Gender classification

2022/5/25
Data science fundamentals

Saud Almoter 439101963
Musab Alghamdi 438102298

Gender classification models

In this report we will compare a deep learning model given to us by the instructor with a modified version of the given model first, then we will compare many pretrained model with our modified model.

https://www.kaggle.com/datasets/ashwingupta3012/male-and-female-faces-dataset

the above link is the Kaggle dataset that is use in this project.

After running the instructor model, we got %99.85 accuracy.

```
OG_acc = OG_history.history['accuracy']
print(" This is the accuracy of the original or given model {:.2f} % " .format(
```

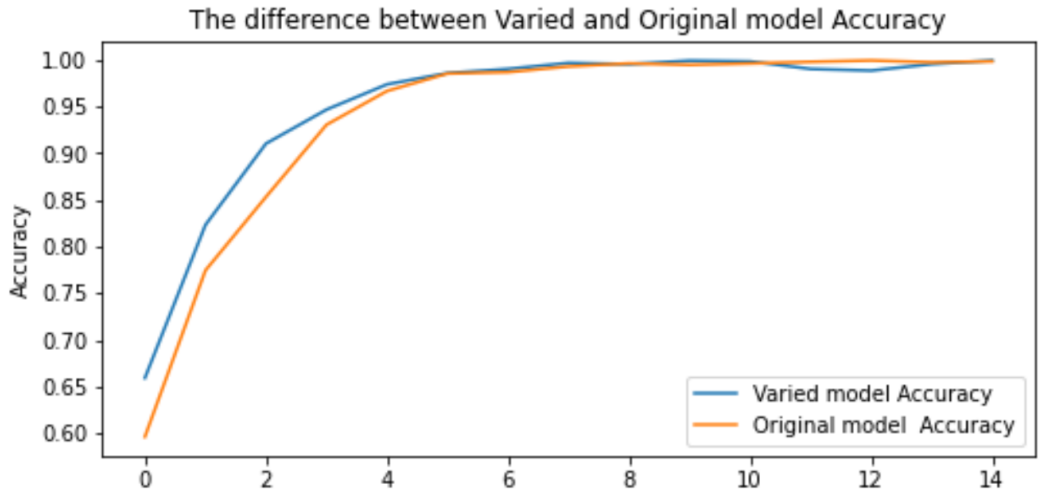
```
 This is the accuracy of the original or given model 99.85 %
```

And after modifying the model we got %99.95 accuracy.

```
acc = history.history['accuracy']
print(" The accuracy of the varied model is {:.2f} % " .format(acc[-1]*100))
```

```
 The accuracy of the varied model is 99.95 %
```

Here we show the difference between the two model on each epoch. Both models are trained with 15 epochs



Gender classification models

In this part on the report, we will compare our model with pretrained models that we got from TensorFlow hub and TensorFlow keras the models we used are:

- MobileNetV2
- ResNet50V2
- Xception
- VGG16
- InceptionV3

First, we will start with the MobileNetV2 pretrained model and we got %91.81 accuracy.

```
[ ]  MB_acc = historyOfMN.history['accuracy']
     print(" This is the accuracy of the original or given model {:.2f} % " .format(M
```

```
 This is the accuracy of the original or given model 91.81 %
```

Then we tried the Inception pretrained model and we got %79.26 accuracy.

```
[ ]  inception_acc = historyOfIN.history['accuracy']
     print(" This is the accuracy of the original or given model {:.2f} % " .format(
```

```
 This is the accuracy of the original or given model 79.26 %
```

After that we run the ResNet50 pertained model, and we got % 82.31 accuracy.

```
[ ]  RN_acc = historyOfRN.history['accuracy']
     print(" This is the accuracy of the original or given model {:.2f} % " .format(
```

```
 This is the accuracy of the original or given model 82.31 %
```

These three above models we utilis TensorFlow Hub and we only needed to specify the input shape and add a Dence layer and specify the number of classes which in our case is 2.

```
resnet_v2_50_classlifer = tf.keras.Sequential(
          hub.KerasLayer("https://tfhub.dev/google/imagenet/resnet_v2_50/featur
          )
RNclasslicer = tf.keras.Sequential([
              resnet_v2_50_classlifer,
              tf.keras.layers.Dense(num_classes)
])
RNclasslicer.summary()
```

After that we compile the model with an Adam optimizer and Sparse Categorical Cross entropy loss function

```
[ ]  RNclasslicer.compile(
         optimizer = 'adam',
         loss = tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
         metrics=['accuracy']
     )
     historyOfRN = RNclasslicer.fit(train_ds,validation_data=val_ds,epochs=15)
```

This is the same process we runed for the above models.

For the below models we used Keras

```
     VGG16Test = tf.keras.applications.VGG16(input_shape = (224,224,3),
                                             include_top = False,
                                             weights = 'imagenet',
                                             classes = num_classes)
     new_model = Sequential()
     new_model.add(VGG16Test)
     new_model.add(layers.Dropout(0.2))
     new_model.add(layers.Flatten())
     new_model.add(tf.keras.layers.Dense(128, activation = 'relu'))
     new_model.add(tf.keras.layers.Dense(num_classes))
     new_model.trainable = False
     new_model.summary()
```

After we made an instance of the VGG16 model and enter the input shape and classes and used the pretrained weights ‘imagenet’ we add a drop out layer then a flatten layer an add the output layer “Dence layer”, also we set the trainable status to false which means that all hidden layers will note be touched only the input and output layers. Then we complied the model same as the above models.

```
new_model.compile(
    optimizer='adam',
    loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True
    metrics=['accuracy']
)
historyOfV = new_model.fit(train_ds,validation_data=val_ds,epochs=15)
```

After fitting the model, we got % 50.34 accuracy.

```
[ ]  VGG16_acc = historyOfV.history['accuracy']
     print(" This is the accuracy of the original or given model {:.2f} % " .format(V
```

```
 This is the accuracy of the original or given model 50.34 %
```

And for the last model Xception we run the same process of VGG16 model and got %50.00 accuracy.

```
[ ]  X_acc = historyOfX.history['accuracy']
     print(" This is the accuracy of the original or given model {:.2f} % " .format()
```

```
 This is the accuracy of the original or given model 50.00 %
```

The below plot will compare our model with all pretrained models mentioned in this project.

All models are fitted with the same data and same number of epachs and loss function and optimizer.