LAPORAN AKHIR TUGAS

PENGENALAN POLA AKSARA JAWA

Dosen Mata Kuliah :

Imam Much. Ibnu Subroto, ST., M.Sc., Ph.D

Disusun Oleh Kelompok 8 :

1. Muhammad Irfan Rosadi (32602100071)
2. Muhammad Agil Chabibun Najah (32602100072)
3. Muhammad Dzaki Wicaksono (32602100073)
4. Muhammad Fakhrul Reza (32602100075)

PRODI TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM SULTAN AGUNG SEMARANG

2024

TUGAS PENGENALAN POLA AKSARA JAWA

A. Perancangan Pengambilan Data dan Preprocessing

1. Pengumpulan Data

Dataset yang digunakan dalam tugas ini bersumber dari kaggle : https://www.kaggle.com/datasets/phiard/aksara-jawa?resource=download-directory. Dataset yang digunakan dalam format .zip.

2. Preprocessing
   a. Greyscale convertion

      Mengubah gambar berwarna menjadi gambar grayscale untuk mengurangi kompleksitas data.

   b. Resizeing

      Teknik ini berguna untuk mengubah ukuran gambar atau video menjadi resolusi yang diinginkan oleh model Anda.

   c. Augmentasi Data

      Lakukan augmentasi data seperti rotasi, flipping, atau perubahan kontras untuk meningkatkan variasi data pelatihan.

B. Membaca Data dan Preprocessing Data

1. Menghubungkan Google Colab dengan drive

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

Pada tahap ini adalah proses menghubungkan google colaboratory dengan google drive pada email yang akan digunakan.

2. Mengekstrak file yang terdapat dalam arsip .zip ke dalam direktori

```
[ ]  import zipfile

     filename = '/content/drive/My Drive/dataset/datasetv3.zip'
     zip_ref = zipfile.ZipFile(filename)
     zip_ref.extractall()
     zip_ref.close()
```

Kode ini membuka file ZIP yang berisi dataset gambar dari Google Drive, mengekstrak isinya ke direktori kerja saat ini, dan kemudian menutup file ZIP.

3. Mengimport library yang akan digunankan

```
import os
import time

import numpy as np
import matplotlib.pyplot as plt
import matplotlib.pyplot as plt_image
plt.style.use('seaborn')

import tensorflow as tf

from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.preprocessing import image as keras_image

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPool2D
from tensorflow.keras.layers import Flatten, Dense, Dropout

from tensorflow.keras.callbacks import TensorBoard
%load_ext tensorboard
```

Kode ini digunakan untuk mengimport libraty yang diperlukan untuk membangun dan melatih model CNN menggunakan TensorFlow dan Keras untuk tugas klasifikasi gambar, serta menyediakan alat untuk memantau proses pelatihan menggunakan TensorBoard.

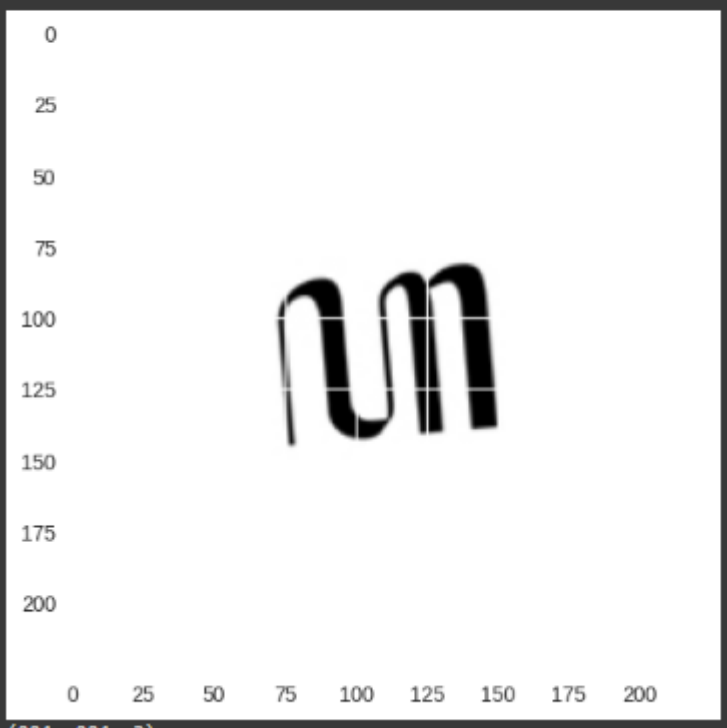
4. Overview dari dataset

```
[ ] os.listdir()

['.config', 'drive', 'datasetv2', 'sample_data']

os.listdir('datasetv2')

['training', 'testing', 'prediction']

print(len(os.listdir('datasetv2/training/ha')))
print(len(os.listdir('datasetv2/training/na')))
print(len(os.listdir('datasetv2/training/ca')))

148
156
156

[ ] print(len(os.listdir('datasetv2/testing/ha')))
print(len(os.listdir('datasetv2/testing/na')))
print(len(os.listdir('datasetv2/testing/ca')))

65
65
65

[ ] print(len(os.listdir('datasetv2/prediction/')))

25
```

Pada tahap ini adalah sebuah proses untuk membaca sebuah folder yang ada pada dataset yang kami gunakan. Di dalam folder dataset yang kami gunakan terdapat 3 folder yaitu folder training, tersing, dan prediction. Kode eksekusi yang digunakan adalah os.lostdir().

5. Memvisualisasikan beberapa contoh datanya

```
preview = plt_image.imread('datasetv2/training/ha/ha.3.jpg')
plt.imshow(preview)
plt.show()
print(preview.shape)
```




```
(224, 224, 3)
```

Di sini kita bisa melihat dataset kita berukuran 224x224 dengan profil warna RGB, namun nyatanya gambarnya hanya hitam putih saja, jadi nanti perlu kita ubah ke skala abu-abu.

6. Preview dari folder datasetv2

```
TOTAL = 6                                                    # Total of preview images
index = np.random.randint(20)                                # To randomize the index photo

plt.figure(figsize=(3*TOTAL, 9))

list_ha = [os.path.join('datasetv2/training/ha', name) for name in os.listdir('datasetv2/training/ha')[index:index+TOTAL]]
list_na = [os.path.join('datasetv2/training/na', name) for name in os.listdir('datasetv2/training/na')[index:index+TOTAL]]
list_ca = [os.path.join('datasetv2/training/ca', name) for name in os.listdir('datasetv2/training/ca')[index:index+TOTAL]]

for i, path in enumerate(list_ha+list_na+list_ca):
  plt.subplot(3, TOTAL, i+1)
  plt.axis('off')

  plt.title(path.replace('datasetv2/training/', ''))
  img = plt_image.imread(path)
  plt.imshow(img)
```

7. Data processing

```
train_datagen = ImageDataGenerator(rescale=1/255,
                                   rotation_range=20,
                                   zoom_range=0.2,
                                   shear_range=0.1)

validation_datagen = ImageDataGenerator(rescale=1/255)

train_generator = train_datagen.flow_from_directory(
    'datasetv2/training',
    target_size=(224, 224),
    batch_size=32,
    color_mode='grayscale',
    class_mode = 'categorical'
)

val_generator = validation_datagen.flow_from_directory(
    'datasetv2/testing',
    target_size=(224, 224),
    batch_size=32,
    color_mode='grayscale',
    class_mode = 'categorical'
)
```

```
Found 3120 images belonging to 20 classes.
Found 1299 images belonging to 20 classes.
```

Kode ini mengatur generator data untuk pelatihan dan validasi, yang mempermudah proses loading, preprocessing, dan augmentasi data gambar secara efisien selama pelatihan model.

```
[ ]  #@title Helper function for plotting Accuracy and Loss
     def plot(hist):
       history = hist.history
       history['epoch'] = hist.epoch

       plt.figure(figsize=(12, 5))

       plt.subplot(121)
       plt.plot(history['epoch'], history['loss'], label='Loss')
       plt.plot(history['epoch'], history['val_loss'], label='Val Loss', color='orange')
       plt.legend()

       plt.subplot(122)
       plt.plot(history['epoch'], history['accuracy'], label='Acc')
       plt.plot(history['epoch'], history['val_accuracy'], label='Val Acc', color='orange')
       plt.legend()

       return plt.show()
```

Fungsi ini kode ini adalah membuat dua subplot yang masing-masing menampilkan tren loss dan accuracy baik untuk data pelatihan maupun validasi selama epoch. Ini berguna untuk memvisualisasikan performa model dan melihat apakah ada overfitting atau underfitting.

```
#@title Helper function for testing model
classes = ['ba', 'ca', 'da', 'dha', 'ga', 'ha', 'ja', 'ka', 'la', 'ma',
           'na', 'nga', 'nya', 'pa', 'ra', 'sa', 'ta', 'tha', 'wa', 'ya']

def test(model, width):
  test_images_paths = os.listdir('datasetv2/prediction')
  for path in test_images_paths:
    image_path = os.path.join('datasetv2/prediction', path)

    image = keras_image.load_img(image_path,
                                 color_mode='grayscale',
                                 target_size=(width, width))
    x = keras_image.img_to_array(image)
    x = np.expand_dims(x, axis=0)

    test_image = np.vstack([x])
    result = model.predict(test_image, batch_size=8)

    print(image_path)
    print(classes[np.argmax(result)])

    preview = plt_image.imread(image_path)
    plt.imshow(preview)
    plt.show()
  return print('Prediction Done')
```

Fungsi test ini memuat gambar dari direktori prediksi, memprosesnya, melakukan prediksi menggunakan model yang sudah dilatih, dan menampilkan hasil prediksi serta gambar tersebut. Fungsi ini membantu dalam mengevaluasi performa model pada gambar baru.

C. Penggunaan Model Dalam Pengenalan Pola Aksara Jawa

1. Membuat Baseline Model CNN

```
base_model = Sequential([Conv2D(16, (3, 3), activation='relu', input_shape=(224, 224, 1)),
                         MaxPool2D(2, 2),
                         Conv2D(32, (3, 3), activation='relu'),
                         MaxPool2D(2, 2),
                         Conv2D(32, (3, 3), activation='relu'),
                         MaxPool2D(2, 2),
                         Conv2D(64, (3, 3), activation='relu'),
                         MaxPool2D(2, 2),
                         Conv2D(64, (3, 3), activation='relu'),
                         MaxPool2D(2, 2),
                         Flatten(),
                         Dropout(0.5),
                         Dense(128, activation='relu'),
                         Dense(20, activation='softmax')])

base_model.compile(
    loss='categorical_crossentropy',
    optimizer='Adam',
    metrics=['accuracy']
)

base_model.summary()
```

Kode program ini membangun model Convolutional Neural Network (CNN) menggunakan Keras di TensorFlow. Model ini terdiri dari beberapa lapisan konvolusi (Conv2D) dengan filter berukuran 3x3 dan fungsi aktivasi ReLU, diikuti oleh lapisan pooling (MaxPool2D) yang mengurangi dimensi data. Setelah beberapa lapisan konvolusi dan pooling, data diubah menjadi satu dimensi dengan lapisan Flatten. Lapisan Dropout dengan tingkat dropout 50% ditambahkan untuk mencegah overfitting. Kemudian, terdapat dua lapisan dense: satu dengan 128 neuron dan aktivasi ReLU, dan lapisan output dengan 20 neuron dan aktivasi softmax untuk klasifikasi multi-kelas. Model ini dikompilasi dengan fungsi loss categorical crossentropy, optimizer Adam, dan metrik akurasi. Terakhir, ringkasan arsitektur model ditampilkan menggunakan summary().

```
!mkdir '/content/gdrive/My Drive/logs2/'
LOGS = '/content/gdrive/My Drive/logs2/'
NAME = f'baseline_model-{time.time()}'
tensorboard = TensorBoard(log_dir=os.path.join(LOGS, NAME), histogram_freq=1)

%tensorboard --logdir='/content/gdrive/My Drive/logs2/'
```

Kode ini membuat direktori baru untuk menyimpan log TensorBoard di Google Drive, tepatnya di /content/gdrive/My Drive/logs2/. Variabel LOGS menyimpan path tersebut, dan NAME menyimpan nama unik untuk sesi pelatihan berdasarkan timestamp saat ini. Sebuah callback TensorBoard dibuat dengan mengarahkan log ke path yang telah ditentukan, dengan

parameter histogram_freq=1 untuk merekam histogram dari aktivasi dan bobot pada setiap epoch. Perintah %tensorboard --logdir='/content/gdrive/My Drive/logs2/' digunakan untuk memulai TensorBoard dan menampilkan log yang tersimpan di direktori tersebut, memungkinkan pengguna untuk memantau metrik pelatihan, seperti akurasi dan loss, secara visual dalam browser.

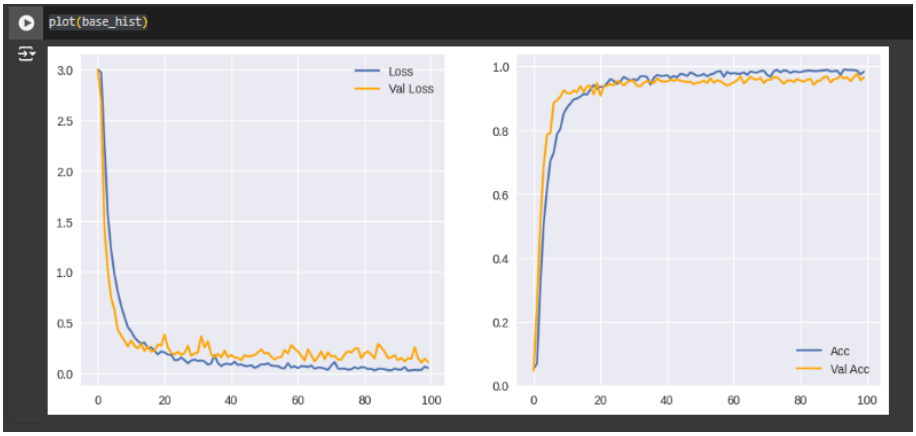
2. Training Model

```
base_hist = base_model.fit(
    train_generator,
    epochs = 100,
    steps_per_epoch = int(1762/32),
    validation_data = val_generator,
    validation_steps= int(979/32),
    callbacks=[tensorboard],
    verbose=1
)
```

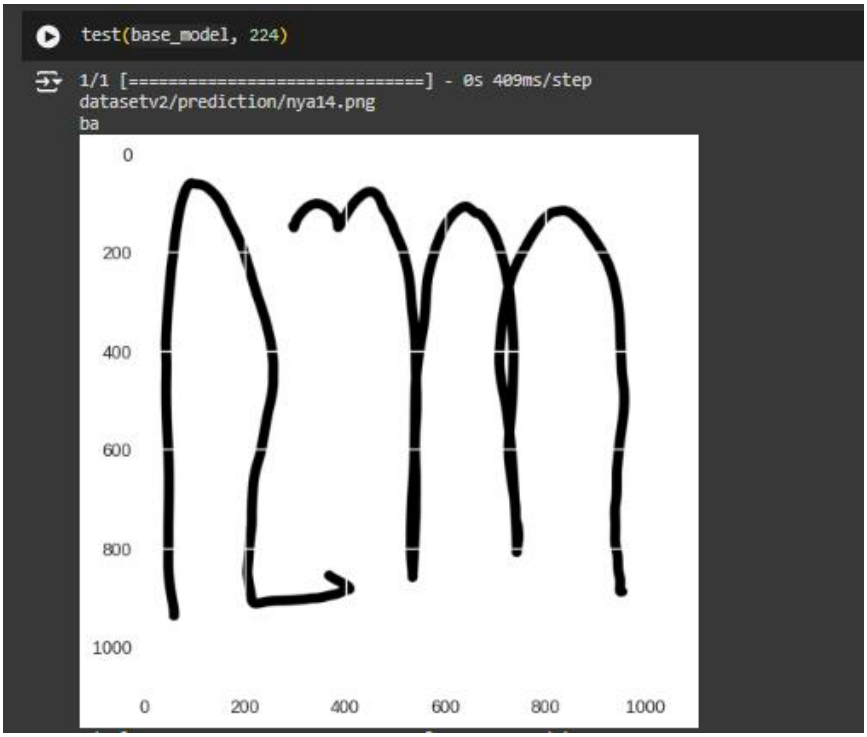
```
55/55 [==============================] - 96s 2s/step - loss: 0.0281 - accuracy: 0.9869 - val_loss: 0.1472 - val_accuracy: 0.9604
Epoch 85/100
55/55 [==============================] - 98s 2s/step - loss: 0.0447 - accuracy: 0.9852 - val_loss: 0.2918 - val_accuracy: 0.9427
Epoch 86/100
55/55 [==============================] - 95s 2s/step - loss: 0.0463 - accuracy: 0.9851 - val_loss: 0.2559 - val_accuracy: 0.9542
Epoch 87/100
55/55 [==============================] - 97s 2s/step - loss: 0.0416 - accuracy: 0.9869 - val_loss: 0.2122 - val_accuracy: 0.9563
Epoch 88/100
55/55 [==============================] - 94s 2s/step - loss: 0.0318 - accuracy: 0.9874 - val_loss: 0.1502 - val_accuracy: 0.9667
Epoch 89/100
55/55 [==============================] - 98s 2s/step - loss: 0.0295 - accuracy: 0.9892 - val_loss: 0.1586 - val_accuracy: 0.9677
Epoch 90/100
55/55 [==============================] - 96s 2s/step - loss: 0.0476 - accuracy: 0.9845 - val_loss: 0.1798 - val_accuracy: 0.9500
Epoch 91/100
55/55 [==============================] - 98s 2s/step - loss: 0.0384 - accuracy: 0.9851 - val_loss: 0.1305 - val_accuracy: 0.9594
Epoch 92/100
55/55 [==============================] - 97s 2s/step - loss: 0.0376 - accuracy: 0.9864 - val_loss: 0.1531 - val_accuracy: 0.9625
Epoch 93/100
55/55 [==============================] - 96s 2s/step - loss: 0.0614 - accuracy: 0.9733 - val_loss: 0.1186 - val_accuracy: 0.9698
Epoch 94/100
55/55 [==============================] - 97s 2s/step - loss: 0.0265 - accuracy: 0.9903 - val_loss: 0.1493 - val_accuracy: 0.9625
Epoch 95/100
55/55 [==============================] - 96s 2s/step - loss: 0.0304 - accuracy: 0.9891 - val_loss: 0.1443 - val_accuracy: 0.9656
Epoch 96/100
55/55 [==============================] - 96s 2s/step - loss: 0.0354 - accuracy: 0.9885 - val_loss: 0.2586 - val_accuracy: 0.9542
Epoch 97/100
55/55 [==============================] - 96s 2s/step - loss: 0.0326 - accuracy: 0.9886 - val_loss: 0.1501 - val_accuracy: 0.9667
Epoch 98/100
55/55 [==============================] - 96s 2s/step - loss: 0.0355 - accuracy: 0.9862 - val_loss: 0.1065 - val_accuracy: 0.9771
Epoch 99/100
55/55 [==============================] - 93s 2s/step - loss: 0.0651 - accuracy: 0.9756 - val_loss: 0.1469 - val_accuracy: 0.9563
Epoch 100/100
55/55 [==============================] - 93s 2s/step - loss: 0.0546 - accuracy: 0.9835 - val_loss: 0.1117 - val_accuracy: 0.9656
```

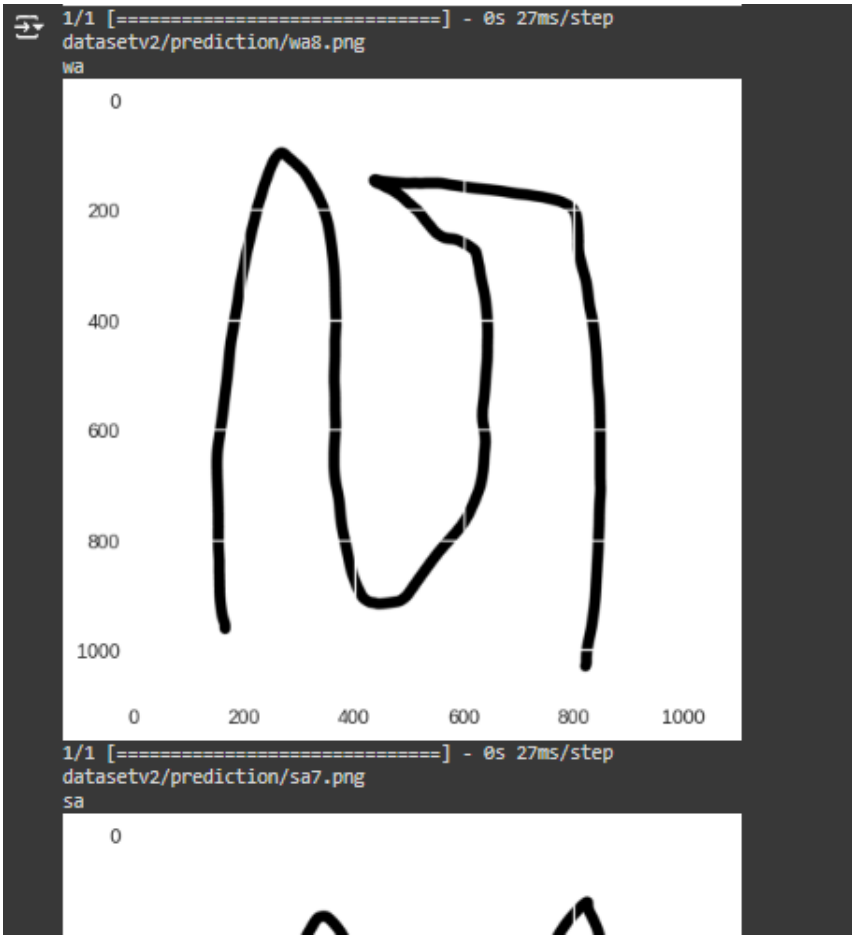
Kode ini melatih model `base_model` menggunakan data yang disediakan oleh `train_generator` selama 100 epoch. Setiap epoch terdiri dari jumlah langkah (`steps_per_epoch`) yang dihitung sebagai `int(1762/32)`, dengan 1762 menjadi jumlah total sampel dalam set pelatihan dan 32 sebagai ukuran batch. Validasi dilakukan menggunakan `val_generator`, dengan `validation_steps` dihitung sebagai `int(979/32)`, di mana 979 adalah jumlah total sampel dalam set validasi. Callback TensorBoard (`tensorboard`) digunakan untuk mencatat log pelatihan, sehingga memungkinkan pengguna memantau metrik pelatihan

secara visual. Parameter `verbose=1` memastikan bahwa kemajuan pelatihan ditampilkan di konsol. Hasil pelatihan disimpan dalam variabel `base_hist`, yang mencakup informasi seperti loss dan akurasi untuk setiap epoch.



3. Testing Model





Hasil setelah testing model akan muncul prediksi huruf aksara jawa yang berjumlah 20.

4. Menyimpan Baseine Model

```
!mkdir '/content/gdrive/My Drive/model/baseline'
MODEL_PATH = '/content/gdrive/My Drive/model/baseline'
tf.saved_model.save(base_model, os.path.join(MODEL_PATH, 'saved_model'))
```

```
mkdir: cannot create directory ‘/content/gdrive/My Drive/model/baseline’: File exists
```

```
tf.keras.models.save_model(base_model, os.path.join(MODEL_PATH, 'base_model.h5'))
```

Proses ini merupakan sebuah proses untuk menympan model yang telah dibuat. File model ini akan tersimpan dengan format .h5.

Sumber dataset : https://www.kaggle.com/datasets/phiard/aksara-jawa?resource=download-directory

Link google colaboratory : https://colab.research.google.com/drive/1l2pIMFG5tOfg2bL3pJDXj7WxuXCyScHW

Link github : https://github.com/fakhrulreza/Pengenalan-Pola-Aksara-Jawa