is in file **list_eval_partition.csv**

Due time execution, by now we will be using a reduced number of images:

- Training 20000 images
- Validation 5000 images
- Test 5000 Images

In [20]:
```
# Recomended partition
df_partition = pd.read_csv(main_folder + 'list_eval_partition.csv')
df_partition.head()
```

Out[20]:

	image_id	partition
0	000001.jpg	0
1	000002.jpg	0
2	000003.jpg	0
3	000004.jpg	0
4	000005.jpg	0

In [21]:
```
# display counter by partition
# 0 -> TRAINING
# 1 -> VALIDATION
# 2 -> TEST
df_partition['partition'].value_counts().sort_index()
```

Out[21]:
```
0   162770
1    19867
2    19962
Name: partition, dtype: int64
```

Join the partition and the attributes in the same data frame

In [22]:
```
# join the partition with the attributes
df_partition.set_index('image_id', inplace=True)
df_par_attr = df_partition.join(df_attr['Male'], how='inner')
df_par_attr.head()
```

Out[22]:

image_id	partition	Male
000001.jpg	0	0
000002.jpg	0	0
000003.jpg	0	1
000004.jpg	0	0
000005.jpg	0	0