BAB 4

HASIL DAN PEMBAHASAN

4.1 Inisiasi Library dan Package

Dalam pengembangan aplikasi untuk klasifikasi citra menggunakan Python, beberapa pustaka yang sering digunakan termasuk Numpy, Matplotlib, TensorFlow, dan Keras. Namun, ada juga library tambahan yang mendukung pengembangan proses klasifikasi yang lebih mendalam, yang dapat dilihat pada Gambar 4.1 di bawah ini :

```
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
import os
import tqdm as tqdm
import random as rn
import seaborn as sns
from tensorflow import keras
from keras import layers, models , optimizers
from keras.layers import Dense, Conv2D, Flatten, Dropout, MaxPooling2D, BatchNormalization , Activation, GlobalAveragePooling2D
from skimage.transform import resize
from sklearn.utils import shuffle
from sklearn.model_selection import train_test_split
from keras.utils import to_categorical
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from keras.applications import VGG16
from tensorflow.keras.callbacks import EarlyStopping
from sklearn.metrics import classification_report, accuracy_score
import seaborn as sns
from sklearn.metrics import confusion_matrix
```

Gambar 4. 1 Inisiasi Library dan Package

Seperti beberapa library yang digunakan pada Gambar. 7 untuk mengimplementasikan model dan mengembangkan proses klasifikasi antara lain adalah: Numpy digunakan untuk melakukan operasi numerik secara efisien pada array multidimensi dan matriks, serta menyediakan fungsi matematika tingkat tinggi untuk manipulasi data. Matplotlib merupakan library plotting yang digunakan untuk visualisasi data, pembuatan grafik, dan plot. Berguna untuk memvisualisasikan gambar dan hasil klasifikasi. TensorFlow merupakan platform yang menyediakan berbagai alat dan library untuk pengembangan dan pelatihan model deep learning. OS merupakan modul untuk berinteraksi dengan sistem operasi, seperti pengelolaan file dan direktori. Tqdm untuk membuat progress bar, memudahkan pemantauan iterasi loop yang panjang. Random untuk menghasilkan angka acak dan melakukan operasi acak lainnya.

Seaborn adalah sebuah pustaka visualisasi data yang dibangun di atas matplotlib, dirancang untuk menciptakan grafik statistik yang lebih menarik dan informatif. Keras, API tinggi yang berjalan di atas TensorFlow, menyediakan antarmuka yang mudah digunakan untuk

membangun dan melatih model neural network. Keras Layer (Dense, Conv2D, Flatten, Drpout, MaxPooling2D, BatchNormalization, Activation, GlobabAveragePooling2D) berbagai lapisan neural network yang digunakan untuk membangun aesitektur model, seperti lapisan konvolusi, pooling, normalisasi batch, aktivasi, dan lapisan yang sepenuhnya terhubung. Skinmage.transform (resize) modul dari scikit-image untuk mengubah ukuran gambar, membantu menstandarkan ukuran gambar sebelum diproses oleh model. Sklearn.utils (shuffle) fungsinya untuk mengacak data, yang membantu dalam meningkatkan kinerja model dengan memastikan data latih dalam urutan yang berbeda. `train_test_split` dari `sklearn.model_selection` berfungsi untuk memisahkan dataset menjadi bagian untuk pelatihan dan pengujian, yang membantu dalam mengevaluasi model dengan lebih akurat. Keras.utils (to_categorical) fungsinya untuk mengubah label kelas menjadi bentuk one-hot encoding untuk output model klasifikasi. ImageDataGenerator digunakan untuk augmentasi gambar, memeperbesar dataset dengan variasi gambar yang dihasilkan secara acak. Keras.applications (VGG16) model yang digunakan sebagai dasar untuk transfer learning, memanfaatkan pengetahuan dari model yang telah dilatih pada dataset besar.

4.2 Pembagian Dataset

Pada tahap pembagian dataset, pembagian data dilakukan menjadi data latih dan data uji untuk mengukur kinerja model secara objektif dan menghindari adanya overfitting. Berikut pembagian dataset dapat dilihat pada Gambar 4.2. dibawah ini:

```
# Split the data into train and test
from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.2,random_state=seed)
```

Gambar 4. 2 *Splitting Dataset*

Dalam proses ini, digunakan fungsi 'train_test_split' dari pustaka 'sklearn.model_selection' untuk memecah dataset 'X' dan 'Y' menjadi dua bagian, yaitu data latih ('X_train' dan 'Y_train') serta data uji ('X_test' dan 'Y_test'). Parameter ‘test_size=0.2’ menunjukkan bahwa 20% dari total data yaitu sebanyak 2.305 gambar digunakan untuk pengujian, sedangkan 80% sisanya yaitu sebanyak 9.220 gambar digunakan untuk pelatihan model. Penggunakan parameter ‘random_state=seed’ memastikan bahwa pembagian dataset dilakukan secara konsisten setiap kali kode dijalankan.

4.3 Augmentasi Data

Augmentasi data digunakan untuk membuat variasi baru dari dataset yang ada dengan mengubah sedikit data latih tanpa mengubah maknanya yang membantu meningkatkan generaliasi dan mencegah overfitting. Berikut proses augmentasi data dapat dilihat pada Gambar 4.3. dibawah ini:

```
# Data augmentation
train_datagen = ImageDataGenerator(
    rescale=1.0/255,
    rotation_range=20,
    horizontal_flip=True,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    validation_split=0.25
)

valid_datagen = ImageDataGenerator(
    rescale=1.0/255,
    validation_split=0.25
)
```

Gambar 4. 3 Augmentasi Data

Dalam proses augmentasi diatas digunakan 'ImageDataGenerator' untuk meningkatkan variasi dan generalisasi model. Pada 'train_datagen' dilakukan berberapa transformasi terhadap gambar pada dataset pelatihan. Pertama, dilakukan normaliasi pixel gambar ke rentang [0,1] dengan 'rescale=1.0/255'. Selanjutnya dilakukan rotasi acak hingga 20 derajat ('rotation_range=20') dan flipping horizontal ('horizontal_flip=True') untuk menciptakan variasi posisi dan orientasi objek. Selain itu, diterapkan juga pergeseran lebar dan tinggi gambar hingga 20% ('width_shift_range=0.2' dan 'height_shift_range=0.2') untuk mengatasi variasi posisi objek dalam gambar. Transformasi seperti shear ('shear_range=0.2') dan zoom ('zoom_range=0.2') diterapkan untuk menambah variasi bentuk dan ukuran objek. Kemudian dalam 'valid_datagen' digunakan untuk mengatur data validasi dalam proses pelatihan model. Parameter 'rescale=1.0/255' menskalakan nilai piksel gambar rentang 0-255 ke rentang 0-1 untuk normalisasi data, sementara 'validation_split=0.25' membagi dataset sehingga 25% dari data akan digunakan untuk validasi dan 75% untuk pelatihan.

Penerapan data augmentasi pada penelitian ini sangat penting karena dapat menciptakan variasi yang diperlukan dalam dataset pelatihan tanpa memerlukan pengumpulan data baru yang kompleks. Dengan menghasilkan variasi dalam posisi,

orientasi, dan bentuk objek dalam gambar, augmentasi membantu model untuk mempelajari pola yang lebih umum dan menghindari overfitting terhadap detail spesifik dari dataset latih. Data augmentasi tidak hanya meningkatkan kemampuan model untuk menerapkan generalisasi pada data yang belum pernah dikenali sebelumnya, tetapi juga mengurangi risiko overfitting, di mana model menyesuaikan diri dengan data pelatihan yang spesifik. Dengan demikian, augmentasi data memainkan peran pentimg dalam meningkatkan akurasu dan kestabilan model, serta memastikan bahwa model yang dikembangkan dapat diterapkan dengan efektif dalam berbagai situasi.

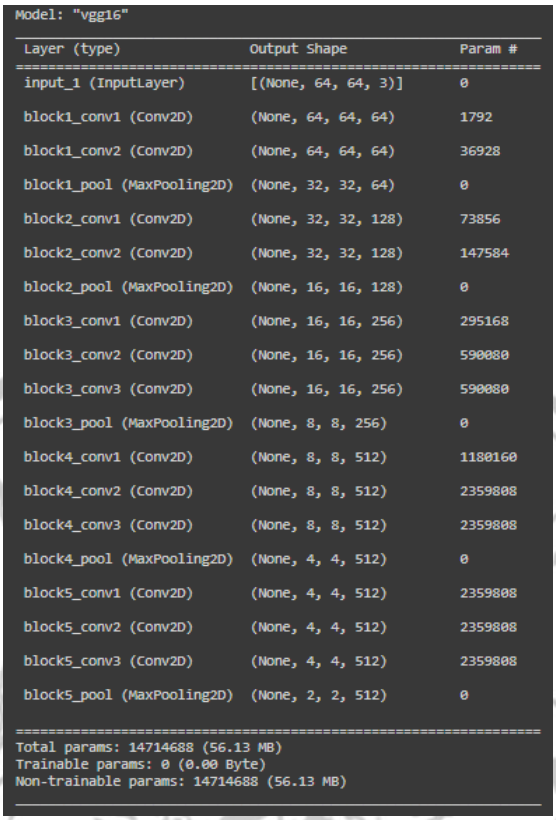
4.4 Implementasi Model

```
# load the vgg16 model #pembuatan arsitektur
from keras.applications import VGG16
base_model = VGG16(weights='imagenet', include_top=False, input_shape=(64, 64, 3))
base_model.trainable = False

# add some layers
model = models.Sequential()
model.add(base_model)
model.add(layers.Flatten())
model.add(layers.Dense(1024, activation='relu'))
model.add(layers.Dense(512, activation='relu'))
model.add(layers.Dense(2, activation='softmax'))
model.summary()
```

Gambar 4. 4 Implementasi Model

Implementasi model ini menggunakan arsitektur VGG16 yang telah dilatih sebelumnya pada dataset ImageNet untuk membangun model deep learning. Tahap awal, model VGG16 diimport dari library Keras dan dimuat dengan parameter yang telah dilatih, tanpa menyertakan lapisan klasifikasi atasnya (‘include_top=False’). Model ini dikonfigurasi untuk menerima gambar yang berukuran 64x64 piksel dengan tiga saluran warna yaitu RGB. Semua lapisam dalam model VGG16 dibekukan (‘trainable=False’) agar parameter tersebut tidak berubah selama proses pelatihan. Ringkasan dari model dasar kemudian ditampilkan untuk memeriksa strukturnya. Struktur model dapat dilihat pada Gambar 4.5.



```
Model: "vgg16"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 64, 64, 3)]       0

 block1_conv1 (Conv2D)       (None, 64, 64, 64)        1792

 block1_conv2 (Conv2D)       (None, 64, 64, 64)        36928

 block1_pool (MaxPooling2D)  (None, 32, 32, 64)        0

 block2_conv1 (Conv2D)       (None, 32, 32, 128)       73856

 block2_conv2 (Conv2D)       (None, 32, 32, 128)       147584

 block2_pool (MaxPooling2D)  (None, 16, 16, 128)       0

 block3_conv1 (Conv2D)       (None, 16, 16, 256)       295168

 block3_conv2 (Conv2D)       (None, 16, 16, 256)       590080

 block3_conv3 (Conv2D)       (None, 16, 16, 256)       590080

 block3_pool (MaxPooling2D)  (None, 8, 8, 256)         0

 block4_conv1 (Conv2D)       (None, 8, 8, 512)         1180160

 block4_conv2 (Conv2D)       (None, 8, 8, 512)         2359808

 block4_conv3 (Conv2D)       (None, 8, 8, 512)         2359808

 block4_pool (MaxPooling2D)  (None, 4, 4, 512)         0

 block5_conv1 (Conv2D)       (None, 4, 4, 512)         2359808

 block5_conv2 (Conv2D)       (None, 4, 4, 512)         2359808

 block5_conv3 (Conv2D)       (None, 4, 4, 512)         2359808

 block5_pool (MaxPooling2D)  (None, 2, 2, 512)         0

=================================================================
Total params: 14714688 (56.13 MB)
Trainable params: 0 (0.00 Byte)
Non-trainable params: 14714688 (56.13 MB)
_________________________________________________________________
```

Gambar 4. 5 Struktur Model VGG16

Kemudian, model baru dibangun menggunakan Keras (‘Sequential()’) dan beberapa lapisan ditambahkan di atas model VGG16. Pertama, lapisan 'Flatten' digunakan untuk merubah data dari format dua dimensi menjadi satu dimensi. Setelah itu, dua lapisan 'Dense' ditambahkan, masing-masing dengan 1024 dan 512 neuron, menggunakan fungsi aktivasi ReLU untuk memperkenalkan non-linearitas. Lapisan 'Dense' terakhir memiliki 2 neuron dan menggunakan fungsi aktivasi softmax untuk mengklasifikasikan input ke dalam dua kelas. Model ini memanfaatkan kemampuan VGG16 dalam mengenali fitur dan menambahkan lapisan tambahan untuk menyesuaikan tugas klasifikasi yang spesifik.

4.5 Pelatihan Model

```
# Melatih model dengan menggunakan data augmentation pada setiap epoch
history = model.fit(X_train, Y_train_to_categorical, epochs=10, batch_size=256, validation_split=0.2)
```

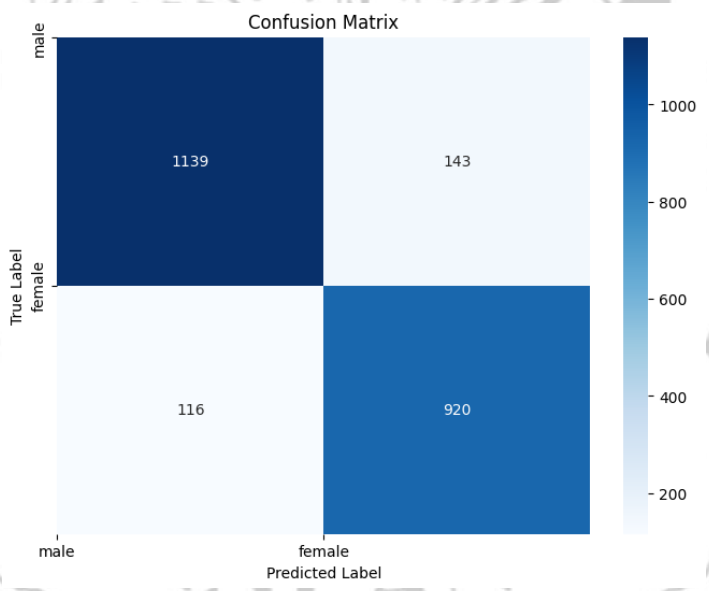
Gambar 4. 6 Pelatihan Model

Proses pelatihan dilakukan dengan memanggil fungsi ‘fit’ pada objek model menggunakan dataset pelatihan dan label. Selama pelatihan, model diperbarui dan disesuaikan dengan data. Parameter yang digunakan untuk pelatihan adalah sebagai berikut: jumlah epoch sebanyak 10, ukuran batch sebanyak 256 dan pembagian validasi sebesar 20% dari data pelatihan.

4. 6 Evaluasi Model

Pada tahap ini, model dievaluasi dengan menggunakan confusion matrix, tingkat akurasi, serta grafik loss. Tujuannya adalah untuk mengukur kinerja model, membandingkannya dengan model lain, dan mengidentifikasi kemungkinan masalah seperti overfitting..

4.6.1 Confusion Matrix



Gambar 4. 7 Confusion Matrix

Dapat dilihat pada Gambar Confusion Matrix diatas bahwa 1139 data kelas male diprediksi dengan benar sebagai kelas male dan 143 data kelas male salah prediksi sebagai kelas female. 920 data kelas female diprediksi dengan benar sebagai kelas female dan 116 data kelas female salah prediksi sebagai kelas male.

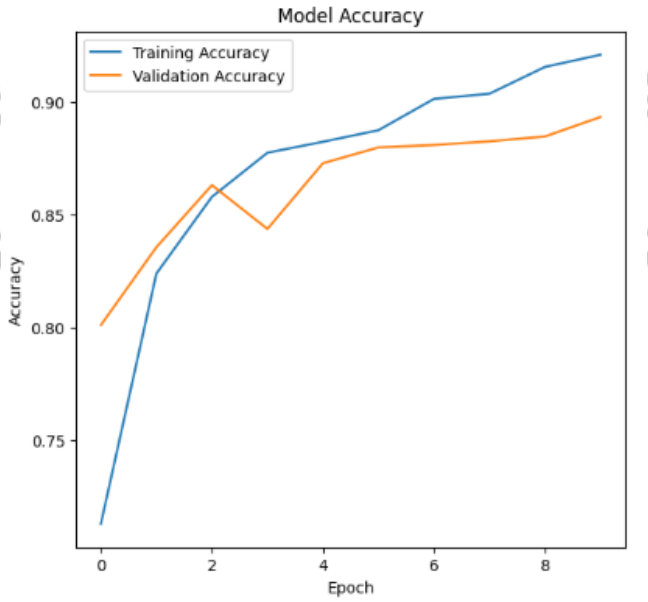
Dari hasil Confusion Matrix yang diperoleh, terlihat bahwa model cenderung melakukan prediksi yang lebih baik untuk kelas female dari pada male. Jumlah data yang diprediksi dengan benar untuk kelas female (920) lebih tinggi dari pada kelas male (1139). Sementara jumlah data yang salah diprediksi sebagai female (143) lebih rendah dari pada jumlah yang salah diprediksi sebagai male (116). Hal ini menunjukkan kecenderungan model untuk memprediksi kelas mayoritas yaitu female.

Beberapa faktor yang menyebabkan kondisi ini antara lain, termasuk ketidakseimbangan dalam jumlah sampel antara kelas male dan female, karakteristik fitur yang lebih representatif untuk kelas female, atau kesulitan model dalam mempelajari pola-pola yang membedakan sampel dalam kelas male.

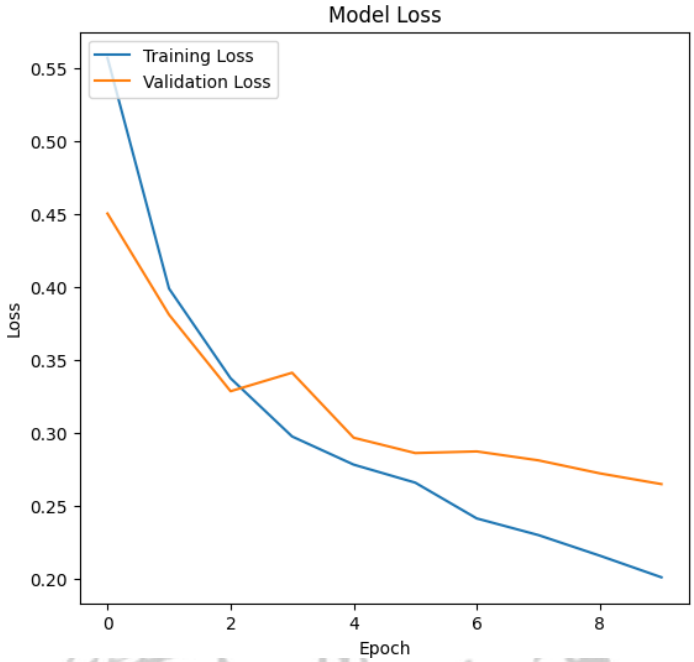
Gambar 4. 8 Sampel Kelas Male dan Female

Pada gambar sampel kelas male dan female diatas untuk label:0 menunjukkan kelas male sedangkan label:1 menunjukkan kelas female.

4.6.2 Accuracy dan Loss



Gambar 4. 9 Grafik Model Accuracy



Gambar 4. 10 Grafik Model Loss

Dari Gambar 4.9 model menunjukkan performa terbaiknya pada epoch ke-10 dengan validasi accuracy sebesar 0.8933 dan pada Gambar 4.10 menunjukkan validasi loss terendah dicapai pada epoch ke-10 sebesar dengan nilai 0.2648.

4. 7 Perbandingan Performa Model

Penelitian ini menggunakan dataset yang serupa dengan penelitian sebelumnya, tentang klasifikasi gender melalui citra mata. Hasil penelitian sebelumnya menunjukkan bahwa tingkat akurasi masih memiliki ruang untuk ditingkatkan. Oleh karena itu, penelitian ini mengembangkan pendekatan dengan menerapkan Convolutional Neural Network menggunakan arsitektur model VGG16 untuk mencapai tingkat akurasi yang lebih tinggi. Hasil dari penelitian ini dapat dilihat pada Gambar 4.11 di bawah ini :

	precision	recall	f1-score	support
male	0.91	0.89	0.90	1282
female	0.87	0.89	0.88	1036
accuracy			0.89	2318
macro avg	0.89	0.89	0.89	2318
weighted avg	0.89	0.89	0.89	2318

Accuracy: 88.83%

Gambar 4. 11 Classification Report

Untuk perbandingan performa model dapat dilihat pada tabel 3 dibawah ini:

Tabel 4. 1 Perbandingan Performa Model

Model	Akurasi
CNN	84.78%
VGG16	88.83%

Dari hasil pada tabel di atas, dapat diamati bahwa model Arsitektur VGG16 menunjukkan peningkatan akurasi dibandingakan dengan penerapan model CNN saja. Hal ini menunjukkan keunggulan penerapan arsitektur VGG16 dalam meningkatkan akurasi klasifikasi gender melalui citra mata.