NAMA: DANANG HILAL KURNIAWAN

- *EMAIL: danang.122450085@student.itera.ac.id*
- *Alamat: Kelurahan Way Huwi, Jati Agung , KAB. LAMPUNG SELATAN, JATI AGUNG, LAMPUNG*

1. Mengimport dataset yang sudah ditentukan yaitu rockpaperscissor

```
!wget https://github.com/dicodingacademy/assets/releases/download/release/rockpaperscissors.zip
!unzip -q rockpaperscissors.zip
```

```
--2024-04-21 01:56:40--  https://github.com/dicodingacademy/assets/releases/download/release/rockpaperscissors.zip
Resolving github.com (github.com)... 140.82.116.4
Connecting to github.com (github.com)|140.82.116.4|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/391417272/7eb836f2-695b-4a46-9c78
--2024-04-21 01:56:40--  https://objects.githubusercontent.com/github-production-release-asset-2e65be/391417272/7eb836f2
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.109.133, 185.199.111.133, 185.199.110
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.109.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 322873683 (308M) [application/octet-stream]
Saving to: ‘rockpaperscissors.zip’

rockpaperscissors.z 100%[===================>] 307.92M   224MB/s    in 1.4s

2024-04-21 01:56:42 (224 MB/s) - ‘rockpaperscissors.zip’ saved [322873683/322873683]
```

2. Augmentasi gambar dan data generator

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator

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,
    fill_mode='nearest',
    validation_split=0.4
)

train_dir = '/content/rockpaperscissors/rps-cv-images/'

train_generator = train_datagen.flow_from_directory(
    train_dir,
    target_size=(150, 150),
    batch_size=32,
    class_mode='categorical',
    subset='training'
)

validation_generator = train_datagen.flow_from_directory(
    train_dir,
    target_size=(150, 150),
    batch_size=32,
    class_mode='categorical',
    subset='validation'
)
```

```
Found 1314 images belonging to 3 classes.
Found 874 images belonging to 3 classes.
```

```
from sklearn.model_selection import train_test_split
import os

base_dir = '/content/rockpaperscissors/rps-cv-images'

rock_dir = os.path.join(base_dir, 'rock')
paper_dir = os.path.join(base_dir, 'paper')
scissors_dir = os.path.join(base_dir, 'scissors')

rock_files = [os.path.join(rock_dir, file) for file in os.listdir(rock_dir)]
paper_files = [os.path.join(paper_dir, file) for file in os.listdir(paper_dir)]
scissors_files = [os.path.join(scissors_dir, file) for file in os.listdir(scissors_dir)]

rock_train, rock_val = train_test_split(rock_files, test_size=0.4, random_state=42)
paper_train, paper_val = train_test_split(paper_files, test_size=0.4, random_state=42)
scissors_train, scissors_val = train_test_split(scissors_files, test_size=0.4, random_state=42)

num_rock_train = len(rock_train)
num_paper_train = len(paper_train)
num_scissors_train = len(scissors_train)

num_rock_val = len(rock_val)
num_paper_val = len(paper_val)
num_scissors_val = len(scissors_val)

print("Jumlah sampel dalam setiap kelas pada training set:")
print("Rock (train):", num_rock_train, "sampel")
print("Paper (train):", num_paper_train, "sampel")
print("Scissors (train):", num_scissors_train, "sampel")
print()
print("Jumlah sampel dalam setiap kelas pada validation set:")
print("Rock (validation):", num_rock_val, "sampel")
print("Paper (validation):", num_paper_val, "sampel")
print("Scissors (validation):", num_scissors_val, "sampel")
```

```
Jumlah sampel dalam setiap kelas pada training set:
Rock (train): 435 sampel
Paper (train): 427 sampel
Scissors (train): 450 sampel

Jumlah sampel dalam setiap kelas pada validation set:
Rock (validation): 291 sampel
Paper (validation): 285 sampel
Scissors (validation): 300 sampel
```

3. Model Sequential

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense

model = Sequential()

model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(150, 150, 3)))
model.add(MaxPooling2D(2, 2))

model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(2, 2))

model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D(2, 2))

model.add(Flatten())

model.add(Dense(512, activation='relu'))

model.add(Dense(3, activation='softmax'))

model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_3 (Conv2D)           (None, 148, 148, 32)      896

 max_pooling2d_3 (MaxPoolin  (None, 74, 74, 32)        0
 g2D)

 conv2d_4 (Conv2D)           (None, 72, 72, 64)        18496
```

```
 max_pooling2d_4 (MaxPoolin  (None, 36, 36, 64)        0
 g2D)

 conv2d_5 (Conv2D)           (None, 34, 34, 128)       73856

 max_pooling2d_5 (MaxPoolin  (None, 17, 17, 128)       0
 g2D)

 flatten_1 (Flatten)         (None, 36992)             0

 dense_2 (Dense)             (None, 512)               18940416

 dense_3 (Dense)             (None, 3)                 1539

=================================================================
Total params: 19035203 (72.61 MB)
Trainable params: 19035203 (72.61 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

4. Model pelatihan tidak lebih dari 30 menit

```
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint
import time

early_stopping = EarlyStopping(monitor='val_loss', patience=5, verbose=1, restore_best_weights=True)

checkpoint_path = "best_model.h5"
model_checkpoint = ModelCheckpoint(checkpoint_path, monitor='val_loss', save_best_only=True, verbose=1)
start_time = time.time()

history = model.fit(
    train_generator,
    epochs=100,
    validation_data=val_generator,
    callbacks=[early_stopping, model_checkpoint]
)

end_time = time.time()
training_time = end_time - start_time
val_loss, val_accuracy = model.evaluate(val_generator)

print("Total waktu pelatihan:", training_time, "detik")
print("Akurasi model pada set data validasi:", val_accuracy)
```

```
Epoch 1/100
42/42 [==============================] - ETA: 0s - loss: 1.1339 - accuracy: 0.3980
Epoch 1: val_loss improved from inf to 0.92816, saving model to best_model.h5
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3103: UserWarning: You are saving your model as an
  saving_api.save_model(
42/42 [==============================] - 22s 437ms/step - loss: 1.1339 - accuracy: 0.3980 - val_loss: 0.9282 - val_accur
Epoch 2/100
42/42 [==============================] - ETA: 0s - loss: 1.0049 - accuracy: 0.5053
Epoch 2: val_loss improved from 0.92816 to 0.71696, saving model to best_model.h5
42/42 [==============================] - 17s 396ms/step - loss: 1.0049 - accuracy: 0.5053 - val_loss: 0.7170 - val_accur
Epoch 3/100
42/42 [==============================] - ETA: 0s - loss: 0.8014 - accuracy: 0.6606
Epoch 3: val_loss improved from 0.71696 to 0.46642, saving model to best_model.h5
42/42 [==============================] - 18s 427ms/step - loss: 0.8014 - accuracy: 0.6606 - val_loss: 0.4664 - val_accur
Epoch 4/100
42/42 [==============================] - ETA: 0s - loss: 0.5371 - accuracy: 0.8097
Epoch 4: val_loss improved from 0.46642 to 0.23461, saving model to best_model.h5
42/42 [==============================] - 18s 433ms/step - loss: 0.5371 - accuracy: 0.8097 - val_loss: 0.2346 - val_accur
Epoch 5/100
42/42 [==============================] - ETA: 0s - loss: 0.3541 - accuracy: 0.8760
Epoch 5: val_loss improved from 0.23461 to 0.10016, saving model to best_model.h5
42/42 [==============================] - 16s 395ms/step - loss: 0.3541 - accuracy: 0.8760 - val_loss: 0.1002 - val_accur
Epoch 6/100
42/42 [==============================] - ETA: 0s - loss: 0.3532 - accuracy: 0.8729
Epoch 6: val_loss improved from 0.10016 to 0.09464, saving model to best_model.h5
42/42 [==============================] - 17s 417ms/step - loss: 0.3532 - accuracy: 0.8729 - val_loss: 0.0946 - val_accur
Epoch 7/100
42/42 [==============================] - ETA: 0s - loss: 0.2730 - accuracy: 0.9018
Epoch 7: val_loss did not improve from 0.09464
42/42 [==============================] - 16s 375ms/step - loss: 0.2730 - accuracy: 0.9018 - val_loss: 0.1023 - val_accur
Epoch 8/100
42/42 [==============================] - ETA: 0s - loss: 0.2381 - accuracy: 0.9193
Epoch 8: val_loss improved from 0.09464 to 0.07296, saving model to best_model.h5
42/42 [==============================] - 17s 415ms/step - loss: 0.2381 - accuracy: 0.9193 - val_loss: 0.0730 - val_accur
Epoch 9/100
42/42 [==============================] - ETA: 0s - loss: 0.2156 - accuracy: 0.9292
Epoch 9: val_loss did not improve from 0.07296
42/42 [==============================] - 17s 408ms/step - loss: 0.2156 - accuracy: 0.9292 - val_loss: 0.1244 - val_accur
Epoch 10/100
42/42 [==============================] - ETA: 0s - loss: 0.1684 - accuracy: 0.9513
```

```
Epoch 10: val_loss improved from 0.07296 to 0.06334, saving model to best_model.h5
42/42 [==============================] - 22s 529ms/step - loss: 0.1684 - accuracy: 0.9513 - val_loss: 0.0633 - val_accur
Epoch 11/100
42/42 [==============================] - ETA: 0s - loss: 0.1862 - accuracy: 0.9498
Epoch 11: val_loss did not improve from 0.06334
42/42 [==============================] - 17s 418ms/step - loss: 0.1862 - accuracy: 0.9498 - val_loss: 0.0698 - val_accur
Epoch 12/100
42/42 [==============================] - ETA: 0s - loss: 0.1830 - accuracy: 0.9391
Epoch 12: val_loss improved from 0.06334 to 0.05650, saving model to best_model.h5
42/42 [==============================] - 22s 525ms/step - loss: 0.1830 - accuracy: 0.9391 - val_loss: 0.0565 - val_accur
Epoch 13/100
42/42 [==============================] - ETA: 0s - loss: 0.1432 - accuracy: 0.9490
Epoch 13: val_loss improved from 0.05650 to 0.04629, saving model to best_model.h5
42/42 [==============================] - 17s 405ms/step - loss: 0.1432 - accuracy: 0.9490 - val_loss: 0.0463 - val_accur
Epoch 14/100
42/42 [==============================] - ETA: 0s - loss: 0.1589 - accuracy: 0.9467
Epoch 14: val_loss improved from 0.04629 to 0.04302, saving model to best_model.h5
```

5. Mengecek akurasi model

```
val_datagen = ImageDataGenerator(rescale=1./255)
val_dir = '/content/rockpaperscissors/rps-cv-images/'

val_generator = val_datagen.flow_from_directory(
    val_dir,
    target_size=(150, 150),
    batch_size=32,
    class_mode='categorical'
)

val_loss, val_accuracy = model.evaluate(val_generator)

print("Akurasi model pada set data validasi:", val_accuracy)
```

```
Found 2188 images belonging to 3 classes.
69/69 [==============================] - 6s 84ms/step - loss: 0.0430 - accuracy: 0.9881
Akurasi model pada set data validasi: 0.988116979598999
```

6. Prediksi gambar

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

uploaded = files.upload()

for name in uploaded.keys():
    img = image.load_img(name, target_size=(150,150))
    image_plot = plt.imshow(img)
    image_arr = image.img_to_array(img)
    image_arr = np.expand_dims(image_arr, axis=0)

    images = np.vstack([image_arr])
    pred = model.predict(images, batch_size=10)

    print(name)
    if pred[0][0] == 1:
        print("scissors")
    elif pred[0][1] == 1:
        print("paper")
    else:
        print("rock")
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving Saving scissors.png to Saving scissors.png
1/1 [==============================] - 0s 20ms/step
Saving scissors.png
scissors
```