```
#20180805014 Ahmet Aydeniz
"""Untitled1.ipynb

Automatically generated by Colaboratory.

Original file is located at

https://www.kaggle.com/divyansh22/sheep-breed-classification
"""
import os


import shutil
from keras import layers
from keras import models
import matplotlib.pyplot as plt
import scipy
import numpy as np
import pandas
import matplotlib.pyplot as plt
import tensorflow as tf
tf.keras.utils.get_custom_objects()
from keras.optimizers import Adam

from tensorflow import keras
from keras.models import load_model
from keras import optimizers
from tensorflow.keras.optimizers import RMSprop



original_Marino_dir
='C:\\Users\\Aydeniz\\Desktop\deeplearning\\SheepFaceImages\\Marino'
original_PollDorset_dir
='C:\\Users\\Aydeniz\\Desktop\\deeplearning\\SheepFaceImages\\PollDorset'
original_Suffolk_dir
='C:\\Users\\Aydeniz\\Desktop\\deeplearning\\SheepFaceImages\\Suffolk'
original_WhiteSuffolk_dir
='C:\\Users\\Aydeniz\\Desktop\deeplearning\\SheepFaceImages\\WhiteSuffolk'

base_dir=('C:\\Users\Aydeniz\Desktop\deepout')

if os.path.exists(base_dir):
 shutil.rmtree(base_dir)

os.mkdir(base_dir)
train_dir = os.path.join(base_dir, 'train')
os.mkdir(train_dir)
validation_dir = os.path.join(base_dir, 'validation')
```

```
os.mkdir(validation_dir)
test_dir = os.path.join(base_dir, 'test')
os.mkdir(test_dir)
# Yeni Bölüm"""

train_Marino_dir = os.path.join(train_dir, 'Marino')
os.mkdir(train_Marino_dir)


train_PollDorset_dir = os.path.join(train_dir, 'PollDorset')
os.mkdir(train_PollDorset_dir)

train_Suffolk_dir = os.path.join(train_dir, 'Suffolk')
os.mkdir(train_Suffolk_dir)

train_WhiteSuffolk_dir = os.path.join(train_dir, 'WhiteSuffolk')
os.mkdir(train_WhiteSuffolk_dir)

validation_Marino_dir = os.path.join(validation_dir, 'Marino')
os.mkdir(validation_Marino_dir)

validation_PollDorset_dir = os.path.join(validation_dir, 'PollDorset')
os.mkdir(validation_PollDorset_dir)


validation_Suffolk_dir = os.path.join(validation_dir, 'Suffolk')
os.mkdir(validation_Suffolk_dir)

validation_WhiteSuffolk_dir = os.path.join(validation_dir, 'WhiteSuffolk')
os.mkdir(validation_WhiteSuffolk_dir)

test_Marino_dir = os.path.join(test_dir, 'Marino')
os.mkdir(test_Marino_dir)

test_PollDorset_dir = os.path.join(test_dir, 'PollDorset')
os.mkdir(test_PollDorset_dir)

test_Suffolk_dir = os.path.join(test_dir, 'Suffolk')
os.mkdir(test_Suffolk_dir)

test_WhiteSuffolk_dir = os.path.join(test_dir, 'WhiteSuffolk')
os.mkdir(test_WhiteSuffolk_dir)





### 1.KISIM - B - Resimleri oluşturduğum training, validation ve test
klasörlerine gönderme
```

```python
for i in range(1,301):
 source =
"C:\\Users\\Aydeniz\\Desktop\\deeplearning\\SheepFaceImages\\Marino\\"+str(i)+
".jpg"
 destination =
"C:\\Users\\Aydeniz\\Desktop\\deepout\\train\\Marino\\"+str(i)+".jpg"

 shutil.copyfile(source, destination)

for i in range(301,361):
 source =
"C:\\Users\\Aydeniz\\Desktop\\deeplearning\\SheepFaceImages\\Marino\\"+str(i)+
".jpg"
 destination =
"C:\\Users\\Aydeniz\\Desktop\\deepout\\validation\\Marino\\"+str(i)+".jpg"

 shutil.copyfile(source, destination)


for i in range(361,421):
 source =
"C:\\Users\\Aydeniz\\Desktop\\deeplearning\\SheepFaceImages\\Marino\\"+str(i)+
".jpg"
 destination =
"C:\\Users\\Aydeniz\\Desktop\\deepout\\test\\Marino\\"+str(i)+".jpg"

 shutil.copyfile(source, destination)

#-----------------------------------------------------------------------------
-

for i in range(1,301):
 source =
"C:\\Users\\Aydeniz\\Desktop\\deeplearning\\SheepFaceImages\\PollDorset\\"+str
(i)+".jpg"
 destination =
"C:\\Users\\Aydeniz\\Desktop\\deepout\\train\\PollDorset\\"+str(i)+".jpg"

 shutil.copyfile(source, destination)

for i in range(301,361):
 source =
"C:\\Users\\Aydeniz\\Desktop\\deeplearning\\SheepFaceImages\\PollDorset\\"+str
(i)+".jpg"
 destination =
"C:\\Users\\Aydeniz\\Desktop\\deepout\\validation\\PollDorset\\"+str(i)+".jpg"
```

```
 shutil.copyfile(source, destination)


for i in range(361,421):
 source =
"C:\\Users\\Aydeniz\\Desktop\\deeplearning\\SheepFaceImages\\PollDorset\\"+str
(i)+".jpg"
 destination =
"C:\\Users\\Aydeniz\\Desktop\\deepout\\test\\PollDorset\\"+str(i)+".jpg"

 shutil.copyfile(source, destination)

#-----------------------------------------------------------------------------
-

for i in range(1,301):
 source =
"C:\\Users\\Aydeniz\\Desktop\\deeplearning\\SheepFaceImages\\Suffolk\\"+str(i)
+".jpg"
 destination =
"C:\\Users\\Aydeniz\\Desktop\\deepout\\train\\Suffolk\\"+str(i)+".jpg"

 shutil.copyfile(source, destination)

for i in range(301,361):
 source =
"C:\\Users\\Aydeniz\\Desktop\\deeplearning\\SheepFaceImages\\Suffolk\\"+str(i)
+".jpg"
 destination =
"C:\\Users\\Aydeniz\\Desktop\\deepout\\validation\\Suffolk\\"+str(i)+".jpg"

 shutil.copyfile(source, destination)


for i in range(361,421):
 source =
"C:\\Users\\Aydeniz\\Desktop\\deeplearning\\SheepFaceImages\\Suffolk\\"+str(i)
+".jpg"
 destination =
"C:\\Users\\Aydeniz\\Desktop\\deepout\\test\\Suffolk\\"+str(i)+".jpg"

 shutil.copyfile(source, destination)

#-----------------------------------------------------------------------------
-
for i in range(1,301):
 source =
"C:\\Users\\Aydeniz\\Desktop\\deeplearning\\SheepFaceImages\\WhiteSuffolk\\"+s
tr(i)+".jpg"
```

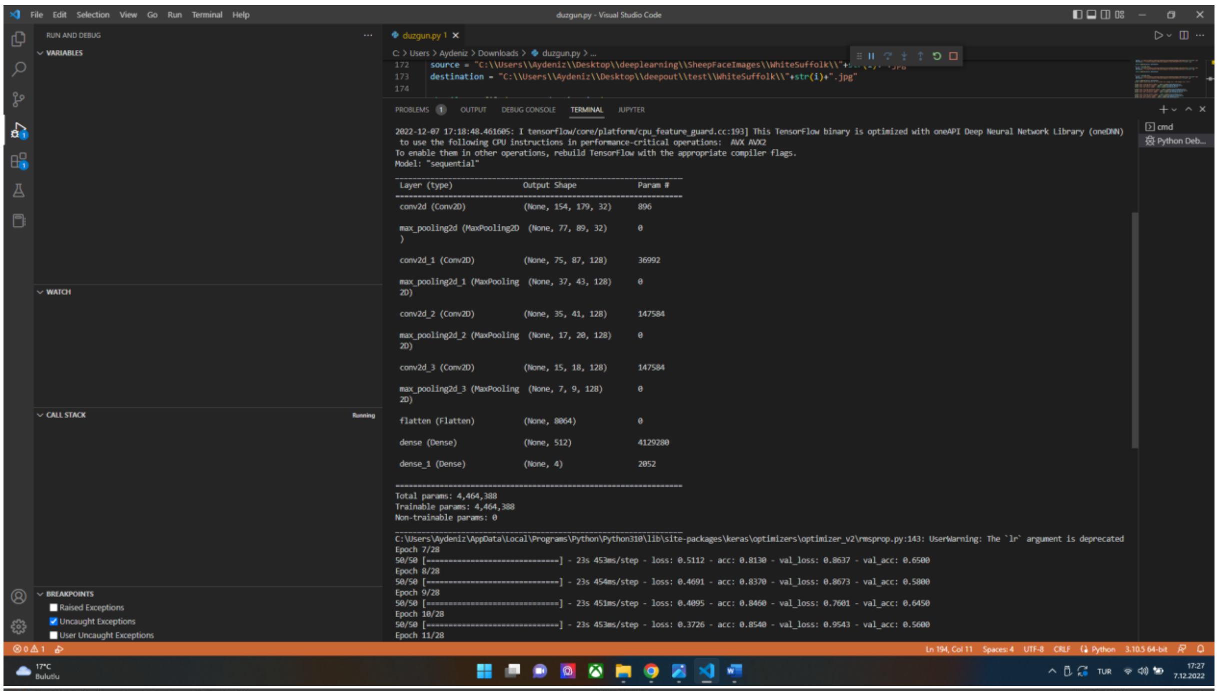
```
 destination =
"C:\\Users\\Aydeniz\\Desktop\\deepout\\train\\WhiteSuffolk\\"+str(i)+".jpg"

 shutil.copyfile(source, destination)

for i in range(301,361):
 source =
"C:\\Users\\Aydeniz\\Desktop\\deeplearning\\SheepFaceImages\\WhiteSuffolk\\"+s
tr(i)+".jpg"
 destination =
"C:\\Users\\Aydeniz\\Desktop\\deepout\\validation\\WhiteSuffolk\\"+str(i)+".jp
g"

 shutil.copyfile(source, destination)


for i in range(361,421):
 source =
"C:\\Users\\Aydeniz\\Desktop\\deeplearning\\SheepFaceImages\\WhiteSuffolk\\"+s
tr(i)+".jpg"
 destination =
"C:\\Users\\Aydeniz\\Desktop\\deepout\\test\\WhiteSuffolk\\"+str(i)+".jpg"

 shutil.copyfile(source, destination)
print('common sense baseline tum 4 classtaki veri sayısı esit = 420 oldugu
icin = 1/4')

print('total training Marino images:', len(os.listdir(train_Marino_dir)))
print('total valid Marino images:', len(os.listdir(validation_Marino_dir)))
print('total test Marino images:', len(os.listdir(test_Marino_dir)))

print('total training PollDorset images:',
len(os.listdir(train_PollDorset_dir)))
print('total valid PollDorset images:',
len(os.listdir(validation_PollDorset_dir)))
print('total test PollDorset images:', len(os.listdir(test_PollDorset_dir)))

print('total training Suffolk images:', len(os.listdir(train_Suffolk_dir)))
print('total valid Suffolk images:', len(os.listdir(validation_Suffolk_dir)))
print('total test Suffolk images:', len(os.listdir(test_Suffolk_dir)))

print('total training WhiteSuffolk images:',
len(os.listdir(train_WhiteSuffolk_dir)))
print('total valid WhiteSuffolk images:',
len(os.listdir(validation_WhiteSuffolk_dir)))
print('total test WhiteSuffolk images:',
len(os.listdir(test_WhiteSuffolk_dir)))

import cv2
```

```
from keras import layers
from keras import models

model = models.Sequential()

model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(156, 181,
3)))

model.add(layers.MaxPooling2D((2,2)))
model.add(layers.Conv2D(128, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2,2)))
model.add(layers.Conv2D(128, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2,2)))
model.add(layers.Conv2D(128, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2,2)))
model.add(layers.Flatten())
model.add(layers.Dense(512, activation='relu'))
model.add(layers.Dense(4, activation='softmax'))

model.summary()

model.compile(loss='categorical_crossentropy',
            optimizer = optimizers.RMSprop(lr=1e-4),
            metrics =['acc'])


from keras.preprocessing.image import ImageDataGenerator

train_datagen = ImageDataGenerator(rescale=1./255)
test_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
train_dir,
target_size=(156, 181),
batch_size=20,
class_mode='categorical')

validation_generator=test_datagen.flow_from_directory(
validation_dir,
target_size=(156, 181),
batch_size=20,
class_mode='categorical')

for data_batch, labels_batch in train_generator:
    print('data batch shape:', data_batch.shape)
    print('labels batc  h shape:', labels_batch.shape)
    break
```

```
model.compile(loss = "categorical_crossentropy",
              optimizer = RMSprop(learning_rate
              =1e-4),
              metrics = ["acc"])
for data_batch, labels_batch in train_generator:
    print('data batch shape:', data_batch.shape)
    print('labels batch shape:', labels_batch.shape)
    break
for data_batch, labels_batch in validation_generator:
    print('data batch shape:', data_batch.shape)
    print('labels batch shape:', labels_batch.shape)
    break



history = model.fit(
    train_generator,
    steps_per_epoch=50,
    validation_data=validation_generator,
    validation_steps=10,
    epochs=28)

model.save('koyunlar')

import matplotlib.pyplot as plt

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

epochs = range(1, len(acc) + 1)

plt.plot(epochs, acc, 'bo', label='Training acc')
plt.plot(epochs, val_acc, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.legend()

plt.figure()

plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.legend()

plt.show()
```

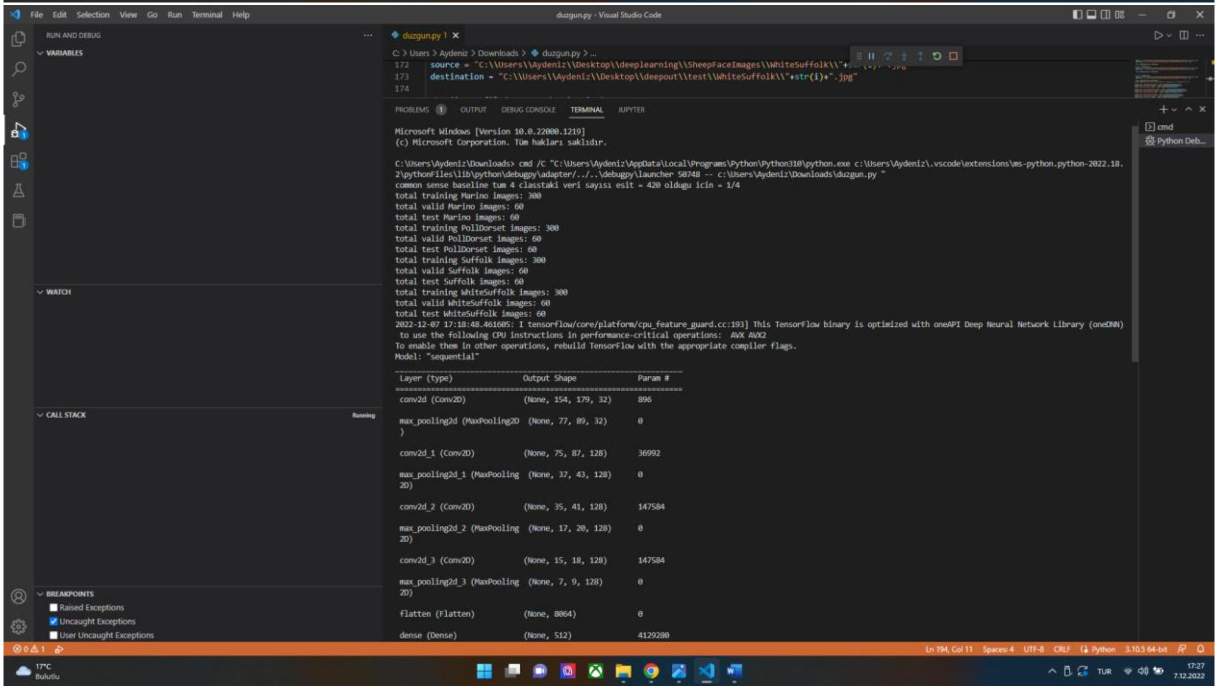
```
datagen = ImageDataGenerator(
          rotation_range=40,
          width_shift_range=0.2,
          height_shift_range=0.2,
          shear_range=0.2,
          zoom_range=0.2,
          horizontal_flip=True,
          fill_mode='nearest')

from keras import layers
from keras import models

model = models.Sequential()
model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(150, 150,
3)))

model.add(layers.MaxPooling2D((2,2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2,2)))
model.add(layers.Conv2D(128, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2,2)))
model.add(layers.Conv2D(128, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2,2)))
model.add(layers.Flatten())
model.add(layers.Dropout(0.5))
model.add(layers.Dense(512, activation='relu'))
model.add(layers.Dense(4, activation='softmax'))

from keras import optimizers


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)

test_datagen = ImageDataGenerator(rescale=1./255)

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

```
validation_generator=test_datagen.flow_from_directory(
validation_dir,
target_size=(150, 150),
batch_size=32,
class_mode='categorical')

acc = history.history['acc']
val_acc = history.history['val_acc']
loss = history.history['loss']
val_loss = history.history['val_loss']
history = model.fit(
train_generator,steps_per_epoch=100,
epochs=28,
validation_data=validation_generator,
validation_steps=50)


epochs = range(1, len(acc) + 1)

plt.plot(epochs, acc, 'bo', label='Training acc')
plt.plot(epochs, val_acc, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.legend()

plt.figure()

plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.legend()

plt.show()

model.save('koyunlar')

test_generator=test_datagen.flow_from_directory(
test_dir,
target_size=(150, 150),
batch_size=20,
class_mode='categorical')

test_loss, test_acc = model.evaluate(test_generator, steps=18)
print('test acc:', test_acc)
```



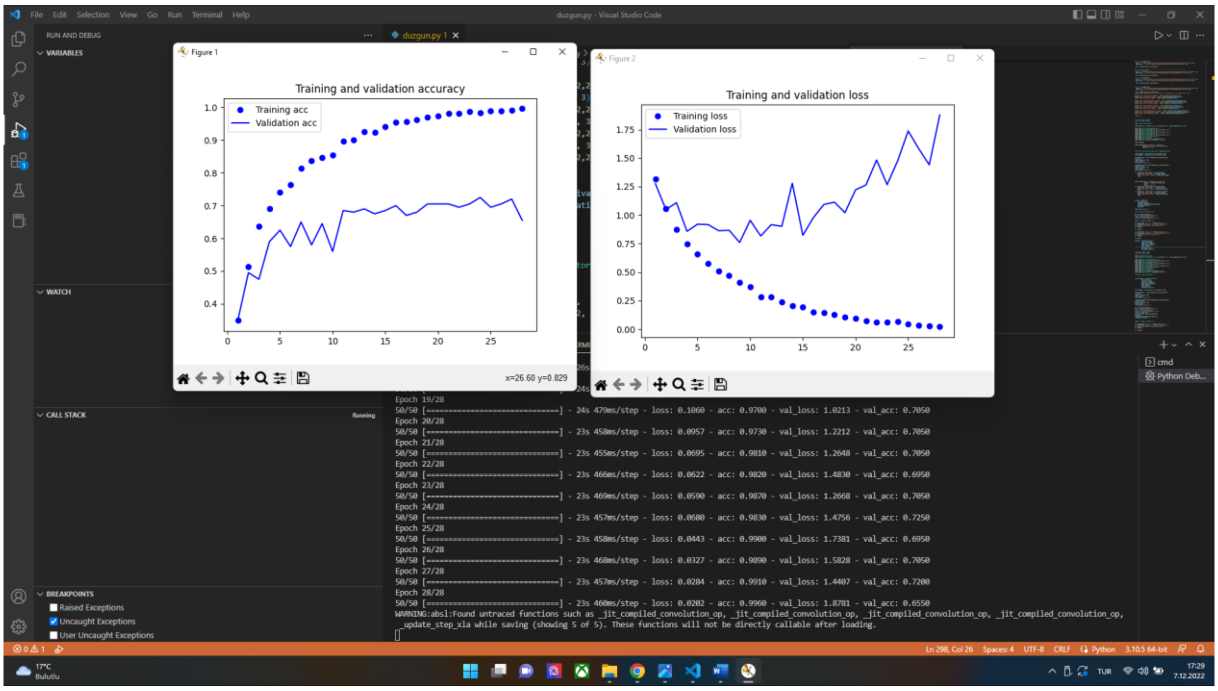
```
source = "C:\\Users\\Aydeniz\\Desktop\\deeplearning\\SheepFaceImages\\WhiteSuffolk\\"+
destination = "C:\\Users\\Aydeniz\\Desktop\\deepout\\test\\WhiteSuffolk\\"+str(i)+".jpg"
```

```
2022-12-07 17:18:48.461605: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN)
 to use the following CPU instructions in performance-critical operations:  AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 154, 179, 32)      896

 max_pooling2d (MaxPooling2D  (None, 77, 89, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 75, 87, 128)       36992

 max_pooling2d_1 (MaxPooling  (None, 37, 43, 128)      0
 2D)

 conv2d_2 (Conv2D)           (None, 35, 41, 128)       147584

 max_pooling2d_2 (MaxPooling  (None, 17, 20, 128)      0
 2D)

 conv2d_3 (Conv2D)           (None, 15, 18, 128)       147584

 max_pooling2d_3 (MaxPooling  (None, 7, 9, 128)        0
 2D)

 flatten (Flatten)           (None, 8064)              0

 dense (Dense)               (None, 512)               4129280

 dense_1 (Dense)             (None, 4)                 2052

=================================================================
Total params: 4,464,388
Trainable params: 4,464,388
Non-trainable params: 0
_________________________________________________________________
C:\Users\Aydeniz\AppData\Local\Programs\Python\Python310\lib\site-packages\keras\optimizers\optimizer_v2\rmsprop.py:143: UserWarning: The `lr` argument is deprecated
Epoch 7/28
50/50 [==============================] - 23s 453ms/step - loss: 0.5112 - acc: 0.8130 - val_loss: 0.8637 - val_acc: 0.6500
Epoch 8/28
50/50 [==============================] - 23s 454ms/step - loss: 0.4691 - acc: 0.8370 - val_loss: 0.8673 - val_acc: 0.5800
Epoch 9/28
50/50 [==============================] - 23s 451ms/step - loss: 0.4095 - acc: 0.8460 - val_loss: 0.7601 - val_acc: 0.6450
Epoch 10/28
50/50 [==============================] - 23s 453ms/step - loss: 0.3726 - acc: 0.8540 - val_loss: 0.9543 - val_acc: 0.5600
Epoch 11/28
```



```
source = "C:\\Users\\Aydeniz\\Desktop\\deeplearning\\SheepFaceImages\\WhiteSuffolk\\"+
destination = "C:\\Users\\Aydeniz\\Desktop\\deepout\\test\\WhiteSuffolk\\"+str(i)+".jpg"
```

```
Microsoft Windows [Version 10.0.22000.1219]
(c) Microsoft Corporation. Tüm hakları saklıdır.

C:\Users\Aydeniz\Downloads> cmd /C "C:\Users\Aydeniz\AppData\Local\Programs\Python\Python310\python.exe c:\Users\Aydeniz\.vscode\extensions\ms-python.python-2022.18.
2\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher 50748 -- c:\Users\Aydeniz\Downloads\duzgun.py "
common sense baseline tum 4 classtaki veri sayısı esit = 420 oldugu icin = 1/4
total training Marino images: 300
total valid Marino images: 60
total test Marino images: 60
total training PollDorset images: 300
total valid PollDorset images: 60
total test PollDorset images: 60
total training Suffolk images: 300
total valid Suffolk images: 60
total test Suffolk images: 60
total training WhiteSuffolk images: 300
total valid WhiteSuffolk images: 60
total test WhiteSuffolk images: 60
2022-12-07 17:18:48.461605: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN)
 to use the following CPU instructions in performance-critical operations:  AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 154, 179, 32)      896

 max_pooling2d (MaxPooling2D  (None, 77, 89, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 75, 87, 128)       36992

 max_pooling2d_1 (MaxPooling  (None, 37, 43, 128)      0
 2D)

 conv2d_2 (Conv2D)           (None, 35, 41, 128)       147584

 max_pooling2d_2 (MaxPooling  (None, 17, 20, 128)      0
 2D)

 conv2d_3 (Conv2D)           (None, 15, 18, 128)       147584

 max_pooling2d_3 (MaxPooling  (None, 7, 9, 128)        0
 2D)

 flatten (Flatten)           (None, 8064)              0

 dense (Dense)               (None, 512)               4129280
```



```
Epoch 19/28
50/50 [==============================] - 24s 479ms/step - loss: 0.1060 - acc: 0.9700 - val_loss: 1.0213 - val_acc: 0.7050
Epoch 20/28
50/50 [==============================] - 23s 458ms/step - loss: 0.0957 - acc: 0.9730 - val_loss: 1.2212 - val_acc: 0.7050
Epoch 21/28
50/50 [==============================] - 23s 455ms/step - loss: 0.0695 - acc: 0.9810 - val_loss: 1.2648 - val_acc: 0.7050
Epoch 22/28
50/50 [==============================] - 23s 466ms/step - loss: 0.0622 - acc: 0.9820 - val_loss: 1.4830 - val_acc: 0.6950
Epoch 23/28
50/50 [==============================] - 23s 469ms/step - loss: 0.0590 - acc: 0.9870 - val_loss: 1.2668 - val_acc: 0.7050
Epoch 24/28
50/50 [==============================] - 23s 457ms/step - loss: 0.0680 - acc: 0.9830 - val_loss: 1.4756 - val_acc: 0.7250
Epoch 25/28
50/50 [==============================] - 23s 458ms/step - loss: 0.0443 - acc: 0.9900 - val_loss: 1.7381 - val_acc: 0.6950
Epoch 26/28
50/50 [==============================] - 23s 468ms/step - loss: 0.0327 - acc: 0.9890 - val_loss: 1.5828 - val_acc: 0.7050
Epoch 27/28
50/50 [==============================] - 23s 457ms/step - loss: 0.0284 - acc: 0.9910 - val_loss: 1.4407 - val_acc: 0.7200
Epoch 28/28
50/50 [==============================] - 23s 460ms/step - loss: 0.0202 - acc: 0.9960 - val_loss: 1.8781 - val_acc: 0.6550
WARNING:absl:Found untraced functions such as _jit_compiled_convolution_op, _jit_compiled_convolution_op, _jit_compiled_convolution_op, _jit_compiled_convolution_op,
 _update_step_xla while saving (showing 5 of 5). These functions will not be directly callable after loading.
```