Facial Attribute Classification

Amanuel Tesfaye & Wasay Qureshi

Dataset

- Celeb A Dataset
- 200,000 images of people's faces
- Centered and aligned

Visualizing the Data

- 202599 images, 40 attributes

```
202599
5_o_Clock_Shadow Arched_Eyebrows Attractive Bags_Under_Eyes Bald Bangs Big_Lips Big_Nose Black_Hair Blond_Hair Blurry Brown_Hair
Bushy_Eyebrows Chubby Double_Chin Eyeglasses Goatee Gray_Hair Heavy_Makeup High_Cheekbones Male Mouth_Slightly_Open Mustache Narrow_Eyes
No_Beard Oval_Face Pale_Skin Pointy_Nose Receding_Hairline Rosy_Cheeks Sideburns Smiling Straight_Hair Wavy_Hair Wearing_Earrings
Wearing_Hat Wearing_Lipstick Wearing_Necklace Wearing_Necktie Young
000001.jpg -1  1  1 -1 -1 -1 -1 -1 -1 -1 -1  1 -1 -1 -1 -1 -1 -1 -1  1  1 -1  1 -1 -1  1 -1 -1  1 -1 -1 -1  1  1 -1  1 -1  1 -1 -1  1
000002.jpg -1 -1 -1  1 -1 -1 -1  1 -1 -1 -1  1 -1 -1 -1 -1 -1 -1 -1 -1  1 -1  1 -1 -1  1 -1 -1 -1 -1 -1 -1 -1  1 -1 -1 -1 -1 -1 -1 -1  1
000003.jpg -1 -1 -1 -1 -1 -1  1 -1 -1 -1  1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  1 -1 -1  1  1 -1 -1  1 -1 -1 -1 -1 -1 -1  1 -1 -1 -1 -1 -1 -1  1
000004.jpg -1 -1  1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  1 -1 -1  1 -1 -1 -1 -1  1 -1  1 -1  1  1 -1  1
000005.jpg -1  1  1 -1 -1 -1  1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  1 -1 -1 -1 -1  1  1 -1 -1  1 -1 -1 -1 -1 -1 -1 -1 -1 -1  1 -1 -1  1
000006.jpg -1  1  1 -1 -1 -1  1 -1 -1 -1 -1  1 -1 -1 -1 -1 -1 -1 -1  1 -1 -1  1 -1 -1  1 -1 -1 -1 -1 -1 -1 -1 -1  1  1 -1  1 -1 -1  1
000007.jpg 1 -1  1  1 -1 -1  1  1  1 -1 -1 -1  1 -1 -1 -1 -1 -1 -1 -1 -1  1 -1 -1 -1  1 -1 -1  1 -1 -1 -1 -1  1 -1 -1 -1 -1 -1 -1  1
000008.jpg 1  1 -1  1 -1 -1  1 -1  1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  1 -1 -1 -1  1 -1 -1  1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  1
000009.jpg -1  1  1 -1 -1  1  1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  1  1 -1  1 -1 -1  1  1 -1  1 -1  1 -1  1 -1 -1  1 -1  1 -1 -1  1
000010.jpg -1 -1  1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  1  1 -1 -1 -1 -1  1 -1 -1 -1 -1 -1 -1 -1 -1  1 -1 -1  1 -1 -1  1
```

Visualizing our Data - Continued

- Not Attractive

- Attractive

Preparing Our Data

- Generator Object

```python
# Creating a Generator object with unlimited data generation
def image_batch_generator(image_dir, filenames, labels, batch_size):
    num_batches = len(filenames) // batch_size

    # Define the target image size
    target_size = (64, 64)
    # Define the number of channels
    num_channels = 3

    while True:
        for batch_index in range(num_batches):
            batch_start = batch_index * batch_size
            batch_end = batch_start + batch_size
            batch_files = filenames[batch_start:batch_end]

            images = np.empty((batch_size, *target_size, num_channels), dtype=np.float32
            batch_labels = labels[batch_start:batch_end]

            for i, file_name in enumerate(batch_files):
                image_path = os.path.join(image_dir, file_name)
                img = Image.open(image_path).resize(target_size)
                # Convert the image to a NumPy array
                img_array = np.array(img, dtype=np.float32)
                # Normalize the pixel values to the range [0, 1]
                img_array /= 255.0
                # Add the preprocessed image to the batch
                images[i] = img_array

            yield np.array(images), np.array(batch_labels)
```

Reduced Dataset

```python
# Creating numpy array of size (num_images, height, width, channels) - takes about 10 mi

# Define the target image size
target_size = (64, 64)

# Define the number of channels
num_channels = 3

# Create an empty array to hold the images
x_reduced = np.empty((num_images_reduced, *target_size, num_channels), dtype=np.float32)

# Iterate over the image filenames
for i, img_filename in enumerate(img_filenames_reduced):
    # Load the image and resize it
    img_path = os.path.join(img_folder, img_filename)
    img = Image.open(img_path).resize(target_size)

    # Convert the image to a NumPy array
    img_array = np.array(img, dtype=np.float32)

    # Normalize the pixel values to the range [0, 1]
    img_array /= 255.0

    # Add the image to the array
    x_reduced[i] = img_array

    if (i%10000 == 0):
      print("Progress: " + str(((i)/num_images_reduced * 100)) + " %")
```

Model Performance

- CNN model architecture

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 62, 62, 32)        896

 max_pooling2d (MaxPooling2D  (None, 31, 31, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 29, 29, 64)        18496

 max_pooling2d_1 (MaxPooling  (None, 14, 14, 64)       0
 2D)

 conv2d_2 (Conv2D)           (None, 12, 12, 128)       73856

 max_pooling2d_2 (MaxPooling  (None, 6, 6, 128)        0
 2D)

 flatten (Flatten)           (None, 4608)              0

 dense (Dense)               (None, 128)               589952

 dense_1 (Dense)             (None, 1)                 129

=================================================================
Total params: 683,329
Trainable params: 683,329
Non-trainable params: 0
```

Accuracy of our model

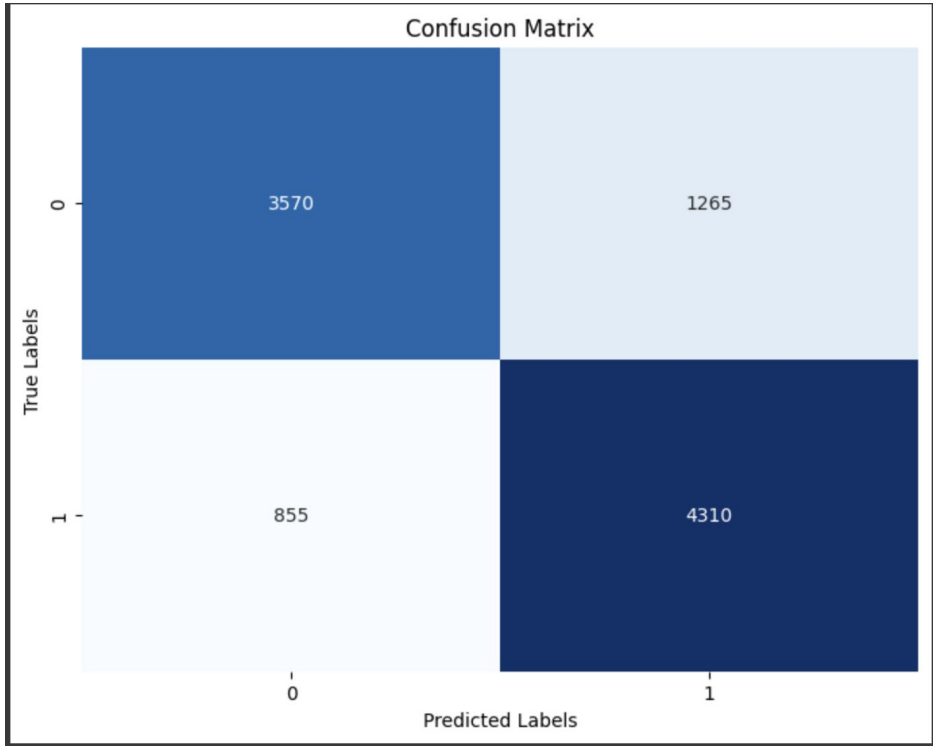
Using Generator object

```
Epoch 1/3
20259/20259 [==============================] - 292s 14ms/step - loss: 0.4810 - accuracy: 0.7645 - val_loss: 0.4491 - val_accuracy: 0.7809
Epoch 2/3
20259/20259 [==============================] - 279s 14ms/step - loss: 0.4458 - accuracy: 0.7869 - val_loss: 0.4417 - val_accuracy: 0.7873
Epoch 3/3
20259/20259 [==============================] - 279s 14ms/step - loss: 0.4343 - accuracy: 0.7936 - val_loss: 0.4346 - val_accuracy: 0.7892
```

Using Reduced Dataset

```
Epoch 8/10
1000/1000 [==============================] - 5s 5ms/step - loss: 0.3328 - accuracy: 0.8432 - val_loss: 0.4697 - val_accuracy: 0.7839
Epoch 9/10
1000/1000 [==============================] - 5s 5ms/step - loss: 0.3157 - accuracy: 0.8509 - val_loss: 0.5059 - val_accuracy: 0.7814
Epoch 10/10
1000/1000 [==============================] - 5s 5ms/step - loss: 0.2966 - accuracy: 0.8633 - val_loss: 0.5329 - val_accuracy: 0.7750
```

Evaluation/Results



Extensions

- Trying other models good at face recognition (e.g VGG Face)
- Predicting other facial attributes (facial hair, makeup, glasses...)
- Automating Dataset download