Final Report: Facial Recognition for Age and Gender estimation

Team Members:

Zharas Nurmukhammed

Arman Kozhakhmet

1. Introduction

Facial recognition for age and gender estimation is an essential task in various applications such as biometrics, personalized marketing, and demographic analysis. This project compares two different approaches:

1. **Combined Model**: A single Convolutional Neural Network (CNN) with dual outputs for gender classification and age estimation.
2. **Separate Models**: Independent CNNs optimized specifically for each task.

The goal is to evaluate the trade-offs between computational efficiency and task-specific accuracy.

---

2. Dataset

Source: UTKFace dataset.

Details:

- Includes labeled images with attributes for age and gender.
- Diversity in ethnicity, lighting, and age ranges.

Preprocessing:

- Resized all images to 128×128 \model-2: 200x200.
- Normalized pixel values to the range [0, 1].
- Applied data augmentation (e.g., flips, shifts) to enhance robustness.

---

3. Methodology

Model 1: Combined Model

A single CNN handles both tasks simultaneously, sharing feature extraction layers. Outputs:

- Gender: Binary classification using sigmoid activation.
- Age: Regression using ReLU activation.

```
inputs = Input((input_shape))
conv_1 = Conv2D(32, kernel_size=(3, 3), activation='relu')(inputs)
max_1 = MaxPooling2D(pool_size=(2, 2))(conv_1)
conv_2 = Conv2D(64, kernel_size=(3, 3), activation='relu')(max_1)
max_2 = MaxPooling2D(pool_size=(2, 2))(conv_2)
conv_3 = Conv2D(128, kernel_size=(3, 3), activation='relu')(max_2)
max_3 = MaxPooling2D(pool_size=(2, 2))(conv_3)
conv_4 = Conv2D(256, kernel_size=(3, 3), activation='relu')(max_3)
max_4 = MaxPooling2D(pool_size=(2, 2))(conv_4)

flatten = Flatten()(max_4)

# fully connected layers
dense_1 = Dense(256, activation='relu')(flatten)
dense_2 = Dense(256, activation='relu')(flatten)

dropout_1 = Dropout(0.3)(dense_1)
dropout_2 = Dropout(0.3)(dense_2)

output_1 = Dense(1, activation='sigmoid', name='gender_out')(dropout_1)
output_2 = Dense(1, activation='relu', name='age_out')(dropout_2)

model = Model(inputs=[inputs], outputs=[output_1, output_2])

model.compile(loss=['binary_crossentropy', 'mae'],
              optimizer='adam', metrics=[['accuracy'], ['mae']])
```

Model 2: Separate Models

Two independent CNNs were built, each optimized for one task:

- **Age Estimation Model**: Outputs continuous age values.
- **Gender Classification Model**: Outputs binary gender values.

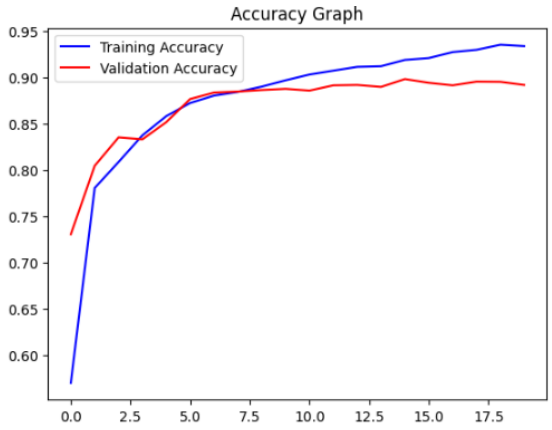
```
agemodel = Sequential()
agemodel.add(Conv2D(32, (3,3), activation='relu', input_shape=(200, 200, 3)))
agemodel.add(MaxPooling2D((2,2)))
agemodel.add(Conv2D(64, (3,3), activation='relu'))
agemodel.add(MaxPooling2D((2,2)))
agemodel.add(Conv2D(128, (3,3), activation='relu'))
agemodel.add(MaxPooling2D((2,2)))
agemodel.add(Flatten())
agemodel.add(Dense(64, activation='relu'))
agemodel.add(Dropout(0.5))
agemodel.add(Dense(1, activation='relu'))

agemodel.compile(loss='mean_squared_error',
            optimizer=optimizers.Adam(learning_rate=0.0001))

genmodel = Sequential()
genmodel.add(Conv2D(32, (3,3), activation='relu', input_shape=(200, 200, 3)))
genmodel.add(MaxPooling2D((2,2)))
genmodel.add(Conv2D(64, (3,3), activation='relu'))
genmodel.add(MaxPooling2D((2,2)))
genmodel.add(Conv2D(128, (3,3), activation='relu'))
genmodel.add(MaxPooling2D((2,2)))
genmodel.add(Flatten())
genmodel.add(Dense(64, activation='relu'))
genmodel.add(Dropout(0.5))
genmodel.add(Dense(1, activation='sigmoid'))

genmodel.compile(loss='binary_crossentropy',
            optimizer=optimizers.Adam(learning_rate=0.0001),
            metrics=['accuracy'])
```

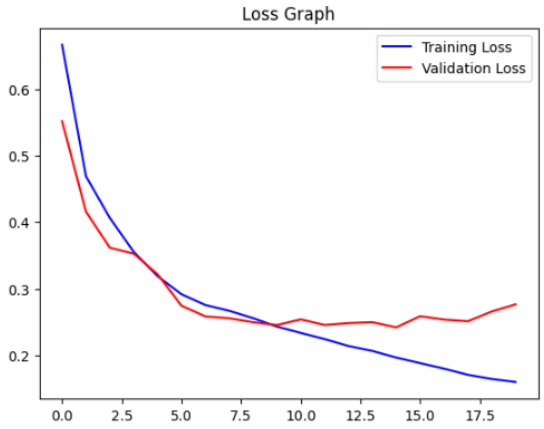
4. Metrics and Results

Metric	Combined Model	Separate Models
Gender Accuracy	~93%	~87.8%
Age Estimation (MAE)	~7 years	~17 years
Training Time	Faster	Slower
Inference Time	Faster	Slower

5. Visualizations

Model 2





6. Deployment

```python
def process_and_predict(file):
    im = Image.open(file)
    width, height = im.size
    if width == height:
        im = im.resize((200,200), Image.Resampling.LANCZOS)
    else:
        if width > height:
            left = width/2 - height/2
            right = width/2 + height/2
            top = 0
            bottom = height
            im = im.crop((left,top,right,bottom))
            im = im.resize((200,200), Image.Resampling.LANCZOS)
        else:
            left = 0
            right = width
            top = 0
            bottom = width
            im = im.crop((left,top,right,bottom))
            im = im.resize((200,200), Image.Resampling.LANCZOS)

    ar = np.asarray(im)
    ar = ar.astype('float32')
    ar /= 255.0
    ar = ar.reshape(-1, 200, 200, 3)

    age = agemodel.predict(ar)
    gender = np.round(genmodel.predict(ar))
    if gender == 0:
        gender = 'male'
    elif gender == 1:
        gender = 'female'

    print('Age:', int(age), '\n Gender:', gender)
    return im.resize((300,300), Image.Resampling.LANCZOS)
```

```
Age: 17
 Gender: male
C:\Users\zhara\AppData\Local\Temp\ipykernel_20596\2624513457.py:34: Deprecatio
ement from your array before performing this operation. (Deprecated NumPy 1.25
  print('Age:', int(age), '\n Gender:', gender)
```

54]: