DAFTAR PUSTAKA

Huang, G., Liu, Z., Maaten, L. v., & Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2261-2269.

Ilahiyah, S., & Nilogiri, A. (2018). Implementasi Deep Learning Pada Identifikasi Jenis Tumbuhan Berdasarkan Citra Daun Menggunakan Convolutional Neural Network. *Jurnal Sistem dan Teknologi Informasi Indonesia (justindo), III*, 49-56.

Mahmud, K. H., Adiwijaya , A., & Faraby, S. A. (2019). Klasifikasi Citra Multi-kelas Menggunakan Convolutional Neural Network. *eProceedings of Engineering, VI*, 2127-2136.

Nisa, C., Puspaningrum, E. Y., & Maulana, H. (2020, November 1). Penerapan Metode Convolutional Neural Network untuk Klasifikasi Penyakit Daun Apel pada Imbalanced Data. *Prosiding Seminar Nasional Informatika Bela Negara*, 169-175.

Pardede , J., & Putra, D. A. (2020, Desember 3). Implementasi DenseNet Untuk Mengidentifikasi Kanker Kulit Melanoma. *Jurnal Teknik Informatika dan Sistem Informasi*, 3-10.

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., & Citro, C. (2016). TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *Cornell University*, 172.

Abas, M. A., Ismail, N., Yassin, A. I., & Taib, M. N. (2018, Oktober 2). VGG16 for Plant Image Classification with Transfer Learning and Data Augmentation. *International Journal of Engineering & Technology, VII*(11), 90 - 94.

Agarap, A. F. (2018, Maret 22). Deep Learning using Rectified Linear Units (ReLU). *arXiv:1803.08375*.

Albawi, S., Mohammed, T. A., & Al-Zawi, S. (2017, Agustus 21). Understanding of a convolutional neural network. *International Conference on Engineering and Technology (ICET)*.

Albert, A., Gunadi, K., & Setyati, E. (2020). Deteksi Helm Pada Pengguna Sepeda Motor Dengan Metode Convolutional Neural Network. *Jurnal Infra, 8*, 295-301.

Altuntas, Y., Comert, Z., & Kocamaz, A. F. (2019, Agustus). Identification of haploid and diploid maize seeds using convolutional neural networks and a transfer learning approach. *Computers and Electronics in Agriculture, 206*, 163.

Asmara, R. A., Syulistyo, A. R., & Qudsi, N. K. (2020, Januari 22). Identifikasi Citra Tulisan Tangan Digital Menggunakan Convolutional Neural Network (CNN). *Seminar Informatika Aplikatif Polinema (SIAP)*, 48-53.

Faisal, M. R. (2017, Januari 12). *Menghitung kinerja algoritma klasifikasi: Pilih ROC Curve atau Precision-Recall Curve?* Diambil Maret 2022, dari M Reza Faisal: http://etd.repository.ugm.ac.id/penelitian/detail/162253

Hariyani, Y. S., Hadiyoso, S., & Siadari, T. S. (2020). Deteksi Penyakit Covid-19 Berdasarkan Citra X-Ray Menggunakan Deep Residual Network. *ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika, VIII*, 443.

Kusmantoro, Z. N. (2018). Akurasi Uji Diagnostik Menggunakan Luasan Bawah Kurva ROC Smoothed Empirical . *Universitas Gadjah Mada*.

LeCun, Y., Bengio, Y., & Hinton, G. (2015, Mei 27). Deep learning. *nature*, 436-444.

Manliguez, C. (2016, November). Generalized Confusion Matrix for Multiple Classes. *Researchgate*, 4-6.

Maulana, F. F., & Rochmawati, N. (2019). Klasifikasi Citra Buah Menggunakan Convolutional Neural Network. *Journal of Informatics and Computers Science, I*.

Murphy, K. M. (2018). *Machine Learning: A Probabilistic Perspective (Adaptive Computation and Machine Learning series)*. London: The MIT Press.

Oliver, A. (2022, Januari 25). *Mengenal Google Colab: Mulai dari Definisi, Cara Menggunakan, hingga Manfaatnya*. Retrieved Desember 2022, from glints.com.

Pamungkas, A. (2017, Juli 26). *Pengolahan Citra DIgital*. Diambil Desember 2022, dari Pemograman Matlab: https://pemrogramanmatlab.com/2017/07/26/pengolahan-citra-digital/

Pangestu, M. A., & Bunyamin, H. (2018, Agustus 24). Analisis Performa dan Pengembangan Sistem Deteksi Ras Anjing pada Gambar dengan Menggunakan Pre-Trained CNN Model. *Jurnal Teknik Informatika dan Sistem Informasi (JuTISI), IV*, 341-348.

Rakamawati, J. V. (2021). Klasifikasi Diabetic Retinopathy Berdasarkan Foto. 29.

Ramdan, A., Zilvan, V., Suryawati, E., Pardede, H. F., & Rahadi, V. P. (2020, Oktober). Klasifikasi klon teh berbasis deep CNN dengan residual dan densely connections. *Jurnal Teknologi Sistem Komputer, VIII*(4).

Schimdhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks, XVI*, 85-117.

Sentosa, B., & Umam, A. (2018). *Data Mining Dan Big Data Analytics : Teori Dan Implementasi : Mengunakan Python & Apche Spark.* Penerbit Media Pustaka.

Takaoka, Y., & Kawakami, N. (2013, Desember 20). Fruit and vegetable consumption in adolescence and health in early adulthood: a longitudinal

analysis of the statistics Canada's National Population Health Survey. *National Library of Medicine*.

Yohannes, Y., Pribadi, M. R., & Chandra, L. (2020). Klasifikasi Jenis Buah dan Sayuran Menggunakan SVM Dengan Fitur Saliency-HOG dan Color Moments. *Jurnal Teknik Elektro Universitas Tanjungpura, XII*, 125-131.

Yulianto, M. I. (2021, Maret 15). *Membuat Aplikasi Web Sains Data dengan Mudah Menggunakan Streamlit*. Diambil Desember 2022, dari Universitas Islam Indonesia: https://informatics.uii.ac.id/2021/03/15/streamlit-membuat-aplikasi-web-sains-data/

Zhi, T., Duan, L.-y., Wang, Y., & Huang, T. (2016, Agustus). Two-stage pooling of deep convolutional features for image retrieval. *IEEE xplore*, 2465-2469.

LAMPIRAN

Lampiran 1. Kode Arsitektur DenseNet201

```
from tensorflow.keras.layers import Dense, Conv2D, MaxPooling2D, Flatten
import keras
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import matplotlib.pyplot as plt
import cv2
import numpy as np
import PIL
from tensorflow.keras import layers
#Mengimport atau Memanggil setiap input yang akan digunakan
from google.colab import drive
drive.mount('/content/gdrive')
#Menghubungkan ke google drive untuk mendapatkan dataset


training_images_generator=ImageDataGenerator(rescale=1.0/255.0,
validation_split=0.25)
testing_images_generator = ImageDataGenerator(rescale=1.0/255.0)


training_images = training_images_generator.flow_from_directory(
                                                    '/content/gdrive/MyDrive/Colab
Notebooks/fruit360CNN/Training',
                        color_mode='rgb',
                        target_size=(224, 224),
                        batch_size=64,
                        class_mode='categorical')


valid_images = testing_images_generator.flow_from_directory(
                                                    '/content/gdrive/MyDrive/Colab
Notebooks/fruit360CNN/Test',
```

```
                    color_mode='rgb',
                    target_size=(224, 224),
                    batch_size=64,
                    shuffle=False,
                    seed=42,
                    class_mode='categorical')
# preprocessing data
import keras
base_model = keras.applications.densenet.DenseNet201(
    weights='imagenet',  # Muat bobot yang telah dilatih sebelumnya di ImageNet.
   input_shape=(224, 224, 3),
    include_top=False)  # Jangan sertakan pengklasifikasi ImageNet di bagian atas.

base_model.trainable = False

inputs = keras.Input(shape=(224, 224, 3))
#memastikan bahwa base_model berjalan dalam mode inferensi di sini,
# by passing `training=False`. This is important for fine-tuning, as you will
# learn in a few paragraphs.
x = base_model(inputs, training=False)
# Convert features of shape `base_model.output_shape[1:]` to vectors
x = keras.layers.GlobalAveragePooling2D()(x)
# A Dense classifier with a multi unit (categorical classification)
outputs = keras.layers.Dense(16, activation='softmax')(x)
model = keras.Model(inputs, outputs)

model.summary()
```

```
optimizer = keras.optimizers.Adam(learning_rate=1e-5)
model.compile(loss='categorical_crossentropy', optimizer=optimizer, metrics=['accuracy'])
#model cnn
checkpointer = keras.callbacks.ModelCheckpoint(filepath='model_densenet_{epoch}.h5', period=1, monitor='val_loss', mode='min', verbose=1)

hist = model.fit(training_images, epochs=60, callbacks=[checkpointer],
                 validation_data=valid_images, shuffle=True)
#training data
import matplotlib.pyplot as plt
plt.figure(figsize=(10,5))
plt.plot(hist.history['loss'], label='train')
plt.plot(hist.history['val_loss'], label='valid')
plt.legend()
plt.show()
#grafik loss
plt.figure(figsize=(10,5))
plt.plot(hist.history['accuracy'], label='train')
plt.plot(hist.history['val_accuracy'], label='valid')
plt.legend()
plt.show()
#grafik akurasi
model.save("DenseNet 1e-5 60 epoch.h5")
#menyimpan model
optimizer = keras.optimizers.Adam(learning_rate=1e-3)
model.compile(loss='categorical_crossentropy', optimizer=optimizer, metrics=['accuracy'])
checkpointer = keras.callbacks.ModelCheckpoint(filepath=' ', period=1, monitor='val_loss', mode='min', verbose=1)
```

```
hist = model.fit(training_images, epochs=60, callbacks=[checkpointer],
                 validation_data=valid_images, shuffle=True)
#training data lr10-3
import matplotlib.pyplot as plt
plt.figure(figsize=(10,5))
plt.plot(hist.history['loss'], label='train')
plt.plot(hist.history['val_loss'], label='valid')
plt.legend()
plt.show()
#grafik loss lr10-4
plt.figure(figsize=(10,5))
plt.plot(hist.history['accuracy'], label='train')
plt.plot(hist.history['val_accuracy'], label='valid')
plt.legend()
plt.show()
#grafik akurasi 10-3
model.save("DenseNet 1e-3 60 epoch.h5")
optimizer = keras.optimizers.Adam(learning_rate=1e-4)
model.compile(loss='categorical_crossentropy', optimizer=optimizer,
metrics=['accuracy'])
checkpointer = keras.callbacks.ModelCheckpoint(filepath=' ', period=1,
monitor='val_loss', mode='min', verbose=1)


hist = model.fit(training_images, epochs=60, callbacks=[checkpointer],
                 validation_data=valid_images, shuffle=True)
#model lr 10-4
import matplotlib.pyplot as plt
plt.figure(figsize=(10,5))
plt.plot(hist.history['loss'], label='train')
plt.plot(hist.history['val_loss'], label='valid')
```

```
plt.legend()
plt.show()
#grafik loss lr 10-4
plt.figure(figsize=(10,5))
plt.plot(hist.history['accuracy'], label='train')
plt.plot(hist.history['val_accuracy'], label='valid')
plt.legend()
plt.show()
#grafik akurasi lr 10-4
model.save("DenseNet 1e-4 60 epoch.h5")
```

Lampiran 2. Kode Pengujian Model

```
from tensorflow.keras.layers import Dense, Conv2D, MaxPooling2D, Flatten
import keras
import PIL
# to prepare data set for images
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras import layers
import matplotlib.pyplot as plt
import cv2
import numpy as np
#mengimport semua data yang digunakan
from google.colab import drive
drive.mount('/content/gdrive')
#menyimpan direktori
# STEP-1 Prepare DataSet
training_images_generator = ImageDataGenerator(rescale=1.0/255.0, validation_split=0.25)
testing_images_generator = ImageDataGenerator(rescale=1.0/255.0)
```

```
training_images = training_images_generator.flow_from_directory(
                                                    '/content/gdrive/MyDrive/Colab Notebooks/fruit360CNN/Training',
                        color_mode='rgb',
                        target_size=(224, 224),
                        batch_size=64,
                        class_mode='categorical')



valid_images = testing_images_generator.flow_from_directory(
                                                    '/content/gdrive/MyDrive/Colab Notebooks/fruit360CNN/Test',
                        color_mode='rgb',
                        target_size=(224, 224),
                        batch_size=64,
                        shuffle=False,
                        seed=42,
                        class_mode='categorical')
#preprocessing data
new_model = keras.models.load_model("/content/gdrive/MyDrive/Colab Notebooks/fruit360CNN/model Densenet/DenseNet 1e-5 60 epoch.h5")
valid_images.reset()
y_pred =new_model.predict(valid_images, verbose=1)
y_true = valid_images.classes
#model cnn lr 10-5
from sklearn.metrics import roc_auc_score, accuracy_score, confusion_matrix, roc_curve, auc, precision_recall_curve, f1_score, classification_report
accuracy_score(y_true, np.argmax(y_pred,axis=1))
from sklearn.metrics import classification_report
print(classification_report(y_true, np.argmax(y_pred,axis=1)))
conf_mat = confusion_matrix(y_true, np.argmax(y_pred,axis=1))
```

```
import pandas as pd
import seaborn as sns
list_classes = ['apple_braeburn_1','apple_crimson_snow_1','apple_golden_1','apple_granny_smith_1','apple_hit_1','apple_pink_lady_1','apple_red_1','apple_red_yellow_1','apple_rotten_1','cabbage_white_1','carrot_1','cucumber_1','eggplant_violet_1','pear_1','zucchini_1','zucchini_dark_1']
df_cm = pd.DataFrame(conf_mat,list_classes,list_classes)
plt.figure(figsize=(10,7))
sns.set(font_scale=1.1) # for label size
sns.heatmap(df_cm, annot=True, annot_kws={"size": 16}, fmt='g') # font size
plt.xlabel('Predicted')
plt.ylabel('Truth')
plt.show()
#klasifikasi gambar
from sklearn.preprocessing import label_binarize
y_true_binarized=label_binarize(y_true, classes=np.unique(y_true))
#binarize the y_values

#roc curve for classes
fpr = {}
tpr = {}
thresh ={}
roc_auc=dict()

n_class = 16
plt.figure(figsize=(10 ,7))
for i in range(n_class):
  fpr[i], tpr[i], thresh[i]=roc_curve(y_true_binarized[:,i], y_pred[:,i])
  roc_auc[i] = auc(fpr[i], tpr[i])
#kurva roc
```

```
  #plotting
  plt.plot(fpr[i], tpr[i], linestyle='--',
           label='%s vs Rest (AUC=%0.2f)'%(list_classes[i], roc_auc[i]))


plt.plot([0,1], [0,1],'b--')
plt.xlim([0,1])
plt.ylim([0,1.05])
plt.title('Multiclass ROC curve')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.legend(loc='lower right')
plt.show()


new_model = keras.models.load_model("/content/gdrive/MyDrive/Colab Notebooks/fruit360CNN/model Densenet/DenseNet 1e-3 60 epoch.h5")


valid_images.reset()


y_pred = new_model.predict(valid_images, verbose=1)
y_true = valid_images.classes
#model cnn lr 10-3
from sklearn.metrics import roc_auc_score, accuracy_score, confusion_matrix, roc_curve, auc, precision_recall_curve, f1_score, classification_report


accuracy_score(y_true, np.argmax(y_pred,axis=1))
conf_mat = confusion_matrix(y_true, np.argmax(y_pred,axis=1))


import pandas as pd
import seaborn as sns
list_classes = ['apple_braeburn_1','apple_crimson_snow_1','apple_golden_1','apple_granny_smith_1','apple_hit_1','apple_pink_lady_1','apple_red_1','apple_red_yellow_1','a
```

```
pple_rotten_1','cabbage_white_1','carrot_1','cucumber_1','eggplant_violet_1','pe
ar_1','zucchini_1','zucchini_dark_1']
df_cm = pd.DataFrame(conf_mat,list_classes,list_classes)
plt.figure(figsize=(10,7))
sns.set(font_scale=1.1) # for label size
sns.heatmap(df_cm, annot=True, annot_kws={"size": 16}, fmt='g') # font size
plt.xlabel('Predicted')
plt.ylabel('Truth')
plt.show()
#kurva roc
from sklearn.preprocessing import label_binarize
y_true_binarized=label_binarize(y_true, classes=np.unique(y_true))
#binarize the y_values

#roc curve for classes
fpr = {}
tpr = {}
thresh ={}
roc_auc=dict()

n_class = 16
plt.figure(figsize=(10 ,7))
for i in range(n_class):
  fpr[i], tpr[i], thresh[i]=roc_curve(y_true_binarized[:,i], y_pred[:,i])
  roc_auc[i] = auc(fpr[i], tpr[i])

   #plotting
  plt.plot(fpr[i], tpr[i], linestyle='--',
           label='%s vs Rest (AUC=%0.2f)'%(list_classes[i], roc_auc[i]))
```

```
plt.plot([0,1], [0,1],'b--')
plt.xlim([0,1])
plt.ylim([0,1.05])
plt.title('Multiclass ROC curve')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.legend(loc='lower right')
plt.show()
```

Lampiran 3. Kode Aplikasi *Web*

```
!pip install streamlit
%%writefile app.py
import streamlit as st
import cv2
import tensorflow as tf
import keras
import numpy as np
import tensorflow_hub as hub
from tensorflow.keras import preprocessing
from PIL import Image, ImageOps
import requests
from tensorflow.keras.models import load_model
from tensorflow.keras.activations import softmax
import h5py
import matplotlib.pyplot as plt
import os
#mengimport semua yang digunakan
def main():
    file_uploaded = st.file_uploader("Silakan mengunggah citra gambar",
```

```
type=["jpg","png"])
if file_uploaded is not None:
        image = Image.open (file_uploaded)
        figure = plt.figure()
        plt.axis('off')
        result = predict_class(image)
        st.write (result)
        st.pyplot(figure)
st.write("""
         # Fruit and Vegetable Classification
       """
         )

def predict_class(image):
                                        classifier_model                                =
tf.keras.models.load_model(r'/content/gdrive/MyDrive/Colab
Notebooks/fruit360CNN/model Densenet/DenseNet 1e-5 60 epoch.h5')
  model = tf.keras.Sequential([hub.KerasLayer(classifier_model)])
  test_image = image.resize ((224, 224))
  test_image = preprocessing.image.img_to_array(test_image)
  test_image = test_image.astype('float32')/255
  test_image = np.expand_dims(test_image, axis=0)
  class_names =['apple_braeburn_1',
                'apple_crimson_snow_1',
                'apple_golden_1',
                'apple_granny_smith_1',
                'apple_hit_1',
                'apple_pink_lady_1',
                'apple_red_1',
                'apple_red_yellow_1',
                'apple_rotten_1',
```

```
            'cabbage_white_1',
            'carrot_1',
            'cucumber_1',
            'eggplant_violet_1',
            'pear_1',
            'zucchini_1',
            'zucchini_dark_1']

  st.image(image, use_column_width=True)
  predictions = model.predict(test_image)
  scores = tf.nn.softmax(predictions[0])
  scores = scores.numpy()
  image_class = class_names[np.argmax(scores)]
  result="Gambar ini diprediksi sebagai citra {}".format(image_class)
  return result

if __name__ == "__main__":
    main()
#mengatur tata letak pada website

!pip install pyngrok
!ngrok                                                    authtoken
2J5ALms75wEE25eGc4T4501hpAT_4zZHGQYSDVXMmSr5okJSe
!nohup streamlit run --server.port 80 app.py &
from pyngrok import ngrok
url = ngrok.connect(port='80')
url
#menginstal ngrok
```