Muhammad Risalah Naufal

21051214008

S1 Sistem Informasi 2021

PRAKTIKUM: IMAGE DETECTION

STUDI KASUS GENDER DETECTION

A. PENDAHULUAN

Dalam era perkembangan teknologi informasi dan kecerdasan buatan, penggunaan teknik transfer learning dalam bidang pengolahan citra semakin menjadi sorotan. Transfer learning memungkinkan kita untuk memanfaatkan pengetahuan yang telah diperoleh oleh model yang sudah dilatih pada tugas tertentu dan mengadaptasikannya untuk menyelesaikan tugas serupa yang mungkin memiliki dataset yang lebih kecil.

Pada sesi coding kali ini, fokus penelitian kita adalah penerapan transfer learning dalam konteks klasifikasi gambar untuk membedakan antara gambar Pria dan Wanita. Klasifikasi jenis kelamin melalui gambar merupakan salah satu tantangan dalam pengolahan citra yang memiliki dampak signifikan dalam berbagai aplikasi, seperti identifikasi wajah otomatis, pemasaran berbasis gambar, dan pemahaman konten multimedia.

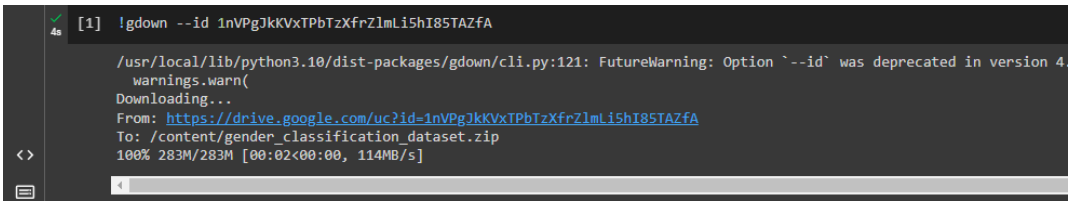
Metode transfer learning memungkinkan kita menggunakan arsitektur deep learning yang sudah terbukti efektif dalam mempelajari fitur-fitur umum pada dataset yang besar. Dengan menerapkan model yang telah dilatih pada dataset besar, kita dapat menghindari kendala terkait kekurangan data yang sering muncul dalam proyek-proyek pengolahan citra kecil.

Pada eksperimen ini, kita akan memanfaatkan pretrained model yang telah berhasil dalam tugas pengenalan gambar umum dan menyelaraskannya untuk tugas klasifikasi khusus kita. Dengan demikian, diharapkan bahwa model yang dihasilkan mampu mengidentifikasi ciri-ciri yang membedakan gambar Pria dan Wanita dengan akurasi yang baik.

Langkah-langkah praktis yang akan dijelaskan dalam sesi coding melibatkan pengenalan konsep-konsep dasar deep learning, transfer learning, dan implementasi teknik-teknik tersebut menggunakan framework tertentu, seperti TensorFlow atau PyTorch. Dengan demikian, diharapkan pembaca dapat memahami dan mengaplikasikan metode-metode ini dalam konteks proyek-proyek klasifikasi gambar yang serupa.

B. PEMBAHASAN

1. Download dan Unzio File



Download File Menggunakan Google Drive dengan Google Colab

Kode di atas menggunakan perintah **!gdown** untuk mengunduh file dari Google Drive menggunakan Google Colab. Parameter **--id** digunakan untuk menentukan ID unik file yang akan diunduh. Dengan demikian, file dengan ID yang ditentukan akan diunduh dan tersedia untuk digunakan dalam lingkungan Google Colab.

```
[2]  import os
     import zipfile

     # unzip dataset yang masih berbentuk file zip
     zip_ref = zipfile.ZipFile("./gender_classification_dataset.zip", 'r')
     zip_ref.extractall("./")
     zip_ref.close()
```

Mengekstrak Dataset dari File ZIP dalam Python

Kode di atas menggunakan modul Python **zipfile** untuk mengekstrak isi dari file ZIP yang bernama "gender_classification_dataset.zip". Setelah mengekstrak, dataset akan tersedia dalam direktori yang sama dengan file ZIP. Dengan langkah-langkah ini, dataset yang mungkin awalnya terkompres dalam format ZIP dapat diakses dan digunakan untuk analisis atau pelatihan model dalam pengembangan proyek.

```
[3]  # menentukan path dari dataset untuk training dan dataset untuk validasi
     train_dir = '/content/Training'
     validation_dir = '/content/Validation'

     train_data_male = os.listdir(train_dir + '/male/')
     train_data_female = os.listdir(train_dir + '/female/')
```

Menentukan Path dan Membaca Nama File dalam Dataset untuk Training

Kode di atas bertujuan untuk menentukan path (jalur) dari dataset untuk training dan validasi, serta membaca nama file dalam dataset tersebut. Variabel **train_dir** dan **validation_dir** menunjukkan path dari direktori dataset untuk training dan validasi. Selanjutnya, dua variabel **train_data_male** dan **train_data_female** digunakan untuk menyimpan daftar nama file dalam direktori khusus untuk gambar pria dan wanita dalam dataset training. Langkah ini merupakan langkah awal dalam persiapan data sebelum dilakukan proses training model.

```
[4]  %matplotlib inline

     import matplotlib.pyplot as plt
     import matplotlib.image as mpimg

     # Parameters for our graph; we'll output images in a 4x4 configuration
     nrows = 4
     ncols = 4

     fig = plt.gcf()
     fig.set_size_inches(ncols * 4, nrows * 2)

     next_male_pic = train_data_male[:8]
     next_female_pic = train_data_female[:8]

     for i, img_path in enumerate(next_male_pic):
       # Set up subplot; subplot indices start at 1
       sp = plt.subplot(2, 4, i + 1)
       sp.axis('Off') # Don't show axes (or gridlines)

       img = mpimg.imread(train_dir + '/male/' + img_path)
       plt.imshow(img)
     plt.show()

     fig = plt.gcf()
     fig.set_size_inches(ncols * 4, nrows * 2)

     for i, img_path in enumerate(next_female_pic):
       # Set up subplot; subplot indices start at 1
       sp = plt.subplot(2, 4, i + 1)
       sp.axis('Off') # Don't show axes (or gridlines)

       img = mpimg.imread(train_dir + '/female/' + img_path)
       plt.imshow(img)
     plt.show()
```

Visualisasi Sampel Gambar dari Dataset Training

Kode di atas digunakan untuk visualisasi sampel gambar dari dataset training. Pertama, gambar-gambar dari kategori pria (**next_male_pic**) ditampilkan dalam layout 4x4, dan kemudian gambar-gambar dari kategori wanita (**next_female_pic**) ditampilkan dalam layout yang sama. Visualisasi ini membantu dalam memahami struktur data dan memverifikasi bahwa gambar-gambar dalam dataset telah dimuat dengan benar sebelum dilakukan proses training model. Fungsi **imshow** dari modul **matplotlib.image** digunakan untuk menampilkan gambar pada setiap subplot.

2. Pre-Processing

```
[5]  from tensorflow.keras.preprocessing.image import ImageDataGenerator
     import tensorflow as tf

     # melakukan image augmentation pada dataset untuk menambah variasi data
     train_datagen = ImageDataGenerator(rescale = 1./255.,
                                        rotation_range = 40,
                                        width_shift_range = 0.2,
                                        height_shift_range = 0.2,
                                        shear_range = 0.2,
                                        zoom_range = 0.2,
                                        horizontal_flip = True,)
                                        #dtype=tf.float32

     test_datagen = ImageDataGenerator( rescale = 1.0/255.,)
                                        #dtype=tf.float32)

     # melakukan data generator untuk membaca dataset training di setiap label
     train_generator = train_datagen.flow_from_directory(train_dir,
                                                         batch_size = 32,
                                                         class_mode = 'binary',
                                                         target_size = (224, 224))

     # # melakukan data generator untuk membaca dataset testing di setiap label
     validation_generator =  test_datagen.flow_from_directory( validation_dir,
                                                               batch_size  = 32,
                                                               class_mode  = 'binary',
                                                               target_size = (224, 224))

     Found 47009 images belonging to 2 classes.
     Found 11649 images belonging to 2 classes.
```

“Image Augmentation dan Data Generator untuk Training dan Validation”

Kode di atas menggunakan TensorFlow dan modul ImageDataGenerator untuk melakukan augmentasi gambar pada dataset training. Image augmentation dilakukan untuk menambah variasi data dan membantu meningkatkan performa generalisasi model. Augmentasi termasuk rotasi, pergeseran lebar dan tinggi, shear, zoom, dan flip horizontal. Selanjutnya, data generator dibuat untuk membaca dataset training dan validation dari direktori yang telah ditentukan. Batch size ditetapkan sebagai 32, class_mode disetel sebagai 'binary', dan ukuran target gambar adalah 224x224 piksel. Data generator ini akan digunakan saat melatih dan menguji model untuk klasifikasi jenis kelamin berdasarkan gambar.

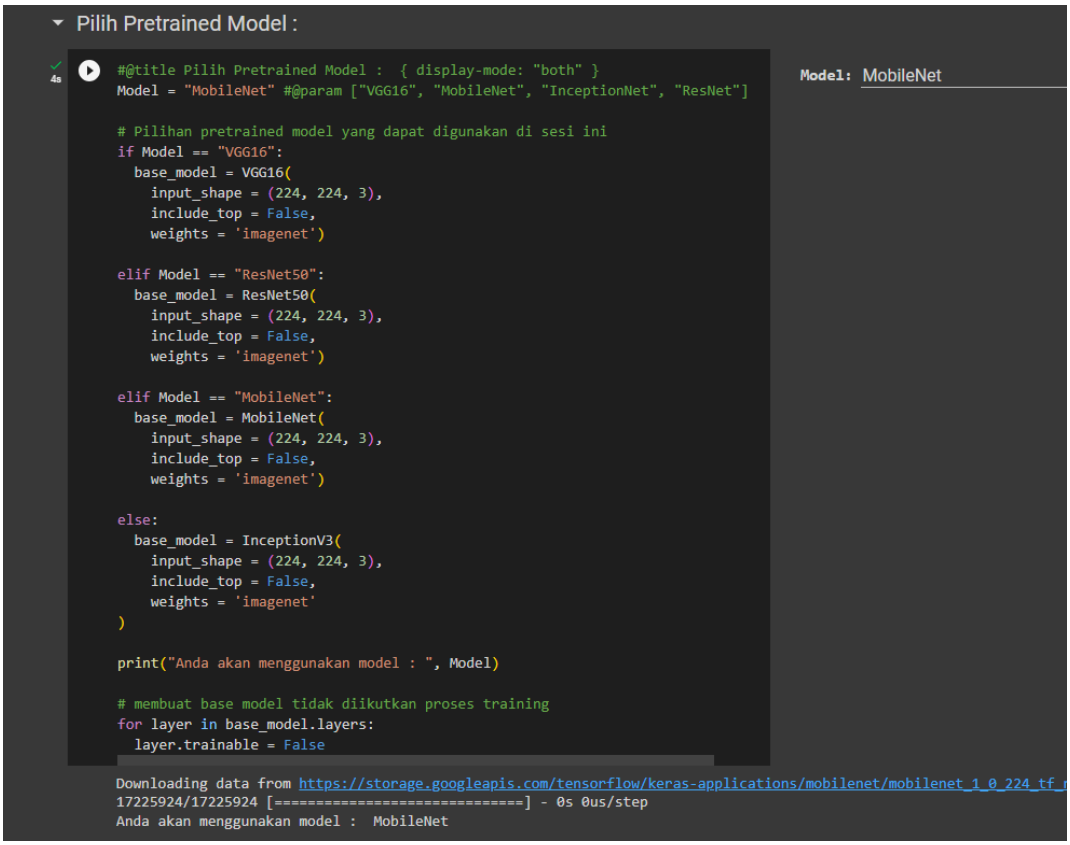
3. Modeling

```
[6]  from tensorflow.keras.applications.vgg16 import VGG16
     from tensorflow.keras.applications.resnet50 import ResNet50
     from tensorflow.keras.applications.inception_v3 import InceptionV3
     from tensorflow.keras.applications.mobilenet import MobileNet
     from tensorflow.keras.utils import plot_model
```

Importing Pretrained Convolutional Neural Network (CNN) Architectures for Image Classification

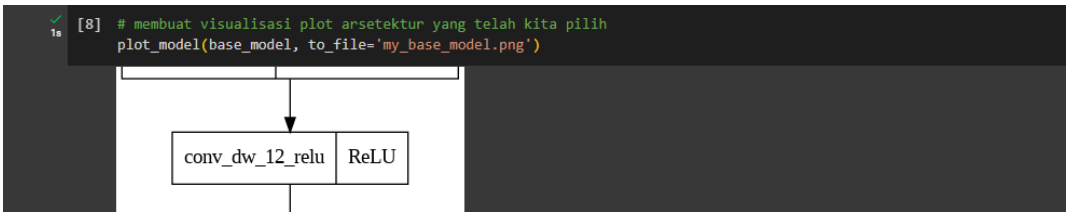
Kode di atas menggunakan TensorFlow dan modul **tensorflow.keras.applications** untuk mengimpor arsitektur Convolutional Neural Network (CNN) yang telah dilatih sebelumnya, yaitu VGG16, ResNet50, InceptionV3, dan MobileNet. Arsitektur-arsitektur ini telah terbukti efektif dalam tugas-tugas pengenalan gambar umum. Selanjutnya, modul **tensorflow.keras.utils** digunakan untuk mengimpor fungsi **plot_model**, yang memungkinkan visualisasi grafik arsitektur model. Kode ini mempersiapkan dasar untuk

menggunakan arsitektur-arsitektur ini sebagai bagian dari model klasifikasi gambar yang akan dibangun.



Pemilihan dan Konfigurasi Pretrained Model untuk Transfer Learning

Kode di atas memungkinkan pengguna untuk memilih dan mengkonfigurasi model Convolutional Neural Network (CNN) yang telah dilatih sebelumnya untuk digunakan dalam transfer learning. Pengguna dapat memilih model dari opsi VGG16, MobileNet, InceptionNet, atau ResNet. Setelah pemilihan, model tersebut diimpor dengan konfigurasi yang sesuai menggunakan modul **tensorflow.keras.applications**. Selanjutnya, parameter model seperti input shape, penggunaan layer teratas (**include_top**), dan bobot (**weights**) diatur sesuai kebutuhan. Model yang dipilih kemudian diinisialisasi, dan semua layer pada model tersebut diatur agar tidak mengalami proses training (freeze) selama transfer learning. Konfigurasi ini memungkinkan pengguna untuk menggunakan fitur-fitur ekstraktor yang telah dipelajari pada dataset besar sebelumnya untuk tugas klasifikasi gambar khusus yang sedang dihadapi.



Visualisasi Arsitektur Pretrained Model untuk Transfer Learning

Kode di atas menggunakan fungsi **plot_model** dari modul **tensorflow.keras.utils** untuk membuat visualisasi plot arsitektur dari model Convolutional Neural Network (CNN) yang telah dipilih sebelumnya. Visualisasi ini disimpan dalam file gambar dengan nama "my_base_model.png". Proses ini membantu pengguna untuk secara grafis

memahami struktur dan hubungan antar-layer dalam arsitektur model yang akan digunakan dalam transfer learning.

```
[9]  # kita juga bisa menggunakan .summary() untuk menampilkan arsitektur kita
     base_model.summary()

 conv_pw_10_bn (BatchNormal  (None, 14, 14, 512)       2048
 ization)

 conv_pw_10_relu (ReLU)      (None, 14, 14, 512)       0

 conv_dw_11 (DepthwiseConv2  (None, 14, 14, 512)       4608
 D)
```

Menampilkan Ringkasan (Summary) Arsitektur Pretrained Model

Kode di atas menggunakan fungsi **.summary()** pada objek **base_model** untuk menampilkan ringkasan arsitektur dari model Convolutional Neural Network (CNN) yang telah dipilih. Ringkasan ini mencakup informasi tentang jumlah parameter, ukuran output setiap layer, dan jumlah parameter yang dapat di-train atau di-freeze. Pemanggilan fungsi **.summary()** membantu pengguna untuk dengan cepat memeriksa dan memahami struktur model serta melihat jumlah parameter yang akan terlibat selama proses training.

```
[10] import tensorflow as tf


     x = tf.keras.layers.GlobalAveragePooling2D()(base_model.output)
     x = tf.keras.layers.Dense(64, activation='relu')(x)
     x = tf.keras.layers.Dense(32, activation='relu')(x)
     x = tf.keras.layers.Dropout(0.15)(x)
     outputs = tf.keras.layers.Dense(1, activation='sigmoid')(x)

     model = tf.keras.Model(inputs=base_model.input, outputs=outputs)

[11] model.summary()

 conv_dw_11_relu (ReLU)      (None, 14, 14, 512)       0

 conv_pw_11 (Conv2D)         (None, 14, 14, 512)       262144

 conv_pw_11_bn (BatchNormal  (None, 14, 14, 512)       2048
 ization)

 dropout (Dropout)           (None, 32)                0

 dense_2 (Dense)             (None, 1)                 33

=================================================================
Total params: 3296577 (12.58 MB)
Trainable params: 67713 (264.50 KB)
Non-trainable params: 3228864 (12.32 MB)
_________________________________________________________________
```

Membangun Model Klasifikasi dengan Menggunakan Pretrained Model untuk Transfer Learning

Kode di atas menggunakan TensorFlow untuk membangun model klasifikasi. Model ini memanfaatkan pretrained model yang telah dipilih sebelumnya (diwakili oleh **base_model**) sebagai ekstraktor fitur. Kemudian, lapisan-lapisan tambahan ditambahkan di atas pretrained model untuk menyesuaikan model dengan tugas klasifikasi yang spesifik. Proses ini melibatkan Global Average Pooling, beberapa lapisan Dense dengan fungsi aktivasi ReLU, sebuah lapisan Dropout untuk mengurangi overfitting, dan sebuah lapisan Dense output dengan aktivasi sigmoid untuk masalah klasifikasi biner. Model ini diinisialisasi sebagai objek **model** dengan menggunakan **tf.keras.Model** dengan input dari pretrained model dan output dari lapisan-lapisan yang baru ditambahkan.

```
[12] model.compile(
         optimizer=tf.keras.optimizers.Adam(learning_rate=1e-3),
         loss='binary_crossentropy',
         metrics=['accuracy']
     )

[13] history = model.fit(train_generator, validation_data=validation_generator,
                         steps_per_epoch = 60, epochs = 10)

     Epoch 1/10
     60/60 [==============================] - 56s 732ms/step - loss: 0.5756 - accuracy: 0.7000 - val_loss: 0.4094 - val_
     Epoch 2/10
     60/60 [==============================] - 42s 708ms/step - loss: 0.3796 - accuracy: 0.8453 - val_loss: 0.3736 - val_
     Epoch 3/10
     60/60 [==============================] - 42s 712ms/step - loss: 0.3611 - accuracy: 0.8469 - val_loss: 0.4113 - val_
     Epoch 4/10
     60/60 [==============================] - 42s 697ms/step - loss: 0.3155 - accuracy: 0.8714 - val_loss: 0.2872 - val_
     Epoch 5/10
     60/60 [==============================] - 42s 704ms/step - loss: 0.3052 - accuracy: 0.8740 - val_loss: 0.3404 - val_
     Epoch 6/10
     60/60 [==============================] - 42s 698ms/step - loss: 0.3096 - accuracy: 0.8729 - val_loss: 0.3142 - val_
     Epoch 7/10
     60/60 [==============================] - 42s 700ms/step - loss: 0.3220 - accuracy: 0.8646 - val_loss: 0.2490 - val_
     Epoch 8/10
     60/60 [==============================] - 42s 697ms/step - loss: 0.2817 - accuracy: 0.8885 - val_loss: 0.2439 - val_
     Epoch 9/10
     60/60 [==============================] - 42s 711ms/step - loss: 0.2869 - accuracy: 0.8911 - val_loss: 0.2548 - val_
     Epoch 10/10
     60/60 [==============================] - 42s 698ms/step - loss: 0.2721 - accuracy: 0.8969 - val_loss: 0.2389 - val_
```

Kompilasi dan Pelatihan Model Klasifikasi

Kode di atas melakukan dua langkah kunci dalam pelatihan model klasifikasi. Pertama, **model.compile** digunakan untuk mengonfigurasi proses pelatihan dengan menentukan optimizer (dalam hal ini, menggunakan Adam dengan learning rate 1e-3), fungsi loss (binary_crossentropy untuk tugas klasifikasi biner), dan metrik evaluasi yang diinginkan (akurasi).

Kemudian, **model.fit** digunakan untuk melatih model menggunakan dataset yang telah dipersiapkan sebelumnya (**train_generator** untuk data training dan **validation_generator** untuk data validasi). Proses pelatihan dilakukan selama 10 epoch dengan 60 langkah per epoch (steps_per_epoch). Hasil pelatihan disimpan dalam variabel **history** untuk analisis lebih lanjut atau visualisasi. Proses ini bertujuan untuk mengoptimalkan parameter model agar dapat melakukan klasifikasi gambar Pria dan Wanita dengan akurasi tinggi.

4. Evaluasi

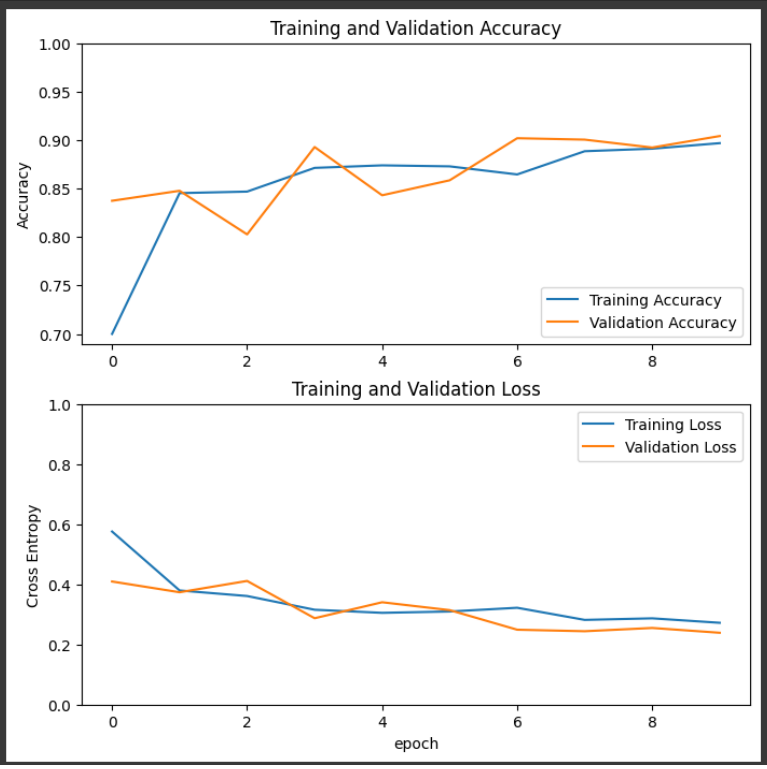
```
[14] import matplotlib.pyplot as plt

     acc = history.history['accuracy']
     val_acc = history.history['val_accuracy']

     loss = history.history['loss']
     val_loss = history.history['val_loss']

     plt.figure(figsize=(8, 8))
     plt.subplot(2, 1, 1)
     plt.plot(acc, label='Training Accuracy')
     plt.plot(val_acc, label='Validation Accuracy')
     plt.legend(loc='lower right')
     plt.ylabel('Accuracy')
     plt.ylim([min(plt.ylim()),1])
     plt.title('Training and Validation Accuracy')

     plt.subplot(2, 1, 2)
     plt.plot(loss, label='Training Loss')
     plt.plot(val_loss, label='Validation Loss')
     plt.legend(loc='upper right')
     plt.ylabel('Cross Entropy')
     plt.ylim([0,1.0])
     plt.title('Training and Validation Loss')
     plt.xlabel('epoch')
     plt.show()
```



Visualisasi Performa Pelatihan Model Klasifikasi

Kode di atas menggunakan matplotlib untuk membuat visualisasi performa pelatihan model klasifikasi. Dua subplot digunakan untuk memplot metrik akurasi dan loss pada data training dan validasi sepanjang epoch. Pada subplot pertama, kurva akurasi untuk training dan validasi ditampilkan, sedangkan pada subplot kedua, kurva loss untuk training dan validasi ditampilkan.

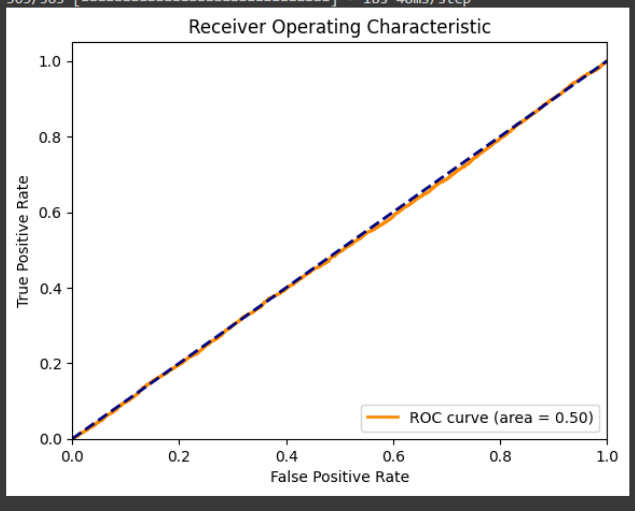
Visualisasi ini memberikan pemahaman tentang sejauh mana model dapat menggeneralisasi dari data pelatihan ke data validasi. Grafik akurasi memberikan informasi tentang seberapa baik model dapat melakukan klasifikasi dengan benar, sementara grafik loss memberikan gambaran tentang seberapa baik model melakukan optimisasi. Pemantauan kedua metrik ini membantu dalam mengevaluasi dan memahami performa model selama proses pelatihan.

```
[15] from sklearn.metrics import roc_auc_score
     from sklearn.metrics import roc_curve, auc

     preds = model.predict(validation_generator, verbose=1)
     fpr, tpr, _ = roc_curve(validation_generator.classes, preds)
     roc_auc = auc(fpr, tpr)

     plt.figure()
     lw = 2
     plt.plot(fpr, tpr, color='darkorange',
              lw=lw, label='ROC curve (area = %0.2f)' % roc_auc)
     plt.plot([0, 1], [0, 1], color='navy', lw=lw, linestyle='--')
     plt.xlim([0.0, 1.0])
     plt.ylim([0.0, 1.05])
     plt.xlabel('False Positive Rate')
     plt.ylabel('True Positive Rate')
     plt.title('Receiver Operating Characteristic')
     plt.legend(loc="lower right")
     plt.show()
```

```
365/365 [==============================] - 18s 48ms/step
```



Visualisasi Kurva Karakteristik Operasi Penerima (ROC) dan Perhitungan Area di Bawah Kurva (AUC)

Kode di atas menggunakan scikit-learn untuk menghitung prediksi model pada dataset validasi dan kemudian menghitung kurva karakteristik operasi penerima (ROC) beserta area di bawah kurva (AUC). ROC curve adalah grafik yang memvisualisasikan hubungan antara True Positive Rate (sensitivitas) dan False Positive Rate (1 - spesifisitas) untuk berbagai nilai ambang batas klasifikasi. AUC merupakan luasan di bawah kurva ROC dan memberikan ukuran seberapa baik model dapat membedakan antara kelas positif dan negatif.

Visualisasi ini membantu dalam mengevaluasi kemampuan model dalam membuat prediksi positif dan negatif serta memberikan gambaran tentang seberapa baik model dapat membedakan antara kelas yang berbeda. Semakin tinggi AUC, semakin baik model dalam membedakan antara kelas-kelas tersebut.

5. Testing Pada Image

```
%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
from google.colab import files
from keras.preprocessing import image

uploaded = files.upload()

for fn in uploaded.keys():

  # predicting images
  path = '/content/' + fn
  img = image.load_img(path, target_size=(224, 224))
  x = image.img_to_array(img)

  plt.imshow(x/255.)
  plt.axis('off')
  plt.show()

  x = np.expand_dims(x, axis=0)
  # images = np.vstack([x])

  classes = model.predict(x)

  if classes[0]<0.5:
    print("\n ini adalah pria")
  else:
    print("\n ini adalah wanita")
```

Choose Files IMG_1510.JPG

Prediksi Jenis Kelamin Berdasarkan Gambar dengan Model Klasifikasi

Kode di atas memungkinkan pengguna untuk mengunggah gambar baru dan melakukan prediksi jenis kelamin (Pria atau Wanita) menggunakan model klasifikasi yang telah dilatih sebelumnya. Setelah gambar diunggah, kode akan menampilkan gambar tersebut dan memberikan prediksi berdasarkan model. Model ini memanfaatkan pretrained model dan transfer learning untuk mengidentifikasi fitur-fitur yang membedakan gambar Pria dan Wanita. Prediksi tersebut kemudian ditampilkan dengan keterangan yang sesuai, yaitu "ini adalah pria" atau "ini adalah wanita".

6. Hasil Testing

```
Choose Files IMG_1507.JPG
• IMG_1507.JPG(image/jpeg) - 7910855 bytes, last modified: 8/25/2023 - 100% done
Saving IMG_1507.JPG to IMG_1507.JPG
```

```
1/1 [==============================] - 0s 31ms/step

 ini adalah pria
```

```
Choose Files pas foto.png
• pas foto.png(image/png) - 634825 bytes, last modified: 11/20/2023 - 100% done
Saving pas foto.png to pas foto.png
```

```
1/1 [==============================] - 0s 20ms/step

 ini adalah wanita
```

```
Choose Files WhatsApp I...cc1e119.jpg
• WhatsApp Image 2023-11-26 at 21.36.21_7cc1e119.jpg(image/jpeg) - 123211 bytes, last modified: 11/26/2023 - 100% done
Saving WhatsApp Image 2023-11-26 at 21.36.21_7cc1e119.jpg to WhatsApp Image 2023-11-26 at 21.36.21_7cc1e119.jpg
```

```
1/1 [==============================] - 0s 28ms/step

 ini adalah wanita
```

Diatas adalah beberapa hasil testing yang telah dicoba pada beberapa foto.

C. KESIMPULAN

- Jumlah foto pria yang digunakan pada file training berjumlah 23.766
- Jumlah foto wanita yang digunakan pada file training berjumlah 23.243
- Jumlah foto pria yang digunakan pada file validation berjumlah 5.808
- Jumlah foto wanita yang digunakan pada file validation berjumlah 5.841
- Data foto dominan diisi oleh foto artis Hollywood
- Pada penelitian ini digunakan arsitektur pretrained MobileNet dikarenakan setelah berbagai macam arsitektur dicoba, MobileNet lah yang paling bagus untuk mendeteksi gambar untuk gender detection
- Hasil testing menunjukan ada beberapa gambar yang salah diprediksi. Ada beberapa hal yang membuat hasil deteksi salah, salah satunya adalah karena dataset dominan diisi oleh foto orang-orang yang berasal dari benua eropa dan amerika yang dimana struktur muka yang dimiliki cukup berbeda dengan struktur muka yang dimiliki orang asia.

D. REFERENSI

- Dataset
  https://drive.google.com/uc?id=1nVPgJkKVxTPbTzXfrZlmLi5hI85TAZfA
- Colab
  https://colab.research.google.com/drive/1S52BxWMplHVxZVM4uJxtDWfg6xkzcpmY?usp=sharing